Inconsistency of LLMs in Molecular Representations

Bing Yan,* Angelica Chen, and Kyunghyun Cho*

Department of Computer Science, New York University, New York

E-mail: bing.yan@nyu.edu; kyunghyun.cho@nyu.edu

Abstract

Large language models (LLMs) have shown promising potential across diverse chemistry tasks, including forward reaction prediction, retrosynthesis, and property prediction. However, their ability to capture the intrinsic chemistry of molecules remains unclear. To study this, we evaluate the consistency of state-of-the-art LLMs when using different molecular representations, such as SMILES strings and IUPAC names. Our results reveal strikingly low consistency rates of below 1% for commercial state-of-the-art LLMs.

To cope with the imbalance in molecular representation in the training data, we finetune the models using data represented in both SMILES and IUPAC, but the models still produce inconsistent predictions. To address this, we regularize training by a sequence-level, symmetric Kullback-Leibler (KL) divergence loss. Although the proposed KL divergence loss improves surface-level consistency, it does not lead to better accuracy, due to the apparent orthogonality between consistency and accuracy, suggesting that these models do not understand chemistry, as we expect them to. These findings point to the inherent limitations of recent LLMs and the need for more advanced approaches that encourage these LLMs to capture intrinsic chemistry, resulting in both accurate and consistent predictions.

Introduction

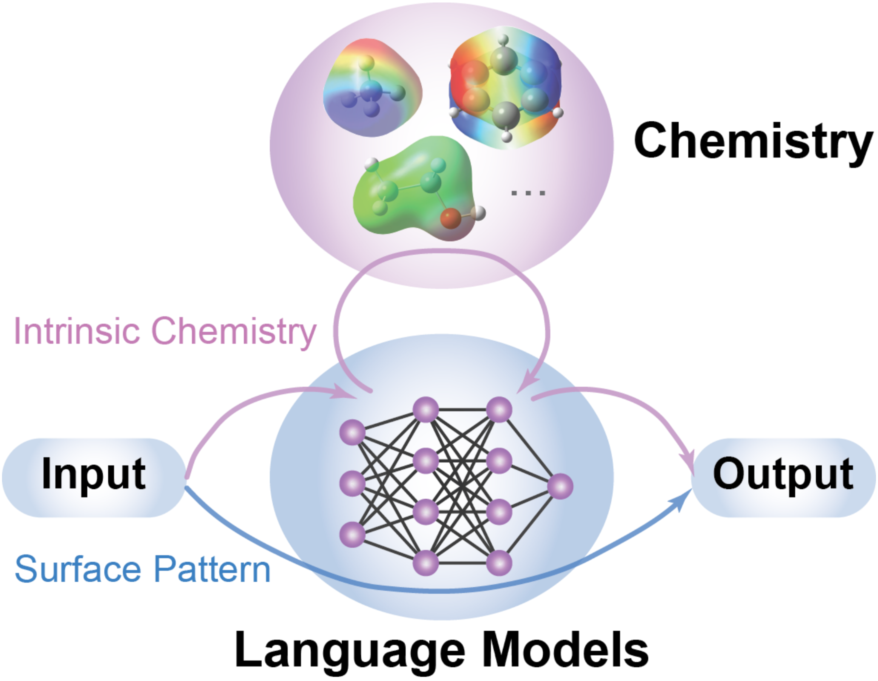


Figure 1: Illustration of how language models approach predictions for chemistry tasks. It remains unclear whether their predictions rely on surface-level patterns in molecular representations (blue pathway) or on the intrinsic chemical properties (pink pathway) of the molecules.

Large language models (LLMs) have achieved remarkable success in answering chemistry-related questions and performing tasks such as reaction prediction and property estimation[1–6]. While the potential of LLMs in chemistry reasoning is exciting, a fundamental question remains: do LLMs truly understand the underlying chemistry of molecules[7–10]?

Molecules can be represented in various forms—1D strings, 2D graphs, or 3D coordinates—but their intrinsic chemical nature, such as the atomic composition, electronic structure, and spatial arrangement, remains invariant across representations. LLMs predominantly operate on 1D representations, such as SMILES strings[11], due to their simplicity and compatibility with LLM architectures, which excel at processing sequential, tokenized data[12,13]. However, this raises a critical concern: are LLMs merely learning surface-level patterns embedded in these string-based formats (Figure 1, blue pathway), or are they capturing the intrinsic chemical properties that govern molecular behavior (Figure 1, pink pathway)?

To explore this, we evaluate the consistency of LLMs in chemistry tasks across different string-based molecular representations, focusing on two widely used formats: SMILES strings and IUPAC names[14]. Our findings reveal that LLMs exhibit strikingly low consistency between these representations, even when trained on carefully curated, one-to-one mapped datasets. To address this limitation, we introduce a sequence-level Kullback–Leibler (KL) divergence[15] loss during training, aimed at encouraging LLMs to produce more consistent outputs across representations. While the KL divergence loss improves consistency, it also amplifies consistently incorrect predictions. Further analysis demonstrates the orthogonality between consistency and accuracy. These results suggest that LLMs still struggle with fully capturing the fundamental chemistry underlying molecular representations.

These findings underscore the limitations of current LLM architectures and the pressing need for more advanced models that can understand the invariant properties of molecules. Such models would enable seamless integration of diverse molecular datasets, ensure reliable performance across representations, and enhance versatility in chemistry applications.

Experiments

Problem setup

In this work, we study three different tasks, forward reaction prediction, retrosynthesis, and property prediction. Each task can be formulated as a conditional generation problem: given an input sequence x, predict the output sequence y. The tasks are defined as follows:

1. Forward reaction prediction: the input x consists of the reactants and reagents, and the output y is the predicted product.

2. Retrosynthesis: the input x is the target product, and the output y comprises the reactants.

3. Property prediction: the input x is a single molecule, and the output y is the predicted property, which can be (1) a binary classification (True/False), for tasks such as blood-brain barrier penetration (BBBP), toxicity to humans (ClinTox), HIV replication inhibition (HIV), and drug side effects (SIDER), or (2) a continuous numeric value, for properties such as water solubility (ESOL) or the octanol/water distribution coefficient (LIPO).

For all tasks, we employ language models to predict the output sequence distribution $P_\theta(y|x)$ where θ denotes the model parameters.

The input sequence x, which may represent one or multiple molecules, can be encoded in different formats, such as SMILES strings or IUPAC names. These varying input representations can result in different output distributions, $P_\theta(y|x_\mathrm{S})$ for SMILES and $Q_\theta(y|x_\mathrm{I})$ for IUPAC. By evaluating the consistency between these distributions, we aim to assess whether language models can capture the intrinsic chemistry underlying these symbolic representations.

Consistency

Consistency measures how often the model generates identical outputs when provided with different molecular representations as input.

1. Forward reaction prediction and retrosynthesis: For a given input format, the model is tested with generating outputs in both SMILES and IUPAC representations. For SMILES input (x_S), the model generates SMILES ($y_\mathrm{S}^{x_\mathrm{S}}$) and IUPAC outputs ($y_\mathrm{I}^{x_\mathrm{S}}$); for IUPAC input ($x_\mathrm{I}$), the model generates SMILES ($y_\mathrm{S}^{x_\mathrm{I}}$) and IUPAC output (($y_\mathrm{I}^{x_\mathrm{I}}$)).

   The outputs from different input representations are considered "match" if identical:

$$\begin{aligned} \mathrm{MATCH_S} &= \mathbb{1}[y_\mathrm{S}^{x_\mathrm{S}} = y_\mathrm{S}^{x_\mathrm{I}}] \\ \mathrm{MATCH_I} &= \mathbb{1}[y_\mathrm{I}^{x_\mathrm{S}} = y_\mathrm{I}^{x_\mathrm{I}}] \end{aligned} \quad (1)$$

$\mathbb{1}[\cdot]$ is the indicator function which returns 1 if the condition inside is true and 0 otherwise. The consistency score for a single entry is the average of SMILES and IUPAC matches. For a dataset of N entries, the overall consistency is calculated as:

$$\begin{aligned}\text{Consistency(overall)} &= \frac{1}{2N}\sum_{i=1}^{N}(\text{MATCH}_{\text{S},i} + \text{MATCH}_{\text{I},i}) \\ &= \frac{1}{2N}\sum_{i=1}^{N}(\mathbb{1}[y_{\text{S},i}^{x_\text{S}} = y_{\text{S},i}^{x_\text{I}}] + \mathbb{1}[y_{\text{I},i}^{x_\text{S}} = y_{\text{I},i}^{x_\text{I}}])\end{aligned} \tag{2}$$

We also compute the false consistency, defined as the consistency of entries that produce incorrect predictions from both SMILES and IUPAC inputs. For M such entries, the false consistency is:

$$\text{Consistency(false)} = \frac{1}{2M}\sum_{i=1}^{M}(\mathbb{1}[y_{\text{S},i}^{x_\text{S}} = y_{\text{S},i}^{x_\text{I}}] + \mathbb{1}[y_{\text{I},i}^{x_\text{S}} = y_{\text{I},i}^{x_\text{I}}]) \tag{3}$$

2. Binary property prediction: The predictions are denoted as y^{x_S} and y^{x_I} for SMILES and IUPAC inputs, separately. The consistency score for a dataset with N entries is:

$$\text{Consistency(binary)} = \frac{1}{N}\sum_{i=1}^{N}(\mathbb{1}[y_i^{x_\text{S}} = y_i^{x_\text{I}}]) \tag{4}$$

3. Numeric property prediction: consistency is measured as the mean squared error (MSE) between the predictions from SMILES and IUPAC inputs:

$$\text{Consistency(numeric)} = \frac{1}{N}\sum_{i=1}^{N}(y_i^{x_\text{S}} - y_i^{x_\text{I}})^2 \tag{5}$$

Accuracy

Accuracy evaluates how closely the model's predictions align with the ground truth.

