Modeling Molecular Sequences with Learning-Order Autoregressive Models

Zhe Wang [1 2] **Jiaxin Shi** [1] **Michalis K. Titsias** [1] **Nicolas Heess** [1] **Arthur Gretton** [1 2] **Yee Whye Teh** [1 3]

Abstract

Text-based autoregressive models (ARMs) are popular for SMILES (Simplified Molecular Input Line Entry System) string generation due to their simplicity and state-of-the-art performance, but typically use a fixed left-to-right order. Since optimal SMILES ordering is less obvious than for natural text, we developed LO-ARM (Learning-Order ARM) to learn a data-dependent generation order. Evaluated on ChEMBL, LO-ARM learns consistent and meaningful orderings that reveal molecular substructures, and matches or surpasses state-of-the-art models, offering a well-balanced yet competitive model option for practical uses.

1. Introduction

Molecular generation in large chemical spaces has important real-world applications such as in drug discovery and material design. While deep generative models for molecular graphs based on diffusion models (Vignac et al., 2023; Eijkelboom et al., 2024; Jo et al., 2024; Wang et al., 2025) are emerging as a promising solution, SMILES string-based methods (Brown et al., 2019; Irwin et al., 2022; Ross et al., 2022; Schwaller et al., 2019) remain popular in practice. This is because SMILES strings provide an human interpretable representation, lead to algorithms that are computationally efficient (less intensive than handling graph structures) and give state-of-the-art generation performance. Technically, SMILES-based models adopt text-based autoregressive architectures (e.g., Recurrent Neural Networks) and inherit their left-to-right generation ordering. However, unlike text data, for which left-to-right appears to be a natural ordering, SMILES data actually encodes tree-like structures and its natural "canonical" ordering between data dimensions is less obvious. Therefore, it is desirable to consider a variant of autoregressive models (ARMs) that do not treat the ordering as fixed, but rather as a latent random variable that follows a probability distribution that adapts to the evolving state of the generation process.

To address this issue, we apply Learning-Order Autoregressive Models (LO-ARM) (Wang et al., 2025) to model SMILES strings. In short, LO-ARM can flexibly learn probabilistic orderings for generating the data dimensions. Our main contributions and findings include:

- We develop a variant of Learning-Order Autoregressive Models (LO-ARMs) to model molecular sequences (i.e., SMILES strings). In addition, we introduce a novel tokenization algorithm that losslessly compresses parentheses in raw SMILES strings with a prefix notation. We show that such tokenization significantly boosts LO-ARM's performance.
- We evaluate the LO-ARM variant against the GuacaMol benchmark (Brown et al., 2019), and our results match or exceed the performance of the state-of-the-art left-to-right ARM on most of the metrics.
- We find that LO-ARM can learn consistent and human interpretable orderings to generate new molecules with high validity and uniqueness. Moreover, we show that such orderings can reflect meaningful substructures and provide an example to showcase how the the rich signals outputted by LO-ARM, including the learned ordering, can allow for finetuning generated molecules.

2. Background and Related Work

2.1. Modeling SMILES data with ARMs

An autoregressive model (ARM) defines a joint probability distribution over $\boldsymbol{x}$ that factorizes as $p_\theta(\boldsymbol{x}) = \prod_{i=1}^{L} p_\theta(x_i|\boldsymbol{x}_{<i})$, where x_i denotes the i-th dimension of $\boldsymbol{x}$, $\boldsymbol{x}_{<i} = (x_1, \ldots, x_{i-1})$ denotes the first $i-1$ elements of the vector $\boldsymbol{x}$ and $p_\theta(x_i|\boldsymbol{x}_{<i})$ is the conditional distribution with the convention $p_\theta(x_1|\boldsymbol{x}_{<1}) = p_\theta(x_1)$. They are a prominent approach for modeling SMILES (Simplified Molecular Input Line Entry System) strings, which are linear textual representations of non-linear molecular structures. Through directly adopting the well-developed deep learning architectures (e.g, Recurrent Neural Network and Transformer), these models (Brown et al., 2019; Irwin et al., 2022; Ross

[1]Google DeepMind [2]University College London [3]University of Oxford. Correspondence to: Zhe Wang <zhewang@google.com>, Jiaxin Shi <jiaxins@google.com>, Michalis K. Titsias <mtitsias@google.com>.

