How to Make Large Language Models Generate 100% Valid Molecules?

Wen Tao[1] **Jing Tang**[2,3*] **Alvin Chan**[1] **Bryan Hooi**[4] **Baolong Bi**[5]
Nanyun Peng[6] **Yuansheng Liu**[7*] **Yiwei Wang**[8]
[1]Nanyang Technological University [2]HKUST(GZ) [3]HKUST
[4]NUS [5]UCAS [6]UCLA [7]Hunan University [8]UC Merced
taowen228@gmail.com, jingtang@ust.hk, yuanshengliu@hnu.edu.cn

Abstract

Molecule generation is key to drug discovery and materials science, enabling the design of novel compounds with specific properties. Large language models (LLMs) can learn to perform a wide range of tasks from just a few examples. However, generating valid molecules using representations like SMILES is challenging for LLMs in few-shot settings. In this work, we explore how LLMs can generate 100% valid molecules. We evaluate whether LLMs can use SELFIES, a representation where every string corresponds to a valid molecule, for valid molecule generation but find that LLMs perform worse with SELFIES than with SMILES. We then examine LLMs' ability to correct invalid SMILES and find their capacity limited. Finally, we introduce SmiSelf, a cross-chemical language framework for invalid SMILES correction. SmiSelf converts invalid **SMI**LES to **SELF**IES using grammatical rules, leveraging SELFIES' mechanisms to correct the invalid SMILES. Experiments show that SmiSelf ensures 100% validity while preserving molecular characteristics and maintaining or even enhancing performance on other metrics. SmiSelf helps expand LLMs' practical applications in biomedicine and is compatible with all SMILES-based generative models. Code is available at https://github.com/wentao228/SmiSelf.

1 Introduction

Generating novel molecules has been a fundamental and crucial problem in drug discovery and material design (Cheng et al., 2021). Advances in machine learning, particularly deep learning, have accelerated progress in this area (Zeng et al., 2022). Molecules can be represented as Simplified Molecular-Input Line-Entry System (SMILES) strings (Weininger, 1988) or SELF-referencIng Embedded Strings (SELFIES) (Krenn et al., 2020),

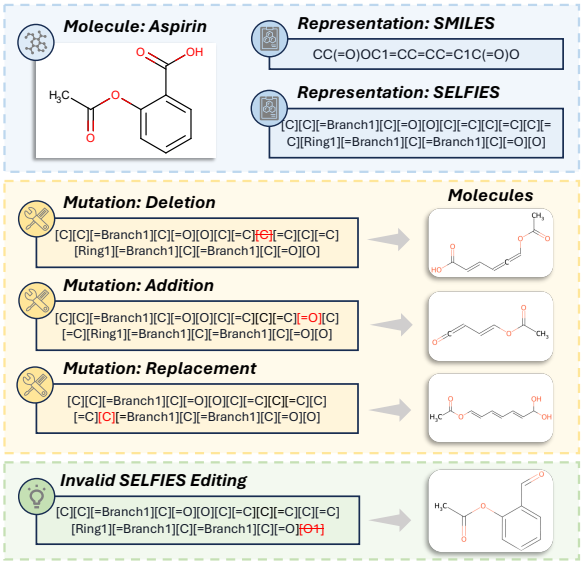


Figure 1: **Top:** SMILES and SELFIES representations of a molecule. **Middle:** Mutating the SELFIES of the molecule always results in valid molecules. **Bottom:** Proposed Invalid SELFIES Editing.

both of which are compatible with language models, as shown in Figure 1 (Top).

Prompting Large Language Models (LLMs), such as ChatGPT, with demonstrations has showcased their impressive ability to leverage extensive pretraining to perform diverse tasks (Mei et al., 2025), opening up new opportunities for efficient and effective molecule generation. For instance, as shown in Figure 2, given a molecule caption (i.e., a text description of a molecule's structure and properties), LLMs can generate the corresponding molecule. This enables more comprehensive and fine-grained control over molecule design. However, generating valid molecules using representations like SMILES is challenging due to strict syntax rules, such as the correct use of parentheses for branching, proper ring closure numbers, and adherence to atom valence limits. These constraints are difficult to convey through a few examples. For instance, as shown in Figure 2, the

*Corresponding Author

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 26565–26580
November 4-9, 2025

Text-based molecule generation via LLMs

Prompt: Given the caption of a molecule, predict the SMILES representation of the molecule.

Input Description: The molecule is a member of the class of tetralins that is tetralin substituted by methyl groups at positions 1, 1 and 6 respectively. It has a role as a metabolite. It is a member of tetralins and an ortho-fused bicyclic hydrocarbon. It derives from a hydride of a tetralin.

Output SMILES: CC1=CC2=C(C=C1)C(CCC2)(C)C)

Figure 2: An example of an invalid SMILES string generated by text-based molecule generation via LLMs.

generated SMILES string is syntactically invalid due to an extra closing parenthesis. This makes it impossible for the string to be decoded into a valid chemical structure. In contrast, SELFIES is a robust molecular representation that guarantees 100% validity, even for randomly generated strings.

In this paper, we explore and answer three questions through extensive experiments:

- ***Can LLMs use SELFIES to guarantee valid molecule generation?*** **— Yes, but at the cost of poor performance on other metrics.** Based on the robustness of SELFIES, we emphasize that SELFIES serves as a structured representation. As shown in Figure 1 (Middle), deleting, adding, or replacing symbols still yields a valid molecule. Leveraging this property, as shown in Figure 1 (Bottom), we propose Invalid SELFIES Editing, which directly employs SELFIES for molecular generation with LLMs, ensuring validity by filtering out non-alphabet symbols. However, we find that LLMs perform worse with SELFIES compared to SMILES, with SMILES being the most suitable representation for molecule generation using LLMs.

- ***Can LLMs efficiently correct the invalid SMILES they generate?*** **— No, while LLMs demonstrate potential in correcting invalid SMILES, they also face challenges in improving validity without significant degradation in other metrics.** We propose using LLMs as post-hoc invalid SMILES correctors. As shown in Algorithm 1, given invalid SMILES generated by LLMs, the model is first prompted to generate a corrected SMILES string based on the invalid SMILES and a textual description, and then uses an external tool to verify the output SMILES. The LLMs iterate this process to continuously refine the output until it becomes a valid SMILES. We find that while LLMs can correct invalid SMILES, this is accompanied by a significant reduction in other metrics, with variations in correction rates across models and error types.

- ***How can we make LLMs generate 100% valid molecules while keeping good performance on other metrics?*** **— SmiSelf, a cross-chemical language framework for invalid SMILES correction.** As shown in Figure 3, SmiSelf converts invalid SMILES generated by LLMs into SELFIES using grammatical rules, then transforms them back into SMILES, leveraging the mechanism of SELFIES to correct the invalid SMILES. Experiments demonstrate that SmiSelf guarantees 100% validity, preserves molecular characteristics, and maintains or enhances performance on other metrics.

Overall, this work provides insights into the capabilities of current LLMs and expands their practical applications in biomedicine.

2 LLMs Perform Worse With SELFIES

2.1 Molecular Representations

As shown in Figure 1 (Top), SMILES (Weininger, 1988) and SELFIES (Krenn et al., 2020) are two of the most widely used molecular representations. Like human language, the SMILES syntax enforces strict rules regarding which strings are syntactically valid. As a result, language models may generate SMILES that do not correspond to any valid chemical structure. SELFIES is a molecular string representation that guarantees 100% robustness, ensuring that every possible combination of symbols from its alphabet corresponds to a valid chemical structure.

