Pushing the Limits of ChatGPT on NLP Tasks

Anonymous ACL submission

Abstract

Despite the success of ChatGPT, its performances on most NLP tasks are still well below the supervised baselines. In this work, we looked into the causes, and discovered that its subpar performance was caused by the following factors: (1) mismatch between the generation nature of ChatGPT and NLP tasks; (2) token limit in the prompt does not allow for the full utilization of the supervised datasets; (3) insufficient utilization of the reasoning power of ChatGPT. (4) intrinsic pitfalls of LLMs models, e.g., hallucination, overly focus on certain keywords, etc.

In this work, we propose a collection of general modules to address these issues, in an attempt to push the limits of ChatGPT on NLP tasks: (1) proper task formalization to better align with the generation nature of LLMs; (2) one-input-multiple-prompts strategy to overcome token limitations and maximize training data utilization; (3) demonstration retrieval using fine-tuned model for k-nearest neighbor (kNN) search to improve the selection of semantically relevant demonstrations; (4) chain-of-Thoughts reasoning that are tailored to addressing the task-specific complexity; (5) self-verification to address the hallucination issue of LLMs; (6) paraphrase voting to improve the robustness of model predictions.

We conduct experiments on 21 datasets of 10 representative NLP tasks. Using the proposed assemble of techniques, we are able to significantly boost the performance of ChatGPT on the selected NLP tasks, achieving performances comparable to or better than supervised baselines, or even existing SOTA performances.

1 Introduction

In recent years, interest in large language models (LLMs) such as ChatGPT[1] arises from their significant capabilities across a wide range of natural language tasks. Despite the success achieved by ChatGPT, its performances on most NLP tasks are still significantly below supervised baselines (Qin et al., 2023). This is due to the following reasons: (1) the mismatch between ChatGPT and many NLP tasks: ChatGPT is a text generation model, while many NLP tasks cannot be easily formatted as a text generation task, e.g., named entity recognition (NER), dependency parsing, semantic role labeling, etc. The adaptation from the original NLP task to a text generation task comes at a heavy cost in performance; (2) token limit: there is a hard token limit (4,096) for the input to the ChatGPT, which means only a small fraction of the labeled data can be used in the prompt for in-context learning (ICL); On the contrary, supervised baselines can harness the full labeled dataset; (3) the reasoning power of ChatGPT has not been fully fulfilled with respect to different tasks, which may require different reasoning ability to address the task-specific language complexity; and (4) the intrinsic pitfalls from ChatGPT itself: ChatGPT severely suffers from the hallucination issue (Ji et al., 2023), where in the context of NLP tasks, it tends to overconfidently label *null* instances with labels that they don't belong to.

In this paper, we explore how we can systematically address the aforementioned issues of ChatGPT, in an attempt to push the limit of its performances on different NLP tasks. We proposed a collection of strategies to systematically address the issues of using ChatGPT on NLP task: (1) *Proper Task Formalization* strategy is designed to address the mismatch problem. It reconstructs NLP tasks to formats that are more tailored to the generation nature. Prompting ChatGPT to copy the input and transforming labels to tokens surrounded with special symbols can preserve the generation nature of ChatGPT. Transforming N-class multi-

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

class classification task to N binary classification tasks further simplifying the process of extracting labels from output. (2) *One-input-multiple-prompts* strategy aims to alleviate the adverse effects of token limit and take advantage of more training data. It employs multiple prompts for one input to accommodate more demonstrations. Each prompt is filled with distinct demonstrations and fed into ChatGPT separately. The final decision is made by voting among all prompt belongs to one input. (3) *Demonstration Retrieval* strategy shares the goal of addressing the token limit problem as well. It uses representations from the fine-tuned model for kNN search to achieve better demonstration retrieval. kNN-based retrieval can select demonstrations more relevant in semantics to the input, making every token in the prompt count; (4) *Chain-of-Thoughts Reasoning* strategy is tailored to unleash the reasoning power of LLMs addressing the task-specific complexity. By prompting LLMs to generate chain-of-thoughts explanations for demonstrations before making decisions, it can reduce the randomness of model decoding and enhance LLMs' performance. (5) *Self-verification* strategy is dedicated to improve the robustness of model predictions and reduce hallucination. After obtaining the generated task results from ChatGPT, it concatenate the task description with the generated result and ask ChatGPT answer whether the generated result is correct or not. (6) *Paraphrase Voting* strategy is a way to mitigate the surface word domination issue of LLMs. Each input is paraphrased by LLMs with the same meaning but in different expressions. The final decision is made by voting among all output generated by LLMs with paraphrased prompts.

With the combination of the proposed strategies, we are able to significantly boost the performance of ChatGPT on all selected NLP tasks. We conduct experiments on 21 datasets of 10 representative NLP tasks, including question answering, commonsense reasoning, natural language inference, sentiment analysis, named entity recognition, entity-relation extraction, event extraction, dependency parsing, semantic role labeling, and part-of-speech tagging. Using the proposed assemble of techniques, we are able to significantly boost the performance of ChatGPT on the selected NLP tasks, achieving performances comparable to or better than supervised baselines, or even existing SOTA performances.

2 Methodology

In this section, we detail our proposed specific strategies to address the aforementioned disadvantages of the ChatGPT system.

2.1 Proper Task Formalization

We propose proper task formalization to restructure NLP tasks in a generative manner to meet the generation nature of ChatGPT. The first effective recipe we find effective is to prompt ChatGPT to copy the input while modifying labeled tokens by surrounding them with special symbols. For example, to extract location (LOC) entities in the input "*he lives in Seattle*" in the NER tasks, the output from ChatGPT surrounds the LOC entity "*Seattle*" with special symbols ## and @@, making the output "*he lives in ## Seattle@@*". This copy-and-modify approach not only preserves the continuity of the output, but also simplifies the process of connecting the output to the extracted tokens, resulting in superior results compared to other methods.

The second recipe is to transform the N-class multi-class classification task to N **binary classification** tasks. The intuition is that, for each class, we are able to show more illustrations with respect to that class with the binary-transformation strategy.

2.2 One-input-multiple-prompts

For ChatGPT, there is a hard limit of 4,096 token in the input. Therefore, only a very small fraction of training examples can be used. To address this issue, we propose the *one-input-multiple-prompts* strategy. Let N denotes the number of prompts for each input. Each prompt is filled with distinct demonstrations. Demonstrations are retrieved using random or kNN strategies. Prompts are fed to ChatGPT separately. We thus will get N predictions from ChatGPT. The final result is made via voting among the individual judgments made by ChatGPT for each prompt. By doing this, we can get around the restriction on tokens, allowing us to take the advantage of more training data.

2.3 Demonstration Retrieval

Another direction to address the limited token issue is to improve the quality of demonstrations to make every token in the prompt count. kNN-based retrieval is based on general sentence-level representations (Gao et al., 2021; Sun et al., 2022; Seonwoo et al., 2022) and retrieves similar demonstrations in terms of the general semantic. They surely perform better than random retrieval, but come with the key disadvantage: they do not extract features tailored to the specific task. A better alternative is to use the fine-tuned (FT for short) model on the training set as the similarity measurement function. Specifically, we first fine-tune a supervised model (e.g., RoBERTa (Liu et al., 2019)) based on the full training set, and use representations from the fine-tuned model for kNN search. From a global perspective, the FT-retrieval strategy bridges the gap between ChatGPT and the supervised model: though ChatGPT cannot fully use the training data as input due to the token limit, we can still setup their connection through the FT retrieval since the latter is trained based on the full training data.