Proceedings of the Workshop on Generative AI for Biology at the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

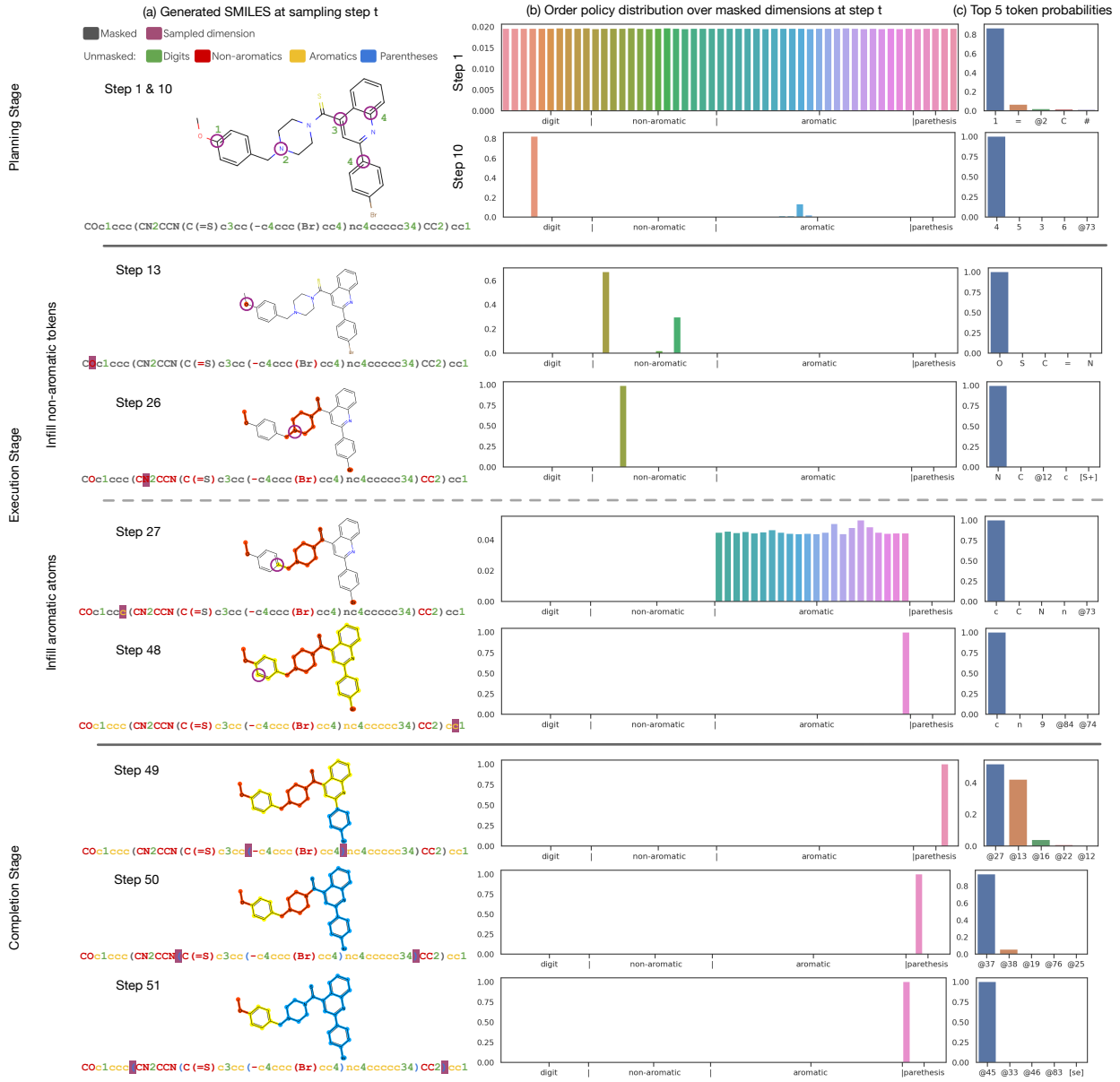

Figure 1. **An example of generating SMILES sample with our best LO-ARM model trained on the GuacaMol dataset**. Our model generates SMILES strings step-by-step, commencing with all dimensions masked (in the figures masked dimensions are colored in grey) and adding token at a time. First, an *order-policy* selects which dimension to fill, and then a *classifier* determines its value. Each step is illustrated in the provided figures: Column (a) illustrate the (partially) generated SMILES string and the corresponding unmasked substructures in the final molecule (highlighted in colors). Columns (b) and (c) provide detailed insights: (b) the order-policy's probability distribution over dimensions, and (c) the classifier's prediction at the selected dimension. Note that, we only display the tokens of top 5 probabilities, and the order-policy is zeroed for unmasked dimensions. To facilitate visualization, we group the dimensions of the generated sample with respect to their dimension/token types: 1) digits (e.g., `1`, `2`), 2) non-aromatic tokens, (e.g., uppercase letters) 3) aromatic tokens (i.e., lowercase letters) and 4) parenthesis pairs. Notably, `@N` represents a pair of parentheses spanning N dimensions between them. The generation proceeds through three phases: 1) Planning (Step 1 to 10): LO-ARM first generates pairs of digits (highlighted in green), which represents ring closures. This step determines the number of rings and estimates their potential connections in the molecule. The digits together with their associated ring-cut atoms in the final sample are highlighted in the first molecule. 2) Execution (Step 13 to 48): The model then infills the molecular structure, characteristically generating non-aromatic tokens (red) before aromatic ones (yellow). 