Leveraging Language-based Representations for Better Solving Symbol-related Problems with Large Language Models

Yile Wang[1], **Sijie Cheng**[2,3], **Zixin Sun**[2], **Peng Li***[3], **Yang Liu***[2,3,4]
[1]College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China
[2]Dept. of Comp. Sci. & Tech., Institute for AI, Tsinghua University, Beijing, China
[3]Institute for AI Industry Research (AIR), Tsinghua University, Beijing, China
[4]Jiangsu Collaborative Innovation Center for Language Competence, Jiangsu, China
wangyile@szu.edu.cn, lipeng@air.tsinghua.edu.cn, liuyang2011@tsinghua.edu.cn

Abstract

Symbols such as numerical sequences, chemical formulas, and table delimiters exist widely, playing important roles in symbol-related tasks such as abstract reasoning, chemical property prediction, and tabular question-answering. Compared to tasks based on natural language expressions, large language models (LLMs) have limitations in understanding and reasoning on symbol-based representations, making it difficult for them to handle symbol-related problems. In this paper, we propose symbol-to-language (S2L), a method that converts symbol-based representations to language-based representations, providing valuable information for language models during reasoning. We found that, for both closed-source and open-source LLMs, the capability to solve symbol-related problems can be largely enhanced by incorporating such language-based representations. For example, by employing S2L for GPT-4, there can be substantial improvements of +21.9% and +9.5% accuracy for 1D-ARC and Dyck language tasks, respectively. There are also consistent improvements in other six general symbol-related tasks such as table understanding and Tweet analysis. We release the GPT logs in https://github.com/THUNLP-MT/symbol2language[1].

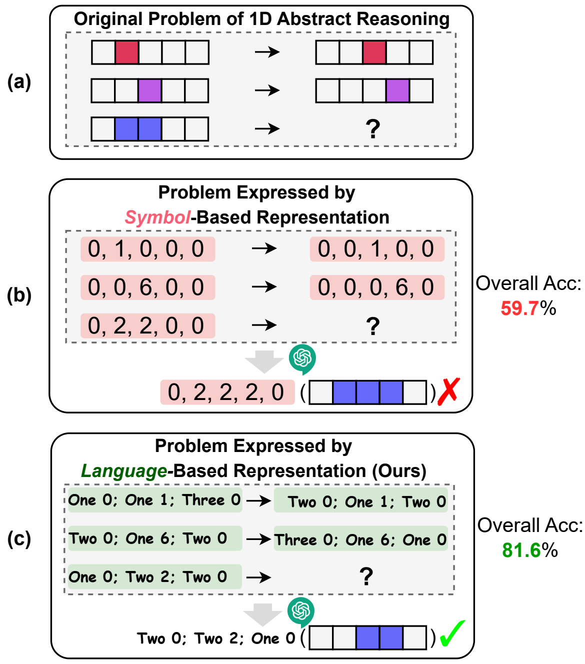

Figure 1: (a) An example of abstract reasoning problem (move 1 pixel forward). (b) The same problem in *symbol*-based representation (Xu et al., 2024c) and the response by GPT-4. (c) The same problem in our *language*-based representation and the response by GPT-4. The overall accuracy has been largely improved.

1 Introduction

Symbols, or more broadly, non-natural language representations such as numerical sequences, brackets, chemical formulas, emojis, table delimiters, and abbreviations can be frequently encountered in the real world. They convey special meanings and play important roles in a variety of tasks such as abstract reasoning (Moskvichev et al., 2023; Xu et al., 2024c), chemical property prediction (Ross et al., 2022; Guo et al., 2023), and tabular question-answering (Chen et al., 2020; Chen, 2023). Furthermore, the understanding and reasoning abilities of symbols are of great value in assessing current artificial intelligence (Chollet, 2019).

Recently, large language models (LLMs; Brown et al., 2020; OpenAI, 2022, 2023a; Jiang et al., 2023; Google et al., 2023) are developing rapidly and can be applied to various tasks. GPT-3 (Brown et al., 2020) has showcased capabilities of zero-shot inference for solving problems directly without demonstrations. Kojima et al. (2022) further propose zero-shot-CoT through additional prompting like "*Let's think step by step*" for solving arithmetic and commonsense reasoning tasks, achieving promising results. However, compared to problems expressed by natural language, the capability of LLMs for solving the aforementioned symbol-

*Corresponding authors.
[1]Work done while Yile Wang was at Tsinghua University.

Proceedings of the 31st International Conference on Computational Linguistics, pages 5544–5557
January 19–24, 2025. ©2025 Association for Computational Linguistics

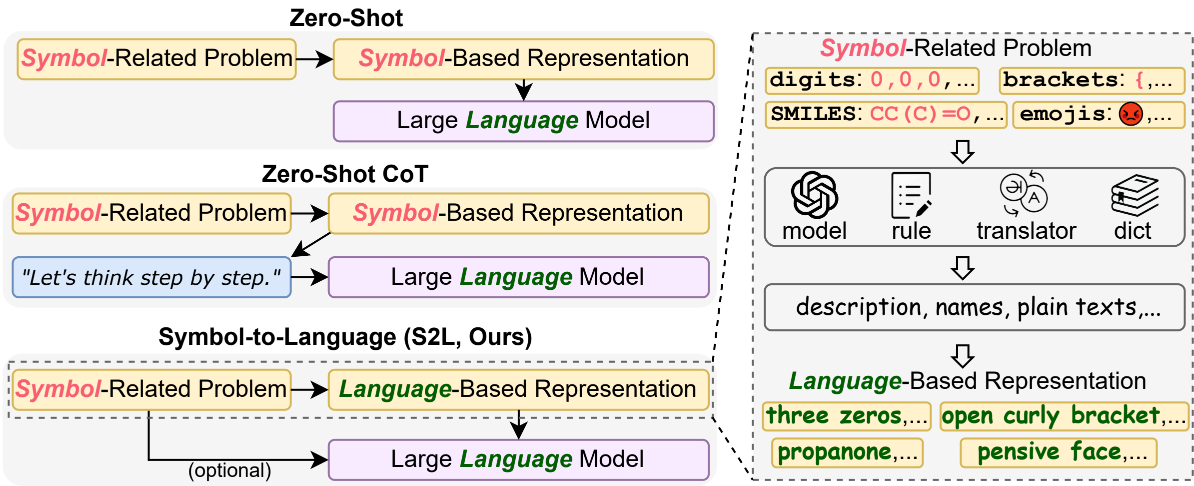

Figure 2: Comparison between zero-shot, zero-shot chain-of-thought (CoT), and our proposed symbol-to-language (S2L) for using LLMs to solve symbol-related problems. The main difference is that we leverage the *language*-based representation beyond the original *symbol*-based representations for LLMs.

related problems are still unsatisfied. Mitchell et al. (2023) reveal that GPT-4 (OpenAI, 2023a) and GPT-4V (OpenAI, 2023b) only achieve 65% and 25% accuracy on abstract reasoning tasks requiring inductive ability through a series of regular numbers, which is significantly lower than the 95% accuracy by humans. Gendron et al. (2024) further demonstrate that existing LLMs exhibit limited performance for symbol-related problems in contrast to natural language tasks.