2.4 Chain-of-Thoughts Reasoning

Wei et al. (2022b) propose the chain-of-thoughts (COT) strategy to enhance LLMs' reasoning abilities for solving math tasks: COT first generates intermediate rationale explanations and then followed by the task-related decision. For the kNN strategy, which is adopted in most NLP scenarios, each instance in the training set has a chance to be selected. Therefore, we need to prepare intermediate reasoning explanations for all training examples. To address this issue, we propose to use ChatGPT to generate rationale for all training examples. Specifically, at the rational-preparing stage, we first transform each data (INPUT, LABEL) in the training set to (INPUT, RATIONALE EXPLANATIONS, LABEL) by prompting ChatGPT to generate intermediate rationale explanations that support model decisions. At test time, we feed the concatenation of the task description, demonstrations that involve rationales, and the test instance to ChatGPT, in which case ChatGPT should generate a string that includes the reasoning process of ChatGPT, followed by its task-related decision for the input test.

2.5 Self-Verification

ChatGPT suffers from the hallucination issue (Ji et al., 2023), which generates false positive predictions with high confidence. Using the named entity recognition task as an example, the hallucination issue refers to ChatGPT extracting entities from sentences that do not contain any entities.

We propose the self-verification strategy (SV for short) to address the above issue. After obtaining the generated task results from ChatGPT, we concatenate the task description with the generated result and ask ChatGPT answer whether the generated result is correct or not. ChatGPT will generate a "*yes*" or "*no*" to determine whether the generated results are reasonable for the original task.

Let's take the named entity recognition task as an example to illustrate. Given the input "*Hunan Office in Beijing*". ChatGPT has completed the first step of extracting the location (LOC) entity and identified "Hunan" as a LOC entity. We employ the self-verification strategy to validate the LOC result obtained in the first step. We prompt ChatGPT:

> *INPUT: Hunan# Office in Beijing*
> *Based on the context, is the labeled 'Hunan' in the INPUT a location entity?*

ChatGPT should generate "*no*" indicating that "*Hunan*" is not a location entity. Afterward, we remove "*Hunan*" from the list of LOC entities predicted by ChatGPT in the first step.

2.6 Paraphrase Voting

ChatGPT often faces the issue that predictions are dominated by surface words. This is due to the limited demonstrations in prompts. Using the question-answering task as an example for an illustration: given the context "*The news agency reports that the goverment ...*", and the question "*What is the topic of the input text?*", ChatGPT is dominated by the phrase "*The news agency*" and generates "*news*" as the answer to the given question.

To address the surface word domination issue, we propose the paraphrase strategy. Specifically, we use ChatGPT to paraphrase the given text and get multiple versions of the input with the same meaning but in different expressions. Next, we use paraphrases as the input to ChatGPT one at a time, then employ a voting strategy to obtain the final decision.

It is worth noting that the paraphrase strategy can only be applicable to sentence-level tasks, but not token-level tasks (e.g., NER, POS). Because words in the generated paraphrases usually cannot be accurately aligned back to the original input.

3 Task Description and Re-formalization

In this section, we introduce the 10 representative NLP tasks description and corresponding re-formalization under ChatGPT. Detailed examples of each task formalization is shown in Figure 1.

3.1 Question Answering

Question answering (QA) (Seo et al., 2016; Xiong et al., 2016; Wang et al., 2017) is a task that generates an answer to a given natural language question, normally formalized as a multi-class (start, end, not part) classification problem.

Under ChatGPT, QA is formalized as a text generation task. We first split the given context into individual sentences and assign them a sentence index based on their position in the context. Then we concatenate the modified context and the given question to elicit a response from ChatGPT. The generated text string from ChatGPT should consist of two components: (1) the index of the sentence of which the answer is a substring; and (2) the substring that answers the question.

a strategy akin to multi-task learning.

3.2 Commonsense Reasoning

Commonsense reasoning (Bailey et al., 2015; Trinh and Le, 2018; Rajani et al., 2019a) is a task that uses human consensus and logical inference abilities to generate an answer to a given natural language question. The commonsense reasoning task is normally formalized as a binary classification task.

Under ChatGPT, commonsense reasoning is formalized as a text completion task to copy the right answer from the given multi-choice options.

3.3 Natural language inference

Natural language inference (NLI) (Wang and Jiang, 2015; Mou et al., 2015; Liu et al., 2016) is a task that aims to determine whether the given hypothesis can be logically inferred from the given premise and normally formalized as a three-class (entailment, contradiction and neural) classification problem.

Under ChatGPT, NLI is formalized as prompting ChatGPT to generate *yes/no* with respect to each logical relation (e.g., entailment), given the premise and the hypothesis. If the response is *yes*, it denotes that the relation holds between the premise and the hypothesis. Since there are three candidate logical relations, the prompting process should be repeated three times.

3.4 Sentiment Analysis

Sentiment analysis (Wilson et al., 2005; Devlin et al., 2018; Basiri et al., 2021) is a task to determine the sentimental polarity (e.g., positive, negative) of a given text. The task is normally formalized as a binary or multi-class classification problem, which assigns a sentiment class label to the given text.

Under ChatGPT, the task of sentiment analysis can be formalized as prompting ChatGPT to generate sentiment-indicative text given the input (e.g., *decide the sentiment of the following text*). The generated sentiment-indicative text contains sentiment keyword (e.g., positive, negative, etc) and will be latter mapped to a sentiment label.

3.5 Named Entity Recognition

Named entity recognition (NER) (Chiu and Nichols, 2015; Shang et al., 2018; Wang et al., 2023b) is a task that extracts named entities of pre-defined categories (e.g., location, organization, etc.) from a given text, normally formalized as a sequence labeling problem.

Under ChatGPT, NER is formalized as a text generation task, where given an input text (e.g., "*He lives in Chicago*"), and a certain entity type (e.g., *location*), we prompt ChatGPT to surround entities belonging to the entity type in the original sequence with special symbols:

> *He lives in ## Chicago* @@, where ## and @@ denote the start and end of a named entity.

If there is no location entity in the input, ChatGPT just copies the original input as the output. This strategy was adopted in Wang et al. (2023b).

3.6 Entity-Relation Extraction

Entity-relation extraction (Mintz et al., 2009; Miwa and Bansal, 2016; Wan et al., 2023) is a task that aims to extract named entities in a given text, and identify relations between the extracted entity pairs. The entity-relation extraction task is normally formalized as a two-stage problem: assigning an entity label then assigning a relation label.

Under ChatGPT, the entity-relation extraction is formalized as a two-step text completion task. Step 1, similar to that of NER, ChatGPT extracts named entities with respect to a certain type (e.g., location) by rewriting the input sentence and surrounding the entity with special tokens. Step 2, prompt ChatGPT to output a yes-or-no decision to determine whether a certain relation holds between two specified entities.

3.7 Event Extraction

Event extraction (EE) (Ahn, 2006; Nguyen and Grishman, 2018; Wang et al., 2021b) is a task that aims to identify the event type and extract information (i.e., trigger, arguments) of an identified event in the given text, normally formalized as a two-step classification problem.

