SPEAK IT OUT: SOLVING SYMBOL-RELATED PROBLEMS WITH SYMBOL-TO-LANGUAGE CONVERSION FOR LANGUAGE MODELS

Yile Wang[1] **Sijie Cheng**[1,2,3] **Zixin Sun**[2] **Peng Li**[*1,4] **Yang Liu**[*1,2,4]
[1]Institute for AI Industry Research (AIR), Tsinghua University, Beijing, China
[2]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[3]01.AI [4]Shanghai Artificial Intelligence Laboratory, Shanghai, China
{wangyile,lipeng}@air.tsinghua.edu.cn, liuyang2011@tsinghua.edu.cn

ABSTRACT

Symbols (or more broadly, non-natural language representations) such as numerical sequences, molecular formulas, and table delimiters widely exist, playing important roles in various tasks such as abstract reasoning, chemical property prediction, and table question answering. Despite the impressive natural language comprehension capabilities of large language models (LLMs), their reasoning abilities for symbols remain inadequate, which could attributed to the difference between symbol representations and general natural languages. We propose symbol-to-language (S2L), a tuning-free method that enables large language models to solve *symbol*-related problems with information expressed in natural *language*. Specifically, S2L first converts the symbols involved to language-based representations, which can be implemented by prompting LLMs or leveraging external tools, then these language-based representations are integrated into the original problem via direct substitution or concatenation, serving as useful input information for LLMs. We evaluate the S2L method using both API-based (GPT-4, ChatGPT) and open-source (OpenChat) models over eight symbol-related tasks, ranging from symbol-only abstract reasoning to sentiment analysis in social media. Experimental results show that S2L consistently leads to superior performance. For example, by employing S2L for GPT-4, there can be average significant improvements of $+21.9\%$ and $+9.5\%$ for subtasks in 1D-ARC and Dyck language, respectively. Codes and data are available at https://github.com/THUNLP-MT/symbol2language.

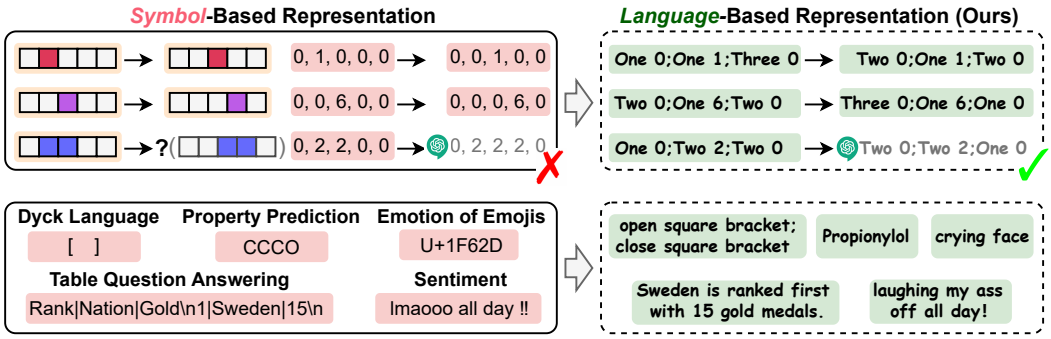

Figure 1: **Top**: We show a symbol-related problem (move 1 pixel forward) from the 1D-ARC benchmark, comparing the responses of large language models using both conventional symbol-based and our language-based representations with symbol-to-language (S2L) conversion. **Bottom**: The S2L conversion has broad applicability across various scenarios involving diverse types of symbols.

*Corresponding authors.

1 INTRODUCTION

Symbols, or more broadly, non-natural language representations such as brackets, digits, molecular formulas, emojis, table delimiters, and abbreviations are ubiquitously encountered in the real world. They are of significant relevance in our daily lives, convey distinct meanings, and play important roles in a variety of tasks. These symbol-related tasks include abstract reasoning (Moskvichev et al., 2023; Xu et al., 2023c), chemical property prediction (Ross et al., 2022; Guo et al., 2023), and tabular question-answering (Chen et al., 2020; Chen, 2023), as exemplified in Table 1. Consequently, the understanding and reasoning abilities of symbols are of paramount importance for artificial intelligence (Chollet, 2019).

Nowadays, large language models (LLMs; Brown et al., 2020; Ouyang et al., 2022; OpenAI, 2022; 2023b; Jiang et al., 2023; Google et al., 2023) have demonstrated impressive abilities in comprehending and generating natural language. GPT-3 (Brown et al., 2020) has showcased capabilities of zero-shot inference that solve problems directly without demonstrations. Kojima et al. (2022) further propose zero-shot-CoT through additional prompting like "*Let's think step by step*" to enhance the zero-shot reasoning capability. However, the understanding and reasoning capability of symbols for LLMs still fall behind compared to general natural language. For example, Mitchell et al. (2023) reveal that GPT-4 (OpenAI, 2023b) and GPT-4V (OpenAI, 2023a) only achieve an accuracy of 65% and 25% on minimal abstract reasoning tasks (Moskvichev et al., 2023) requiring inductive reasoning through a series of regular numbers or pixels, which is significantly lower than the human accuracy of 95%. Gendron et al. (2023) further demonstrate that existing LLMs exhibit limited performance for symbol-related problems in contrast with other natural language tasks.

The inadequate symbol-related reasoning capability of LLMs can be attributed to two primary factors. First, symbols are significantly underrepresented in the training corpus compared to natural language (Ohsuga, 2007), leading to an understanding gap between low-frequency symbols and LLMs (Kandpal et al., 2023; Tang et al., 2023a). Thus, recent studies have explored the collection of symbolic data (*e.g.*, first-order logic, biomolecule, SQL) for continuous training of LLMs (Yang et al., 2023; Fang et al., 2023; Xu et al., 2023a), which demands massive human annotations and computational resources. Second, the reasoning capability of LLMs is compromised when dealing with symbolic-related problems due to the subpar understanding of symbol-based representations. Previous studies (Xu et al., 2023a; Gendron et al., 2023; Wang et al., 2023b) directly use these symbol-based representations as inputs for reasoning, while language-based rationales (Wang et al., 2022; Kojima et al., 2022; Wei et al., 2022) would further induce error propagation due to suboptimal understanding capability of the involved symbols.

Considering the gap between symbol-based representations and LLMs, our intuition is converting symbols into corresponding language-based expressions. This conversion serves as a bridge, providing LLMs with more friendly and comprehensible information. In this paper, we propose symbol-to-language (S2L), a tuning-free method that leverages LLMs to better solve symbol-related problems. The S2L method is both simple and feasible, with its core focus on uncovering language-based expressions that are equivalent or approximate to symbols. In particular, S2L first converts the symbols involved in the problems into their language-based representations, which can either be implemented by prompting the LLMs themselves or by leveraging external cost-friendly tools, such as rules, translators, dictionaries, *etc*. Then these language-based representations are integrated into the original questions through direct substitution or concatenation, providing valuable contextual information to assist LLMs in solving symbol-related problems.

