MOL-MOE: TRAINING PREFERENCE-GUIDED ROUTERS FOR MOLECULE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in language models have enabled framing molecule generation as sequence modeling. However, existing approaches often rely on single-objective reinforcement learning, limiting their applicability to real-world drug design, where multiple competing properties must be optimized. Traditional multi-objective reinforcement learning (MORL) methods require costly retraining for each new objective combination, making rapid exploration of trade-offs impractical. To overcome these limitations, we introduce MOL-MOE, a mixture-of-experts (MoE) architecture that enables efficient test-time steering of molecule generation without retraining. Central to our approach is a preference-based router training objective that incentivizes the router to combine experts in a way that aligns with user-specified trade-offs. This provides improved flexibility in exploring the chemical property space at test time, facilitating rapid trade-off exploration. Benchmarking against state-of-the-art methods, we show that MOL-MOE achieves superior sample quality and steerability.

1 INTRODUCTION

De-novo drug design is inherently a multi-objective optimization problem (Kerns & Di, 2008). While drug discovery often focuses on pharmacological efficacy, successful therapeutics must simultaneously optimize across multiple axes: selectivity, clinical safety, and toxicity profiles. These compounds must exist within a highly constrained subset of chemical space that satisfies all these criteria, making the search for viable drug candidates particularly challenging and computationally intensive.

Recent advances in large language models have enabled powerful generative capabilities for molecule design through text-based representations. We reformulate this sequence generation task as a multi-objective reinforcement learning (MORL) problem, following the blueprint of Reinforcement Learning from Human Feedback (RLHF, Ouyang et al. (2022)) but extending it to handle multiple competing objectives simultaneously. While there exists a rich literature in MORL (Hayes et al., 2022), traditional scalarization techniques require complete retraining to explore different trade-offs, making them computationally impractical for rapid iteration in drug discovery workflows. To address this limitation, we follow the design choices adopted for rewarded soups (Ramé et al., 2023) and more generally, model merging. Precisely, rewarded soups are obtained by linearly interpolating different fine-tuned versions of the same model, a simple form of task arithmetics Yadav et al. (2023). While rewarded soups show advantages over direct prompting Wang et al. (2024), their linear interpolation of rewards proves restrictive for exploring the complex trade-off landscape of drug design.

We therefore introduce Molecule-MoE (MOL-MOE), a mixture-of-experts approach that enables dynamic steering of molecular generation across multiple objectives while maintaining computational efficiency through activation-space manipulation rather than full model retraining.

2 BACKGROUND

While a generative model trained on molecules reproduces the underlying distribution of compounds in the dataset, our goal extends beyond this: we aim to sample more compounds from promising

regions of the chemical space by targeting specific areas of the learned molecular space that align with desired properties. Although the features describing molecules in this target region can be learned from wet lab discoveries, extracting high-quality labels is costly Huang et al. (2021). This is why machine learning classifiers trained on this data are often used to provide approximate targets for downstream tasks Ghugare et al. (2023); Tripp et al. (2022) rather than relying on human annotators. This enables the alignment of language models in the molecular space with desired molecule properties through RLHF techniques, such as Direct Preference Optimization (DPO) Cavanagh et al. (2024), Proximal Policy Optimization (PPO) Schulman et al. (2017), or REINFORCE Leave-One-Out (RLOO) Ahmadian et al. (2024).

The need to balance multiple criteria in real-world applications has motivated two primary approaches to multi-objective alignment in large language models. The first approach uses supervised fine-tuning (SFT) Yang et al. (2024), which has been applied to drug discovery tasks. The second approach employs model merging techniques and multi-objective RLHF (MORLHF) Ramé et al. (2023), though its application to drug discovery remained unexplored prior to our work. Our experiments reveal fundamental limitations in both approaches: linear weight interpolation (model soups) fails to capture complex nonlinear relationships between objectives, while pareto-conditioning SFT methods Yang et al. (2024) struggle with extrapolation beyond their training datasets.

Desiderata In the context of drug design, we aim to assess these approaches against two main criteria: quality of the generated samples and precision in following target preferences (steerability). Our analysis reveals that while model merging approaches show promise in combining multiple objectives, they suffer from two key limitations: (1) the inability to capture complex non-linear relationships between different molecular properties and (2) sub-optimal performance when precise control over individual properties is required. These limitations stem from the inherent constraints of linear interpolation in model merging, where the influence of each expert model is determined by fixed weights rather than as a by-product of a dynamic and input-dependent *routing* decision.

To address these challenges, we draw inspiration from Mixture of Experts (MoE) Jiang et al. (2024) architectures, which have demonstrated success in handling complex, multi-modal tasks by dynamically routing inputs to specialized expert networks Komatsuzaki et al. (2023). We further introduce MOL-MOE, a novel architecture built upon model merging that leverages the strengths of both approaches: we combine multiple expert MLP layers in a mixture of experts blocks Zhou et al. (2022) and devise a routing training task to improve steerability. This architecture introduces dynamic, input-dependent weighting of various molecular property objectives, enhancing steerability while preserving the test-time advantages of model merging, rather than relying solely on training-time optimizations.

To summarize, we list our contributions:

- We extend recent advancements in multi-objective RL—such as supervised fine-tuning (SFT) and model merging—to the domain of molecule design for drug discovery. We evaluate their performance in terms of both quality and controllability, as detailed in Sections 5.2 and 5.3.
- We propose a novel approach based on model merging and mixture of experts: MOL-MOE, combining property-specific molecule LMs with a tuning task aimed at improving steerability (Section 4).
- We conduct a series of analyses to investigate several key factors: out-of-distribution performance, the relationship between merging hyperparameters and performance, the impact of the RLHF algorithm and model size, and the choice of merging technique (Section 5 and Appendix E).

3 PROBLEM SETUP

Drug design requires optimizing multiple molecular properties across diverse categories: ADME (absorption, distribution, metabolism, excretion) properties Huang et al. (2021), drug-likeness scores, and binding affinities. For high-quality drug candidates, the number of relevant properties typically reaches around 10 Fang et al. (2023), varying based on the stage of drug development, desired property granularity, and available domain knowledge. Additionally, as clinical trials progress, new data from diverse patient cohorts necessitates efficient updates to multiple property estimators

Fang et al. (2023). However, for the purpose of our experiments, we consider a stationary multi-objective MDP where these properties remain fixed.

Under this formulation, a variety of techniques is applicable:

1. **Multi-objective RL**: a single policy optimizes the combination of multiple objectives and a provided tradeoff, Appendix C.1.
2. **Expert policy interpolation**: the combination of single-task expert policies in the weight space, Appendix C.2.
3. **Supervised fine-tuning**: a single policy is trained on a causal language modeling task to predict the molecule from the provided desired properties, Appendix C.3.

