Toward Better Geometric Representations for Molecule Generative Models

Shaoheng Yan[1,2,*] **Zian Li**[1,3,*] **Cai Zhou**[5] **Qiaojing Huang**[2,4]

Kai Liu[2,4] **Muhan Zhang**[1,6,†]

[1]Institute for Artificial Intelligence, PKU, [2]ByteDance AI Drug Discovery
[3]School of Intelligence Science and Technology, PKU, [4]Anew Labs
[5]Department of Electrical Engineering and Computer Science, MIT
[6]State Key Laboratory of General Artificial Intelligence, PKU

Abstract

Geometric representation-conditioned generation decouples molecule representation modeling from 3D structure generation: a molecule-level representation is generated first, and a 3D molecule is then generated conditioned on it, which significantly improves both efficiency and quality. However, its effectiveness is fundamentally bounded by the quality of the representation space: representations from pretrained molecular encoders such as UniMol are non-smooth, suffer from dimensional collapse, and are not fully exploited during generative training. In this work, we propose LENSEs, a framework that better exploits molecule representations in representation-conditioned generation. LENSEs introduces three complementary mechanisms: (1) a *representation head*, trained jointly with the generator, that aggregates multi-level features from the pretrained encoder; (2) a *molecule perceptual loss* that optimizes the generator in a semantically informative representation space; and (3) a *node-level representation alignment (REPA) loss* that aligns the generator's hidden states with encoder representations, narrowing the semantic gap between pretraining and generation. On the challenging GEOM-DRUG benchmark, LENSEs achieves 97.28% validity and 98.51% molecule stability, surpassing existing advanced methods. Beyond generation quality, the refined representations are substantially richer: the effective rank increases by 1.35× on GEOM-DRUG. With the encoder and representation head frozen, the refined representations also outperform the raw encoder features on *all* QM9 properties and MoleculeNet datasets, establishing generative training with alignment objectives as a promising *pretraining paradigm* for molecular encoders.

1 Introduction

3D molecule generation has emerged as a cornerstone of modern drug design [1–3], evolving from language-model-based string generation ([4–6]) toward direct 3D conformation generation [7, 8]. As the molecules being modeled grow increasingly large and complex, generation methods must capture fine-grained geometric and topological features with high precision [9, 10]. Inspired by effective methods in image domains such as Diffusion [11, 12] and Flow Matching [13], advanced generative methods have been proposed for molecule generation [14–24].

* Equally contributed.
† Corresponding to Muhan Zhang <muhan@pku.edu.cn>, Kai Liu <liukai0824@bytedance.com> and Qiaojing Huang <huangqiaojing@bytedance.com>.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

A promising recent paradigm for molecular generation is *geometric representation-conditioned generation* (GeoRCG) [25], which decouples molecular representation modeling from generator training. Instead of directly generating a 3D molecule with an equivariant backbone [7], GeoRCG first samples a molecular representation, i.e., a molecule-level vector, using a Representation Diffusion Model (RDM) [26, 24], and then generates the 3D molecular structure conditioned on this representation. The representation space is defined by a pretrained encoder, such as UniMol [27] or Frad [28]. During training, the encoder extracts molecular representations, which are then used to train both the RDM and the representation-conditioned molecular generator in a self-conditioning framework [29]. As shown in [25], this design substantially improves both sample quality and generation efficiency.

However, the success of this paradigm is fundamentally bounded by the quality of the molecule representations [25, 30, 31]. We identify two concrete manifestations of this bottleneck. First, **a collapsed and non-smooth representation manifold**. Current pretrained molecular encoders concentrate their molecule-level representations in a low-dimensional subspace (Table 3). Such a degenerate geometry leaves little room for the RDM to place distinguishable molecules, and small conformational perturbations can still induce disproportionately large latent shifts, forcing prior work [25] to inject *hand-tuned noise* during training as a compensatory mechanism. Moreover, due to the lack of generative pretraining, these encoder are optimized primarily for property prediction rather than generation, which may lack enough semantic guidance for generation tasks.

Second, **underexploited multi-level structural semantics**. We observe that pretrained molecular encoders exhibit hierarchical features: shallow layers capture local structural motifs such as functional groups, while deeper layers encode more global molecular semantics, analogously to hierarchical representations in computer vision [32, 33]. However, existing representation-conditioned generation pipelines use the encoder mainly as a condition provider, without explicitly transferring this hierarchical structured knowledge to the generator. As a result, the generator must relearn semantics from scratch, increasing optimization difficulty and cost.

Motivated by these observations, we develop a mechanism that allows the representation space to be optimized during molecule generator training, so that it can guide molecular generation in a smoother and higher-quality manner, while also transferring the encoder's hierarchical molecular semantics to the generator for better training. To this end, we propose **L**atent **E**nhancement for **N**on-smooth **S**tructural **E**ncoding**s** (**LENSEs**), a framework that systematically improves the conditioning latent space for geometric representation-conditioned molecule generation. Rather than passively relying on a frozen pretrained encoder, LENSEs explicitly refines the representation space during generator training through three mechanisms: (1) a representation head that maps the encoder's multi-layer features into a variationally regularized latent space to improve smoothness and sampleability; (2) a molecule-specific perceptual loss [34] that preserves multi-scale semantic information in the encoder feature space, encouraging the generated molecules to match meaningful structural patterns; and (3) a representation alignment (REPA) loss [35] that directly aligns the generator's internal representations with the encoder's higher-level features, thereby reducing the generator's learning burden and accelerating training.

We validate our approach through complementary experiments: (i) LENSEs improves over the direct GeoRCG baseline [25] on GEOM-DRUG generation and outperforms recent competitive approaches such as CanonFlow [36], establishing state-of-the-art performance in both structural and physical plausibility metrics; (ii) Lipschitz constant analysis reveals that the refined representation space is significantly smoother than that of the raw encoder; (iii) downstream QM9 regression with the refined representation outperforms the pretrained encoder output, demonstrating that our generative training pipeline also serves as an effective pretraining paradigm.

2 Related Work

3D Molecule Generation with Equivariant Models. Generating 3D molecules requires models that respect geometric symmetries. EDM [7] established a key paradigm by jointly modeling atom coordinates and atom types with an equivariant diffusion framework. GCDM [37] improved structural plausibility by incorporating local geometric features such as bond angles and torsions, while MiDi [38] improved chemical validity through mixed discrete-continuous diffusion. To improve efficiency, EquiFM [39] replaced diffusion simulation with flow matching [13], and geometric optimal transport

methods [40] further optimized transport paths. Other structure-based drug design methods include TargetDiff [41] and DiffSBDD [42], which generate molecules conditioned on 3D protein pockets.

Latent Diffusion and Representation-Conditioned Generation. To avoid the high cost of direct coordinate-space generation, latent diffusion methods [43] generate molecules in compressed latent spaces. GeoLDM [44] pioneered this idea by learning an equivariant autoencoder that maps molecular geometries into continuous latent codes, reducing dimensionality while preserving equivariance. GeoRCG [25] further introduced representation-conditioned generation, where a pretrained geometric encoder provides property-rich representations, a lightweight diffusion model captures their distribution, and a generator reconstructs structures conditioned on them. Despite their efficiency, these methods share a key limitation: they assume the encoder latent space is naturally suitable for generation. However, encoders trained for property prediction often induce semantically disorganized and non-smooth latent spaces, leaving a fundamental bottleneck that prior work has mainly treated with ad hoc fixes such as noise injection.

Perceptual Losses and Representation Alignment. In the image domain, perceptual loss is often preferred over low-level pixel-wise objectives because it aligns more closely with human perception [45]. It has been shown to be effective in a range of optimization settings [43, 46, 47], where directly optimizing pixel loss can lead to undesirable blurriness. REPA [35] and REED [48] further brought this idea to image diffusion by aligning diffusion hidden states with a pretrained vision encoder. In molecular generation, geometry-complete methods [37] suggested that richer structural supervision also improves plausibility, while REED [48] aligned the internal features of molecular diffusion models with external molecule-level pretrained representations to accelerate and improve generative training. Our work systematically leverages representations for both *conditioning* and *alignment* to improve 3D molecular generation, establishing a bidirectional relationship between generation quality and representation quality.

3 LENSEs: Latent Enhancement for Non-smooth Structural Encodings

3.1 Observations

Degenerate latent geometry of the pretrained encoder. Current pretrained encoders are primarily trained on coordinate denoising tasks [49, 28, 27]. Specifically, for a given molecule, random noise is added to its 3D coordinates, which are then input to the encoder, whose objective is to denoise the molecule and recover its original stable conformation. As shown in [49], this paradigm is analogous to learning a physical force field, with denoising corresponding to the conservative case.

Despite its downstream effectiveness, atom-level pretraining can yield pooled molecular representations with limited effective dimensionality. On GEOM-DRUG, raw UniMol representations exhibit an effective rank of only 65.32 (Table 3) and tightly concentrated intermolecular distances (Figure 5), suggesting limited variation along many latent directions. This degenerate geometry leaves the RDM with little room to place distinguishable molecules apart. The same representations are also locally unstable—small coordinate perturbations can induce large shifts in the latent space, as reflected by the large Lipschitz constants in Table 3, a phenomenon also observed in [30, 50]. Both properties are detrimental to representation-conditioned generative tasks. To mitigate this issue, GeoRCG [25] improves the representation–molecule connection learned by the conditional generator by adding random, hand-crafted channel-uniform noise to the encoder's raw representations during training.

Semantic Information in Shallow Encoder Layers. The GeoRCG framework leverages the pretrained encoder only as a representation-space definer, limiting its role to providing molecule-level representations. However, as shown in [45, 35], the intermediate features of deep pretrained networks can capture richer semantic information, providing stronger semantic similarity between entities and aligning well with the generator's learned intermediate features.

To investigate this in the molecular domain, we conduct an additional probing experiment to examine whether shallow encoder layers preserve meaningful chemical semantics. Specifically, we use Frad [28] to perform a systematic layer-wise probe analysis on 30,000 randomly selected QM9 [51] molecules. We train a linear probe on the frozen representation of each layer to classify ten common molecule functional groups, including hydroxyl, amino, and carbonyl. The goal is to assess whether such information is linearly accessible from each layer's representation.

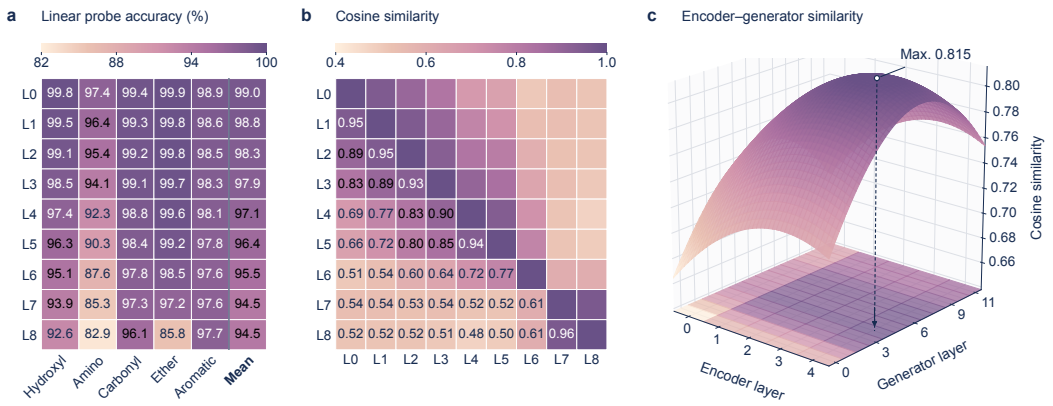


Figure 1: **Layer-wise representation analysis.** (a) Linear probe accuracy for selected functional groups and the overall mean across nine Frad encoder layers on QM9. (b) Cross-layer cosine similarity of encoder representations. (c) Node-level cosine similarity between generator and encoder layers, with projections onto the layer plane. Darker colors indicate higher values within each panel.

As shown in Figure 1a, with full results in Table 6, first-layer probes achieve the highest classification accuracy across all ten functional groups, with gaps up to 14% for groups such as amino and ether. To understand this trend, we visualize layer-wise latent similarity in Figure 1b, which reveals a clear two-block structure: the first six layers are mutually similar, while the last two form a separate block. This suggests that final-layer conditioning may discard semantic signals preserved in earlier layers.

This observation is further supported by the learned layer weights used for generation: instead of assigning all weight to the final-layer representation, the model allocates nearly 20% of the total weight to earlier encoder layers. Together, these results show that early encoder layers encode motif semantics that should be preserved and exploited during generation.

Alignment between Encoder and Generator Features. Beyond encoder-side analysis, we further observe from the public checkpoint of prior work [25] that the layer-wise representations of the molecular generator are highly similar to those of the pretrained encoder. The detailed grid-wise comparison is shown in Figure 1c. Intuitively, this indicates that the two models share common structural patterns and encoded knowledge. The similarity between representations of generators and perceptual models has also been observed in computer vision [35]. This suggests that the generator is not learning from scratch, but instead tends to recover part of the encoder's representational structure during training. Motivated by this observation, we explicitly align the representations of the pretrained encoder and the generator through a projector, thereby accelerating and improving the generator's acquisition of chemically meaningful structural knowledge.

3.2 Unlocking Potentials of Molecule Representations

In this section, we present LENSEs, an enhanced framework for equivariant molecule generation conditioned on geometric representations. It introduces three components: (1) a representation head over the pretrained encoder latent space to improve representation quality; (2) a perceptual loss ($\mathcal{L}_{\text{PERC}}$) for semantic supervision in encoder space; and (3) a representation projection alignment loss ($\mathcal{L}_{\text{REPA}}$) to align generator and encoder features.