Under ChatGPT, the event extraction task is formalized as a two-step text completion problem. Step 1, we prompt ChatGPT to generate a text string to determine whether the input contains the trigger word with respect to a certain event type. If ChatGPT responds with a substring of the input, it denotes that the substring is the trigger with respect to the certain event. If ChatGPT generates *null*, it indicates that the input does not contain an event with respect to the certain type. Step 2, we use ChatGPT to generate a text string, which is an argument with respect to a certain role for the identified event in Step 1. If ChatGPT generates a substring from the input, it denotes that the substring is the argument with respect to a certain role for the event, and *null* denotes otherwise.

3.8 Part-of-speech Tagging

Part-of-speech (POS) tagging (Brill, 1992; Owoputi et al., 2013; Chiche and Yitagesu, 2022) is a task that aims to assign a part-of-speech label to each word in the given sequence based on its morphology (e.g., past tense), semantic meaning (e.g., move or action), and syntactic functions (e.g., preposition). The POS task is normally formalized as a sequence labeling problem.

Under ChatGPT, the POS task is formalized as a text completion problem, where ChatGPT is prompted to generate a POS-indicative text for an annotated word in the sentence at a time. Specifically, we prompt ChatGPT to generate the POS for the marked word in the sentence with all POS options given. Suppose there are N words in the sentence, the above prompting process should be repeated N times.

3.9 Dependency Parsing

Dependency parsing (McDonald et al., 2005; Ma et al., 2018; Gan et al., 2021) is a task that aims to identify whether there are dependency relations between words in a sentence and determine the dependency relations. It is usually formalized as a multi-class classification task.

Under ChatGPT, dependency parsing is formalized as a two-step text completion task. Step 1, We use ChatGPT to rewrite the input sentence where dependent words for the given head word are marked with special tokens @#, where @ denotes the start of a dependent word, and # denotes the end of a dependent word. Step 2, we use ChatGPT to generate yes or no to determine whether a given dependency relation holds between the head and the dependent word.

3.10 Semantic Role Labeling

Semantic role labeling (SRL) (Zhou and Xu, 2015; He et al., 2018; Jia et al., 2022) is a task that aims to identify arguments for each predicate in a given sentence, along with determining semantic roles to the identified arguments. SRL is normally formalized as a two-stage problem: a multi-class classification task followed by a sequence labeling problem.

Under ChatGPT, the SRL task is formalized as a two-step text completion problem. Step 1, we use ChatGPT to determine the word sense of the predicate by iteratively asking ChatGPT whether the predicate belongs to each sense. Step 2, we use ChatGPT to output an argument that belongs to a certain semantic role with respect to the given predicate. Arguments are substrings of the input sentence. Suppose that there are N semantic roles, we need to ask ChatGPT N times, each of which corresponds to each role.

4 Experiments

In this section, we introduce the datasets used in 10 representative NLP tasks and the experimental results, following with corresponding analysis of the effectiveness of proposed 6 strategies. The overall comparisons of experiment results on ten NLP downstream tasks is shown in Figure 2, where with the proposed strategies, ChatGPT achieves comparable or better results to the supervised RoBERTa on 17 out of 21 datasets across 10 representative NLP tasks.

4.1 Datasets and Results

We conduct experiments on 21 widely-used benchmarks across 10 NLP tasks: (1) Question

Model	SQuADv2 (EM)	TQA (EM)	MRQA-OOD (F1)
RoBERTa-Large	86.8	**81.1**	72.4
ChatGPT *(few-shot)*			
+Random demo	70.1	70.9	63.1
+SimCSE kNN	73.5	72.5	65.7
+FT kNN	78.9	75.8	68.3
+FT kNN+Multi	83.6	78.0	71.0
+FT kNN+Multi+Reason	87.2	79.3	75.6
+FT kNN+Multi+Reason+SV	**88.2**	80.8	**76.1**

Table 1: Experimental results for the question answering task. We abbreviate the self-verification strategy as SV.

Models	CSQA (ACC)	StrategyQA (ACC)
RoBERTa-Large	**79.2**	**72.0**
ChatGPT *(few-shot)*		
+Random demo	74.8	59.5
+SimCSE kNN	74.7	59.6
+FT kNN	76.6	65.4
+FT kNN+Multi	78.0	67.8
+FT kNN+Multi+Reason	78.2	69.4
+FT kNN+Multi+Reason+SV	79.0	69.9

Table 2: Experimental results on commonsense reasoning datasets.

Models	RTE (ACC)	CB (ACC)
RoBERTa-Large	92.8	**98.2**
ChatGPT *(few-shot)*		
+Random demo	90.5	90.5
+SimCSE kNN	90.7	90.4
+FT kNN	92.2	93.8
+FT kNN+Multi	92.6	95.2
+FT kNN+Multi+Reason	92.9	95.6
+FT kNN+Multi+Reason+SV	92.9	96.5
+FT kNN+Multi+Reason+SV+Paraphrase	**93.1**	96.7

Table 3: Experiment results on natural language inference benchmarks.

Models	SST-2 (ACC)	IMDB (ACC)	Yelp (ACC)
RoBERTa-Large	95.9	**95.4**	98.0
ChatGPT *(few-shot)*			
+Random demo	92.6	90.4	95.5
+SimCSE kNN	92.8	90.5	95.7
+FT kNN	94.6	94.4	97.5
+FT kNN+Multi	95.2	94.8	97.8
+FT kNN+Multi+Reason	95.7	94.9	97.9
+FT kNN+Multi+Reason+SV	95.7	94.9	98.2
+FT kNN+Multi+Reason+SV+Paraphrase	**96.2**	95.1	**98.4**

Table 4: Experimental results for the sentiment analysis task.

Answering. SQuADv2 (Rajpurkar et al., 2018), TQA (Joshi et al., 2017), and MRQA-OOD. Results are shown in Table 1; (2) Commonsense Reasoning. CSQA (Talmor et al., 2018) and StrategyQA (Geva et al., 2021). Results are shown in Table 2; (3) Natural language inference. RTE, CommitmentBank (CB) (De Marneffe et al., 2019). Results are shown in Table 3; (4) Sentiment Analysis. SST-2, IMDb, and Yelp. Results are shown in Table 4; (5) Named Entity Recognition. CoNLL2003 (Sang and De Meulder, 2003) and OntoNotes5.0 (Pradhan et al., 2013a). Results are shown in Table 5; (6) Entity-Relation Extraction. English ACE2004 and ACE2005. Results are shown in Table 7. (7) Event Extraction. English ACE2005. Results are shown in Table 6; (8) Part-of-speech Tagging. WSJ Treebank and Tweets dataset. Results are shown in Table 8; (9) Dependency Parsing. English Penn Treebank v3.0 (Marcus et al., 1993). Results are shown in Table 10. (10) Semantic role labeling. CoNLL2005 (Carreras and i Villodre, 2005), CoNLL2009 (Hajic et al., 2009) and CoNLL2012 (Pradhan et al., 2013b). Results are shown in Table 9.

4.2 Analysis

In general, with the proposed series of strategies, ChatGPT is able to achieve comparable performances to the supervised baselines on 17 out of 21 datasets across 10 NLP tasks. In QA, under the out-of-domain setting of MRQA, ChatGPT significantly outperforms the supervised RoBERTa model by +3.7, which indicates the significantly better domain-adaptable ability of ChatGPT.