We discuss in Appendix C the limitations of these methodologies, further motivating the introduction of MOL-MOE.

4 MOL-MOE: MOLECULE-MIXTURE OF EXPERTS

Linear interpolation methods in model soups and reward scalarization share surface-level similarities, but their underlying mechanisms differ in important ways. The relationship between preference vectors $\mathbf{w}_{ij}$ for policy parameters $\boldsymbol{\theta}_i$ and model soup coefficients is neither equivalent in magnitude nor necessarily linear.

Our correlation analysis (Appendix E.4) reveals that simply assigning equal weights to policies that maximize individual tasks does not guarantee optimal multi-task performance. This limitation has led researchers to explore more sophisticated approaches, such as iterative and gradient-based optimization methods, to find optimal parameter combinations Huang et al. (2024); Prabhakar et al. (2024).

Furthermore, the challenge becomes more complex with increasing network depth, as the semantic meaning of representations evolves across layers. Recent work has proposed systematic approaches to address this issue. Model Breadcrumbs Davari & Belilovsky (2024) computes update masks based on the distribution of task-specific parameters to mitigate interference patterns. TIES Yadav et al. (2023) extends this by introducing element-wise merging rules based on the relative magnitudes and signs of corresponding parameters - when parameters from different tasks have opposing signs above a threshold, it preserves the dominant direction rather than averaging them.

While these approaches introduce data-driven methods to improve parameter merging through similarity analysis and interference detection, they ultimately rely on predetermined merging rules. To move beyond fixed combination strategies, we propose leveraging the **Mixture of Experts** Jiang et al. (2024) architecture, which learns to route inputs to specialized components dynamically. In Appendix B we describe in detail the implementation of MOL-MOE and the training procedure.

5 EXPERIMENTS

In this section, we evaluate MOL-MOE against scalarized multi-objective RLHF (MORLHF) variants under a fixed scalarization with uniform weights, Rewarded Soups (RS), and Rewards in Context (RiC). Our evaluation is based on two main criteria: *maximization*, that is sample quality over the considered tasks (Section 5.2); *steerability*, how effectively each method tunes the model to follow conditioning preferences $\mathbf{w}_j \in \mathcal{W}$ at test-time (Section 5.3).

We report further details concerning the implementation details in Appendix D and supplementary analyses supporting our design choices in Appendix E. In Appendix E.4 we further investigate the relationship between conditioning preferences $\mathbf{w}_j$ and property scores $r_i(x)$.

5.1 EXPERIMENTAL SETTING

Molecule pre-training We assemble a training set of approximately 3.65 million molecule sequences by merging datasets from CHEMBL Gaulton et al. (2012), ZINC 250k Sterling & Irwin (2015), and MOSES Polykovskiy et al. (2020). Using the Llama 3.2 tokenizer Aaron Grattafiori

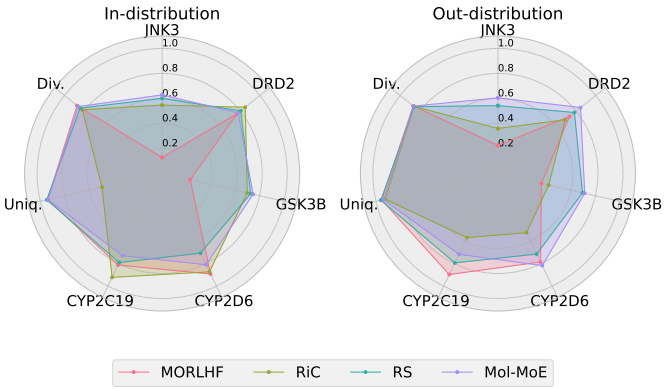

Figure 1: Average best task scores achieved by models trained on the full dataset (left) or with held-out high-quality samples (right), by varying tuning method. MOL-MOE outperforms the baselines particularly out of distribution, where RiC fails to improve beyond on the training examples. Additionally, MORLHF fails to learn more than three tasks.

(2024), this dataset translates to approximately 151 million training tokens, with an average of 41.44 tokens per molecule. We represent molecule sequences in the SMILES format Weininger (1988), allowing us to work in text space and ensuring compatibility with existing foundational models.

To create the base molecule language model, MOL-LLAMA we fine-tune the Llama 3.2 1B model on the 3.65M dataset for causal language modeling, where the model learns to predict each token based on all previous tokens in the sequence. The scaling analysis detailed in Section E.3 informs the model size choice.

Property RLHF fine-tuning We derive MOL-MOE from the pipeline shown in Figure 2, implementing our approach using MergeKit Goddard et al. (2025). First, we obtain five experts by fine-tuning five copies (see appendix D.2) of the base model on five molecule properties separately, using the scoring functions provided by Huang et al. (2021), i.e. receptor binding scores.

To combine these experts effectively, we create MoE blocks where property-specific patterns guide the routing. Specifically, we initialize the router weights using activation patterns obtained by passing each property name (e.g., `JNK3`, `DRD2`) through the model. This initialization ensures the routers begin with a strong bias toward directing inputs to their corresponding expert models. After initialization, we train only the router networks using RLOO Ahmadian et al. (2024), keeping all other layers frozen. Training details are reported in Appendix D.3.

We report in Appendix A a visualization of the pipeline implemented to train MOL-MOE.

5.2 MAXIMIZING MOLECULE PROPERTIES

We evaluate different model tuning approaches across seven metrics. Five are target molecule properties from the Therapeutics Data Commons (TDC) Huang et al. (2021): `JNK3`, `DRD2`, `GSK3`β, `CYP2D6` and `CYP2C19`. We additionally evaluate on *diversity* – which we define as the average pairwise Tanimoto distance between generated molecules Ghugare et al. (2023) – and *uniqueness*, which is the fraction of unrepeated samples.

To evaluate each method's peak performance capability, we generate $2,048$ molecules across 20 different random seeds with each tuned model. We identify the molecule with the highest average score across all properties for each set, then average these peak scores across all seeds to obtain our final performance metric. For fair comparison across methods, we configure MORLHF and RS with uniform weights ($w_{ij} = \frac{1}{m} = 0.2$ for each of the 5 properties). RiC uses maximum target values ($\hat{r}_i = 1.0$) in its prompt format: `<JNK3=1.00><DRD2=1.00>...<s>`. MOL-MOE uses the same prompt format but with normalized preference weights ($\hat{w}_{ij} = \frac{w_{ij}}{\sum_{k=1}^{m} w_{kj}}$).