The reasons for the limited performance can be twofold: First, symbols are significantly underrepresented in the training corpus compared to natural language (Ohsuga, 2007), leading to the understanding gap for low-frequency symbols using language models (Kandpal et al., 2023; Tang et al., 2023). Second, the abstract reasoning task further introduces the challenge that requires the model to understand symbol-based representations and perform reasoning simultaneously. As an example shown in Figure 1(b), GPT-4 model responds incorrectly to the problem expressed by symbols of numerical sequences, where such symbol-based representations have been widely used among existing work (Chollet, 2019; Xu et al., 2024a; Gendron et al., 2024; Wang et al., 2024b), which may limit the ability of LLMs to solve such problems.

Considering the limitations of using language models to handle symbol-based expressions, our intuition is to convert symbols into corresponding language-based expressions, which we hope can serve as a bridge to provide LLMs with more friendly and comprehensible information when dealing with symbols. In particular, we propose symbol-to-language (S2L) conversion, a tuning-free method that leverages language-based representations to enhance model input beyond symbol-based information, aiming at better solving symbol-related problems with language models. As an example shown in Figure 1(c), the response of the GPT-4 model becomes accurate when dealing with the same problems expressed by our language-based representations.

The overall S2L framework is shown in Figure 2. To validate the effectiveness of our method, we perform experiments on eight symbol-related tasks using three LLMs including GPT-4 (OpenAI, 2023a), GPT-3.5 (OpenAI, 2022), and OpenChat (Wang et al., 2024a), an open-source model continuously trained on Mistral (Jiang et al., 2023). Experimental results demonstrate that our S2L leads to consistent improvements from symbol-only reasoning to conventional natural language processing tasks involving different types of 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.

2 Method

2.1 Task Definition

We consider a range of symbol-related problems depicted in Table 1, in which interpreting the meanings of symbol is crucial for performing the asso-

<table>
<tr><th>Symbol</th><th>Instance (s_i)</th><th>Task</th><th>Example of Symbol-related Problem (q_s)</th></tr>
<tr><td>Sequence of Numbers</td><td>1,0,0</td><td>Abstract Reasoning (§ 3.1)</td><td>1,0,0 → 0,0,1; 5,0,0 → ?</td></tr>
<tr><td>String of Brackets</td><td>{[()]}</td><td>Dyck Language (§ 3.2)</td><td>([] →); {(<>) → }; {}[→ ?</td></tr>
<tr><td>SMILES</td><td>CCCO</td><td>Property Prediction (§ 3.3)</td><td>CCCO (Toxicity: Yes or No?)</td></tr>
<tr><td>Emoji (Unicode)</td><td>😭 (U+1F62D)</td><td>Emotion Analysis (§ 3.3)</td><td>😭 (Anger? Fear? Joy? Sadness?)</td></tr>
<tr><td>Table Delimiter</td><td>|,\n</td><td>Question Answering (§ 3.3)
Fact Verification (§ 3.3)</td><td>rank|name|wins\n1|jack|3\n
(Statement: [...] True or False?)</td></tr>
<tr><td>Abbreviation</td><td>LMAO</td><td>Stance Detection (§ 3.3)
Sentiment Classification (§ 3.3)</td><td>Imagine being that bold LMAO
(Positive or Negative?)</td></tr>
</table>

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

ciated tasks. We formalize this type of problem as a question q_s that incorporates a set of symbols $s = \{s_1, ..., s_m\}$ and try to use LLMs to solve q_s.

Vanilla zero-shot (Brown et al., 2020) and zero-shot-CoT (Kojima et al., 2022) methods directly solve the original problem q_s. The responses of these two methods by LLM $\mathcal{M}$ can be written as:

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

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 an external prompt p, we focus on converting the question** q_s and propose utilizing language-based representation to leverage the strong capabilities of LLMs in natural language to better solve 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 question q_l or $q_{s \oplus l}$ as the converted or combined input for LLMs to generate answers.

2.2 Symbol-to-Language Conversion

Conversion with LLMs. We first employ the LLM $\mathcal{M}$ to convert symbols s_i to language-based representations l_i^{LLM} via zero-shot prompting:

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

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 SMILES (simplified molecular-input line-entry system) string, p_{s2l} could be "*What does the following SMILES represent?*". Alternatively, if q_s involves emotion analysis and s_i is an emoji, p_{s2l} could be "*Describe the emoji in plain text:*". l_i^{LLM} denotes the language-based representations of corresponding SMILES or emoji. See appendix A for results of different prompts.

Conversion with Tools. Considering there exist some matched "symbol-language" pairs, we further propose using external tools for conversion, which can take on several forms. (i) *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`"; (ii) *Translators* can transform SMILES strings into their formal names, such as converting s_i = "`CCCO`" into $l_i^{\text{translator}}$ = "Propionylo"; (iii) *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 avoids the costs associated with using LLMs; 2) it provides verified language-based information, which can help reduce potential errors in descriptions generated by LLMs.

2.3 Using Language-Based Representations

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

Substitution. The first way of utilization is to directly substitute 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

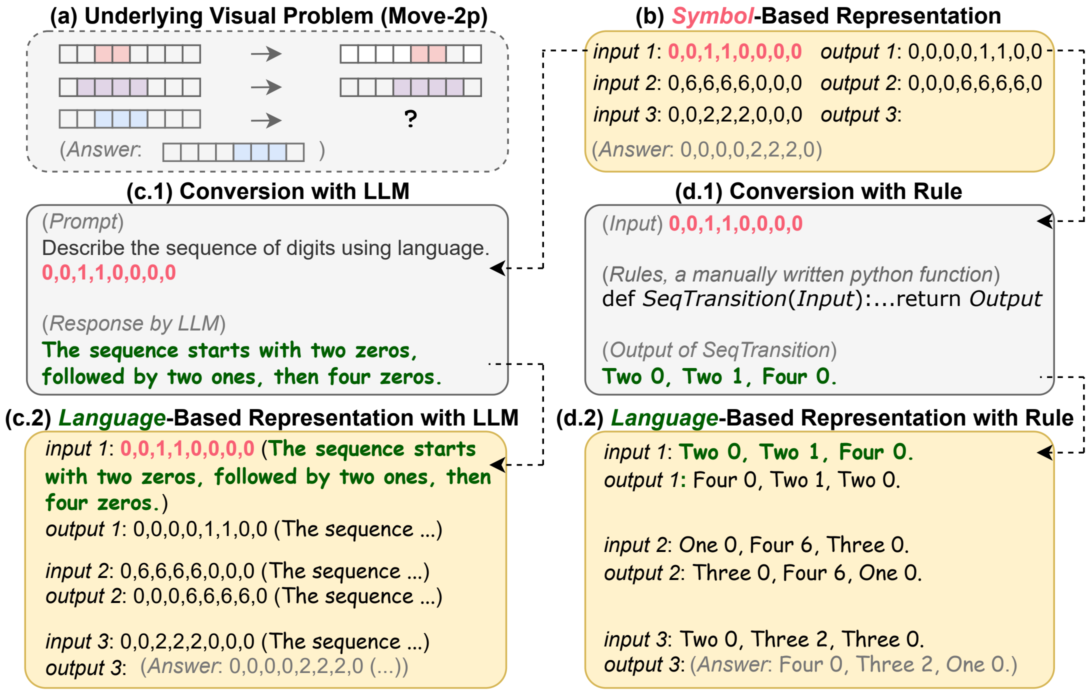

Figure 3: Example of applying symbol-to-language for 1D abstract reasoning task. We convert each numerical sequence to its language-based representation by prompting LLM or using simple rules implemented in code, and then we transform the symbolized problem to language-enhanced or language-only representations.

the symbol-based representations for both the question and ground-truth label. The response by using S2L can be written as:

$$\begin{aligned} \mathcal{R}_{\text{s2l}} &= \mathcal{M}(q_l), \\ q_l &= q_{f\circ\{s_1,\ldots,s_m\}} = q_{f\circ s_1,\ldots,f\circ s_m}. \end{aligned} \tag{3}$$

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 combined input. This allows LLM $\mathcal{M}$ to reason on questions that include rich contextual information from two distinct perspectives:

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

3 Experiments

We evaluate both closed-source and open-source models, including GPT-4 (`gpt-4-0613`), GPT-3.5 (`gpt-3.5-turbo-1106`), and OpenChat-7B (`openchat-3.5-0106`, Wang et al., 2024a). The decoding temperature is set as 0 during inference.