We conduct experiments across eight symbol-related problems, encompassing inductive abstract reasoning over numerical sequences, Dyck language involving strings of brackets, chemical property predictions based on molecular formulas, emotion analysis of emojis, question answering on structured tabular data, and stance and sentiment analysis in social media. We employ both API-based and open-source LLMs including GPT-4 (OpenAI, 2023b), ChatGPT (OpenAI, 2022), and OpenChat (Wang et al., 2023a) to validate the generalization of S2L method. Experimental results demonstrate that the S2L leads to significant and consistent improvements under zero-shot or zero-shot-CoT settings, ranging from symbol-only reasoning to conventional natural language processing tasks involving symbols. These results underscore the effectiveness of leveraging language-based representations in better addressing symbol-related problems, thereby expanding the potential applicability of LLMs in a broader range of scenarios.

Symbol	Instance	Task	Example
Sequence of Numbers	1,0,0	Abstract Reasoning (§ 4.1)	1,0,0 → 0,0,1; 5,0,0 → ?
String of Brackets	[( )]	Dyck Language (§ 4.2)	([] →); {(<>) → }; {}[→ ?
Molecular Formula	CCCO	Property Prediction (§ 4.3)	CCCO (*Toxicity: Yes or No*?)
Emoji (Unicode)	😭 (U+1F62D)	Emotion Analysis (§ 4.4)	😭 (*Anger*? *Fear*? *Joy*? *Sadness*?)
Table Delimiter	|,\n	Question Answering (§ 4.5) Fact Verification (§ 4.5)	rank|name|wins\n1|jack|3\n (*Statement: True or False*?)
Abbreviation	LMAO	Stance Detection (§ 4.6) Sentiment Classification (§ 4.6)	Imagine being that bold LMAO (*Text: Positive or Negative*?)

Table 1: Symbols, instances, and examples across eight different types of tasks in our experiments, varying from symbol-only inductive abstract reasoning to sentiment classification in social media.

2 Related Work

Reasoning on Symbol-Related Problems. Various studies have explored the capabilities of LLMs in symbol-based understanding and reasoning. Wang et al. (2023b) suggest that LLMs can improve the resolution of abstract reasoning tasks based on the strong ability to generate executable codes. Qiu et al. (2023) assess LLMs on symbol-based inductive reasoning tasks, uncovering a range of counter-intuitive behaviors. Gendron et al. (2023) demonstrate that LLMs possess a constrained capacity for abstract reasoning compared with other natural language tasks. These works indicate that there is still room for improvement in the reasoning ability of LLMs for symbol-related problems. Our method tries to bridge the gap between symbol understanding and language models by replenishing language-based representation, eliciting the LLMs' capability of solving symbol-related problems without affecting their general abilities.

Reasoning by Chain-of-Thought Prompting. Chain-of-Thought style prompting (Wei et al., 2022; Kojima et al., 2022; Chen et al., 2022; Besta et al., 2023; Yao et al., 2023; Zhang et al., 2023b) have become integral in augmenting the reasoning capabilities of LLMs. Cheng et al. (2023) and Tang et al. (2023b) propose to generate readable explanations for solving commonsense and program translation problems, respectively. Deng et al. (2023) introduce rephrase-and-respond to tackle potentially *ambiguous questions* by using self-rephrased questions, which shares similarities with our symbol-to-language method in *rephrasing*. However, we aim to address *symbol-related problems*, rephrasing symbols into their natural language equivalents to allow LLMs to engage with more accessible language-based information for reasoning.

Reasoning with Symbolic Methods. There is a series of studies on the integration of symbolic methods to solve general reasoning tasks. Wei et al. (2023) propose symbol-tuning to fine-tune LLMs using substituted labels, aiming to bolster their in-context learning abilities. Hu et al. (2023) propose chain-of-symbol to solve planning-based tasks. Wang et al. (2023c) introduce meta-reasoning as a means to construct generic symbolic representations for reasoning tasks. Fang et al. (2024) design symbolic module in LLM agent for solving text-based games. Trinh et al. (2024) combine LLMs and symbolic engines to solve geometry problems. As a comparison, our work focuses on symbol-related tasks and proposes eliciting the powerful comprehension and reasoning abilities of LLMs at the *language* level to help solve the problems.

3 Symbol-to-Language

We consider a range of symbol-related problems depicted in Table 1, in which the interpretation of symbol meanings is crucial for accomplishing the associated tasks. We formalize this type of problem as a question q_s that incorporates a set of symbols $s = \{s_1, s_2, ..., s_m\}$ and try using current LLMs to solve q_s. Vanilla zero-shot (Brown et al., 2020) and zero-shot-CoT (Kojima et al., 2022) method directly solve the original problem and the responses of these two methods by LLM $\mathcal{M}$ can be written as:

$$\mathcal{R}_{\text{zs}} = \mathcal{M}(q_s); \quad \mathcal{R}_{\text{zsc}} = \mathcal{M}(q_s \oplus p), \tag{1}$$

where zs and zsc indicate zero-shot and zero-shot-CoT, respectively. p is a prompt like "*Let's think step by step*", and $\oplus$ is the concatenation operation.

The above methods directly tackle the problem with symbol-based representations. Instead of designing external prompt p, we focus on the question q_s and propose utilizing language-based representation to better leverage the LLMs' strong capabilities of natural language for solving symbol-related problems. Specifically, the S2L framework first converts the symbols s_i $(i = 1, ..., m)$ to its corresponding plain text l_i with a conversion operation f, which can be implemented by either LLMs themselves or external tools. Then the S2L framework incorporates the converted language-based representation l_i into two alternative questions q_l or $q_{s\oplus l}$ as the input for LLMs to generate answers. The details are discussed as follows.

3.1 Symbol-to-Language Conversion

Conversion with LLMs. We first employ the LLMs $\mathcal{M}$ to convert symbols s_i to their corresponding natural language descriptions l_i^{LLM} via zero-shot prompting, as expressed by:

$$l_i^{\text{LLM}} = f_{\text{LLM}} \circ s_i = \mathcal{M}(p_{\text{s2l}} \oplus s_i), \tag{2}$$

where $f_{\text{LLM}} \circ s_i$ denotes converting s_i using the LLM $\mathcal{M}$, p_{s2l} is the task-specific prompt facilitating the S2L conversion. For instance, when the symbol-related question q_s is about property prediction and s_i is a molecular formula, p_{s2l} could be "*What does the following molecular formula represent?*".

Conversion with Tools. Considering there exist some constructed "symbol-language" pairs, we further propose using external tools for conversion, which can take on several forms. *Rule-based codes*, for example, can convert s_i = "`rank|nation\n1|SWE`" into l_i^{rule} = "rank: 1; nation: SWE" according to the table delimiters "`|`" and "`\n`". *Translators* can transform molecular formulas into their formal names, such as converting s_i = "`CCCO`" into $l_i^{\text{translator}}$ = "Propionylo". *Unicode dictionaries* can provide descriptions of emojis, like converting s_i = "`U+1F62D`" into l_i^{dict} = "crying face". Despite having some limitations in terms of usage scenarios, conversion with tools offers two primary advantages: 1) it circumvents the costs associated with using LLMs; 2) it provides verified language-based information, which can help reduce potential errors in descriptions generated by LLMs.