Figure 3 shows that single-task RLHF achieves suboptimal performance across objectives, though it demonstrates positive transfer effects, particularly from `CYP2D6` and `CYP2C19` to `DRD2` and

GSK3β. MORLHF performs comparably to single-task tuning, but notably optimizes only a subset of the considered tasks rather than the full objective set. The base model fine-tuned with RiC excels in-distribution, performing comparably or better than RS and MOL-MOE for DRD2 and CYP2C19. However, its performance degrades substantially when evaluating on out-of-distribution high-quality molecules. In contrast, the Rewarded Soups (RS) approach maintains robust performance even out of distribution, leveraging experts explicitly trained to maximize individual objectives. MOL-MOE emerges as the strongest performer, achieving superior results in both scenarios and significantly outperforming RiC and MORLHF, particularly in out-of-distribution cases.

We further observe that with temperature $t = 1.0$, all the models do not repeat sequences from the training set, leading to a maximum novelty score with the metric implemented by Huang et al. (2021). Tabular results are reported in Table 2.

5.3 MEASURING STEERABILITY

To evaluate steerability, we generate a test set by sampling 20 preference vectors from a 5-dimensional Dirichlet distribution, yielding $W \in \mathbb{R}^{20\times5}$. For each preference vector, we sample 2,048 molecules. We then compute steerability error for each molecular property. For RiC, which directly predicts absolute reward values, the error is measured against the prompted rewards:

$$\mathrm{MAE}(x) = \frac{1}{m}\sum_{i=1}^{m}(r_i(x) - \hat{r}_i)$$

For RS and MOL-MOE, we instead measure how well the generated molecules' properties match the requested preference proportions. This is done by comparing the requested preference weights against the normalized property scores:

$$\mathrm{MAE}(x) = \frac{1}{m}\sum_{i}^{m}\left(w_{ij} - \frac{r_i(x)}{\sum_{k=1}^{m} r_k(x)}\right),\ \mathbf{w}_j \in \mathcal{W}$$

where w_{ij} represents the requested preference weight for property i, and the fraction $\frac{r_i(x)}{\sum_l^m r_l(x)}$ represents the actual proportion of property i in the generated molecule relative to all properties.

Figure 1 and Figure 10 report the performance of different methods in terms of maximization and steerability metrics, showing that MOL-MOE improves over baselines in both evaluation criteria.

We observe that JNK3 is a hard task in both maximization (Figure 1) and steerability (Figure 10), as all three methods achieve comparable scores. In GSK3β, CYP2D6, CYP2C19, MOL-MOE achieves the lowest error across properties, suggesting that the routing networks successfully learned to aggregate the contribution from different property experts as instructed.

6 CONCLUSION

This work approaches molecule generation as a multi-objective optimization problem using language modeling. We evaluated three existing approaches: Rewards in Context (RiC), multi-objective reinforcement learning (MORLHF), and Rewarded Soups (RS), each with distinct limitations. To address these, we introduced MOL-MOE, a mixture-of-experts architecture with preference-guided routing. This novel approach enables better calibration between preference weights and model outputs, resulting in more precise steerability and finer-grained control over molecular properties. While RiC and RS showed strong in-distribution performance, MOL-MOE maintains this for novel compounds while offering superior control over property trade-offs. However, our approach assumes independent optimization of molecular properties, which may not hold in real-world drug design due to non-monotonic interactions. Future work should explore graph-based routing architectures or geometric deep learning approaches to model these interactions explicitly. Additionally, scaling MOL-MOE to billion-scale molecular datasets could enhance its potential in molecular design, enabling discovery of novel trade-offs and efficient adaptation to new optimization objectives.

REFERENCES

Abhinav Jauhri Abhinav Pandey the Meta LLaMA team et al. Aaron Grattafiori, Abhimanyu Dubey. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024. URL https://arxiv.org/abs/2402.14740.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling, 2023. URL https://arxiv.org/abs/2304.01373.

Joseph M. Cavanagh, Kunyang Sun, Andrew Gritsevskiy, Dorian Bagni, Thomas D. Bannister, and Teresa Head-Gordon. Smileyllama: Modifying large language models for directed chemical space exploration, 2024. URL https://arxiv.org/abs/2409.02231.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling, 2021. URL https://arxiv.org/abs/2106.01345.

MohammadReza Davari and Eugene Belilovsky. Model breadcrumbs: Scaling multi-task model merging with sparse masks, 2024. URL https://arxiv.org/abs/2312.06795.

Cheng Fang, Ye Wang, Richard Grater, Sudarshan Kapadnis, Cheryl Black, Patrick Trapa, and Simone Sciabola. Prospective validation of machine learning algorithms for absorption, distribution, metabolism, and excretion prediction: An industrial perspective. *Journal of Chemical Information and Modeling*, 63(11):3263–3274, 2023. doi: 10.1021/acs.jcim.3c00160. URL https://doi.org/10.1021/acs.jcim.3c00160. PMID: 37216672.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis, 2020. URL https://arxiv.org/abs/1912.05671.

A. Gaulton, L. J. Bellis, A. P. Bento, J. Chambers, M. Davies, A. Hersey, Y. Light, S. McGlinchey, D. Michalovich, B. Al-Lazikani, and J. P. Overington. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Res*, 40(Database issue):D1100–D1107, January 2012. doi: 10.1093/nar/gkr777.

Raj Ghugare, Santiago Miret, Adriana Hugessen, Mariano Phielipp, and Glen Berseth. Searching for high-value molecules using reinforcement learning and transformers, 2023. URL https://arxiv.org/abs/2310.02902.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's mergekit: A toolkit for merging large language models, 2025. URL https://arxiv.org/abs/2403.13257.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, Johan Ferret, and Mathieu Blondel. Direct language model alignment from online ai feedback, 2024. URL https://arxiv.org/abs/2402.04792.

Conor F. Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1), April 2022. ISSN 1573-7454. doi: 10.1007/s10458-022-09552-y. URL http://dx.doi.org/10.1007/s10458-022-09552-y.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic lora composition, 2024. URL https://arxiv.org/abs/2307.13269.

Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W. Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development, 2021. URL https://arxiv.org/abs/2102.09548.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts, 2024. URL https://arxiv.org/abs/2401.04088.

Edward Kerns and Li Di. *Drug-like Properties: Concepts, Structure Design, and Methods: From ADME to Toxicity Optimization*. Academic Press, San Diego, California, 1st edition, 2008. ISBN 978-0-12-369520-8.

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, 2023. URL https://arxiv.org/abs/2212.05055.

Abhishek Naik, Yi Wan, Manan Tomar, and Richard S. Sutton. Reward centering, 2024. URL https://arxiv.org/abs/2405.09999.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, Artur Kadurin, Simon Johansson, Hongming Chen, Sergey Nikolenko, Alan Aspuru-Guzik, and Alex Zhavoronkov. Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models. *Frontiers in Pharmacology*, 2020.

