# OMNI-MOL: EXPLORING UNIVERSAL CONVERGENT SPACE FOR OMNI-MOLECULAR TASKS

**Chengxin Hu**[1*] **Hao Li**[2*] **Yihe Yuan**[1*] **Zezheng Song**[3] **Haixin Wang**[4†]
[1]National University of Singapore [2]Independent Researcher
[3]University of Maryland, Collage Park [4]University of California, Los Angeles
whx@cs.ucla.edu

**Code:** Omni-Mol-Code **Data:** Omni-Mol-Data

## ABSTRACT

Building generalist models has recently demonstrated remarkable capabilities in diverse scientific domains. Within the realm of molecular learning, several studies have explored unifying diverse tasks across diverse domains. However, negative conflicts and interference between molecules and knowledge from different domain may have a worse impact in threefold. First, conflicting molecular representations can lead to optimization difficulties for the models. Second, mixing and scaling up training data across diverse tasks is inherently challenging. Third, the computational cost of refined pretraining is prohibitively high. To address these limitations, this paper presents **Omni-Mol**, a scalable and unified LLM-based framework for direct instruction tuning. Omni-Mol builds on three key components to tackles conflicts: (1) a unified encoding mechanism for any task input; (2) an active-learning-driven data selection strategy that significantly reduces dataset size; (3) a novel design of the adaptive gradient stabilization module and anchor-and-reconcile MoE framework that ensures stable convergence. Experimentally, Omni-Mol achieves state-of-the-art performance across most of the 15 molecular tasks, demonstrates the presence of scaling laws in the molecular domain, and is supported by extensive ablation studies and analyses validating the effectiveness of its design. The dataset, code and weights of the AI-driven chemistry generalist are open-sourced.

## 1 INTRODUCTION

Large language models (LLMs), especially multimodal LLMs, have achieved significant breakthroughs in various scientific tasks due to their powerful representational capabilities and general reasoning abilities, spanning domains such as medicine (Jee et al., 2024; Zhou et al., 2024), chemistry (Boiko et al., 2023), and biology (Zhang et al., 2024b). This cutting-edge technology has also sparked an increasing number of studies exploring how to align molecular representation spaces with textual representation spaces (Cao et al., 2025; Chen et al., 2024; Fang et al., 2024; Cao et al., 2024). These works hold great promise to build powerful AI chemist for advancing molecule captioning, property/structure prediction, and text-conditioned de novo drug design.

The first step in creating an AI chemist is to develop a generalist model with universal capabilities, enabling it to understand diverse molecular structures and their interactions under multiple chemical domains. Pioneering works, such as Text+Chem T5 (Christofidellis et al., 2023), introduces the first multi-domain, multi-task language model capable of unifying molecular and textual representations. Following this, PRESTO (Cao et al., 2024) further enhances performance by progressively improving multimodal LLMs through cross-modal alignment and multi-graph understanding. Similarly, in the field of general LLMs, the platonic representation (Huh et al., 2024) introduces the concept of the multitask scaling hypothesis, which suggests that as models are trained on an increasing number of tasks, they are driven to develop representations capable of addressing all tasks. They highlight the potential of constructing the generalist model capable of handling diverse molecular tasks.

However, we have yet to observe a model that achieves outstanding performances across as many tasks as possible, nor have we seen a clear trend toward scalability in this direction. For instance,

[*]Equal contribution, listing order is alphabetic.
[†]Corresponding author.

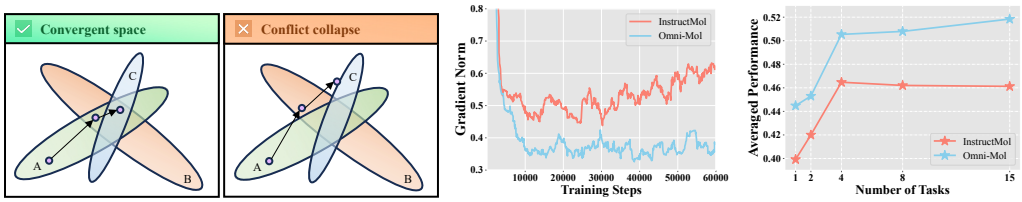



Figure 1: (Left[1]) Illustration of the universal convergent representation space for omni-molecule tasks. (Left[2]) Conflict between tasks from different domains makes vanilla models fail to converge. (Right[1]) Gradient norm of unified training on 15 tasks. Due to the conflicts of multiple tasks, the gradient norm of InstructMol competes and shows a significant increase, while the gradient norm of Omni-Mol remains relatively stable. (Right[2]) The scaling trend with task numbers on reagent prediction. As the number of tasks increases, Omni-Mol is benefited and achieves better performances averagely, while InstructMol fails to scale up.

InstructMol (Cao et al., 2025) attempts to scale up large language models but yields negligible gains, while PRESTO relies on a complex training strategy and requires extensive computational resources for pre-training. We propose that the fundamental challenge is *conflict collapse*, illustrated in Figure 1 left, which limits the emergence of truly generalist model in three key ways. **First**, potential conflicts may arise among various functional groups within a molecule and across the entire molecular structure, making it difficult to optimize the semantic relationships among different molecular representations. **Second**, data with conflicts from different domains often exhibit divergent distributions and interfere with each other, rendering it elusive to determine an ideal training data mixture. **Third**, the complexity of multi-task conflicts grows explosively as the volume of molecular data increases, requiring models with limited capacity to consume significantly greater resources in order to resolve these conflicts.

The recognition of the existing limitations naturally raises a pivotal question:

*Is it possible to develop a generalist model that converges to a universal representation space for omni-molecular tasks?*

It drives us to develop Omni-Mol, a scalable and unified LLM-based framework for direct instruction tuning. Omni-Mol is the first generalist to effectively mitigate *conflict collapse* in three respects: **(1)** Omni-Mol proposes a unified encoding mechanism applicable to any task input, leveraging the most comprehensive instruction-following omni-molecular dataset to date, which comprises 1.8M samples across 15 tasks. **(2)** From the perspective of data, Omni-Mol employs an active learning-based dynamic data selection after recognizing that not all tasks are equally important. We significantly reduced the dataset size to 40% of its original volume while maintaining comparable performance with the full dataset. **(3)** Our novel designed adaptive gradient stabilization successfully mitigate the rapid gradients growth caused by task conflicts (shown in Figure 1 right). Besides, our anchor-and-reconcile mixture-of-expert (MoE) architecture reduces interference by dynamically routing tasks to reconcile experts and capturing common knowledge through anchor experts.

Extensive experiments show that we achieve significant improvements across 15 tasks simultaneously, setting new state-of-the-art results among LLM-based models. Additionally, we observe that Omni-Mol scales effectively with increases in data volume, model size, and the number of tasks, indicating the model's great potential under larger computational budgets. Furthermore, by analyzing the representations of models trained with progressively more tasks, we discover that the representations become increasingly similar as the number of tasks grows. This provides robust evidence supporting our hypothesis of convergence space toward a universal molecular representation.

## 2 METHOD

Omni-Mol is a multimodal LLM framework to handle $K$ diverse molecular tasks simultaneously. It comprises a language model, a graph encoder $f_{\mathcal{G}}$, and a projector $f_p$. The inputs include a text instruction $\mathbf{X}_I$, a SELFIES string $\mathbf{X}_S$, and the graph data $\mathbf{X}_G$ corresponding to the input molecules, where $\mathbf{X}_G$ is converted from $\mathbf{X}_S$ using RDKit (Landrum et al., 2013). We model the response $\mathbf{Y}$ as the probability of the next token as: $P(\mathbf{Y}|\mathbf{X}_I, \mathbf{X}_S, \mathbf{H}_G) = \prod_{i=1}^{L} P_\theta(\mathbf{Y}_i|\mathbf{X}_I, \mathbf{X}_S, \mathbf{H}_G, \mathbf{Y}_{<i})$, where $\mathbf{H}_G = f_p(f_{\mathcal{G}}(\mathbf{X}_G))$, and $\theta$ is the parameter. The graph encoder encodes the molecule graph into its representation $\mathbf{h_g} \in \mathbb{R}^{n \times d_1}$, where $n$ is the length of the representation, the projector then projects its dimension to the LLM's hidden size and obtain $\mathbf{H}_G \in \mathbb{R}^{n \times d_2}$.

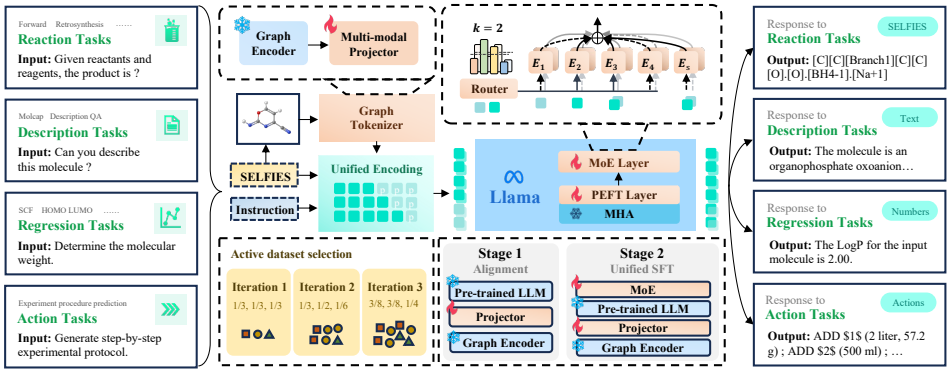



Figure 2: Overview of our proposed Omni-Mol, a scalable and unified LLM-based framework for direct instruction tuning.

### 2.1 Unified Encoding for Instruction Tuning

**Unified Input format.** We collect and format the data of any task into the following structure.

- Instruction $\mathbf{X}_I$: A brief, clear, and distinguishable guidance for what task to perform.
- Input $\mathbf{X}_S$: A sequence of molecules represented using SELFIES separated by a dot.
- Output $\mathbf{Y}$: The output corresponding to the task, which may be a number, a SELFIES representation, or a textual description, all in text string.

The data will be processed according to a specific template, details can be found in Appendix K.

**Unified Encoding.** To unify tokens from diverse tasks and modalities, and to enable parallel training on samples of varying lengths, we apply uniform padding to the mini-batch samples. First, padding tokens are added to the right side of the text input, and the batch is then passed through an embedding layer to obtain text embeddings. Next, since the number of atoms in each molecule varies, the number of graph tokens per sample differs as well. After inserting the graph tokens, we continue to add padding token embeddings to the right end of the sequence, ensuring the mini-batch becomes a well-formed tensor. We then generate an attention mask based on the padded tensor and assign an '*ignore_index*' in the labels to prevent Omni-Mol from learning to generate padding tokens.

### 2.2 Active Learning-based Data Selection

To facilitate effective unified tuning, we systematically screen data from multiple tasks to minimize redundancy and conflicts, while determining an optimal mixing ratio. Actually, not all task-specific data is equally crucial. Inspired by Yu et al. (2024b), we employ an iterative task-centric data filtering approach to actively screen multiple datasets, substantially reducing training costs.

Given $K$ task collections $\mathcal{T} = \bigcup_{q=1}^{K} \mathcal{T}_q$, along with a total budget $\mathcal{B}$ and a maximum iteration count $J$. For the $j$-th iteration $\left(1 \leq j \leq J\right)$, we select a data portion of size $\alpha_j = \mathcal{B}/J$. Assume the initial parameters of the model is $\theta_0$. For the $K$ subsets, we first initialize the distribution $\boldsymbol{\pi}^0 = \left(\pi_1^0, \pi_2^0, \ldots, \pi_K^0\right)$, such that $\pi_q^0 = 1/K$, Then, we sample each task dataset $\mathcal{T}_q$ to a subset $\mathcal{T}_q^{(1)}$ according to the distribution and create the sampled training set $\mathcal{T}^{(1)}$. We then finetune $\theta_0$ on $\mathcal{T}^{(1)}$ for $M$ epochs, yielding the updated parameter $\theta_1$.

After obtaining $\theta_1$, we perform inference on the sampled dataset $\mathcal{T}^{(1)}$, we collect sample pair $(u_{p,+}, u_{p,-})$ indexed by $p$, where $u_{p,+}$ is the ground truth and $u_{p,-}$ is the answer generated by the model. For each sample pair, we calculate the score of the sample by:

$$\nu(u_{p,+}, u_{p,-}) = \max\{\rho(\cdot,\cdot)\} - \rho(u_{p,+}.u_{p,-}) \tag{1}$$

where $\rho(\cdot,\cdot)$ is a normalized metric function for each respective task. For each sample pair $(u_{p,+}, u_{p,-})$ in the task dataset $\mathcal{T}_q$, we define the average rating as:

$$\mu_q = \frac{1}{|\mathcal{T}_q|} \sum_{(u_{p,+}, u_{p,-}) \in \mathcal{T}_q} \nu\left(u_{p,+}, u_{p,-}\right). \tag{2}$$

A larger $\mu_q$ indicates higher complexity, implying that samples in this task are more instrumental for the model. At the end of the $j$-th iteration, we reweight the distribution via:

$$\pi_q'^{(j)} = \frac{\mu_q}{\sum_{r=1}^{K} \mu_r} \pi_q^{(j-1)}, \quad \pi_q^{(j)} = \frac{\pi_q'^{(j)}}{\sum_{r=1}^{K} \pi_r'^{(j)}} \tag{3}$$

We then select an additional batch of size $\alpha_{j+1} = \mathcal{B}/J$ denoted $\mathcal{T}^{(j+1)}$, from the *unused portion* of the dataset. We concatenate it with the previously chosen data for further training. This *sample–train–evaluate–reweight* routine continues until $J$ total iterations are reached.

### 2.3 Stable & MoE Expanded Framework

**Adaptive Gradient Stabilization.** During unified training, each mini-batch consists of samples from several tasks. The resulting loss and gradients will be a combination of contributions from each task. Let $B_k$ denotes the number of samples for task $k$, and $T$ the sequence length. The total loss for the model can then be formulated as:

$$\mathcal{L}_{\text{total}} = \frac{1}{B} \sum_{k=1}^{K} \sum_{b=1}^{B_k} \sum_{t=1}^{T_k} \mathcal{L}(m_{b,t}^{(k)}, o_{b,t}^{(k)}) \tag{4}$$

where $m_{b,t}^{(k)}$ and $o_{b,t}^{(k)}$ are the predicted logits and label for batch element $b$ at time step $t$ in task $k$.