3) Completion (Step 49 to 51): Finally, it generates `@N` parenthesis tokens (blue) to enclose and finalize substructures. A key learned behavior is the consistent preference for enclosing child substructures (smaller parentheses) before their parent structures (larger parentheses). This learned, interpretable ordering is highly consistent: for valid generations containing rings, 93.4% adhere to this overall generation pattern. Furthermore, among valid molecules with multiple pairs of parentheses (whether in flat chains or ring structures), 96.6% generate these parentheses in ascending order of size, from children to parents.

et al., 2022; Schwaller et al., 2019) treat SMILES strings as sequences, akin to sentences in natural language and predict the next character (or token) in the SMILES string with a fixed left-to-right order.

2.2. Learning-Order ARMs

The Learning-Order ARMs (LO-ARM) (Wang et al., 2025) address a fundamental limitation in (ARMs) — the assumption of a fixed generation order, which may not be optimal for complex data types like graphs and images. To do this, through introducing a latent variable z_i to represent the order of token x_i, LO-ARM incorporates a trainable probability distribution that dynamically decides the sampling order of the data dimensions. The log-likelihood of one data point $\boldsymbol{x}$ can be written as $\log p_\theta(\boldsymbol{x}) = \log \sum_{\boldsymbol{z}} p_\theta(\boldsymbol{z}, \boldsymbol{x})$, where $p_\theta(\boldsymbol{z}, \boldsymbol{x}) = \prod_{i=1}^{L} p_\theta(z_i|\boldsymbol{z}_{<i}, \boldsymbol{x}_{\boldsymbol{z}_{<i}}) p_\theta(x_{z_i}|\boldsymbol{x}_{\boldsymbol{z}_{<i}})$. Specifically, $p_\theta(z_i|\boldsymbol{z}_{<i}, \boldsymbol{x}_{\boldsymbol{z}_{<i}})$ is called *order-policy* and $p_\theta(x_{z_i}|\boldsymbol{x}_{\boldsymbol{z}_{<i}})$ is called *classifier*, and both factors depend on parameters θ that we want to learn. Since the exact likelihood is intractable, LO-ARM maximizes an evidence lower bound (ELBO) on the log-likelihood through introducing a *variational order-policy* over $\boldsymbol{z}$ that conditions on the full data vector $\boldsymbol{x}$, and has the general form $q_\theta(\boldsymbol{z}|\boldsymbol{x}) = \prod_{i=1}^{L} q_\theta(z_i|\boldsymbol{z}_{<i}, \boldsymbol{x})$.