Akshara Prabhakar, Yuanzhi Li, Karthik Narasimhan, Sham Kakade, Eran Malach, and Samy Jelassi. Lora soups: Merging loras for practical skill composition tasks, 2024. URL https://arxiv.org/abs/2410.13025.

Alexandre Ramé, Guillaume Couairon, Mustafa Shukor, Corentin Dancette, Jean-Baptiste Gaya, Laure Soulier, and Matthieu Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards, 2023. URL https://arxiv.org/abs/2306.04488.

Mathieu Reymond, Eugenio Bargiacchi, and Ann Nowé. Pareto conditioned networks, 2022. URL https://arxiv.org/abs/2204.05036.

Amelie Royer, Ilia Karmanov, Andrii Skliar, Babak Ehteshami Bejnordi, and Tijmen Blankevoort. Revisiting single-gated mixtures of experts, 2023. URL https://arxiv.org/abs/2304.05497.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Teague Sterling and John J. Irwin. Zinc 15 – ligand discovery for everyone. *Journal of Chemical Information and Modeling*, 55(11):2324–2337, 2015. doi: 10.1021/acs.jcim.5b00559. URL https://doi.org/10.1021/acs.jcim.5b00559. PMID: 26479676.

Austin Tripp, Wenlin Chen, and José Miguel Hernández-Lobato. An evaluation framework for the objective functions of de novo drug design benchmarks. In *ICLR2022 Machine Learning for Drug Discovery*, 2022. URL https://openreview.net/forum?id=W1tcNQNG1S.

Kaiwen Wang, Rahul Kidambi, Ryan Sullivan, Alekh Agarwal, Christoph Dann, Andrea Michi, Marco Gelmi, Yunxuan Li, Raghav Gupta, Avinava Dubey, Alexandre Ramé, Johan Ferret, Geoffrey Cideron, Le Hou, Hongkun Yu, Amr Ahmed, Aranyak Mehta, Léonard Hussenot, Olivier Bachem, and Edouard Leurent. Conditional language policy: A general framework for steerable multi-objective finetuning, 2024. URL https://arxiv.org/abs/2407.15762.

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. doi: 10.1021/ci00057a005. URL https://doi.org/10.1021/ci00057a005.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models, 2023. URL https://arxiv.org/abs/2306.01708.

Prateek Yadav, Tu Vu, Jonathan Lai, Alexandra Chronopoulou, Manaal Faruqui, Mohit Bansal, and Tsendsuren Munkhdalai. What matters for model merging at scale?, 2024. URL https://arxiv.org/abs/2410.03617.

Rui Yang, Xiaoman Pan, Feng Luo, Shuang Qiu, Han Zhong, Dong Yu, and Jianshu Chen. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment, 2024. URL https://arxiv.org/abs/2402.10207.

Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation, 2019. URL https://arxiv.org/abs/1908.08342.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch, 2024. URL https://arxiv.org/abs/2311.03099.

Maryam Zare, Parham M. Kebria, Abbas Khosravi, and Saeid Nahavandi. A survey of imitation learning: Algorithms, recent developments, and challenges, 2023. URL https://arxiv.org/abs/2309.02473.

Artem Zholus, Maksim Kuznetsov, Roman Schutski, Rim Shayakhmetov, Daniil Polykovskiy, Sarath Chandar, and Alex Zhavoronkov. Bindgpt: A scalable framework for 3d molecular design via language modeling and reinforcement learning, 2024. URL https://arxiv.org/abs/2406.03686.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew Dai, Zhifeng Chen, Quoc Le, and James Laudon. Mixture-of-experts with expert choice routing, 2022. URL https://arxiv.org/abs/2202.09368.

APPENDIX

A PIPELINE

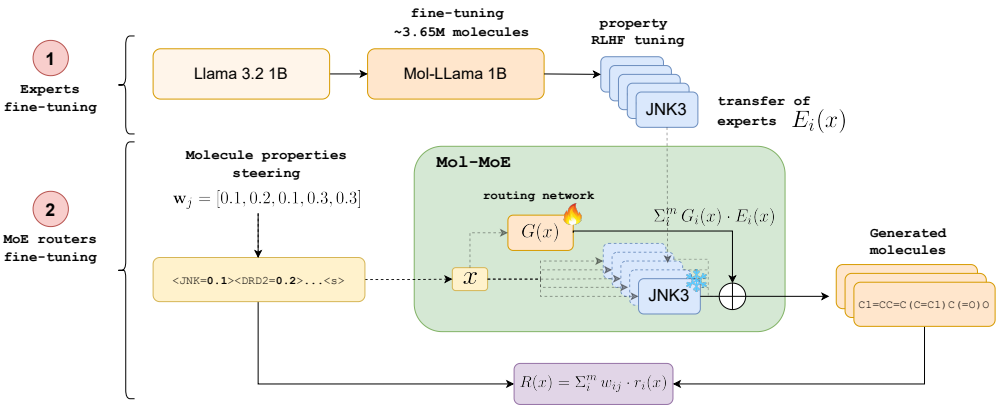

Figure 2: Pipeline illustration of MOL-MOE: a reference model is pre-trained on a large set of molecules; expert models are derived with RLHF-tuning on each desired molecule property; in the fine-tuned model, MoE blocks are added to combine the tuned expert layers, only the router networks are further tuned with the routing task.

B MOL-MOE: TRAINING PREFERENCE-GUIDED ROUTERS FOR MOLECULE GENERATION

We propose leveraging the **Mixture of Experts** Jiang et al. (2024) architecture, which learns to route inputs to specialized components dynamically.

In each Transformer layer ℓ, the feed-forward layers after attention are replaced with a MoE-block consisting of m expert networks $E_i : \mathbb{R}^d \to \mathbb{R}^d$, where each $E_i(\mathbf{x})$, $i \in \{1, \ldots, m\}$ is implemented as a multi-layer perceptron (MLP). The routing mechanism is controlled by a router network $f_{\boldsymbol{\theta}_g} : \mathbb{R}^d \to \mathbb{R}^m$ with parameters $\boldsymbol{\theta}_g$, which computes routing weights for each expert. For each input token $\mathbf{x} \in \mathbb{R}^d$, the router outputs a probability distribution $\mathbf{G}(\mathbf{x}) \in \mathbb{R}^m$ over experts:

$$\mathbf{G}(\mathbf{x}) := \text{softmax}(\text{topk}(f_{\boldsymbol{\theta}_g}(\mathbf{x}))) \ ,$$

where topk selects the K largest logits and sets the rest to $-\infty$, ensuring that each token is processed by at most K experts. Here, K represents the number of experts assigned to each token. In our setup, we set $K = m$, as the number of specialists is known a priori and corresponds directly to the properties we aim to optimize. This ensures that all specialist layers contribute to constructing the target molecule, each addressing a specific property. Additionally, we include the initialization model—trained on a large dataset of molecules—as an extra expert to provide regularization Royer et al. (2023). The final output for each token combines the expert outputs weighted by their corresponding routing probabilities:

$$\hat{\mathbf{x}} = \sum_{i=1}^{m} G_i(\mathbf{x}) \cdot E_i(\mathbf{x}) \ ,$$

where $G_i(\mathbf{x})$ denotes the i-th component of the routing vector $\mathbf{G}(\mathbf{x})$.