2.2 Invalid SELFIES Editing

Based on the robustness of SELFIES, we emphasize that SELFIES serves as a structured molecular representation. As shown in Figure 1 (Middle), modifying a SELFIES string—whether by deleting a symbol, adding an alphabetic symbol, replacing a symbol, splitting the SELFIES string, or merging one SELFIES with another—always results in a

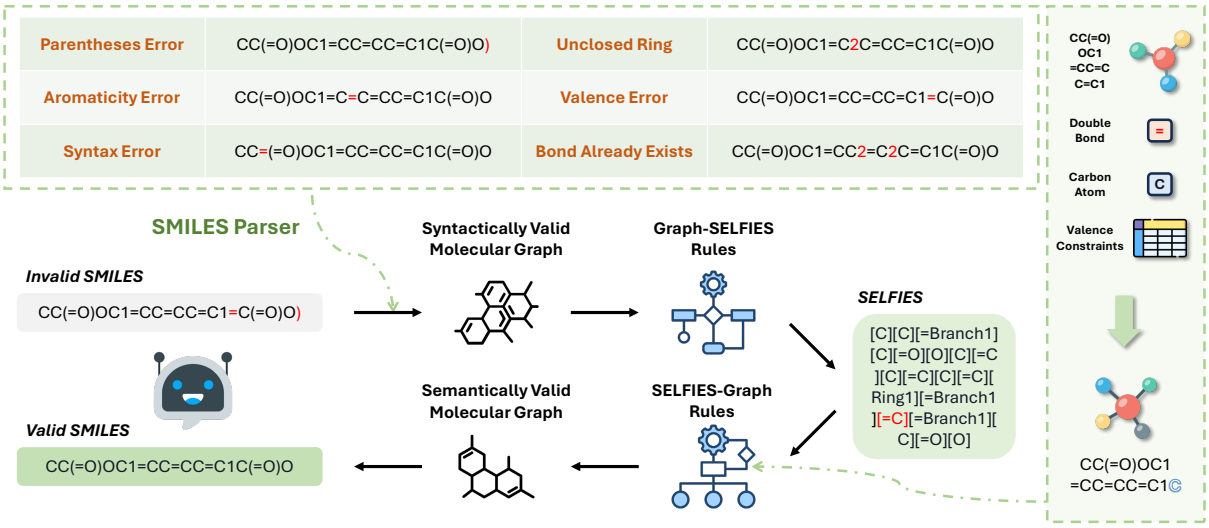


Figure 3: Overview of SmiSelf. An invalid SMILES string generated by an LLM is processed by a SMILES parser capable of handling various errors, converted into a syntactically valid molecular graph, and then transformed into a SELFIES string. The SELFIES string is re-converted into a semantically valid molecular graph, ensuring compliance with syntactic and semantic constraints, thus guaranteeing 100% validity. Finally, the molecular graph is translated back into a valid SMILES string.

valid molecule. Leveraging this property, as illustrated in Figure 1 (Bottom), we introduce Invalid SELFIES Editing. We directly use LLMs to generate SELFIES representations for molecule generation. If the generated SELFIES are invalid (i.e., if they contain symbols not in the alphabet), we perform editing (removing non-alphabetic symbols) to make the SELFIES valid, ensuring the validity of the generated molecules.

2.3 Task Description

We evaluate molecular representations for LLMs using the following two tasks: *Text-Based Molecule Generation* and *Molecule Captioning* (Edwards et al., 2022). The aim of *Text-Based Molecule Generation* is to generate molecules that match the given natural language text describing a molecule's structures and properties. *Molecule Captioning* is the reverse of text-based molecule generation, aiming to generate textual descriptions for a given molecule. Compared to the typical de novo molecule generation task (Polykovskiy et al., 2020), which aims to generate a variety of possible new molecules, these two tasks are much more challenging for deep generative models. They can assess the model's ability to understand molecules and generate them from descriptions.

2.4 Experiment Setting

We use the ChEBI-20 dataset (Edwards et al., 2021) and evaluation metrics identical to those used in MolT5 (Edwards et al., 2022). The baselines include RNN (Cho et al., 2014), Transformer (Vaswani et al., 2017), T5 (Raffel et al., 2020), MolT5 (Edwards et al., 2022), GPT-3.5, GPT-4 (Achiam et al., 2023), and LLaMA2 (Touvron et al., 2023). See Appendix B for details.

2.5 Experiment Results

As shown in Tables 1 and 2, experimental results for both tasks indicate that LLMs perform worse when using SELFIES as a molecular representation compared to SMILES. One reason for this is that SMILES was introduced much earlier than SELFIES, resulting in its much greater presence in the training data for LLMs. Evidence supporting this can be drawn from three key aspects: First, the zero-shot results of GPT-3.5 and LLaMA2-7B in text-based molecule generation demonstrate that SMILES strings are included in their pre-training corpus, as they can generate mostly valid SMILES representations of molecules based on zero-shot prompts. Second, the zero-shot performance of GPT-3.5 and LLaMA2-7B is lower compared to task-specific small-scale models, and significantly inferior to that of T5 and MolT5 in text-based molecule generation. This suggests that these LLMs have not been specifically trained on the ChEBI-20 dataset. Finally, as shown in Figure 4, citation counts over the past decade reveal that publications referencing SMILES substantially outnumber those mentioning SELFIES.

Method	#Params.	BLEU↑	EM↑	Levenshtein↓	MACCS FTS↑	RDK FTS↑	Morgan FTS↑	FCD↓	Text2Mol↑	Validity↑
RNN (task-specific)	56M	0.652	0.005	38.09	0.591	0.400	0.362	4.55	0.409	0.542
Transformer (task-specific)	76M	0.499	0.000	57.66	0.480	0.320	0.217	11.32	0.277	0.906
T5-Base (fine-tuned)	248M	0.762	0.069	24.950	0.731	0.605	0.545	2.48	0.499	0.660
T5-Large (fine-tuned)	783M	0.854	0.279	16.721	0.823	0.731	0.670	1.22	0.552	0.902
MolT5-Base	248M	0.769	0.081	24.458	0.721	0.588	0.529	2.18	0.496	0.772
MolT5-Large	783M	0.854	**0.311**	**16.071**	0.834	0.746	0.684	1.20	0.554	0.905
LLaMA2-7B (zero-shot)	7B	0.104	0.000	84.18	0.243	0.119	0.089	42.01	0.148	0.631
LLaMA2-7B (2-shot)	7B	0.693	0.022	36.77	0.808	0.717	0.609	4.90	0.149	0.761
GPT-3.5 (zero-shot)	N/A	0.489	0.019	52.13	0.705	0.462	0.367	2.05	0.479	0.802
GPT-3.5 (10-shot)	N/A	0.790	0.139	24.91	0.847	0.708	0.624	0.57	0.571	0.887
GPT-4 (10-shot)	1.76T	**0.857**	0.280	17.14	**0.903**	**0.805**	**0.739**	**0.41**	**0.593**	0.899
GPT-4-SELFIES (10-shot)	1.76T	0.682	0.179	26.596	0.756	0.624	0.541	1.666	0.468	**1.000**

Table 1: Text-based molecule generation results on ChEBI-20. The **best** scores are in bold, and the second-best scores are underlined. "N/A" indicates that the parameter size is unknown.

Methods	#Params.	BLEU-2↑	BLEU-4↑	ROUGE-1↑	ROUGE-2↑	ROUGE-L↑	METEOR↑	Text2Mol↑
RNN (task-specific)	56M	0.251	0.176	0.450	0.278	0.394	0.363	0.426
Transformer (task-specific)	76M	0.061	0.027	0.204	0.087	0.186	0.114	0.057
T5-Base (fine-tuned)	248M	0.511	0.423	0.607	0.451	0.550	0.539	0.523
T5-Large (fine-tuned)	783M	0.558	0.467	0.630	0.478	0.569	0.586	0.563
MolT5-Base	248M	0.540	0.457	0.634	0.485	0.578	0.569	0.547
MolT5-Large	783M	0.594	0.508	**0.654**	**0.510**	**0.594**	**0.614**	0.582
LLaMA2-7B (zero-shot)	7B	0.094	0.039	0.169	0.054	0.142	0.175	0.153
LLaMA2-7B (2-shot)	7B	0.489	0.409	0.535	0.374	0.472	0.495	0.466
GPT-3.5 (zero-shot)	N/A	0.103	0.050	0.261	0.088	0.204	0.161	0.352
GPT-3.5 (10-shot)	N/A	0.565	0.482	0.623	0.450	0.543	0.585	0.560
GPT-4 (10-shot)	1.76T	**0.607**	**0.525**	0.634	0.476	0.562	0.610	**0.585**
GPT-4-SELFIES (10-shot)	1.76T	0.569	0.488	0.607	0.445	0.538	0.577	0.550

Table 2: The performance of molecule captioning on ChEBI-20. The **best** scores are in bold, and the second-best scores are underlined. "N/A" indicates that the parameter size is unknown.

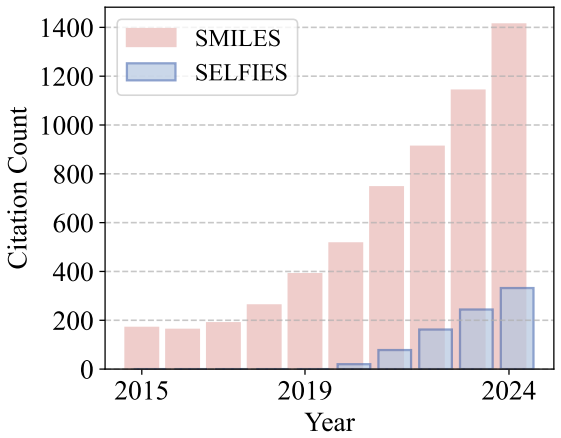


Figure 4: Comparison of Citation Counts.