3.1 Abstract Reasoning

Abstract reasoning (Webb et al., 2023; Gendron et al., 2024; Wang et al., 2024b) involves summarizing patterns from limited observations. We perform experiments on tasks from the 1D-ARC benchmark introduced by Xu et al. (2024c). The 1D-ARC comprises various 1D object-based visual problems, as depicted in Figure 3(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 3(b).

Symbol-to-Language Conversion. We apply both LLMs and rule-based codes to convert the sequences. We find that LLMs describe the sequence similar to humans coincidentally, which employs merging or counting when describing numerical 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

Method	Move-1p		Move-2p		Move-3p		Avg.
	#obs = 3	#obs = 4	#obs = 3	#obs = 4	#obs = 3	#obs = 4	
GPT-4	93.3	90.0	48.3	46.7	35.0	45.0	59.7
+S2L w/ model (ours)	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 (ours)	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-4-CoT	93.3	90.0	55.0	50.0	36.7	41.7	61.1
+S2L w/ model (ours)	95.0 (+1.7)	91.7 (+1.7)	53.3 (−1.7)	51.7 (+1.7)	43.3 (+6.6)	48.3 (+6.6)	**63.9** (+2.8)
+S2L w/ rule (ours)	96.7 (+3.4)	100.0 (+10.0)	86.7 (+31.7)	88.3 (+38.3)	50.0 (+13.3)	66.7 (+25.0)	**81.4** (+20.3)
GPT-3.5	68.3	71.7	11.7	25.0	23.3	26.7	37.8
+S2L w/ model (ours)	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 (ours)	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)
GPT-3.5-CoT	76.7	78.3	15.0	25.0	26.7	25.0	41.1
+S2L w/ model (ours)	75.0 (−1.7)	80.0 (+1.7)	30.0 (+15.0)	35.0 (+10.0)	35.0 (+8.3)	30.0 (+5.0)	**47.5** (+6.4)
+S2L w/ rule (ours)	75.0 (−1.7)	86.7 (+8.4)	30.0 (+15.0)	36.7 (+11.7)	21.7 (−5.0)	25.0 (–)	**45.8** (+4.7)
OpenChat-7B	61.7	71.7	15.0	21.6	11.7	11.7	32.2
+S2L w/ model (ours)	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 (ours)	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)
OpenChat-7B-CoT	63.3	71.7	18.3	26.7	11.7	13.3	34.2
+S2L w/ model (ours)	71.7 (+8.4)	81.7 (+10.0)	20.0 (+1.7)	30.0 (+3.3)	23.3 (+11.6)	21.7 (+8.4)	**41.4** (+7.2)
+S2L w/ rule (ours)	66.7 (+3.4)	76.7 (+5.0)	21.7 (+3.4)	35.0 (+8.3)	23.3 (+11.6)	26.7 (+13.4)	**41.7** (+7.5)

Table 2: Results for abstract reasoning task. #obs indicates the number of input-output observations given. w/ rule denotes conversion with manually designed rules using codes.

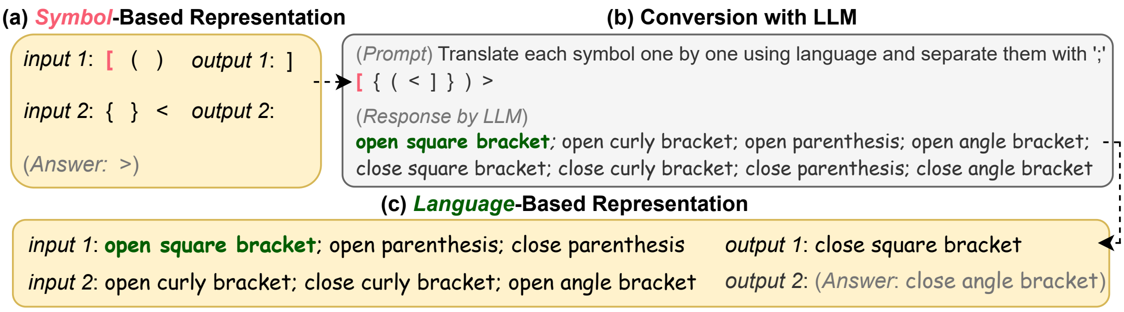

Figure 4: Example of applying symbol-to-language for Dyck language task. We convert every symbol (*e.g.*, "[") to its textual description (*e.g.*, "open square bracket") by prompting LLMs.

Figure 3(c.1) and Figure 3(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 3(c.2). In contrast, the outputs of the rule-based codes l_i^{rule} are equal to the original sequence of numbers, so we directly use them as shown in Figure 3(d.2).

Settings and Results. We use Move-1p, Move-2p, and Move-3p tasks (move 1, 2, and 3 pixels, respectively). We collect the 1D sequence and result in 60 problems for each task with 3 or 4 input-output observations. The experimental results are presented in Table 2. GPT-4 achieves an accuracy above 90.0% on Move-1p. However, the performance drops rapidly to 30∼50% on Move-2p and Move-3p, demonstrating that the model struggles to reason on sequences with slightly more complex patterns. This phenomenon is even more evident for GPT-3.5 and OpenChat, where the overall accuracy is much lower. By employing conversion with LLMs (*i.e.*, S2L w/ model), the results improve by +2.5∼8.6%, suggesting the positive impact of additional language-based information. By employing conversion with rule-based codes, GPT-4 gets 100% accuracy given 4 input-output observations, 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 among three LLMs, indicating the positive and distinct impact of additional language-based representation across models.

3.2 Dyck Language

Dyck language aims to predict the closing parentheses of a given sequence (Srivastava et al., 2023). To evaluate the inductive reasoning ability on sym-

Method	Dyck Language					AVG.
	#obs = 1	#obs = 2	#obs = 3	#obs = 4	#obs = 5	
GPT-4	60.0	82.2	86.6	91.4	92.2	82.5
+S2L w/ model (ours)	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-4-CoT	56.0	80.2	86.4	91.6	92.6	81.4
+S2L w/ model (ours)	75.4 (+19.4)	91.2 (+11.0)	95.8 (+9.4)	97.6 (+6.0)	98.4 (+5.8)	**91.7** (+10.3)
GPT-3.5	65.0	77.0	78.0	80.8	78.2	75.8
+S2L w/ model (ours)	69.6 (+4.6)	82.8 (+5.8)	88.2 (+10.2)	93.2 (+12.4)	94.0 (+15.8)	**85.6** (+9.8)
GPT-3.5-CoT	73.0	75.0	74.2	80.4	78.0	76.1
+S2L w/ model (ours)	68.8 (−4.2)	83.0 (+8.0)	90.0 (+15.8)	93.6 (+13.2)	93.8 (+15.8)	**85.8** (+9.7)
OpenChat-7B	3.4	3.6	4.2	5.2	7.8	4.8
+S2L w/ model (ours)	21.6 (+18.2)	22.0 (+18.4)	32.0 (+27.8)	43.8 (+38.6)	59.0 (+51.2)	**35.7** (+30.9)
OpenChat-7B-CoT	3.4	3.8	4.6	5.2	7.4	4.9
+S2L w/ model (ours)	22.0 (+18.6)	22.6 (+18.8)	31.6 (+27.0)	44.6 (+39.4)	59.0 (+51.6)	**36.0** (+31.1)

Table 3: Results for Dyck language task. #obs indicates the number of input-output observations given.

bols, we do not ask LLMs to "complete parentheses" (*i.e.*, prompting LLMs to output the remaining parentheses). Instead, following abstract reasoning, we only give some input-output observations and let LLMs to "deduce the output" according to the implicit patterns, as shown in Figure 4(a).