3.2 Utilizing Language-Based Representations

We propose two alternative ways that incorporate the language-based representation l_i into the final input.

Direct Substitution. The first way of utilization is to directly substitute the symbol-based representation s_i with language-based representation l_i. To some extent, l_i can be regarded as the language-based equivalent of s_i. Thus we can use them to replace the symbol-based representations for both the question and ground-truth label. The response by using S2L can be written as:

$$\mathcal{R}_{\text{s2l}} = \mathcal{M}(q_l), \quad q_l = q_{f\circ s} = q_{f\circ\{s_1,...,s_m\}} = q_{f\circ s_1,...,f\circ s_m}. \tag{3}$$

Concatenation. However, in some other tasks, the generated l_i by LLMs may not always be a perfect substitution or convey complete information of s_i. This can occur for two reasons: 1) l_i might be incorrect due to the undesired output formats, misleading content, or noisy context; and 2) l_i could lose some information during S2L conversion. For instance, the ground truth might be a span-based abbreviation for table understanding (*e.g.*, the "`SWE`" in table "`rank|nation\n1|SWE`"), meaning that the converted l_i as a full name (*e.g.*, "Sweden") may not exactly match the final answer. Therefore, the second method uses both the original symbol-based representation s_i and the language-based representation l_i as the combined input. This approach allows the LLMs $\mathcal{M}$ to reason on questions that include rich contextual information from two distinct perspectives:

$$\mathcal{R}_{\text{s2l}} = \mathcal{M}(q_{s\oplus l}), \quad q_{s\oplus l} = q_{s_1\oplus l_1,...,s_m\oplus l_m} = q_{s_1\oplus\{f\circ s_1\},...,s_m\oplus\{f\circ s_m\}}. \tag{4}$$

4 Experiments

To assess the performance of the S2L framework, we conduct six categories of symbol-related problems with eight specific tasks as shown in Table 1, varying from symbol-only inductive abstract reasoning to traditional sentiment analysis in social media. As for LLMs, we evaluate both API-based and open-source models, including GPT-4 (OpenAI, 2023b), ChatGPT (OpenAI, 2022), and

Model	Move-1p		Move-2p		Move-3p		Avg.
	$n=3$	$n=4$	$n=3$	$n=4$	$n=3$	$n=4$	
				(*Zero-Shot*)			
`gpt-4`	93.3	90.0	48.3	46.7	35.0	45.0	59.7
+S2L w/ model	90.0 (−3.3)	91.7 (+1.7)	48.3 (–)	56.7 (+10.0)	38.3 (+3.3)	48.3 (+3.3)	**62.2** (+2.5)
+S2L w/ rule	96.7 (+3.4)	100.0 (+10.0)	88.3 (+40.0)	96.7 (+50.0)	48.3 (+13.3)	60.0 (+15.0)	**81.6** (+21.9)
`gpt-3.5-turbo`	68.3	71.7	11.7	25.0	23.3	26.7	37.8
+S2L w/ model	80.0 (+11.7)	75.0 (+3.3)	31.6 (+19.9)	26.6 (+1.6)	33.3 (+10.0)	31.7 (+5.0)	**46.4** (+8.6)
+S2L w/ rule	71.6 (+3.3)	88.3 (+16.6)	25.0 (+13.3)	31.7 (+6.7)	25.0 (+1.7)	26.7 (–)	**44.7** (+6.9)
`openchat-3.5-7b`	61.7	71.7	15.0	21.6	11.7	11.7	32.2
+S2L w/ model	68.3 (+6.6)	78.3 (+6.6)	23.3 (+8.3)	25.0 (+3.4)	25.0 (+13.3)	21.7 (+10.0)	**40.3** (+8.1)
+S2L w/ rule	63.3 (+1.6)	75.0 (+3.3)	16.6 (+1.6)	18.3 (−3.3)	15.0 (+3.3)	11.7 (–)	**33.3** (+1.1)

Table 2: Results for three subtasks on 1D-ARC. n indicates the number of given input-output pairs for finding the patterns. The values in parentheses indicate the difference between the baseline and our results. w/ model denotes conversion with LLMs, and w/ rule denotes conversion with manually designed rules using codes.

OpenChat-7b (Wang et al., 2023a). To ensure the reproducibility of the responses generated by LLMs, we set the decoding temperature as 0. For each task, we show the case of S2L conversion in the appendix B.

4.1 Abstract Reasoning

Abstract reasoning (Webb et al., 2023; Gendron et al., 2023; Wang et al., 2023b) is a type of task that involves summarizing patterns from limited observations. We conduct experiments on subtasks from the 1D-ARC benchmark proposed by Xu et al. (2023c), which is a simplified version of the abstract reasoning corpus proposed by Chollet (2019). The 1D-ARC comprises various 1D object-based visual problems, as depicted in Figure 2(a). To enable LLMs to process these problems, visual information is transformed into a symbol-based representation with sequences of numbers, as illustrated in Figure 2(b).

Symbol-to-Language. As for the conversion methods, we apply both LLMs and rule-based codes to transform the sequence of numbers into natural language descriptions. We find that LLMs describe the sequence similarly to humans coincidentally, which employs *merging* or *counting* when describing number sequences. Thus, for the conversion with rules, we implement code as a rule for merging and counting numbers in a sequence. The specific prompts for LLMs and the rule-based codes are presented in Figure 2 (c.1) and Figure 2 (d.1). Due to the potential loss of information in the generated description l_i^{LLM}, we attach the language-based representation to each original sequence of numbers to generate answers, as shown in Figure 2 (c.2). On the contrary, the outputs by rule-based codes l_i^{rule} are equal to the original sequence of numbers, thus we directly replace them to get responses, as shown in Figure 2 (d.2).

Settings and Results. We use the Move-1p, Move-2p, and Move-3p tasks (move 1, 2, and 3 pixels forward, respectively) from 1D-ARC, where each task contains 50 problems that involve fixed $n = 3$ input-output pairs. We collect and combine the given input-output pairs for each task, resulting in 60 problems for each task with $n = 3$ or $n = 4$ input-output pairs. The experimental results are presented in Table 2. GPT-4 achieves an accuracy above 90.0% on the Move-1p task. However, the performance drops rapidly to $30\sim50\%$ in the Move-2p and Move-3p tasks, demonstrating that the model struggles to reason on sequences with slightly more complex patterns. This phenomenon is even more evident in ChatGPT and OpenChat, where the overall accuracy is much lower. Upon employing conversion with LLMs (*i.e.*, S2L w/ model), the results improve by $2.5\sim8.6\%$, suggesting the positive impact of additional language-based information. By using conversion with rule-based codes, GPT-4 gets 100% accuracy given $n = 4$ input-output pairs, and the performance on Move-2p and Move-3p improves substantially, with 96.7% and 60.0% accuracy, respectively. Moreover, we observe varying degrees of improvement across models, indicating their differing abilities to understand the additional language-based representation.