Another reason lies in the inherent characteristics of the representations themselves. Several studies (Skinnider, 2024; Skinnider et al., 2021; Edwards et al., 2022; Guo et al., 2023) have shown that language models trained on SMILES outperform those trained on SELFIES. Surprisingly, although models may produce invalid molecules using the SMILES format, a significantly larger number of SELFIES was required to train a model of equivalent quality to one trained on SMILES strings (Skinnider et al., 2021).

From the experimental results, we also observe that increasing the size of the language model leads to significant performance improvements. While it is well known that scaling up model size and pretraining data generally enhances performance (Kaplan et al., 2020), it is still surprising to see that when using SMILES as the molecular representation, LLMs outperform MolT5-Large—specifically pre-trained and fine-tuned for text-based molecule generation—across most metrics, with only 10-shot in-context examples.

In summary, while our proposed Invalid SELFIES Editing ensures the validity of generated molecules, **LLMs perform worse when using SELFIES. SMILES remains the most suitable molecular representation for molecule generation using language models.**

3 LLMs Are Inefficient Invalid SMILES Correctors

In Section 5, we discuss approaches to address the validity issue in LLM-based SMILES generation and explain why they cannot fully resolve it by analyzing their limitations and comparing per-

formance. Recent research (Zhong et al., 2024) has demonstrated that LLMs can function as post-hoc correctors, proposing corrections for tasks like molecular property prediction. This section explores the question: *Can LLMs efficiently correct the invalid SMILES they generate?*

3.1 Iterative SMILES Generation

To answer the question, we propose using LLMs as post-hoc invalid SMILES correctors. Given an invalid SMILES string, an LLM is first prompted to generate a possibly valid SMILES string based on the current invalid SMILES and a textual description of the desired molecule. This output is then verified using the external tool RDKit (Landrum, 2013). This process is repeated iteratively, where the cycle of "*Correct SMILES* $\Rightarrow$ *Verify SMILES*" continues until the generated SMILES string is valid. See Algorithm 1 for a summary of the method.

3.2 Experiment Setting

We evaluate on the *Text-Based Molecule Generation* task using the ChEBI-20 dataset (Edwards et al., 2021) and evaluation metrics identical to those used in MolT5 (Edwards et al., 2022). The baseline results are 10-shot example results of GPT-3.5 and GPT-4 (Achiam et al., 2023). LLMs used as post-hoc correctors include GPT-3.5, GPT-4o-mini (Hurst et al., 2024), LLaMA2 (Touvron et al., 2023), and LLaMA3 (Grattafiori et al., 2024). See the Appendix B for further details.

Types of Errors. To assess the validity of model outputs, we used RDKit to identify invalid SMILES generated by LLMs—those that could not be converted into valid molecules. These invalid SMILES were classified into six categories based on RDKit error messages: syntax error, unclosed ring, parentheses error (extra open or close parentheses), bond already exists (dual occurrence of a bond between the same two atoms), aromaticity error (non-ring atom marked aromatic and kekulization errors), and valence error (exceeding an atom's maximum number of bonds). If strings contain multiple error types, only the first error is reported.

Correction Rate. To evaluate how effectively the model can self-correct, we introduce the correction rate, which is the ratio of valid SMILES generated after correction to the total number of invalid SMILES before correction.

Algorithm 1 LLMs as invalid SMILES correctors

Require: Input description x, initial invalid SMILES $\hat{y_0}$, prompt p, model $\mathcal{M}$, external tool $\mathcal{T}$, number of iterations n

Get initial invalid SMILES $\hat{y_0}$ ▷ Initialization
for $i \leftarrow 0$ to $n-1$ **do**
  $y_{i+1} \sim \mathbb{P}_{\mathcal{M}}(\cdot|p \oplus x \oplus y_i)$ ▷ Correction
  Verify y_{i+1} through $\mathcal{T}$ to obtain feedback f_i ▷ Verification
  if f_i indicates that y_{i+1} is valid **then** ▷ Stopping Criteria
    return y_{i+1}
  end if
end for
return $\hat{y_n}$

3.3 Experiment Results

As shown in Table 3, LLMs can improve the validity of molecules generated by them with feedback from an external tool. However, this enhancement is accompanied by a reduction in other metrics. In particular, there is a noticeable reduction in both the BLEU score and the Levenshtein score, as well as a slight reduction in other metrics. These results indicate that, while the molecules are corrected to be valid, they deviate more from the ground truth and become less aligned with the given description, despite the description being provided during the refinement process.

As shown in Figure 5, LLMs predominantly produced "parentheses error", which accounted for approximately half of all invalid SMILES. The second most common error was "valence errors", constituting 22.31% of invalid SMILES generated by GPT-3.5 and 22.42% by GPT-4.

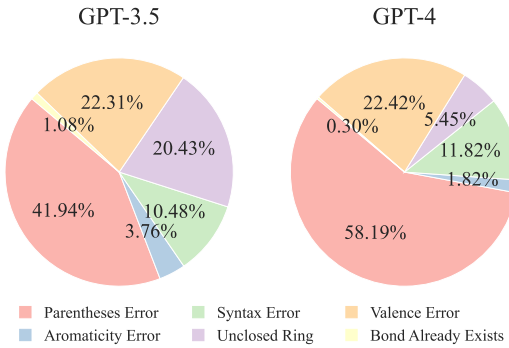
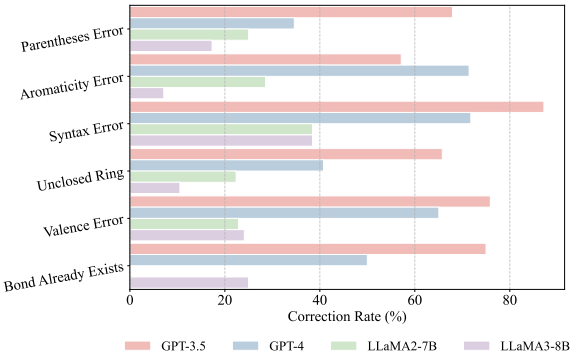


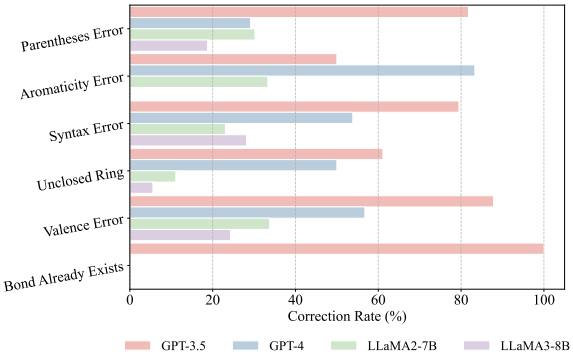
Figure 5: Distribution of error types in the invalid SMILES generated by LLMs.

As shown in Figure 6, LLMs demonstrate potential in correcting invalid SMILES. However, there are significant variations in correction rates across different models and error types. Overall, the GPT series tends to outperform the LLaMA series in correcting various errors, with GPT-3.5 notably

surpassing all other models.



(a) GPT-3.5



(b) GPT-4

Figure 6: Comparison of correction rates across different LLMs for various error types in invalid SMILES generated by GPT-3.5 and GPT-4.

In summary, for the text-based molecule generation task, **LLMs have demonstrated potential in correcting invalid SMILES strings. However, they continue to face challenges in enhancing validity while maintaining other metrics without significant degradation.** Additionally, there are notable variations in correction rates across different models and error types.

4 Making LLMs Generate 100% Valid Molecules

We present SmiSelf (invalid **SMI**LES to **SELF**IES), a cross-chemical language framework that ensures valid molecule generation through mutual conversion between two chemical languages: SMILES and SELFIES.

4.1 SmiSelf: Cross-Chemical Language for Invalid SMILES Correction

Although SELFIES is a 100% robust molecular string representation, based on our observations, it is not as suitable as SMILES for molecule generation with LLMs. We propose converting invalid SMILES generated by LLMs into SELFIES, then transforming them back into SMILES, leveraging the mechanism of SELFIES to correct the SMILES.

SMILES and SELFIES, though both string-based molecular representations, have distinct grammars. Precise conversion that preserves molecular characteristics from invalid SMILES to SELFIES cannot be fully achieved through in-context learning. To achieve this precise conversion, we use molecular graphs as intermediaries to convert between these two representations, as molecules can be represented as molecular graphs that adhere to chemical constraints. Our goal is to eliminate both syntactic and semantic errors in invalid SMILES to ensure syntactic and semantic validity. Syntactic errors involve strings that cannot be interpreted as molecular graphs, while semantic errors involve strings that form molecular graphs but violate basic chemical rules. See the Appendix E for details of the distinction between syntactic validity and semantic validity.