Actually, we observe a noticeable increase in gradient norms (shown in Figure 1 right), which leads to training instability and hindered faster loss convergence. We attribute the main cause of these divergences to issues arising from the softmax operation when handling tasks drawn from different domains with substantially varied entropy, stemming from softmax's translation invariance (*i.e.*, $softmax\,(z) = softmax\,(z + c)$). Since all model parameters are shared among multiple tasks, each task competes by incrementally growing its norms. while this is not immediately detrimental, it leads to divergence once norms extend beyond the effective range of bf16 (Kalamkar et al., 2019). To mitigate this problem, we employ an adaptive coefficient $\gamma_\theta = \frac{\alpha_\theta}{||r||_p} + \beta_\theta$ during the parameter-efficient fine-tuning with LoRA (Hu et al., 2021), where $\alpha_\theta$ and $\beta_\theta$ are learnable variables as scaling factors and $r$ represents the rank of LoRA. Through this mechanism, gradients can be adaptively stabilized on data $\mathcal{D}$:

$$\nabla_{\boldsymbol{\Delta\mathcal{W}}} = \frac{\partial \mathcal{L}_{total}\left(\mathcal{D}; \boldsymbol{\mathcal{W}}_0 + \gamma_\theta \cdot \boldsymbol{\Delta\mathcal{W}}\right)}{\partial \boldsymbol{\Delta\mathcal{W}}} \tag{5}$$

where $\Delta\mathcal{W}$ is the updated parameters to the pre-trained $\mathcal{W}_0$. More details are in Appendix D.3.

**Anchor-and-Reconcile Experts Expansion.** Omni-Mol needs to learn a wide range of tasks and handle multiple modalities, including graph features, text, and SELFIES. While SELFIES is treated as regular text input to the LLM, it inherently differs significantly from natural language semantics, requiring the model to separately learn how to understand and generate SELFIES expressions.

We aim for the model to simultaneously learn general knowledge while also differentiating for different modalities and tasks. Hence, we borrow the idea of MoE (Dai et al., 2024) and perform upcycling (Komatsuzaki et al., 2023; Lin et al., 2024). We first construct $\mathcal{N}$ reconcile experts, each targeting specialized knowledge areas, and dynamically balances conflicting signals among these experts to effectively mitigate task-level conflicts. Besides, we introduce an additional anchor expert to learn the common knowledge that underpins fundamental understanding across tasks, by consistently capturing and aligning shared features to maintain a stable global representation. To be specific, for a regular decoder layer $l = 1 \ldots L$ from a pre-trained LLM, $h_l' = h_{l-1} + \text{MHA}_\phi(\text{LN}(h_{l-1}))$ and $h_l = h_l' + \text{FFN}_\gamma(\text{LN}(h_l'))$, where $\phi$ and $\gamma$ are parameters of the pre-trained LLM, MHA and FFN refers to Multi-Head Attention and Fully Forward Network respectively. We convert the decoder layer into:

$$\begin{aligned} h_l' &= h_{l-1} + \text{MHA}_{\phi'}(\text{LN}(h_{l-1})) \\ h_l &= \begin{cases} h_l' + \text{FFN}_{\gamma'}(\text{LN}(h_l')), & l = 1 \ldots l_{\text{MoE}} \\ h_l' + \text{MoE}_{\gamma_i, \psi}(\text{LN}(h_l')), & l = l_{\text{MoE}} \ldots L \end{cases} \end{aligned} \tag{6}$$

where $l_{\text{MoE}}$ represents the layer starts to utilize MoE. The converted $\phi' = \phi + \Delta\phi_{\text{LoRA}}^{(\text{MHA})}$, $\gamma' = \gamma + \Delta\gamma_{\text{LoRA}}^{(\text{FFN})}$. For MoE layer, we initialize $\mathcal{N} + 1$ experts with the weight of the pre-trained FFN

$\gamma$. Here, it concludes $\mathcal{N}$ reconcile experts to learn specialized knowledge and 1 anchor experts to learn the common knowledge. Let $E_{\gamma_i}$ denotes the $i$-th expert, $i = 1 \ldots \mathcal{N} + 1$. And $\gamma_i$ is the parameter of the $i$-th expert, at the beginning of the training, these experts have identical weights, *i.e.*, $\gamma_1 = \gamma_2 = \cdots = \gamma$. Router $R_\psi$ is random initialized, where $\psi$ is the parameter of the router.

Let $s$ denote the output of the router logits for the $i$-th expert, assume $\mathcal{E}$ experts are chosen the output of the MoE layer can be written as: $\mathbf{x}'_\mathbf{t} = \sum_{i=1}^{\mathcal{N}} \text{TopK}(s, \mathcal{E}) E_{\gamma_i}(\mathbf{x_t}) + E_{\gamma_{\mathcal{N}+1}}(\mathbf{x_t})$.

### 2.4 Optimization

Training strategy of Omni-Mol consists of two stages.

**Stage 1:** We perform multimodal alignment on PubChem (Kim et al., 2022), learning to describe molecules through graph modality features. The input consists of instructions and graph data, excluding SELFIES. Only the multimodal projector $f_p$ is trainable.

**Stage 2:** We fine-tune Omni-Mol by freezing the pre-trained parameters that are wrapped by PEFT (Parameter Efficient Fine-Tuning) adapters, while the adapters, the MoE layers and the multimodal projector stay active.

Training loss of both stages for language modeling is $\mathcal{L}_{\text{LM}} = -\sum_i \log P_\theta(\mathbf{Y}_i | \mathbf{X}_I, \mathbf{X}_S, \mathbf{H}_G, \mathbf{Y}_{<i})$.

For stage 2, we incorporate an additional auxiliary load balancing loss for the MoE layers, assume an input tensor $x \in \mathbb{R}^{B \times T \times d}$, and Top-$K$ experts out of $\mathcal{N}$ is selected, the load balancing loss is: $\mathcal{L}_{\text{aux}} = \frac{1}{B} \sum_{i=1}^{B} \sum_{j=1}^{\mathcal{N}} C_{ij} \cdot \bar{s}_{ij}$, where $C_{ij} = \frac{\mathcal{N}}{T\mathcal{E}} \sum_{t=1}^{T\mathcal{E}} \mathbf{1}\{\text{t'th token selects expert j}\}, s_{ij} = \frac{1}{T} \sum_{t=1}^{T} s_{i,j,t}$ and $\mathbf{1}\{\cdot\}$ is an indicator function. This load balancing loss used in Liu et al. (2024a) additionally considers the sequence-level information.

The total loss is a combination of $\mathcal{L}_{\text{LM}}$ and $\mathcal{L}_{\text{aux}}$ with a coefficient $\lambda$: $\mathcal{L} = \mathcal{L}_{\text{LM}} + \lambda \mathcal{L}_{\text{aux}}$.

### 2.5 Theoretical Analysis

In theory, the regularization effect of omni-molecular tasks training can enable Omni-Mol to learn more general representations. However, demonstrating this phenomenon empirically remains challenging. Our key insight is that as more tasks are learned together, the solution space of the problems becomes progressively smaller. Assume the hypothesis space $\mathcal{V}$ of the model, and the solution of a task $i$ is $\mathcal{F}_i$, where $\mathcal{F}_i \subseteq \mathcal{V}$, for omni-molecular tasks learning with $n$ different tasks, the solution space will be: $\mathcal{F}^{(n)}_{\text{general}} = \bigcap_{i=1}^{n} \mathcal{F}_i$.

**Theorem 2.1.** *For $m > n, m, n \in \mathbb{Z}^+$, we have $\mathcal{F}^{(m)}_{general} \subseteq \mathcal{F}^{(n)}_{general}$. If $\forall n \leq i, j \leq m, i \neq j$, we have $\mathcal{F}_i \neq \mathcal{F}_j$, then $\mathcal{F}^{(m)}_{general} \subsetneq \mathcal{F}^{(n)}_{general}$.*

The proof of Theorem 2.1 can be found in Appendix I.1 Therefore, as the number of tasks $n$ increases, the similarity between the representations learned from the solution space of $n-1$ tasks and the solution space of $n$ tasks will become increasingly higher. Let $R_n \in \mathcal{F}^{(n)}_{\text{general}}$ denotes the representation learned from a solution space, consider a series of learned representation $S_R = \{R_1, R_2, \ldots, R_n\}$ learned from $S_\mathcal{F} = \{F^{(1)}_{\text{general}}, F^{(2)}_{\text{general}}, \ldots, F^{(n)}_{\text{general}}\}$. Assume function $\Gamma(\cdot, \cdot)$ measures the similarity between two representations. we expect that, $\exists N \in \mathbb{Z}^+$, such that $\forall n > N, \exists i < N$, we have $\Gamma(R_n, R_{n+1}) > \Gamma(R_i, R_{i+1})$.

In our experimental validation, we construct a series of mixed datasets comprising 1, 2, 4, and 8 tasks to form the solution space sequence $S_\mathcal{F}$. on each of these datasets and subsequently extract representations for a specific task. Finally, we calculate the similarity between representations $R_i$ and $R_j$ for $i, j = 0, \ldots, 3$. See Appendix D.5 for the calculation of similarity.

## 3 Experiments

We aim to address the following concerns: (1) Compared with existing baselines, can Omni-Mol achieve the best performances on the comprehensive omni-molecular datasets with 15 tasks simultaneously? (2) Is Omni-Mol a scalable framework with the capacity and potential to solve complex molecular tasks? (3) Are all key components of Omni-Mol essential for solving conflict collapse? (4) How can we verify that Omni-Mol converges reliably and progressively refines its representations toward a universal convergent space? We begin by describing the experimental setup, then answer all the questions in the subsequent sections.

| Model | Type | #Param | Exact ↑ | BLEU ↑ | Levenshtein ↓ | RDK ↑ | MACCS ↑ | Morgan ↑ | Validity ↑ |
|---|---|---|---|---|---|---|---|---|---|
| Forward Reaction Prediction Task | | | | | | | | | |
| Vicuna (Zheng et al., 2023b) | In-Context Learning | 6.7B | 0.000 | 0.057 | 41.690 | 0.007 | 0.016 | 0.006 | 0.059 |
| LLaMA2 (Touvron et al., 2023) | Specialist(PEFT) | 6.7B | 0.012 | 0.804 | 29.947 | 0.499 | 0.649 | 0.407 | 1.000 |
| Mol-Instruction (Fang et al., 2024) | Specialist(PEFT) | 6.7B | 0.045 | 0.654 | 27.262 | 0.313 | 0.509 | 0.262 | 1.000 |
| HIGHT (Chen et al., 2024) | Specialist(PEFT) | 6.7B | 0.293 | 0.935 | 16.687 | 0.774 | 0.618 | 0.566 | 1.000 |
| InstructMol (Cao et al., 2025) | Specialist(PEFT) | 6.7B | 0.536 | 0.967 | 10.851 | 0.776 | 0.878 | 0.741 | 1.000 |
| PRESTO* (Cao et al., 2024) | Generalist | 3.2B | 0.691 | 0.976 | **6.525** | 0.871 | 0.931 | 0.841 | 1.000 |
| **Omni-Mol** | Generalist | **1.7B** | **0.718** | **0.981** | 6.528 | **0.878** | **0.934** | **0.854** | **1.000** |
| Retrosynthesis Task | | | | | | | | | |
| Vicuna (Zheng et al., 2023b) | In-Context Learning | 6.7B | 0.000 | 0.057 | 46.877 | 0.025 | 0.030 | 0.021 | 0.017 |
| LLaMA2 (Touvron et al., 2023) | Specialist(PEFT) | 6.7B | 0.000 | 0.283 | 53.510 | 0.136 | 0.294 | 0.106 | 1.000 |
| Mol-Instruction (Fang et al., 2024) | Specialist(PEFT) | 6.7B | 0.009 | 0.705 | 31.227 | 0.283 | 0.487 | 0.230 | 1.000 |
| HIGHT (Chen et al., 2024) | Specialist(PEFT) | 6.7B | 0.202 | 0.914 | 20.194 | 0.772 | 0.623 | 0.577 | 0.999 |
| InstructMol (Cao et al., 2025) | Specialist(PEFT) | 6.7B | 0.407 | 0.941 | 13.967 | 0.753 | 0.852 | 0.714 | 1.000 |
| PRESTO* (Cao et al., 2024) | Generalist | 3.2B | 0.531 | 0.958 | 10.298 | 0.823 | 0.887 | 0.790 | 1.000 |
| **Omni-Mol** | Generalist | **1.7B** | **0.559** | **0.961** | **9.263** | **0.840** | **0.900** | **0.809** | **1.000** |
| Reagent Prediction Task | | | | | | | | | |
| Vicuna (Zheng et al., 2023b) | In-Context Learning | 6.7B | 0.000 | 0.010 | 27.948 | 0.038 | 0.002 | 0.001 | 0.007 |
| LLaMA2 (Touvron et al., 2023) | Specialist(PEFT) | 6.7B | 0.000 | 0.283 | 53.510 | 0.136 | 0.294 | 0.106 | 1.000 |
| Mol-Instruction (Fang et al., 2024) | Specialist(PEFT) | 6.7B | 0.044 | 0.224 | 23.167 | 0.237 | 0.364 | 0.213 | 1.000 |
| HIGHT (Chen et al., 2024) | Specialist(PEFT) | 6.7B | 0.067 | 0.482 | 27.167 | 0.462 | 0.346 | 0.303 | 1.000 |
| InstructMol (Cao et al., 2025) | Specialist(PEFT) | 6.7B | 0.129 | 0.610 | 19.664 | 0.444 | 0.539 | 0.400 | 1.000 |
| PRESTO* (Cao et al., 2024) | Generalist | 3.2B | 0.212 | 0.712 | 16.313 | 0.544 | 0.607 | 0.479 | 1.000 |
| **Omni-Mol** | Generalist | **1.7B** | **0.257** | **0.763** | **13.558** | **0.601** | **0.660** | **0.556** | **1.000** |

Table 1: Comprehensive comparisons on three reaction tasks. PEFT is short for parameter-efficient fine-tuning. PRESTO* represents our re-implementation based on source codes.

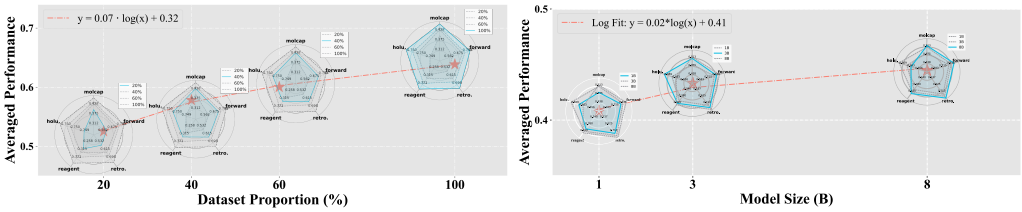



Figure 3: Scaling trend of Omni-Mol. (Left) The scaling trend respect to dataset proportion, metrics are averaged across tasks, (Right) The scaling trend respect to model size, the metrics of Quantum Mechanics Property Prediction task are normalized. we observed a clear log scaling behavior.

### 3.1 Setup and Baselines

**Datasets.** To construct a general-purpose model, we select 15 tasks across 4 categories to cover as many diverse tasks as possible. Details of datasets can be found in Appendix B.

**Baselines.** To ensure a fair comparison, we first choose representative LLM-based models such as InstructMol and HIGHT, and also report several previous baselines, including Mol-Instruction, Llama, Vicuna, among others, some of which are derived through In-Context Learning. For datasets with fewer models, we re-implement PRESTO as baseline.

**Backbone.** We utilize LLaMA 3.2-1B (Dubey et al., 2024) as the backbone, a single linear layer as the projector, and MoleculeSTM (Mustafa et al., 2022) as the graph encoder for processing molecular graphs. For MoE expansion, we set $l_{\text{moe}} = 1/4L$ and number of experts to 3. More details about model implementation can be found in Appendix D.