Symbol-to-Language Conversion. The symbols include totally eight brackets ("[]{}()<>"). We convert each symbol s_i to language description l_i^{LLM}. Thus, we can convert the problem into language-based representations as in Figure 4(c).

Settings and Results. We first set the number of observations as 5 and evaluate the accuracy among 500 questions. Then we gradually remove the last input-output observations to test the ability with fewer (*i.e.*, 4, 3, 2, and 1) observations. The results are shown in Table 3. For GPT-4 and GPT-3.5, the performance ranges from 60.0∼92.2% and 65.0∼78.2%, respectively. The accuracy further improves with +9.5% and +9.8% by using S2L. For OpenChat, the performance is extremely low with accuracy below 10% and S2L improves the performance by a large margin with +30.9%.

3.3 Other Tasks

We further show results for six more general tasks. Due to the space limit, the readers are referred to appendix B and C for settings of symbol-to-language conversion and case study for these tasks.

Chemical Property Prediction. ChemLLMBench (Guo et al., 2023) is used for property prediction given SMILES string, including BACE (bindings results for a set of inhibitors of human beta-secretase), BBBP (penetration to the brain-blood barrier), and Tox21 (toxicity of compounds).

Method	BACE	BBBP	Tox21	AVG.
GPT-4	48.2	40.2	47.2	45.2
+S2L w/ model (ours)	53.0 (+4.8)	53.0 (+12.8)	48.6 (+1.4)	**51.5** (+6.3)
+S2L w/ translator (ours)	48.4 (+0.2)	58.8 (+18.6)	49.0 (+1.8)	**52.1** (+6.9)
GPT-4-CoT	50.4	36.8	37.4	41.5
+S2L w/ model (ours)	55.6 (+5.2)	54.2 (+17.4)	44.0 (+6.6)	**51.3** (+9.8)
+S2L w/ translator (ours)	51.2 (+0.8)	64.0 (+27.2)	45.2 (+7.8)	**53.5** (+12.0)
GPT-3.5	52.4	24.6	34.2	37.1
+S2L w/ model (ours)	53.0 (+0.6)	35.0 (+10.4)	34.8 (+0.6)	**40.9** (+3.8)
+S2L w/ translator (ours)	54.8 (+2.4)	53.8 (+29.2)	51.0 (+16.8)	**53.2** (+16.1)
GPT-3.5-CoT	44.2	32.0	35.8	37.3
+S2L w/ model (ours)	48.2 (+4.0)	35.8 (+3.8)	38.2 (+2.4)	**40.7** (+3.4)
+S2L w/ translator (ours)	48.6 (+4.4)	41.4 (+9.4)	41.8 (+6.0)	**43.9** (+6.6)
OpenChat-7B	48.2	46.8	62.6	52.5
+S2L w/ model (ours)	48.8 (+0.6)	56.2 (+9.4)	65.0 (+2.4)	**56.7** (+4.2)
+S2L w/ translator (ours)	55.8 (+7.6)	59.2 (+12.4)	67.8 (+5.2)	**60.9** (+8.4)
OpenChat-7B-CoT	49.6	46.2	62.2	52.7
+S2L w/ model (ours)	47.8 (−1.8)	48.4 (+2.2)	60.4 (−1.8)	52.2 (−0.5)
+S2L w/ translator (ours)	52.2 (+2.6)	65.2 (+19.0)	61.4 (−0.8)	**59.6** (+6.9)

Table 4: Results for chemical property prediction. w/ translator denotes conversion with an external translator.

Following Guo et al. (2023), we randomly sample 500 instances from the full test set over repeated five times and report the averaged accuracy, the results are shown in Table 4. The overall performance of GPT-4 and GPT-3.5 is relatively low, showing the difficulty for LLMs to understand the chemical formula and their property. Similarly, the chain-of-thought reasoning does not always effective and even lead to performance drop for GPT-4 model (from 45.2% to 41.5%). Our method generally lead to better performance with varying degrees. For example, the improvement is large for the BBBP dataset (+12.8∼27.2%) while it becomes relatively low for the BACE dataset (+0.2∼5.2%) using GPT-4. Overall, the results show that S2L is effective by providing language-based information that aids in chemical property prediction tasks.

Method	EmoTag1200 (Pearson correlation $r \times 100$)								AVG.
	ANGER	ANTICIPATE	DISGUST	FEAR	JOY	SADNESS	SURPRISE	TRUST	
GPT-4	85.5	29.0	78.2	80.9	88.7	90.3	53.1	73.1	72.4
+S2L w/ model (ours)	87.8	30.8	79.2	81.9	89.7	92.2	56.2	76.6	**74.3** (+1.9)
+S2L w/ dict (ours)	85.6	34.5	75.3	81.3	89.4	92.6	61.6	76.0	**74.5** (+2.1)
GPT-4-CoT	85.4	20.5	72.3	81.0	88.9	91.7	50.7	74.4	70.6
+S2L w/ model (ours)	85.4	33.0	70.5	81.4	88.9	92.3	63.2	74.1	**73.6** (+3.0)
+S2L w/ dict (ours)	86.5	33.7	66.1	82.5	88.9	93.0	62.7	73.0	**73.3** (+2.7)
GPT-3.5	71.0	21.4	33.4	58.9	69.5	79.0	19.2	56.0	51.0
+S2L w/ model (ours)	70.0	15.6	40.1	62.9	78.6	85.0	48.3	63.0	**57.8** (+6.8)
+S2L w/ dict (ours)	75.7	26.9	48.7	69.6	79.6	85.4	28.5	70.0	**60.5** (+9.5)
GPT-3.5-CoT	55.9	1.30	5.10	15.6	26.4	14.5	−8.40	6.00	14.6
+S2L w/ model (ours)	70.2	15.2	39.3	63.1	78.5	84.5	48.5	63.3	**57.8** (+43.2)
+S2L w/ dict (ours)	73.4	20.6	50.5	66.4	79.9	85.3	35.4	54.3	**58.2** (+43.6)
OpenChat-7B	41.3	3.00	43.8	26.4	16.1	23.2	22.1	−8.60	20.9
+S2L w/ model (ours)	46.5	6.60	53.5	63.9	35.5	58.3	1.00	−11.8	**31.7** (+10.8)
+S2L w/ dict (ours)	57.2	0.20	58.7	64.8	37.7	58.7	2.30	−7.80	**33.9** (+13.0)
OpenChat-7B-CoT	45.5	−1.40	27.7	34.8	45.4	74.1	16.3	1.20	30.5
+S2L w/ model (ours)	63.2	17.2	43.8	53.2	66.9	69.2	34.4	9.80	**44.7** (+14.2)
+S2L w/ dict (ours)	56.4	−5.00	38.1	61.3	68.4	58.1	19.4	6.40	**38.0** (+7.5)

Table 5: Results for emotion analysis of emojis. The numbers are Pearson correlation coefficient with ratings by humans. w/ dict denotes conversion with the Unicode dictionary.