As shown in Figure 3, we implement a SMILES parser that converts invalid SMILES into a molecular graph. The string `CC(=O)OC1=CC=CC=C1=C(=O)O)` represents an invalid SMILES with both syntactic (extra closing parenthesis) and semantic (exceeding valence bond limits) errors. Carbon "`C`" and oxygen "`O`" atoms are parsed as nodes, connected by edges representing single (denoted by no symbol between atoms) or double (denoted by "`=`") bonds. The number "`1`" shows that the ring is formed between the two carbon atoms labeled "`C1`". Branched structures "`(=O)`" are given using brackets. We skip the extra closing parenthesis based on predefined rules and ignore the semantic error during graph construction, ensuring syntactic validity.

Next, the molecular graph is converted into a SELFIES string using the Graph-SELFIES rules (SELFIES grammar) (Krenn et al., 2020). The SELFIES string is transformed into a semantically valid molecular graph according to the SELFIES-Graph rules (Table 8). As shown in Figure 3, the molecule is constructed from the partial SELFIES string, corresponding to the SMILES string `CC(=O)OC1=CC=CC=C1`. The next SELFIES symbol, `[=C]`, adds a carbon atom with a double bond. However, this would violate valence constraints, so the bond is converted to a single bond by the SELFIES-Graph rules. Finally, the molecular graph is translated back into a SMILES

Model	BLEU↑	EM↑	Levenshtein↓	MACCS FTS↑	RDK FTS↑	Morgan FTS↑	FCD↓	Text2Mol↑	Validity↑
GPT-3.5 (10-shot)									
Baseline	**0.790**	0.139	**24.910**	**0.847**	**0.708**	**0.624**	0.570	**0.571**	0.887
+ LLM Corrector (GPT-3.5)	0.738	**0.141**	29.171	0.836	0.692	0.600	**0.537**	0.561	**0.967**
+ LLM Corrector (GPT-4o-mini)	0.753	0.140	29.018	0.837	0.693	0.606	0.550	0.563	0.942
+ LLM Corrector (LLaMA2-7B)	0.685	0.139	30.663	0.830	0.692	0.609	0.776	0.556	0.916
+ LLM Corrector (LLaMA3-8B)	0.732	0.140	31.400	0.841	0.700	0.615	0.568	0.566	0.909
GPT-4 (10-shot)									
Baseline	**0.857**	**0.280**	**17.140**	**0.903**	**0.805**	**0.739**	0.410	**0.593**	0.899
+ LLM Corrector (GPT-3.5)	0.772	0.280	22.220	0.890	0.784	0.710	**0.375**	0.582	**0.981**
+ LLM Corrector (GPT-4o-mini)	0.788	0.280	21.299	0.894	0.790	0.722	0.401	0.586	0.940
+ LLM Corrector (LLaMA2-7B)	0.718	0.280	25.034	0.886	0.787	0.722	0.562	0.578	0.929
+ LLM Corrector (LLaMA3-8B)	0.779	0.280	24.549	0.897	0.795	0.728	0.405	0.587	0.920

Table 3: Results of using LLMs as post-hoc correctors for correcting invalid SMILES in text-based molecule generation. The **best** scores are in bold.

string. These transformations ensure the resulting SMILES satisfy both syntactic and semantic constraints, guaranteeing 100% validity.

4.2 Experiment Setting

4.2.1 Text-Based Molecule Generation

For this task, we use the ChEBI-20 dataset (Edwards et al., 2021) and evaluation metrics identical to those used in MolT5 (Edwards et al., 2022). The baseline results include n-shot (0, 1, 2, 5, and 10) in-context example results of GPT-3.5 and 10-shot in-context example results of GPT-4. These are used to evaluate the performance of our proposed SmiSelf method in correcting SMILES strings generated by LLMs at varying quality levels. See the Appendix B for further details.

4.2.2 Class-Specific Molecule Generation

This task aims to generate molecules specific to a given class, based on a limited number of exemplars from that class. The dataset contains 32 Acrylates, 11 Chain Extenders, and 11 Isocyanates. For each class, 100 molecules are generated using LLMs. The evaluation metrics include: Validity, percentage of chemically valid molecules, diversity, average pairwise Tanimoto distance over Morgan fingerprints (Rogers and Hahn, 2010); Membership, and the percentage of molecules that belong to the desired monomer class.

We employ grammar prompting (Wang et al., 2024) during in-context learning to evaluate the benefits of explicitly incorporating generic SMILES grammar into LLM-based molecule generation. Grammar prompting enables LLMs to incorporate external knowledge and domain-specific constraints, expressed through a Backus–Naur Form grammar, during in-context learning.

Unlike prompting-based methods, the baseline model DEG (Guo et al., 2022) generates molecules using a graph-based grammar, which is learned through a sequence of production rules automatically derived from the training data.

4.3 Experiment Results

4.3.1 Text-Based Molecule Generation

As shown in Table 4, the molecules corrected by SmiSelf are 100% valid. However, we also observe that some metrics for the corrected molecules are worse than those for the uncorrected ones. These results are to be expected. First, the calculation of metrics—excluding BLEU, Exact Match, Levenshtein, and Validity—considers only valid SMILES, so the correction phase broadens the scope of the metrics. Second, since generating molecules from molecular descriptions is a one-to-one task with a ground truth, the process of correcting the molecules inevitably introduces some distortion, which can affect the original information and slightly reduce the metrics.

These results align with those of TGM-DLM (Gong et al., 2024), which trains a diffusion model in its second phase to correct invalid SMILES generated in the first phase. However, the performance reduction observed with our method is significantly lower across most metrics compared to the reduction caused by TGM-DLM's second phase, as well as our previously proposed Invalid SELFIES Editing and LLM Corrector. Additionally, SmiSelf improves the EM score, indicating that some invalid SMILES can exactly match the ground truth after correction, whereas TGM-DLM's EM score remains unchanged. We provide these additional comparison results in the Table 6.

Method	BLEU↑	EM↑	Levenshtein↓	MACCS FTS↑	RDK FTS↑	Morgan FTS↑	FCD↓	Text2Mol↑	Validity↑
GPT-3.5 (zero-shot)	0.489	0.019	52.13	**0.705**	**0.462**	**0.367**	2.05	**0.479**	0.802
+ SmiSelf	**0.544**	**0.020**	**46.967**	0.688	0.447	0.339	**1.986**	0.456	**1.000**
GPT-3.5 (1-shot)	**0.706**	0.074	**33.38**	**0.799**	**0.620**	**0.526**	0.84	**0.540**	0.842
+ SmiSelf	0.701	**0.075**	33.49	0.790	0.610	0.505	**0.719**	0.527	**1.000**
GPT-3.5 (2-shot)	**0.748**	0.101	**28.89**	**0.827**	**0.668**	**0.578**	0.67	**0.557**	0.860
+ SmiSelf	0.741	**0.102**	29.311	0.815	0.654	0.552	**0.604**	0.545	**1.000**
GPT-3.5 (5-shot)	**0.771**	0.121	**26.78**	**0.836**	**0.686**	**0.599**	0.60	**0.564**	0.882
+ SmiSelf	0.761	**0.122**	27.51	0.827	0.674	0.576	**0.542**	0.554	**1.000**
GPT-3.5 (10-shot)	**0.790**	0.139	**24.91**	**0.847**	**0.708**	**0.624**	0.57	**0.571**	0.887
+ SmiSelf	0.778	**0.141**	25.938	0.838	0.695	0.602	**0.492**	0.561	**1.000**
GPT-4 (10-shot)	**0.857**	0.280	**17.14**	**0.903**	**0.805**	**0.739**	0.41	**0.593**	0.899
+ SmiSelf	0.846	**0.282**	17.668	0.892	0.789	0.718	**0.312**	0.584	**1.000**

Table 4: Few-shot text-based molecule generation results on ChEBI-20, along with results corrected by SmiSelf. The **better** scores are in bold.

	Acrylates			**Chain Extenders**			**Isocyanates**		
Model	V	D	M	V	D	M	V	D	M
Graph Grammar	100	0.83	30	100	0.86	98	100	0.93	83
Standard Prompting	23	0.74	19	**100**	**0.81**	**99**	94	**0.82**	94
+ SmiSelf	**100**	**0.75**	**83**	**100**	**0.81**	**99**	**100**	**0.82**	**100**
Grammar Prompting	97	0.77	56	86	0.91	84	71	**0.83**	65
+ SmiSelf	**100**	**0.78**	**57**	**100**	**0.92**	**96**	**100**	**0.83**	**79**

Table 5: Results for class-specific molecule generation with GPT-3.5, along with results corrected by SmiSelf. The metrics are validity (V), diversity (D), and membership (M). The **better** scores are in bold.