**Training Details.** We use PyTorch (Paszke et al., 2019) with DeepSpeed ZeRO-2 (Rajbhandari et al., 2020) for more efficient parallel training. For unified tuning, we train 15 epochs with LoRA rank of 64. For separate tuning, Omni-Mol is trained for 10 epochs with the same LoRA configuration. The learning rate is set to 8e-5 from grid search for all experiments. For experiment consistency, random seed is set to 0. More details can be found in Appendix E.

### 3.2 Main Results

Here, we obtain the answer that *Omni-Mol can achieve the best performance across almost all tasks.* As the results shown in Table 2.5, we have the following observations. Omni-Mol significantly outperforms all specialist baselines while utilizing only 25% of the parameters. Furthermore, Omni-Mol surpasses the corresponding state-of-the-art generalist baseline by an average of approximately 1%, 4%, 13%, 15%, and 40% across forward prediction, retrosynthesis, reagent prediction, molcap, and Description QA separately. That is to say, Omni-Mol achieves superior performance with greater

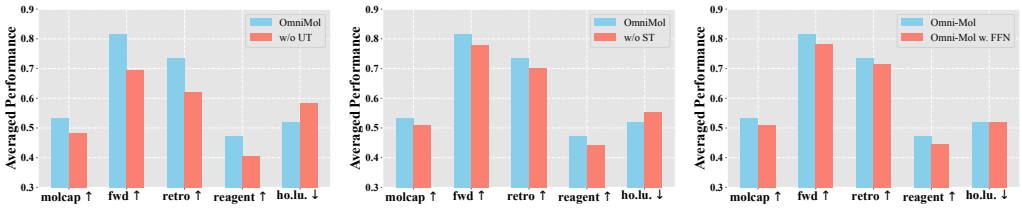



Figure 5: Ablation studies: (Left) Ablation of unified training. The performance is averaged across all metrics. (Mid) Ablation of adaptive gradient stabilization. (Right) Ablation of MoE expansion.

parameter efficiency, demonstrating its effectiveness in becoming a general AI chemist. Due to the page limit, we only report 3 tasks, the remaining results can be found in Appendix F.

### 3.3 Is Omni-Mol a Scalable Framework?

One critical property of LLMs is their scaling behavior in relation to both model and data size. In this study, we demonstrate that *Omni-Mol* is a scalable framework by conducting three distinct types of scaling experiments: **(1)** We select three different sizes of LLMs from the LLaMA 3 series, 1B, 3B, and 8B, for language backbone scaling. **(2)** We evaluate the impact of dataset size by down-sampling the original dataset to 20%, 40%, 60%, and 100% of its full size. **(3)** To examine task scaling, we train Omni-Mol on different numbers of tasks, specifically, 1, 2, 4, 8, and 15 tasks, and observe the performance of individual tasks within these multi-task settings.

**(1)** As shown in the left of Figure 3, we observe a clear logarithmic scaling trend as the dataset proportion increases. The relationship between the model's average performance and the dataset proportion can be approximately expressed as $y = 0.07 \cdot \log(x) + 0.41$. The overlaid radar charts further demonstrate that this trend holds true across all tasks.

**(2)** As shown on the right side of Figure 3, the performance of Omni-Mol across all tasks increases as the model size grows. We also observe a clear logarithmic scaling trend, where this relationship can be approximated as $y = 0.02 \cdot \log(x) + 0.32$.

**(3)** The average results are shown in the right side of Figure 1 based on reagent prediction. This indirectly supports our theory about the value of a high-quality, universal representation space. In contrast, InstructMol experiences a performance decline when the number of tasks exceeds eight, as conflicts between tasks hinder its ability to simultaneously learn all tasks effectively.

### 3.4 Is Unified Instruction Tuning Essential?

One key aspect of Omni-Mol is its ability to leverage unified learning across omni-molecular tasks, enabling the convergence to more generalizable representations. To evaluate this, we compare the performance of separate tuning on individual tasks (Omni-Mol w/o UT) against our unified tuning. As shown in Figure 5, Omni-Mol w/o UT performs significantly worse across five tasks compared to ours. This indicates that the representations learned through unified tuning are superior and benefit from shared knowledge. Interestingly, even tasks that are not directly related, such as the molcap task, which is distinct from both reaction and regression tasks, still show improvements with unified tuning. Additional ablation results are provided in Appendix G.

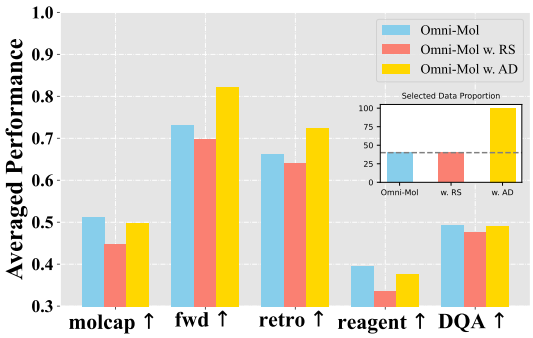



Figure 4: Ablation study of active learning-based dataset selection. Omni-Mol w. RS: we randomly sample all datasets and mix them, Omni-Mol w. AD: we use all data samples.

### 3.5 Is Dataset Selection Essential?

In Figure 4, we report the results compared with random sampling (Omni-Mol w. RS) and all-data training (Omni-Mol w. AD). The figure also includes the ratio of full data to down-sampled data.

With 40% of the data, the Omni-Mol significantly outperforms (Omni-Mol w. RS) and surpasses (Omni-Mol w. AD) on the MolCap, Description QA, and Reagent tasks. This supports the hypothesis that redundant samples exist within the data, and selecting appropriate subsets from different datasets can lead to better performance. In the remaining two tasks, our results are also comparable, considering that we used only a small portion of the data.

### 3.6 Can We Mitigate the Conflict Collapse?

**How Do Adaptive Gradient Stabilization Helps?** We compare our Omni-Mol with Omni-Mol w/o ST, which replaces our adaptive module with the standard LoRA adapter. As shown in the middle of Figure 5, Omni-Mol w/o ST consistently exhibits lower performances than Omni-Mol across all tasks. This consistent decline underscores the effectiveness of our adaptive module in enhancing performance by mitigating task conflicts. This mitigation of task conflicts ensures that Omni-Mol can leverage shared knowledge without detrimental interference, thereby enhancing its ability to generalize across various tasks and modalities.

**Is MoE Expansion Essential?** We conduct an ablation study by replacing our Anchor-and-Reconcile Experts Expansion with a single activated Feed-Forward Network (FFN). The comparison results are shown in the right of Figure 5. We observe that Omni-Mol consistently outperforms the Omni-Mol w. FFN across all tasks, including molcap, forward prediction, retrosynthesis, and reagent prediction. Additionally, for the homo-lumo, where lower values are preferable, Omni-Mol achieves a better score. This demonstrates that Omni-Mol effectively enhances performance by leveraging specialized experts, as opposed to a single FFN. The most significant improvement is observed in forward prediction, where the diverse experts contribute to better generalization. More insights on MoE layers can be found in Appendix A.

### 3.7 Convergence Analysis via Mutual Similarity

We aim to validate the theory in Section 2.5 and verify the convergence of Omni-Mol, we compute the representation sequence with model trained on 1, 2, 4, and 8 tasks, we use `mutual_knn` (Huh et al., 2024) as our similarity function $\Gamma(\cdot, \cdot)$, the results are shown in Figure 6.

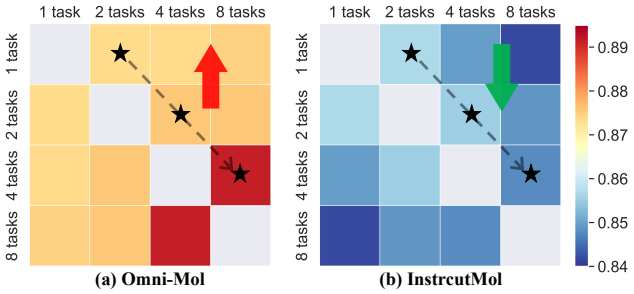



Figure 6: Demonstration of similarity scores heatmap for methods trained on varying numbers of tasks. Surprisingly, Omni-Mol shows rising similarity scores as the task count increases (direction of the black dashed line). Compared to the decreasing trend of InstructMol, this suggests that Omni-Mol converges toward a more consistent representation space universally.

Obviously, when the number of tasks increases, the similarity of the representations learned by Omni-Mol also increases. This indicates that the model's representations are gradually converging. This outcome supports our hypothesis that adding more tasks reduces the size of the model's general solution space. As a result, the model is forced to learn representations within a smaller and more focused space, leading to the convergence of the representations. Ultimately, these representations converge to a universal form that can effectively solve all tasks.

Interestingly, in the mutual similarity analysis of InstructMol, we observe the opposite trend. As the number of tasks increases, the representations learned by InstructMol become progressively less similar to those learned previously. This suggests that with each added task, the changes in the solutions learned by InstructMol become larger, indicating that it is unable to converge to a universal representation space through unified training. In fact, the model may be moving further away from such a space.

### MEANINGFULNESS STATEMENT

We consider that the meaningful representation of life refers to a model's exploration of the underlying principles governing all living and physical phenomena. Omni-Mol represents an initial attempt to develop a chemical AI model with as universal a representation as possible. We aim for this approach to aid the community in building a more general and powerful AI chemist.

## REFERENCES

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

Derek T Ahneman, Jesús G Estrada, Shishi Lin, Spencer D Dreher, and Abigail G Doyle. Predicting reaction performance in c–n cross-coupling using machine learning. *Science*, 360(6385):186–190, 2018.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023.

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

He Cao, Yanjun Shao, Zhiyuan Liu, Zijing Liu, Xiangru Tang, Yuan Yao, and Yu Li. PRESTO: Progressive pretraining enhances synthetic chemistry outcomes. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 10197–10224, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.findings-emnlp.597.

He Cao, Zijing Liu, Xingyu Lu, Yuan Yao, and Yu Li. Instructmol: Multi-modal integration for building a versatile and reliable molecular assistant in drug discovery. In *Proceedings of the 31st International Conference on Computational Linguistics (COLING 2025)*, Abu Dhabi, UAE, Jan 2025.

Yongqiang Chen, Quanming Yao, Juzheng Zhang, James Cheng, and Yatao Bian. Hight: Hierarchical graph tokenization for graph-language alignment. *arXiv preprint arXiv:2406.14021*, 2024.

Dimitrios Christofidellis, Giorgio Giannone, Jannis Born, Ole Winther, Teodoro Laino, and Matteo Manica. Unifying molecular and textual representations via multi-task language modelling. In *International Conference on Machine Learning*, pp. 6140–6157. PMLR, 2023.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

Wenliang Dai, Junnan Li, D Li, AMH Tiong, J Zhao, W Wang, B Li, P Fung, and S Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. arxiv 2023. *arXiv preprint arXiv:2305.06500*, 2, 2023.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, pp. 1–16, 2023.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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

Yin Fang, Xiaozhuan Liang, Ningyu Zhang, Kangwei Liu, Rui Huang, Zhuo Chen, Xiaohui Fan, and Huajun Chen. Mol-instructions: A large-scale biomolecular instruction dataset for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Tlsdsb6l9n.

Janna Hastings, Gareth Owen, Adriano Dekker, Marcus Ennis, Namrata Kale, Venkatesh Muthukrishnan, Steve Turner, Neil Swainston, Pedro Mendes, and Christoph Steinbeck. Chebi in 2016: Improved services and an expanding collection of metabolites. *Nucleic acids research*, 44(D1): D1214–D1219, 2016.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis. *arXiv preprint arXiv:2405.07987*, 2024.

Justin Jee, Christopher Fong, Karl Pichotta, Thinh Ngoc Tran, Anisha Luthra, Michele Waters, Chenlian Fu, Mirella Altoe, Si-Yang Liu, Steven B Maron, et al. Automated real-world data integration improves cancer outcome prediction. *Nature*, pp. 1–9, 2024.

Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with lora. *arXiv preprint arXiv:2312.03732*, 2023.

Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee, Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen, et al. A study of bfloat16 for deep learning training. *arXiv preprint arXiv:1905.12322*, 2019.

Steven M Kearnes, Michael R Maser, Michael Wleklinski, Anton Kast, Abigail G Doyle, Spencer D Dreher, Joel M Hawkins, Klavs F Jensen, and Connor W Coley. The open reaction database. *Journal of the American Chemical Society*, 143(45):18820–18826, 2021.

Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. Pubchem in 2021: new data content and improved web interfaces. *Nucleic acids research*, 49(D1):D1388–D1395, 2021.

Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A. Shoemaker, Paul A. Thiessen, Bo Yu, Leonid Y. Zaslavsky, Jian Zhang, and Evan E. Bolton. Pubchem 2023 update. *Nucleic acids research*, 2022. URL https://api.semanticscholar.org/CorpusID:253182955.

Jing Yu Koh, Daniel Fried, and Russ R Salakhutdinov. Generating images with multimodal language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Walter Kohn and Lu Jeu Sham. Self-consistent equations including exchange and correlation effects. *Physical review*, 140(4A):A1133, 1965.

Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. Sparse upcycling: Training mixture-of-experts from dense checkpoints. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=T5nUQDrM4u.

Greg Landrum et al. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 8(31.10):5281, 2013.

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

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pp. 12888–12900. PMLR, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

Sihang Li, Zhiyuan Liu, Yanchen Luo, Xiang Wang, Xiangnan He, Kenji Kawaguchi, Tat-Seng Chua, and Qi Tian. Towards 3d molecule-text interpretation in language models. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=xI4yNlkaqh.

Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Junwu Zhang, Munan Ning, and Li Yuan. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*, 2024.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024b.

Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. Pre-training molecular graph representation with 3d geometry. In *International Conference on Learning Representations*, 2022.

Shengchao Liu, Weili Nie, Chengpeng Wang, Jiarui Lu, Zhuoran Qiao, Ling Liu, Jian Tang, Chaowei Xiao, and Animashree Anandkumar. Multi-modal molecule structure–text model for text-based retrieval and editing. *Nature Machine Intelligence*, 5(12):1447–1457, 2023.

Zhiyuan Liu, Yaorui Shi, An Zhang, Sihang Li, Enzhi Zhang, Xiang Wang, Kenji Kawaguchi, and Tat-Seng Chua. Reactxt: Understanding molecular" reaction-ship" via reaction-contextualized molecule-text pretraining. *arXiv preprint arXiv:2405.14225*, 2024c.

Daniel Lowe. Chemical reactions from us patents (1976-sep2016). https://figshare.com/articles/dataset/Chemical_reactions_from_US_patents_1976-Sep2016_/5104873, 2017.

Jieyu Lu and Yingkai Zhang. Unified deep learning model for multitask reaction predictions with explanation. *Journal of chemical information and modeling*, 62(6):1376–1387, 2022.

Yizhen Luo, Jiahuan Zhang, Siqi Fan, Kai Yang, Yushuai Wu, Mu Qiao, and Zaiqing Nie. Biomedgpt: Open multimodal generative pre-trained transformer for biomedicine. *arXiv preprint arXiv:2308.09442*, 2023.