Model	Dyck Language					Avg.
	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	
			(*Zero-Shot*)			
`gpt-4`	60.0	82.2	86.6	91.4	92.2	82.5
+S2L w/ model	77.6 (+17.6)	90.0 (+7.8)	95.6 (+9.0)	98.2 (+6.8)	98.6 (+6.4)	**92.0** (+9.5)
`gpt-3.5-turbo`	65.0	77.0	78.0	80.8	78.2	75.8
+S2L w/ model	69.6 (+4.6)	82.8 (+5.8)	88.2 (+10.2)	93.2 (+12.4)	94.0 (+15.8)	**85.6** (+9.8)
`openchat-3.5-7b`	8.6	3.6	4.2	2.2	7.8	5.3
+S2L w/ model	21.6 (+13.0)	22.0 (+18.4)	32.0 (+27.8)	43.8 (+41.6)	59.0 (+51.2)	**35.7** (+30.4)

Table 3: Results for Dyck language task. n indicates the number of given input-output pairs for finding the patterns. w/ model denotes conversion with LLMs.

4.2 Dyck Language

Dyck language is a subtask in BigBench (Srivastava et al., 2022), aiming to predict the closing parentheses of a given sequence. To evaluate the inductive reasoning ability over *symbols*, we do not prompt LLMs to "complete parentheses" (*i.e.*, prompting LLMs to output the remaining parentheses). Instead, following the settings of the ARC benchmark, we *only* give n input-output pairs and let LLMs to *deduce* the output according to the patterns, as shown in Figure 3(a).

Symbol-to-Language. The symbols in this task include totally eight different brackets ("`[]{}()<>`"). We consider converting each symbol s_i to natural language description l_i^{LLM} via prompting. Thus we can translate the problem in language-based representations, as shown in Figure 3(c).

Remark. The responses during S2L conversion may not be the same among different LLMs. An interesting phenomenon is that GPT-4 recognizes the symbols "<" and ">" as "less than" and "great than", instead of "open angle bracket" and "close angle bracket". However, we do *not* consider this to be a mistake and do *not* intend to modify the prompts to "correct" the results (*e.g.*, provide some cues indicating these are different types of parentheses). In contrast, we let the LLMs convert the symbols according to their understanding and deduce the final answer through the generated representations by themselves.

Settings and Results. We first set the number of examples as $n = 5$ and randomly choose six input-output pairs (five of them as the examples and the rest as the target) from the entire dataset, we repeat 500 times and evaluate the overall accuracy among these 500 questions. Then we gradually remove the last input-output pair to test the ability with fewer examples (*i.e.*, $n = 4, 3, 2, 1$).

The results are shown in Table 3. For GPT-4 and ChatGPT, the performance ranges from $60.0\sim92.2\%$ and $65.0\sim78.2\%$, respectively. The accuracy further improves with $+9.5\%$ and $+9.8\%$ by using S2L. For OpenChat, the performance is extremely low with below 10% accuracy and S2L improves the performance by a large margin with $+30.4\%$ on average.

4.3 Property Prediction

We use ChemLLMBench (Guo et al., 2023) to predict the chemical property given a molecule's SMILES (simplified molecular-input line-entry system) string. Three datasets are used, including BACE (bindings results for a set of inhibitors of human beta-secretase), BBBP (penetration/non-penetration to the brain-blood barrier), and Tox21 (toxicity of compounds).

Symbol-to-Language. We use a unified prompt to transfer each SMILES to its language-based representation l_i^{LLM} in all three datasets, as shown in Figure 4(a). Instead of using LLMs, we further propose S2L with STOUT V2.0 (Rajan et al., 2021), a translator offering the IUPAC (A universally accepted naming scheme established by the International Union of Pure and Applied Chemistry) name $l_i^{\text{translator}}$ of a given SMILES, as shown in Figure 4(b). Finally, we append the obtained information to each SMILES notation as language-enhanced input for LLMs.

Settings and Results. Following Guo et al. (2023), we randomly sample 500 instances from the full test set and report the averaged results over repeated five times, the results are shown in Table 4. The

Model	*Zero-Shot*			*Zero-Shot-CoT*			Avg.
	BACE	**BBBP**	**Tox21**	**BACE**	**BBBP**	**Tox21**	
`gpt-4`	48.2	40.2	47.2	50.4	36.8	37.4	43.4
+S2L w/ model	53.0 (+5.2)	53.0 (+12.8)	48.6 (+1.4)	55.6 (+5.2)	54.2 (+17.4)	44.0 (+6.6)	**51.4** (+8.0)
+S2L w/ translator	48.4 (+0.2)	58.8 (+18.6)	49.0 (+1.8)	51.2 (+0.8)	64.0 (+27.2)	45.2 (+7.8)	**52.8** (+9.4)
`gpt-3.5-turbo`	52.4	24.6	34.2	44.2	32.0	35.8	37.2
+S2L w/ model	53.0 (+0.6)	35.0 (+10.4)	34.8 (+0.6)	48.2 (+4.0)	35.8 (+3.8)	38.2 (+2.4)	**40.8** (+3.6)
+S2L w/ translator	54.8 (+2.4)	53.8 (+29.2)	51.0 (+16.8)	48.6 (+4.4)	41.4 (+9.4)	41.8 (+6.0)	**48.6** (+11.4)
`openchat-3.5-7b`	48.2	46.8	62.6	49.6	46.2	62.2	52.6
+S2L w/ model	48.8 (+0.6)	56.2 (+9.4)	65.0 (+2.4)	47.8 (−1.8)	48.4 (+2.2)	60.4 (−1.8)	**54.4** (+1.8)
+S2L w/ translator	55.8 (+7.6)	59.2 (+12.4)	67.8 (+5.2)	52.2 (+2.6)	65.2 (+19.0)	61.4 (−0.8)	**60.3** (+7.7)

Table 4: Results for three property prediction tasks on ChemLLMBench by using zero-shot inference, zero-shot-CoT inference, and our symbol-to-language. w/ model denotes conversion with LLMs, and w/ translator denotes conversion with an external translator.