4.3.2 Class-Specific Molecule Generation

In Table 5, we observe that applying our proposed SmiSelf method results in improvements across all metrics. This outcome can be attributed to the one-to-many nature of the task (learn the distribution of a class from a few examples and sample from it to generate multiple new molecules), and the results indicate that the molecules decoded from the corrected SMILES successfully capture the specifics of the monomer class. Notably, while standard prompting results in very low validity for acrylate molecules generated by LLMs, these molecules—once corrected by our SmiSelf method—achieve 100% validity and show significant improvement in the Membership metric. These findings suggest that although LLMs face challenges in generating valid SMILES strings, they can still capture class-specific molecular characteristics through low-shot examples. Furthermore, this highlights that our proposed SmiSelf method not only corrects invalid molecules but also preserves their molecular characteristics.

We also observe that, compared to standard prompting, grammar prompting does not consistently improve validity or other performance metrics. This suggests that explicitly incorporating generic SMILES grammar into the prompt may not provide additional benefits. Moreover, while the baseline method DEG achieves 100% validity in its generated molecules, its Membership metric across all three molecular classes is lower compared to the prompting-based methods and significantly lower than that of the molecules corrected using our SmiSelf method. This is because LLMs have encountered SMILES strings during pretraining, allowing them to acquire extensive domain knowledge about molecules. In contrast, DEG cannot incorporate external knowledge beyond the 11 or 32 molecules provided in its training data. Additionally, the high computational complexity of grammar construction limits DEG to being applied only to structurally similar low-shot molecules. Results for more baselines are in Appendix C.

5 Related Work

In this section, we introduce various methods to improve the validity of generated molecules. For a broader discussion, see Appendix A.

The existing potential solutions for generating valid SMILES with LLMs can be categorized into training-time correction, generation-time correction, and post-hoc correction (Pan et al., 2024).

Model	BLEU↑	EM↑	Levenshtein↓	MACCS FTS↑	RDK FTS↑	Morgan FTS↑	FCD↓	Text2Mol↑	Validity↑
Ground Truth	1.000	1.000	0.000	1.000	1.000	1.000	0.00	0.609	1.000
Constrained Decoding for Generation-Time Correction									
MolT5-Large	0.858	0.318	15.957	0.890	0.813	0.750	0.38	0.590	0.958
MolT5-Large-HV	0.810	0.314	16.758	0.872	0.786	0.722	0.44	0.582	0.996
	-5.59%	-1.26%	+5.02%	-2.02%	-3.32%	-3.73%	+15.79%	-1.36%	+3.97%
Training Generative Models for Post-Hoc Correction									
TGM-DLM$_{\text{w/o corr}}$	0.828	0.242	16.897	0.874	0.771	0.722	0.89	0.589	0.789
TGM-DLM	0.826	0.242	17.003	0.854	0.739	0.688	0.77	0.581	0.871
	-0.24%	0.00%	+0.63%	-2.29%	-4.15%	-4.71%	-13.48%	-1.36%	+10.39%
Our Methods for Post-Hoc Correction									
GPT-4 (10-shot)	0.857	0.280	17.14	0.903	0.805	0.739	0.41	0.593	0.899
+ Invalid SELFIES Editing	0.682	0.179	26.596	0.756	0.624	0.541	1.666	0.468	1.000
	-20.42%	-36.07%	+55.17%	-16.28%	-22.48%	-26.79%	+306.34%	-21.08%	+11.23%
GPT-4 (10-shot)	0.857	0.280	17.14	0.903	0.805	0.739	0.41	0.593	0.899
+ LLM Corrector (GPT-3.5)	0.772	0.280	22.220	0.890	0.784	0.710	0.375	0.582	0.981
	-9.92%	0.00%	+29.64%	-1.44%	-2.61%	-3.92%	-8.54%	-1.85%	+9.12%
GPT-4 (10-shot)	0.857	0.280	17.14	0.903	0.805	0.739	0.41	0.593	0.899
+ SmiSelf	0.846	0.282	17.668	0.892	0.789	0.718	0.312	0.584	1.000
	-1.28%	+0.71%	+3.08%	-1.22%	-1.99%	-2.84%	-23.90%	-1.52%	+11.23%

Table 6: Results of methods for improving the validity of text-based molecule generation, with relative improvements marked in blue and declines marked in pink.

Since training-time correction is limited by the infeasibility of fine-tuning giant closed-source LLMs, we will focus on generation-time correction and post-hoc correction.

Constrained Decoding for Generation-Time Correction. Constrained decoding is a technique used to enforce constraints on language model outputs. It restricts model outputs to adhere to predefined constraints without requiring retraining or modifications to the model architecture (Geng et al., 2023, 2024). While constrained decoding can improve the validity of molecule generation, it reduces the search space and significantly lowers other metrics (Wang et al., 2024; Edwards et al., 2022). Additionally, constrained decoding increases the number of LLM API calls.

Training Generative Models for Post-Hoc Correction. Another possible approach is training generative models to correct invalid SMILES generated by LLMs post hoc. Theoretically, invalid SMILES strings could also be corrected using translator models, as employed in the field of grammatical error correction (Yuan and Briscoe, 2016). However, this approach requires both invalid and ground-truth molecules to form input-output pairs for training, and thus may be task-specific (Gong et al., 2024; Zheng et al., 2019). Moreover, such models cannot correct 100% of invalid outputs, and the percentage of corrected outputs varies across different invalid output generators (Schoenmaker et al., 2023).

To compare our methods with these approaches, we calculate the relative improvement in text-based molecule generation. As shown in Table 6, all methods come with trade-offs. SmiSelf provides a promising approach for generating 100% valid molecules with LLMs, while keeping the performance on other metrics.

6 Conclusion

This paper studies how to ensure that the molecules generated by LLMs are 100% valid. To this end, we first propose Invalid SELFIES Editing and LLMs as post-hoc correctors. Through our experiments, we find that: 1) LLMs perform worse when using SELFIES compared to SMILES; 2) LLMs face challenges in correcting and refining the invalid SMILES they generate. We then present SmiSelf, a cross-chemical language framework for invalid SMILES correction. We propose converting invalid SMILES generated by LLMs into SELFIES and transforming them back into SMILES, leveraging the mechanism of SELFIES to correct the SMILES. Experiments demonstrate that SmiSelf effectively corrects invalid SMILES generated by LLMs, ensuring 100% validity while preserving their original molecular characteristics and maintaining or even enhancing performance on other metrics. SmiSelf helps expand the practical applications of LLMs in the biomedical domain and is compatible with all SMILES-based generative models.

Limitations

Like other post-hoc correction methods, SmiSelf introduces some distortion in the correction process for the text-based molecule generation task, which can lead to corrected molecules deviating further from the ground truth and being less aligned with the given descriptions.

Acknowledgements

We thank Kai-Wei Chang from the UCLA NLP group for the support and suggestions. This work is supported by the National Key R&D Program of China under Grant No. 2024YFA1012700 and No. 2023YFF0725100, by the National Natural Science Foundation of China (NSFC) under Grant No. 62372159, No. 62402410, and No. U22B2060, by Guangdong Provincial Project (No. 2023QN10X025), by Guangdong Basic and Applied Basic Research Foundation under Grant No. 2023A1515110131, by Guangzhou Municipal Science and Technology Bureau under Grant No. 2024A04J4454, by Guangzhou Municipal Education Bureau (No. 2024312263), by Guangzhou Industrial Information and Intelligent Key Laboratory Project (No. 2024A03J0628), by Guangzhou Municipal Key Laboratory of Financial Technology Cutting-Edge Research (No. 2024A03J0630), by NTU Start-Up Grant, and by the Ministry of Education, Singapore, under its Academic Research Fund Tier 1 (RG22/24) and Academic Research Fund Tier 2 (FY2025) (Grant MOE-T2EP20124-0009).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

Darko Butina. 1999. Unsupervised data base clustering based on daylight's fingerprint and tanimoto similarity: A fast and automated way to cluster small and large data sets. Journal of Chemical Information and Computer Sciences, 39(4):747–750.

Austin H Cheng, Andy Cai, Santiago Miret, Gustavo Malkomes, Mariano Phielipp, and Alán Aspuru-Guzik. 2023. Group selfies: a robust fragment-based molecular string representation. Digital Discovery, 2(3):748–758.