Nakata Maho. The pubchemqc project: A large chemical database from the first principle calculations. In *AIP conference proceedings*, volume 1702. AIP Publishing, 2015.

Basil Mustafa, Carlos Riquelme, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. Multimodal contrastive learning with limoe: the language-image mixture of experts. *Advances in Neural Information Processing Systems*, 35:9564–9576, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Damith Perera, Joseph W Tucker, Shalini Brahmbhatt, Christopher J Helal, Ashley Chong, William Farrell, Paul Richardson, and Neal W Sach. A platform for automated nanomole-scale reaction screening and micromole-scale synthesis in flow. *Science*, 359(6374):429–434, 2018.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16. IEEE, 2020.

Lars Ruddigkeit, Ruud Van Deursen, Lorenz C Blum, and Jean-Louis Reymond. Enumeration of 166 billion organic small molecules in the chemical universe database gdb-17. *Journal of chemical information and modeling*, 52(11):2864–2875, 2012.

Philippe Schwaller, Alain C Vaucher, Teodoro Laino, and Jean-Louis Reymond. Prediction of chemical reaction yields using deep learning. *Machine learning: science and technology*, 2(1):015016, 2021.

Fang Sun, Zijie Huang, Haixin Wang, Yadi Cao, Xiao Luo, Wei Wang, and Yizhou Sun. Graph fourier neural odes: Bridging spatial and temporal multiscales in molecular dynamics. *arXiv preprint arXiv:2411.01600*, 2024.

Ruoxi Sun, Hanjun Dai, and Adams Wei Yu. Does gnn pretraining help molecular representation? *Advances in Neural Information Processing Systems*, 35:12096–12109, 2022.

Matt Swain. Pubchempy: A python wrapper for the pubchem pug rest api. `https://github.com/mcs07/PubChemPy`, 2024. Accessed: 2024-12-15.

Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

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

USPTO. Fy 2020 performance and accountability report. Technical report, USPTO, 2020. URL `https://www.uspto.gov/sites/default/files/documents/USPTOFY20PAR.pdf`.

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

Haixin Wang, Xinlong Yang, Jianlong Chang, Dian Jin, Jinan Sun, Shikun Zhang, Xiao Luo, and Qi Tian. Parameter-efficient tuning of large-scale multimodal foundation model. *Advances in Neural Information Processing Systems*, 36:15752–15774, 2023.

Haixin Wang, Jianlong Chang, Yihang Zhai, Xiao Luo, Jinan Sun, Zhouchen Lin, and Qi Tian. Lion: Implicit vision prompt tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 5372–5380, 2024.

David Weininger. 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, 1988.

Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Bruce XB Yu, Jianlong Chang, Haixin Wang, Lingbo Liu, Shijie Wang, Zhiyu Wang, Junfan Lin, Lingxi Xie, Haojie Li, Zhouchen Lin, et al. Visual tuning. *ACM Computing Surveys*, 56(12): 1–38, 2024a.

Simon Yu, Liangyu Chen, Sara Ahmadian, and Marzieh Fadaee. Diversify and conquer: Diversity-centric data selection with iterative refinement. *arXiv preprint arXiv:2409.11378*, 2024b.

Yihang Zhai, Haixin Wang, Jianlong Chang, Xinlong Yang, Jinan Sun, Shikun Zhang, and Qi Tian. When parameter-efficient tuning meets general-purpose vision-language models. *arXiv preprint arXiv:2312.12458*, 2023.

Qianru Zhang, Haixin Wang, Siu-Ming Yiu, and Hongzhi Yin. Graph masked autoencoder for spatio-temporal graph learning. *arXiv preprint arXiv:2410.10915*, 2024a.

Qianru Zhang, Xinyi Gao, Haixin Wang, Siu-Ming Yiu, and Hongzhi Yin. Efficient traffic prediction through spatio-temporal distillation. *arXiv preprint arXiv:2501.10459*, 2025.

Shanghang Zhang, Gaole Dai, Tiejun Huang, and Jianxu Chen. Multimodal large language models for bioimage analysis. *nature methods*, 21(8):1390–1393, 2024b.

Kaizhi Zheng, Xuehai He, and Xin Eric Wang. Minigpt-5: Interleaved vision-and-language generation via generative vokens. *arXiv preprint arXiv:2310.02239*, 2023a.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023b.

Juexiao Zhou, Xiaonan He, Liyuan Sun, Jiannan Xu, Xiuying Chen, Yuetan Chu, Longxi Zhou, Xingyu Liao, Bin Zhang, Shawn Afvari, et al. Pre-trained multimodal large language model enhances dermatological diagnosis using skingpt-4. *Nature Communications*, 15(1):5649, 2024.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

# A INSIGHT FOR BIO-CHEMICAL COMMUNITY

We visualize the routing of tokens representing different functional groups through various experts in Omni-Mol. The results clearly demonstrate that different functional groups activate distinct experts, highlighting that molecular representations differ fundamentally from pure textual semantics. This suggests the existence of intrinsic interactions, particularly between different functional groups, rather than isolated token representations. Our visualization provides strong evidence for this phenomenon, emphasizing the structured nature of molecular representations. This insight offers valuable guidance for future research on integrating molecular representations with LLMs, paving the way for more chemically informed architectures, even more powerful AI chemist.

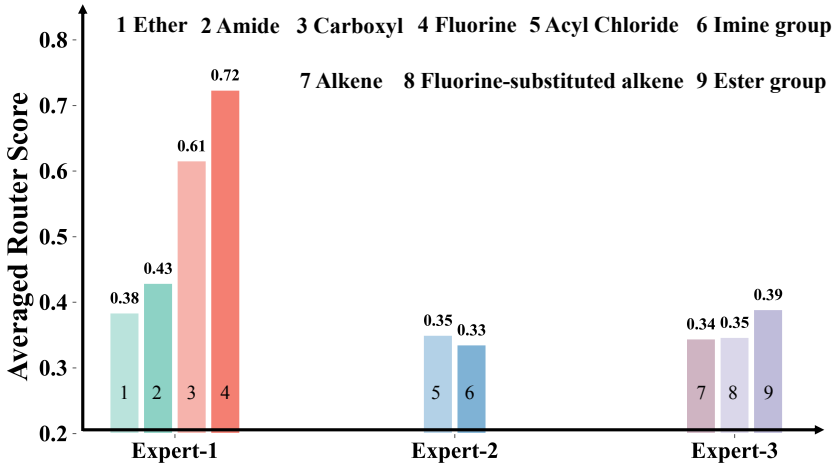



Figure 7: Illustration of router scores in our MoE framework for tokens representing different functional groups. The numbers correspond to specific tokens associated with 9 visualized functional groups.

# B FURTHER DETAILS ON DATASETS

## B.1 COMPREHENSIVE DATASETS CONSTRUCTION

In this subsection, we provide a comprehensive list of the datasets used in our study along with their respective sources. While datasets vary across different papers, we observed that many are derived and processed from common sources. To clarify this overlap, we summarize the information in Table and provide a detailed analysis below.

**(1) USPTO** (USPTO, 2020). The USPTO (United States Patent and Trademark Office) dataset is a widely used large-scale chemical reaction dataset extracted and processed from US patent texts. It encompasses a diverse range of organic reaction types, including esterification, amidation, halogenation, Suzuki coupling, Buchwald–Hartwig coupling, addition reactions, condensation reactions, and redox reactions. Following Fang et al. (2024), for the **Forward Reaction Prediction** task, we extract data from USPTO, and split the dataset into 124,384 training instances and 1,000 test instances. Partially following Cao et al. (2024), for the **Catalyst Prediction** and **Solvent Prediction** tasks, we similarly extract data from USPTO, splitting the training/test sets into 10,079/1,015 and 67,099/7,793, respectively.

USPTO_500_MT (Lu & Zhang, 2022) is a high-quality multi-task reaction prediction dataset, derived from USPTO through manual processing (including data filtering, deduplication, etc.). This subset retains the 500 most common reaction types. Following Fang et al. (2024), for the **Reagent Prediction** task, we split the dataset into 124,384 training instances and 1,000 test instances.

USPTO_500K (Lu & Zhang, 2022), a subset of organic chemical reaction data extracted from USPTO, is widely used in chemoinformatics for the single-step retrosynthesis task. Following Fang et al. (2024), for the **Retrosynthesis** task, the dataset is divided into 128,684 training instances and 1,000 test instances.

USPTO-Applications (Lowe, 2017) is another commonly used subset of USPTO, primarily derived from data samples in patent applications. For the **Experiment Procedure Prediction** task, following Liu et al. (2024c) (along with the introduction of ORD data), we split the dataset into 80% training, 10% validation and 10% test sets.

**(2) ChEBI-20** (Edwards et al., 2021). ChEBI-20 is derived from the ChEBI-16 (Hastings et al., 2016) dataset, with further annotations based on PubChem, forming a comprehensive database of chemical entities in the field of biochemistry. Compared to Fang et al. (2024), ChEBI-20 provides a more extensive and detailed description of chemical compounds. Therefore, for the **Molecular Captioning** task, following Cao et al. (2025), we split the ChEBI-20 dataset (which contains a total of 33,010 instances) into 26,420 training instances, 3295 validation instances and 3,295 test instances.

**(3) QM9** (Wu et al., 2018). QM9 is a subset of the GDB-17 (Ruddigkeit et al., 2012) database, focusing on quantum chemical property prediction for small organic molecules. It provides comprehensive quantum chemical attributes for molecular compounds, including spatial geometries and electronic properties, such as HOMO/LUMO energy levels obtained via DFT calculations (Kohn & Sham, 1965). In this work, we focus on the HOMO/LUMO energy levels of molecules. For the **Quantum Mechanics Property Prediction** task, following Fang et al. (2024), we split the dataset into 360,113 training instances and 1,987 test instances.

**(4) PubChem** (Kim et al., 2021). PubChem is the world's largest open-access chemical information database, focusing on chemistry, bioinformatics, and drug discovery. It provides comprehensive support for the retrieval and analysis of molecular compound data. Partiallly following Li et al. (2024b), for the **Molecular Weight Prediction**, **LogP Prediction**, **Topological Polar Surface Area Prediction**, and **Complexity Prediction** tasks, we split the dataset into 11,979/2,000, 10,673/1,785, 11,979/2,000, and 11,979/2,000 for training and test sets, respectively. Additionally, for the **Description Q&A** task, also following Li et al. (2024b), we split the dataset into 56,885 training instances and 10,000 test instances.

PubChemQC (Maho, 2015) is a large-scale chemical database generated through ab initio quantum chemistry calculations, with molecular compounds sourced from PubChem. Partially following Li et al. (2024b), for the **SCF Energy Prediction** task, we split the dataset into 623,418 training instances and 77,993 test instances.

**(5) RNX Yields** (Schwaller et al., 2021). The RNX Yields dataset consists of the Buchwald–Hartwig reaction (Ahneman et al., 2018) dataset and the Suzuki–Miyaura reaction (Perera et al., 2018) dataset, both collected through high-throughput experimentation (HTE). It is designed to predict reaction yields for these two reaction types. Following PRESTO, we split the dataset into 9,515 training instances and 200 test instances for **Yields Regression**.

**(6) ORD** (Kearnes et al., 2021). The ORD (Open Reaction Database) is an open-source database dedicated to the standardization, storage, and sharing of organic chemistry reaction data, providing a unified data schema with structured text for organic reaction datasets. Following Liu et al. (2024c) (along with the USPTO-Applications), for **Experimental Procedure Prediction** task, We partition the dataset into 90% for training, 10% for validation, and 10% for testing, based on the total data volume.

Based on the six datasets presented above, we construct a total of 15 tasks spanning four task types, amounting to 1.8 million data samples. To the best of our knowledge, this represents the most comprehensive dataset to date in the molecular domain. The specific partitions are illustrated in Figure 8 below.

### B.2 PRE-PROCESSING

We encounter several issues during processing the datasets, we list them below and elaborate our solutions.

**Unable to obtain SELFIES.** We retrieve the SMILES representation of a molecule with its CID using pubchempy (Swain, 2024) API, for CIDs that cannot be found with `pubchempy.Compound.from_cid()`, we discard them. For molecules that cannot be con-

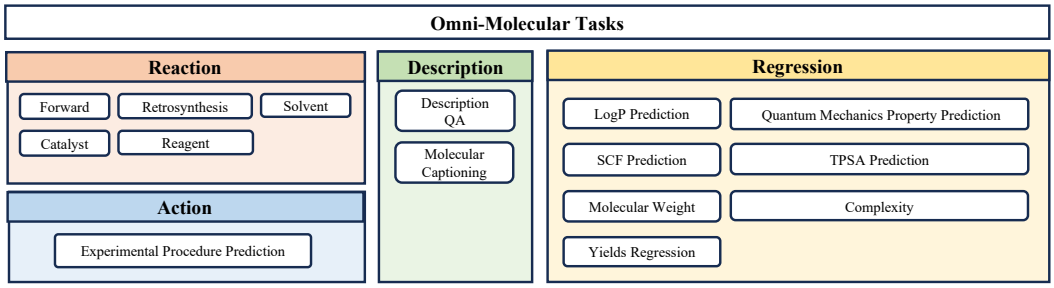



Figure 8: Domains and Tasks. We choose 15 tasks across 4 domains, including reaction, regression, description and action.

verted to SELFIES, we discard them. This could happen when processing datasets from Li et al. (2024b).

**Overlapped samples.** Datasets from different sources often contain overlapping samples, leading to potential data leakage. For example, solvent and catalyst prediction are subsets of reagent prediction, and molecule description data from 3D-MOIT (Li et al., 2024b) may include samples that overlap with those in ChEBI-20 (Edwards et al., 2021). Such overlaps create scenarios where a sample from one dataset's training set appears in the test set of another, compromising the reliability of model evaluation. To address this issue, we conduct a thorough dataset comparison to identify potential overlaps and systematically remove any samples from the training sets that also appear in the test sets of other datasets.

### B.3 Evaluation Metrics

**Exact Match.** The Exact Match Score evaluates whether two SMILES strings unequivocally correspond to the same molecular structure. Specifically, a score of 1 is assigned when both SMILES strings are identical following normalization, indicating they represent the same molecule. Meanwhile, a score of 0 is given when the normalized SMILES strings differ, signifying that they correspond to distinct molecules.

**Levenshtein Score.** The Levenshtein Score scores the smallest number of edit operations needed to transform one SMILES string into another. These edit operations typically encompass: (1) Insertion, which involves adding a character at a specific position; (2) Deletion, the removal of a character from a designated location; and (3) Substitution, replacing a character at a particular position with a different one.

**MACCS Similarity.** Within cheminformatics, MACCS Similarity is used to assess and compare the structural likeness of molecules. This approach is grounded in MACCS keys, which are a standardized set of structural descriptors developed by the Molecular ACCess System. These keys capture and represent essential molecular substructures. To determine the similarity between two molecules, the method evaluates the presence or absence of these predefined structural features.

**RDK Similarity.** The RDK Similarity generally involves evaluating and quantifying the similarity between molecules by utilizing fingerprints produced with RDKit.