Notations. A 3D molecule is represented as $\mathcal{M} = (\boldsymbol{x}, \boldsymbol{a}, \mathcal{E})$, where $\boldsymbol{x} \in \mathbb{R}^{N\times 3}$ denotes atomic coordinates, $\boldsymbol{a}$ encodes atom types, and $\mathcal{E}$ encodes chemical bonds. The goal is to learn an equivariant generative distribution $p_\theta(\mathcal{M})$ satisfying SE(3)-equivariance. Given a molecule $\mathcal{M}$, a pretrained encoder f_{enc} extracts a geometric representation:

$$\boldsymbol{h} = f_{\text{enc}}(\mathcal{M}) \in \mathbb{R}^D. \tag{1}$$

Representation Space Trained with Generative Tasks. The raw output of the pretrained encoder is a deterministic function of the input molecule, and its latent geometry is not explicitly optimized during generative training. To obtain a more structured and generation-friendly latent space, we introduce a representation head on top of the multi-layer encoder representations, enabling adaptive

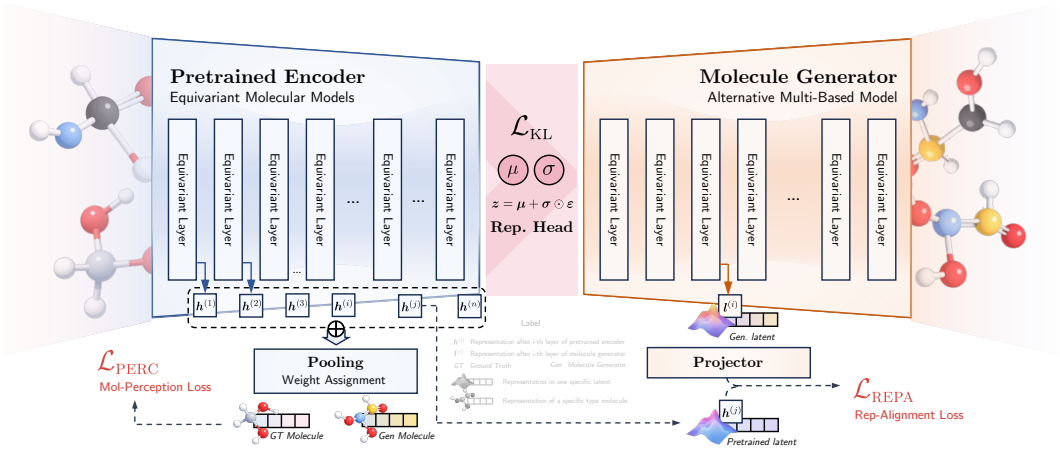


Figure 2: **The framework of LENSEs. Phase I:** the molecule generator and the representation head are trained based on a pretrained encoder. The representation head takes as input a learnable pooling of representations from different encoder layers, while the molecule generator performs conditional generation using the reparameterized latent produced by the representation head. **Phase II:** the RDM is trained to model the latent distribution of target molecules in the mean space of the representation head, thereby serving as a representation generator. **Phase III:** molecule generation is carried out using the trained RDM together with the molecule generator.

aggregation of different level semantics. Specifically, let $\boldsymbol{H}^{(l)}(\mathcal{M}) \in \mathbb{R}^d$ denote the representation of molecule $\mathcal{M}$ extracted from the l-th encoder layer, where $l = 1, \ldots, L$ and L is the number of encoder layers. To exploit information from different representation depths, we aggregate these layer-wise graph representations through learnable layer pooling:

$$\boldsymbol{g}(\mathcal{M}) = \sum_{l=1}^{L} \alpha_l \boldsymbol{H}^{(l)}(\mathcal{M}), \qquad \alpha_l = \textsc{Softmax}(\boldsymbol{w})_l, \tag{2}$$

where $\boldsymbol{g}(\mathcal{M}) \in \mathbb{R}^d$ is the pooled graph representation, $\boldsymbol{w} \in \mathbb{R}^L$ denotes the learnable layer-weight logits, and α_l is the normalized importance weight assigned to the l-th encoder layer.

The pooled representation $\boldsymbol{g}(\mathcal{M})$ is then mapped to the parameters of a diagonal Gaussian posterior:

$$\boldsymbol{\mu} = f_\mu(\boldsymbol{g}(\mathcal{M})), \qquad \log \boldsymbol{\sigma}^2 = \text{clamp}\big(f_{\log \sigma^2}(\boldsymbol{g}(\mathcal{M})),\, v_{\min},\, v_{\max}\big)\,, \tag{3}$$

where f_μ and $f_{\log \sigma^2}$ are two learnable projection heads that predict the mean and log-variance, respectively. The clamp operation constrains the predicted log-variance within $[v_{\min}, v_{\max}]$ to improve numerical stability. Finally, we sample the latent code using the reparameterization trick [52]:

$$\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \tag{4}$$

where $\odot$ denotes element-wise multiplication, $\boldsymbol{\sigma}$ is obtained from the predicted log-variance, and $\boldsymbol{z}$ is the stochastic latent representation used for generation. The representation head is optimized using two loss sources: 1) Generator Loss: During representation-conditioned generator training, the generator attempts to predict the clean molecule from its noisy state [7, 25], conditioned on the specific molecule's representation $\boldsymbol{z}$ extracted above. For this purpose, the representation head must learn to extract the most meaningful molecule representation for generative tasks, thereby enabling targeted refinement of the final representation and providing a better representation space. 2) KL-Loss $\mathcal{L}_{\text{KL}}$ [52], which enforces μ and σ to follow a normal distribution, leading to a more compact representation space. This process resembles the training of a Variational Autoencoder (VAE) [52], where the latent space, in our case, corresponds to the molecule representation itself, and the decoded output is the 3D molecular structure.

Molecule Perceptual Loss. Inspired by perceptual losses in image generation [45] and the semantic information contained in the shallow layers of the encoder, as discussed in Section 3, we introduce a high-level perceptual representation loss $\mathcal{L}_{\text{PERC}}$. Instead of solely minimizing the low-level mean

squared loss between the denoised and clean molecule coordinates, this loss aligns the representation of the denoised molecule with that of the clean molecule. Specifically, during training, both the denoised molecule $\hat{\mathcal{M}}$ and the clean molecule $\mathcal{M}$ are passed through the frozen pretrained encoder to extract graph-level representations. The perceptual loss is defined as

$$\mathcal{L}_{\text{PERC}} = w(t) \left\| \text{sg}(\boldsymbol{g}(\mathcal{M})) - \boldsymbol{g}(\hat{\mathcal{M}}) \right\|_2^2, \tag{5}$$

where $\boldsymbol{g}(\cdot)$ denotes the pooled graph-level representation extracted by the frozen encoder, $\text{sg}(\cdot)$ denotes the stop-gradient operation, and $w(t)$ is a time-dependent weighting function. The stop-gradient operation treats the clean-molecule representation $\boldsymbol{g}(\mathcal{M})$ as a fixed semantic target, while allowing gradients to flow through $\boldsymbol{g}(\hat{\mathcal{M}})$ to optimize the generator. Although the encoder parameters are frozen, gradients can still be backpropagated through the encoder computation graph to the generated molecular coordinates and atom-type logits. Since $\hat{\mathcal{M}}$ also contains discrete atom types, which cannot be directly optimized by standard backpropagation, we employ a straight-through estimator [53] to approximate gradients through this discrete path. Finally, $w(t)$ is implemented as a cosine schedule that assigns larger weights to later timesteps, where the generated structure is closer to clean data and the encoder representations provide a more informative alignment signal.

Representation Alignment between Generator and Encoder. To further leverage the encoder beyond graph-level conditioning, motivated by the preceding analysis in Section 3, we introduce a representation projection alignment loss ($\mathcal{L}_{\text{REPA}}$) to align generator features with pretrained encoder representations at the node level. Let $\boldsymbol{H}_{\text{enc}}^{(l_e)} \in \mathbb{R}^{N \times d_e}$ and $\boldsymbol{H}_{\text{gen}}^{(l_g)} \in \mathbb{R}^{N \times d_g}$ denote node features from the l_e-th encoder layer and l_g-th generator layer, respectively. To reconcile the differences in feature dimensions and scales, we employ a learnable projection $\text{PROJ} : \mathbb{R}^{d_g} \to \mathbb{R}^{d_e}$ to map generator features into the encoder space, and maximize their cosine similarity over valid atoms $\mathcal{A}$:

$$\mathcal{L}_{\text{REPA}} = -\frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \left\langle \frac{\text{PROJ}(\boldsymbol{H}_{\text{gen},i}^{(l_g)})}{\left\| \text{PROJ}(\boldsymbol{H}_{\text{gen},i}^{(l_g)}) \right\|_2}, \frac{\text{sg}(\boldsymbol{H}_{\text{enc},i}^{(l_e)})}{\left\| \text{sg}(\boldsymbol{H}_{\text{enc},i}^{(l_e)}) \right\|_2} \right\rangle, \tag{6}$$

Compared to MSE, cosine similarity is adopted to mitigate the influence of feature-scale discrepancies across different representation spaces [35].

Combining all the previous designs leads to the final full Phase-I training objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{gen}} + \lambda_{\text{KL}} \mathcal{L}_{\text{KL}} + \lambda_{\text{PERC}} \mathcal{L}_{\text{PERC}} + \lambda_{\text{REPA}} \mathcal{L}_{\text{REPA}}. \tag{7}$$

Here, $\mathcal{L}_{\text{gen}}$ depends on the chosen generative backbone. The complete training algorithm and additional implementation details are provided in Section A.1.

Inference. At inference time, generation follows GeoRCG [25], which proceeds in two steps. First, the RDM samples a latent representation $\hat{\boldsymbol{\mu}} \sim p_\phi$, optionally conditioned on the desired atom count, molecular properties or other constraints. Second, the molecule generator produces a 3D structure $\hat{\mathcal{M}} \sim p_\theta(\cdot \mid \hat{\boldsymbol{\mu}})$ conditioned on the sampled representation. This decoupled design allows latent generation and structure generation to be modeled separately.

4 Experiments

We evaluate LENSEs on two standard benchmarks for 3D molecule generation: GEOM-DRUG [54] and QM9 [51]. Beyond generation quality, we conduct representation-quality analyses , including effective rank and Lipschitz constant estimation and downstream regression, to validate our core claim that improving representation quality is key to improving generation quality. We further demonstrate that the refined representations transfer to a broader range of tasks, such as regression, thereby pointing to a potential paradigm for generative pretraining .

4.1 Experimental Setup

Datasets. **QM9** [51] is a benchmark of 133,885 small organic molecules with up to nine heavy atoms, each associated with a DFT-optimized equilibrium geometry and quantum chemical properties computed at the B3LYP/6-31G(2df,p) level. Its high-quality geometries and relatively simple

Table 1: Performance of LENSEs on GEOM-DRUG dataset. Atom stability, molecule stability, validity, and uniqueness are reported as percentages; Energy and strain are reported in kcal/mol.

Method	Atom Stab ↑	Mol Stab ↑	Valid ↑	Unique ↑	Energy ↓	Strain ↓
FlowMol	99.0	67.5	51.2	—	—	—
MiDi	99.8	91.6	77.8	**100.0**	—	—
EQGAT-diff	$99.8_{\pm 0.0}$	$93.4_{\pm 0.21}$	$94.6_{\pm 0.24}$	$\mathbf{100.0}_{\pm \mathbf{0.0}}$	$148.8_{\pm 0.9}$	$140.2_{\pm 0.7}$
SemlaFlow	$99.8_{\pm 0.0}$	$97.3_{\pm 0.08}$	$93.9_{\pm 0.19}$	$\mathbf{100.0}_{\pm \mathbf{0.0}}$	—	—
CanonFlow	$99.8_{\pm 0.00}$	$98.4_{\pm 0.02}$	$95.9_{\pm 0.08}$	$\mathbf{100.0}_{\pm \mathbf{0.00}}$	—	—
GeoRCG	$99.8_{\pm 0.00}$	$97.20_{\pm 0.00}$	$95.3_{\pm 0.13}$	$\mathbf{100.0}_{\pm \mathbf{0.00}}$	$88.6_{\pm 1.03}$	$47.64_{\pm 1.10}$
LENSEs (Ours)	$\mathbf{99.88}_{\pm \mathbf{0.01}}$	$\mathbf{98.51}_{\pm \mathbf{0.16}}$	$\mathbf{97.28}_{\pm \mathbf{0.18}}$	$\mathbf{100.00}_{\pm \mathbf{0.00}}$	$\mathbf{85.53}_{\pm \mathbf{0.45}}$	$\mathbf{43.86}_{\pm \mathbf{0.27}}$

Table 2: Performance of LENSEs on QM9 dataset. Atom stability, molecule stability, and validity are reported as percentages; Opt-RMSD is reported in Å.

Methods	Atom Stab ↑	Mol Stab ↑	Valid ↑	Opt-RMSD ↓	NFE ↓
EDM	98.7	82.0	91.9	–	1000
GCDM	98.7	85.7	94.8	–	1000
MUDiff	98.3	89.9	95.3	–	1000
JODO	99.9	98.8	96.0	–	1000
FlowMol	99.7	96.2	97.3	–	100
MiDi	99.8	97.5	97.9	–	500
EQGAT-diff	$99.9_{\pm 0.0}$	$98.7_{\pm 0.18}$	$99.0_{\pm 0.16}$	–	500
Semlaflow	$99.9_{\pm 0.0}$	$99.56_{\pm 0.07}$	$99.31_{\pm 0.08}$	$0.23_{\pm 0.0}$	100
LENSEs (Ours)	$\mathbf{99.9}_{\pm \mathbf{0.0}}$	$\mathbf{99.68}_{\pm \mathbf{0.04}}$	$\mathbf{99.48}_{\pm \mathbf{0.08}}$	$\mathbf{0.21}_{\pm \mathbf{0.0}}$	100

chemical space make it a standard testbed for evaluating 3D molecular generation on small molecules. **GEOM-DRUG** [54] is a large-scale 3D molecular conformation benchmark of flexible, drug-like molecules, featuring larger molecular size, richer stereochemistry, and more complex conformational landscapes than QM9. It therefore provides a realistic and challenging testbed for evaluating 3D molecular generation on drug-like compounds.