Yu Cheng, Yongshun Gong, Yuansheng Liu, Bosheng Song, and Quan Zou. 2021. Molecular design in drug discovery: a comprehensive review of deep generative models. Briefings in bioinformatics, 22(6):bbab344.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

Hanjun Dai, Yingtao Tian, Bo Dai, Steven Skiena, and Le Song. 2018. Syntax-directed variational autoencoder for structured data. In International Conference on Learning Representations.

Joseph L Durant, Burton A Leland, Douglas R Henry, and James G Nourse. 2002. Reoptimization of mdl keys for use in drug discovery. Journal of chemical information and computer sciences, 42(6):1273–1280.

Carl Edwards, Tuan Lai, Kevin Ros, Garrett Honke, Kyunghyun Cho, and Heng Ji. 2022. Translation between molecules and natural language. In 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022.

Carl Edwards, ChengXiang Zhai, and Heng Ji. 2021. Text2mol: Cross-modal molecule retrieval with natural language queries. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 595–607.

Daniel Flam-Shepherd, Kevin Zhu, and Alán Aspuru-Guzik. 2022. Language models can learn complex molecular distributions. Nature Communications, 13(1):3293.

Saibo Geng, Berkay Döner, Chris Wendler, Martin Josifoski, and Robert West. 2024. Sketch-guided constrained decoding for boosting blackbox large language models without logit access. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 234–245.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. 2023. Grammar-constrained decoding for structured nlp tasks without finetuning. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 10932–10952.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. 2018. Automatic chemical design using a data-driven continuous representation of molecules. ACS central science, 4(2):268–276.

Haisong Gong, Qiang Liu, Shu Wu, and Liang Wang. 2024. Text-guided molecule generation with diffusion language model. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pages 109–117.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. arXiv preprint arXiv:2407.21783.

Minghao Guo, Veronika Thost, Beichen Li, Payel Das, Jie Chen, and Wojciech Matusik. 2022. Data-efficient graph grammar learning for molecular generation. In International Conference on Learning Representations.

Taicheng Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh Chawla, Olaf Wiest, Xiangliang Zhang, and 1 others. 2023. What can large language models do in chemistry? a comprehensive benchmark on eight tasks. Advances in Neural Information Processing Systems, 36:59662–59688.

Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. 2022. Equivariant diffusion for molecule generation in 3d. In International conference on machine learning, pages 8867–8887. PMLR.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. arXiv preprint arXiv:2410.21276.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2018. Junction tree variational autoencoder for molecular graph generation. In International conference on machine learning, pages 2323–2332. PMLR.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2020. Hierarchical generation of molecular graphs using structural motifs. In International conference on machine learning, pages 4839–4848. PMLR.

Hiroshi Kajino. 2019. Molecular hypergraph grammar with its application to molecular optimization. In International Conference on Machine Learning, pages 3183–3191. PMLR.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

Seojin Kim, Jaehyun Nam, Sihyun Yu, Younghoon Shin, and Jinwoo Shin. 2024. Data-efficient molecular generation with hierarchical textual inversion. In Proceedings of the 41st International Conference on Machine Learning, pages 24392–24414.

Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. 2020. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. Machine Learning: Science and Technology, 1(4):045024.

Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. 2017. Grammar variational autoencoder. In International conference on machine learning, pages 1945–1954. PMLR.

Greg Landrum. 2013. Rdkit documentation. Release, 1(1-79):4.

Jiatong Li, Yunqing Liu, Wenqi Fan, Xiao-Yong Wei, Hui Liu, Jiliang Tang, and Qing Li. 2024. Empowering molecule discovery for molecule-caption translation with large language models: A chatgpt perspective. IEEE Transactions on Knowledge and Data Engineering.

Youzhi Luo, Keqiang Yan, and Shuiwang Ji. 2021. Graphdf: A discrete flow model for molecular graph generation. In International conference on machine learning, pages 7192–7203. PMLR.

Andreas Mayr, Günter Klambauer, Thomas Unterthiner, Marvin Steijaert, Jörg K Wegner, Hugo Ceulemans, Djork-Arné Clevert, and Sepp Hochreiter. 2018. Large-scale comparison of machine learning methods for drug target prediction on chembl. Chemical science, 9(24):5441–5451.

Lingrui Mei, Jiayu Yao, Yuyao Ge, Yiwei Wang, Baolong Bi, Yujun Cai, Jiazhi Liu, Mingyu Li, Zhong-Zhi Li, Duzhen Zhang, Chenlin Zhou, Jiayi Mao, Tianze Xia, Jiafeng Guo, and Shenghua Liu. 2025. A survey of context engineering for large language models. Preprint, arXiv:2507.13334.

Noel O'Boyle and Andrew Dalke. 2018. Deepsmiles: an adaptation of smiles for use in machine-learning of chemical structures.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2024. Automatically correcting large language models: Surveying the landscape of diverse automated correction strategies. Transactions of the Association for Computational Linguistics, 12:484–506.

Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, and 1 others. 2020. Molecular sets (moses): a benchmarking platform for molecular generation models. Frontiers in pharmacology, 11:565644.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of machine learning research, 21(140):1–67.

Stephen Robertson, Hugo Zaragoza, and 1 others. 2009. The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends® in Information Retrieval, 3(4):333–389.

David Rogers and Mathew Hahn. 2010. Extended-connectivity fingerprints. Journal of chemical information and modeling, 50(5):742–754.

Nadine Schneider, Roger A Sayle, and Gregory A Landrum. 2015. Get your atoms in order an open-source implementation of a novel and robust molecular canonicalization algorithm. Journal of chemical information and modeling, 55(10):2111–2120.

Linde Schoenmaker, Olivier JM Béquignon, Willem Jespers, and Gerard JP van Westen. 2023. Uncorrupt smiles: a novel approach to de novo design. Journal of Cheminformatics, 15(1):22.

Michael A Skinnider. 2024. Invalid smiles are beneficial rather than detrimental to chemical language models. Nature Machine Intelligence, 6(4):437–448.

Michael A Skinnider, R Greg Stacey, David S Wishart, and Leonard J Foster. 2021. Chemical language models enable navigation in sparsely populated chemical space. Nature Machine Intelligence, 3(9):759–770.

Teague Sterling and John J Irwin. 2015. Zinc 15–ligand discovery for everyone. Journal of chemical information and modeling, 55(11):2324–2337.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288.

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

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. 2023. Digress: Discrete denoising diffusion for graph generation. In The Eleventh International Conference on Learning Representations.

Bailin Wang, Zi Wang, Xuezhi Wang, Yuan Cao, Rif A Saurous, and Yoon Kim. 2024. Grammar prompting for domain-specific language generation with large language models. Advances in Neural Information Processing Systems, 36.

David Weininger. 1988. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. Journal of chemical information and computer sciences, 28(1):31–36.

Minkai Xu, Alexander S Powers, Ron O Dror, Stefano Ermon, and Jure Leskovec. 2023. Geometric latent diffusion models for 3d molecule generation. In International Conference on Machine Learning, pages 38592–38610. PMLR.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In Proceedings of the 2016 conference of the north American Chapter of the Association for computational linguistics: Human language technologies, pages 380–386.

Zheni Zeng, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2022. A deep-learning system bridging molecule structure and biomedical text with comprehension comparable to human professionals. Nature communications, 13(1):862.

Shuangjia Zheng, Jiahua Rao, Zhongyue Zhang, Jun Xu, and Yuedong Yang. 2019. Predicting retrosynthetic reactions using self-corrected transformer neural networks. Journal of chemical information and modeling, 60(1):47–55.

Zhiqiang Zhong, Kuangyu Zhou, and Davide Mottin. 2024. Harnessing large language models as post-hoc correctors. arXiv preprint arXiv:2402.13414.

A Broader Related Work

Molecule Generation. Existing methods can be categorized according to their molecular representation. Molecules, for example, can be treated as chemical graphs (Luo et al., 2021; Vignac et al., 2023), combinations of substructures (Jin et al., 2018), or three-dimensional objects (Hoogeboom et al., 2022; Xu et al., 2023). However, these approaches have yet to consistently surpass the earlier chemical language models (Gómez-Bombarelli et al., 2018; Flam-Shepherd et al., 2022). These models represent molecules as text strings, typically using the SMILES (Weininger, 1988) or SELFIES (Krenn et al., 2020) formats.

Validity of Generated Molecules. SMILES (Weininger, 1988) strings have been a prominent molecular representation since they were invented. However, the SMILES representation is not inherently robust, meaning that generative models are likely to produce strings that do not represent valid molecules. A large body of work has been dedicated to addressing this issue in recent years, whether by developing alternative textual representations of molecules (O'Boyle and Dalke, 2018; Krenn et al., 2020; Cheng et al., 2023), methods that generate valid SMILES by design (Kusner et al., 2017; Dai et al., 2018), or techniques to correct invalid SMILES post hoc (Schoenmaker et al., 2023; Zheng et al., 2019; Kim et al., 2024; Gong et al., 2024).