**Morgan Similarity.** Morgan Similarity is used to evaluate and measure the structural resemblance between molecules by utilizing Morgan fingerprints as its foundational basis.

**Mean Absolute Error (MAE)**. The MAE quantifies the average absolute deviations between predicted results and actual values, which provides a straightforward metric for assessing the accuracy of predictive models by averaging the absolute differences across all instances.

**$R^2$**. The $R^2$ metric scores the proportion of variability in the target variable that can be explained by the model's predictors. It can be served as an indicator of the model's explanatory strength, reflecting how well the observed data points are captured by the regression model.

# C Related Works

## C.1 Molecular Foundation Models

Researchers are trying to leverage the world knowledge embedded in LLMs to build higher-quality molecular representations by fine-tuning on task-specific instructions. Mol-Instruction (Fang et al., 2024) pioneers the instruction fine-tuning dataset, demonstrating the potential of LLMs in molecular modeling. Subsequently, InstructMol (Cao et al., 2025) introduces 2D graph features of molecules based on SMILES (Weininger, 1988), showing that LLMs can also enhance performance by aligning and fine-tuning their understanding of graph-based features. Soon after, 3D-MoLM (Li et al., 2024b) explores the advantages of 3D molecular representations in multimodal LLMs, while HIGHT (Chen et al., 2024) investigates the impact of multi-level 2D graph features on molecular understanding. More recently, PRESTO (Cao et al., 2024) enhances LLMs' comprehension of molecular-related knowledge through extensive domain-specific pretraining across eight tasks.

## C.2 Unified Generative Modeling

The GPT models (Brown et al., 2020; Achiam et al., 2023) have achieved unification across all text-based tasks through large-scale pretraining and instruction tuning. Subsequently, the community has successfully constructed models that can understand data from multiple modalities and simultaneously perform tasks related to different modalities by converting features from each modality into tokens (Alayrac et al., 2022; Li et al., 2022; 2023; Dai et al., 2023; Liu et al., 2024b). More recently, the community has also been exploring unified understanding and generation, allowing models not only to understand multimodal data but also to generate multimodal data (Zhu et al., 2023; Zheng et al., 2023a; Koh et al., 2024). This development is driving models towards convergence into a truly general-purpose model capable of solving all tasks. Huh et al. (2024) suggests that as models grow more powerful and general, their representations tend to converge, approaching a universal space that reflects the fundamental laws of the world. This insight inspires us to explore whether a universal convergent space also exists in the molecular domain.

# D Further Details on Model Implementation

## D.1 Graph Tokenizer

**Molecule to graph conversion.** Graph neural network is widely used in many scenarios like traffic (Zhang et al., 2025), society relationships (Zhang et al., 2024a), and also molecules (Sun et al., 2022; 2024). Following vanilla setting, we utilize RDKit (Landrum et al., 2013) to transform SELFIES into graph structure in our experiments. For tasks involving a single molecule as input, the molecule is converted directly. For tasks requiring multiple molecules as input, only the first molecule in the input sequence is converted into a graph. Our model does not incorporate multi-graph understanding; instead, it processes both the graph and SELFIES representation of the first molecule, while only the SELFIES representations are provided for the remaining molecules. Meanwhile, since MoleculeSTM (Liu et al., 2023) incorporates additional molecular graph-text contrastive training compared to GraphMVP (Liu et al., 2022), leading to improved multimodal model training efficiency, we adopt MoleculeSTM as the graph encoder.

**Insertion**. For graph tokens $\mathbf{H}_G = \{H_1, H_2, \ldots, H_n\}$ after projection, we always insert the graph token at the beginning of user instruction $\mathbf{X_I}$. The input instruction will be updated to the concatenation of $\{\mathbf{H}_G, \mathbf{X}_I\}$.

## D.2 Multimodal Alignment

To balance the molecular graph and text modalities while ensuring training efficiency, we employ a single-layer linear projector in Stage 1. Following Liu et al. (2023), we carefully filter PubChem to obtain 310K+ graph-text pairs and convert them into instruction-following data for pretraining. The alignment between the molecular graph and text modalities is enhanced solely by adjusting the parameters of the single-layer linear projector.

### D.3 PARAMETER-EFFICIENT FINE-TUNING (PEFT)

As the size of recent models increases rapidly, updating the models in parameter-efficient ways becomes crucial. PEFT (Ding et al., 2023; Wang et al., 2023; Zhai et al., 2023; Yu et al., 2024a; Wang et al., 2024) methods diverge from the conventional approach of fine-tuning the entire pre-trained model, instead only learning a few additional parameters for knowledge transfer. Due to the redundancies of attention matrix in LLMs' pre-trained parameter $\mathcal{W}_0$, we hope to implement the low-rank approximation (LoRA) of the tensor $\mathcal{W}_0$ to get the new learnable weight tensor $\Delta\mathcal{W} \in \mathbb{R}^{m\times n}$ for downstream knowledge transfer. $\Delta\mathcal{W}$ can be constructed with the product of two lower-dimensional matrices $\mathcal{A} \in \mathbb{R}^{m\times r}$ and $\mathcal{B} \in \mathbb{R}^{r\times n}$, where $r \ll \min(m, n)$. The goal is to find $\mathcal{A}$ and $\mathcal{B}$ that minimize the approximation error between $W$ and $\mathcal{AB}$. A common objective is the Frobenius norm, leading to the minimization problem $\min_{\mathcal{A},\mathcal{B}} \|\Delta\mathcal{W} - \mathcal{AB}\|_F^2$. Here, $\mathcal{A}$ and $\mathcal{B}$ together have significantly fewer parameters than $\Delta\mathcal{W}$ itself, making $\mathcal{AB}$ an effective rank-$r$ approximation. Therefore, the backward propagation on the downstream fine-tuning data $\mathcal{D}$ can be expressed as:

$$\nabla_{\boldsymbol{\Delta\mathcal{W}}} = \frac{\partial\mathcal{L}\left(\mathcal{D}; \boldsymbol{\mathcal{W}}_0 + \boldsymbol{\Delta\mathcal{W}}\right)}{\partial\boldsymbol{\Delta\mathcal{W}}} \tag{7}$$

However, in Figure 1 right, we observe that the conventional LoRA approach can encounter intricate divergences in the middle to later phases of training, primarily due to a gradual increase in gradient norm values. Motivated by work Kalajdzievski (2023); Team (2024), we attribute the main cause of these divergences to issues arising from the softmax operation when handling tasks drawn from different domains with substantially varied entropy, stemming from softmax's translation invariance (*i.e.*, $softmax\left(z\right) = softmax\left(z + c\right)$). Since all model parameters are shared among multiple tasks, each task competes by incrementally growing its norms. while this is not immediately detrimental, it leads to divergence once norms extend beyond the effective range of bf16. To mitigate this, we introduce the Adaptive Gradient Stabilization module in Section 2.3, employing an adaptive coefficient $\gamma_\theta = \frac{\alpha_\theta}{||r||_p} + \beta_\theta$, where $\alpha_\theta$ and $\beta_\theta$ are learnable variables and $r$ represents the rank. Through this mechanism, gradients can be adaptively stabilized in the form of:

$$\nabla_{\boldsymbol{\Delta\mathcal{W}}} = \frac{\partial\mathcal{L}\left(\mathcal{D}; \boldsymbol{\mathcal{W}}_0 + \gamma_\theta \cdot \boldsymbol{\Delta\mathcal{W}}\right)}{\partial\boldsymbol{\Delta\mathcal{W}}} \tag{8}$$

### D.4 FROZEN PRE-TRAINED BACKBONE

We adopt LLaMA 3 (Dubey et al., 2024) as the backbone, a standard dense Transformer (Vaswani, 2017) architecture. It employs grouped query attention (Ainslie et al., 2023), which generalizes multi-query attention by introducing an intermediate set of key-value heads. Furthermore, LLaMA 3 applies an attention mask that blocks cross-document self-attention within a single sequence; while this feature shows minimal influence during standard pre-training, it becomes crucial for continued pre-training on long sequences. Lastly, LLaMA 3 supports an expanded vocabulary of 128K tokens.

### D.5 MUTUAL REPRESENTATION SIMILARITY

**Task Scaling Setup.** We build a sequence of multi-task dataset with detailed composition as follows:

- **1 task**: Reagent Prediction.
- **2 tasks**: Reagent Prediction + Molecular Captioning.
- **4 tasks**: Reagent Prediction + Molecular Captioning + Solvent Prediction + Catalyst Prediction.
- **8 tasks**: Reagent Prediction + Molecular Captioning + Solvent Prediction + Catalyst Prediction + Forward Prediction + Retrosynthesis + Property Prediction + Yield Regression.

**Similarity Calculation.** We first extract features $R \in \mathbb{R}^{B\times L\times T\times d}$ from all decoder layers in LLM, where $B, L, T, d$ is batch size, number of decoder layers, sequence length and the hidden dimension of LLM. The sequence dimension is then averaged.

$$R' = \frac{\left(\sum_{t=1}^{T}(R[:,:,t,:] * m[:,t])\right)}{\sum_{t=1}^{T} m[:,t]} \tag{9}$$

where $R' \in \mathbb{R}^{B\times L\times d}$, and $m \in \mathbb{R}^{B\times T}$ is the mask indicating the padding tokens. We then flatten the first two dimensions and get $R'' \in \mathbb{R}^{(B*L)\times d}$ and calculate the similarity with `mutual_knn` (Huh et al., 2024).

| | Learning rate | Num Epoch | LR Decay | Stop Epoch | Batch Size | Warmup Ratio |
|---|---|---|---|---|---|---|
| Forward Reaction Prediction<br>Reagent Prediction<br>Retrosynthesis<br>Quantum Mechanics Property Prediction<br>Catalyst Prediction<br>Solvent Prediction<br>Yield Regression<br>Experimental Procedure Prediction<br>Description Q&A<br>SCF Energy Prediction<br>Topological Polar Surface Area Prediction<br>Complexity Prediction<br>Molecular Weight Prediction<br>LogP Prediction | 8e-5 | 15 | cosine | 10 | 32 | 0.0075 |
| Molecular Captioning | | 10 | | 8 | | |
| Omni-Molecular Tasks | | 15 | | 15 | | |

Table 2: An overview of the hyper-parameters and training configurations used in all molecular task experiments.

Let $N = B * L$, and we have two models $A$ and $B$ trained on different multi-task datasets, we first find their $k$ nearest neighbors $\text{knn}^A$ and $\text{knn}^B$.

$$\text{knn}^A = \text{KNN}(R^A, k) \quad \text{knn}^B = \text{KNN}(R^B, k) \tag{10}$$

where $\text{knn}^* \in \mathbb{R}^{N \times k}$, we then create indicator matrices

$$M^A_{i,j} = \begin{cases} 1, & j \in \text{knn}^A[i,:] \\ 0, & \text{otherwise} \end{cases} \qquad M^B_{i,j} = \begin{cases} 1, & j \in \text{knn}^B[i,:] \\ 0, & \text{otherwise} \end{cases} \qquad i, j \in 1, \ldots, N \tag{11}$$

The accuracy of a sample is

$$\text{acc}[i] = \frac{1}{k} \left| \text{knn}^A[i,:] \cap \text{knn}^B[i,:] \right| = \frac{1}{k} \sum_{j=1}^{N} M^A_{i,j} \cdot M^B_{i,j} \tag{12}$$

Finally, the alignment score of two models is

$$\text{Score} = \frac{1}{N} \sum_{i=1}^{N} \text{acc}[i] \tag{13}$$

## E Further Details on Training

Specialist models are typically fine-tuned on a single task at a time, repeating the process separately for each task, a strategy known as separate tuning. In contrast, generalist models undergo simultaneous fine-tuning across multiple tasks, a process referred to as unified tuning. In this section, we present a detailed training framework for both of them on all experiments.

**Separate instruction tuning.** We follow the training recipe outlined in Cao et al. (2025). However, we observe significant overfitting when training the model on the molcap task for 20–50 epochs, as suggested in Cao et al. (2025). To address this issue, we manually allocate 10% of the training set for validation and re-evaluated all tasks, we find that the recipes for forward prediction, reagent prediction, retrosynthesis, HOMO LUMO prediction from the original paper matches our results, however, we identify an updated training strategy tailored to the molcap task. The revised training recipe is summarized in Table E.

**Unified instruction tuning.** For unified training, we apply a fixed training recipe as shown in Table E, this recipe is consistent across all Unified Instruction Tuning.

**Environment Settings.** We employ common techniques to boost performance and conserve memory, including FlashAttention 2 (Dao et al., 2022), activation checkpointing, and bf16 training. All

| Model | Type | #Param | B-2 | B-4 | R-1 | R-2 | R-L | M |
|---|---|---|---|---|---|---|---|---|
| Molecular Captioning Task | | | | | | | | |
| GPT-3.5-turbo[1] | Retrieval | - | 0.565 | 0.482 | 0.623 | 0.450 | 0.543 | 0.585 |
| GPT-4-0314[1] | Retrieval | - | 0.607 | 0.525 | 0.634 | 0.476 | 0.562 | 0.610 |
| BioMedGPT[2] | Generalist | 10B | 0.234 | 0.141 | 0.386 | 0.206 | 0.332 | 0.308 |
| Mol-Instruction | Specialist | 6.7B | 0.249 | 0.171 | 0.331 | 0.203 | 0.289 | 0.271 |
| HIGHT | Specialist | 6.7B | 0.498 | 0.397 | 0.582 | 0.414 | 0.518 | 0.525 |
| InstructMol | Specialist | 6.7B | 0.475 | 0.371 | 0.566 | 0.394 | 0.502 | 0.509 |
| **Omni-Mol** | Generalist | **1.7B** | **0.544** | **0.456** | **0.610** | **0.456** | **0.549** | **0.579** |
| Description Q&A Task | | | | | | | | |
| Llama2[3] | Specialist | 6.7B | 0.282 | 0.232 | 0.351 | 0.221 | 0.304 | 0.469 |
| 3D-MoLM(S)[4] | Specialist | 6.7B | 0.320 | 0.261 | 0.401 | 0.256 | 0.346 | 0.522 |
| 3D-MoLM(G)[4] | Generalist | 6.7B | 0.318 | 0.261 | 0.401 | 0.259 | 0.350 | 0.519 |
| **Omni-Mol** | Generalist | **1.7B** | **0.516** | **0.440** | **0.529** | **0.382** | **0.492** | **0.580** |

| Model | Type | #Param | HOMO | LUMO | GAP | Avg. |
|---|---|---|---|---|---|---|
| Quantum Mechanics Property Prediction Task | | | | | | |
| Alpaca[1] | In-Context | 6.7B | - | - | - | 322.109 |
| LLaMA2[2] | In-Context | 6.7B | 0.7367 | 0.8641 | 0.5152 | 0.7510 |
| Vicuna[2] | In-Context | 13B | 0.7135 | 3.6807 | 1.5407 | 1.9783 |
| Mol-Instruction | Specialist | 6.7B | 0.0210 | 0.0210 | 0.0203 | 0.0210 |
| HIGHT | Specialist | 6.7B | 0.0056 | 0.0065 | 0.0077 | 0.0066 |
| InstructMol | Specialist | 6.7B | 0.0048 | **0.0050** | 0.0061 | **0.0050** |
| **Omni-Mol** | Generalist | **1.7B** | **0.0047** | 0.0056 | **0.0060** | 0.0052 |

Table 3: Main results of molecular captioning and description QA task, [1]: 10-shot results from Li et al. (2024a), [2,3]: results from Luo et al. (2023); Li et al. (2024b), [4]: (S,G) means the specialist and generalist version of 3D-MoLM separately. B: BLEU, R: ROUGE, M: METEOR.

experiments are conducted on 8×NVIDIA A100 GPUs (80GB). For all experiments, the weight decay is set to 0. The term *Stop Epoch* in Table E shows the epoch that the experiment stops. This is because the early stop mechanism we used to prevent overfitting.