One-input-multiple-prompts. In Table 1, we observe that using the multiple-prompt strategy obtains a significant performance boost across three QA datasets, i.e., +4.7, +2.2, +2.7, respectively on SQuAD V2.0, TQA, and MRQA-OOD datasets. We can also observe that it gains significant performance boosts on two commonsense reasoning benchmarks: +1.4 on the CSQA and +2.4 on the StrategyQA datasets. This demonstrates that the multiple-prompt strategy effectively addresses the input token limit issue and allows ChatGPT to take advantage of more annotated examples.

Demonstration Retrieval. As shown in Table 1, the SimCSE-kNN retriever outperforms the random retriever, which demonstrates the importance of selecting semantically similar examples as demonstrations for QA. In Table 2 using the fine-tuned model to retrieve kNN introduces a huge performance boost compared with the SimCSE and the random selection strategies, i.e., +1.9 and +5.8 on the CSQA and StrategyQA dataset, respectively. Same boost by FT-kNN can be observed from Table 3 to Table 9 as

Model	CoNLL 2003 (Span-F1)	OntoNotes 5.0 (Span-F1)
RoBERTa-Large	**93.0**	**89.9**
ChatGPT *(few-shot)*		
+Random demo	68.4	58.3
+SimCSE kNN	80.2	72.1
+FT kNN	84.8	78.6
+FT kNN+Multi	88.2	81.4
+FT kNN+Multi+Reason	88.4	81.6
+FT kNN+Multi+Reason+SV	88.9	81.9

Table 5: Experimental results on the named entity recognition benchmarks.

Model	Trigger (Span-F1)	Argument (Span-F1)
RoBERTa-Large	74.5	63.6
ChatGPT *(few-shot)*		
+Random demo	60.3	50.2
+SimCSE kNN	65.5	55.0
+FT kNN	72.5	63.3
+FT kNN+Multi	74.3	64.5
+FT kNN+Multi+Reason	74.6	64.7
+FT kNN+Multi+Reason+SV	**74.6**	**64.9**

Table 6: Experimental results for the event extraction task.

Models	ACE 2004 (Span-F1)	ACE 2005 (Span-F1)
RoBERTa-Large	60.4	64.5
ChatGPT *(few-shot)*		
+Random demo	49.8	56.2
+SimCSE kNN	53.2	59.6
+FT kNN	59.2	63.9
+FT kNN+Multi	61.2	65.6
+FT kNN+Multi+Reason	61.7	66.0
+FT kNN+Multi+Reason+SV	**62.5**	**66.4**

Table 7: Experimental results for the entity-relation extraction task.

Models	Peen WSJ (ACC)	Tweets (ACC)
RoBERTa-Large	**98.9**	92.3
ChatGPT *(few-shot)*		
+Random demo	90.3	84.7
+SimCSE kNN	93.4	88.2
+FT kNN	98.2	92.4
+FT kNN+Multi	98.7	92.6
+FT kNN+Multi+Reason	98.7	92.6
+FT kNN+Multi+Reason+SV	**98.9**	**92.7**

Table 8: Experimental results on the part-of-speech datasets.

well. FT-kNN introduces a significant performance boost over SimCSE-kNN. This indicates that using the FT model, which is fine-tuned on the given training set, retrieves similar examples with respect to the specific task, and can help improve ChatGPT's performances.

Chain-of-Thoughts Reasoning. In Table 1 and Table 3, we find that the rational-based prompting strategy can further boost the performances, +1.8 on SQuADv2, +1.1 on TQA, +0.8 on MRQA-OOD compared to FTKNN+Multi-Prompts, and +0.3 on RTE and +0.4 on CB. This phenomenon is in line with our expectation that intermediate rationales enhances models' reasoning abilities.

Self-verification. In Table 1, the proposed self-verification strategy introduces further performance boosts, i.e., +1.0, +1.5, and + 1.5 on SQuADv2, TQA, and MRQA-OOD, respectively. Self-verification strategy yields minor performance improvement on datasets in Table 8, Table 10, Table 6 and Table 9. The explanation is that the performance without SV is already high enough that adding SV provides only a marginal improvement.

Paraphrase Voting. Similar to self-verification, the reason of paraphrase voting strategy bringing only minor performance improvement might be diminishing marginal effect. However, we are able to achieve consistent performance improvement across all datasets in Table 1 and Table 3, which indicates that shallow linguistic features (e.g. keywords, common words) mislead model decisions and the paraphrasing strategy can address the issue.

5 Related Work

5.1 Large language models (LLMs)

Large language models are models that aim to learn general language patterns and linguistic features by training in an unsupervised manner on large unannotated corpora (Zhu et al., 2015; Raffel et al., 2019; Lo et al., 2019; Gao et al., 2020; Kopf et al., 2023). With the scale increases, LLMs achieve great performance boosts on various NLP tasks while unlocking emergent capabilities (Xie et al., 2021; Wei et al., 2022a). Other efforts (Sanh et al., 2021; Wang et al., 2022; Longpre et al., 2023; Zhang et al., 2023) use human-instructions to boost LLM's ability. Based on model architectures, LLMs can be categorized into three branches: **(1) encoder-only models** (Devlin et al., 2018; Liu et al., 2019; Sun et al., 2020; Clark et al., 2020; Feng et al., 2020; Joshi et al., 2020; Sun et al., 2020, 2021) like BERT (Devlin et al., 2018) are discriminative models that use a transformer (Vaswani et al., 2017) encoder for getting the representation of a given sequence; **(2) decoder-only models** (Radford et al., 2019a; Dai et al., 2019; Keskar et al., 2019; Radford et al.,

Model	CoNLL 2009 Predicate Disambiguation (ACC)	CoNLL 2009 Argument Labeling (F1)	CoNLL 2005 Argument Labeling (F1)	CoNLL 2012 Argument Labeling (F1)
RoBERTa-Large	97.3	93.3	89.3	87.6
ChatGPT *(few-shot)*				
+Random demo	83.2	79.0	76.8	76.4
+SimCSE kNN	89.4	84.8	83.1	82.8
+FT kNN	97.4	93.5	88.9	87.4
+FT kNN+Multi	97.8	93.8	89.8	88.2
+FT kNN+Multi+Reason	97.8	94.0	90.4	88.4
+FT kNN+Multi+Reason+SV	**97.7**	**94.1**	**90.8**	**88.6**

Table 9: Experimental results for the semantic role labeling task.

Model	PTB (UAS)	PTB (LAS)
RoBERTa-Large	**96.87**	**95.34**
ChatGPT *(few-shot)*		
+Random demo	79.04	77.32
+SimCSE kNN	85.45	84.01
+FT kNN	92.45	90.98
+FT kNN+Multi	93.72	92.20
+FT kNN+Multi+Reason	94.24	92.72
+FT kNN+Multi+Reason+SV	94.88	93.20

Table 10: Experimental results for the dependency parsing task.

2019b; Brown et al., 2020; Chowdhery et al., 2022; Ouyang et al., 2022; Zhang et al., 2022a; Scao et al., 2022; Zeng et al., 2022; Touvron et al., 2023; Taori et al., 2023; Chiang et al., 2023; Peng et al., 2023; Anand et al., 2023; OpenAI, 2023) like GPT (Radford et al., 2019a) are generative models that use the decoder of an auto-regressive transformer (Vaswani et al., 2017) for predicting the next token in a sequence; **(3) encoder-decoder models** (Lewis et al., 2019; Raffel et al., 2020; Xue et al., 2020) like T5 (Raffel et al., 2020) are generative models that use both the encoder and decoder of the transformer (Vaswani et al., 2017) model. Models finish downstream tasks by generating new sentences depending on a given input.