B Molecule-Caption Generation

B.1 Evaluation Metrics

BLEU (Bilingual Evaluation Understudy) measures the similarity between generated and reference texts (e.g., molecule captions). Higher is better.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) measures overlap between generated and reference molecule captions. Higher is better.

METEOR (Metric for Evaluation of Translation with Explicit ORdering) measures similarity between generated and reference molecule captions, considering precision, recall, synonyms, and word order. Higher is better.

EM (Exact Match) checks if the generated molecule exactly matches the ground truth. Higher is better.

Levenshtein (Edit Distance) measures the minimum number of insertions, deletions, or substitutions needed to convert one string to another. Lower is better.

MACCS FTS (MACCS Fingerprint Tanimoto Similarity) measures Tanimoto similarity between target and generated molecules using MACCS fingerprints (Durant et al., 2002). Higher is better.

RDK FTS (RDKit Fingerprint Tanimoto Similarity) is similar to MACCS FTS but uses RDKit fingerprints (Schneider et al., 2015). Higher is better.

Morgan FTS (Morgan Fingerprint Tanimoto Similarity) measures Tanimoto similarity of target and generated molecules using Morgan fingerprints (Rogers and Hahn, 2010). Higher is better.

FCD (Fréchet ChemNet Distance) measures the distance between generated and target molecule distributions using ChemNet (Mayr et al., 2018). Lower is better.

Text2Mol measures relevance between textual descriptions and generated molecules using the Text2Mol model (Edwards et al., 2021). Higher is better.

Validity measures whether generated strings are valid chemical representations using RDKit (Landrum, 2013). Higher is better.

B.2 Datasets

We utilize the ChEBI-20 dataset (Edwards et al., 2021), which contains 33,010 molecule-caption pairs. The dataset is split into 80% for training, 10% for validation, and 10% for testing. For in-context learning, the training set serves as a local database to retrieve n-shot examples.

B.3 Baselines

RNN. RNN-GRU (Cho et al., 2014) with a 4-layer bidirectional encoder, trained from scratch on ChEBI-20.

Transformer. A vanilla Transformer (Vaswani et al., 2017) with six encoder–decoder layers, trained from scratch on ChEBI-20.

T5. A model based on T5 (Raffel et al., 2020), pre-trained on C4 and directly fine-tuned on ChEBI-20, with small, base, and large variants. No molecular knowledge is used in pre-training.

MolT5. MolT5 (Edwards et al., 2022) is initialized from pre-trained T5, jointly pre-trained on ZINC-15 SMILES (Sterling and Irwin, 2015) and C4 text (Raffel et al., 2020), and then fine-tuned on ChEBI-20. It is available in small, base, and large sizes.

LLMs. GPT-3.5 (GPT-3.5-Turbo), GPT-4 (GPT-4-0314) (Achiam et al., 2023), and GPT-4o-mini (Hurst et al., 2024) are accessed via the OpenAI API. The open-source LLMs LLaMA2-7B (Touvron et al., 2023) and LLaMA3-8B (Grattafiori et al., 2024) are used without fine-tuning. Inputs follow the five-part structure of (Li et al., 2024): role, task, examples, output instruction, and user prompt, with examples retrieved using BM25 (Robertson et al., 2009) (text-based molecule generation) or Morgan Fingerprint (Butina, 1999) similarity (molecule captioning).

C Class-Specific Molecule Generation

We compare our method with four baselines for class-specific molecule generation: JT-VAE (Jin et al., 2018), HierVAE (Jin et al., 2020), MHG (Kajino, 2019), and DEG (Guo et al., 2022), as shown in Table 7.

From the results, we observe that the vocabulary-based method JT-VAE fails to extract a vocabulary that enables it to generate diverse molecules on small datasets. HierVAE, another vocabulary-based method with a more diverse vocabulary, addresses this limitation; however, its low membership scores indicate that it does not capture class-specific characteristics. Among grammar-based methods, MHG employs fine-grained rules that simply attach atoms, resulting in high diversity. Nevertheless, these rules fail to capture domain-specific characteristics when compared to another

Model	Acrylates			Chain Extenders			Isocyanates		
	V	D	M	V	D	M	V	D	M
Task-Specific									
JT-VAE	100	0.29	49	100	0.62	80	100	0.72	67
HierVAE	100	0.83	1	100	0.83	44	100	0.83	0
MHG	100	0.89	1	100	0.90	41	100	0.88	12
DEG	100	0.83	30	100	0.86	98	100	0.93	83
Prompting + SmiSelf									
GPT-3.5	100	0.75	83	100	0.81	99	100	0.82	100

Table 7: Results for class-specific molecule generation. The metrics are validity (V), diversity (D), and membership (M). Higher is better for all metrics.

grammar-based method, DEG.

Overall, the results demonstrate that molecules generated using the prompting-based method and subsequently corrected with our proposed SmiSelf successfully capture class-specific features and consistently achieve stable performance. These findings clearly distinguish our approach from the baselines.

D SMILES vs. SELFIES

SMILES (Simplified Molecular-Input Line-Entry System) (Weininger, 1988) is the de facto standard representation in cheminformatics. In SMILES, molecules are represented as a chain of atoms, written as letters in a string. Branches in the molecule are enclosed in parentheses, while ring closures are indicated by two matching numbers. Although the SMILES grammar is simple, it allows for the description of complex structures, as well as properties such as stereochemistry. However, SMILES lacks a mechanism to ensure that molecular strings are valid in terms of both syntax and physical principles.

SELFIES (SELF-referencIng Embedded Strings) (Krenn et al., 2020), on the other hand, is a 100% robust molecular string representation. That is, SELFIES cannot produce an invalid molecule, as every combination of symbols in the SELFIES alphabet corresponds to a chemically valid graph. SELFIES is a formal grammar with derivation rules (Table 8). It can be understood as a small computer program with minimal memory that guarantees 100% robust derivation. The SELFIES grammar is specifically designed to eliminate both syntactically and semantically invalid molecules, which is especially important in generative tasks.

E Syntactic Validity vs. Semantic Validity

Syntactic validity refers to whether the string conforms to specific syntactic rules and can be parsed into a molecular graph. For example, the SMILES string `C#C=C` is syntactically valid because it adheres to SMILES syntax rules.

Semantic validity refers to whether the molecular graph represented by the string adheres to fundamental chemical rules, such as the valence rules for atoms. For example, the SMILES string `C#C=C` is semantically invalid because the middle carbon (bonded via # and =) exceeds carbon's maximum valency of 4.

A syntactically invalid string is always semantically invalid because it cannot be parsed into a molecular graph and therefore cannot be assessed for semantic validity.

We provide examples of three possible cases:

- *Syntactically invalid:* The SMILES string `C#C=C)` is syntactically invalid because of the non-matched `)`.
- *Syntactically valid but semantically invalid:* The SMILES string `C#C=C` is syntactically valid, but the middle carbon (bonded via # and =) exceeds carbon's maximum valency of 4, which violates chemical rules and is therefore semantically invalid.
- *Both syntactically and semantically valid:* The SMILES string `C=C=C` is both syntactically and semantically valid, representing a molecule that adheres to both syntactic and chemical rules.

F Fine-tuning vs. SmiSelf

Although fine-tuning can be applied to achieve higher validity and improve other metrics, we would like to highlight several crucial factors to consider when deciding whether to use it:

State	[ϵ]	[F]	[=O]	[#N]	[O]	[N]	[=N]	[C]	[=C]	[#C]	[Branch1]	[Branch2]	[Branch3]	[Ring]
X_0	X_0	F X_1	O X_2	N X_3	O X_2	N X_3	N X_3	C X_4	C X_4	C X_4	ign X_0	ign X_0	ign X_0	ign X_0
X_1	ϵ	F	O	N	O X_1	N X_2	N X_2	C X_3	C X_3	C X_3	ign X_1	ign X_1	ign X_1	R(N)
X_2	ϵ	F	=O	=N	O X_1	N X_2	=N X_1	C X_3	=C X_2	=C X_2	B(N, X_5)X_1	B(N, X_5)X_1	B(N, X_5)X_1	R(N) X_1
X_3	ϵ	F	=O	#N	O X_1	N X_2	=N X_1	C X_3	=C X_2	#C X_1	B(N, X_5)X_2	B(N, X_6)X_1	B(N, X_5)X_2	R(N) X_2
X_4	ϵ	F	=O	#N	O X_1	N X_2	=N X_1	C X_3	=C X_2	#C X_1	B(N, X_5)X_3	B(N, X_7)X_1	B(N, X_6)X_2	R(N) X_3
X_5	C	F	O	N	O X_1	N X_2	N X_2	C X_3	C X_3	C X_3	X_5	X_5	X_5	X_5
X_6	C	F	=O	=N	O X_1	N X_2	=N X_1	C X_3	=C X_2	=C X_2	X_6	X_6	X_6	X_6
X_7	C	F	=O	#N	O X_1	N X_2	=N X_1	C X_3	=C X_2	#C X_1	X_7	X_7	X_7	X_7
N	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Table 8: Derivation rules of SELFIES for small organic molecules.