Moreover, LO-ARM models the generation process with ARMs as an unmasking process. Specifically, starting with a fully masked state, at each sampling/unmasking step, LO-ARM firstly samples a dimension z_i from the order-policy , and then samples a categorical value x_i from the classifier to fill in z_i. Then, LO-ARM repeats the process until all dimensions are unmasked, which yields a final generated data point $\boldsymbol{x}$. Under this representation LO-ARM also connects with recent masked discrete diffusion models (see, e.g., Shi et al., 2024). However, note that unlike discrete diffusion models we do not specify a forward process in our framework, but only the unmasking or backward process.

In this research, we employ LO-ARM's mathematical framework and adapt it to model SMILES data. We provide more detail of the adaptation in Section 3.

3. Methods

3.1. Data Preprocessing

We preprocess SMILES strings in two main steps. First, we apply standard tokenization using a widely adopted regular expression (Irwin et al., 2022; Schwaller et al., 2019). Second, to address the strict paired-parenthesis constraint in SMILES grammar— a challenge for models without fixed left-to-right ordering (like LO-ARM or diffusion-based methods) which contrasts with simpler handling in autoregressive generation—we represent parenthesis pairs as individual tokens. Specifically, these pairs are formatted as `@N`, where N is the size of the parentheses (the number of tokens between the brackets, including the right parenthesis). Using these new tokens, we then transform the raw SMILES strings into a prefix notation, where each `@N` parenthesis token precedes the substructure or branch it encloses. An example of this transformation is provided below. It's important to note that this prefix transformation for parentheses is bijective and lossless. We provide an ablation analysis on different tokenization algorithms in Appendix C. Following this transformation, we filter out low-frequency tokens (fewer than 100 occurrences) and the corresponding samples containing them. The preprocessed dataset is summarized in Appendix A.

Raw SMILES: `CCOc1ccc(S(=O)(=O)Nc2ccccc2Cl)cc1`
Converted: `CCOc1ccc@20S@3=O@3=ONc2ccccc2Clcc1`

3.2. Model Parameterization

We formulate the model and variational distributions in the same way as (Wang et al., 2025), i.e., $q_\theta(z_i = k|\boldsymbol{z}_{<i}, \boldsymbol{x}) = \frac{e^{g_{\theta,k}(\boldsymbol{x})}}{\sum_{k' \in \boldsymbol{z}_{\geq i}} e^{g_{\theta,k'}(\boldsymbol{x})}}$, $p_\theta(x_{z_i=k}|\boldsymbol{x}_{\boldsymbol{z}_{<i}}) = \text{softmax}(f_{\theta,k}(\bar{\boldsymbol{x}}_{\boldsymbol{z}_{<i}}))$, and $p_\theta(z_i = k|\boldsymbol{z}_{<i}, \boldsymbol{x}_{\boldsymbol{z}_{<i}}) = \frac{e^{h_{\theta,k}(\bar{\boldsymbol{x}}_{\boldsymbol{z}_{<i}})}}{\sum_{k' \in \boldsymbol{z}_{\geq i}} e^{h_{\theta,k'}(\bar{\boldsymbol{x}}_{\boldsymbol{z}_{<i}})}}$, where $i \in [1, 2 \dots L]$ is the i^{th} sampling step, and $k \in [1, 2 \dots L]$ is a masked dimension at this step.

For modeling with SMILES sequences, we parameterize g_θ, f_θ, and h_θ with Transformers. Specifically, we collocate the classifier f_θ and order-policy h_θ through sharing a torso and parametrize them with two output heads, and parameterize g_θ with a separate Transformer.

4. Evaluation and Analysis

Table 1. Molecule generation on GuacaMol SMILES dataset. We directly cite the results of other methods on the following metrics: **V**alidity, **U**niqueness, **N**ovelty, and FCD. The metrics are calculated with the generated samples with the corresponding methods. In particular, the random sampler uniformly samples the test set.