Emotion Analysis of Emojis. We use the EmoTag1200 (Shoeb and de Melo, 2020) for analyzing the emotions of the emojis. Specifically, 150 most frequently used emojis are evaluated and the task is to score each of them from 0∼1 based on eight basic emotions, including *anger*, *anticipation*, *disgust*, *fear*, *joy*, *sadness*, *surprise*, and *trust*.

The results are shown in Table 5. GPT-4 gives a relatively high correlation coefficient of 72.4. However, GPT-3.5 and OpenChat perform poorly with results of 51.0 and 20.9, respectively. Again, the performance does not improve consistently when adding "*Let's think step by step*" for chain-of-thought reasoning, even lead to large performance decrease for GPT-3.5. By employing S2L with model or dictionary, the performance improves to different degrees, showing that language information plays an role for analyzing emojis.

Table Understanding. For structured data, we use WikiTableQuestions (Pasupat and Liang, 2015) for tabular question-answering, including 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.

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 still consistently bring about +1.7∼4.2% improvements among different models, showing that external natural language information is still effective for tabular tasks.

Method	TableQA (F1)	TableQA (EM)	TabFact (Acc.)	AVG.
GPT-4	82.0	79.8	93.6	85.1
+S2L w/ model (ours)	84.6 (+2.6)	82.0 (+2.2)	95.6 (+2.0)	**87.4** (+2.3)
+S2L w/ rule (ours)	86.5 (+4.5)	84.2 (+4.4)	95.8 (+2.2)	**88.8** (+3.7)
GPT-4-CoT	80.7	76.8	93.2	83.6
+S2L w/ model (ours)	82.9 (+2.2)	80.2 (+3.4)	94.6 (+1.4)	**85.9** (+2.3)
+S2L w/ rule (ours)	84.2 (+3.5)	80.6 (+3.8)	96.4 (+3.2)	**87.1** (+3.5)
GPT-3.5	66.4	63.0	82.0	70.5
+S2L w/ model (ours)	69.0 (+2.6)	66.0 (+3.0)	84.6 (+2.6)	**73.2** (+2.7)
+S2L w/ rule (ours)	68.5 (+2.1)	64.8 (+1.8)	86.2 (+4.2)	**73.2** (+2.7)
GPT-3.5-CoT	73.4	69.8	77.0	73.4
+S2L w/ model (ours)	73.6 (+0.2)	71.2 (+1.4)	83.0 (+6.0)	**75.9** (+2.5)
+S2L w/ rule (ours)	77.0 (+3.6)	72.8 (+3.0)	83.0 (+6.0)	**77.6** (+4.2)
OpenChat-7B	61.7	58.2	79.0	66.3
+S2L w/ model (ours)	64.1 (+2.4)	59.0 (+0.8)	81.0 (+2.0)	**68.0** (+1.7)
+S2L w/ rule (ours)	62.1 (+0.4)	58.8 (+0.6)	83.0 (+4.0)	**67.9** (+1.6)
OpenChat-7B-CoT	60.8	57.0	83.8	67.2
+S2L w/ model (ours)	64.2 (+3.4)	60.4 (+3.4)	83.2 (−0.6)	**85.9** (+2.3)
+S2L w/ rule (ours)	66.1 (+5.3)	63.0 (+6.0)	84.6 (+0.8)	**87.1** (+3.5)

Table 6: Results for tabular question-answering and fact verification. w/ rule denotes conversion with manually designed rules for aligning the contents of table rows.

Tweet Analysis. We use the TweetSentimentExtraction dataset from Massive Text Embedding Benchmark (Muennighoff et al., 2023) for sentiment classification, and we follow Zhang et al. (2023) by using the P-Stance (Li et al., 2021) dataset for stance detection.

Method	P-Stance (Acc.)	P-Stance (F1)	Sentiment (Acc.)	AVG.
GPT-4	86.2	86.7	89.4	87.4
+S2L w/ model (ours)	87.1 (+0.9)	88.2 (+1.5)	90.3 (+0.9)	**88.5** (+1.1)
GPT-4-CoT	83.8	84.0	86.1	84.6
+S2L w/ model (ours)	86.6 (+2.8)	87.5 (+3.5)	87.2 (+1.1)	**87.1** (+2.5)
GPT-3.5	65.3	68.6	83.7	72.5
+S2L w/ model (ours)	71.0 (+5.7)	71.5 (+2.9)	89.9 (+6.2)	**77.5** (+5.0)
GPT-3.5-CoT	61.5	60.8	76.5	66.3
+S2L w/ model (ours)	64.7 (+3.2)	63.5 (+2.7)	78.4 (+1.9)	**68.9** (+2.6)
OpenChat-7B	70.9	66.7	89.1	75.6
+S2L w/ model (ours)	71.4 (+0.5)	67.5 (+0.8)	89.0 (−0.1)	**76.0** (+0.4)
OpenChat-7B-CoT	72.2	69.1	84.8	75.4
+S2L w/ model (ours)	77.1 (+4.9)	75.8 (+6.7)	82.1 (−2.7)	**78.3** (+2.9)

Table 7: Results for stance detection and sentiment analysis in social media texts.

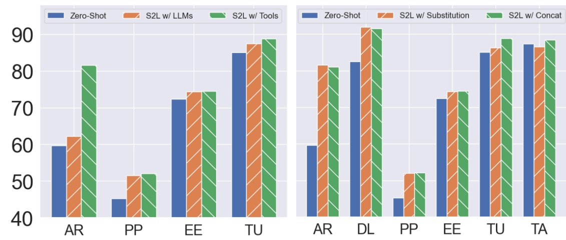

Figure 5: Using LLMs or tools (left) and substituting or concatenating (right) for S2L conversion with GPT-4. AR: abstract reasoning, DL: Dyck language, PP: property prediction, EE: emotion of emojis, TU: table understanding, TA: tweet analysis.

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. We find that, although stance and sentiment analysis in social media has been extensively studied in NLP, our S2L can improve the performance of current advanced models, even by a large margin of $+5.0\%$ accuracy for GPT-3.5.

4 Analysis

Conversion with LLMs or Tools. As shown in Figure 5(left), we find that using tools generally yields better results than using models for S2L conversion[2]. For abstract reasoning, the tools enable LLMs to explicitly discover patterns of numerical sequences from language-based representations. For other tasks, LLMs may already capture some knowledge of SMILES, emojis or table delimiters, thus the benefits brought by tools is reduced. In general, both approaches outperform zero-shot reasoning over symbol-based representations.

Substitution or Concatenation. We also compare the different ways for leveraging the language-based representation in Figure 5(right). For abstract reasoning and Dyck language, the language-based representation can convey enough information for solving the problems, thus directly substituting language-based representation for symbol-based representation can slightly improve the results, without influenced by abstract symbols. However, for other tasks, we find that concatenate both representations further improve the performance. The reasons can be that LLMs have already trained on similar dataset, allowing them to provide better answers by simultaneously combining information from both types of representations.