1. Forward reaction prediction and retrosynthesis: For SMILES input, accuracy is cal-

culated as the percentage of exact matches between the predicted SMILES output ($y_{\mathrm{S}}^{x_{\mathrm{S}}}$) and the target SMILES output ($y_{\mathrm{S}}^{\mathrm{target}}$); for IUPAC input, accuracy is calculated between the predicted IUPAC output ($y_{\mathrm{I}}^{x_{\mathrm{I}}}$) and the target IUPAC output ($y_{\mathrm{I}}^{\mathrm{target}}$).

$$\begin{aligned}\text{Accuracy(SMILES)} &= \frac{1}{N}\sum_{i}^{N}(\mathbb{1}[y_{\mathrm{S},i}^{x_{\mathrm{S}}} = y_{\mathrm{S},i}^{\mathrm{target}}]) \\ \text{Accuracy(IUPAC)} &= \frac{1}{N}\sum_{i}^{N}(\mathbb{1}[y_{\mathrm{I},i}^{x_{\mathrm{I}}} = y_{\mathrm{I},i}^{\mathrm{target}}])\end{aligned} \tag{6}$$

2. Binary property prediction: accuracy is calculated as the percentage of predictions same to the ground-truth y^{target}.

$$\begin{aligned}\text{Accuracy(SMILES)} &= \frac{1}{N}\sum_{i}^{N}(\mathbb{1}[y_{i}^{x_{\mathrm{S}}} = y_{i}^{\mathrm{target}}]) \\ \text{Accuracy(IUPAC)} &= \frac{1}{N}\sum_{i}^{N}(\mathbb{1}[y_{i}^{x_{\mathrm{I}}} = y_{i}^{\mathrm{target}}])\end{aligned} \tag{7}$$

3. Numeric property prediction: accuracy is measured as the MSE between the predicted outputs and the ground truth values.

$$\begin{aligned}\text{Accuracy(SMILES)} &= \frac{1}{N}\sum_{i=1}^{N}(y_{i}^{x_{\mathrm{S}}} - y_{i}^{\mathrm{target}})^2 \\ \text{Accuracy(IUPAC)} &= \frac{1}{N}\sum_{i=1}^{N}(y_{i}^{x_{\mathrm{S}}} - y_{i}^{\mathrm{target}})^2\end{aligned} \tag{8}$$

Evaluation of state-of-the-art LLMs

We evaluated the consistency and accuracy of state-of-the-art LLMs for forward reaction prediction. The models assessed include GPT-4[13], GPT-4o[16], o1-preview, o1-mini[17], Claude 3 Opus[18], Llama 3.1 8B[19], and the instruction-tuned Llasmol$_{\mathrm{Mistral}}$[20]. A test set of 300 chemical reactions was used for the evaluation.

To guide the models, we provided explicit instructions tailored to the input and output molecular representations. For instance, when both the input and output were in SMILES format, the instruction read: "Based on the SMILES strings of reactants and reagents, predict the SMILES string of the product. Please output the product directly."

Finetuning LLMs with mapped SMILES & IUPAC data

To address possible biases in pre-trained data, we finetuned GPT2, Mistral 7B, and CodeT5 models using carefully curated datasets containing one-to-one mapped SMILES and IUPAC input representations. These datasets were designed to isolate the impact of input representation differences while keeping the underlying chemistry constant.

At training time, for forward reaction prediction and retrosynthesis, the model generates either SMILES or IUPAC outputs with equal probability. To explicitly specify the output representation, we appended a flag at the end of each input sequence: "S" for SMILES output and "I" for IUPAC output. Examples of input and output sequences used in the training set are provided in the Supporting Information. All models were trained using a cross-entropy loss function.

To examine the effect of model size, we conducted experiments with four variants of GPT2: small (124M parameters), medium (355M parameters), large (774M parameters), and extra-large (1.5B parameters).

For each model and task, we varied random seeds to calculate standard deviations in consistency and accuracy. Detailed training hyperparameters and additional implementation details are provided in the Supporting Information (Table 3 and Section).

Sequence-level KL divergence loss

To improve consistency across molecular representations, we introduce a sequence-level KL divergence loss during training. This loss minimizes the divergence between the probabilistic distributions generated from SMILES and IUPAC inputs, $P_\theta(y|x_\mathrm{S})$ and $Q_\theta(y|x_\mathrm{I})$.

We consider both directions of the KL divergence, $D_{KL}(P||Q)$ and $D_{KL}(Q||P)$:

$$\begin{aligned} D_{KL}(P||Q) &= \sum_{y \in Y} P_\theta(y|x_{\mathrm{S}}) \log \frac{P_\theta(y|x_{\mathrm{S}})}{Q_\theta(y|x_{\mathrm{I}})} \\ D_{KL}(Q||P) &= \sum_{y \in Y} Q_\theta(y|x_{\mathrm{I}}) \log \frac{Q_\theta(y|x_{\mathrm{I}})}{P_\theta(y|x_{\mathrm{S}})} \end{aligned} \tag{9}$$

where Y is the set of all possible output sequences.

However, the sequence-level KL divergence is computationally intractable. Therefore, we estimate the KL divergence using the Monte-Carlo sampling method. Details of KL divergence loss can be found in the Supporting Information.

Data

We base our work on the SMolInstruct dataset, which is a large-scale instruction-tuning dataset for chemistry[20]. We used the Property Prediction and Chemical Reaction subsets. The original datasets use SMILES representation. We translated SMILES into IUPAC to construct one-to-one mapped SMILES and IUPAC input datasets for the finetuning.

For each molecule in the dataset, we first used `PubChemPy`[21], a Python wrapper for the PubChem PUG REST API, to retrieve its IUPAC name. If no IUPAC name was found, we use an open-source model, `Chemical-Converters`[22], an open-source model based on the Google MT5 architecture, to translate SMILES into IUPAC. This model achieves an accuracy of 86.9% for SMILES-to-IUPAC conversion.

The training dataset for the forward reaction prediction task consists of 1M entries. For most models, we used an 80k subset for fine-tuning. To evaluate the impact of dataset size, we also trained a GPT2 model on the full dataset.

Similarly, the training dataset for the retrosynthesis task contains 1M entries. Most models were fine-tuned using an 80k subset, with the exception of a GPT2 model, which was trained on the full dataset.

The statistics of the mapped datasets are listed in the Supporting Information Table 4.

Results and discussion

Evaluation of state-of-the-art LLMs

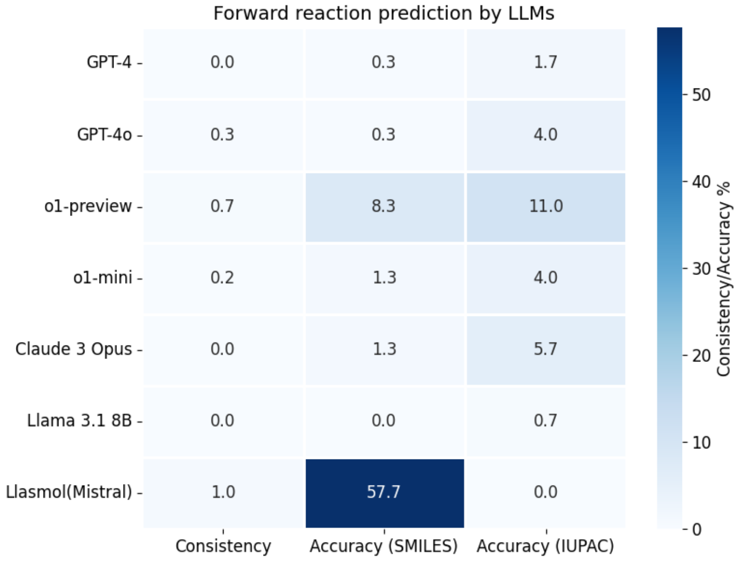


Figure 2: Consistency and accuracy of forward reaction predictions by state-of-the-art LLMs. Across all models, consistency remains below 1%. Most models exhibit higher accuracy for IUPAC inputs, except for LlaSMol$_{\text{Mistral}}$, which is instruction-tuned on a SMILES dataset. Darker colors represent higher values, while lighter colors indicate lower values.

We evaluated the consistency and accuracy of forward reaction prediction across seven state-of-the-art LLMs, focusing on their performance when using SMILES versus IUPAC input representations. The results revealed four key insights (Figure 2).

First, across all models, consistency scores ranged from 0% to 1%, revealing the poor alignment between SMILES and IUPAC representations. The result indicates that LLMs struggle to maintain consistent outputs when tasked with generating predictions from different input representations.

Second, LLMs without instruction tuning achieved higher accuracy for IUPAC inputs. This discrepancy is likely due to the training data distribution, which tends to include more examples using IUPAC[23–25], providing the models with a familiarity advantage for this representation.

Third, models designed for reasoning, such as o1-preview and o1-mini, demonstrated improved accuracy. However, this increase in accuracy did not translate to higher consistency between SMILES and IUPAC representations. This observation suggests that accuracy and consistency are orthogonal metrics, with improvements in one not necessarily leading to improvements in the other. This orthogonality is further explored in the discussion.

Finally, the instruction-tuned model, $\text{Llasmol}_{\text{Mistral}}$, achieved significantly higher accuracy with SMILES inputs, reflecting the impact of its SMILES-specific training. However, this tuning did not enhance accuracy with IUPAC inputs, indicating a lack of generalization between the two representations. This result highlights a key limitation of current LLMs—they fail to develop an intrinsic understanding of the chemical equivalence between different molecular representations.