Method	**V.%↑**	**U.%↑**	**N.%↑**	**FCD↓**
Random sampler	100.0	99.7	0.0	0.368
AAE	82.2	100.0	**99.8**	3.18
VAE	87.0	99.9	97.4	0.737
LSTM	95.9	100.0	91.2	0.455
Our Results				
AO-ARM	63.3	99.8	99.4	1.63
FO-ARM	91.8	100.0	88.3	**0.384**
LO-ARM	92.6	100.0	95.3	1.15

As SMILES data does not exhibit a canonical ordering, we are interested in the following questions: 1) whether LO-

ARM can learn a different ordering from the prescribed left-to-right ordering, which is fully data-dependent and human-interpretable, 2) whether the learned ordering can yield better generation performance. To answer these, we evaluate LO-ARM on the GuacaMol/ChEMBL dataset (Brown et al., 2019). For each model variant, we generate 16384 samples and evaluate the generated molecules based on two key aspects: 1) Validity and Uniqueness: Assessing individual molecules for chemical correctness and whether they are distinct. 2) Novelty and Frèchet ChemNet Distance (FCD): Novelty is the proportion of generated molecules not found in the training set. Lower novelty suggests the model might be overfitting. FCD measures the similarity between the distributions of generated and real molecules using ChemNet activations. It's important to note that novelty and FCD can be negatively correlated. For instance, if generated molecules are very similar to the training data (resulting in a lower FCD), they are also less likely to be novel. Furthermore, these two categories of metrics—individual molecule properties (validity, uniqueness) versus distributional properties (novelty, FCD)—are not directly linked. For example, an invalid molecule could still produce latent activations close to those of valid molecules, impacting FCD. Therefore, a generative model of a balanced performance across all these evaluation metrics would still be useful in practice.

To evaluate the order policy, we introduce two baselines, i.e., 1) AO-ARM (Any-Order) in which both the variational (q_θ) and model (p_θ) order policies are set to uniform, in which the order policies always uniformly select among the masked dimensions. 2) FO-ARM (Fixed-Order), which is a standard ARM with fixed left-to-right generation ordering. We implement FO-ARM with Transformer (Vaswani et al., 2017) to reproduce the results from the prior LSTM model (Brown et al., 2019) and to provide a fair baseline for LO-ARM, which is also Transformer-based.

First, data in Table 1 reveals that both FO-ARM and LO-ARM outperform AO-ARM on FCD, emphasizing that an ordering strategy (prescribed or data-dependent) is crucial for generating SMILES sequences. FO-ARM, similar to LSTM, achieves the best FCD but shows low novelty (88.3%), suggesting a tendency to overfit, a pattern also seen in LSTM. Moreover, no single model excels across all metrics. For instance, VAEs offer decent novelty and FCD but struggle with validity, potentially requiring more computational effort to sample valid molecules. Conversely, LSTMs generate more valid molecules but tend to repeat training data, which can hinder new drug discovery. Amidst these trade-offs, LO-ARM stands out as a well-balanced option for practical applications.

Next, LO-ARM learns a consistent and human-interpretable generation order without needing specific inductive biases, as illustrated in Figure 1. The typical learned process is: 1) Estimate molecular structure (rings and their connections) by first generating digit tokens which indicate ring enclosures and cuts. 2) Infill this structure, prioritizing non-aromatic tokens before aromatic ones. 3) Complete the molecule by generating parentheses to finalize substructures, ordered from child structures (smaller parentheses) to parent structures (larger parentheses). This interpretable ordering shows high consistency. For valid generations containing rings, 93.4% follow this overall generation pattern. Furthermore, for valid SMILES strings with multiple parenthesis pairs, including side chains (e.g., `(=O)`), 96.6% generate these parentheses in ascending order of size (from children to parents).

Thirdly, we demonstrate that the learned ordering is largely driven by certainty at masked dimensions. To do this, for each sample's generation trajectory, we calculated per-step correlation coefficients between the order policy probabilities and the classifier entropy (our certainty measure) over all masked dimensions. We then performed one-sample t-tests on each sequence to obtain a mean and a significance level. We found that 85.4% of samples exhibited a negative mean correlation, and of these, 85.6% were statistically significant ($p < 0.05$). Furthermore, 88.9% of all samples had $p < 0.10$. This confirms a significant negative correlation, meaning the order policy prioritizes dimensions with higher certainty (i.e., lower entropy).