Backbone and Encoder. We take GeoRCG [25], built upon the SemlaFlow [55] framework, as our base generation model. We instantiate the pretrained encoder with either UniMol [27] or Frad [28]. The RDM is trained in the representation-head's $\boldsymbol{\mu}$-space with a SimpleMLP backbone.

Baselines. We compare our method with recent state-of-the-art 3D molecular generation methods: FlowMol [56], MiDi [38], JODO [57], EQGAT-diff [58], SemlaFlow [55], CanonFlow [36], and GeoRCG [25]. Baseline details are provided in Appendix B.10.

4.2 Main Results

Table 1 presents the results on the GEOM-DRUG. LENSEs achieves *state-of-the-art* performance across all reported metrics, surpassing all baselines. Notably, LENSEs achieves 97.28% validity (vs. the previous best GeoRCG (95.3%) and CanonFlow (95.9%)) and 98.51% molecule stability, while simultaneously achieving the lowest energy and strain energy. The improvement over the original GeoRCG (from 95.3% to 97.28% validity, and from 97.20% to 98.51% molecule stability) directly demonstrates the benefit of improved representation quality. Energy and strain metrics, which reflect the physical plausibility of generated 3D conformations, also improve consistently, confirming that our method leads to more chemically realistic structures.

Table 2 reports the results on QM9. LENSEs achieves strong performance across all four metrics while matching the fewest NFE among compared methods. This result supports our hypothesis that a better-structured latent space can provide more informative conditioning signals, thereby enabling the generator to achieve stronger generalization.

4.3 Representation Quality Analysis

We provide two complementary pieces of evidence that LENSEs yields a better-conditioned representation space: (1) latent-geometry measurements showing a less degenerate and more locally stable manifold; (2) downstream regression performance demonstrating richer representations.

Latent Geometry of the Representation Space. We first measure the *effective rank* of the representation, which counts the number of effectively independent dimensions in the latent space. As shown in Table 3, the effective rank increases from 65.32 to 88.42 on GEOM-DRUG (1.35×), and from 95.49 to 109.40 on QM9. This indicates that the raw encoder representation occupies only a narrow subspace, while the representation head expands it into a richer, better-separated manifold.

A second, complementary property is local stability. The Lipschitz constant of a mapping measures how much the representation responds to input perturbations: a lower value indicates that small perturbations produce proportionally small latent changes, which is desirable for RDM sampling stability. Estimating local Lipschitz constants following the procedure described in Section A.3.2, we find that the latent space learned by LENSEs is substantially more stable than the raw encoder representation, with the mean estimate reduced by $4.6\times$ on GEOM-DRUG and by $8.7\times$ on QM9 (Table 3).

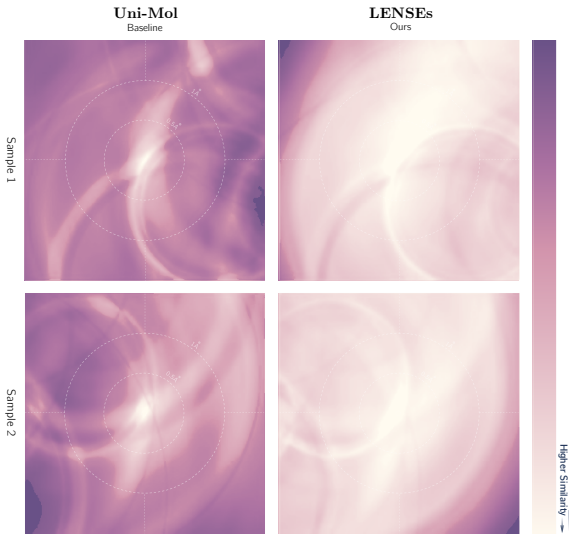


Figure 3: Latent space visualization. The two rows show local latent-space visualizations around two randomly selected atoms from two molecules. The color encodes $1-\frac{\langle \boldsymbol{h}, \boldsymbol{h}_0 \rangle}{\|\boldsymbol{h}\|\|\boldsymbol{h}_0\|}$, where $\boldsymbol{h}_0$ is the representation of the center point. Lighter purple regions indicate smaller values, i.e., higher similarity to the center point, reflecting a smoother local representation geometry.

For intuition, we further visualize the latent space in Figure 3, where the flattened region around stable conformations is substantially larger than that of the baseline.

Table 3: Representation quality on QM9 and GEOM-DRUG.

Representation	QM9 *with Frad*			GEOM-DRUG *with UniMol*		
	Lip. mean	Lip. max	Eff. rank	Lip. mean	Lip. max	Eff. rank
Raw Encoder	249.31	4600.07	95.49	2659.81	3545.67	65.32
LENSEs	28.77	452.25	**109.40**	570.49	1781.41	**88.42**

Downstream Regression on QM9. To examine whether the learned latent representations remain useful beyond generation, we freeze the pretrained encoder and the representation head of LENSEs, and train only a lightweight readout on QM9 property prediction tasks.

As shown in Table 4(b), LENSEs consistently improves frozen-representation regression over the raw Frad encoder across the evaluated QM9 properties. This indicates that representation-aligned generative training does not merely benefit sampling, but also refines the encoder features into a more predictive molecular representation.

4.4 Frozen Representation Transfer Across Datasets

The Frad results above show that the learned representation supports property prediction even when the encoder and representation head are frozen. Here we test whether this benefit carries over to other datasets. In every evaluation below, the pretrained encoder, the learned pooling, and the representation head stay frozen, and we fit only a task-specific linear probe. These experiments therefore measure how useful a representation adapter learned through generative training is downstream, without updating the encoder or the head on downstream labels.

QM9 property regression. Table 4(a) reports frozen-probing results on 12 QM9 properties. The learned representation lowers MAE on every property, with relative reductions from 21.6% to 92.4%. Because this evaluation reuses the head trained on GEOM-DRUG, it is a cross-dataset transfer setting that complements the original Frad experiment in Table 4(b).

Table 4: **Frozen-transfer performance.** Green values indicate relative regression-error reductions (%) or absolute ROC-AUC gains (pp). QM9 energy errors are reported in meV.

(a) UniMol → QM9 *MAE ↓*

Method	μ D	α bohr^3	$\Delta\varepsilon$ meV	$\varepsilon_{\mathrm{HOMO}}$ meV	$\varepsilon_{\mathrm{LUMO}}$ meV	$\langle R^2 \rangle$ bohr^2	ZPVE meV	C_v cal/mol/K	U_0 meV	U meV	H meV	G meV
Baseline	0.8096	2.0130	587.7	281.8	503.1	61.0551	85.0	1.0032	2.2913	2.1217	2.2519	2.1076
LENSEs	**0.6344**	**0.5584**	**330.3**	**190.3**	**265.6**	**11.1059**	**6.50**	**0.1719**	**0.2205**	**0.2248**	**0.2230**	**0.2228**
	21.6%↑	72.3%↑	43.8%↑	32.5%↑	47.2%↑	81.8%↑	92.4%↑	82.9%↑	90.4%↑	89.4%↑	90.1%↑	89.4%↑

(b) Frad → QM9 *MAE ↓*

Method	μ D	α bohr^3	$\Delta\varepsilon$ meV	$\varepsilon_{\mathrm{HOMO}}$ meV	$\varepsilon_{\mathrm{LUMO}}$ meV	$\langle R^2 \rangle$ bohr^2	ZPVE meV	C_v cal/mol/K
Baseline	0.152	0.247	96.67	58.70	54.65	7.23	11.58	0.100
LENSEs	**0.119**	**0.134**	**68.76**	**47.87**	**41.90**	**3.33**	**3.69**	**0.0481**
	21.7%↑	45.7%↑	28.9%↑	18.4%↑	23.3%↑	53.9%↑	68.1%↑	51.9%↑

(c) UniMol → MoleculeNet

Method	*Classification: ROC-AUC (%) ↑*							*Regression: RMSE ↓*		
	BACE	BBBP	ClinTox	HIV	Tox21	SIDER	**Average**	ESOL	FreeSolv	Lipophilicity
Baseline	75.44	89.13	78.17	74.51	79.33	58.85	75.91	0.836	1.867	0.870
LENSEs	**82.02**	**89.43**	**85.02**	**75.90**	**81.99**	**66.70**	**80.18**	**0.562**	**1.372**	**0.657**
	6.58 pp↑	0.30 pp↑	6.85 pp↑	1.39 pp↑	2.66 pp↑	7.85 pp↑	4.27 pp↑	32.8%↑	26.5%↑	24.5%↑

Table 5: Ablation of representation aggregation and loss weights on GEOM-DRUG. Loss-weight rows use learned pooling unless marked as final-layer variants. The REPA block reports a local sweep; the original logarithmic sweep and full metrics are given in Table 12.

Part	Setting	λ_{KL}	λ_{perc}	λ_{REPA}	Validity (%) ↑	Mol. Stability (%) ↑	Energy ↓	Opt. RMSD ↓	Strain ↓
Baseline (reproduced)	Final layer	–	–	–	94.20	97.20	103.33	0.8730	66.95
Full LENSEs	Learned pooling	5×10^{-7}	1×10^{-3}	1×10^{-3}	**97.28**	98.51	85.53	0.7766	43.86
KL loss *w/o Perc. and REPA*	KL	1×10^{-4}	–	–	0.00	0.00	NaN	NaN	NaN
	KL	1×10^{-5}	–	–	93.40	96.35	110.58	0.9301	70.28
	KL	1×10^{-6}	–	–	95.00	96.63	107.43	0.9395	64.60
	KL	5×10^{-7}	–	–	96.52	98.01	90.66	0.7780	43.38
	KL	1×10^{-8}	–	–	95.53	97.22	93.04	0.8564	47.00
Perc. loss *w/o REPA*	KL + Perc.	5×10^{-7}	5×10^{-4}	–	94.20	98.60	79.29	0.8303	45.00
	KL + Perc.	5×10^{-7}	7.5×10^{-4}	–	96.00	98.30	80.18	0.8455	42.55
	KL + Perc.	5×10^{-7}	1×10^{-3}	–	96.60	98.70	82.63	0.8317	39.98
	KL + Perc.	5×10^{-7}	5×10^{-3}	–	93.80	98.00	87.93	0.9572	40.79
	KL + Perc.	5×10^{-7}	1×10^{-2}	–	96.50	98.80	91.17	0.8303	44.79
REPA loss *w/o Perc.*	KL + REPA	5×10^{-7}	–	5×10^{-4}	96.60	98.40	87.34	0.8375	45.01
	KL + REPA	5×10^{-7}	–	1×10^{-3}	96.40	98.00	86.87	0.8091	45.83
	KL + REPA	5×10^{-7}	–	1.5×10^{-3}	96.20	98.70	88.12	0.8412	46.92
	KL + REPA	5×10^{-7}	–	2×10^{-3}	96.10	97.70	87.02	0.8253	47.73
w/o Multi-layer Agg.	KL	5×10^{-7}	–	–	95.90	97.80	93.25	0.8138	46.29
w/o Multi-layer Agg.	All	5×10^{-7}	1×10^{-3}	1×10^{-3}	97.10	98.20	86.20	0.8200	44.72

MoleculeNet classification and regression. We also evaluate the frozen UniMol-based adapter on nine MoleculeNet datasets [59]: six classification datasets scored by ROC-AUC and three regression datasets scored by RMSE (Table 4(c)). The refined representation improves results on every dataset. Mean ROC-AUC across the classification datasets rises from 75.91% to 80.18%, and RMSE falls by 32.8% on ESOL, 26.5% on FreeSolv, and 24.5% on Lipophilicity. For multi-task datasets, each reported score is the average over that dataset's endpoints.

Combined with the Frad results in Table 4(b), the refined representation outperforms the raw encoder features under both encoders and in all 21 frozen-transfer comparisons. These gains therefore indicate that generative training with alignment objectives learns a transferable representation adapter on top of a frozen encoder, rather than features specialized to the generation dataset.

4.5 Ablation Study

We ablate the three key objectives in LENSEs: the KL loss, the perceptual loss, and the REPA loss. For the KL loss, too large a weight can collapse the representation, while too small a weight under-regularizes the latent space. For the perceptual and REPA losses, large weights can overwhelm the generation objective or introduce conflicting gradients, while small weights weaken encoder-space semantic supervision and node-level alignment, respectively. LENSEs is robust to its loss weights

over wide ranges. Every REPA weight across four orders of magnitude (10^{-6}–10^{-2}) and every perceptual weight across a $20\times$ range (5×10^{-4}–10^{-2}) lowers both energy and strain relative to the baseline, and all REPA settings also improve validity (Tables 5 and 12). The KL weight is likewise stable across nearly two orders of magnitude (10^{-8}–5×10^{-7}); only beyond this range does the penalty pull the posterior toward the prior and degrade the conditioning signal, as is typical for KL regularization. Table 5 also compares final-layer aggregation with our learned multi-layer pooling. All reproduced variants share the same data, training budget, codebase, checkpoint-selection rule, and evaluation protocol.

Taken together, the ablations show that the components contribute complementary benefits. The KL-regularized representation head smooths the conditioning latent space, improving validity and physical plausibility. The perceptual loss reduces energy and strain, and REPA adds node-level alignment that further improves the overall trade-off when combined with the other components. The full LENSEs achieves the *highest validity*, maintains strong stability, and has substantially lower energy and strain than the baseline.

Section B gives further analyses. Beyond, Section B.7 shows that the refined representations keep generation quality strong with only a few sampling steps.