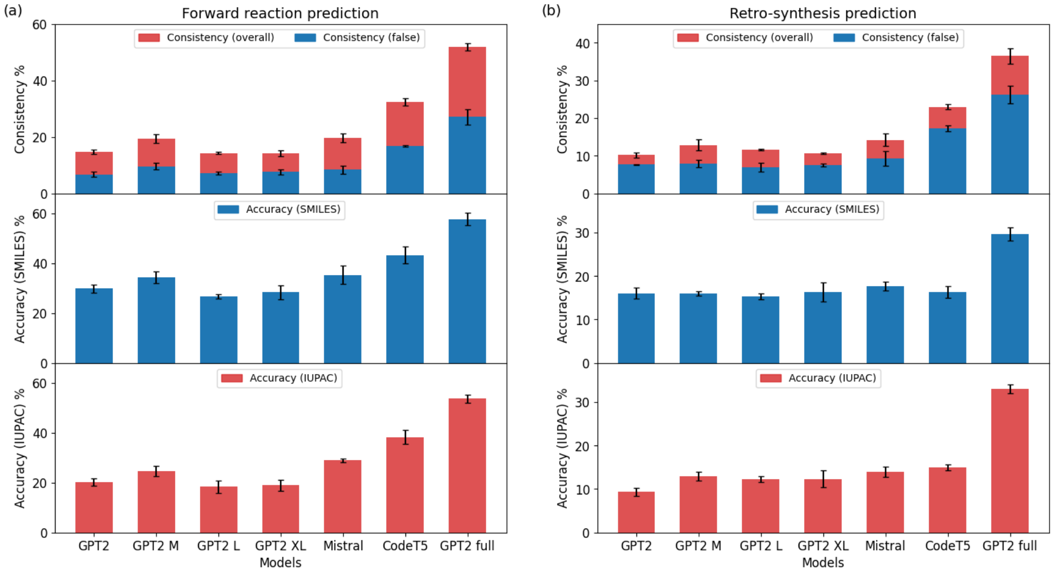


Figure 3: Consistency and accuracy of LLMs in (a) forward reaction prediction and (b) retrosynthesis after finetuning on one-to-one mapped data. The overall consistency (red) and false consistency (blue) are overlaid. Most models are finetuned on an 80k dataset subset, except for “GPT2 full”, which refers to a GPT2 small model trained on the full 1M dataset. Error bars represent the standard deviation across training runs with varying random seeds.

Finetuning LLMs with mapped SMILES & IUPAC data

The state-of-the-art LLMs discussed earlier do not utilize one-to-one mapped training data, which may introduce biases that favor either IUPAC or SMILES representations. To mitigate these biases, we performed finetuning using a one-to-one mapped dataset of SMILES and IUPAC representations, ensuring that the representation format was the only variable.

We evaluated three architectures – GPT2, Mistral 7B[26], and CodeT5 Small[27] – on three tasks: forward reaction prediction, retrosynthesis, and property prediction. For GPT2, we further varied the model size, GPT2 Small, GPT2 Medium (M), GPT2 Large (L), and GPT2 XL, to examine the impact of scaling. Additionally, we compared performance using two training data sizes: 80k and 1M data points.

Performance was evaluated using two metrics: consistency and accuracy. We analyzed both overall consistency and false consistency (cases where SMILES and IUPAC inputs produce the same incorrect predictions), which is critical for disentangling consistency from accuracy. Accuracy was measured separately for SMILES and IUPAC inputs. The results are presented in Figure 3, Table 1 and Table 2.

Impact of model architectures. For forward reaction prediction and retrosynthesis tasks, CodeT5 consistently outperformed Mistral and GPT2. Its encoder-decoder architecture likely contributes to this by constructing a structured latent representation of the input, enabling better transformation into the output space[27]. In contrast, the decoder-only architectures of GPT2 and Mistral, designed for autoregressive generation, may be less suited for these structured prediction tasks.

For property prediction, however, the results vary across models and tasks. These mixed results indicate that certain architectures, such as CodeT5's encoder-decoder framework, may excel at capturing structural patterns important for some properties, while decoder-only models like GPT2 and Mistral may generalize better for less complex tasks[28].

Impact of model sizes. Scaling up the GPT2 model from Small to XL showed no significant improvements in consistency or accuracy for forward reaction prediction or ret-

Table 1: Consistency and accuracy of LLMs in binary property prediction after controlled fine-tuning (columns 3–5) and with KL divergence loss (columns 6–8). Entries with improvements following the addition of KL divergence loss are highlighted in bold. Error bars represent the standard deviation across training runs with varying random seeds. An upward arrow (↑) indicates that higher values correspond to better performance.

Properties	Models	Performance (%) ↑			Performance w/ KL (%) ↑		
		Consist.	Acc. (S)	Acc. (I)	Consist.	Acc. (S)	Acc. (I)
BBBP	GPT2	83.6 ± 1.1	83.6 ± 1.7	81.0 ± 2.1	**91.5 ± 1.8**	**86.2 ± 0.9**	**82.0 ± 1.1**
	Mistral	85.2 ± 6.8	68.3 ± 5.8	76.7 ± 1.3	**90.5 ± 1.1**	**84.1 ± 4.3**	**78.8 ± 5.3**
	CodeT5	85.7 ± 2.0	85.7 ± 0.3	85.2 ± 2.9	**88.9 ± 2.4**	**86.2 ± 1.5**	82.5 ± 0.3
ClinTox	GPT2	95.4 ± 1.9	93.1 ± 0.4	91.6 ± 1.5	**96.2 ± 2.0**	93.1 ± 1.2	**92.4 ± 0.0**
	Mistral	100.0±4.8	92.4 ± 0.0	92.4 ± 4.0	99.2 ± 0.4	92.4 ± 0.0	91.6 ± 0.4
	CodeT5	87.0 ± 2.0	89.3 ± 1.2	85.5 ± 3.1	**94.7 ± 0.4**	**91.6 ± 0.9**	**90.8 ± 1.2**
HIV	GPT	97.3 ± 0.7	95.3 ± 0.4	95.3 ± 0.3	**98.3 ± 0.0**	**96.3 ± 0.3**	95.3 ± 0.2
	Mistral	99.7 ± 0.2	95.7 ± 0.2	95.3 ± 0.0	99.7 ± 0.2	95.3 ± 0.0	95.0 ± 0.2
	CodeT5	96.7 ± 0.5	96.0 ± 0.5	96.0 ± 0.2	**97.3 ± 1.1**	95.7 ± 0.2	**96.3 ± 0.2**
SIDER	GPT	61.3 ± 1.2	55.7 ± 1.2	62.0 ± 2.5	**77.7 ± 3.8**	55.7 ± 0.3	**65.7 ± 0.3**
	Mistral	98.3 ± 0.8	65.0 ± 3.5	66.0 ± 0.2	96.7 ± 1.3	64.7 ± 3.6	63.3 ± 1.5
	CodeT5	71.3 ± 4.3	60.7 ± 2.8	60.7 ± 1.0	**76.7 ± 5.9**	**62.3 ± 1.3**	**61.7 ± 1.2**

Table 2: Consistency and accuracy of LLMs in numeric property prediction after controlled fine-tuning (columns 3–5) and with KL divergence loss (columns 6–8). Entries with improvements following the addition of KL divergence loss are highlighted in bold. Error bars denote the standard deviation across training runs with varying random seeds. A downward arrow (↓) indicates that lower values correspond to better performance.

Properties	Models	Performance (MSE) ↓			Performance w/ KL (MSE) ↓		
		Consist.	Acc. (S)	Acc. (I)	Consist.	Acc. (S)	Acc. (I)
ESOL	GPT2	4.3 ± 0.5	1.5 ± 0.1	3.3 ± 0.6	**2.7 ± 0.3**	1.6 ± 0.3	**3.1 ± 0.1**
	Mistral	4.9 ± 0.5	1.7 ± 0.8	4.5 ± 0.6	**2.1 ± 0.2**	**1.3 ± 0.3**	**2.9 ± 0.4**
	CodeT5	5.9 ± 0.5	0.9 ± 0.2	5.4 ± 0.4	**3.1 ± 0.7**	1.8 ± 0.3	**3.6 ± 0.2**
LIPO	GPT2	1.1 ± 0.1	1.2 ± 0.0	1.2 ± 0.0	**0.7 ± 0.0**	**1.0 ± 0.1**	**1.0 ± 0.0**
	Mistral	0.9 ± 0.2	1.5 ± 0.2	1.2 ± 0.0	**0.5 ± 0.1**	**1.2 ± 0.0**	**1.1 ± 0.0**
	CodeT5	1.0 ± 0.2	1.0 ± 0.0	0.9 ± 0.1	**1.0 ± 0.0**	1.1 ± 0.0	1.0 ± 0.1

rosynthesis. These results suggest that simply increasing model size does not enhance the ability to generalize between SMILES and IUPAC representations or improve performance in reaction prediction tasks.

Impact of data size. For GPT2, increasing the training dataset size from 80k to 1M led to substantial improvements in both consistency and accuracy for forward reaction prediction and retrosynthesis. The increase in overall consistency aligns with the improvement in accuracy, indicating that the larger dataset enhances the model's ability to make correct predictions for both SMILES and IUPAC inputs. However, the gap between overall consistency and false consistency widened, suggesting that the additional data results in limited improvement in false consistency.