Model	**EmoTag1200** (Pearson correlation r)								Avg.
	Anger	Anticipate	Disgust	Fear	Joy	Sadness	Surprise	Trust	
					(*Zero-Shot*)				
`gpt-4`	0.855	0.290	0.782	0.809	0.887	0.903	0.531	0.731	0.724
+S2L w/ model	0.878	0.308	0.792	0.819	0.897	0.922	0.562	0.766	**0.743** (+0.019)
+S2L w/ dict	0.856	0.345	0.753	0.813	0.894	0.926	0.616	0.760	**0.745** (+0.021)
`gpt-3.5-turbo`	0.710	0.214	0.334	0.589	0.695	0.790	0.192	0.560	0.510
+S2L w/ model	0.700	0.156	0.401	0.629	0.786	0.850	0.483	0.630	**0.578** (+0.068)
+S2L w/ dict	0.757	0.269	0.487	0.696	0.796	0.854	0.285	0.700	**0.605** (+0.095)
`openchat-3.5-7b`	0.413	0.030	0.438	0.264	0.161	0.232	0.221	−0.086	0.209
+S2L w/ model	0.465	0.066	0.535	0.639	0.355	0.583	0.010	−0.118	**0.317** (+0.108)
+S2L w/ dict	0.572	0.002	0.587	0.648	0.377	0.587	0.023	−0.078	**0.339** (+0.130)
					(*Zero-Shot-CoT*)				
`gpt-4`	0.854	0.205	0.723	0.810	0.889	0.917	0.507	0.744	0.706
+S2L w/ model	0.854	0.330	0.705	0.814	0.889	0.923	0.632	0.741	**0.736** (+0.030)
+S2L w/ dict	0.865	0.337	0.661	0.825	0.889	0.930	0.627	0.730	**0.733** (+0.027)
`gpt-3.5-turbo`	0.559	0.013	0.051	0.156	0.264	0.145	−0.084	0.060	0.146
+S2L w/ model	0.702	0.152	0.393	0.631	0.785	0.845	0.485	0.633	**0.578** (+0.432)
+S2L w/ dict	0.734	0.206	0.505	0.664	0.799	0.853	0.354	0.543	**0.582** (+0.436)
`openchat-3.5-7b`	0.455	−0.014	0.277	0.348	0.454	0.741	0.163	0.012	0.305
+S2L w/ model	0.632	0.172	0.438	0.532	0.669	0.692	0.344	0.098	**0.447** (+0.142)
+S2L w/ dict	0.564	−0.050	0.381	0.613	0.684	0.581	0.194	0.064	**0.380** (+0.075)

Table 5: Results for emotion analysis of emojis by using zero-shot inference, zero-shot-CoT inference, and our symbol-to-language. The numbers indicate the Pearson correlation with ratings by humans. w/ model denotes conversion with LLMs, and w/ rule denotes conversion with the Unicode dictionary.

overall zero-shot performance of GPT-4 and ChatGPT is relatively low, showing the difficulty for LLMs to understand the molecular formula and their chemical property. Zero-shot-CoT does not lead to stable improvement, showing that a single prompt "*Let's think step by step*" is not helpful for these types of problems. Our method generally improves the performance to varying degrees (except for slight decreases using OpenChat and zero-shot-CoT settings). For example, the improvement is large for the BBBP dataset ($+9.4\sim29.2\%$ across models) while it becomes relatively low for the BACE dataset ($+0.2\sim7.6\%$ across models). Overall, the results show that S2L is effective by providing language-based information that aids in chemical property prediction tasks.

4.4 Emotion Analysis of Emojis

We use EmoTag1200 (Shoeb & de Melo, 2020) for emotion analysis of the emojis. Specifically, 150 most frequently used emojis are selected and the task is to score each of them from $0\sim1$ based on eight basic emotions, including *anger*, *anticipation*, *disgust*, *fear*, *joy*, *sadness*, *surprise*, and *trust*.

Symbol-to-Language. To understand emojis with language-based information, we get the description l_i^{LLM} by prompting LLMs or directly obtaining the names l_i^{dict} from the Unicode dictionary, as shown in Figure 5.

Settings and Results. We use the Pearson correlation coefficient between the predictions and ratings by humans for evaluation, and the results are shown in Table 5. GPT-4 gives a relatively high

Model	*Zero-Shot*			*Zero-Shot-CoT*			AVG.
	TableQA (F1)	**TableQA** (EM)	**TabFact** (Acc.)	**TableQA** (F1)	**TableQA** (EM)	**TabFact** (Acc.)	
`gpt-4`	82.0	79.8	93.6	80.7	76.8	93.2	84.4
+S2L w/ model	84.6 (+2.6)	82.0 (+2.2)	95.6 (+2.0)	82.9 (+2.2)	80.2 (+3.4)	94.6 (+1.4)	**86.7** (+2.3)
+S2L w/ rule	86.5 (+4.5)	84.2 (+4.4)	95.8 (+2.2)	84.2 (+3.5)	80.6 (+3.8)	96.4 (+3.2)	**88.0** (+3.6)
`gpt-3.5-turbo`	66.4	63.0	82.0	73.4	69.8	77.0	71.9
+S2L w/ model	69.0 (+2.6)	66.0 (+3.0)	84.6 (+2.6)	73.6 (+0.2)	71.2 (+1.4)	83.0 (+6.0)	**74.6** (+2.7)
+S2L w/ rule	68.5 (+2.1)	64.8 (+1.8)	86.2 (+4.2)	77.0 (+3.6)	72.8 (+3.0)	83.0 (+6.0)	**75.4** (+3.5)
`openchat-3.5-7b`	61.7	58.2	79.0	60.8	57.0	83.8	66.8
+S2L w/ model	64.1 (+2.4)	59.0 (+0.8)	81.0 (+2.0)	64.2 (+3.4)	60.4 (+3.4)	83.2 (−0.6)	**68.6** (+1.8)
+S2L w/ rule	62.1 (+0.4)	58.8 (+0.6)	83.0 (+4.0)	66.1 (+5.3)	63.0 (+6.0)	84.6 (+0.8)	**69.6** (+2.8)

Table 6: Results for table question answering and table fact verification tasks using zero-shot inference, zero-shot-CoT inference, and our symbol-to-language. w/ model denotes conversion with LLMs, and w/ rule denotes conversion with manually designed rules by codes for aligning contents from tables.

Model	*Zero-Shot*			*Zero-Shot-CoT*			AVG.
	P-Stance (Acc.)	**P-Stance** (F1)	**Sentiment** (Acc.)	**P-Stance** (Acc.)	**P-Stance** (F1)	**Sentiment** (Acc.)	
`gpt-4`	86.2	86.7	89.4	83.8	84.0	86.1	86.0
+S2L w/ model	87.1 (+0.9)	88.2 (+1.5)	90.3 (+0.9)	86.6 (+2.8)	87.5 (+3.5)	87.2 (+1.1)	**87.8** (+1.8)
`gpt-3.5-turbo`	65.3	68.6	83.7	61.5	60.8	76.5	69.4
+S2L w/ model	71.0 (+5.7)	71.5 (+2.9)	89.9 (+6.2)	64.7 (+3.2)	63.5 (+2.7)	78.4 (+1.9)	**73.2** (+3.8)
`openchat-3.5-7b`	70.9	66.7	89.1	72.2	69.1	84.8	75.5
+S2L w/ model	71.4 (+0.5)	67.5 (+0.8)	89.0 (−0.1)	77.1 (+4.9)	75.8 (+6.7)	82.1 (−2.7)	**77.2** (+1.7)

Table 7: Results for stance detection and sentiment classification in social media using zero-shot inference, zero-shot-CoT inference, and our symbol-to-language.

correlation coefficient of 0.724. However, ChatGPT and OpenChat perform badly with only 0.510 and 0.209 correlation coefficients, respectively. By using zero-shot-CoT, the performance even drops for GPT-4 and ChatGPT, showing the limitations in LLMs' emoji understanding ability. When using S2L with either model or dictionary, the performance improves to different degrees, showing that the language information can also help understand emoji-based symbols.