5 Conclusion

We presented LENSEs, a framework for geometric representation-conditioned molecule generation that improves generation by explicitly refining the conditioning latent space. Our work is motivated by the observation that useful molecular representations are not necessarily generation-friendly: pretrained encoder semantics may remain underutilized, poorly aligned with generator dynamics, or embedded in a latent space with suboptimal geometry.

LENSEs addresses this gap through three complementary components. The representation head with learnable layer pooling extracts smoother and more expressive multi-layer encoder features. The perceptual-style representation loss transfers multi-scale molecular semantics by aligning generated and ground-truth molecules in the pretrained encoder feature space. The REPA loss further aligns the generator's hidden states with encoder representations at the node level.

Experiments on GEOM-DRUG demonstrate that LENSEs achieves SOTA structural validity and physical plausibility. These gains come with a better-conditioned representation space: effective rank increases, and the mean local Lipschitz estimate falls on GEOM-DRUG and QM9. These results suggest that high-quality generation depends not only on the generator, but also on a conditioning representation that is smooth, semantically rich, and aligned with the generative objective.

Beyond generation, our results show that *generative training is an effective way to pretrain molecular representations*. With the encoder and representation head frozen, the refined features outperform the raw encoder features on every QM9 property under both Frad and UniMol, and on all MoleculeNet datasets. These gains include MAE reductions on QM9 and an increase in mean ROC-AUC on MoleculeNet. Because the UniMol head is trained only on GEOM-DRUG and never sees downstream labels, these gains reflect transferable representation learning rather than in-domain fitting. Generation is therefore not only a consumer of pretrained representations but also a source of better ones. We believe future molecular foundation models should treat generation as a core pretraining objective, unifying representation learning, semantic alignment, and generation within a single framework to support both molecular understanding and design.

Future Work. Future work will focus on three directions: (1) applying LENSEs to more efficient training and inference frameworks, including few-step generation settings [36]; (2) scaling generative pretraining to larger encoders and datasets, and characterizing when and why it improves downstream generalization; and (3) developing molecular-domain perceptual models for more effective semantic extraction.

Acknowledgments

This work is supported by the Beijing Natural Science Foundation (Z260022) and National Natural Science Foundation of China (62550138).

The first author thanks everyone who helped him find his way into AI for Science, and who kept him going when he wasn't sure he had chosen right. To Prof. Muhan Zhang and Zian Li, who first brought him into research and patiently guided his early steps; to Qiaojing Huang, for his support and generous help during his internship at ByteDance; and to Yuli She and Kai Liu, for the opportunities and the room to grow.

He is deeply grateful to his parents for their understanding and care through all the late nights. Thanks to Jiangyue Zeng for her companionship, patience, and support throughout, and to all his friends at *Lvzhiqiang* for years of cheering each other on.

To everyone who has been part of this journey: thank you!

References

[1] Xiangxiang Zeng, Fei Wang, Yuan Luo, Seung-gu Kang, Jian Tang, Felice C Lightstone, Evandro F Fang, Wendy Cornell, Ruth Nussinov, and Feixiong Cheng. Deep generative molecular design reshapes drug discovery. *Cell Reports Medicine*, 3(12), 2022.

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

[3] Xiangru Tang, Howard Dai, Elizabeth Knight, Fang Wu, Yunyang Li, Tianxiao Li, and Mark Gerstein. A survey of generative ai for de novo drug design: new frontiers in molecule and protein generation. *Briefings in Bioinformatics*, 25(4):bbae338, 2024.

[4] 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.

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

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

[7] Emiel Hoogeboom, Víctor Garcia Satorras, Clement Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. *International Conference on Machine Learning*, pages 9087–9102, 2022.

[8] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

[9] Deepan Adak, Yogesh Singh Rawat, and Shruti Vyas. Molvision: Molecular property prediction with vision language models. *arXiv preprint arXiv:2507.03283*, 2025.

[10] Zhanfeng Wang, Wenhao Zhang, Minghong Jiang, Yicheng Chen, Zhenyu Zhu, Wenjie Yan, Jianming Wu, and Xin Xu. X2-gnn: A physical message passing neural network with natural generalization ability to large and complex molecules. *The Journal of Physical Chemistry Letters*, 15(51):12501–12512, 2024.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[12] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[13] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[14] Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

[15] Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. *arXiv preprint arXiv:2210.01776*, 2022.

[16] W Patrick Walters and Regina Barzilay. Applications of deep learning in molecule generation and molecular property prediction. *Accounts of chemical research*, 54(2):263–270, 2020.

[17] Gary Tom, Stefan P Schmid, Sterling G Baird, Yang Cao, Kourosh Darvish, Han Hao, Stanley Lo, Sergio Pablo-García, Ella M Rajaonson, Marta Skreta, et al. Self-driving laboratories for chemistry and materials science. *Chemical Reviews*, 124(16):9633–9732, 2024.

[18] Dongyu Xue, Yukang Gong, Zhaoyi Yang, Guohui Chuai, Sheng Qu, Aizong Shen, Jing Yu, and Qi Liu. Advances and challenges in deep generative models for de novo molecule generation. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 9(3):e1395, 2019.

[19] Ian Dunn and David R Koes. Flowmol3: flow matching for 3d de novo small-molecule generation. *Digital Discovery*, 2026.

[20] Cheng Zeng, Jirui Jin, Connor Ambrose, George Karypis, Mark Transtrum, Ellad B Tadmor, Richard G Hennig, Adrian Roitberg, Stefano Martiniani, and Mingjie Liu. Propmolflow: property-guided molecule generation with geometry-complete flow matching. *Nature Computational Science*, pages 1–10, 2026.

[21] Danny Reidenbach, Filipp Nikitin, Olexandr Isayev, and Saee Gopal Paliwal. Applications of modular co-design for de novo 3d molecule generation. *Digital Discovery*, 5(2):754–768, 2026.

[22] Roman Poletukhin, Marcel Kollovieh, Eike Eberhard, and Stephan Günnemann. 3d molecule generation from rigid motifs via se (3) flows. *arXiv preprint arXiv:2601.16955*, 2026.

[23] Francesca Grisoni, Michael Moret, Robin Lingwood, and Gisbert Schneider. Bidirectional molecule generation with recurrent neural networks. *Journal of chemical information and modeling*, 60(3):1175–1183, 2020.

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

[25] Zian Li, Cai Zhou, Xiyuan Wang, Xingang Peng, and Muhan Zhang. Geometric representation condition improves equivariant molecule generation. *arXiv preprint arXiv:2410.03655*, 2024.

[26] Xingyi Yang and Xinchao Wang. Diffusion model as representation learner. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18938–18949, 2023.

[27] Gengmo Zhou, Zhifeng Gao, Qiankun Ding, Hang Zheng, Hongteng Xu, Zhewei Wei, Linfeng Zhang, and Guolin Ke. Uni-mol: A universal 3d molecular representation learning framework. *International Conference on Learning Representations*, 2023.

[28] Shikun Feng, Zhi Ni, Yanyan Lan, Zhi-Ming Ma, and Wei-Ying Ma. Fractional denoising for 3d molecular pre-training. *International Conference on Machine Learning*, 2023.

[29] Tianhong Li, Dina Katabi, and Kaiming He. Return of unconditional generation: A self-supervised representation generation method. *Advances in Neural Information Processing Systems*, 37:125441–125468, 2024.

[30] Shaoheng Yan, Zian Li, and Muhan Zhang. Georecon: Graph-level representation learning for 3d molecules via reconstruction-based pretraining. *arXiv preprint arXiv:2506.13174*, 2025.

[31] Rong Yin, Ruyue Liu, Xiaoshuai Hao, Xingrui Zhou, Yong Liu, Can Ma, and Weiping Wang. Multi-modal molecular representation learning via structure awareness. *IEEE Transactions on Image Processing*, 2025.

[32] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[33] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[34] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *European Conference on Computer Vision*, pages 694–711, 2016.

[35] Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. *arXiv preprint arXiv:2410.06940*, 2024.

[36] Cai Zhou, Zijie Chen, Zian Li, Jike Wang, Kaiyi Jiang, Pan Li, Rose Yu, Muhan Zhang, Stephen Bates, and Tommi Jaakkola. Rethinking diffusion models with symmetries through canonicalization with applications to molecular graph generation. *arXiv preprint arXiv:2602.15022*, 2026.

[37] Alex Morehead and Jianlin Cheng. Geometry-complete diffusion for 3D molecule generation and optimization. *Communications Chemistry*, 7(1):150, 2024. doi: 10.1038/s42004-024-01233-z.

[38] Clement Vignac, Nagham Osman, Laura Toni, and Pascal Frossard. Midi: Mixed graph and 3d denoising diffusion for molecule generation. *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2023.

[39] Yuxuan Song, Jingjing Gong, Minkai Xu, Ziyao Cao, Yanyan Lan, Stefano Ermon, Hao Zhou, and Wei-Ying Ma. Equivariant flow matching with hybrid probability transport for 3d molecule generation. *Advances in Neural Information Processing Systems*, 36:549–568, 2023.

[40] Haokai Hong, Wanyu Lin, and Kay Chen Tan. Accelerating 3d molecule generation via jointly geometric optimal transport. *arXiv preprint arXiv:2405.15252*, 2024.

[41] Jiaqi Guan, Wesley Wei Qian, Xingang Peng, Yufeng Su, Jian Peng, and Jianzhu Ma. 3d equivariant diffusion for target-aware molecule generation and affinity prediction. *arXiv preprint arXiv:2303.03543*, 2023.

[42] Arne Schneuing, Charles Harris, Yuanqi Du, Kieran Didi, Arian Jamasb, Ilia Igashov, Weitao Du, Carla Gomes, Tom L Blundell, Pietro Lio, et al. Structure-based drug design with equivariant diffusion models. *Nature Computational Science*, 4(12):899–909, 2024.

[43] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[44] Minkai Xu, Alexander Powers, Ron Dror, Stefano Ermon, and Jure Leskovec. Geometric latent diffusion models for 3D molecule generation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38592–38610. PMLR, 2023.

[45] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[46] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023.

[47] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Konstantinos G Derpanis, Jonathan Kelly, Marcus A Brubaker, Igor Gilitschenski, and Alex Levinshtein. Spin-nerf: Multiview segmentation and perceptual inpainting with neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20669–20679, 2023.

[48] Chenyu Wang, Cai Zhou, Sharut Gupta, Zongyu Lin, Stefanie Jegelka, Stephen Bates, and Tommi Jaakkola. Learning diffusion models with flexible representation guidance. In *ICML 2025 Generative AI and Biology (GenBio) Workshop*, 2025. URL `https://openreview.net/forum?id=o2W4FTtBVJ`.

[49] Sheheryar Zaidi, Michael Schaarschmidt, James Martens, Hyunjik Kim, Yee Whye Teh, Alvaro Sanchez-Gonzalez, Peter Battaglia, Razvan Pascanu, and Jonathan Godwin. Pre-training via denoising for molecular property prediction. *arXiv preprint arXiv:2206.00133*, 2022.

[50] Tynan Perez and Rafael Gomez-Bombarelli. Self-conditioned denoising for atomistic representation learning. *arXiv preprint arXiv:2603.17196*, 2026.

[51] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1(1):1–7, 2014.

[52] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2014.

[53] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[54] Simon Axelrod and Rafael Gomez-Bombarelli. Geom, energy-annotated molecular conformations for property prediction and molecular generation. *Scientific Data*, 9(1):185, 2022.

[55] Ross Irwin, Alessandro Tibo, Jon Paul Janet, and Simon Olsson. Semlaflow–efficient 3d molecular generation with latent attention and equivariant flow matching. *arXiv preprint arXiv:2406.07266*, 2024.

[56] Ian Dunn and David Ryan Koes. Mixed continuous and categorical flow matching for 3d de novo molecule generation. *ArXiv*, pages arXiv–2404, 2024.

[57] Han Huang, Leilei Sun, Bowen Du, and Weifeng Lv. Learning joint 2d & 3d diffusion models for complete molecule generation. *arXiv preprint arXiv:2305.12347*, 2023.

[58] Tuan Le, Julian Cremer, Frank Noe, Djork-Arné Clevert, and Kristof Schütt. Navigating the design space of equivariant diffusion-based generative models for de novo 3d molecule generation. *arXiv preprint arXiv:2309.17296*, 2023.

[59] 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. *arXiv preprint arXiv:1703.00564*, 2017. URL `https://arxiv.org/abs/1703.00564`.

[60] Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *2007 15th European signal processing conference*, pages 606–610. IEEE, 2007.

Appendix Contents

Appendix Contents **15**

A Methods and Algorithms **16**

A.1 Additional Implementation Details 16

A.2 Training Algorithm 17

A.3 Auxiliary Metric Algorithms 18

A.3.1 Effective Rank 18

A.3.2 Empirical Lipschitz Constant 18

A.3.3 Pairwise Distance Distribution 19

A.3.4 k-Nearest Neighbor Smoothness 19

B Experimental Results **20**

B.1 Layer Probe Analysis: Complete Results 20

B.1.1 Experimental Setup 20

B.1.2 PCA Variance Decomposition per Layer 20

B.1.3 Cross-Layer Correlation Matrix 21

B.2 Representation Smoothness Analysis: Supplementary Figures 21

B.2.1 Pairwise Distance Summary 21

B.3 Representation Quality Metrics 21

B.4 Subset Ablations 21

B.5 Full Metric of Fullset Ablations 23

B.6 Transfer to an EDM Generator Backbone 23

B.7 Effect of Reducing Generation Steps 23

B.8 Dataset Splits and Sampling Strategy 25

B.9 Computational Resources 25

B.10 Baselines 26

C Broader Impacts **26**

D Theoretical Motivation for Representation Refinement **29**

D.1 Definitions and Scope 29

D.2 Noisy Latent-Score Regularity 29

D.3 KL Regularization and Latent Second Moments 30

D.4 Representation Alignment and Conditional Prediction 30

D.5 Perceptual Loss and Feature-Distribution Discrepancy 31

D.6 A Local Quadratic Optimization Surrogate 31

A Methods and Algorithms

A.1 Additional Implementation Details

In this section, we provide additional formulation details for LENSEs, including the representation head, the molecule perceptual loss, the representation projection alignment loss, and the overall training objective.