[2]We find that conversion with LLMs are effective in interpreting the brackets for Dyck language, leading to the same results by using tools such as dictionary. Also, the types of symbol involved in Tweet are not fixed. Therefore, we do not employ S2L conversion with specific tools for these two tasks.

5 Related Work

Reasoning on Symbol-Related Problems. Various studies have explored the capabilities of LLMs in symbol-based tasks. Gendron et al. (2024) demonstrate that LLMs possess a limited capacity for symbol-based reasoning compared to other natural language tasks. Wang et al. (2024b) suggest that LLMs can improve the performance of abstract reasoning once having strong ability to generate executable codes. Qiu et al. (2024) assess LLMs on inductive reasoning tasks, uncovering a range of counter-intuitive behaviors. These works indicate that there is room for improving reasoning ability for symbol-related problems with LLMs.

Reasoning by Chain-of-Thought Prompting. Chain-of-Thought prompting (Wei et al., 2022; Kojima et al., 2022; Chen et al., 2023; Yao et al., 2023; Besta et al., 2024) have become integral in augmenting the reasoning abilities of LLMs. In particular, Xu et al. (2023) propose to re-read the question before reasoning. Deng et al. (2023) introduce rephrase-and-respond to tackle ambiguous questions by using self-rephrased questions, which shares similarities with our method in rephrasing. However, we aim to address symbol-related problems, convert 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 enhance their in-context learning abilities. Hu et al. (2024) propose chain-of-symbol to solve planning-based tasks. Wang et al. (2024c) introduce meta-reasoning as a means to construct generic symbolic representations for reasoning tasks. Xu et al. (2024b) integrates logic rules with CoT prompting. In contrast, we propose eliciting capabilities of LLMs by leveraging converted information in *language*-level to help solving the symbol-related problems.

6 Conclusion

We propose symbol-to-language conversion to leverage language-based representation for solving symbol-related problems using LLMs. The motivation is to elicit the knowledge behind the symbols through natural language, serving as useful information for models. Experiments with three LLMs show that we consistently improve performance across eight tasks, *e.g.*, abstract reasoning, Dyck language, and chemical property prediction. We hope it can further harness the power of language-based representations and explore the untapped potential of LLMs to play roles in more scenarios.

Limitations

Although we verified symbol-to-language conversion 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, some of the original 2D visual problems from the ARC dataset (Chollet, 2019) are still difficult to describe in brief language-based representations, even with tools. Second, for tasks that we cannot 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, though we propose to append the original information to alleviate the impact.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 62306161, 61925601, 62276152).

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michał Podstawski, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. *Proc. of AAAI*.

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. *Proc. of NeurIPS*.

Wenhu Chen. 2023. Large language models are few(1)-shot table reasoners. In *Proc. of ACL Findings*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*.

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

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

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

Gaël Gendron, Qiming Bao, Michael Witbrock, and Gillian Dobbie. 2024. Large language models are not strong abstract reasoners. In *Proc. of IJCAI*.

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. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Taicheng Guo, kehan Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh Chawla, Olaf Wiest, and Xiangliang Zhang. 2023. What can large language models do in chemistry? a comprehensive benchmark on eight tasks. In *Proc. of NeurIPS*.

Hanxu Hu, Hongyuan Lu, Huajian Zhang, Yun-Ze Song, Wai Lam, and Yue Zhang. 2024. Chain-of-symbol prompting for spatial reasoning in large language models. In *First Conference on Language Modeling*.

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

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *Proc. of ICML*.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Proc. of NeurIPS*.

Yingjie Li, Tiberiu Sosea, Aditya Sawant, Ajith Jayaraman Nair, Diana Inkpen, and Cornelia Caragea. 2021. P-stance: A large dataset for stance detection in political domain. In *Proc. of ACL Findings*.

Melanie Mitchell, Alessandro B. Palmarini, and Arsenii Kirillovich Moskvichev. 2023. Comparing humans, GPT-4, and GPT-4v on abstraction and reasoning tasks. In *Proc. of AAAI*.

Arsenii Kirillovich Moskvichev, Victor Vikram Odouard, and Melanie Mitchell. 2023. The conceptARC benchmark: Evaluating understanding and generalization in the ARC domain. *Transactions on Machine Learning Research*.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proc. of EACL*.

Setsuo Ohsuga. 2007. Bridging the gap between non-symbolic and symbolic processing–how could human being acquire language? *Fundamenta Informaticae*.

OpenAI. 2022. GPT-3.5 turbo model.

OpenAI. 2023a. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

OpenAI. 2023b. GPT-4V(ision) system card.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proc. of ACL*.

Linlu Qiu, Liwei Jiang, Ximing Lu, Melanie Sclar, Valentina Pyatkin, Chandra Bhagavatula, Bailin Wang, Yoon Kim, Yejin Choi, Nouha Dziri, and Xiang Ren. 2024. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. In *Proc. of ICLR*.

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

Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. 2022. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence*.

Abu Awal Md Shoeb and Gerard de Melo. 2020. EmoTag1200: Understanding the association between emojis and emotions. In *Proc. of EMNLP*.

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. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

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

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. 2024a. Openchat: Advancing open-source language models with mixed-quality data. In *Proc. of ICLR*.

Ruocheng Wang, Eric Zelikman, Gabriel Poesia, Yewen Pu, Nick Haber, and Noah Goodman. 2024b. Hypothesis search: Inductive reasoning with language models. In *Proc. of ICLR*.

Yiming Wang, Zhuosheng Zhang, and Rui Wang. 2024c. Meta-reasoning: Semantics-symbol deconstruction for large language models. In *Proc. of ACL Findings*.

Taylor Webb, Keith J Holyoak, and Hongjing Lu. 2023. Emergent analogical reasoning in large language models. *Nature Human Behaviour*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proc. of NeurIPS*.

Jerry Wei, Le Hou, Andrew Lampinen, Xiangning Chen, Da Huang, Yi Tay, Xinyun Chen, Yifeng Lu, Denny Zhou, Tengyu Ma, and Quoc Le. 2023. Symbol tuning improves in-context learning in language models. In *Proc. of EMNLP*.

Fangzhi Xu, Zhiyong Wu, Qiushi Sun, Siyu Ren, Fei Yuan, Shuai Yuan, Qika Lin, Yu Qiao, and Jun Liu. 2024a. Symbol-LLM: Towards foundational symbol-centric interface for large language models. In *Proc. of ACL*.

Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. 2024b. Faithful logical reasoning via symbolic chain-of-thought. In *Proc. of ACL*.

Xiaohan Xu, Chongyang Tao, Tao Shen, Can Xu, Hongbo Xu, Guodong Long, and Jian-guang Lou. 2023. Re-reading improves reasoning in language models. *arXiv preprint arXiv:2309.06275*.

Yudong Xu, Wenhao Li, Pashootan Vaezipoor, Scott Sanner, and Elias Boutros Khalil. 2024c. LLMs and the abstraction and reasoning corpus: Successes, failures, and the importance of object-based representations. *Transactions on Machine Learning Research*.

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

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

A Performance of Different Prompts for Conversion with LLM

We compare using different prompts for S2L conversion with LLM in Table 8. We found that different prompts have a slight impact, but overall do not affect the effectiveness of the proposed method.