B.1 PREFERENCE-GUIDED ROUTER TRAINING

Our method employs a data-driven strategy for dynamic model merging via learned routing. Given a set of expert policies $\{E_i\}_{i=1}^m$ and a preference vector $\mathbf{w}_j \in \Delta^{m-1}$, where Δ^{m-1} represents the probability simplex (i.e., $\sum_{i=1}^m w_{j,i} = 1$ and $w_{j,i} \geq 0$ for all i), we encode preferences into

the input prompt using a structured format such as "<JNK3=w1><DRD2=w2>...<s>". This prompted input is then processed through our routing architecture to learn an input-dependent routing function $\mathbf{G}(\mathbf{x}; \boldsymbol{\theta}_g)$, which determines how to combine the outputs of the experts. Similar to the preference weights, the routing weights also form a probability distribution over the experts, i.e., $\sum_{i=1}^{m} G_i(\mathbf{x}) = 1$ and $G_i(\mathbf{x}) \geq 0$ for all i.

The router parameters $\boldsymbol{\theta}_g$ are trained using reinforcement learning across a distribution of preference vectors. For each training episode, we sample $\mathbf{w}_j$ from a set of possible preferences $\{\mathbf{w}_j\}_{j=1}^{N} \subset \Delta^{m-1}$, encode it into the prompt, and use this to define the episode's scalarized reward. By training over the distribution of preference vectors, we optimize the expected reward across all possible preference configurations:

$$\mathbb{E}_{\mathbf{w}_j \sim P(\Delta^{m-1})} \left[R_j(\mathbf{x}) \right] = \mathbb{E}_{\mathbf{w}_j \sim P(\Delta^{m-1})} \left[\sum_{i=1}^{m} w_{j,i} \cdot r_i(\mathbf{x}) \right].$$

This formulation ensures that the router learns to interpret preferences from the prompt and dynamically adjusts the contributions of each expert to align with these preferences, without requiring post-hoc calibration. In contrast, static merging approaches like Model Breadcrumbs or TIES must solve an additional optimization problem to match their fixed merging rules with desired preference weights, often resulting in suboptimal trade-offs due to their predetermined combination strategies.

Our approach shares the high-level goal of preference-conditioned generation with Conditional Language Policy (CLP) Wang et al. (2024), but differs significantly in implementation. While CLP fine-tunes the entire model to condition its policy on user preferences directly, our method leaves the underlying expert policies unchanged. Instead, we introduce a routing mechanism that dynamically combines the outputs of pre-trained experts based on preferences encoded in the input prompt. By embedding preferences directly into the prompt using a structured format, our architecture provides a flexible and robust way to condition model behavior using natural language while minimizing sensitivity to prompt variations.

Unlike traditional Mixture of Experts (MoE) architectures Jiang et al. (2024), which train experts jointly from scratch, our method leverages pre-trained task-specific policies through activation-level routing—a process referred to as "upcycling" Komatsuzaki et al. (2023). This approach eliminates the need for explicit expert specialization, as each expert is already optimized for a specific molecular property, while the router learns to blend their outputs based on input preferences adaptively.

We choose RLOO Ahmadian et al. (2024) as training algorithm, after an empirical evaluation across alternatives (Section E.1). The objective functions $r_i(x)$ consist of MLP property classifiers trained on clinical data Huang et al. (2021): the models estimate the likelihood of an input molecule to satisfy the target property, similarly to examples observed in the past; the assumption is that the training set is sufficiently large for these models to well extrapolate for similar structures.

C Baseline methodologies

C.1 Multi-Objective RL

We formulate drug design as a multi-objective Markov decision problem (MOMDP), where we aim to optimize a set of competing molecular properties represented by the reward vector $\mathbf{r}(x) = [r_1(x), ..., r_m(x)] \in \mathbb{R}^m$. Each objective is weighted by a preference vector $\mathbf{w}_j \in \mathcal{W} \subseteq \mathbb{R}^m$, where $\mathcal{W}$ is the space of valid preference vectors, and can be either learned (using preference modelling methods) or pre-specified. Let $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{A}$ be a policy function mapping states to actions, parameterized by $\boldsymbol{\theta}$. Multi-objective RL (MORL) approaches can be categorized into two main design paradigms: *single-policy* or *multi-policy* methods. Single-policy methods train a unique policy parameterized by $\boldsymbol{\theta}^*$ that either generalizes across all possible preferences $\mathcal{W}$ or maximizes a specific preference vector $\mathbf{w}_j$. However, these approaches often yield suboptimal performance or fail to scale to complex tasks as the burden falls primarily on the network's generalization capacity and quality of the training distribution Yang et al. (2019).

Conversely, *multi-policy* methods instead maintain a set of policies, each parameterized by $\boldsymbol{\theta}_j^* \in \Theta$, and optimized for a specific preference vector $\mathbf{w}_j = [w_{j,1}, ..., w_{j,m}]$. In the most straightforward

implementation, objectives are combined through scalarization:

$$R_j(x) = \sum_{i=1}^{m} w_{j,i} r_i(x)\,,\ \mathbf{w}_j \in \mathcal{W}$$
$$\boldsymbol{\theta}_j^* = \text{argmax}_{\boldsymbol{\theta}\in\Theta}\, R_j(f_{\boldsymbol{\theta}}(x))$$

The scalarized reward function $R_j : \mathcal{X} \to \mathbb{R}$ then reduces the multi-objective problem to a standard single-objective RL optimization for the given preference vector $\mathbf{w}_j$. In our context, we refer to multi-policy MORL methods as *Multi Objective RLHF* (MORLHF) since traditional approaches in this field tend to elicit the reward vector $\mathbf{r}$ from human feedback. Despite not requiring human annotation in our work, we maintain this terminology for consistency. In the molecular sequence modeling context, a policy parameterized by $\boldsymbol{\theta}$ and fine-tuned on a large molecular space can be optimized for a preference vector $\mathbf{w}_j \in \mathcal{W}$ using any alignment algorithm (e.g., RLOO, DPO, PPO), yielding $\boldsymbol{\theta}_j^*$.