Molecular Representation and Encoder Outputs. A 3D molecule is represented as $\mathcal{M} = (\boldsymbol{x}, \boldsymbol{a}, \mathcal{E})$, where $\boldsymbol{x} \in \mathbb{R}^{N\times 3}$ denotes atomic coordinates, $\boldsymbol{a}$ denotes atom types, and $\mathcal{E}$ denotes chemical bonds. We additionally use an atom mask $\boldsymbol{m} \in \{0,1\}^N$ to distinguish valid atoms from padding atoms, where $m_i = 1$ indicates that atom i is valid.

Given a molecule $\mathcal{M}$, the frozen pretrained encoder extracts layer-wise representations. For the graph-level representation head, we denote the graph-level output from the l-th encoder layer as

$$\boldsymbol{H}^{(l)}(\mathcal{M}) \in \mathbb{R}^d, \qquad l = 1, \ldots, L, \tag{8}$$

where L is the number of encoder layers and d is the graph representation dimension. In practice, these graph-level representations are obtained from atom-level hidden states through masked pooling over valid atoms, followed by normalization.

Representation Head. To obtain a more structured and generation-friendly latent space, we introduce a representation head on top of the multi-layer graph representations of the pretrained encoder. Specifically, we aggregate layer-wise representations through learnable layer pooling:

$$\boldsymbol{g}(\mathcal{M}) = \sum_{l=1}^{L} \alpha_l \boldsymbol{H}^{(l)}(\mathcal{M}), \qquad \alpha_l = \textsc{Softmax}(\boldsymbol{w})_l, \tag{9}$$

where $\boldsymbol{g}(\mathcal{M}) \in \mathbb{R}^d$ is the pooled graph representation, $\boldsymbol{w} \in \mathbb{R}^L$ denotes the learnable layer-weight logits, and α_l is the normalized importance weight assigned to the l-th encoder layer.

The pooled representation is then mapped to the parameters of a diagonal Gaussian posterior:

$$\boldsymbol{\mu} = f_\mu(\boldsymbol{g}(\mathcal{M})), \qquad \log \boldsymbol{\sigma}^2 = \operatorname{clamp}\left(f_{\log \sigma^2}(\boldsymbol{g}(\mathcal{M})), v_{\min}, v_{\max}\right), \tag{10}$$

where f_μ and $f_{\log \sigma^2}$ are learnable projection heads for predicting the mean and log-variance, respectively. The clamp operation constrains the predicted log-variance within $[v_{\min}, v_{\max}]$ for numerical stability. The latent representation is sampled via the reparameterization trick:

$$\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \tag{11}$$

where $\odot$ denotes element-wise multiplication. The sampled latent code $\boldsymbol{z}$ is then used as the conditioning representation for molecule generation.

The representation head is trained jointly with the generator. On the one hand, the generation loss encourages the representation head to extract molecule representations that are useful for reconstructing clean molecular structures from noisy states. On the other hand, we regularize the posterior with a KL divergence toward a standard Gaussian prior:

$$\mathcal{L}_{\mathrm{KL}} = -\frac{1}{2} \sum_{j=1}^{d_z} \left(1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2\right), \tag{12}$$

where d_z is the latent dimension. This regularization encourages a compact and smooth latent representation space.

Straight-Through Estimator for Discrete Atom Types. The molecule perceptual loss requires feeding the generated molecule $\hat{\mathcal{M}}$ back into the frozen pretrained encoder. Since the encoder takes discrete atom types as input, while the generator predicts continuous atom-type logits, we use a straight-through estimator to construct differentiable atom-type inputs:

$$\hat{\boldsymbol{a}}_{\mathrm{ST}} = \operatorname{onehot}\left(\arg\max \hat{\boldsymbol{a}}_{\mathrm{logits}}\right) + \hat{\boldsymbol{a}}_{\mathrm{logits}} - \operatorname{sg}\left(\hat{\boldsymbol{a}}_{\mathrm{logits}}\right), \tag{13}$$

where $\hat{\boldsymbol{a}}_{\mathrm{logits}}$ denotes the predicted atom-type logits and $\operatorname{sg}(\cdot)$ denotes the stop-gradient operation. This keeps the forward input discrete while allowing gradients to flow through the continuous logits during backpropagation.

Molecule Perceptual Loss. During training, the generator predicts a denoised molecule $\hat{\mathcal{M}}$ from a noisy input. We pass both the clean molecule $\mathcal{M}$ and the denoised molecule $\hat{\mathcal{M}}$ through the frozen pretrained encoder, and compute their pooled graph-level representations using the same learnable layer-pooling module as in the representation head. The molecule perceptual loss is defined as

$$\mathcal{L}_{\text{PERC}} = w(t) \left\| \text{sg}\left(\boldsymbol{g}(\mathcal{M})\right) - \boldsymbol{g}(\hat{\mathcal{M}}) \right\|_2^2, \tag{14}$$

where $\boldsymbol{g}(\cdot)$ denotes the pooled graph-level representation extracted by the frozen encoder, and $w(t)$ is a time-dependent weighting function. The clean-molecule representation is treated as a fixed semantic target, while gradients are allowed to flow through $\boldsymbol{g}(\hat{\mathcal{M}})$ to optimize the generator. Although the encoder parameters are frozen, gradients can still be backpropagated through the encoder computation graph to the generated molecular coordinates and atom-type logits.

We implement $w(t)$ as a cosine schedule that assigns larger weights to later denoising timesteps, where the generated molecule is closer to clean data and the encoder representations provide a more informative semantic alignment signal. Concretely, with normalized diffusion time $t \in [0, 1]$, we use

$$w(t) = w_{\min} + (w_{\max} - w_{\min}) \cdot \frac{1 + \cos(\pi t)}{2}, \tag{15}$$

where $w_{\min}$ and $w_{\max}$ control the minimum and maximum alignment weights.

Representation Projection Alignment Loss. Beyond graph-level perceptual alignment, we further align the internal node-level features of the generator with the pretrained encoder representations. Let

$$\boldsymbol{H}_{\text{enc}}^{(l_e)} \in \mathbb{R}^{N \times d_e}, \qquad \boldsymbol{H}_{\text{gen}}^{(l_g)} \in \mathbb{R}^{N \times d_g} \tag{16}$$

denote node features from the l_e-th encoder layer and the l_g-th generator layer, respectively. Here, d_e and d_g are the corresponding feature dimensions. To reconcile dimension and scale differences, we use a learnable projection

$$\text{PROJ} : \mathbb{R}^{d_g} \to \mathbb{R}^{d_e} \tag{17}$$

to map generator features into the encoder feature space.

The representation projection alignment loss is defined over the set of valid atoms $\mathcal{A} = \{i : m_i = 1\}$:

$$\mathcal{L}_{\text{REPA}} = -\frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \left\langle \frac{\text{PROJ}\left(\boldsymbol{H}_{\text{gen},i}^{(l_g)}\right)}{\left\| \text{PROJ}\left(\boldsymbol{H}_{\text{gen},i}^{(l_g)}\right) \right\|_2}, \frac{\text{sg}\left(\hat{\boldsymbol{H}}_{\text{enc},i}^{(l_e)}\right)}{\left\| \text{sg}\left(\hat{\boldsymbol{H}}_{\text{enc},i}^{(l_e)}\right) \right\|_2} \right\rangle, \tag{18}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. We use cosine similarity rather than MSE to reduce the influence of feature-scale discrepancies across the encoder and generator representation spaces [35].

Total Training Objective. Combining the generation objective with the proposed representation-level regularization terms gives the full Phase-I training objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{gen}} + \lambda_{\text{KL}} \mathcal{L}_{\text{KL}} + \lambda_{\text{PERC}} \mathcal{L}_{\text{PERC}} + \lambda_{\text{REPA}} \mathcal{L}_{\text{REPA}}, \tag{19}$$

where $\mathcal{L}_{\text{gen}}$ is the base molecule generation loss determined by the chosen generative backbone, and λ_{KL}, λ_{PERC}, and λ_{REPA} are loss-weight hyperparameters.

Inference. At inference time, generation follows the two-stage GeoRCG pipeline [25]. First, the representation diffusion model samples a latent representation $\hat{\boldsymbol{\mu}} \sim p_\phi$, optionally conditioned on atom count, molecular properties, or other constraints. Second, the molecule generator produces a 3D molecular structure conditioned on the sampled representation:

$$\hat{\mathcal{M}} \sim p_\theta\left(\cdot \mid \hat{\boldsymbol{\mu}}\right). \tag{20}$$

This decoupled design allows latent representation generation and 3D molecular structure generation to be modeled separately.

A.2 Training Algorithm

Algorithm 1 summarizes the Phase-I training procedure of LENSEs for the representation-conditioned molecular generator.

Algorithm 1 LENSEs Phase-I Training

Require: Frozen pretrained encoder f_{enc}, generator p_θ, representation head q_ψ, training set $\mathcal{D}$

for each training iteration **do**
  Sample a clean molecule $\mathcal{M} = (\boldsymbol{x}, \boldsymbol{a}, \mathcal{E}) \sim \mathcal{D}$
  Extract multi-layer graph representations $\{\boldsymbol{H}^{(l)}(\mathcal{M})\}_{l=1}^{L}$ using f_{enc}
  Compute pooled representation $\boldsymbol{g}(\mathcal{M}) = \sum_{l=1}^{L} \alpha_l \boldsymbol{H}^{(l)}(\mathcal{M})$
  Predict posterior parameters $(\boldsymbol{\mu}, \log \boldsymbol{\sigma}^2) \leftarrow q_\psi(\boldsymbol{g}(\mathcal{M}))$
  Sample latent code $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  Generate a noisy molecular state and compute the backbone generation loss $\mathcal{L}_{\text{gen}}$ conditioned on $\boldsymbol{z}$
  Obtain the denoised molecule prediction $\hat{\mathcal{M}}$ from the generator
  Feed $\hat{\mathcal{M}}$ into f_{enc} using the straight-through estimator for predicted atom types
  Compute the molecule perceptual loss $\mathcal{L}_{\text{PERC}}$ between $\boldsymbol{g}(\hat{\mathcal{M}})$ and $\text{sg}(\boldsymbol{g}(\mathcal{M}))$
  Extract encoder node features $\boldsymbol{H}_{\text{enc}}^{(l_e)}$ and generator node features $\boldsymbol{H}_{\text{gen}}^{(l_g)}$
  Map $\boldsymbol{H}_{\text{gen}}^{(l_g)}$ into encoder feature space with PROJ
  Compute the representation projection alignment loss $\mathcal{L}_{\text{REPA}}$
  Compute $\mathcal{L}_{\text{KL}}$ from $(\boldsymbol{\mu}, \boldsymbol{\sigma})$
  Compute $\mathcal{L}_{\text{total}}$ according to Eq. 19
  Update θ, ψ and PROJ by gradient descent on $\mathcal{L}_{\text{total}}$, keeping f_{enc} frozen
end for

A.3 Auxiliary Metric Algorithms

This section provides algorithmic details for the auxiliary metrics used in the representation quality analysis.

A.3.1 Effective Rank

The effective rank [60] measures how many dimensions are effectively used by a representation matrix $\mathbf{Z} \in \mathbb{R}^{N \times D}$, where N is the number of samples and D is the representation dimensionality. We compute it from the normalized eigenvalue spectrum of the representation covariance matrix. In practice, this is equivalently obtained from the squared singular values of the centered representation matrix.

Algorithm 2 Effective Rank

Require: Representation matrix $\mathbf{Z} \in \mathbb{R}^{N \times D}$
Ensure: Effective rank $\text{er} \in \mathbb{R}$

Center the representation matrix: $\bar{\mathbf{Z}} \leftarrow \mathbf{Z} - \mathbf{1}\mathbf{1}^\top \mathbf{Z}/N$
Compute singular values $\mathbf{s} = (\sigma_1, \ldots, \sigma_r)$ of $\bar{\mathbf{Z}}$ via SVD, where $r = \min(N, D)$
Normalize the squared singular-value spectrum: $p_i \leftarrow \sigma_i^2 / \sum_{j=1}^{r} \sigma_j^2$
Filter out near-zero values: keep $p_i > \epsilon$ with $\epsilon = 10^{-10}$
Compute Shannon entropy: $H(\mathbf{p}) = -\sum_i p_i \log p_i$
return $\exp(H(\mathbf{p}))$

A.3.2 Empirical Lipschitz Constant

The **empirical Lipschitz constant** measures how much the representation output changes in response to small input perturbations. For a representation mapping $f : \mathbb{R}^{N \times 3} \to \mathbb{R}^D$ (molecule coordinates $\to$ latent vector) we estimate:

$$L_{\text{emp}} = \max_i \frac{\|f(\mathbf{X}_i + \boldsymbol{\epsilon}_i) - f(\mathbf{X}_i)\|_2}{\|\boldsymbol{\epsilon}_i\|_2} \tag{21}$$

where $\boldsymbol{\epsilon}_i$ are small random coordinate perturbations applied to each molecule i.