Finally, LO-ARM's rich signals and human-interpretable ordering directly enable the fine-tuning of generated molecular structures by allowing researchers to act on its detailed feedback. As shown in Figure 1, at Step 49, the model started to generate large paretheses to finalize the substructures. It sampled one option (i.e.., parenthesis pair `@13`) in reality while also indicating high confidence in an alternative (i.e., `@27`) that could also form a valid molecule. If the model had pursued this alternative, following the learned ordering principles (i.e., generating parentheses from small to large), it would likely produce a different yet valid molecule, as illustrated in our counterfactual example Figure 2. Such counterfactual investigation is impractical with fixed-order autoregressive models, which would require extensive rollouts of all subsequent tokens and would change the generated atoms completely.

5. Conclusion

We have developed a LO-ARM variant for modeling SMILES data, incorporating a novel tokenization algorithm. Evaluated on the ChEMBL dataset, LO-ARM achieves competitive results across most metrics. Notably, it learns a consistent and human-interpretable ordering for generating new molecules without requiring inductive bias, which in turn can facilitate the fine-tuning of generated samples. However, a gap in FCD persists when compared to state-

of-the-art fixed-order ARMs, an area we've identified for future improvement.

References

Brown, N., Fiscato, M., Segler, M. H., and Vaucher, A. C. Guacamol: Benchmarking models for de novo molecular design. *Journal of Chemical Information and Modeling*, 59(3):1096–1108, 2019. doi: 10.1021/acs.jcim.8b00839. URL https://doi.org/10.1021/acs.jcim.8b00839. PMID: 30887799.

Eijkelboom, F., Bartosh, G., Naesseth, C. A., Welling, M., and van de Meent, J.-W. Variational flow matching for graph generation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Irwin, R., Dimitriadis, S., He, J., and Bjerrum, E. J. Chemformer: a pre-trained transformer for computational chemistry. *Machine Learning: Science and Technology*, 3(1):015022, jan 2022. doi: 10.1088/2632-2153/ac3ffb. URL https://dx.doi.org/10.1088/2632-2153/ac3ffb.

Jo, J., Kim, D., and Hwang, S. J. Graph generation with diffusion mixture, 2024. URL https://arxiv.org/abs/2302.03596.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Ross, J., Belgodere, B., Chenthamarakshan, V., Padhi, I., Mroueh, Y., and Das, P. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence*, 4(12):1256–1264, 2022. doi: 10.1038/s42256-022-00580-7.

Schwaller, P., Laino, T., Gaudin, T., Bolgar, P., Hunter, C. A., Bekas, C., and Lee, A. A. Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction. *ACS central science*, 5(9):1572–1583, 2019.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. K. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, 2024.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. Digress: Discrete denoising diffusion for graph generation. In *International Conference on Learning Representations*, 2023.

Wang, Z., Shi, J., Heess, N., Gretton, A., and Titsias, M. K. Learning-order autoregressive models with application to molecular graph generation, 2025. URL https://arxiv.org/abs/2503.05979.

A. Dataset Summary

We tokenized the ChEMBL dataset with the tokenization algorithm described in Section 3. Specifically, we augment the vocabulary with parenthesis tokens. Moreover, we filtered out the low-frequency tokens (fewer than 100 occurrences) together with the samples containing them. The preprocessed dataset is summarized in Table 2. After filtering, the vocabulary size is almost halved while the dataset remains the same scale, only fewer than 1000 samples were filtered out.

Table 2. Dataset statistics before and after filtering. Both cases use the augmented vocabulary and transform SMILES strsings with prefix notation described in Section 3.

	#training samples	#validation samples	#test samples	Vocabulary size
Raw dataset	1273114	79568	238706	203
Preprocessed	1272277	79506	238538	129