5.2 Adapting LLMs to NLP tasks

In-context Learning (ICL) has been adopted as a general strategy to apply LLMs to downstream NLP tasks. Brown et al. (2020) prompted LLMs to generate textual responses (i.e., label words) conditioning on the given prompt with a few annotated examples without gradient updates. There are a variety of strategies to improve ICL performances on NLP tasks: Li and Liang (2021); Zhong et al. (2021); Qin and Eisner (2021) propose to optimize prompts in the continuous space; Rubin et al. (2021); Das et al. (2021); Liu et al. (2021); Gonen et al. (2022); Su et al. (2022); Wang et al. (2023b); Wan et al. (2023) investigate different strategies for selecting in-context examples; Gonen et al. (2022) exploit different strategies for orders of in-context examples. More advanced reasoning strategies Wei et al. (2022b); Zhang et al. (2022b); Han et al. (2021); Fu et al. (2022); Zhou et al. (2022a); Sun et al. (2023b) also use in-context learning as the backbone.

6 Conclusion

In this paper, we present a comprehensive set of strategies with the aim of advancing the performance boundaries of ChatGPT. These strategies encompass: (1) proper task formalization; (2) one-input-multiple-prompts; (3) demonstrations retrieval; (4) chain-of-thoughts reasoning; (5) self-verification; (6) paraphrase voting. These proposed strategies effectively target the underlying factors that contribute to ChatGPT's performance falling below optimal levels: (1) addressing the incongruence between ChatGPT's generative nature and the demands of NLP tasks; (2) overcoming the token limit constraint in input prompts to maximize the utility of supervised datasets; (3) unlocking the untapped reasoning capabilities of ChatGPT; (4) mitigating intrinsic challenges observed in Large Language Models (LLMs), such as hallucination and excessive focus on specific keywords. With the proposed strategies, ChatGPT achieves comparable or better results to the supervised RoBERTa on 17 of 21 datasets across 10 representative NLP tasks.

Limitations

This paper acknowledges the limitations inherent in using ChatGPT for natural language processing (NLP) tasks. One primary limitation is the model's dependency on its training data, which may not encompass the entire breadth and diversity of human language, leading to potential biases or gaps in knowledge. ChatGPT, like other large language models, may struggle with understanding and generating contextually appropriate responses, especially in nuanced or highly specialized topics. Additionally, the model's ability to discern and replicate factual accuracy is not foolproof, as it can inadvertently propagate misinformation present in its training data. Another key limitation is the handling of real-time data or events post its last training update, leaving it unable to provide insights on very recent developments. The computational resource requirement for operating such a model is significant, which could pose scalability challenges.

A Releated work

A.1 Generation Intermediate Rationales

Rajani et al. (2019b) improve the interpretability of the model without sacrificing its performance by training a language model on the "explain-then-predict" commonsense answering dataset. Recently, Nye et al. (2021) find that a step-by-step computation “scratchpads” can improve LLM’s performances on arithmetic, polynomial evaluation, and program evaluation tasks and etc. Wei et al. (2022b) use manually annotated "chain-of-thoughts" prompts and greatly improve performances of LLM on complex reasoning tasks. After that, Li et al. (2022); Fu et al. (2022); Ye et al. (2022); Shao et al. (2023) use higher reasoning complexity examples as demonstrations and further improve LLMs performances on complex reasoning tasks. Zhou et al. (2022b); Press et al. (2022) decompose a complex problem into a series of simpler subproblems and then solve them step-by-step toward the final answer. Zhang et al. (2022b); Kojima et al. (2022); Zelikman et al. (2022); Chen et al. (2022); Sun et al. (2023a); Wang et al. (2023a); Sun et al. (2023b) propose strategies to use LLMs generate explicit intermediate reasoning chains and then improve LLMs’ complex reasoning ability with self-generated "chain-of-thoughts".

B Task formalaition under ChatGPT

B.1 Question Answering

For example, with the prompt to ChatGPT being:

Context: (1) The capital of Japan is Tokyo. (2) The capital of China is Beijing. (3) The capital of South Korea is Seoul.
Question: What is the capital of South Korea?

ChatGPT should output

(3) Seoul

where "*(3)*" denotes the index of the sentence where the answer is located and "*Seoul*" represents the answer. This strategy provides the model with further guidance by first predicting the index of the sentence within the context, and then deciding which substring in that sentence should be used as the answer, a strategy akin to multi-task learning. Examples of the task formalization are shown in Figure 1

B.2 Commonsense Reasoning

For example, the question is "*Where on a river can you hold a cup upright to catch water on a sunny day?*", the answer choices are "*(A) waterfall; (B) bridge; (C) valley; (D) pebble*" and the prompt to ChatGPT is:

Please select the answer to the question from several options.
Question: Where on a river can you hold a cup upright to catch water on a sunny day?
Options: (A) waterfall; (B) bridge; (C) valley; (D) pebble.

The ChatGPT should generate "*(D) pebble*" which denotes that the *pebble* is the answer to the given question. Examples of the task formalization are shown in Figure 1.

B.3 Natural Language Inference

For example, given the premise "*Pibul Songgramwas the pro-Japanese military dictator of Thailand during World War 2.*", the hypothesis "*Pibul was the dictator of Thailand.*", and the prompts to ChatGPT is

Does the premise entail the hypothesis? Please answer yes or no.
Premise: <premise>
Hypothesis: <hypothesis>

where *<premise>* and *<hypothesis>* should be replaced by the given premise and hypothesis, respectively. ChatGPT should output "*no*" ideally in this case, which denotes that the premise does not entail the hypothesis. Subsequently, the same procedures are used to verify the contradiction and neural relations. Examples of the task formalization are shown in Figure 1.

Figure 1: Task Formalizations under ChatGPT, including question answering, commonsense reasoning, natural language inference, sentiment analysis, named entity recognition, entity-relation extraction, event extraction, dependency parsing, semantic role labeling, and part-of-speech tagging.

B.4 Entity-relation Extraction

For example, given the input sentence "*In 2002, Musk founded SpaceX*", we would like to extract entities with respect to the *organization* type. The input to ChatGPT is:

Please mark the start and end of ORG entities in the INPUT with @ and #, respectively.
INPUT: In 2002, Musk founded SpaceX.

and ChatGPT should output "*In 2002, Musk founded @SpaceX#,*" where @ and # are the starting and ending boundaries of ORG entities and the substring "*SpaceX*" is an ORG entity. If there is no ORG entity in the given text, ChatGPT should copy the input. Suppose the number of NER categories is N, the above prompt should be repeated N time for one input. The process here is similar to that of NER in Section **??**.

Step 2: Prompt ChatGPT to output a yes-or-no decision to determine whether a certain relation holds between two specified entities. In the example above, we have already identified the *person* entity "*Musk*" and the *organization* entity "*SpaceX*" at stage-1, the second step involves determining the relation between them. Assuming that there are M possible relationships between entities, we ask ChatGPT each relation at a time, e.g., regarding the relation type *founded*, the input to ChatGPT is:

Please determine whether the relationship between the entities Musk (person) and SpaceX (organization) in the input sentence is 'founded.' Please answer with Yes/No.
Input: In 2002, Musk# founded SpaceX#.