Algorithm 3 Empirical Lipschitz Constant

Require: Encoder E, molecules $\mathbf{C} \in \mathbb{R}^{N\times 3}$, radius ϵ, num samples S
Ensure: Max and mean Lipschitz constants $(L_{\max}, L_{\text{mean}})$
$L_{\max} \leftarrow 0$, $L_{\text{sum}} \leftarrow 0$
for $i = 1$ **to** S **do**
  $\mathbf{c}_i \leftarrow \mathbf{C}[i]$
  Draw $\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, normalize to radius ϵ
  $\mathbf{c}'_i \leftarrow \mathbf{c}_i + \boldsymbol{\delta}$
  Compute representations: $\mathbf{z}_0 \leftarrow E(\mathbf{c}_i)$, $\mathbf{z}_{\text{pert}} \leftarrow E(\mathbf{c}'_i)$
  $d_z \leftarrow \|\mathbf{z}_{\text{pert}} - \mathbf{z}_0\|_2/\epsilon$
  $L_{\max} \leftarrow \max(L_{\max}, d_z)$
  $L_{\text{sum}} \leftarrow L_{\text{sum}} + d_z$
end for
$L_{\text{mean}} \leftarrow L_{\text{sum}}/S$
return $(L_{\max}, L_{\text{mean}})$

Note: We use $\epsilon = 10^{-3}$ and $S = 100$ perturbations. A *smaller* empirical Lipschitz constant indicates a *smoother* representation mapping.

A.3.3 Pairwise Distance Distribution

To characterize the geometric structure of the representation space, we compute all pairwise Euclidean distances between samples:

Algorithm 4 Pairwise Distance Statistics

Require: Representations $\mathbf{Z} \in \mathbb{R}^{N\times D}$
Ensure: Statistics $(\mu_D, \sigma_D, \text{median}_D, p_{10}, p_{90})$
Compute all pairwise distances: $D_{ij} = \|\mathbf{z}_i - \mathbf{z}_j\|_2 \quad \forall i < j$
$N_{\text{pair}} \leftarrow N(N-1)/2$
$\mu_D \leftarrow \frac{1}{N_{\text{pair}}} \sum_k D_k$
$\sigma_D \leftarrow \sqrt{\frac{1}{N_{\text{pair}}} \sum_k (D_k - \mu_D)^2}$
$\text{median}_D \leftarrow \text{median}(D_k)$
$p_{10} \leftarrow \text{percentile}(D, 10)$
$p_{90} \leftarrow \text{percentile}(D, 90)$
return $(\mu_D, \sigma_D, \text{median}_D, p_{10}, p_{90})$

Interpretation: A *low* mean and std indicates the representation is compressed into a small region of space; a *larger* mean and std indicates a well-spread representation space. Values for both representation spaces are reported in Table 9.

A.3.4 k-Nearest Neighbor Smoothness

Local smoothness measures how close each sample's neighbors are in the representation space:

Algorithm 5 k-NN Average Distance (Local Smoothness)

Require: Representations $\mathbf{Z} \in \mathbb{R}^{N \times D}$, neighbor count k
Ensure: Local smoothness score
1: Fit k-NN model on $\mathbf{Z}$ with Euclidean metric
2: **for** $i = 1$ **to** N **do**
3: Find k-nearest neighbors (excluding self): $N_k(i) = \{j_1, \ldots, j_k\}$
4: $d_i \leftarrow \frac{1}{k} \sum_{j \in N_k(i)} \|\mathbf{z}_i - \mathbf{z}_j\|_2$
5: **end for**
6: smoothness $\leftarrow \frac{1}{N} \sum_i d_i$
7: **return** smoothness

Note: We exclude the sample itself from its own neighborhood (use neighbors $k + 1$, then discard self).

B Experimental Results

B.1 Layer Probe Analysis: Complete Results

This section provides the complete linear probe results for all 10 functional groups across all 9 encoder layers, supplementing the summary presented in Figure 1a.

B.1.1 Experimental Setup

We follow the methodology described in section 3. The Frad encoder [28] (Equivariant Transformer, 9 layers, 256-dimensional representations) is applied to 30,000 QM9 molecules in a frozen state. For each layer $l \in \{0, 1, \ldots, 8\}$, a logistic regression probe is trained on 80% of the data and evaluated on the held-out 20% (stratified split). We report balanced accuracy to handle class imbalance. See Table 6 for full results.

Table 6: **Complete linear probe results: all 10 functional groups $\times$ all 9 layers.** Top-1 balanced accuracy (%) on QM9 validation set. The L0 advantage (L0 minus L8) is shown with 95% confidence intervals from the normal approximation with Bonferroni correction over the 10 comparisons. All differences are significant at $p < 0.001$ except fluoro, where both layers are at ceiling.

Layer	Hydroxyl	Amino	Carbonyl	Carboxyl	Cyano	Ether	Fluoro	Alkene	Alkyne	Aromatic
L0	99.8	97.4	99.4	98.2	99.98	99.9	100.0	96.4	99.99	98.9
L1	99.5	96.4	99.3	98.0	99.97	99.8	100.0	96.1	99.99	98.6
L2	99.1	95.4	99.2	97.8	99.96	99.8	100.0	95.8	99.98	98.5
L3	98.5	94.1	99.1	97.5	99.95	99.7	100.0	95.5	99.98	98.3
L4	97.4	92.3	98.8	97.1	99.94	99.6	100.0	94.8	99.97	98.1
L5	96.3	90.3	98.4	96.8	99.93	99.2	100.0	94.2	99.96	97.8
L6	95.1	87.6	97.8	96.5	99.92	98.5	100.0	93.7	99.95	97.6
L7	93.9	85.3	97.3	96.4	99.91	97.2	100.0	93.2	99.94	97.6
L8	92.6	82.9	96.1	96.4	99.62	85.8	100.0	92.5	99.31	97.7
L0 − L8	$+7.2_{\pm 0.3}$	$+14.5_{\pm 0.5}$	$+3.3_{\pm 0.2}$	$+1.8_{\pm 0.2}$	$+0.4_{\pm 0.0}$	$+14.1_{\pm 0.3}$	0.0	$+3.9_{\pm 0.3}$	$+0.7_{\pm 0.1}$	$+1.2_{\pm 0.2}$
n_{pos}	10,016	9,332	8,886	142	3,278	19,088	354	2,926	3,747	9,563

B.1.2 PCA Variance Decomposition per Layer

Table 7 supplements the analysis in section 3 by reporting the per-component PCA explained variance ratio for each layer.

Early layers concentrate variance in a low-dimensional subspace: the top five principal components of L0 explain 83.3% of the total variance, against 45.0% at L8. Effective rank rises nearly monotonically from 3.1 at L0 to 6.2 at L8. Deeper layers thus encode molecular information across a higher-dimensional subspace, whereas early layers keep functional-group information in a concentrated and linearly separable one, which is what the probes in Table 6 measure.

Table 7: **PCA explained variance ratio per layer.** Each entry shows the fraction of total variance captured by the top-k principal components ($k = 1, 3, 5$). Variance is concentrated in early layers and becomes progressively more distributed in later layers. Effective rank is $\exp(H(\mathbf{p}))$ as defined in Section A.3.1.

Layer	PC1	PC2	PC3	Top-3	Top-5	Eff. Rank
L0	31.0%	23.5%	14.3%	68.8%	83.3%	3.1
L1	23.2%	19.4%	14.1%	56.7%	71.4%	3.8
L2	35.0%	17.8%	10.2%	63.0%	71.5%	3.4
L3	28.5%	15.9%	9.6%	54.0%	62.0%	4.1
L4	23.2%	15.6%	9.4%	48.2%	56.7%	4.6
L5	23.1%	15.3%	8.8%	47.2%	54.6%	4.8
L6	25.7%	15.1%	9.7%	50.5%	61.2%	4.4
L7	16.3%	14.8%	9.3%	40.4%	49.9%	5.5
L8	13.6%	13.2%	9.1%	35.9%	45.0%	6.2

B.1.3 Cross-Layer Correlation Matrix

Table 8 reports the full pairwise cosine similarity matrix between layer-level mean representations, confirming the two-block structure identified in section 3.

Table 8: **Cross-layer cosine similarity matrix.** Each entry (i, j) shows the cosine similarity between the mean graph representation of layer i and layer j. Two blocks are clearly visible: L0–L5 (feature extraction, $r = 0.66$–0.95) and L7–L8 (semantic aggregation, $r = 0.96$). The sharpest transition occurs at L5→L6.

	L0	L1	L2	L3	L4	L5	L6	L7	L8
L0	1.00	0.95	0.89	0.83	0.69	0.66	0.51	0.54	0.52
L1	0.95	1.00	0.95	0.89	0.77	0.72	0.54	0.54	0.52
L2	0.89	0.95	1.00	0.93	0.83	0.80	0.60	0.53	0.52
L3	0.83	0.89	0.93	1.00	0.90	0.85	0.64	0.54	0.51
L4	0.69	0.77	0.83	0.90	1.00	0.94	0.72	0.52	0.48
L5	0.66	0.72	0.80	0.85	0.94	1.00	0.77	0.52	0.50
L6	0.51	0.54	0.60	0.64	0.72	0.77	1.00	0.61	0.61
L7	0.54	0.54	0.53	0.54	0.52	0.52	0.61	1.00	0.96
L8	0.52	0.52	0.52	0.51	0.48	0.50	0.61	0.96	1.00

B.2 Representation Smoothness Analysis: Supplementary Figures

This section supplements the representation quality analysis in subsection 4.3 with additional visualizations of the representation head's latent space on GEOM-DRUG [54].

B.2.1 Pairwise Distance Summary

Figure 5 summarizes the pairwise Euclidean distances between molecular representations in the raw UniMol encoder space versus the representation-head's latent space. The raw encoder representations cluster into an extremely tight region (mean distance: 0.59, std: 0.49). The representation head expands this space by $287\times$ (mean: 169.12, std: 42.49), producing a well-structured latent manifold with meaningful geometric separation between molecules.

B.3 Representation Quality Metrics

B.4 Subset Ablations

To examine the robustness of LENSEs to its loss weights over a broad hyperparameter range, we conduct a grid ablation study on a randomly sampled 10% subset of the GEOM-DRUG dataset. Each configuration is trained for 300 epochs under the same training protocol. We vary the KL

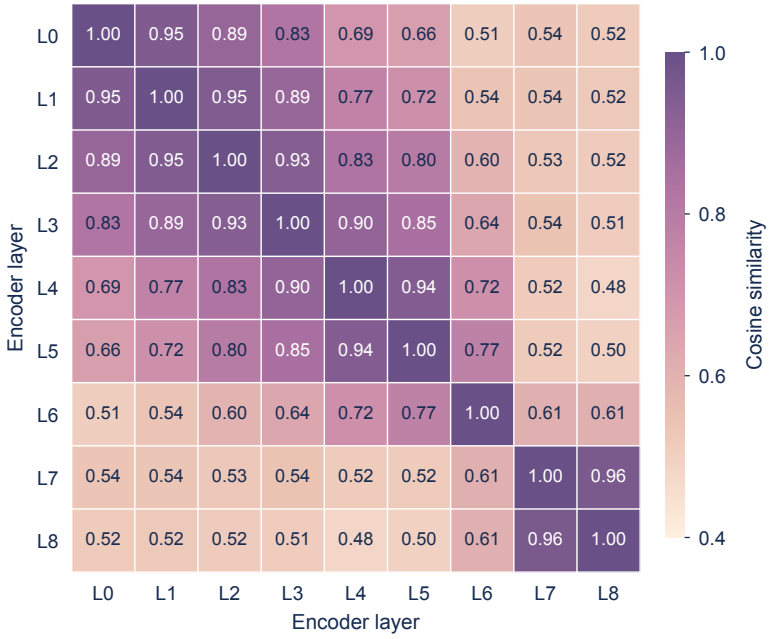


Figure 4: **Cross-layer cosine similarity heatmap.** Pairwise cosine similarity between mean graph-level representations from nine Frad encoder layers on QM9.

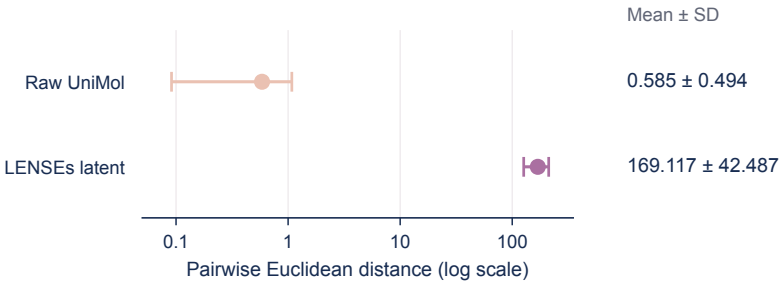


Figure 5: **Pairwise distance summary: raw encoder vs. representation latent space (GEOM-DRUG).** Points show mean pairwise Euclidean distances; error bars indicate one standard deviation across pairwise distances. The horizontal axis is logarithmic.

regularization weight and the perceptual weight, and compare the resulting models against the baseline generator trained with the same subset setting. In the tables, cells highlighted in green indicate configurations that outperform the baseline on the corresponding metric.

Table 10 first compares the baseline with two representative variants of LENSEs. Both variants improve atom stability, molecule stability, validity, energy, and strain. The perceptual-loss variant raises molecule stability from 92.82 to 97.20 and validity from 82.86 to 93.50, and lowers strain from 111.46 to 84.19. The latent refinement therefore improves both chemical validity and stability and the physical quality of generated conformations.

Table 11 reports the full grid over KL and perceptual weights. Of the 180 metric cells in the grid, 135 are better than the baseline. For $\lambda_{\mathrm{KL}} \in [2.5 \times 10^{-8}, 5 \times 10^{-6}]$, molecule stability exceeds the baseline in 24 of 25 configurations and validity in 21 of 25. Within this range the gains therefore do not depend on a single tuned setting. Larger KL weights are examined on the full dataset in Section B.5.