B. A case study of counterfactual finetuning

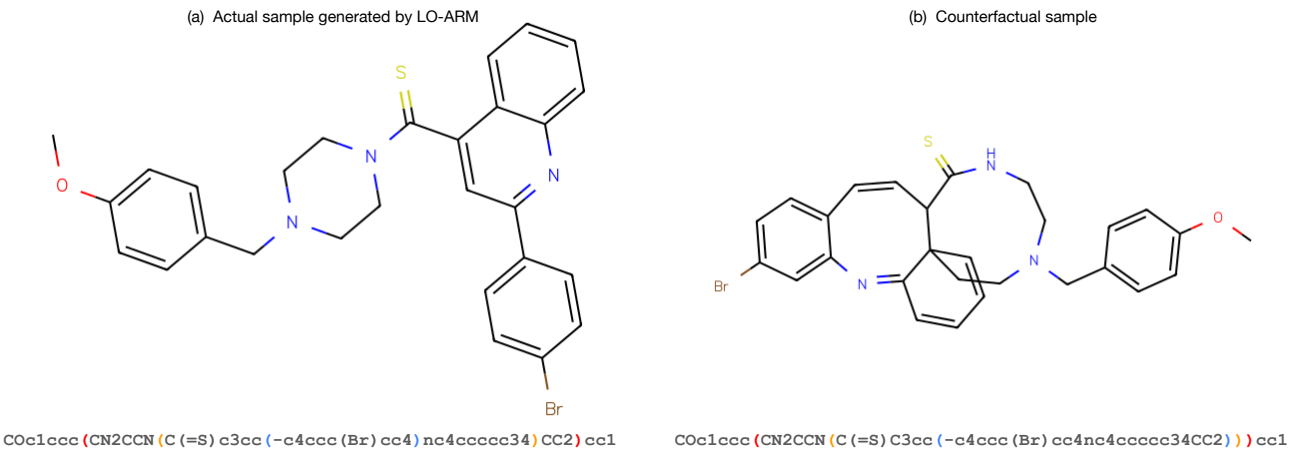

Figure 2. **An example of counterfactual finetuning for generated samples through leveraging the learned ordering**.

As we can see in Figure 1, at Step 49, the model started to complete the molecule by finalizing its substructures. Although it sampled pair of parentheses `@13` to produce molecule (a), the model also assigned high probability to an alternative pair of parentheses, `@27`, suggesting its potential to form another valid molecule. Indeed, had `@27` been selected, following the established generation ordering (i.e., forming smaller parentheses before larger ones), a structurally different yet valid molecule (b) would likely have emerged. As illustrated, molecules (a) and (b) are largely identical in their SMILES strings, differing primarily in the size of their final large parentheses (highlighted). This ability to identify high-probability alternatives during substructure completion enables a "counterfactual fine-tuning." Researchers can intervene at this stage, investigating feasible high-probability options provided by the classifier to explore slight structural variations. Such targeted exploration is impractical with standard left-to-right ARMs, where altering one token typically requires re-generating all subsequent tokens, often completely changing the molecule's core atomic structure.

C. Ablation Analysis

We conducted an ablation study to compare two tokenization algorithms:

- Standard Tokenization: Parentheses are treated as individual tokens. This results in a vocabulary size of 109 after

filtering.

- Augmented Tokenization: Pairs of parentheses are represented as single tokens. This leads to a vocabulary size of 129 after filtering.

As shown in Table 3, the FCD results indicate that while FO-ARM demonstrates robustness across both tokenization methods, augmented tokenization substantially improves LO-ARM's performance. Conversely, standard tokenization achieves higher validity scores compared to augmented tokenization. This suggests that a simpler vocabulary may facilitate the generation of valid molecules. The augmented tokenization in FO-ARM also improves its performance in FCD. This is likely because standard tokenization forces the model to track open parentheses, which complicates the prediction task. Generating parentheses as matched pairs, however, allows the model to avoid this issue entirely.

Table 3. Ablation study on the standard and augmented tokenization algorithms

.

Method	**Tokenization**	**Validity%↑**	**Uniqueness%↑**	**Novelty%↑**	**FCD↓**
FO-ARM	Standard	98.3	100.0	81.5	0.430
	Augmented	91.8	100.0	88.3	0.384
LO-ARM	Standard	94.2	99.7	96.0	5.03
	Augmented	92.6	100.0	95.3	1.15

D. Experiment Setup

The Transformer architecture is adopted from the `llama2.c` project[1]. Both the FO-ARM model and the classifier in LO-ARM consist of 9 attention layers. The variational order-policy used in LO-ARM only has 1 attention layer. Moreover, We report the hyperparameters in Table 4. All experiments were run until convergence.

Table 4. Hyperparameter setup.