The multi-policy (MORLHF) setup faces two practical challenges: first, each preference vector $\mathbf{w}_j \in \mathcal{W}$ requires training a separate policy, making preference space exploration costly; second, the varying scales of rewards $r_i(x)$ necessitate careful scaling of preferences to ensure stable training, often requiring techniques like reward centering Naik et al. (2024). We analyze these challenges in Section E.6.

C.2 Expert Policy Interpolation

Alternatively, Ramé et al. (2023) leverages Linear Mode Connectivity (LMC) Frankle et al. (2020) in Transformers to derive multi-task policies through linear interpolation of task-specialized policies. For a set of m tasks, we first obtain specialized policies parameterized by $\boldsymbol{\theta}_i$ through single-task optimization. For each *task* (property), we compute:

$$\boldsymbol{\theta}_i = \text{argmax}_{\boldsymbol{\theta}\in\Theta}\, r_i(f_{\boldsymbol{\theta}}(x)) \quad \text{for } i \in \{1, \ldots, m\}$$

These task-specific policies form a *model soup* (referred to as a *rewarded soup*, RS, in this context). The key property of this collection is that the policy parameters $\{\boldsymbol{\theta}_i\}_{i=1}^m$ can be linearly combined using preference weights to yield a merged policy whose parameters $\boldsymbol{\theta}_j^*$ are obtained as:

$$\boldsymbol{\theta}_j^* = \sum_{i=1}^{m} w_{j,i} \boldsymbol{\theta}_i \quad \text{where } \mathbf{w}_j \in \mathcal{W}$$

This merged policy aims to reflect the relative importance of each property as specified by the preference vector $\mathbf{w}_j \in \mathcal{W}$. This method features good steerability Wang et al. (2024) while benefiting from the rich advances in model merging Yadav et al. (2023); Davari & Belilovsky (2024) meant to mitigate task interference. Remarkably, generating policies for new preference vectors requires only parameter interpolation, without additional training, as the multi-task policies are computed offline.

This feature is particularly relevant in the context of molecule design, where numerous properties must be incorporated over time. The relative importance of these properties is often difficult to predict in advance, especially as clinical data becomes available and may introduce distributional shifts. This necessitates using flexible methods that can efficiently adapt to evolving preferences without requiring costly retraining.

One of the main drawbacks of this approach, which we aim to address in this work, is that the relationship between the defined preference set $\mathbf{w}_j \in W$ and the corresponding task performance is not always linear. As a result, predicting how a preference vector should be translated into an interpolation weight for the rewarded soup mode is challenging. To address this, we experiment with learning a mapping function, such as $w'_{ij} = z(w_{ij})$ (Yang et al., 2024), to ensure that models derived from rewarded soups are steerable—i.e., capable of responding to the given preference vector as intended. However, we demonstrate that a more effective approach is to learn an input-dependent weighting function through a dedicated *routing* task, as implemented in our proposed MOL-MOE architecture.

C.3 SUPERVISED LEARNING

Yang et al. (2024) propose a fully supervised approach inspired by imitation learning Zare et al. (2023) and aligned with Pareto-conditioning techniques Reymond et al. (2022); Chen et al. (2021). This framework provides the desired property scores as a prefix when conditioning the model on a target molecule. The underlying assumption is that causal language modeling enables the Transformer to learn the relationship between input rewards (interpreted as a state) and the target molecule sequence (interpreted as an action). This mirrors the way decision transformers Chen et al. (2021) learn to associate trajectory returns with optimal actions. The model is initially tuned on a large set of known molecules; then a filtering process defines a subset of Pareto-optimal samples augmented by the model generations, and finally, the network undergoes a second fine-tuning step on the high-quality dataset. This approach is identified as **Rewards in Context** (RiC, Yang et al. (2024)), it achieves comparable or superior sample quality wrt. MORLHF and RS. However, as this method heavily relies on the quality of the training dataset, it can be prone to overfitting. With an ablation analysis in Section E.5, we show that models trained with RiC fail to generate molecules better than the training examples, and the fraction of repeated generations is high. In the context of drug design, our goal is not only to replicate the statistical patterns of the training data but also to surpass the quality of the imitation data—generating novel, high-quality compounds that were not present in the training set. While RiC offers test-time steerability similar to RS approaches, its inability to generalize (or "maximize") beyond the training data motivates our development of the MOL-MOE approach to combine the strengths of both methodologies while avoiding their pitfalls.

D IMPLEMENTATION DETAILS

D.1 MOLECULE FINE-TUNING

We fine-tune the base model "Mol-Llama 1B" for causal language modeling on $\sim$3.65M molecules, no conditioning information are provided: we add segmentation tokens to structure the model outputs, `<s> C1=CC=C(C=C1)C(=O)O </s>`. We consider a standard learning rate $\eta = 2e-5$ (with cosine scheduling), batch size 1024 with gradient accumulation; the model is trained for 1 epoch.

D.2 EXPERTS FINE-TUNING

We employ RLOO Ahmadian et al. (2024) to fine-tune the different experts, rewarding each model on $100,000$ generations prompted with the start of sequence token `<s>`; batch size of 512 samples; a constant learning rate of $\eta = 1e-7$; a KL coefficient of $\beta = 0.2$ to prevent policy collapse. We automatically assign a zero reward score to invalid SMILES string.

D.3 PREFERENCE-GUIDED ROUTER TRAINING

We employ RLOO Ahmadian et al. (2024) to train the router networks in MOL-MOE while keeping the rest of the model layers frozen. The model is rewarded on $100,000$ generations prompted with a conditioning prompt encoding the preferences $w_{ij} \in \mathbf{w}_j, \mathbf{w}_j \in \mathcal{W}$ in the form "`<JNK3=w1><DRD2=w2>...<s>`". We choose a fixed batch size of 128 samples; learning rate $\eta = 1e-3$; KL coefficient $\beta = 5e-2$.

D.4 SAMPLING PARAMETERS

We keep the sampling parameters constant across the experiments: temperature t $= 1.0$; nucleus sampling as default strategy with top_p $= 0.9$; max output length of 64 tokens.

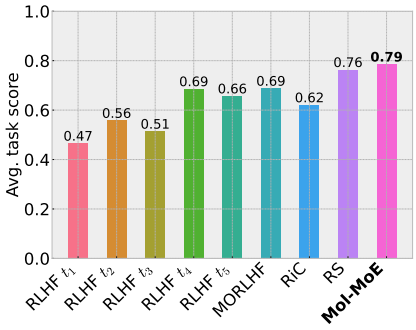

Figure 3: Out of distribution average property score by fine-tuning method. MOL-MOE outperforms multi-task models (MORLHF, RiC, RS) and by a significant margin also task experts (RLHF t_1 =JNK3, t_2 =DRD2, t_3 =GSK3β, t_4 =CYP2D6, t_5 =CYP2C19). Higher scores for experts trained on t_4, t_5 indicate positive transfer.