Prompts for Conversion w/ LLM	Perf.
1D-ARC	
Describe the sequence of digits using language.	**62.2**
Describe the string of digits using verbal expressions.	61.7
Explain the series of numbers using words.	60.5
GPT-4 (w/o using our method)	59.7
Dyck Language	
Translate each symbol one by one using language and separate them with ';'.	**92.0**
Translate each symbol sequentially into words and separate them with ';'.	91.6
Interpret each symbol individually into words and delimit them using ';'.	91.6
GPT-4 (w/o using our method)	82.5
Property Prediction	
What does the following SMILES represent?	**51.5**
Briefly describe what the following SMILES represents.	50.4
Explain the SMILES notation using plain texts:	49.7
GPT-4 (w/o using our method)	45.2
Emotion Analysis	
Describe the emoji in plain text:	**74.3**
Explain the emoji using text:	73.3
Provide a textual description of the emoji:	72.9
GPT-4 (w/o using our method)	72.4
Tabular Question-Answering	
Describe the table in plain texts:	**84.6**
Describe the table row by row:	84.3
Describe all the details of the table:	83.9
GPT-4 (w/o using our method)	82.0
Sentiment Classification	
Transfer to plain text tweets:	**90.3**
Convert the tweet into plain text:	89.8
Transfer the tweet and emoji to plain texts:	89.7
GPT-4 (w/o using our method)	89.4

Table 8: Compare different prompts for S2L conversion.

B Settings and Example of Applying S2L Conversion for Different Tasks

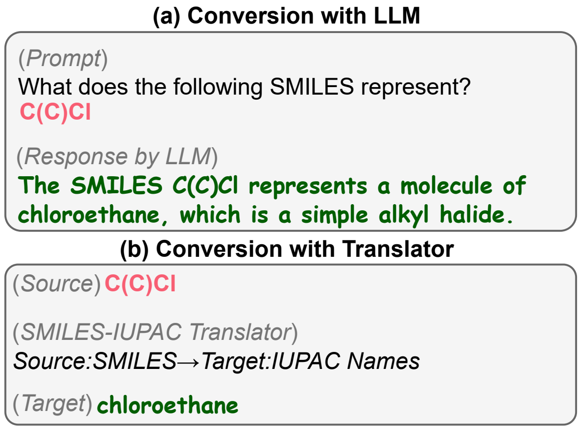

Figure 6: Example of applying symbol-to-language for property prediction. We convert each SMILES to its language-based representation by prompting LLMs or using a translator.

Chemical Property Prediction. 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 6(a). Instead of using LLMs, we further propose S2L with STOUT V2.0[3] (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 $l_i^{\mathrm{translator}}$ of a given SMILES, as shown in Figure 6(b). Finally, we append the obtained information to each SMILES notation as language-enhanced input for LLMs.

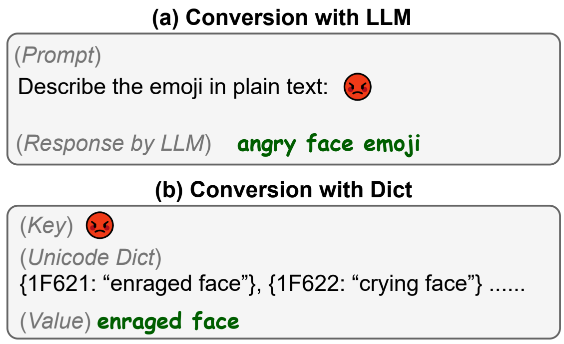

Figure 7: Example of applying symbol-to-language for emotional reranking of emojis. We convert each emoji to its language-based representation by prompting LLMs or using the names from the Unicode dictionary.

Emotion Analysis of Emojis. To understand emojis with language-based information, we get the

[3]https://github.com/Kohulan/Smiles-TO-iUpac-Translator.

(a) Conversion with LLM

(*Prompt*)
Describe the table in plain texts:
Rank|Nation|Gold|Silver|Bronze|Total\n1|Sweden (SWE)|4|3|0|7\n

(*Response by LLM*)
Sweden is ranked first with 4 gold, 3 silver, and **0 bronze** medals, making a total of 7 medals.

(b) Conversion with Rule

(*Input*)
Rank|Nation|Gold|Silver|Bronze|Total\n1|Sweden (SWE)|4|3|0|7\n

(*Rules, a manually written python function*)
def *TableTransition*(*Input*):…return *Output*

(*Output of TableTransition*)
Rank: 1; Nation: Sweden (SWE); Gold: 4; Silver: 3; **Bronze: 0;** Total: 7

Figure 8: Example of applying symbol-to-language for tabular question-answering task. We convert each table to its language-based representation by prompting LLMs or using simple rules implemented in codes.

description l_i^{LLM} by prompting LLMs or directly obtaining the names l_i^{dict} from the Unicode dictionary[4], as shown in Figure 7.

Table Understanding. As shown in Figure 8, 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.

(*Prompt*)
Transfer to plain text tweets: lmaooo all day everyday !! 😂😂

(*Response by LLM*)
laughing my ass off all day everyday! **I'm laughing so hard**.

Figure 9: Example of applying symbol-to-language for tweet understanding. We convert each tweet to plain text by prompting LLMs.

Tweet Analysis. Non-natural language expressions are widely exist in social media such as 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 9, and then we use it as external input for each question.

[4] https://unicode.org/emoji/charts/full-emoji-list.html

C Case Study

We show responses by GPT-4 with and without our method for 1D-ARC task in figures below.

1D-ARC task (by GPT-4)
(Prompt)
```
Let's play some puzzles that focus on reasoning and logic. In each puzzle, you will be provided a few demonstrations of how an "input grid" gets transformed into a corresponding "output grid". At the end, you will get a brand new "input grid", then you must answer the corresponding "output grid".

input 1: 0,0,0,0,3,3,3,3,3,0,0,0,0,0
output 1: 0,0,0,0,0,3,3,3,3,3,0,0,0,0
input 2: 0,7,7,7,7,7,7,7,7,0,0,0,0,0
output 2: 0,0,7,7,7,7,7,7,7,7,0,0,0,0
input 3: 0,0,2,2,2,0,0,0,0,0,0,0,0,0
output 3: 0,0,0,2,2,2,0,0,0,0,0,0,0,0
input 4: 0,0,0,0,7,7,7,7,7,7,0,0,0,0
output 4:
```
(Response)
```
0,0,0,0,0,7,7,7,7,7,7,0,0,0,0,0 (✗)
```

1D-ARC task (by GPT-4 w/ our method)
(Prompt)
```
Let's play some puzzles that focus on reasoning and logic. In each puzzle, you will be provided a few demonstrations of how an "input grid" gets transformed into a corresponding "output grid". At the end, you will get a brand new "input grid", then you must answer the corresponding "output grid".

input 1: 0,0,0,0,3,3,3,3,3,0,0,0,0,0
(The sequence starts with four zeros, followed by five threes, and ends with five zeros.)
output 1: 0,0,0,0,0,3,3,3,3,3,0,0,0,0
(The sequence consists of five 0s, followed by five 3s, and then followed by four 0s.)
input 2: 0,7,7,7,7,7,7,7,7,0,0,0,0,0
(The sequence starts with a single zero, followed by eight sevens, and then ends with five zeros.)
output 2: 0,0,7,7,7,7,7,7,7,7,0,0,0,0
(The sequence starts with two zeros, followed by eight sevens, and ends with four zeros.)
input 3: 0,0,2,2,2,0,0,0,0,0,0,0,0,0
(The sequence starts with two zeros, followed by three twos, and then nine zeros.)
output 3: 0,0,0,2,2,2,0,0,0,0,0,0,0,0
(The sequence starts with three 0s, followed by three 2s, and then eight 0s.)
input 4: 0,0,0,0,7,7,7,7,7,7,0,0,0,0
(The sequence consists of four zeros, followed by six sevens, and then ends with four zeros.)
output 4:
```
(Response)
```
0,0,0,0,0,7,7,7,7,7,7,0,0,0 (✓)
(The sequence starts with five zeros, followed by six sevens, and then ends with three zeros.)
```

We show responses by GPT-4 with and without our method for Dyck Language in figures below.