Table 9: **Representation quality metrics on GEOM-DRUG.**

Metric	Raw Encoder	Representation Head's Latent
Effective Rank	65.32	**88.42**
Lipschitz (mean)	2659.81	**570.49**
Lipschitz (max)	3545.67	**1781.41**
Pairwise Distance (mean)	0.59	169.12
Pairwise Distance (std)	0.49	42.49
k-NN Avg Distance	0.19	102.94

Table 10: Representative subset ablations on a randomly sampled 10% subset of GEOM-DRUG. Cells highlighted in green indicate results better than the baseline.

Metric	Baseline	Rep. head KL weight 5×10^{-7}	Perc. weight 0.001
atom-stability	99.76	99.76	99.84
molecule-stability	92.82	94.70	97.20
validity	82.86	91.00	93.50
energy	149.67	105.01	112.01
strain	111.46	99.48	84.19
opt-rmsd	1.07	0.99	1.11

B.5 Full Metric of Fullset Ablations

To provide a more complete view of the effect of each auxiliary objective, we report the full ablation results over the loss weights in Table 12. The REPA rows cover 10^{-6} to 10^{-2}; a finer sweep around 10^{-3} is given in the main-text Table 5. With $\lambda_{\mathrm{KL}} = 5 \times 10^{-7}$ fixed, every REPA weight across four orders of magnitude (10^{-6}–10^{-2}) and every perceptual weight across a $20\times$ range (5×10^{-4}–10^{-2}) lowers both energy and strain relative to the baseline. The KL weight is stable across nearly two orders of magnitude (10^{-8}–5×10^{-7}); larger weights pull the posterior toward the prior, as is typical for KL regularization, and the representation collapses at 10^{-4}.

B.6 Transfer to an EDM Generator Backbone

We further evaluate LENSEs with EDM [7] as a second generator backbone in the representation-conditioned two-stage framework of GeoRCG [25]. As shown in Table 13, LENSEs increases validity from 95.6% to 97.5% and molecule stability from 87.6% to 94.2%, gains of 1.9 and 6.6 percentage points, respectively. Atom stability increases from 98.9% to 99.3%, and uniqueness remains 100.0%.

B.7 Effect of Reducing Generation Steps

A practical advantage of molecular generative models is their ability to generate high-quality samples with fewer sampling steps. To evaluate whether LENSEs improves sampling efficiency, we compare SemlaFlow and LENSEs under different numbers of generation steps on the GEOM-DRUG dataset. Both methods use the same generator architecture and sampling protocol, while LENSEs replaces the original latent representation with the representation-head-refined latent.

Table 14 shows that from 20 sampling steps onward, LENSEs improves all four metrics over SemlaFlow. In particular, with only 20 sampling steps, LENSEs achieves a validity of 0.967, a connected-validity of 0.928, and a molecule stability of 0.966, already matching or exceeding the performance of SemlaFlow at 50 or 100 steps. This suggests that the representation-head-refined latent provides a more effective conditioning signal, allowing the generator to reach high-quality molecular samples with fewer refinement steps.

The advantage is also reflected in energy-related quality. At 10 steps, validity and molecule stability are on par with SemlaFlow, while energy per atom drops from 4.58 to 3.93. Across 10, 20, 50, and 100 sampling steps, LENSEs obtains lower energy than SemlaFlow, indicating that the generated conformations are not only chemically valid but also more physically plausible. For example, at 20 steps, LENSEs reduces the energy from 2.30 to 1.98, and at 100 steps, from 2.06 to 1.77.

Table 11: Grid ablation of KL regularization weight and perceptual loss weight on a randomly sampled 10% subset of GEOM-DRUG. Each configuration is trained for 300 epochs. Cells highlighted in green indicate results better than the baseline. REPA weight is fixed to 1×10^{-3}.

KL weight	Metric	Perc. Loss weight				
		5×10^{-4}	7.5×10^{-4}	1×10^{-3}	2.5×10^{-3}	5×10^{-3}
1e-08	Energy	119.6	136.9	131.3	184.0	154.4
	Strain	90.77	90.03	91.89	135.53	91.67
	Opt-RMSD	0.874	0.990	1.171	1.285	1.191
	Validity	74.5	37.3	73.0	71.9	79.3
	Atom Stability	99.79	98.99	99.74	99.78	99.82
	Mol Stability	93.50	67.60	91.60	92.60	94.70
2.5e-08	Energy	107.6	104.3	104.6	157.2	143.2
	Strain	80.44	89.82	81.42	125.60	92.96
	Opt-RMSD	1.105	1.144	1.085	1.164	1.143
	Validity	88.1	87.9	85.8	86.3	82.3
	Atom Stability	99.83	99.78	99.74	99.79	99.82
	Mol Stability	94.80	93.70	90.80	93.82	93.62
2.5e-07	Energy	135.9	140.8	122.8	123.8	131.2
	Strain	85.80	83.69	88.98	86.15	94.50
	Opt-RMSD	1.033	1.171	0.921	1.002	1.050
	Validity	83.8	89.9	80.2	90.5	87.2
	Atom Stability	99.89	99.88	99.80	99.81	99.78
	Mol Stability	97.50	96.40	93.50	94.90	93.20
5e-07	Energy	123.5	164.7	183.8	171.8	114.0
	Strain	105.66	107.00	123.39	116.22	83.06
	Opt-RMSD	1.141	1.030	0.995	1.041	1.093
	Validity	82.9	85.1	87.6	79.7	90.9
	Atom Stability	99.80	99.82	99.82	99.80	99.83
	Mol Stability	95.50	95.50	94.20	93.80	96.90
1e-06	Energy	119.9	131.2	159.4	124.2	127.3
	Strain	87.24	86.33	126.01	89.28	88.72
	Opt-RMSD	1.113	1.159	1.068	1.087	1.062
	Validity	94.0	87.8	83.2	92.7	91.0
	Atom Stability	99.84	99.88	99.73	99.89	99.84
	Mol Stability	96.80	97.20	93.00	97.50	96.60
5e-06	Energy	147.3	150.4	139.4	142.8	140.8
	Strain	101.82	91.43	105.54	98.00	90.29
	Opt-RMSD	1.036	1.071	1.070	1.116	1.010
	Validity	88.0	84.5	89.4	80.8	83.2
	Atom Stability	99.85	99.86	99.85	99.91	99.83
	Mol Stability	96.40	96.60	96.70	96.70	95.80

At 5 steps, neither method has converged. Overall, these results indicate that LENSEs improves the sampling efficiency of SemlaFlow by enabling strong generation performance with substantially fewer generation steps.

B.8 Dataset Splits and Sampling Strategy

For experiments on the **GEOM-DRUG** benchmark, we use both the full dataset and a representative subset. The full dataset, denoted as *Fullset*, is split into 240,990 training molecules, 30,434 validation molecules, and 30,434 test molecules.

To enable efficient hyperparameter tuning and ablation studies under reduced computational cost, we additionally construct a representative subset, denoted as *Subset*. The Subset contains 24,099 training molecules, corresponding to exactly 10% of the full training split, and 6,087 validation molecules, corresponding to approximately 20% of the full validation split. Importantly, the test set is kept identical to that of the Fullset, containing 30,434 molecules, for all subset experiments. This design ensures that evaluation metrics obtained from subset-trained models remain directly comparable to those of fullset-trained models and external baselines.

For **QM9** experiments, we adopt the commonly used split setting in prior 3D molecular generation work. The full QM9 dataset contains approximately 134k molecules. Following standard practice, we first randomly shuffle the dataset with a fixed random seed and assign 100,000 molecules to the training split. We then use 13,083 molecules as the test split and the remaining 17,748 molecules as the validation split. All QM9 experiments use the same split to ensure that different model variants are evaluated under identical data protocols.

Stratified Sampling by Atom Count. To ensure that the Subset preserves the structural complexity of the original chemical space, we adopt a stratified sampling strategy based on molecular size, defined by the exact number of atoms in each molecule. Preserving the atom-count distribution is important for training 3D molecular generative models, since both conformational diversity and generation difficulty are strongly affected by molecular size.

The subset construction is performed as follows. First, all molecules in each full split are grouped into distinct strata according to their integer atom counts. For each stratum, we compute its target number of selected samples according to its proportional frequency in the corresponding full split, and take the floor of this value as the initial allocation. Since this flooring operation may lead to a total count smaller than the desired subset size, we distribute the remaining sample slots using the largest-remainder method: strata with the largest fractional remainders receive one additional sample sequentially until the exact target subset size is reached. Within each stratum, molecular indices are randomly shuffled, and the allocated number of indices is selected. Finally, all selected indices from different strata are merged and globally shuffled to remove any ordering bias introduced by the stratification procedure.

Reproducibility. To make the subset splits exactly reproducible across different training runs and baseline evaluations, all random operations in intra-stratum selection and global shuffling are controlled by fixed random seeds. Specifically, we use seed 42 for constructing the training subset and seed 43 for constructing the validation subset.

B.9 Computational Resources

We report the computational resources used for training the molecular generator and the representation head in our main experiments. All comparisons between the baseline generator and LENSEs were conducted under the same training protocol and hyperparameter configuration, ensuring a fair assessment of the additional computational overhead introduced by our method.

On the GEOM-DRUG dataset, both the baseline and LENSEs were trained for 300 epochs on Tesla A100 80GB GPUs with identical hyperparameters. LENSEs adds about 1GB of GPU memory and 30 GPU hours, a 19% increase in training cost (Table 15). The backbone architecture and the main generator training configuration remain unchanged.

On the QM9 dataset, both methods triggered early stopping within 8 GPU hours, and LENSEs used less GPU memory than the baseline (7.0GB vs. 7.8GB).

B.10 Baselines

We compare against the following generators.

MiDi [38] jointly diffuses the molecular graph and 3D coordinates: coordinates receive Gaussian noise in the zero center-of-mass subspace, while atom types, charges, and bond types follow categorical diffusion. The denoiser is a graph Transformer with relaxed EGNN layers, and an adaptive schedule noises atom types and charges faster than coordinates and bonds.

JODO [57] also models 2D topology and 3D geometry jointly. Its Diffusion Graph Transformer uses relational attention to update node and edge features together, and it supports conditioning on quantum properties.

EQGAT-diff [58] is an $E(3)$-equivariant attention-based diffusion model with x_0-parameterization and truncated-SNR time weighting. It adds hybridization, aromaticity, and ring features as inputs.

SemlaFlow [55] replaces diffusion with equivariant flow matching and optimal-transport alignment, using latent graph attention (Semla) for message passing. It samples in as few as 20 steps. It is the backbone of GeoRCG and of LENSEs.

FlowMol [56] applies continuous flow matching to coordinates and continuous relaxations to categorical variables; its final version uses a Gaussian prior for categorical features rather than simplex-constrained flows.

CanonFlow [36] canonicalizes atom order and pose with spectral features so that a non-equivariant backbone can be trained on a single canonical slice; invariance is restored at sampling time by random symmetry transformations.

GeoRCG [25] is the closest baseline to our setting. It first samples a molecule-level representation from a pretrained encoder's distribution with a representation diffusion model, then conditions a base generator (EDM or SemlaFlow) on that representation. LENSEs builds on this framework (Section 4.1).

C Broader Impacts

This work studies representation-conditioned 3D molecular generation, with the goal of improving the validity, stability, and physical plausibility of generated molecular structures. Its potential positive societal impacts include accelerating early-stage molecular design, reducing the cost of computational screening, and providing more reliable generative tools for scientific discovery. By improving the connection between learned molecular representations and 3D structure generation, the proposed method may contribute to more efficient exploration of chemical space and support applications in drug discovery, materials design, and molecular property optimization.

At the same time, molecular generative models may carry potential negative societal impacts. In particular, methods that improve the quality or efficiency of molecule generation could, in principle, be misused to propose harmful, toxic, or otherwise unsafe compounds. Although our work is foundational and does not directly optimize for toxicity, bioactivity, synthesizability, or deployment in real-world discovery pipelines, we acknowledge that improved molecular generation techniques may become part of broader systems with dual-use risks.

Several mitigation strategies are therefore important. Generated molecules should not be treated as actionable candidates without expert review, chemical validity checks, toxicity and safety screening, synthesizability assessment, and domain-specific experimental validation. In addition, practical deployment of such systems should consider access control, monitoring, and integration with filters for known toxicophores or restricted chemical classes. We release the method as a research contribution intended to improve scientific understanding of representation-conditioned molecular generation, rather than as an autonomous molecular design system.

Table 12: Full loss-weight ablations on the full GEOM-DRUG set, covering the KL, perceptual, and REPA losses. For the perceptual and REPA ablations we fix $\lambda_{\mathrm{KL}} = 5 \times 10^{-7}$. Percentage-based metrics are reported in percent. All models in this table are trained under exactly the same protocol, except for the ablated hyperparameters.