E SUPPLEMENTARY ANALYSES

E.1 CHOICE OF THE RLHF ALGORITHM

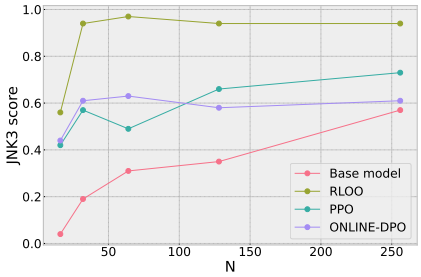

Figure 4: Task performance at sampling size by RL tuning method. Overall, RLOO surpasses alternative methods by a significant margin, while PPO and Online-DPO achieve comparable performance.

We compare three algorithms widely applied in RLHF: PPO Schulman et al. (2017), RLOO Ahmadian et al. (2024) and Online-DPO Guo et al. (2024). Each method is applied to fine-tune the base molecule sequence model on the non-trivial JNK3 property. For each variant, we sample a growing set of N molecules from the model and take the best-of-N property scores. We consider a moderate coefficient $\beta = 0.2$ for the KL divergence.

In Figure 4 we observe the model generates qualitatively the best molecules with RLOO, which is in line with comparisons reported by Ahmadian et al. (2024). Overall, we observe RL fine-tuning significantly improves the output distribution of molecules wrt. the base model, as it is possible to generate equivalently good compounds with $\sim$ 10x less samples.

E.2 CHOICE OF THE MERGING TECHNIQUE

A major motivation underlying this work is to explore the efficacy of weight interpolation in MORL. Combining linearly expert policies, as in Rewarded Soups, is the simplest form of merging, yet successful applications spawned a research line on "task arithmetics" Yadav et al. (2023); Davari & Belilovsky (2024); Yu et al. (2024); Goddard et al. (2025). We test different merging techniques in our experimental setup: from the base molecule model, we derived 2 experts fine-tuned with RLOO on JNK3 and DRD2 respectively; for each merging method, we performed a hyperparameter search for the optimal weight combination $[w_1, w_2] \in [0, 1] \times [0, 1]$ by average task score over $2,048$ generated samples. In Table 1 we observe our policies merged with Model Breadcrumbs Davari & Belilovsky (2024) achieve the highest scores average; we notice that for this solution, a lower

Method	Avg. score	w_1	w_2
breadcrumbs	0.91	0.4	0.6
dare_linear	0.91	0.4	0.6
ties	0.90	0.5	0.5
della	0.90	0.4	0.6
linear	0.89	0.7	0.3
task_arithmetic	0.89	0.7	0.3

Table 1: Merging methods by average task score at optimal interpolation weights for JNK3, DRD2. Whiel Model breadcrumbs and DARE achieve the highest average property score, we do not observe a significant difference from the alternative methods.

weight is assigned to JNK3, which is considered as a hard task. Overall, no significant improvement is observed by varying the merging method, which aligns with the experiments conducted by Yadav et al. (2024).

E.3 Choice of the model size

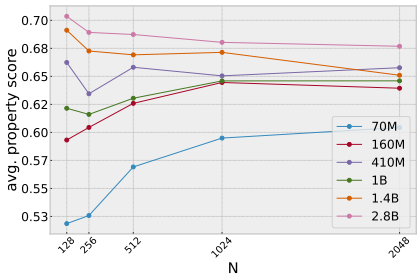

Figure 5: Average task score at increasing N generated samples for different model sizes. We observe diminishing returns past 1B.

We analyze how scaling the model size with fixed compute and data affects sample quality. We choose the Pythia scaling suite Biderman et al. (2023), which offers an exhaustive set of model sizes, and we pre-train all the variants up to 2.8B with our dataset. At test time, we measure the average task score for growing sets of molecules to observe quality over number of generations N. Figure 5 reports comparable performance for all models above 160M of parameters, with diminishing returns past 1B. We choose to fix the default model size to 1B as a tradeoff between model capability and dataset size, in line with Cavanagh et al. (2024); Zholus et al. (2024).

E.4 Analyzing the preference-task correlation

The effectiveness of task arithmetics Ramé et al. (2023); Yadav et al. (2023) is based on the intuition that the merging coefficients $\mathbf{w}_j \in \mathcal{W}$ control the "magnitude" of task directions wrt. a base model, $\Delta\theta = \theta_i - \theta_{\text{base}}$, that is the distribution shift towards a specific objective. In interpolating task directions, we question how directly we can modulate them with defined coefficients. We thus analyze the correlation between preferences $\mathbf{w}_j \in \mathcal{W}$ and measured property scores $\{r\}_{i \in \mathbb{R}^m}$ in merging pairs of policies. We consider 10 task magnitudes per molecule property i, so that $w_{ij} \in [0.1, 1.0]$, $\mathbf{w}_j = [w_{ij}, 1 - w_{ij}]$ for the expert and the base model respectively. A visualization of measured property scores by task magnitude is displayed in Figure 6. We report an average Pearson correlation coefficient of 0.97, indicating a direct relationship between w_{ij} and r_i. We notice "saturation points" varying by property, e.g. JNK3, CYP2C19, approaching values observed at training time. The range of observed rewards not necessarily matches with the provided task magnitudes, e.g. $w_{\texttt{CYP2C19}} \in [0, 1]$ while $r_{\texttt{CYP2C19}} \in [0.4, 0.8]$. We infer that input-output mappings could be derived to better adapt conditioning preferences to each property range, as proposed by Yang et al. (2024)

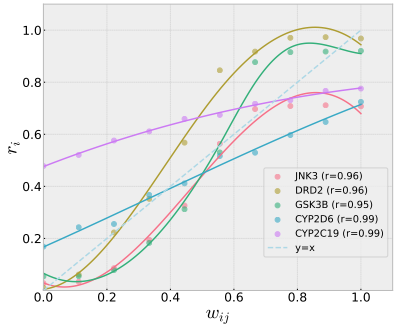

Figure 6: The correlation between task magnitude w_{ij} and r_i is overall positive, with varying value ranges e.g. for JNK3 and GSK3β.