# F Further Details on Experimental Results

In this section, we provide a complete performance evaluation of Omni-Mol across the remaining 11 tasks.

| Model | Type | #Param | Weight(g/mol) | LogP | TPSA(Å$^2$) | Complexity | SCF($10^5$eV) |
|---|---|---|---|---|---|---|---|
| Llama2 | Specialist | 6.7B | 22.10 (96%) | 1.45 (95%) | 15.87 (92%) | 69.74 (93%) | 0.70 (99%) |
| 3D-MoLM(S) | Specialist | 6.7B | 14.79 (95%) | 0.66 (97%) | 9.71 (93%) | 44.85 (94%) | 0.35 (99%) |
| 3D-MoLM(G) | Generalist | 6.7B | 16.58 (92%) | 0.78 (95%) | 10.90 (90%) | 45.49 (89%) | 0.38 (98%) |
| **Omni-Mol** | Generalist | **1.7B** | **15.08 (100%)** | **0.59 (100%)** | 11.17 (**100%**) | 49.38 (**100%**) | 0.55 (99%) |

| Model | Type | #Param | B-H | S-M |
|---|---|---|---|---|
| Llama2 | - | 6.7B | -0.476 | 0.121 |
| Vicuna v1.5 | - | 6.7B | -0.131 | 0.151 |
| PRESTO | Generalist | 6.7B | 0.944 | 0.652 |
| **Omni-Mol** | Generalist | **1.7B** | 0.891 | 0.560 |

Table 4: More results for regression tasks. (Left) Results of property regression tasks, we report MAE for each task with the valid answer rate (%), since LMs sometimes fail to generate numerical responses corresponding to the given prompts. (Right) Results of Yield regression task, we report $R^2$ score.

# G More Ablation Study Results

## G.1 Separate Tuning V.S. Unified Tuning

Here, we present the performance of separate tuning and unified tuning across the remaining 9 tasks.

As show in Figure 9, We observe that tasks like solvent prediction, catalyst prediction, and experiment procedure prediction continue to gain significant improvements from unified training. How-

| Model | Type | #Param | Exact | BLEU | Levenshtein | RDK | MACCS | Morgan | Validity |
|---|---|---|---|---|---|---|---|---|---|
| Catalyst Prediction (PRESTO) | | | | | | | | | |
| Llama2 | - | 6.7B | 0.680 | 0.720 | 2.545 | 0.882 | 0.868 | 0.687 | 1.000 |
| Vicuna v1.5 | - | 6.7B | 0.685 | 0.703 | 2.451 | 0.883 | 0.869 | 0.692 | 1.000 |
| nach0-base | - | - | 0.000 | 0.072 | 36.442 | 0.129 | 0.055 | 0.009 | 0.849 |
| Mol-Instruction | Specialist | 6.7B | 0.000 | 0.110 | 28.424 | 0.031 | 0.045 | 0.015 | 0.999 |
| T5Chem | - | - | 0.022 | 0.346 | 13.408 | 0.146 | 0.268 | 0.200 | 0.996 |
| PRESTO | Generalist | 6.7B | **0.768** | 0.814 | 1.755 | 0.914 | 0.895 | 0.774 | 1.000 |
| **Omni-Mol** | Generalist | **1.7B** | 0.752 | **0.860** | **1.544** | **0.919** | **0.903** | **0.759** | **1.000** |
| Solvent Prediction (PRESTO) | | | | | | | | | |
| Llama2 | - | 6.7B | 0.311 | 0.462 | 3.819 | 0.452 | 0.480 | 0.417 | 1.000 |
| Vicuna v1.5 | - | 6.7B | 0.320 | 0.436 | 3.809 | 0.459 | 0.486 | 0.427 | 1.000 |
| nach0-base | - | - | 0.000 | 0.072 | 36.442 | 0.129 | 0.055 | 0.009 | 0.849 |
| Mol-Instruction | Specialist | 6.7B | 0.000 | 0.155 | 25.117 | 0.030 | 0.122 | 0.035 | 1.000 |
| T5Chem | - | - | 0.083 | 0.311 | 16.224 | 0.458 | 0.424 | 0.397 | 0.995 |
| PRESTO | Generalist | 6.7B | 0.419 | 0.695 | 2.758 | 0.529 | 0.547 | 0.506 | 0.912 |
| **Omni-Mol** | Generalist | **1.7B** | **0.590** | **0.799** | **2.243** | **0.740** | **0.733** | **0.715** | **1.000** |

Table 5: More results for reaction tasks.

| Model | Type | #Param | BLEU-2 | BLEU-4 | ROUGE-1 | ROUGE-2 | ROUGE-L |
|---|---|---|---|---|---|---|---|
| TextChemT5 | Generalist | 220M | 0.541 | 0.406 | 0.615 | 0.403 | 0.564 |
| MolT5-Large | Specialist | 780M | 0.545 | 0.410 | 0.625 | 0.409 | 0.572 |
| Galactica | Specialist | 1.3B | 0.535 | 0.395 | 0.609 | 0.386 | 0.552 |
| MolCA, Galac | Specialist | 1.3B | 0.549 | 0.415 | 0.625 | 0.404 | 0.570 |
| ReactXT, Galac | Specialist | 1.3B | 0.574 | 0.440 | 0.644 | 0.427 | 0.589 |
| **Omni-Mol** | Generalist | 1.7B | 0.569 | **0.445** | 0.523 | 0.270 | 0.460 |

Table 6: Results of experimental procedure prediction.

ever, we also notice that some tasks perform worse after unified training compared to individual training, particularly those related to regression tasks, such as yield regression, logP prediction, and complexity prediction. While we see some modest gains in SCF prediction and weight prediction, there is a noticeable performance degradation in other tasks, especially in the complexity prediction task, where unified training causes a substantial drop in performance.

Regression tasks differ significantly from SELFIES generation and text description tasks in that the model must, after understanding the molecule, directly and accurately predict the specific numerical value of a property. Even though existing literature suggests that LLMs are capable of performing regression, accurately predicting the next number remains a fundamentally different challenge from current tasks, and potentially introduces significant conflicts.

Perhaps incorporating additional information, such as the process of deriving a specific number (*e.g.*, a chain of thought), or using an additional regression head, could help alleviate these conflicts before making an accurate numerical prediction.

## H Further Details on Efficiency Analysis

To ensure a fair comparison of computational efficiency with existing methods, we evaluate Omni-Mol across four key metrics: inference time, memory consumption, token generation speed (tokens per second), and GFLOPs. All comparative experiments are conducted on a single standalone NVIDIA A100-80G GPU to ensure consistency in inference evaluation. As illustrated in Figure 10, Omni-Mol achieves significantly lower inference latency, reducing it by up to 65% compared to PRESTO. This substantial efficiency gain enables faster response times, making Omni-Mol particularly well-suited for high-throughput molecular modeling tasks. Moreover, Omni-Mol demonstrates optimized memory utilization, consuming 28% less memory, which enhances its feasibility for deployment on resource-constrained hardware without compromising performance. Additionally, Omni-Mol achieves a 1.78$\times$ higher token generation rate, ensuring faster sequence generation and significantly improving usability in real-world applications that require rapid molecular property predictions and synthesis planning. Finally, Omni-Mol effectively reduces computational cost by 41% in terms of GFLOPs, striking a favorable balance between model complexity and inference

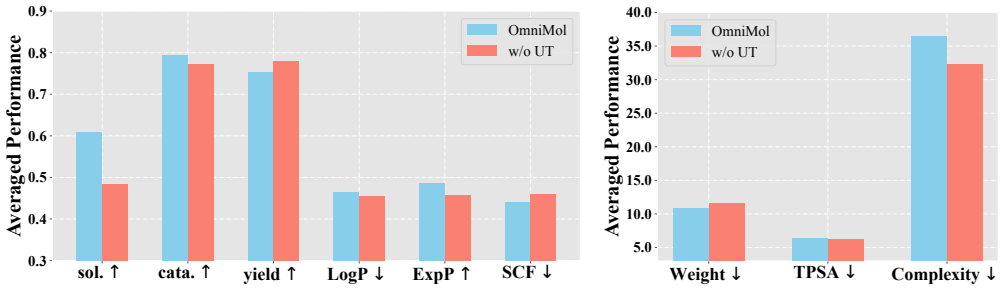



Figure 9: Additional results from the ablation study comparing our unified instruction tuning to separate training.

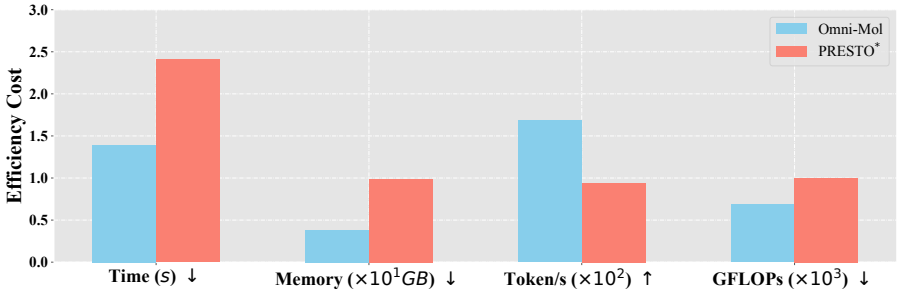



Figure 10: A comprehensive efficiency evaluation of our Omni-Mol, built on LLaMA 3.2-1B, compared to the state-of-the-art baseline. We conduct all experiments three times and compute the average result.

speed. These advantages make Omni-Mol a scalable, cost-efficient, and high-performance solution for large-scale molecular modeling and chemistry-related AI applications.

# I Further Details on Theoretical Analysis

## I.1 Proof of Theorem 2.1

*Proof of Theorem 2.1.* For $F_{\text{general}}^{(n)}$ and $F_{\text{general}}^{(m)}$ with $m > n$, we have

$$\begin{aligned} F_{\text{general}}^{(m)} &= F_1 \cap F_2 \cap \cdots \cap F_n \cap F_{n+1} \cap \cdots \cap F_m \\ &= F_{\text{general}}^{(n)} \cap F_{n+1} \cap \cdots \cap F_m \end{aligned} \tag{14}$$

Since intersection of sets can only become smaller or remain the same as more sets are intersected, therefore

$$F_{\text{general}}^{(m)} \subseteq F_{\text{general}}^{(n)} \tag{15}$$

If $\forall i, j, n < i, j < m$ and $F_i \neq F_j$ when $i \neq j$, then

$$\begin{aligned} &F_{\text{general}}^{(n+1)} = F_{\text{general}}^{(n)} \cap F_{n+1} \subsetneq F_{\text{general}}^{(n)} \\ &F_{\text{general}}^{(n+2)} = F_{\text{general}}^{(n+1)} \cap F_{n+2} \subsetneq F_{\text{general}}^{(n+1)} \subsetneq F_{\text{general}}^{(n)} \\ &\cdots \\ &F_{\text{general}}^{(m)} = F_{\text{general}}^{(n+(m-n-1))} \cap F_{n+(m-n)} \subsetneq F_{\text{general}}^{(n+(m-n-1))} \subsetneq F_{\text{general}}^{(n+(m-n-2))} \subsetneq \cdots \subsetneq F_{\text{general}}^{(n)} \end{aligned} \tag{16}$$

□

# J Case Study

In this section, we visualize specific reactions in reaction tasks. The results in Figure 11 demonstrate that our method exhibits more accurate generation capabilities compared to the baseline.

### J.1 REACTION TASKS

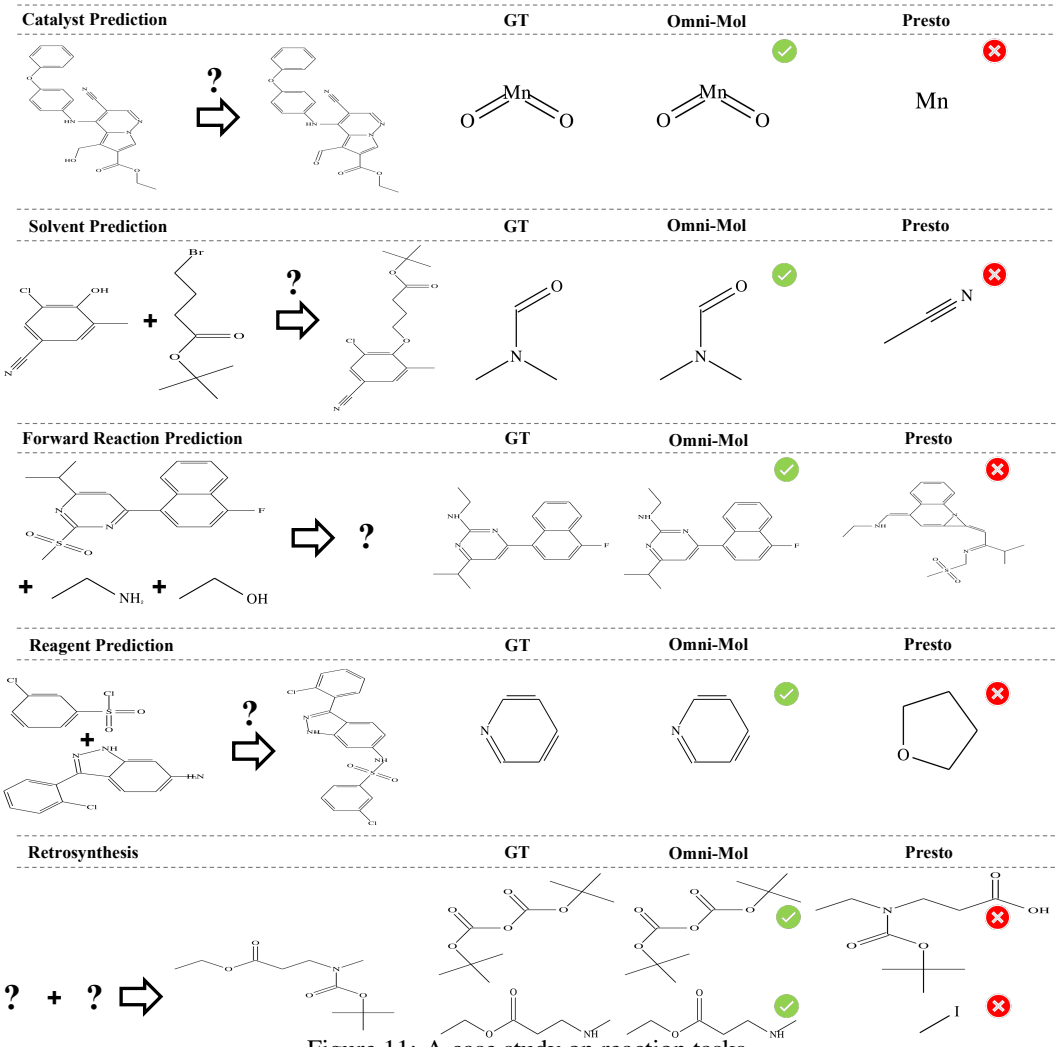



Figure 11: A case study on reaction tasks.