Adding sequence-level KL divergence loss

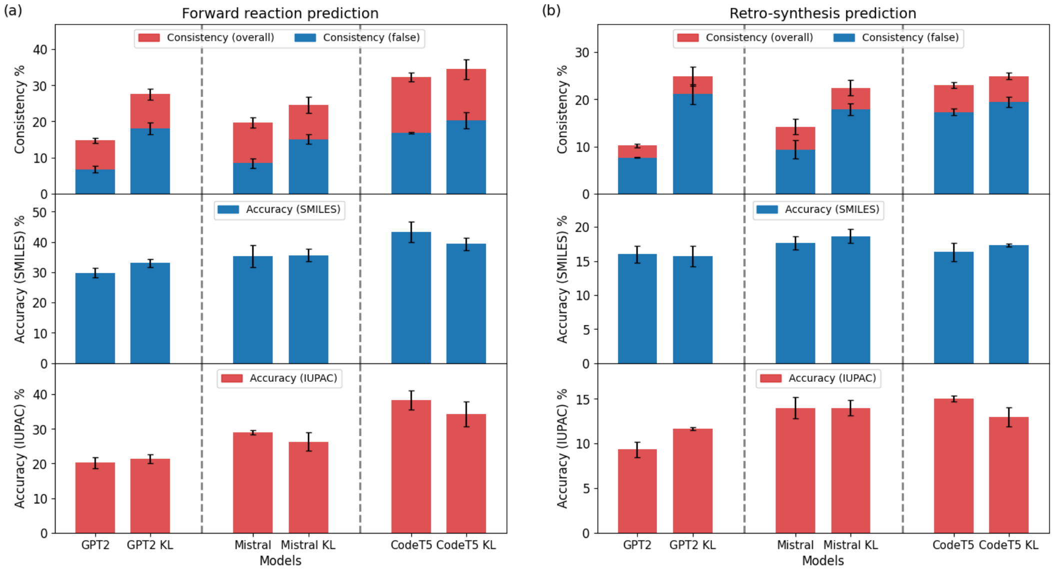


Figure 4: Consistency and accuracy of LLMs in (a) forward reaction prediction and (b) retrosynthesis prediction with the addition of KL divergence loss. Overall consistency (red) and false consistency (blue) are overlaid. All models are fine-tuned on an 80k dataset subset. Error bars represent the standard deviation across training runs with varying random seeds.

In this section, we examined the impact of adding sequence-level KL divergence loss during training on three models: GPT2, Mistral 7B, and CodeT5 Small, for forward reaction prediction, retrosynthesis, and property prediction. The results are summarized in Figure 4, Table 1 and Table 2. Key observations are summarized below.

Consistency Improvements. Adding KL divergence loss led to notable improvements in consistency across all models and tasks. Specifically, for forward reaction prediction and retrosynthesis, false consistency increased, and the gap between overall and false consistency narrowed, contrasting with the trends observed when increasing dataset size. These results confirm that KL divergence loss enhances consistency by aligning predictions across input representations.

Accuracy Unchanged. Despite improvements in consistency, accuracy remained largely unchanged across models and tasks. This suggests that the gains in consistency do not compromise accuracy but also highlights the orthogonality of these two metrics – improving one does not inherently lead to improvements in the other.

Analysis

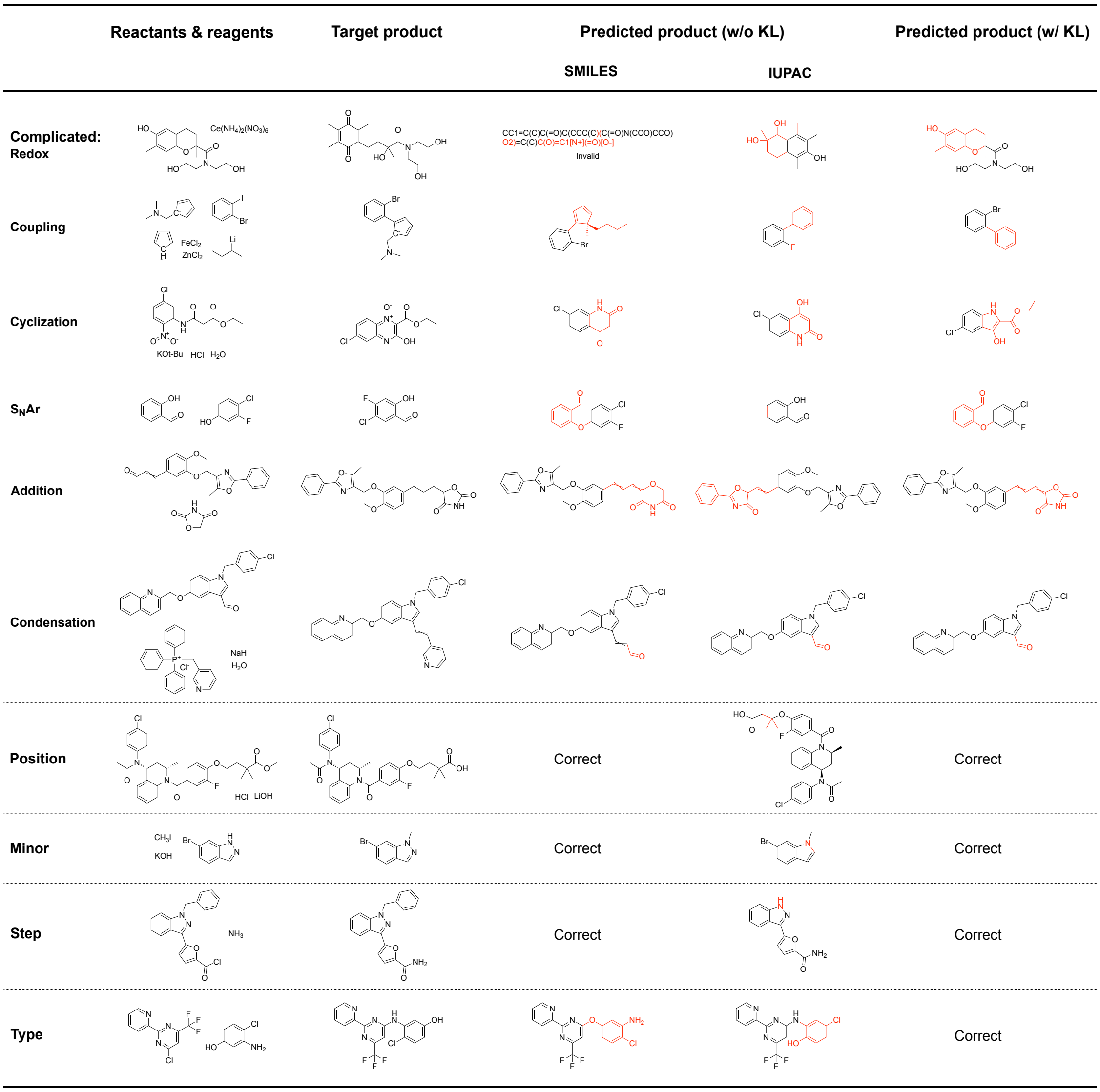
Consistency transition with KL divergence Loss

To explore how KL divergence loss improves consistency, we analyzed forward reaction prediction as a representative task, focusing on reactions where consistency transitions after adding KL divergence loss.

Out of 300 reactions in the test set, 47 reactions transitioned from inconsistent to consistent predictions after adding KL divergence loss. These reactions were categorized into five groups (Figure 5) and listed in Scheme 1, and in Supporting Information Schemes 2-11:

1. Complicated reactions: More than half of the reactions (25/47) fall into this category, which require a good understanding of chemistry and substantial manipulation of symbolic representations. For instance, hydroquinone oxidation by cerium(IV) ammonium nitrate requires recognizing the hydroquinone structure and the oxidant. Besides, the

Scheme 1: Examples of reactions transitioning from inconsistent to consistent predictions after adding KL divergence loss. Incorrect fragments are highlighted in red. For correct predictions, only the label "correct" is written without drawing the chemical structure.



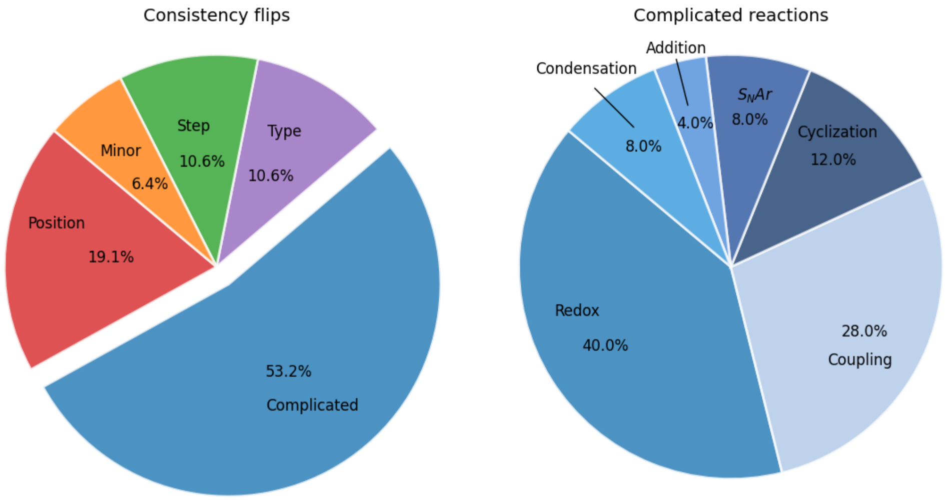


Figure 5: Summary of reactions that transition from inconsistent without KL divergence loss to consistent with KL divergence loss. (Left) Reactions are categorized into five groups: complicated reactions, position inconsistencies, minor mistakes, reaction-step inconsistencies, and reaction-type inconsistencies. (Right) Complicated reactions are further subdivided into six types: redox reactions, coupling reactions, cyclization reactions, addition reactions, condensation reactions, and nucleophilic aromatic substitution (S_NAr) reactions.

product's SMILES string differs from the reactant's SMILES string in multiple positions (Scheme 1, first entry).

These reactions span six types: redox, coupling, cyclization, nucleophilic aromatic substitution (S_NAr), addition, and condensation. Their distribution is shown in Figure 5 (right pie chart). Additional examples are provided in Scheme 1 and Supporting Information.

2. Position inconsistency: The second-largest group consists of reactions whose predicted products are inconsistent in reaction sites or the positions of functional groups between SMILES and IUPAC inputs.