4.5 TABLE UNDERSTANDING

For structured data, we follow Chen (2023) to evaluate the capability of LLMs on table reasoning. Specifically, we use WikiTableQuestions (Pasupat & Liang, 2015) for evaluating table-based question answering, which consists of complex questions based on Wikipedia tables. We also use TabFact (Chen et al., 2020) for fact verification, which consists of claims annotated by the crowd workers based on tables.

Symbol-to-Language. As shown in Figure 6, S2L describes every table in plain text l_i^{LLM} by prompting. Alternatively, we can get the representation l_i^{rule} by using simple rule-based codes to align the content with the table header according to the delimiters " | " row by row. Then we append the external natural language information to the original symbol-based representation for each question.

Settings and Results. For each task, we evaluated 500 pairs of tables and questions and the results are shown in Table 6. The overall performance for different models is relatively high compared with previous symbol-only tasks, for example, GPT-4 gives around 79.8% exact match score and 93.6% accuracy for question answering and fact verification, respectively. Nevertheless, S2L with model can consistently bring $+1.8\sim2.3\%$ improvements, showing that external natural language information is effective. We find that S2L with rule further leads to $+2.8\sim3.6\%$ improvements, indicating that simple cues with aligned information between content and header for each row can already make a positive impact on table understanding.

4.6 TWEET ANALYSIS

We analyze the text in social media and use the TweetSentimentExtraction dataset from Massive Text Embedding Benchmark (Muennighoff et al., 2023) for sentiment classification. Additionally, we follow Zhang et al. (2023a) by using the P-Stance (Li et al., 2021) dataset for stance detection.

Symbol-to-Language. There is a plethora of non-natural language expressions on Tweet, including abbreviations (*e.g.*, LOL: "Laughing Out Loud"), slang (*e.g.*, FTW: "For the Win"), hashtags (*e.g.*, #Trump), emojis (*e.g.*, 🐝), *etc*. We convert the entire tweet to plain text l_i^{LLM} via prompting, as shown in Figure 7, and then we use it as external input for each question.

Settings and Results. For sentiment classification, we predict sentiment polarity from either *positive* or *negative* in a total of 2,104 tweets. For stance detection, we evaluate the attitude towards "Donald Trump" from either *favor* or *against* in 777 test tweets. The results are shown in Table 7. Similarly, the zero-shot-CoT method sometimes causes a decrease in accuracy. Specifically, we find that simply adding the prompt "*Let's think step by step*" leads to more *neutral* responses, even though there are all texts with 2-way labels (*i.e.*, *positive*/*negative* and *favor*/*against*). In general, our S2L can improve the performance under both zero-shot and zero-shot-CoT settings for GPT-4 and ChatGPT models, except that there is a slight decrease using the OpenChat model for sentiment classification.

5 DISCUSSIONS

From the experimental results, our proposed symbol-to-language shows a significant improvement in tasks such as abstract reasoning, Dyck language, and chemical property prediction. It also achieves slight gains in a range of NLP-related tasks. Additionally, compared to zero-shot-CoT, our approach exhibits more stable and effective improvements in solving symbol-related problems.

We further analyze and discuss both the advantages (*i.e.*, *how does it take effect?*) and limitations (shown in the appendix C, *i.e.*, *in which scenarios does it still not have an impact?*) of the proposed symbol-to-language method.

Directly solving symbol-related problems can be difficult for LLMs due to various reasons. We show that S2L can offer some distinct types of language-based information that are important for better solving the mentioned tasks:

Precise Information. As shown in the 1D-ARC tasks, the language-based representations by using rules can reflect the information of sequences precisely, which can compensate for the limitations of language models such as counting numbers, thus enhancing their ability to summarize patterns and deduce the results.

Co-occurrence Information. S2L conversion offers co-occurrence information between contexts and task-level labels. For example, the descriptions of emojis (*e.g.*, "angry face") and emotional dimensions (*e.g.*, "anger") for emoji analysis, and the plain text of abbreviations (*e.g.*, "laughing my ass off") and sentiment polarities (*e.g.*, "positive") for sentiment classification. These language-level co-occurrences can offer complementary information for symbol-based problems.

Alignment Information. Language-based representations can also offer aligned information, which is difficult to extract directly from symbol-based representations. For example, the aligned relationship between "open" and "close" brackets for the Dyck language task, and the alignment between table contents and header. These explicitly aligned contexts can somewhat help LLMs reasoning on complicated symbol-based tasks.

6 CONCLUSION

We propose symbol-to-language, a tuning-free method that converts symbol-based to language-based representation for solving a series of symbol-related problems using large language models. Experiments on GPT-4, ChatGPT, and OpenChat across eight tasks show that symbol-to-language can significantly improve the performance on tasks such as abstract reasoning, Dyck language, and chemical property prediction, *etc*. We hope to further harness the power of language, leverage the advantages of language-based representations, and explore the untapped potential of large language models to play roles in more scenarios.

ACKNOWLEDGEMENTS

This work is supported by the National Key R&D Program of China (2022ZD0160502) and the National Natural Science Foundation of China (No. 61925601, 62276152, 62306161).

REFERENCES

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*, 2023.

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

Wenhu Chen. Large language models are few(1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL 2023*, pp. 1120–1130, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. Tabfact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations*, 2020.

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

Sijie Cheng, Zhiyong Wu, Jiangjie Chen, Zhixing Li, Yang Liu, and Lingpeng Kong. Unsupervised explanation generation via correct instantiations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 12700–12708, 2023.

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

Yihe Deng, Weitong Zhang, Zixiang Chen, and Quanquan Gu. Rephrase and respond: Let large language models ask better questions for themselves. *arXiv preprint arXiv:2311.04205*, 2023.

Meng Fang, Shilong Deng, Yudi Zhang, Zijing Shi, Ling Chen, Mykola Pechenizkiy, and Jun Wang. Large language models are neurosymbolic reasoners. *arXiv preprint arXiv:2401.09334*, 2024.

Yin Fang, Xiaozhuan Liang, Ningyu Zhang, Kangwei Liu, Rui Huang, Zhuo Chen, Xiaohui Fan, and Huajun Chen. Mol-instructions: A large-scale biomolecular instruction dataset for large language models. *arXiv preprint arXiv:2306.08018*, 2023.

Gaël Gendron, Qiming Bao, Michael Witbrock, and Gillian Dobbie. Large language models are not abstract reasoners. *arXiv preprint arXiv:2305.19555*, 2023.

Gemini Team Google, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Taicheng Guo, Kehan Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. What indeed can gpt models do in chemistry? a comprehensive benchmark on eight tasks, 2023.

Hanxu Hu, Hongyuan Lu, Huajian Zhang, Wai Lam, and Yue Zhang. Chain-of-symbol prompting elicits planning in large langauge models. *arXiv preprint arXiv:2305.10276*, 2023.