In this subsection, we visualize specific reactions in three reaction tasks. The results in Figure 11 demonstrate that our method exhibits more accurate generation capabilities compared to the baseline. For example, in the solvent prediction task, we are given the reactants: [C][C][Branch1][C][C][Branch1][C][C][O][C][=Branch1][C][=O][C][C][C][Br] and [C][C][=C][C][Branch1][Ring1][C][#N][=C][C][Branch1][C][Cl][=C][Ring1][=Branch2][O], as well as the product [C][C][=C][C][Branch1][Ring1][C][#N][=C][C][Branch1][C][Cl][=C][Ring1]-[=Branch2][O][C][C][C][C][=Branch1][C][=O][O][C][Branch1][C][C][Branch1][C][C][C]. Omni-Mol correctly predicts the solvent as [C][N][Branch1][C][C][C][=O], whereas PRESTO predicts an incorrect solvent: [C][C][#N].

### J.2 MOLCAP

In the case study of the molecular captioning task, as shown in Figure 12, the model's description of the same molecule becomes more accurate before and after mixed training. It is able to correctly classify and localize functional groups. Does this suggest that the model can learn to identify functional groups from the reaction task? Additionally, constraints from other tasks in the shared representation space also enhance the model's ability to describe molecules. For example, for Case 1 Molecule, Separate Tuning outputs incorrect information regarding the locations of functional groups, whereas Unified Tuning predicts them correctly.

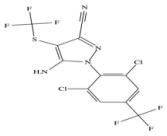
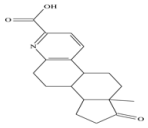
| Case 1 Molecule | GT | Separate Tuning | Unified Tuning |
|---|---|---|---|
|  | The molecule is a member of the class of pyrazoles that is 1H-pyrazole that is substituted at positions 1, 3, 4, and 5 by 2,6-dichloro-4-(trifluoromethyl)phenyl, cyano, (trifluoromethyl)sulfanyl, and amino groups, respectively. It is a metabolite of the agrochemical fipronil. It has a role as a marine xenobiotic metabolite. It is a member of pyrazoles, a dichlorobenzene, a member of (trifluoromethyl)benzenes, an organic sulfide and a nitrile. | The molecule is a member of the class of pyrazoles that is 1H-pyrazole which is substituted at positions 1, 3, and 5 by 2,4-dichloro-5-chlorophenyl, cyano, and trifluoromethylsufanyl groups, respectively. It is a member of pyrazoles, a nitrile, a dichlorobenzene, a member of (trifluoromethyl)benzenes and a member of monochlorobenzenes. | The molecule is a member of the class of pyrazoles that is 1H-pyrazole that is substituted at positions 1, 3, 4, and 5 by 2,6-dichloro-4-(trifluoromethyl)phenyl, cyano, trifluoromethyl, and amino groups, respectively. It is a nitrile, a dichlorobenzene, a secondary amino compound, an aromatic primary alcohol, a member of pyrazoles and a member of (trifluoromethyl)benzenes. |
| **Case 2 Molecule** | **GT** | **Separate Tuning** | **Unified Tuning** |
|  | The molecule is a steroid acid anion, that is the conjugate base of pyridinestrone-3-carboxylic acid. An abiotic metabolite in the 4,5-seco pathway of aerobic estrogen degradation by the bacterium Sphingomonas sp. strain KC8. It is a conjugate base of a pyridinestrone-3-carboxylic acid. | The molecule is a monocarboxylic acid anion that is the conjugate base of 17-oxoestradiol, obtained by deprotonation of the carboxy group; major species at pH 7.3. It is a conjugate base of a 17-oxoestradiol. | The molecule is a steroid acid anion that is the conjugate base of 9-oxo-15alpha-17-epysterone, obtained by deprotonation of the carboxy group; major species at pH 7.3. It is a conjugate base of a 9-oxo-15alpha-17-epysterone. |

Figure 12: A case study on molcap task

# K TASK DEFINITION & PROMPT TEMPLATES

## K.1 BASE CHAT TEMPLATE

For LLaMA 3.2 and LLaMA 3.1 instruction-tuned LLMs, we use the base chat template suggested by the official documents, the multi-modal graph tokens are inserted at the beginning of user instructions.

**System Prompt**

<|begin_of_text|><|start_header_id|>system<|end_header_id|> \n\n A chat between a curious user and an artificial intelligence assistant. The assistant gives helpful, detailed, and polite answers to the user's questions.<|eot_id|>

**User Input**

<|start_header_id|>user<|end_header_id|>\n\n<graph_token>\nInstructions.<|eot_id|><|start_header_id|>assistant<|end_header_id|> \ n \n

**Assistant Output**

Response.<|eot_id|>

We use <|finetune_right_pad_id|> as pad token for SFT.

## K.2 FORWARD REACTION PREDICTION

The forward reaction prediction task focuses on determining the chemical product of a reaction given its reactants and reagents. The forward reaction prediction task involves predicting the chemical product of a reaction given the reactants and reagents as input. The input format is structured as the SELFIES representation of reactants, concatenated with a period (".") and the reagent information (*e.g.*, "reactant1.reactant2.reagent"). The task requires the model to process this input and output the corresponding reaction product. The objective is to accurately map the input reaction components to their chemical outcome, leveraging the model's understanding of reaction patterns and transformations. A key challenge in forward reaction prediction is capturing the underlying chemical rules that govern reactivity. The model must infer how functional groups interact, recognize the role of reagents, and apply appropriate transformations to generate the correct product. This process requires a deep understanding of reaction mechanisms, beyond simple pattern recognition. The prompt template is as follows.

**Template**

User Input: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_reactants>.<SELFIES_reagents> ⊕ <|eot_id|> ⊕ <assistant_identifier>
Assistant Output: <SELFIES_product>.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Given the reactants and reagents provided, what is a possible product that can be formed?
**<SELFIES_reagents>**: [Br][C][C][Br].[O][C][=C][C][Branch1][C][Br][=C][C][=C][Ring1][#Branch1][Br]
**<SELFIES_reactants>**: [Na+1].[OH1-1]

**<SELFIES_product>**: [Br][C][C][O][C][=C][C][Branch1][C][Br][=C][C][=C][Ring1][#Branch1][Br]

### K.3 RETROSYNTHESIS

The retrosynthesis task focuses on predicting the reactants required to synthesize a given chemical product, a fundamental challenge in organic chemistry and computational drug discovery. Unlike forward reaction prediction, which maps reactants to products, retrosynthesis operates in reverse, it seeks to determine the most plausible set of precursors that could yield the target compound under appropriate reaction conditions. This task is crucial for designing efficient synthetic routes, enabling chemists to explore viable pathways for molecule construction while minimizing cost and complexity. At the core of this task is a structured input format using SELFIES representations, ensuring a robust and unambiguous encoding of molecular structures. The input consists of the SELFIES representation of the target product, which the model then processes to generate the corresponding reactants. This structured formulation ensures that the model can generalize across diverse chemical transformations, learning the intricate patterns of bond formation and cleavage. A key challenge in retrosynthesis prediction is handling the inherent one-to-many nature of the problem: a single product can often be synthesized through multiple distinct reaction pathways. The model must effectively navigate this complexity, identifying the most chemically plausible set of reactants based on learned reaction mechanisms. The prompt template is as follows.

**Template**

User Input: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_product> ⊕ <|eot_id|> ⊕ <assistant_identifier>
Assistant Output: <SELFIES_reactants><|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Which reactants could have been used to generate the given product? The product is:
**<SELFIES_product>**: [C][C][=Branch1][C][=O][C][=C][C][=C][Branch1][C][O][C][Branch1][C][Cl][=C][Ring1][Branch2]

**<SELFIES_reactants>**: [C][C][=Branch1][C][=O][Cl].[O][C][=C][C][=C][C][=C][Ring1][=Branch1][Cl]

### K.4 REAGENT PREDICTION

The reagent prediction task focuses on identifying the necessary reagents for a given chemical reaction, a critical step in reaction planning and synthetic chemistry. This task is essential for guiding experimental chemists, as choosing the correct reagents influences reaction efficiency, selectivity, and feasibility. To ensure a structured and standardized input format, we represent the reaction equation using SELFIES, a robust molecular encoding system. The input consists of the SELFIES representations of the reactants, concatenated with a reaction separator ">>", followed by the SELFIES representation of the product. This format (*e.g.*, "reactant1.reactant2>>product") provides a clear,

machine-readable structure that allows the model to infer the necessary reagents based on known reaction mechanisms and transformation rules. One of the core challenges in reagent prediction is handling the diversity of chemical transformations. Different reactions require specific reagents that dictate the reaction type, whether it's an oxidation, reduction, coupling, or substitution reaction. The model must learn to recognize reaction context, interpret functional group interactions, and infer the most likely reagents from training data. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_reactants> >> <SELFIES_product> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: <SELFIES_reagents><|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Can you provide potential reagents for the following chemical reaction? The reaction is
**<SELFIES_reactants>**: [C][C][C][Branch1][#C][C][=C][C][=N][C][Branch1][Ring1][O][C][=C][Ring1][Branch2]-[C][=O][O][C][C][C][O][Ring1][S]
**<SELFIES_product>**: [C][C][C][Branch1][#C][C][=C][C][=N][C][Branch1][Ring1][O][C][=C][Ring1][Branch2][C][O]-[O][C][C][C][O][Ring1][S]

**<SELFIES_reagents>**: [C][C][Branch1][C][C][O].[O].[BH4-1].[Na+1]

### K.5 Molecular Captioning

The molecular captioning (MolCap) task focuses on generating descriptive textual information for a given chemical compound based on its molecular structure. This task plays a crucial role in chemical informatics, enabling automated annotation of molecular properties, classification, and functional characteristics. MolCap leverages machine learning models to infer and generate human-readable descriptions that encapsulate key chemical attributes. The input for this task follows a structured format using SELFIES, a robust molecular representation designed for machine learning applications. The SELFIES encoding of a given compound serves as the input, and the model is responsible for producing a descriptive caption that includes relevant chemical properties. These descriptions can encompass a wide range of molecular characteristics, such as compound classification (*e.g.*, "organic acid," "amine-containing molecule"), pH estimation, presence of functional groups (*e.g.*, "contains a hydroxyl and ketone group"), solubility, toxicity, or other key features. One of the key challenges in molecular captioning is ensuring that the generated text is both chemically accurate and contextually informative. The model must learn to recognize molecular substructures, infer meaningful chemical attributes, and articulate these in a clear and interpretable manner. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token> \n ⊕ Instruction. ⊕ <SELFIES_compound> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Description.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Please give me some details about this molecule. The compound SELFIES sequence is:
**<SELFIES_compound>**:[C][=C][C][=Branch2][Ring1][S][=C][C][=C][Ring1][=Branch1][C][C][Branch2][Ring1]-[Branch1][C][Branch1][P][C] [Branch1][Ring2][O][Ring1][Branch1][C][O][P][=Branch1][C][=O][Branch1][C][O-1][O-1][O][O][O]

**Description**: The molecule is an organophosphate oxoanion obtained by deprotonation of the phosphate OH groups of 4-(5-O-phospho-beta-D-ribofuranosyl)phenol; major species at pH 7.3. It derives from a D-ribofuranose 5-phosphate(2-). It is a conjugate base of a 4-(5-O-phospho-beta-D-ribofuranosyl)phenol.

### K.6 QUANTUM MECHANICS PROPERTY PREDICTION

The quantum mechanics property prediction task focuses on determining key quantum-mechanical properties of a given chemical compound, providing critical insights into its electronic behavior, stability, and potential applications. This task is essential in computational chemistry, materials science, and drug discovery, where quantum properties influence molecular interactions, reactivity, and optoelectronic performance. The input follows a structured format using SELFIES, a robust molecular representation optimized for machine learning applications. Given the SELFIES encoding of a molecule, the model is tasked with predicting its quantum properties, such as the highest occupied molecular orbital (HOMO) energy, lowest unoccupied molecular orbital (LUMO) energy, and the HOMO–LUMO gap. These properties are fundamental in determining a molecule's electronic structure, with implications for charge transfer, chemical reactivity, and photophysical behavior. One of the key challenges in quantum property prediction is capturing the underlying quantum-chemical interactions that govern molecular behavior. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_compound> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Property.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: What is the HOMO-LUMO gap of this molecule? The compound SELFIES sequence is:
**<SELFIES_compound>**: [N][=C][O][C][=C][C][=Branch1][Ring2][=N][Ring1][=Branch1][C][#N]

**Property**: 0.1487

### K.7 CATALYST PREDICTION

The catalyst prediction task focuses on identifying the appropriate catalysts required to facilitate a given chemical reaction. Catalysts play a crucial role in modifying reaction pathways, lowering activation energy, and improving reaction efficiency without being consumed in the process. The input follows the SELFIES representation, a robust molecular encoding system designed for computational applications. The reaction is expressed as an equation where the SELFIES representations of the reactants are concatenated and separated from the product using ">>" (*e.g.*, "reactant1.reactant2>>product"). This structured representation allows the model to process the reaction as a whole and infer the most suitable catalyst that enables the transformation. One of the primary challenges in catalyst prediction is understanding the nuanced role that catalysts play in different reaction mechanisms. Unlike reagents, which directly participate in the reaction, catalysts provide alternative pathways to enhance reaction kinetics. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_reactants> >> <SELFIES_product> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: <SELFIES_catalysts><|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Given this chemical reaction, what are some catalysts that could have been used? The reaction is
**<SELFIES_reactants>**: [C][C][=C][C][=C][Branch1][Ring1][C][#N][C][=C][Ring1][Branch2][C][Branch1][C][F]-[Branch1][C][F][F].[O][=C][C][C][C][=Branch1][C][=O][N][Ring1][=Branch1][Br]
**<SELFIES_product>**: [N][#C][C][=C][C][=C][Branch1][Ring1][C][Br][C][Branch1][=Branch2][C][Branch1][C][F]-[Branch1][C][F][F][=C][Ring1][N]

**<SELFIES_catalysts>**:[O][=C][Branch1][#C][O][O][C][=Branch1][C][=O][C][=C][C][=C][C][=C]-[Ring1][=Branch1][C][=C][C][=C][C] [=C][Ring1][=Branch1]