3. Reaction type inconsistency: SMILES and IUPAC inputs lead to predicted products of different reaction types.

4. Reaction step inconsistency: SMILES and IUPAC inputs result in predicted products

involving different numbers of reaction steps.

5. Minor inconsistency: Reactions with minor errors in either SMILES or IUPAC representations, such as mislabeling a nitrogen atom as carbon.

Interestingly, the reverse transition – from consistent to inconsistent predictions – follows a similar pattern. Out of 300 test reactions, 6 reactions became inconsistent with KL divergence loss: 3 were complicated reactions, and 3 exhibited position inconsistency (reactions listed in Supporting Information Schemes 12 and 13).

The analysis reveals that while KL divergence loss enhances consistency, it does not improve the model's understanding of intrinsic chemistry. For complicated reactions, models often make inconsistent and incorrect predictions without KL divergence loss, and while consistency improves with KL divergence loss, the predictions remain incorrect.

In contrast, for reactions where the model makes correct predictions in one representation (e.g., SMILES) but minor mistakes in the other, KL divergence loss helps align predictions, enabling correct outputs across both representations.

The results suggest that KL divergence loss effectively addresses surface-level inconsistencies, but it falls short of achieving both accuracy and consistency. Advanced techniques will be required to capture the deeper intrinsic chemistry and achieve the ultimate goal of accurate and consistent predictions across all representations.

Orthogonality between consistency and accuracy

To explicitly analyze the relationship between consistency and accuracy, we examined the forward reaction prediction task using the GPT2 small model with various random seeds. We used false consistency instead of overall consistency to exclude cases where both representations produce correct predictions, and to provide a clearer measure of consistency.

We plotted consistency versus accuracy for models finetuned with and without KL divergence loss (Figure 6). In both cases, there was minimal correlation between false consistency

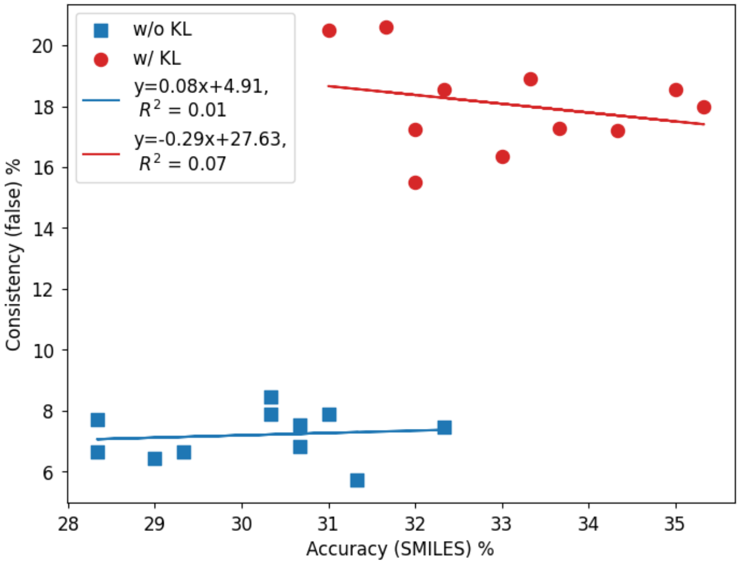


Figure 6: False consistency versus accuracy of the GPT2 model in forward reaction prediction, shown without KL divergence loss (blue squares) and with KL divergence loss (red circles) across different random seeds in training. A linear fit of the data demonstrates minimal correlation between consistency and accuracy.

and accuracy, confirming their orthogonality. Linear regression of the data yielded slopes of -0.29 (with KL divergence loss) and 0.08 (without KL divergence loss), further demonstrating that improvements in accuracy do not directly lead to better consistency. These findings highlight the need for distinct strategies to enhance both metrics independently.

Conclusions

This work explores whether large language models (LLM) truly understand the intrinsic chemistry of molecules. To investigate this, we evaluated the consistency of LLMs across chemistry tasks using different molecular representations, such as SMILES strings and IUPAC names. Our findings reveal that LLMs exhibit low consistency between these representations, even when trained on carefully controlled one-to-one mapped data.

Incorporating sequence-level KL divergence loss improved surface-level consistency by aligning predictions across representations. However, it does not enable the models to capture deeper intrinsic chemical properties. Further analysis revealed the orthogonality of consistency, demonstrating that improvements in one do not inherently lead to enhancements in the other.

These findings underscore the limitations of current LLM architectures and the pressing need for advanced models capable of understanding the intrinsic chemistry of molecules. Such advancements are crucial for achieving both accurate and consistent predictions in chemistry tasks, bridging the gap between symbolic representations and true chemistry understanding.

Acknowledgement

This work was supported by the Samsung Advanced Institute of Technology (under the project Next Generation Deep Learning: From Pattern Recognition to AI) and the National Science Foundation (under NSF Award 1922658).

Data availability

The code and data are available at `https://github.com/bingyan4science/consistency`.

References

(1) Anstine, D. M.; Isayev, O. Generative models as an emerging paradigm in the chemical sciences. *Journal of the American Chemical Society* **2023**, *145*, 8736–8750.

(2) Tran, D.; Pascazio, L.; Akroyd, J.; Mosbach, S.; Kraft, M. Leveraging text-to-text pretrained language models for question answering in chemistry. *ACS omega* **2024**, *9*, 13883–13896.

(3) Jablonka, K. M.; Schwaller, P.; Ortega-Guerrero, A.; Smit, B. Leveraging large language models for predictive chemistry. *Nature Machine Intelligence* **2024**, *6*, 161–169.

(4) M. Bran, A.; Cox, S.; Schilter, O.; Baldassari, C.; White, A. D.; Schwaller, P. Augmenting large language models with chemistry tools. *Nature Machine Intelligence* **2024**, 1–11.

(5) Schwaller, P.; Laino, T.; Gaudin, T.; Bolgar, P.; Hunter, C. A.; Bekas, C.; Lee, A. A. Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction. *ACS central science* **2019**, *5*, 1572–1583.

(6) Guo, T.; Nan, B.; Liang, Z.; Guo, Z.; Chawla, N.; Wiest, O.; Zhang, X.; others What can large language models do in chemistry? a comprehensive benchmark on eight tasks. *Advances in Neural Information Processing Systems* **2023**, *36*, 59662–59688.

(7) Zhao, L.; Edwards, C.; Ji, H. What a Scientific Language Model Knows and Doesn't Know about Chemistry. NeurIPS 2023 AI for Science Workshop. 2023.

(8) Sadeghi, S.; Bui, A.; Forooghi, A.; Lu, J.; Ngom, A. Can large language models understand molecules? *BMC bioinformatics* **2024**, *25*, 225.

(9) Castro Nascimento, C. M.; Pimentel, A. S. Do large language models understand chemistry? A conversation with chatgpt. *Journal of Chemical Information and Modeling* **2023**, *63*, 1649–1655.

(10) White, A. D.; Hocky, G. M.; Gandhi, H. A.; Ansari, M.; Cox, S.; Wellawatte, G. P.; Sasmal, S.; Yang, Z.; Liu, K.; Singh, Y.; others Assessment of chemistry knowledge in large language models that generate code. *Digital Discovery* **2023**, *2*, 368–376.

(11) Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences* **1988**, *28*, 31–36.

(12) Vaswani, A. Attention is all you need. *Advances in Neural Information Processing Systems* **2017**,

(13) Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; others Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* **2023**,

(14) Favre, H. A., Powell, W. H., Eds. *Nomenclature of Organic Chemistry: IUPAC Recommendations and Preferred Names 2013*; Royal Society of Chemistry: Cambridge, UK, 2014.

(15) Kullback, S.; Leibler, R. A. On information and sufficiency. *The annals of mathematical statistics* **1951**, *22*, 79–86.

(16) Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; Radford, A.; others Gpt-4o system card. *arXiv preprint arXiv:2410.21276* **2024**,

(17) OpenAI Introducing OpenAI o1-preview and o1-mini. `https://openai.com/index/introducing-openai-o1-preview`, 2024.

(18) Anthropic Introducing the Next Generation of Claude. `https://www.anthropic.com/news/claude-3-family`, 2024.

(19) AI, M. Introducing Meta Llama 3.1: The Most Capable Openly Available LLM to Date. `https://ai.meta.com/blog/meta-llama-3-1/`, 2024.

(20) Yu, B.; Baker, F. N.; Chen, Z.; Ning, X.; Sun, H. LlaSMol: Advancing Large Language Models for Chemistry with a Large-Scale, Comprehensive, High-Quality Instruction Tuning Dataset. First Conference on Language Modeling. 2024.

(21) Swain, M. A simple Python wrapper around the PubChem PUG REST API. `https://github.com/mcs07/PubChemPy`, Version 1.0.4.

(22) Knowledgator Chemical-Converters is collection of tools for converting one chemical format into another. https://github.com/Knowledgator/chemical-converters?tab=readme-ov-file, Version 0.1.1.

(23) National Center for Biotechnology Information (NCBI) PubMed Central (PMC). https://pmc.ncbi.nlm.nih.gov/, 2024.

(24) Patent, U. S.; (USPTO), T. O. USPTO Patent Database. https://ppubs.uspto.gov/pubwebapp/, 2024.

(25) Foundation, W. Wikimedia Downloads. https://dumps.wikimedia.org.

(26) Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; others Mistral 7B. *arXiv preprint arXiv:2310.06825* **2023**,

(27) Wang, Y.; Wang, W.; Joty, S.; Hoi, S. C. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. *arXiv preprint arXiv:2109.00859* **2021**,

(28) Chen, Y.; Ou, G.; Liu, M.; Wang, Y.; Zheng, Z. Are Decoder-Only Large Language Models the Silver Bullet for Code Search? *arXiv preprint arXiv:2410.22240* **2024**,

(29) Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* **2014**,

(30) Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* **2017**,

Supporting Information Available

Implementation details

Software and hardware

In this work, we use Python 3.10. The major Python packages we used are Transformers 4.43.4, PyTorch 2.1.0, RDKit 2023.3.3.