- availability of training data
- computational cost of fine-tuning
- time cost of fine-tuning
- performance improvement
- feasibility of training LLMs

In contrast, our proposed SmiSelf:

- does not require training data
- eliminates the computational cost of fine-tuning, with only a small overhead
- is rapid
- ensures 100% validity while preserving molecular characteristics and maintaining or even enhancing performance on other metrics
- is compatible with all SMILES-based generative models

G Prompts

Prompt for text-based molecule generation:

```
# System Prompt

You are now working as an excellent expert in chemisrty and drug
discovery. Given the caption of a molecule, your job is to predict the
SMILES representation of the molecule. The molecule caption is a
sentence that describes the molecule, which mainly describes the
molecule's structures, properties, and production. You can infer the
molecule SMILES representation from the caption.

Example 1:
```
Instruction: Given the caption of a molecule, predict the SMILES
representation of the molecule.
Input: The molecule is a steroid ester that is pregn-4-en-21-yl acetate
substituted by oxo group at positions 3 and 20, a methyl group at
position 6 and hydroxy groups at positions 11 and 17 respectively. It is
a 3-oxo-Delta(4) steroid, a steroid ester, an 11beta-hydroxy steroid, a
17alpha-hydroxy steroid, a 20-oxo steroid and a tertiary alpha-hydroxy
ketone. It derives from a hydride of a pregnane.
```

Your output should be:
```
{"molecule":
"C[C@H]1C[C@H]2[C@@H]3CC[C@@](C[C@]3(C[C@@H]([C@@H]2[C@@]4(C1=CC(=O)CC4)C)
O)C)(C(=O)COC(=O)C)O"}
```
```

Your response should only be in the exact JSON format above; THERE
SHOULD BE NO OTHER CONTENT INCLUDED IN YOUR RESPONSE.

User Prompt

Input: The molecule is a steroid ester that is methyl
(17E)-pregna-4,17-dien-21-oate substituted by oxo groups at positions 3
and 11. It is a 3-oxo-Delta(4) steroid, an 11-oxo steroid, a steroid
ester and a methyl ester. It derives from a hydride of a pregnane.
```

## Prompt for molecule captioning:

```
System Prompt

You are now working as an excellent expert in chemisrty and drug
discovery. Given the SMILES representation of a molecule, your job is to
predict the caption of the molecule. The molecule caption is a sentence
that describes the molecule, which mainly describes the molecule's
structures, properties, and production.

Example 1:
```
Instruction: Given the SMILES representation of a molecule, predict the
caption of the molecule.
Input: C[C@]12CCC(=O)C=C1CC[C@@H]3[C@@H]2C(=O)C[C@]4([C@H]3CCC4=O)C
```

Your output should be:
```
{"caption": "The molecule is a 3-oxo Delta(4)-steroid that is
androst-4-ene carrying three oxo-substituents at positions 3, 11 and 17.
It has a role as an androgen, a human urinary metabolite, a marine
metabolite and an EC 1.1.1.146 (11beta-hydroxysteroid dehydrogenase)
inhibitor. It is a 3-oxo-Delta(4) steroid, a 17-oxo steroid, an
androstanoid and an 11-oxo steroid. It derives from a hydride of an
androstane."}
```

Your response should only be in the JSON format above; THERE SHOULD BE
NO OTHER CONTENT INCLUDED IN YOUR RESPONSE.

User Prompt

Input:
C[C@]12CCC(=O)C=C1CC[C@@H]3[C@@H]2C(=O)C[C@]\\4([C@H]3CC/C4=C/C(=O)OC)C
```

## Prompt for LLMs as correctors:

```
System Prompt

You are now working as an excellent expert in chemisrty and drug
discovery. Given the invalid SMILES representation and the caption of a
molecule, your job is to predict the valid SMILES representation of the
molecule. The molecule caption is a sentence that describes the
molecule, which mainly describes the molecule's structures, properties,
and production. You can infer the molecule SMILES representation from
the caption.

Task Format
```
Instruction: Given the invalid SMILES representation and the caption of
a molecule, predict the valid SMILES representation of the molecule.
Input:
Invalid SMILES Representation: [INVALID_SMILES_REPRESENTATION_MASK]
Caption: [CAPTION_MASK]
```

Your output should be:
```
{"molecule": "[VALID_SMILES_REPRESENTATION_MASK]"}
```
```
```

```
Your response should only be in the exact JSON format above; THERE
SHOULD BE NO OTHER CONTENT INCLUDED IN YOUR RESPONSE.

# User Prompt

Input:
Invalid SMILES Representation:
C[C@H]1[C@H]([C@H]([C@@H]([C@@H](O1)O[C@@H]2[C@H]([C@H]([C@H](O[C@H]2O)
CO)O[C@H]3[C@@H]([C@H]([C@@H]([C@H](O3)CO)O)O)NC(=O)C)O)O)NC(=O)C)O)O
Caption: The molecule is a branched amino tetrasaccharide consisting of
N-acetyl-beta-D-glucosamine having two alpha-L-fucosyl residues at the
3- and 6-positions as well as an N-acetyl-beta-D-glucosaminyl residue at
the 4-position. It has a role as a carbohydrate allergen. It is a
glucosamine oligosaccharide and an amino tetrasaccharide. It derives
from an alpha-L-Fucp-(1->3)-[alpha-L-Fucp-(1->6)]-beta-D-GlcpNAc.
```

Prompt for class-specific molecule generation:

```
You are an expert in chemistry. You are given a list of acrylates
molecules in SMILES format. You are asked to write another acrylates
molecule in SMILES format.
Molecule: C=CC(=O)OCCCCCCOC(=O)C=C
Molecule: CCCCCCOC(=O)C=C
Molecule: CCCOC(=O)C(=C)C
Molecule: CCC(C)OC(=O)C(=C)C
Molecule: CCC(COCCCOC(=O)C=C)(COCCCOC(=O)C=C)COCCCOC(=O)C=C
Molecule: C=CC(=O)OC1=CC=CC=C1
Molecule: CCC(C)OC(=O)C=C
Molecule: CCCCCCCCOC(=O)C(=C)C
Molecule: C=CC(=O)OC1=C(C(=C(C(=C1F)F)F)F)F
Molecule: CC(=C)C(=O)OCCOC1=CC=CC=C1
Molecule: CCCCCCCCCCCCOC(=O)C(=C)C
Molecule: CC(=C)C(=O)OC
Molecule:
C=CC(=O)OCC(CO)(COCC(COC(=O)C=C)(COC(=O)C=C)COC(=O)C=C)COC(=O)C=C
Molecule: CC(C)CCCCCCCOC(=O)C=C
Molecule: CCOCCOC(=O)C(=C)C
Molecule: C=CC(=O)OCC1=CC=CC=C1
Molecule: CCCCOC(=O)C=C
Molecule: CCC(COCC(CC)(COC(=O)C=C)COC(=O)C=C)(COC(=O)C=C)COC(=O)C=C
Molecule: CC(=C)C(=O)OCC1=CC=CC=C1
Molecule: CC1CC(CC(C1)(C)C)OC(=O)C(=C)C
Molecule: COC(=O)C=C
Molecule: CC(=C)C(=O)OC1CC2CCC1(C2(C)C)C
Molecule: CCCOC(=O)C=C
Molecule: COCCOC(=O)C=C
Molecule: C=CC(=O)OCCC1=CC=CC=C1
Molecule:
C=CC(=O)OCC(COCC(COC(=O)C=C)(COC(=O)C=C)COC(=O)C=C)(COC(=O)C=C)COC(=O)C=C
Molecule: CC(=C)C(=O)OC1=CC=CC=C1
Molecule: CCCCC(CC)COC(=O)C(=C)C
Molecule: CC(C)(COCCCOC(=O)C=C)COCCCOC(=O)C=C
Molecule: C=CC(=O)OCC(CO)(COC(=O)C=C)COC(=O)C=C
Molecule: CCCCOCCOC(=O)C(=C)C
Molecule: CC(C)COC(=O)C(=C)C
Molecule:
```