### K.8 Solvent Prediction

The solvent prediction task focuses on identifying the appropriate solvents required for a given chemical reaction. Solvents play a crucial role in determining reaction efficiency, influencing factors such as solubility, reaction kinetics, selectivity, and stability of intermediates. To ensure a structured and machine-readable representation, the input follows the SELFIES format, a robust molecular encoding system designed for computational applications. The reaction is expressed as an equation where the SELFIES representations of the reactants are concatenated and separated from the product using the reaction separator ">>" (*e.g.*, "reactant1.reactant2>>product"). This structured format allows the model to interpret the reaction context and infer the most suitable solvents required to facilitate the transformation. One of the key challenges in solvent prediction is understanding the diverse roles solvents play in different reaction mechanisms. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_reactants> >> <SELFIES_product> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: <SELFIES_solvents><|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Please propose potential solvents that might have been utilized in the provided chemical reaction. The reaction is
**<SELFIES_reactants>**: [N][#C][C][=C][C][=C][Branch1][C][F][C][=C][Ring1][#Branch1].[O][C][C][C][N][C][Ring1]-[Branch1]
**<SELFIES_product>**: [N][#C][C][=C][C][=C][Branch1][O][N][C][C][C][Branch1][C][O][C][Ring1][=Branch1][C][=C]-[Ring1][N]

**<SELFIES_solvents>**: [O]

### K.9 Yield Regression

The yield regression task focuses on estimating the proportion of the actual product obtained in a chemical reaction relative to its theoretical maximum. Reaction yield is a critical metric in organic synthesis, pharmaceutical manufacturing, and industrial chemistry, as it directly influences process efficiency, resource utilization, and cost-effectiveness. The input follows the SELFIES format, a robust molecular encoding system tailored for computational chemistry. The reaction is expressed as an equation where the SELFIES representations of the reactants are concatenated and separated from the product using the reaction separator ">>" (*e.g.*, "reactant1.reactant2>>product"). This structured format provides a standardized input for the model, allowing it to interpret the reaction

context and estimate the expected yield. One of the key challenges in yield prediction is capturing the complex interplay between reaction conditions, molecular stability, steric effects, and solvent or catalyst influences. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_reactants> >> <SELFIES_product> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Property.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Please propose potential solvents that might have been utilized in the provided chemical reaction. The reaction is
**<SELFIES_reactants>**: [F][C][Branch1][C][F][Branch1][C][F][C][=C][C][=C][Branch1][C][Cl][C][=C][Ring1][#Branch1].[C][C][=C][C][=C][Branch1] [C][N][C][=C][Ring1][#Branch1].[O][=S][=Branch1][C][=O][Branch2][Ring1][=Branch1][O][Pd][N][C][=C][C][=C][C][=C][Ring1][=Branch1][C][=C][C][=C][C][=C][Ring1][=Branch1][Ring1][=C][C][Branch1][C][F][Branch1][C][F][F].[C][O][C][=C][C][=C][Branch1][Ring1][O][C][C][Branch2][Ring2][=N][P][Branch2][Ring1][Branch1][C][C][C][C][C][Branch1][O][C][C][Branch1][Ring2][C][Ring1][=Branch1][C][Ring1][=Branch2][C][Ring1][#Branch2][C][C][C][C][C][Branch1][O][C][C][Branch1][Ring2][C][Ring1][=Branch1][C][Ring1][=Branch2][C][Ring1][#Branch2][=C][Ring2][Ring1][=N][C][=C][Branch1][=Branch1][C][Branch1][C][C][C][C][=C][Branch1][=Branch1][C][Branch1][C][C][C][C][=C][Ring1][N][C][Branch1][C][C][C].[C][N][C][C][C][N][C][C][C][N][=C][Ring1][#Branch2][Ring1][=Branch1].[C][C][O][C][=Branch1][C][=O][C][C] [=C][Branch1][C][C][O][N][=Ring1][=Branch1]
**<SELFIES_product>**: [C][C][=C][C][=C][Branch2][Ring1][Ring2][N][C][=C][C][=C][Branch1][=Branch2][C][Branch1][C][F][Branch1][C][F][F][C] [=C][Ring1][#Branch2][C][=C][Ring1][P]

**Property**: 0.1449

### K.10 SCF Energy Prediction

The SCF energy prediction task involves determining the self-consistent field (SCF) energy for a given compound. The input is the SELFIES representation of the compound, and the model is tasked with predicting the molecule's SCF energy. The objective is to provide a reliable basis for understanding the compound's total electronic energy, which is critical for assessing its stability, reactivity, and potential applications in computational chemistry. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_compound> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Property.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Please provide the SCF Energy value for this molecule. If uncertain, provide an estimate. Respond with the numerical value only. The molecule SELFIES sequence is:
**<SELFIES_compound>**:[C][C][C][C][C][C][=Branch1][C][=O][C][=C][C][C][Branch2][Ring1][Ring2][C][C][=Branch1][C][=O][C][Ring1] [=Branch1][C][C][C][C][C][C][C][=Branch1][C][=O][O][O]

**Property**: The SCF Energy for the input molecule is -1.988e+05 eV.

### K.11 LogP Prediction

The LogP prediction task focuses on determining the octanol–water partition coefficient (LogP) of a given chemical compound, a key physicochemical property that influences molecular behavior across various environments. LogP quantifies the relative solubility of a compound in octanol versus water, serving as a critical indicator of lipophilicity, hydrophobicity, and membrane permeability. The task employs the SELFIES molecular representation, which encodes chemical structures in a machine-readable form optimized for deep learning models. Given the SELFIES representation

of a compound, the model is responsible for predicting its LogP value, a numerical measure that typically ranges from negative values (indicating high water solubility) to positive values (indicating high lipophilicity). This structured approach allows the model to learn patterns between molecular structure and partitioning behavior, enabling accurate and data-driven LogP estimation. One of the key challenges in LogP prediction is capturing the complex molecular interactions that dictate solubility behavior. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_compound> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Property.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: I am interested in the LogP of this molecule, could you tell me what it is? If uncertain, provide an estimate. Respond with the numerical value only. The molecule SELFIES sequence is:
**<SELFIES_compound>**: [C][O][C][=C][C][=Branch1][=Branch2][=C][C][=Branch1][Ring2][=C][Ring1][=Branch1][O]-[C@H1][C@H1][C@@H1] [Branch1][Branch1][C][O][Ring1][Branch1][C@@H1][Branch1][=Branch1][O][C][Ring1]-[#Branch1][=O][C][=C][C][=Branch1][=C][=C][Branch1] [=Branch2][C][=Branch1][Ring2][=C][Ring1][=Branch1][O]-[C][O][C][O]

**Property**: The LogP for the input molecule is 2.00.

### K.12 Molecular Weight Prediction

The molecular weight prediction task focuses on determining the molecular weight of a given chemical compound, a fundamental property that reflects its size and atomic composition. Molecular weight is a crucial parameter in various scientific disciplines, including organic synthesis, drug design, polymer chemistry, and materials science. It influences key aspects such as reaction stoichiometry, diffusion rates, bioavailability, and stability. The input follows the SELFIES format, a robust molecular encoding system designed for computational chemistry applications. The input consists of the SELFIES representation of a molecule, which the model processes to predict its molecular weight in unified atomic mass units (Da). This structured approach allows the model to learn the relationships between molecular structure and atomic composition, enabling precise and efficient molecular weight estimation. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_compound> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Property.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: Please provide me with the Molecular Weight value of this molecule. Determine the Molecular Weight value of this molecule. If uncertain, provide an estimate. Respond with the numerical value only. The molecule SELFIES sequence is:
**<SELFIES_compound>**: [C][O][C][=C][Branch1][#Branch1][C][=C][C][=N][Ring1][=Branch1][C][=Branch1][C][=O]-[N][C][C][C][C][C][N][Branch1][Branch1][C][C][Ring1][=Branch1][S][=Branch1][C][=O][=Branch1][C][=O][N][C][=Branch1][C][=O][N][C][C][C][C][C][C][Ring1] [Branch1][C][=C][Ring1][Branch1]

**Property**: The Molecular Weight for the input molecule is 491.60 g/mol.

### K.13 TOPOLOGICAL POLAR SURFACE AREA PREDICTION

The topological polar surface area (TPSA) prediction task focuses on determining the TPSA value of a given chemical compound, a key descriptor that reflects its molecular polarity and hydrogen bonding capacity. TPSA is widely used in cheminformatics, particularly in drug discovery, where it serves as an important predictor of solubility, permeability, and absorption. A compound's TPSA value influences its bioavailability, blood-brain barrier penetration, and interactions with biological membranes, making accurate prediction essential for pharmaceutical and materials research. The input is the SELFIES representation of the compound, and the model is tasked with predicting the compound's TPSA. The objective is to provide insights into the compound's polarity, solubility, and potential absorption characteristics, which are crucial considerations in areas such as drug discovery and materials research. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_compound> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Property.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: I would like to know the Topological Polar Surface Area of this molecule, can you provide it? If uncertain, provide an estimate. Respond with the numerical value only. The compound SELFIES sequence is:
**<SELFIES_compound>**:[C][C][=Branch2][=Branch1][=Branch2][=C][C][O][C][C][Branch1][O][C][Branch1][Ring2]-[O][Ring1][=Branch1][Branch1][C][C][C][O][C][C][C][C][Branch2][Ring2][#C][C][Branch2][Ring2][#Branch2][C][Br-anch2][Ring1][=Branch1][C][C][Branch1][P][C][Ring1][=Branch1][Branch1][O][C][Ring1][#Branch2][C][Ring2][Ring1]-[C][O][Ring1][Ring1][O][O][C][C][=C][Ring1][=N][N][C][=C][C][=C][C][=C][Ring1][=Branch2][Ring1][=Branch1][C]-[C][C]

**Property**: The Topological Polar Surface Area for the input molecule is 96.50 Å².

### K.14 COMPLEXITY PREDICTION

The complexity prediction task focuses on determining the structural complexity of a given chemical compound, a key property that reflects its architectural intricacy, stereochemical richness, and molecular connectivity. Molecular complexity plays a crucial role in synthetic chemistry, drug discovery, and materials science, where it impacts synthetic feasibility, resource requirements, and overall manufacturability. Compounds with high complexity may require multiple synthetic steps, specialized reagents, and intricate reaction conditions, whereas simpler molecules are generally easier to produce and optimize for industrial applications. The input is the SELFIES representation of the compound, and the model is tasked with predicting the compound's complexity. The objective is to shed light on the molecule's structural intricacy, which can influence its synthetic accessibility, resource requirements, and overall feasibility in various chemical processes. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_compound> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Property.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: I need to know the Complexity of this molecule, could you please provide it? If uncertain, provide an estimate. Respond with the numerical value only. The compound SELFIES sequence is:
**<SELFIES_compound>**: [C][C@@H1][Branch2][Ring2][#Branch2][C][C@@H1][Branch1][P][C][=Branch1][C][=O][N]-[C@@H1][Branch1][C][C][C] [=N][C][=C][S][Ring1][Branch1][N][Branch1][C][C][C][=Branch1][C][=O][C][C@H1]-[Branch1][C][C][C][Branch1][C][Cl][Branch1][C][Cl][Cl] [C][Branch1][C][Cl][Branch1][C][Cl][Cl]

**Property**: The Complexity for the input molecule is 572.00.

### K.15 DESCRIPTION Q&A

The description question and answer task involves responding to queries regarding a given compound's properties in physical chemistry and related fields. The input is the SELFIES representation of the compound. The model is tasked with providing accurate answers to detailed questions about the compound's physical and chemical attributes, encompassing a broad range of topics—from pharmacological considerations to the specific influence of structural and functional groups on biological activity (*e.g.*, in anticancer agents such as 4-Hydroxycyclophosphamide or Lobaplatin). The objective is to achieve a comprehensive and in-depth understanding of the compound's characteristics. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <SELFIES_compound> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: Description.<|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: What is the main component of Lobaplatin that gives it its anticancer properties? The compound SELFIES sequence is:
**<SELFIES_compound>**: [C][C@@H1][Branch1][=Branch1][C][=Branch1][C][=O][O-1][O-1].[C][C][C@H1]-[Branch1][=Branch1][C@@H1][Ring1][Ring2][C][N][C][N].[Pt+2]

**Description**: The main component of Lobaplatin that gives it its anticancer properties is the platinum cation (Pt+2).

### K.16 EXPERIMENTAL PROCEDURE PREDICTION

The experimental procedure prediction task focuses on generating a detailed, step-by-step protocol for conducting chemical experiments based on a given set of experimental components. This task plays a crucial role in automated synthesis planning, high-throughput experimentation, and robotic chemistry, where structured experimental procedures are essential for reproducibility, efficiency, and accuracy. The input is a mapping between experimental components and their corresponding SELFIES representations (*e.g.*, "Reactants: $index$: SELFIES" ...). The model is tasked with producing a structured sequence of operations that associates each component with the detailed steps of the experiment. The objective is to automate the chemical synthesis process by providing executable, structured experimental procedures. The prompt template is as follows.

**Template**

**User Input**: <user_identifier> ⊕ <graph_token>\n ⊕ Instruction. ⊕ <IDX_Reactants_MAP> ⊕ <IDX_Product_MAP> ⊕ <IDX_Catalysts_MAP> ⊕ <IDX_Solvents_MAP> ⊕ <|eot_id|> ⊕ <assistant_identifier>
**Assistant Output**: <ACTION_Sequence><|eot_id|>
**<user_identifier>**: <|start_header_id|>user<|end_header_id|>\n\n
**<assistant_identifier>**: <|start_header_id|>assistant<|end_header_id|>\n\n

**Example**

**Instruction**: From the provided starting materials and target compound, generate the step-by-step experimental protocol. The Action Sequence:
**<IDX_Reactants_MAP>**:Reactants:$4$:[START_SELFIES][O][=C][C][=C][C][=C][Branch1][O][O][C][C][C][O][C][C]-[O][Ring1][Branch1][C][=C][Ring1][=C][END_SELFIES] $3$: [START_SELFIES][C][C][Branch1][C][C][Branch1][C][C]-[O-1].[K+1][END_SELFIES]
**<IDX_Product_MAP>**:Product:$-1$:[START_SELFIES][C][=C][C][=C][C][=C][Branch1][O][O][C][C][C][O][C][C][O]-[Ring1][Branch1][C][=C][Ring1][=C][END_SELFIES]
**<IDX_Catalysts_MAP>**:Catalysts:$1$:[START_SELFIES][C][P+1][Branch1][=Branch2][C][=C][C][=C][C][=C][Ring1]-[=Branch1][Branch1][=Branch2][C][=C][C][=C][C][=C][Ring1][=Branch1][C][=C][C][=C][C][=C][Ring1][=Branch1].[Br-1][END_SELFIES]
**<IDX_Solvents_MAP>**: Solvents: $2$: [START_SELFIES][C][C][O][C][C][END_SELFIES]

**ACTION_Sequence**: ADD $1$ (2 liter, 57.2 g) ; ADD $2$ (500 ml) ; ADD $2$ ; STIR ; ADD $3$ (18.0 g) ; ADD $4$ (23.7 g) ; FILTER ; WASH with water ; WASH with sodium chloride ; WASH with water ; WASH with sodium chloride ; DRYSOLUTION over magnesium sulfate ; FILTER keep filtrate ; CONCENTRATE ; YIELD $-1$ (20.8 g).