We train the models using Nvidia A100 or H100 GPU. We use one GPU for GPT2 small, GPT2 medium, GPT2 large, and CodeT5 small models, and two GPUs for GPT2 XL and Mistral 7B models.

Hyperparameters

We train all models using the AdamW optimizer[29,30]. We use random seeds of 42, 123, 999, 1234, 2024, 2718, 4321, 5678, 8080, 31415, and 98765. The other hyperparameters for each model are summarized in the table below.

Table 3: Hyperparameters used to finetune LLMs.

Model	Learning rate	Batch size	Accumulation	#Epochs
GPT2 small	1e-4	32	1	20
GPT2 medium	1e-4	16	1	20
GPT2 large	1e-4	8	1	20
GPT2 XL	1e-4	8	2	20
CodeT5 small	1e-4	32	1	20
Mistral 7B	1e-5	8	2	10

Input and output examples

We provide some examples of input and output sequences for model finetuning and evaluation.

1. Evaluation of state-of-the-art LLMs: We provide a simple instruction specifying the input and output representation in the inquiry sent to the API. The molecules are separated by comma (".") For example:

   Input in SMILES: "Based on the SMILES strings of reactants and reagents, predict the SMILES string of the product. Please output the product directly.

   <SMILES> COc1ccc2c(c1)C(=O)c1ccccc1CC2.[BH4-].[OH-].[Na+].CCO <SMILES>"

   Target output in SMILES:

   "COc1ccc2c(c1)C(O)c1ccccc1CC2"

   Input in IUPAC: "Based on the IUPAC names of reactants and reagents, predict the IUPAC name of the product. Please output the product directly.

   <IUPAC> 5-methoxytricyclo[9.4.0.03,8]pentadeca-1(15),3(8),4,6,11,13-hexaen-2-one. boranuide.hydroxide.sodium(1+).ethanol <IUPAC>"

   Target output in IUPAC:

   "5-methoxytricyclo[9.4.0.03,8]pentadeca-1(15),3(8),4,6,11,13-hexaen-2-ol"

2. Finetuning of LLMs: We append a flag in the end to the input sequence to specify the output representation, "S" for SMILES and "I" for IUPAC. For example:

   Input in SMILES expecting output in SMILES:

   "COc1ccc2c(c1)C(=O)c1ccccc1CC2.[BH4-].[OH-].[Na+].CCO.S"

   Target in SMILES:

   "COc1ccc2c(c1)C(O)c1ccccc1CC2"

   Input in SMILES expecting output in IUPAC:

   "COc1ccc2c(c1)C(=O)c1ccccc1CC2.[BH4-].[OH-].[Na+].CCO.I"

   Target in IUPAC:

   "5-methoxytricyclo[9.4.0.03,8]pentadeca-1(15),3(8),4,6,11,13-hexaen-2-ol"

KL divergence loss

Here we show the loss function for the sequence-level KL divergence: $D_{KL}(P||Q)$ and $D_{KL}(Q||P)$. We take the KL divergence $D_{KL}(P||Q)$ as an example to demonstrate the calculation.

The gradient of $D_{KL}(P||Q)$ is (here we simplify $P_\theta(y|x_{\mathrm{S}})$ as $P_\theta(y)$, and $Q_\theta(y|x_{\mathrm{I}})$ as $Q_\theta(y)$):

$$\begin{aligned}\nabla_\theta D_{KL}(P||Q) &= \sum_{y\in Y} \nabla_\theta (P_\theta(y) \log \frac{P_\theta(y)}{Q_\theta(y)}) \\ &= \sum_{y\in Y} \nabla_\theta (P_\theta(y)) \log \frac{P_\theta(y)}{Q_\theta(y)} + P_\theta(y) \nabla_\theta (\frac{P_\theta(y)}{Q_\theta(y)})\end{aligned} \tag{10}$$

Using the trick $\nabla_\theta(P_\theta(y)) = P_\theta(y)\nabla_\theta(\log(P_\theta(y)))$:

$$\begin{aligned}\nabla_\theta D_{KL}(P||Q) &= \sum_{y\in Y} P_\theta(y)\nabla_\theta(\log(P_\theta(y))) \log \frac{P_\theta(y)}{Q_\theta(y)} + P_\theta(y)\nabla_\theta(\frac{P_\theta(y)}{Q_\theta(y)}) \\ &= \mathbb{E}_{y\sim P_\theta(y)}[\nabla_\theta(\log P_\theta(y)) \log \frac{P_\theta(y)}{Q_\theta(y)} + \nabla_\theta(\log \frac{P_\theta(y)}{Q_\theta(y)})]\end{aligned} \tag{11}$$

Therefore, we can define the KL loss corresponding to the KL divergence $D_{KL}(P||Q)$:

$$\text{KL loss} \equiv \mathbb{E}_{y\sim P_\theta(y)}[\log P_\theta(y) \log \frac{P_\theta(y)}{Q_\theta(y)}.\text{detach} + \log \frac{P_\theta(y)}{Q_\theta(y)}] \tag{12}$$

However, the expectation is untractable, so we use a Monte Carlo to estimate it by sampling M sequences $\{y^1, ..., y^m\}$ from $P_\theta(y)$ and pass them through the models $P_\theta(y)$ and $Q_\theta(y)$:

$$\text{KL loss}(PQ) \approx \frac{1}{M}\sum_{m=1}^{M}[\log P_\theta(y^m) \log \frac{P_\theta(y^m)}{Q_\theta(y^m)}.\text{detach} + \log \frac{P_\theta(y^m)}{Q_\theta(y^m)}] \tag{13}$$

Similarly, we can calculate the loss for the KL divergence of $Q_\theta(y)$ from $P_\theta(y)$ ($D_{KL}(Q||P)$)

and the Monte Carlo estimation by sampling N sequences $\{y^1, ..., y^n\}$ from $Q_\theta(y)$:

$$\begin{aligned}\text{KL loss}(QP) &\equiv \mathbb{E}_{y \sim Q_\theta(y)}[\log Q_\theta(y) \log \frac{Q_\theta(y)}{P_\theta(y)}.\text{detach} + \log \frac{Q_\theta(y)}{P_\theta(y)}] \\ &\approx \frac{1}{N}\sum_{n=1}^{N}[\log Q_\theta(y^n) \log \frac{Q_\theta(y^n)}{P_\theta(y^n)}.\text{detach} + \log \frac{Q_\theta(y^n)}{P_\theta(y^n)}]\end{aligned} \tag{14}$$

Dataset

Here we list the statistics of the datasets used to fine-tune LLMs. There are three tasks: forward reaction prediction, retrosynthesis, and property prediction. These datasets are all one-to-one mapped between SMILES and IUPAC inputs.

Table 4: Statistics of the datasets used to finetune LLMs.

Task	#Train	#Valid	#Test
Forward reaction prediction (full)	963,567	1,956	300
Forward reaction prediction (subset)	76,379	1,956	300
Retrosynthesis (full)	932,616	2,004	300
Retrosynthesis (subset)	76,471	2,004	300
Property - BBBP	1,521	188	189
Property - ClinTox	1,063	127	131
Property - HIV	32,864	4,104	300
Property - SIDER	21,800	2,540	300
Property - ESOL	888	111	112
Property - LIPO	3,358	385	300