E.5 Out of distribution behavior

To rigorously test whether our models can discover novel high-quality molecules rather than imitate known ones, we create a challenging evaluation setup: we remove all high-scoring molecules ($r_i(x) \geq 0.6$ for any property i) from the pretraining dataset, leaving only ~ 2.12M molecules (58%). In Figure 1 and Table 3, we observe a significant decline in performance for RiC, based on SFT, for which the model does not generate molecules better than the training distribution. Conversely, policies trained with Mol-MoE and RS maintain performances comparable to the in-distribution setup, while with MORLHF, the network optimizes for only a subset of the objectives. We thus infer that RiC, based solely on increasing the likelihood of high-quality training samples, poorly improves out of distribution. Conversely, RS and Mol-MoE successfully discover molecules with scores above 0.6, demonstrating their ability to explore and optimize in regions of chemical space beyond their training distribution.

E.6 Scaling by number of properties

We investigate how model performance scales with the number of target properties. Using identical configurations, we train each method (MORLHF, RS, RIC, Mol-MoE) on progressively larger subsets of properties: $m \in [2, 3, 4, 5]$. For each subset, we evaluate performance by averaging task scores across $2,048$ generated molecules. Additional properties should improve generation quality by providing more guidance to the models. Figure 7 shows that Mol-MoE, RiC and RS successfully leverage additional properties, with Mol-MoE achieving the highest quality. However, MORLHF's performance degrades when optimizing more than three properties ($m > 3$). This indicates that directly optimizing a scalarized reward becomes intractable as the number of properties increases, likely due to compounding complexity and interference between objectives. In contrast, Mol-MoE's architecture separates concerns: experts can stably optimize individual properties while the router manages interference between them.

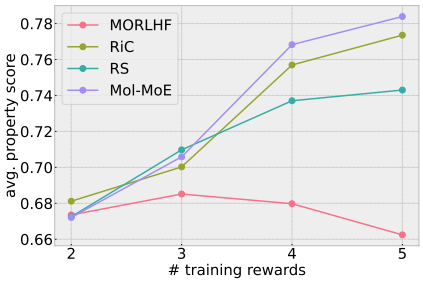

Figure 7: Average score on all mocule properties by number of training objectives by tuning method. **Mol-MoE** benefits the most from increasing reward signals, particularly wrt. RS. Conversely, MORLHF fails beyond three tasks, coherently to what observed in Figure 1.

F ADDITIONAL EXPERIMENTAL RESULTS

Method	JNK3	DRD2	GSK3β	CYP2D6	CYP2C19	Uniqueness	Diversity	Average
MORLHF	0.13	0.76	0.23	**0.89**	0.81	0.94	**0.87**	0.66
Rewards in Context	0.55	**0.85**	0.70	0.87	**0.92**	0.49	0.82	0.74
Rewarded Soups	0.60	0.80	0.73	0.71	0.79	**0.95**	0.84	0.77
MOL-MOE	**0.63**	0.78	**0.75**	0.81	0.73	0.94	0.86	**0.78**

Table 2: Average **in-distribution task scores** by training method, visualized in Figure 1. Overall, MOL-MOE achieves the highest average score, comparably to RS and RiC. RiC generates a significant fraction of repeated samples.

Method	JNK3	DRD2	GSK3β	CYP2D6	CYP2C19	Uniqueness	Diversity	Average
MORLHF	0.22	0.73	0.36	0.79	**0.90**	0.95	**0.87**	0.69
Rewards in Context	0.36	0.69	0.42	0.53	0.57	0.92	0.86	0.62
Rewarded Soups	0.54	0.78	0.69	0.72	0.79	**0.96**	0.86	0.76
MOL-MOE	**0.60**	**0.85**	**0.71**	**0.82**	0.72	0.94	0.86	**0.79**

Table 3: Average **out-distribution task scores** by training method, visualized in Figure 1. Overall, MOL-MOE achieves the highest average score, comparably to RS. RiC considerably underperforms the alternative methods, indicating failure in extrapolating to molecules beyond the training distributions.

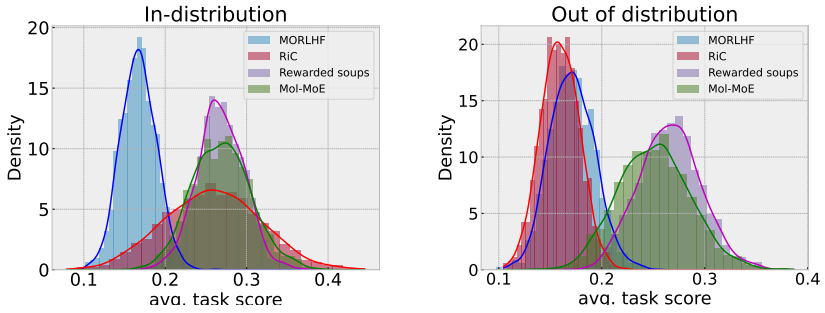

Figure 8: Quality distributions by task score of $2,048$ generated molecules from models trained on the complete dataset (left) or the ablation, held-out best samples (right). **The distribution from MOL-MOE is centered on higher scores than most of the alternatives, with a higher density in the right tail especially out of distribution.** Rewards in Context significantly worsens when high quality molecules are removed at training time.

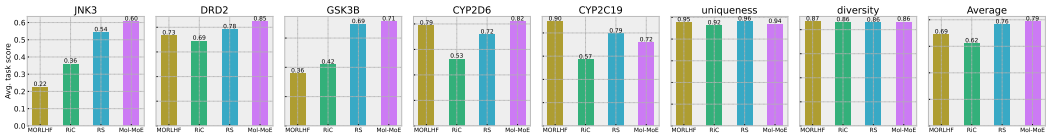

Figure 9: Average property score by tuning method with held-out high quality molecules from the training distribution. **The distribution of molecules obtained with MOL-MOE is comparable or superior in sample quality wrt. alternative methods**. The gap in quality between RiC and RS, MOL-MOE is evident in this setup.

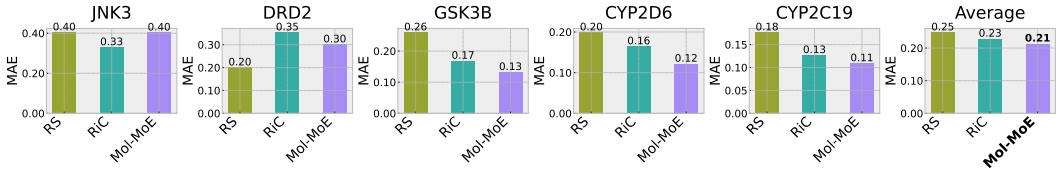

Figure 10: Steerability error measured as Mean Absolute Error (MAE) between the conditioning the measured molecule properties. MOL-MOE is overall more precise than RiC and RS, particularly on `GSK3`β`, CYP2D6, CYP2C19`.