Hyperparameter	ChEMBL/GuacaMol
Optimizer	AdamW
Scheduler	Cosine Annealing
Learning Rate	$5 \cdot 10^{-5}$
Weight Decay	$1 \cdot 10^{-12}$
EMA	0.9999

E. Sample Gallery

[1]https://github.com/karpathy/llama2.c

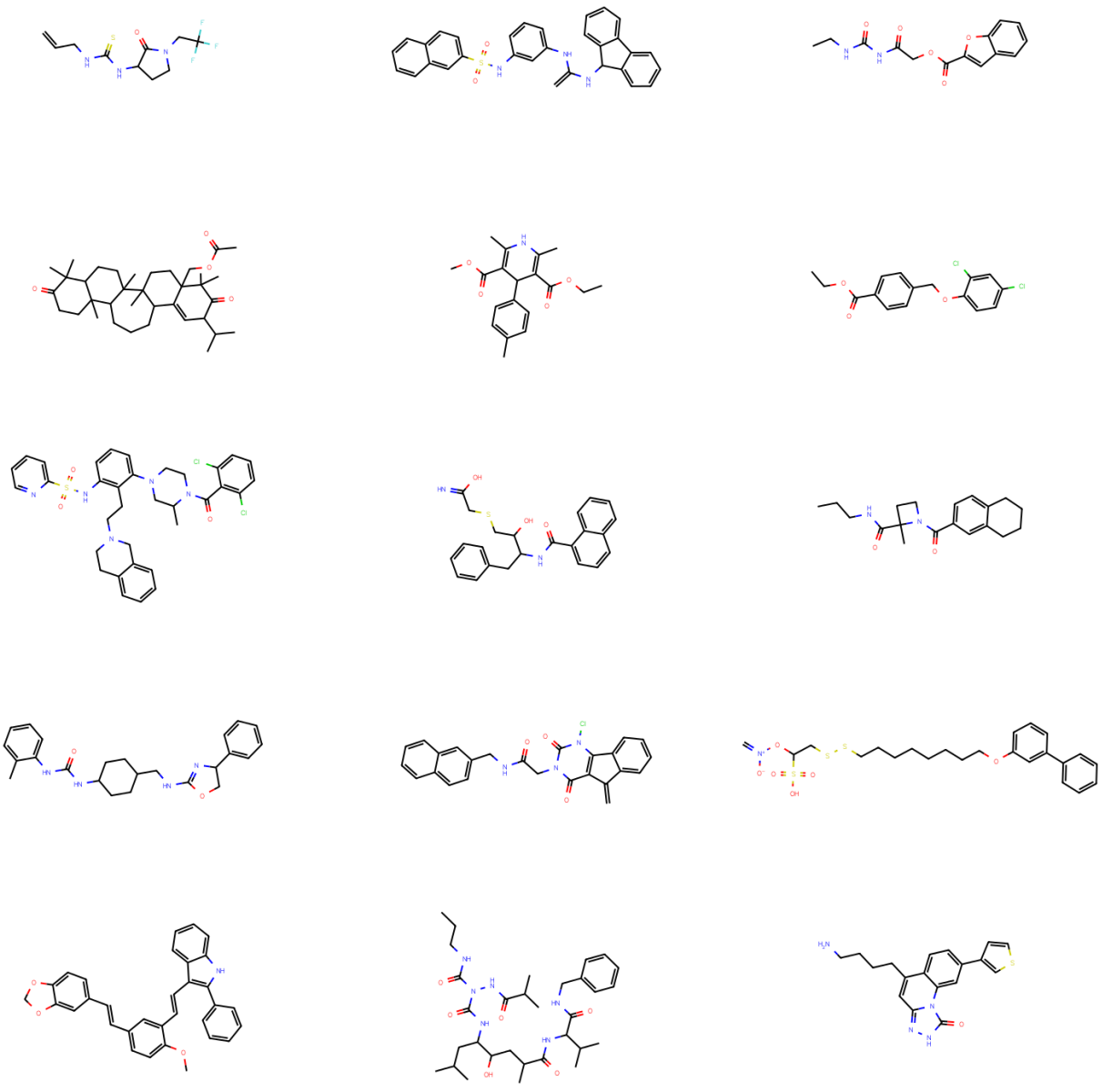

Figure 3. **Samples generated by our best LO-ARM**.