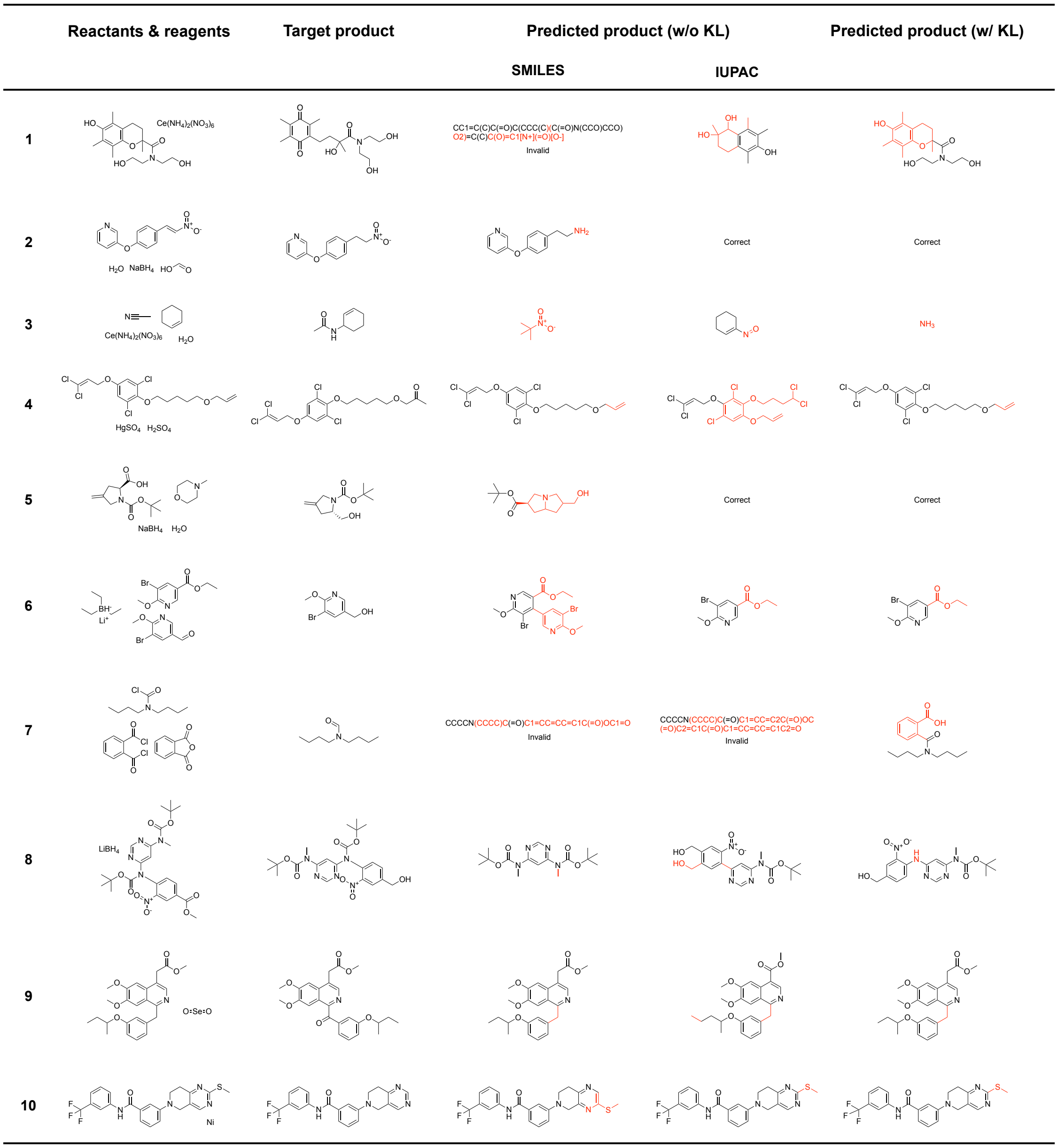
Consistency transition

Here we list all of the reactions that transit either from inconsistent to consistent predictions, or from consistent to inconsistent predictions.

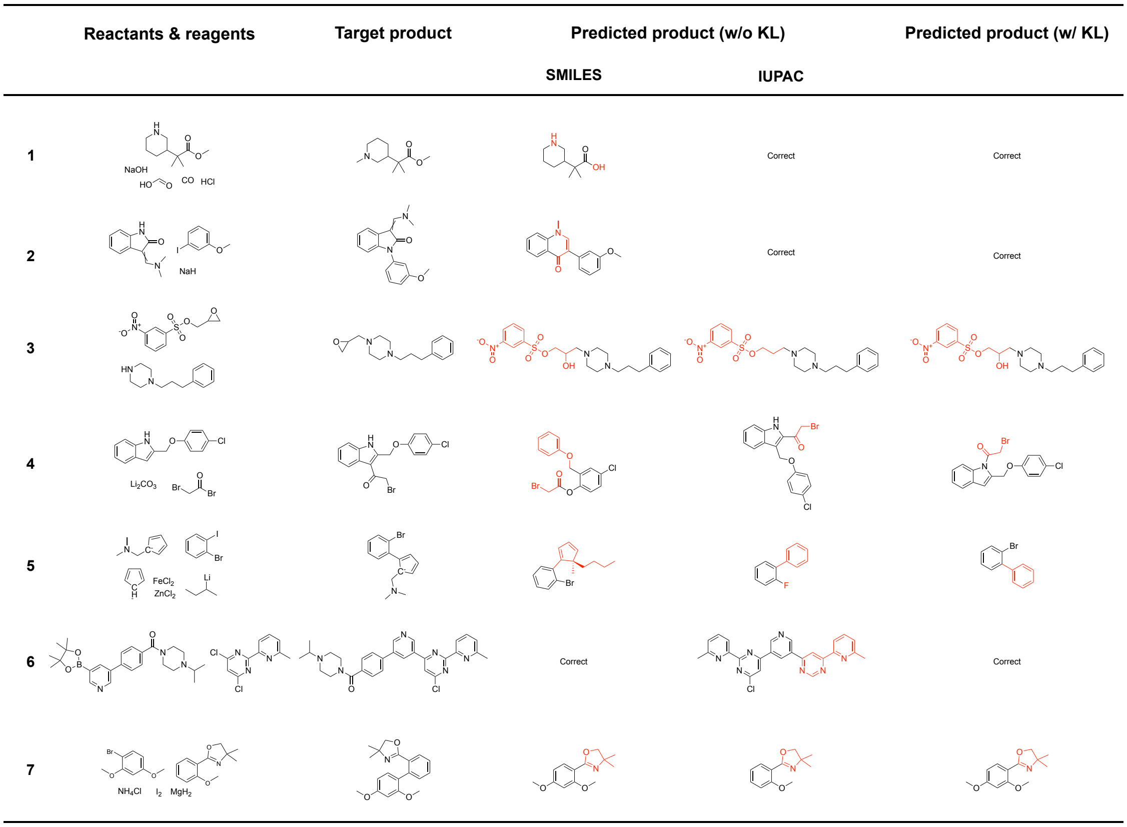
Consistent-to-inconsistent transitions

Here we list all of the 47 reactions that transition from inconsistent to consistent predictions between SMILES and IUPAC inputs after adding KL divergence loss.

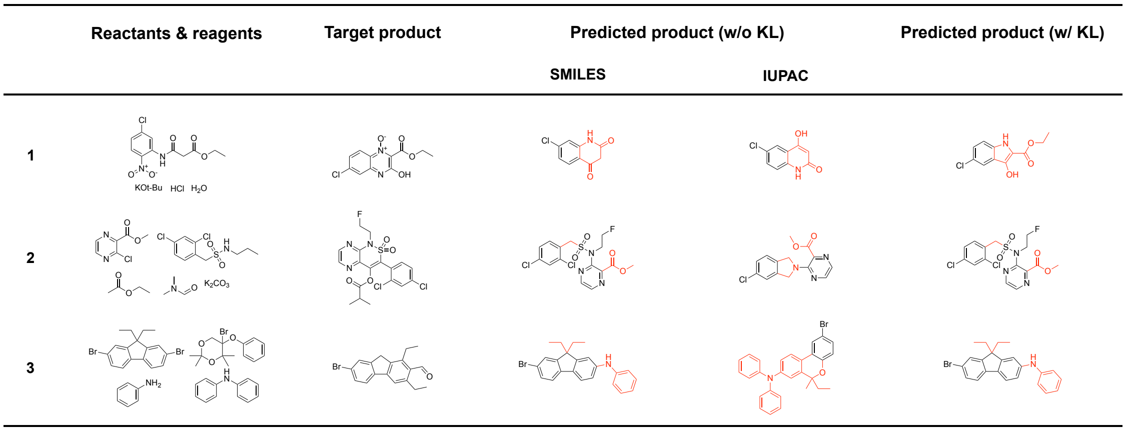
Scheme 2: Complicated redox reactions that transition from inconsistent to consistent predictions after adding KL divergence loss


Reactants & reagents
Target product
Predicted product (w/o KL)
SMILES
IUPAC
Predicted product (w/ KL)
1
Ce(NH4)2(NO3)6
CC1=C(C)C(=O)C(CCC(C)(C(=O)N(CCO)CCO)
O2)=C(C)C(O)=C1[N+](=O)[O-]
Invalid
2
H2O NaBH4
Correct
Correct
3
Ce(NH4)2(NO3)6
H2O
NH3
4
HgSO4 H2SO4
5
NaBH4 H2O
Correct
Correct
6
7
CCCCN(CCCC)C(=O)C1=CC=CC=C1C(=O)OC1=O
Invalid
CCCCN(CCCC)C(=O)C1=CC=C2C(=O)OC
(=O)C2=C1C(=O)C1=CC=CC=C1C2=O
Invalid
8
LiBH4
9
O=Se=O
10
Ni

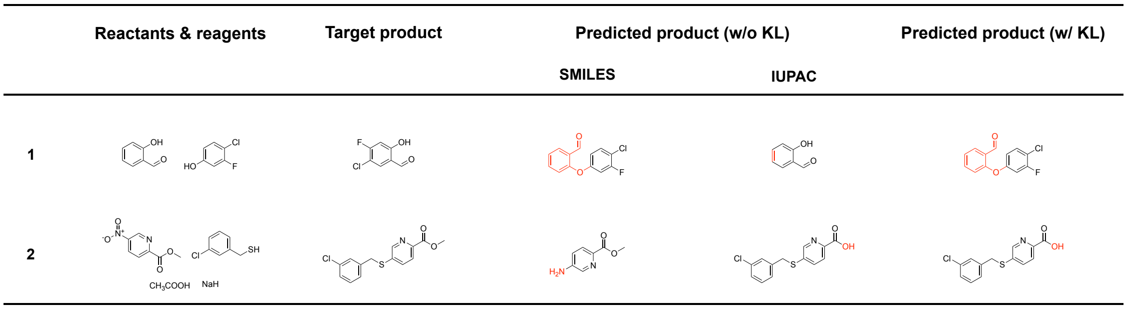
Scheme 3: Complicated coupling reactions that transition from inconsistent to consistent predictions after adding KL divergence loss



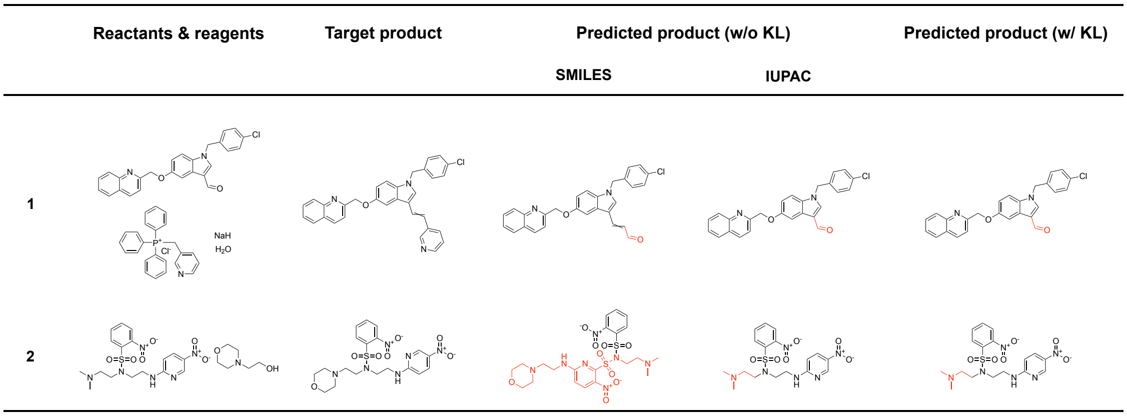
Scheme 4: Complicated cyclization reactions that transition from inconsistent to consistent predictions after adding KL divergence loss