In this case, ChatGPT should generate "*Yes*", indicating that Musk founded SpaceX.

C Evaluation Datasets

C.1 Question answering

- **SQuAD V2.0**: SQuAD V2.0[2] is a collection of 100K crowdsourced question-answer pairs which are originally from a set of Wikipedia articles. In this dataset, the answer to every question is a span of text from the context passage, or the question is unanswerable.

- **TriviaQA**: TriviaQA[3] is a question-answering dataset which which includes 950K question-answer pairs from 662K documents collected from Wikipedia and the web. In TriviaQA, all questions are answerable and answers may not be directly obtained from the given context.

- **MRQA OOD**: MRQA[4] out-of-domain (OOD) is a shared task which is to evaluate generalization to out-of-distribution data. The test data contains 12 subsets, each from a held-out domain.

C.2 Commonsense Reasoning

- **CommonsenseQA**: CommonsenseQA[5] is a multiple-choice question answering dataset which requires commonsense knowledge to select one correct answer from four options to the question. The train/valid/test set contains 9,741, 1,221, and 1,140 questions, respectively.

- **StrategyQA**: StrategyQA[6] is an open-domain question answering dataset which requires implicit reasoning and logical inference to answer the question. There are 111 examples for the train set. The dataset contains 2,821 examples in the train set and 490 examples in the test set.

C.3 Event Extraction

For example, given the input text "*On Sunday, a protester attacked an officer with a paper cutter.*", the event type *Attack*, the prompt to ChatGPT is:

Given the sentence 'On Sunday, a protester stabbed an officer with a paper cutter', what is the trigger word of the attack event?" .

ChatGPT should generate "*stabbed*", which denotes that the sentence contains an attack event and its event trigger is "*stabbed*".

Step 2: We use ChatGPT to generate a text string, which is an argument with respect to a certain role for the identified event in Step 1. Suppose that there are M types of arguments for the event, we should repeat the prompting process M times for one input. If ChatGPT generates a substring from the input, it denotes that the substring is the argument with respect to a certain role for the event. If ChatGPT responds with *null*, it denotes that the input text does not contain an argument with respect to the

[2]https://rajpurkar.github.io/SQuAD-explorer/
[3]https://nlp.cs.washington.edu/triviaqa/
[4]https://mrqa.github.io/
[5]https://www.tau-nlp.sites.tau.ac.il/commonsenseqa
[6]https://leaderboard.allenai.org/strategyqa/submissions/get-started

certain role for the event. In the example above, we have already identified an *attack* event in the input and the trigger for the event is "*stabbed*". In this step, we would like to extract the argument with the *target* role for the *attack* event. We feed the following prompt to ChatGPT:

INPUT: On Sunday, a protester ##stabbed an officer with a paper cutter.
The INPUT contains an "attack" event, and the "stabbed" is the event trigger (marked with ## in the INPUT).
What was the target of the stabbing in the attack?"

In this case, ChatGPT should generate "*an officer*", which represents that the target for the attack event is "*an officer*". Examples of the task formalization are shown in Figure 1.

C.4 Part-of-speech Tagging

For example, given the input sentence "*Vinken,61 years old.*", and the marked word is "*Vinken*". We feed the following prompt to ChatGPT:

Part-of-speech categories are as follows:
1. CC Coordinating conjunction
...
15. NNPS Proper noun, plural
...
45. DT Determiner.
INPUT: ##Vinken, 61 years old
QUESTION: What is the POS tag for the word 'Vinken' which is marked by ## in the INPUT?"

ChatGPT should generate "*15. NNPS Proper noun, plural*", denoting that the POS for "*Vinken*" is NNPS. Examples of the task formalization are shown in Figure 1.

C.5 Part-of-speech Tagging

We use the example above as an illustration, where the input sentence is "*I prefer the morning flight to Denver.*", the head word is "*prefer*", and the prompt fed to ChatGPT is:

The head word in the input is marked with @#.
Find the dependents of the head word.
Input:I @prefer# the morning flight to Denver.

ChatGPT should output:

@I# prefer the morning @flight# to Denver."

where "*I*", "*flight*" are marked and denote the dependent words for the head word "*prefer*". If ChatGPT generates a sentence without the special token, it means that there are no dependent words for the given head word. Suppose that the given sentence is composed of N words, we use one word at a time as the headword and will prompt ChatGPT N times.

As shown in Step 1, "*prefer*" and "*I*" have a dependency relation. In this step, we will determine the category of the relation between the head word "*prefer*" and the dependent word "*I*". Suppose that there are M types of dependency relations, we should prompt ChatGPT M times, each of which corresponds to one type. For example, if we need to identify whether the dependency relation between "*prefer*" and "*I*" is *direct object*, the prompt fed to ChatGPT is:

@I# @prefer# the morning flight to Denver.
whether the relation between "prefer" and "I" is the direct object?

ChatGPT should generate "*No*" in this case. Assuming that we obtain C dependency word pairs from step 1, we should ask ChatGPT $M * C$ times for the given sentence. Examples of the task formalization under ChatGPT are shown in Figure 1.

C.6 Semantic Role Labeling

Use the example above as an illustration, where the sentence is "*The stock has been beaten down for two days*", the predicate is "*beaten*", there are three sense candidates for the predicate: (1)*(Cause) pulsating motion that often makes sound*, (2)*push, cause motion*; and (3)*win over some competitor*. We iteratively ask ChatGPT whether the predicate belongs to each of the three senses.

Suppose that we would like to find the argument with the *thing moving* semantic role, the input to ChatGPT is:

What are the arguments representing the meaning of 'thing moving'?.

ChatGPT should output "*The stock*". If ChatGPT returns *null*, it indicates that there is no argument in the sentence with the semantic role. Examples of the task formalization under ChatGPT are shown in Figure 1.