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

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pp. 15696–15707. PMLR, 2023.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22199–22213, 2022.

Yingjie Li, Tiberiu Sosea, Aditya Sawant, Ajith Jayaraman Nair, Diana Inkpen, and Cornelia Caragea. P-stance: A large dataset for stance detection in political domain. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 2355–2365, Online, August 2021. Association for Computational Linguistics.

Melanie Mitchell, Alessandro B Palmarini, and Arseny Moskvichev. Comparing humans, gpt-4, and gpt-4v on abstraction and reasoning tasks. *arXiv preprint arXiv:2311.09247*, 2023.

Arseny Moskvichev, Victor Vikram Odouard, and Melanie Mitchell. The conceptarc benchmark: Evaluating understanding and generalization in the arc domain. *arXiv preprint arXiv:2305.07141*, 2023.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2014–2037, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics.

Setsuo Ohsuga. Bridging the gap between non-symbolic and symbolic processing–how could human being acquire language? *Fundamenta Informaticae*, 75(1-4):385–406, 2007.

OpenAI. 2022. URL https://openai.com/chatgpt.

OpenAI. GPT-4V(ision) system card. 2023a. URL https://openai.com/research/gpt-4v-system-card.

OpenAI. GPT-4 technical report. 2023b. URL https://cdn.openai.com/papers/gpt-4.pdf.

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

Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1470–1480, Beijing, China, July 2015. Association for Computational Linguistics.

Linlu Qiu, Liwei Jiang, Ximing Lu, Melanie Sclar, Valentina Pyatkin, Chandra Bhagavatula, Bailin Wang, Yoon Kim, Yejin Choi, Nouha Dziri, et al. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. *arXiv preprint arXiv:2310.08559*, 2023.

Kohulan Rajan, Achim Zielesny, and Christoph Steinbeck. Stout: Smiles to iupac names using neural machine translation. *Journal of Cheminformatics*, 13(1):1–14, 2021.

Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence*, 4(12):1256–1264, 2022.

Abu Awal Md Shoeb and Gerard de Melo. EmoTag1200: Understanding the association between emojis and emotions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8957–8967, Online, November 2020. Association for Computational Linguistics.

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. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

Xiaojuan Tang, Zilong Zheng, Jiaqi Li, Fanxu Meng, Song-Chun Zhu, Yitao Liang, and Muhan Zhang. Large language models are in-context semantic reasoners rather than symbolic reasoners. *arXiv preprint arXiv:2305.14825*, 2023a.

Zilu Tang, Mayank Agarwal, Alex Shypula, Bailin Wang, Derry Wijaya, Jie Chen, and Yoon Kim. Explain-then-translate: An analysis on improving program translation with self-generated explanations. *arXiv preprint arXiv:2311.07070*, 2023b.

Trieu Trinh, Yuhuai Wu, Quoc Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 2024. doi: 10.1038/s41586-023-06747-5.

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. Openchat: Advancing open-source language models with mixed-quality data. *arXiv preprint arXiv:2309.11235*, 2023a.

Ruocheng Wang, Eric Zelikman, Gabriel Poesia, Yewen Pu, Nick Haber, and Noah D Goodman. Hypothesis search: Inductive reasoning with language models. *arXiv preprint arXiv:2309.05660*, 2023b.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Rationale-augmented ensembles in language models. *arXiv preprint arXiv:2207.00747*, 2022.

Yiming Wang, Zhuosheng Zhang, and Rui Wang. Meta-reasoning: Semantics-symbol deconstruction for large language models. *arXiv preprint arXiv:2306.17820*, 2023c.

Taylor Webb, Keith J Holyoak, and Hongjing Lu. Emergent analogical reasoning in large language models. *Nature Human Behaviour*, 7(9):1526–1541, 2023.

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

Jerry Wei, Le Hou, Andrew Lampinen, Xiangning Chen, Da Huang, Yi Tay, Xinyun Chen, Yifeng Lu, Denny Zhou, Tengyu Ma, et al. Symbol tuning improves in-context learning in language models. *arXiv preprint arXiv:2305.08298*, 2023.

Fangzhi Xu, Zhiyong Wu, Qiushi Sun, Siyu Ren, Fei Yuan, Shuai Yuan, Qika Lin, Yu Qiao, and Jun Liu. Symbol-llm: Towards foundational symbol-centric interface for large language models. *arXiv preprint arXiv:2311.09278*, 2023a.

Yudong Xu, Elias B. Khalil, and Scott Sanner. Graphs, constraints, and search for the abstraction and reasoning corpus. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4):4115–4122, Jun. 2023b.

Yudong Xu, Wenhao Li, Pashootan Vaezipoor, Scott Sanner, and Elias B Khalil. Llms and the abstraction and reasoning corpus: Successes, failures, and the importance of object-based representations. *arXiv preprint arXiv:2305.18354*, 2023c.

Yuan Yang, Siheng Xiong, Ali Payani, Ehsan Shareghi, and Faramarz Fekri. Harnessing the power of large language models for natural language to first-order logic translation. *arXiv preprint arXiv:2305.15541*, 2023.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

Bowen Zhang, Xianghua Fu, Daijun Ding, Hu Huang, Yangyang Li, and Liwen Jing. Investigating chain-of-thought with chatgpt for stance detection on social media. *arXiv preprint arXiv:2304.03087*, 2023a.

Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew Chi-Chih Yao. Cumulative reasoning with large language models. *arXiv preprint arXiv:2308.04371*, 2023b.

A Details of Models

We use the versions of `gpt-4-0613` and `gpt-3.5-turbo-1106` released by OpenAI, and `openchat_3.5` released in https://huggingface.co/openchat/openchat_3.5.