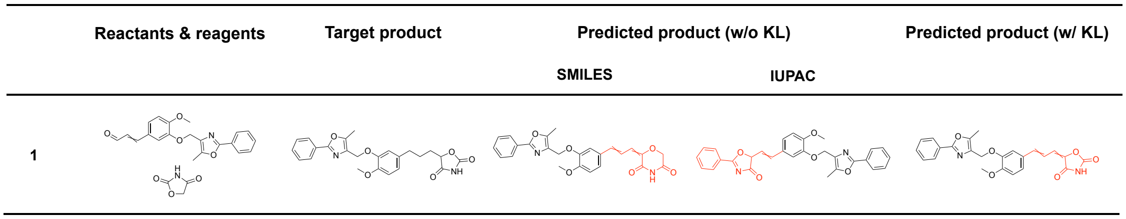
Scheme 5: Complicated S_NAr reactions that transition from inconsistent to consistent predictions after adding KL divergence loss



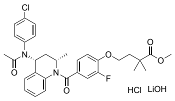
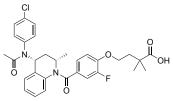
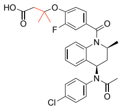
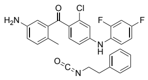
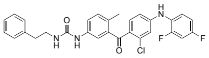
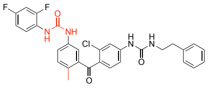
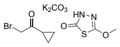
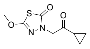
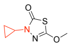
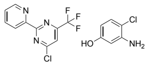
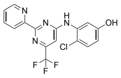
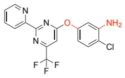
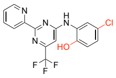
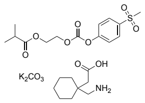
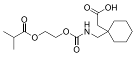
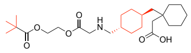
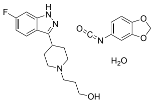
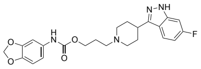
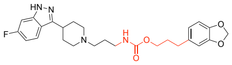
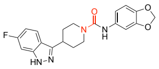
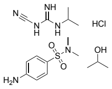
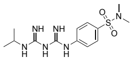
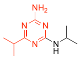
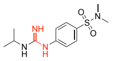
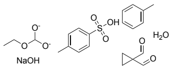
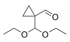
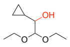
Scheme 6: Complicated condensation reactions that transition from inconsistent to consistent predictions after adding KL divergence loss



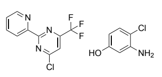
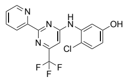
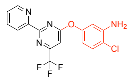
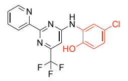
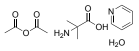
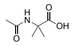
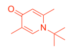
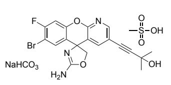
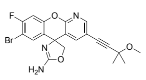
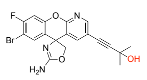
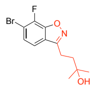
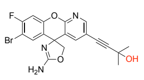
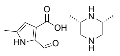
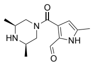
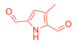
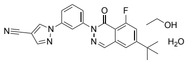
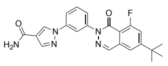
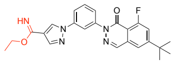
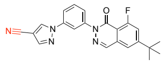
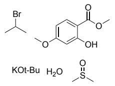
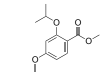
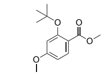
Scheme 7: Complicated addition reactions that transition from inconsistent to consistent predictions after adding KL divergence loss



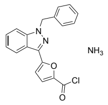
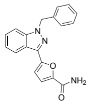
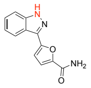
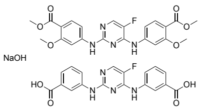
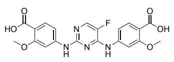
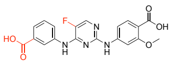
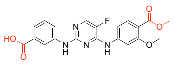
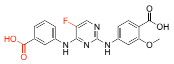
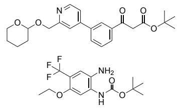
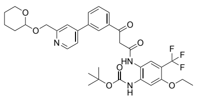
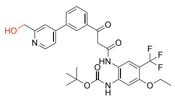
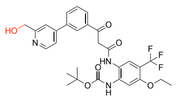
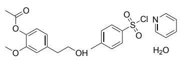
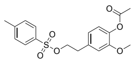
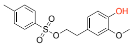
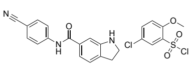
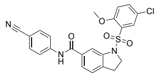
Scheme 8: Position-inconsistent reactions that transition from inconsistent to consistent predictions after adding KL divergence loss

	Reactants & reagents	Target product	Predicted product (w/o KL)		Predicted product (w/ KL)
			SMILES	IUPAC	
1			Correct		Correct
2			Correct		Correct
3			Correct		Correct
4					Correct
5			Correct		Correct
6			Correct		
7			Correct		
8			Correct		Correct

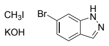
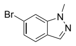
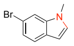
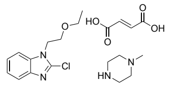
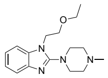
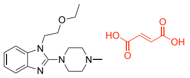
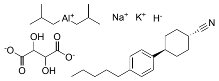
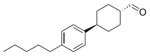
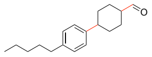
Scheme 9: Reaction type-inconsistent reactions that transition from inconsistent to consistent predictions after adding KL divergence loss

	Reactants & reagents	Target product	Predicted product (w/o KL)		Predicted product (w/ KL)
			SMILES	IUPAC	
1					Correct
2			Correct		Correct
3					
4			Correct		Correct
5					Correct
6				Correct	Correct

Scheme 10: Reaction step-inconsistent reactions that transition from inconsistent to consistent predictions after adding KL divergence loss

	Reactants & reagents	Target product	Predicted product (w/o KL)		Predicted product (w/ KL)
			SMILES	IUPAC	
1			Correct		Correct
2					
3				Correct	
4				Correct	Correct
5			Correct	COC1=CC=C(Cl)C=C1S(=O)(=O)NC1=CC=C(C **Invalid**	Correct

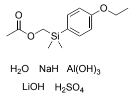
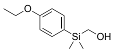
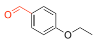
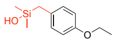
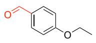
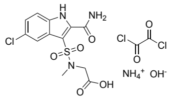
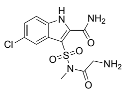
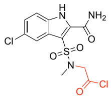
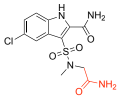
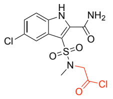
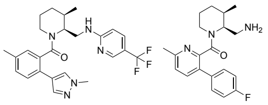
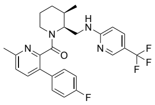
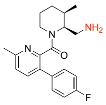
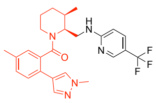
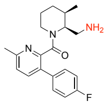
Scheme 11: Minor inconsistent reactions that transition from inconsistent to consistent predictions after adding KL divergence loss

	Reactants & reagents	Target product	Predicted product (w/o KL)		Predicted product (w/ KL)
			SMILES	IUPAC	
1			Correct		Correct
2			Correct		Correct
3			Correct		Correct

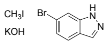
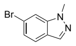
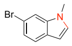
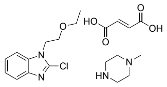
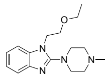
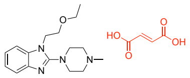
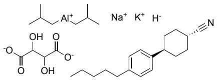
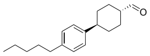
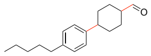
Inconsistent-to-consistent transitions

Here we list the six reactions that transition from consistent to inconsistent predictions between SMILES and IUPAC inputs after adding KL divergence loss.

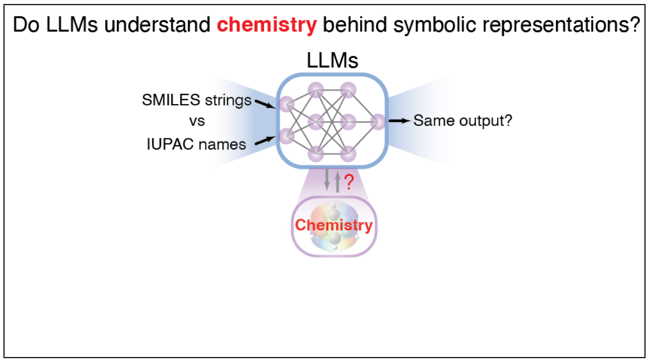
Scheme 12: Complicated reactions that transition from consistent to inconsistent predictions after adding KL divergence loss

	Reactants & reagents	Target product	Predicted product (w/o KL)		Predicted product (w/ KL)
			SMILES	IUPAC	
1					
2					
3					

Scheme 13: Position inconsistent reactions that transition from consistent to inconsistent predictions after adding KL divergence loss

	Reactants & reagents	Target product	Predicted product (w/o KL)		Predicted product (w/ KL)
			SMILES	IUPAC	
1			Correct		Correct
2			Correct		Correct
3			Correct		Correct

TOC Graphic


Do LLMs understand chemistry behind symbolic representations?
LLMs
SMILES strings
vs
IUPAC names
Same output?
?
Chemistry