Dyck Language task (by GPT-4)
(Prompt)
```
Let's play some puzzles that focus on reasoning and logic. In each puzzle, you will be provided a few demonstrations of how an "input sequence" gets transformed into a corresponding "output sequence". At the end, you will get a brand new "input sequence", then you must answer the corresponding "output sequence".

input 1: ( { }
output 1: )
input 2: { { ( ) }
output 2:
```
(Response)
`( ) ) (✗)`

Dyck Language task (by GPT-4 w/ our method)
(Prompt)
```
Let's play some puzzles that focus on reasoning and logic. In each puzzle, you will be provided a few demonstrations of how an "input sequence" gets transformed into a corresponding "output sequence". At the end, you will get a brand new "input sequence", then you must answer the corresponding "output sequence".

input 1: open parenthesis; open curly bracket; close curly bracket
output 1: close parenthesis
input 2: open curly bracket; open curly bracket; open parenthesis; close parenthesis; close curly bracket
output 2:
```
(Response)
`close curly bracket (✓)`

We show responses by GPT-4 with and without our method for property prediction in figures below.

Property prediction task (by GPT-4)
(Prompt)
```
You are an expert chemist, your task is to predict the property of molecule using your experienced chemical property prediction knowledge. Given the SMILES string of a molecule, the task focuses on predicting molecular properties, specifically penetration/non-penetration to the brain-blood barrier, based on the SMILES string representation of each molecule. The task is to predict the binary label for a given molecule whether it has penetrative property (Yes) or not (No), please answer with only Yes or No.

SMILES: FC(Br)C(F)(F)F

Penetration:
```
(Response)
`No (✗)`

Property prediction task (by GPT-4 w/ our method)
(Prompt)
```
You are an expert chemist, your task is to predict the property of molecule using your experienced chemical property prediction knowledge. Given the SMILES string of a molecule, the task focuses on predicting molecular properties, specifically penetration/non-penetration to the brain-blood barrier, based on the SMILES string representation of each molecule. The task is to predict the binary label for a given molecule whether it has penetrative property (Yes) or not (No), please answer with only Yes or No.

SMILES: FC(Br)C(F)(F)F

The SMILES represents the chemical structure of Halothane, a type of general anesthetic.

Penetration:
```
(Response)
`Yes (✓)`

We show responses by GPT-4 with and without our method for tabular question-answering in figures below.

Tabular QA task (by GPT-4)
(Prompt)
```
Your task is to answer the question based on the table. Please show the final answer with only one or few words.

Table:
Rank | Nation | Gold | Silver | Bronze | Total
1 | Malaysia | 3 | 0 | 1 | 4
2 | Indonesia | 1 | 3 | 2 | 6
3 | Korea | 1 | 1 | 2 | 4
4 | Thailand | 1 | 0 | 0 | 1
5 | Chinese Taipei | 0 | 1 | 2 | 3
6 | Denmark | 0 | 1 | 0 | 1
7 | Japan | 0 | 0 | 2 | 2
8 | India | 0 | 0 | 1 | 1
9 | Spain | 0 | 0 | 1 | 1

Question:
how many times has bronze been won total?

Answer:
```
(Response)
`10 (✗)`

Tabular QA task (by GPT-4 w/ our method)
(Prompt)
Your task is to answer the question based on the table. Please show the final answer with only one or few words.

Table:

```
Rank | Nation | Gold | Silver | Bronze | Total
1 | Malaysia | 3 | 0 | 1 | 4
2 | Indonesia | 1 | 3 | 2 | 6
3 | Korea | 1 | 1 | 2 | 4
4 | Thailand | 1 | 0 | 0 | 1
5 | Chinese Taipei | 0 | 1 | 2 | 3
6 | Denmark | 0 | 1 | 0 | 1
7 | Japan | 0 | 0 | 2 | 2
8 | India | 0 | 0 | 1 | 1
9 | Spain | 0 | 0 | 1 | 1
```

Description:
The table ranks nations based on the number of gold, silver, and bronze medals they have won. Malaysia is ranked first with 3 gold medals, no silver medals, and 1 bronze medal, making a total of 4 medals. Indonesia is second with 1 gold, 3 silver, and 2 bronze medals, totaling 6 medals. Korea is third with 1 gold, 1 silver, and 2 bronze medals, totaling 4 medals. Thailand is fourth with 1 gold medal and no silver or bronze medals. Chinese Taipei is fifth with 1 silver and 2 bronze medals, totaling 3 medals. Denmark is sixth with 1 silver medal and no gold or bronze medals. Japan is seventh with 2 bronze medals and no gold or silver medals. India and Spain are ranked eighth and ninth respectively, each with 1 bronze medal and no gold or silver medals.

Question:
how many times has bronze been won total?

Answer:

(Response)
11 (✓)

We show responses by GPT-4 with and without our method for emotion analysis in figures below.

Emotion analysis task (by GPT-4)
(Prompt)
As a social scientist, Your task is to analyze eight emotions (anger, anticipation, disgust, fear, joy, sadness, surprise, trust) of the emoji. For each emotion, please assign a score from 0 to 1 according to the emoji. Please provide your best estimation of the emotion scores with two decimal places.
👉

(Response)
Anger: 0.10 (score by human: 0.06)
Anticipation: 0.60 (score by human: 0.25)
Disgust: 0.10 (score by human: 0)
Fear: 0.10 (score by human: 0)
Joy: 0.20 (score by human: 0)
Sadness: 0.10 (score by human: 0)
Surprise: 0.70 (score by human: 0.03)
Trust: 0.20 (score by human: 0.17)

Emotion analysis task (by GPT-4 w/ our method)
(Prompt)
As a social scientist, Your task is to analyze eight emotions (anger, anticipation, disgust, fear, joy, sadness, surprise, trust) of the emoji. For each emotion, please assign a score from 0 to 1 according to the emoji. Please provide your best estimation of the emotion scores with two decimal places.
👉 (backhand index pointing right)

(Response)
Anger: 0.05 (score by human: 0.06)
Anticipation: 0.20 (score by human: 0.25)
Disgust: 0.05 (score by human: 0)
Fear: 0.05 (score by human: 0)
Joy: 0.20 (score by human: 0)
Sadness: 0.05 (score by human: 0)
Surprise: 0.20 (score by human: 0.03)
Trust: 0.20 (score by human: 0.17)

We show responses by GPT-4 with and without our method for sentiment analysis in figures below.

Sentiment classification task (by GPT-4)
(Prompt)
Sentiment classification, please answer with only Positive or Negative.

aw that wee lassie made me cry tears streamin doon ma face lol thats a wee shame x

(Response)
Positive (✗)

Sentiment classification task (by GPT-4 w/ our method)
(Prompt)
Sentiment classification, please answer with only Positive or Negative.

aw that wee lassie made me cry tears streamin doon ma face lol thats a wee shame x
(Aw, that little girl made me cry. Tears streaming down my face, lol, that's a little shame x.)

(Response)
Negative (✓)