B Applying Symbol-to-Language Conversion to Different Tasks

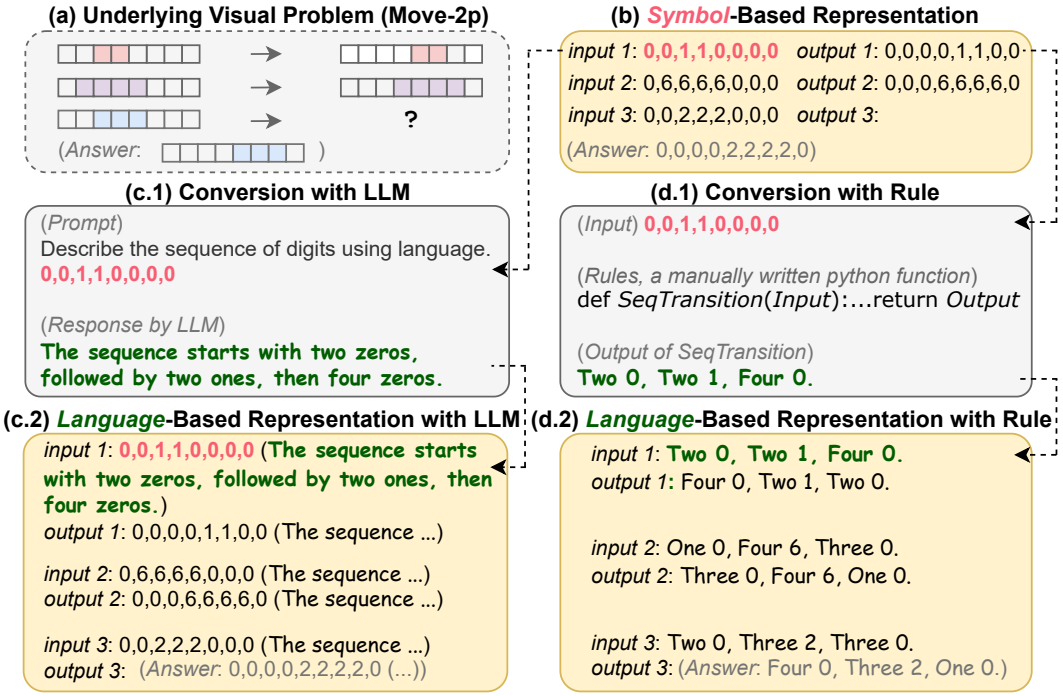

Figure 2: Example of applying symbol-to-language for 1D abstract reasoning task. We convert every sequence to its textual representation by prompting LLMs or using simple rules implemented in code, and then we transform the symbolized problem to language-enhanced or language-only representations.

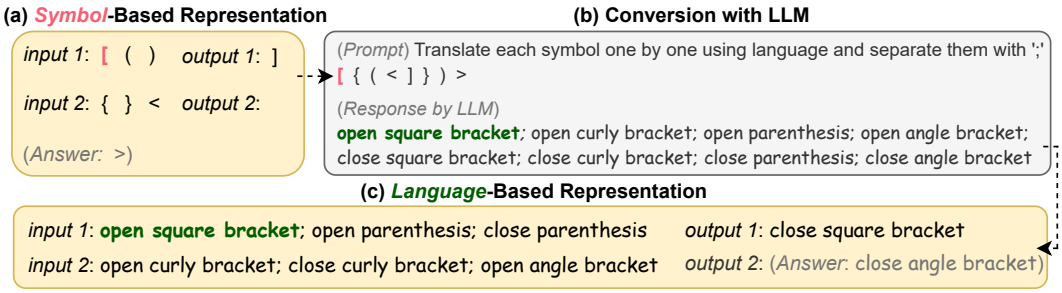

Figure 3: Example of applying symbol-to-language for Dyck language task. We convert every symbol (*e.g.*, "`[`") to its textual description (*e.g.*, "open square bracket") by prompting LLMs, and then transform the symbolized problem into language-based representation for both the question and ground truth.

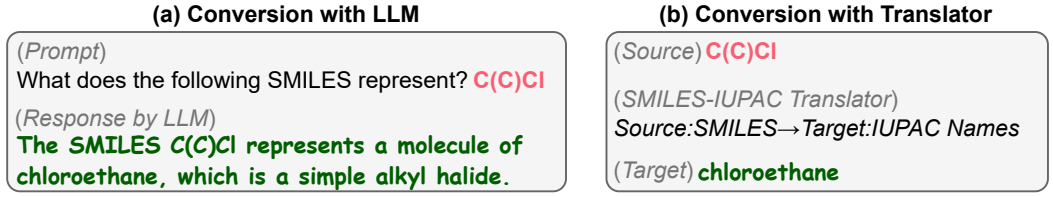

Figure 4: Example of applying symbol-to-language for property prediction. We convert each SMILES to its textual representation by prompting LLMs or using a translator.

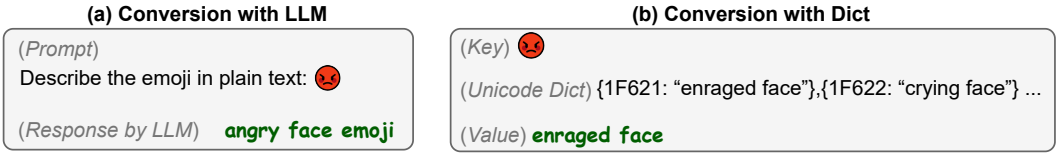

Figure 5: Example of applying symbol-to-language for emotional reranking of emojis. We convert each emoji to its language-based representation by prompting LLMs or using the names from the Unicode dictionary.

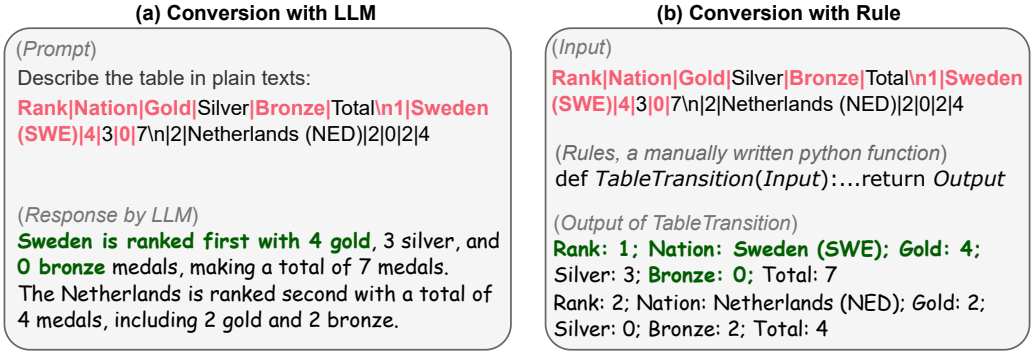

Figure 6: Example of applying symbol-to-language for table question answering. We convert each table to its textual representation by prompting LLMs or using simple rules implemented in codes.

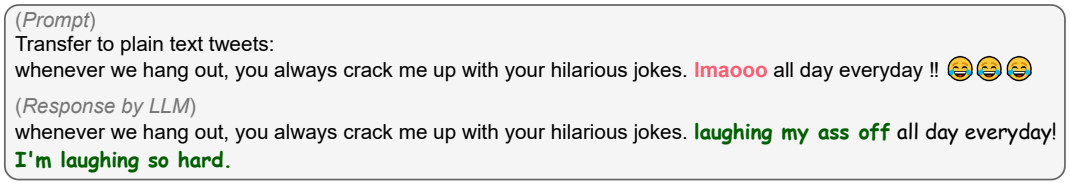

Figure 7: Example of applying symbol-to-language for sentiment classification of tweets. We convert each tweet to plain text by prompting LLMs.

C LIMITATIONS

Although we verified S2L on different models across various tasks, there are still some limitations. First, not all non-natural language representations can be easily converted into natural language. For example, the original 2D visual problems from the ARC dataset (Chollet, 2019) are still difficult to describe in language-based representations, though Xu et al. (2023b) try some "object-based" representations for a tiny portion of the dataset. Second, for tasks that we can not rely on external tools with sufficient prior knowledge, prompting LLMs may generate incorrect descriptions or explanations of the symbols due to hallucinations, which may mislead the understanding of symbols that were originally comprehensible directly.