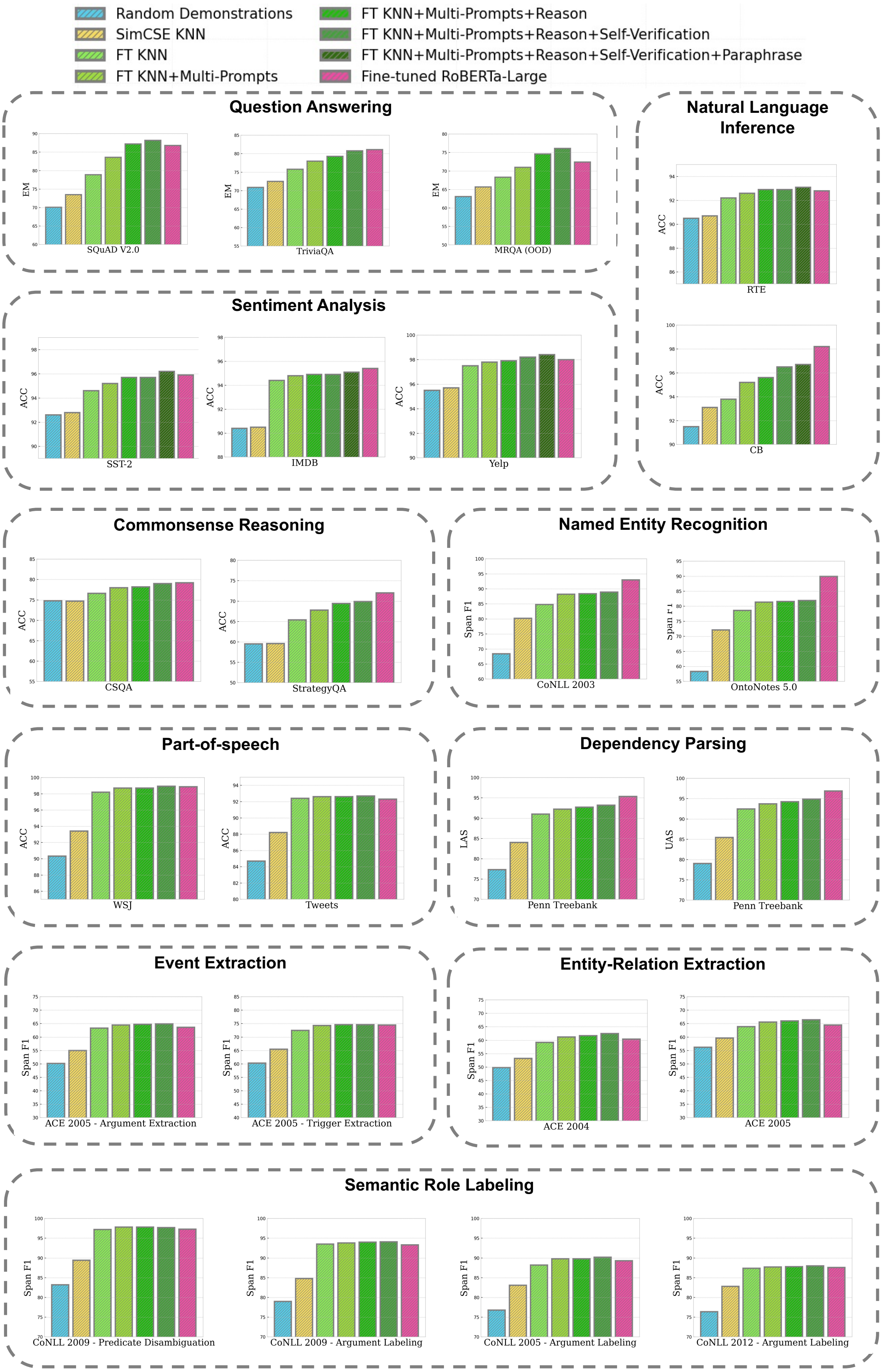

Figure 2: Comparisons of experiment results on ten NLP downstream tasks.

C.7 Natural language inference

- **RTE**: Recognizing Textual Entailment (RTE)[7] is an English dataset which is built from news and Wikipedia and from text entailment challenges.

[7]https://aclweb.org/aclwiki/Recognizing_Textual_Entailment

- **CB**: CB[8] is a corpus of 1,200 naturally occurring discourses whose final sentence contains a clause-embedding predicate under an entailment canceling operator (question, modal, negation, antecedent of conditional).

C.8 Sentiment analysis

- **SST-2**: SST-2 is a binary (i.e., positive, negative) sentiment classification dataset with 11,855 single sentences extracted from snippets of Rotten Tomatoes HTML files.

- **IMDB**: IMDB is a binary (i.e., positive, negative) sentiment classification dataset which includes 25,000 highly polar movie reviews for training and 25,000 for testing.

- **Yelp**: Yelp is a binary (i.e., positive, negative) sentiment classification dataset that contains 560,000 highly polar Yelp reviews for training and 38,000 for testing.

C.9 Named entity recognition

- **CoNLL 2003**: CoNLL 20033 is an English NER benchmark that includes four entity types: location, organization, person and miscellaneous. We follow Ma and Hovy (2016) and use the same train/dev/test split.

- **OntoNotes 5.0**: OntoNotes 5.0 is an English NER dataset and contains 18 entity types. We use the standard train/dev/test split of CoNLL2012 shared task.

C.10 Entity-relation extraction

- **ACE2004**: ACE2004[9] is a multilingual information extraction benchmark. The dataset contains 7 entity types and 7 relation categories. In this paper, we use English annotations and follow Li et al. (2019) to split train/valid/test datasets.

- **ACE2005**: ACE2005 is a multilingual training corpus. It has six relation categories, and we process and split the dataset following the practice in Li et al. (2019) There are six subdomains in the dataset: Broadcast Conversations (BC), Broadcast News (BN), Conversational Telephone Speech (CTS), Newswire (NW), Usenet Newsgroups (UN), and Weblogs (WL).

[8] https://github.com/mcdm/CommitmentBank

[9] https://catalog.ldc.upenn.edu/LDC2005T09

C.11 Event Extraction

- **ACE 2005**: ACE 2005 event corpus defines 8 event types and 33 subtypes, each event subtype corresponding to a set of argument roles. There are 36 argument roles for all event subtypes. In most of researches based on the ACE corpus, the 33 subtypes of events are often treated separately without further retrieving their hierarchical structures. The ACE 2005 corpus contains 599 annotated documents and around 6000 labeled events, including English, Arabic and Chinese events from different media sources like newswire articles, broadcast news and etc. In this paper, use the English subset and follow Liu et al. (2018) to split the train/valid/test sets.

C.12 Part-of-speech tagging

- **Penn WSJ**: Penn Treebank (PTB) is an English dataset corresponding to the articles of Wall Street Journal (WSJ). In this paper, we use sections from 0 to 18 are used for training (38, 219 sentences, 912, 344 tokens), sections from 19 to 21 are used for validation (5,527 sentences, 131,768 tokens), and sections from 22 to 24 are used for testing (5,462 sentences, 129,654 tokens).

C.13 Dependency parsing

- **PTB**: PTB is an English dataset that contains 39,832 sentences for training and 2,416 sentences for testing. We follow Ma et al. (2018) and use the same train/valid/test split.

C.14 Semantic role labeling

- **CoNLL2005**: CoNLL2005 contains a total number of 20 roles, while there are only 2.5 roles per predicate on average. we use sections 02-21 of WSJ corpus as Train data, section 24/23 as Dev/Test data, and three sections (CK01-03) of the Brown corpus as out-of-domain (OOD) data.

- **CoNLL2009**: CoNLL2009 builds on the CoNLL-2008 task and extends it to multiple languages. Data is provided for both statistical training and evaluation, which extract these labeled dependencies from manually annotated treebanks such as the Penn Treebank for English. We follow Wang et al. (2021a) to test on the English data and use the same train/valid/test split for evaluations.

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.

Yuvanesh Anand, Zach Nussbaum, Brandon Duderstadt, Benjamin Schmidt, and Andriy Mulyar. 2023. Gpt4all: Training an assistant-style chatbot with large scale data distillation from gpt-3.5-turbo.

Daniel Bailey, Amelia J Harrison, Yuliya Lierler, Vladimir Lifschitz, and Julian Michael. 2015. The winograd schema challenge and reasoning about correlation. In *2015 AAAI Spring Symposium Series*.

Mohammad Ehsan Basiri, Shahla Nemati, Moloud Abdar, E. Cambria, and U. Rajendra Acharrya. 2021. Abcdm: An attention-based bidirectional cnn-rnn deep model for sentiment analysis. *Future Gener. Comput. Syst.*, 115:279–294.

Eric Brill. 1992. A simple rule-based part of speech tagger. In *Human Language Technology - The Baltic Perspectiv*.

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

Xavier Carreras and Lluís Màrquez i Villodre. 2005. Introduction to the conll-2005 shared task: Semantic role labeling. In *Conference on Computational Natural Language Learning*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *ArXiv*, abs/2211.12588.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Alebachew Chiche and Betselot Yitagesu. 2022. Part of speech tagging: a systematic review of deep learning and machine learning approaches. *Journal of Big Data*, 9:1–25.

Jason P. C. Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay-Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. *arXiv preprint arXiv:2104.08762*.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. 2020. Codebert: A pre-trained model for programming and natural languages. *arXiv preprint arXiv:2002.08155*.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2022. Complexity-based prompting for multi-step reasoning. *arXiv preprint arXiv:2210.00720*.

Leilei Gan, Yuxian Meng, Kun Kuang, Xiaofei Sun, Chun Fan, Fei Wu, and Jiwei Li. 2021. Dependency parsing as mrc-based span-span prediction. *arXiv preprint arXiv:2105.07654*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Hila Gonen, Srini Iyer, Terra Blevins, Noah A Smith, and Luke Zettlemoyer. 2022. Demystifying prompts in language models via perplexity estimation. *arXiv preprint arXiv:2212.04037*.

Jan Hajic, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez i Villodre, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Stepánek, Pavel Stranák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *CoNLL Shared Task*.

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

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Annual Meeting of the Association for Computational Linguistics*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Zixia Jia, Zhaohui Yan, Haoyi Wu, and Kewei Tu. 2022. Span-based semantic role labeling with argument pruning and second-order inference. In *AAAI Conference on Artificial Intelligence*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

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

Andreas Kopf, Yannic Kilcher, Dimitri von Rutte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Rich'ard Nagyfi, ES Shahul, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. 2023. Openassistant conversations - democratizing large language model alignment. *ArXiv*, abs/2304.07327.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. *arXiv preprint arXiv:1905.05529*.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2022. On the advance of making language models better reasoners. *ArXiv*, abs/2206.02336.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Jian Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2018. Event detection via gated multilingual attention mechanism. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidirectional lstm model and inner-attention. *ArXiv*, abs/1605.09090.

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*.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Dan S Weld. 2019. S2orc: The semantic scholar open research corpus. *arXiv preprint arXiv:1911.02782*.

S. Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. *ArXiv*, abs/2301.13688.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Xuezhe Ma, Zecong Hu, J. Liu, Nanyun Peng, Graham Neubig, and Eduard H. Hovy. 2018. Stack-pointer networks for dependency parsing. In *Annual Meeting of the Association for Computational Linguistics*.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (ACL'05)*, pages 91–98.

Mike D. Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Annual Meeting of the Association for Computational Linguistics*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. *ArXiv*, abs/1601.00770.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2015. Natural language inference by tree-based convolution and heuristic matching. *arXiv: Computation and Language*.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *AAAI Conference on Artificial Intelligence*.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Olutobi Owoputi, Brendan T. O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *North American Chapter of the Association for Computational Linguistics*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013a. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013b. Towards robust linguistic analysis using ontonotes. In *Conference on Computational Natural Language Learning*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *ArXiv*.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*.

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

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019a. Language models are unsupervised multitask learners.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019b. Language models are unsupervised multitask learners. *OpenAI blog*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

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*, 21(1):5485–5551.

Nazneen Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019a. Explain yourself! leveraging language models for commonsense reasoning. *ArXiv*, abs/1906.02361.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019b. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Rose Biderman, Leo Gao, Tali

Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask prompted training enables zero-shot task generalization. *ArXiv*, abs/2110.08207.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *ArXiv*, abs/1611.01603.

Yeon Seonwoo, Guoyin Wang, Sajal Choudhary, Changmin Seo, Jiwei Li, Xiang Li, Puyang Xu, Sunghyun Park, and Alice Oh. 2022. Ranking-enhanced unsupervised sentence representation learning. *arXiv preprint arXiv:2209.04333*.

Jingbo Shang, Liyuan Liu, Xiang Ren, Xiaotao Gu, Teng Ren, and Jiawei Han. 2018. Learning named entity tagger using domain-specific dictionary. In *Conference on Empirical Methods in Natural Language Processing*.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Synthetic prompting: Generating chain-of-thought demonstrations for large language models. *ArXiv*.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. 2022. Selective annotation makes language models better few-shot learners. *arXiv preprint arXiv:2209.01975*.

Jiashuo Sun, Yi Luo, Yeyun Gong, Chen Lin, Yelong Shen, Jian Guo, and Nan Duan. 2023a. Enhancing chain-of-thoughts prompting with iterative bootstrapping in large language models. *ArXiv*.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023b. Text classification via large language models. *arXiv preprint arXiv:2305.08377*.

Xiaofei Sun, Yuxian Meng, Xiang Ao, Fei Wu, Tianwei Zhang, Jiwei Li, and Chun Fan. 2022. Sentence similarity based on contexts. *Transactions of the Association for Computational Linguistics*, 10:573–588.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34.

Zijun Sun, Xiaoya Li, Xiaofei Sun, Yuxian Meng, Xiang Ao, Qing He, Fei Wu, and Jiwei Li. 2021. Chinesebert: Chinese pretraining enhanced by glyph and pinyin information. *arXiv preprint arXiv:2106.16038*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models*.

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

Trieu H. Trinh and Quoc V. Le. 2018. A simple method for commonsense reasoning. *ArXiv*, abs/1806.02847.

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

Zhen Wan, Fei Cheng, Zhuoyuan Mao, Qianying Liu, Haiyue Song, Jiwei Li, and Sadao Kurohashi. 2023. Gpt-re: In-context learning for relation extraction using large language models. *arXiv preprint arXiv:2305.02105*.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *ArXiv*.

Nan Wang, Jiwei Li, Yuxian Meng, Xiaofei Sun, Han Qiu, Ziyao Wang, Guoyin Wang, and Jun He. 2021a. An mrc framework for semantic role labeling. *arXiv preprint arXiv:2109.06660*.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. 2023b. Gpt-ner: Named entity recognition via large language models. *arXiv preprint arXiv:2304.10428*.

Shuohang Wang and Jing Jiang. 2015. Learning natural language inference with lstm. *ArXiv*, abs/1512.08849.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and M. Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Annual Meeting of the Association for Computational Linguistics*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language model with self generated instructions. *ArXiv*, abs/2212.10560.

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juan-Zi Li, and Jie Zhou. 2021b. Cleve: Contrastive pre-training for event extraction. *ArXiv*, abs/2105.14485.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Human Language Technology - The Baltic Perspectiv*.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.

Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. *ArXiv*, abs/1611.01604.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Xi Ye, Srini Iyer, Asli Celikyilmaz, Ves Stoyanov, Greg Durrett, and Ramakanth Pasunuru. 2022. Complementary explanations for effective in-context learning. *ArXiv*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*.

Ge Zhang, Yemin Shi, Ruibo Liu, Ruibin Yuan, Yizhi Li, Siwei Dong, Yu Shu, Zhaoqun Li, Zekun Wang, Chenghua Lin, Wen-Fen Huang, and Jie Fu. 2023. Chinese open instruction generalist: A preliminary release. *ArXiv*, abs/2304.07987.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022a. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022b. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall. *arXiv preprint arXiv:2104.05240*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2022a. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Denny Zhou, Nathanael Scharli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Huai hsin Chi. 2022b. Least-to-most prompting enables complex reasoning in large language models. *ArXiv*.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Annual Meeting of the Association for Computational Linguistics*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Example Appendix

This is an appendix.