Setting	λ_{KL}	λ_{perc}	λ_{REPA}	Atom Stable (%) ↑	Mol. Stable (%) ↑	Validity (%) ↑	Unique (%) ↑	Novel (%) ↑	Energy ↓	Energy/Atom ↓	Energy Valid (%) ↑	Opt. Energy Valid (%) ↑	Opt. RMSD ↓	Strain ↓	Strain/Atom ↓
Baseline	–	–	–	99.85	97.20	94.20	100.00	99.58	103.33	2.2978	94.10	94.10	0.8730	66.95	1.4881
LENSEs	5×10^{-7}	1×10^{-3}	1×10^{-3}	99.88	98.51	97.28	100.00	100.00	85.53	1.8787	97.16	97.16	0.7766	43.86	0.9451
KL loss ablation															
KL	1×10^{-4}	–	–	34.78	0.00	0.00	–	–	–	–	0.00	0.00	–	–	–
KL	1×10^{-5}	–	–	99.82	96.35	93.40	100.00	99.68	110.58	2.4482	93.20	93.20	0.9301	70.28	1.5664
KL	1×10^{-6}	–	–	99.68	96.63	95.00	100.00	99.65	107.43	2.4040	94.93	94.93	0.9395	64.60	1.4515
KL	5×10^{-7}	–	–	99.70	98.01	96.52	100.00	100.00	90.66	2.0498	96.32	96.32	0.7780	43.38	0.9852
KL	1×10^{-8}	–	–	99.70	97.22	95.53	100.00	100.00	93.04	2.1498	95.53	95.53	0.8564	47.00	1.0615
Perceptual loss ablation under $\lambda_{\mathrm{KL}} = 5 \times 10^{-7}$															
KL + Perc.	5×10^{-7}	5×10^{-4}	–	99.96	98.60	94.20	100.00	99.58	79.29	2.4552	94.20	94.20	0.8303	45.00	1.6775
KL + Perc.	5×10^{-7}	7.5×10^{-4}	–	99.77	98.30	96.00	100.00	99.79	80.18	1.8298	95.90	95.90	0.8455	42.55	0.9856
KL + Perc.	5×10^{-7}	1×10^{-3}	–	99.73	98.70	96.60	100.00	99.79	82.63	1.8410	96.50	96.50	0.8317	39.98	0.8765
KL + Perc.	5×10^{-7}	5×10^{-3}	–	99.89	98.00	93.80	100.00	99.89	87.93	1.9620	93.80	93.80	0.9572	40.79	0.9178
KL + Perc.	5×10^{-7}	1×10^{-2}	–	99.95	98.80	96.50	100.00	99.79	91.17	2.0296	96.50	96.50	0.8303	44.79	1.0008
REPA loss ablation under $\lambda_{\mathrm{KL}} = 5 \times 10^{-7}$															
KL + REPA	5×10^{-7}	–	1×10^{-6}	99.49	98.10	96.60	100.00	99.59	102.86	2.2461	96.50	96.50	0.8727	59.36	1.2727
KL + REPA	5×10^{-7}	–	1×10^{-5}	99.39	97.70	96.00	100.00	99.48	80.02	1.8149	96.00	96.00	0.8181	37.61	0.8434
KL + REPA	5×10^{-7}	–	1×10^{-4}	99.92	99.00	96.30	100.00	100.00	90.84	2.0476	96.30	96.30	0.9059	46.69	1.0501
KL + REPA	5×10^{-7}	–	1×10^{-3}	99.74	98.00	96.40	100.00	99.69	86.87	1.9782	96.30	96.30	0.8091	45.83	1.0364
KL + REPA	5×10^{-7}	–	1×10^{-2}	99.77	97.10	95.80	100.00	99.79	86.29	1.9443	95.60	95.60	0.8465	44.56	1.0059

Table 13: Transfer of LENSEs to an EDM generator backbone within the GeoRCG two-stage framework. All metrics are reported as percentages.

Method	Validity ↑	Uniqueness ↑	Atom Stab. ↑	Mol. Stab. ↑
GeoRCG-EDM baseline	95.6	100.0	98.9	87.6
EDM + LENSEs	**97.5**	100.0	**99.3**	**94.2**

Table 14: Generation performance comparison between the baseline latent and the representation-head-refined latent under different sampling steps.

Steps	Validity		Conn-Valid		Mol-Stab		Energy/Atom	
	SemlaFlow	LENSEs	SemlaFlow	LENSEs	SemlaFlow	LENSEs	SemlaFlow	LENSEs
5	0.127	**0.141**	**0.062**	0.038	**0.039**	0.031	22.65	**22.57**
10	**0.778**	0.777	**0.742**	0.721	0.732	**0.735**	4.58	**3.93**
20	0.932	**0.967**	0.891	**0.928**	0.952	**0.966**	2.30	**1.98**
50	0.951	**0.974**	0.927	**0.950**	0.976	**0.984**	2.26	**1.84**
100	0.951	**0.973**	0.932	**0.949**	0.975	**0.983**	2.06	**1.77**

Table 15: Computational resources for training the molecular generator and representation head. All GEOM-DRUG experiments were trained for 300 epochs on Tesla A100 80GB GPUs using identical hyperparameters for the baseline and LENSEs. On QM9, both methods triggered early stopping within 8 hours.

Dataset	Method	GPU	GPU Memory	Training Cost
GEOM-DRUG	Baseline	Tesla A100 80GB	18.7GB	155 GPU hours
GEOM-DRUG	LENSEs	Tesla A100 80GB	19.7GB	185 GPU hours
QM9	Baseline	Tesla A100 80GB	7.8GB	< 8 GPU hours
QM9	LENSEs	Tesla A100 80GB	7.0GB	< 8 GPU hours

D Theoretical Motivation for Representation Refinement

This section gives statements and proofs for the theoretical motivation behind the three representation-refinement objectives. We treat four aspects separately—latent regularity, conditional feature readout, feature-distribution discrepancy, and a local optimization surrogate—since each rests on different assumptions.

D.1 Definitions and Scope

Let $(\mathcal{X}, d_{\mathrm{mol}})$ be a molecular data space, and let $\boldsymbol{\mu}_\psi : \mathcal{X} \to \mathbb{R}^{d_r}$ be the mean produced by the learned representation head on top of the frozen encoder. In a fixed-composition local region, d_{mol} can be defined on molecular configurations modulo the relevant rigid-motion and atom-permutation symmetries.

The Phase-I stochastic condition is $\boldsymbol{z} = \boldsymbol{\mu}_\psi(\mathcal{M}) + \boldsymbol{\sigma}_\psi(\mathcal{M}) \odot \boldsymbol{\epsilon}$, whereas the Phase-II RDM targets the distribution of the deterministic mean $R_0 = \boldsymbol{\mu}_\psi(\mathcal{M})$. We use Euclidean norms in representation space, operator norms for matrices, and expectations over the explicitly stated data, noise, or prediction–target coupling. The encoder and representation-head parameters are fixed when analyzing a learned feature map.

Scope. The results below are heuristic motivations derived under idealized assumptions (Lipschitz or bi-Lipschitz feature maps, bounded support, fixed learned maps, and a local quadratic model), not guarantees for the trained model. They explain why each objective is expected to help; the evidence that it does help in practice is the ablation study in Section 4.5 and Section B.5.

D.2 Noisy Latent-Score Regularity

Lemma D.1 (Gaussian smoothing of a bounded mean-space representation) *Suppose the support of the molecular data distribution has diameter at most D_X in d_{mol}, and $\boldsymbol{\mu}_\psi$ is L_μ-Lipschitz on that support. Let*

$$R_t = \alpha_t R_0 + \sigma_t \xi, \qquad R_0 = \boldsymbol{\mu}_\psi(\mathcal{M}), \qquad \xi \sim \mathcal{N}(0, I), \quad \sigma_t > 0. \tag{22}$$

If q_t is the density of R_t, then

$$\mathrm{Lip}(\nabla \log q_t) \le \sigma_t^{-2} + \frac{\alpha_t^2 L_\mu^2 D_X^2}{4\sigma_t^4}. \tag{23}$$

Proof. Write $Y = \alpha_t R_0$ and $\sigma = \sigma_t$. The density is the Gaussian convolution

$$q_t(x) = \int \phi_\sigma(x - y)\, dP_Y(y). \tag{24}$$

Bounded support permits differentiation under the integral. With $m(x) = \mathbb{E}[Y \mid R_t = x]$, differentiating the Gaussian kernel gives

$$\nabla \log q_t(x) = \sigma^{-2}(m(x) - x). \tag{25}$$

The derivative of the log posterior density with respect to x_j is $(y_j - m_j(x))/\sigma^2$. Hence

$$\frac{\partial m_i(x)}{\partial x_j} = \sigma^{-2} \operatorname{Cov}(Y_i, Y_j \mid R_t = x), \tag{26}$$

and therefore

$$\nabla^2 \log q_t(x) = -\sigma^{-2} I + \sigma^{-4} \operatorname{Cov}(Y \mid R_t = x). \tag{27}$$

The support of Y has diameter at most $D_Y = |\alpha_t| L_\mu D_X$. For any unit vector v, the scalar $v^T Y$ lies in an interval of length at most D_Y. If c_v is the midpoint of this interval, then $\operatorname{Var}(v^T Y \mid R_t = x) \le \mathbb{E}[(v^T Y - c_v)^2 \mid R_t = x] \le D_Y^2/4$. Thus

$$\| \operatorname{Cov}(Y \mid R_t = x) \|_{\mathrm{op}} \le D_Y^2/4. \tag{28}$$

Taking operator norms in Equation (27) yields the claimed bound uniformly in x. Integrating the bounded Hessian along a line segment proves the global Lipschitz bound for the score. □

For a fixed noise schedule and molecular support, a smaller L_μ tightens the second term of Equation (23), so a smoother representation map yields a better-conditioned latent score at every positive noise level.

Relation to Representation Sampling Bounds. GeoRCG Proposition D.4 [25] depends on score regularity, a representation second-moment bound, score-estimation error, and a compatible discretization step size. Lemma D.1 provides a regularity estimate at positive noise levels. On an interval $[\tau, T]$ with $\tau > 0$, a schedule with $\inf_{t\in[\tau,T]} \sigma_t > 0$ and bounded α_t gives a finite uniform score bound, which can be combined with an early-stopping argument controlling the bias between q_τ and q_0.

D.3 KL Regularization and Latent Second Moments

Proposition D.2 (Posterior KL controls the mean-space second moment) *Let $q_\psi(z \mid \mathcal{M}) = \mathcal{N}(\boldsymbol{\mu}_\psi(\mathcal{M}), \mathrm{diag}(s_1^2, \ldots, s_{d_r}^2))$, with every $s_j^2 > 0$, and let $\gamma = \mathcal{N}(0, I)$. For the summed Gaussian posterior KL used in Equation* (12)*,*

$$\mathbb{E}\|R_0 - \mathbb{E}R_0\|^2 \le \mathbb{E}\|R_0\|^2 \le 2\mathbb{E}_{\mathcal{M}}\,\mathrm{KL}(q_\psi(\cdot \mid \mathcal{M})\|\gamma). \tag{29}$$

Proof. The Gaussian KL formula gives

$$\mathrm{KL}(q_\psi(\cdot \mid \mathcal{M})\|\gamma) = \frac{1}{2}\sum_{j=1}^{d_r}\left(\mu_j^2 + s_j^2 - 1 - \log s_j^2\right). \tag{30}$$

Since $a - 1 - \log a \ge 0$ for $a > 0$, the expression is at least $\|\boldsymbol{\mu}_\psi(\mathcal{M})\|^2/2$. Taking expectations proves the second inequality. The first follows from $\mathbb{E}\|R_0 - \mathbb{E}R_0\|^2 = \mathbb{E}\|R_0\|^2 - \|\mathbb{E}R_0\|^2$. □

The posterior KL, the aggregated stochastic posterior $q_z = \mathbb{E}_{\mathcal{M}} q_\psi(z \mid \mathcal{M})$, and the mean-space distribution $q_\mu = (\boldsymbol{\mu}_\psi)_{\#} P_{\mathcal{M}}$ must be distinguished. Convexity gives

$$\mathrm{KL}(q_z\|\gamma) \le \mathbb{E}_{\mathcal{M}}\,\mathrm{KL}(q_\psi(\cdot \mid \mathcal{M})\|\gamma), \tag{31}$$

whereas for the deterministic-mean distribution q_μ the relevant controlled quantity is the second moment in Equation (29), which is the term that enters representation-sampling bounds. When the KL is averaged rather than summed over latent coordinates, the right-hand side of Equation (29) is multiplied by d_r.

D.4 Representation Alignment and Conditional Prediction

Proposition D.3 (An oracle-readout bound for normalized alignment) *Let u and e be unit-norm projected generator and encoder features, and define $\ell_{\mathrm{align}} = 1 - \langle u, e\rangle$. Suppose an oracle target s^* admits an L_A-Lipschitz feature readout A with $\mathbb{E}\|A(e) - s^*\|^2 \le \rho^2$. Then*

$$\mathbb{E}\|A(u) - s^*\|^2 \le 4L_A^2\mathbb{E}\ell_{\mathrm{align}} + 2\rho^2. \tag{32}$$

If the actual predictor s_θ has realization residual $\delta^2 = \mathbb{E}\|s_\theta - A(u)\|^2$, then

$$\mathbb{E}\|s_\theta - s^*\|^2 \le 3\delta^2 + 6L_A^2\mathbb{E}\ell_{\mathrm{align}} + 3\rho^2. \tag{33}$$

Proof. Unit normalization gives $\|u - e\|^2 = 2(1 - \langle u, e\rangle)$. Split $A(u) - s^* = [A(u) - A(e)] + [A(e) - s^*]$, apply $\|a+b\|^2 \le 2\|a\|^2 + 2\|b\|^2$, and use the Lipschitz condition to obtain Equation (32). For the actual predictor, use the three-term decomposition

$$s_\theta - s^* = [s_\theta - A(u)] + [A(u) - A(e)] + [A(e) - s^*]. \tag{34}$$

The inequality $\|a + b + c\|^2 \le 3(\|a\|^2 + \|b\|^2 + \|c\|^2)$ gives Equation (33). □

The readout can additionally depend on the noisy state, time, and conditioning representation, provided its Lipschitz constant in the aligned feature is uniform over the considered setting. The bound shows that, for a fixed readout, the alignment loss directly controls the only term in the conditional prediction error that depends on the generator's intermediate features.

The implemented REPA objective is negative cosine similarity, whereas the discrepancy in the proposition is nonnegative $1 - \cos$. Under the same averaging convention, $\ell_{\mathrm{align}} = 1 + \mathcal{L}_{\mathrm{REPA}}$. For node-wise losses, the proposition applies directly to a node-level target; a molecular-level readout picks up the corresponding atom-count or averaging factor.

D.5 Perceptual Loss and Feature-Distribution Discrepancy

Lemma D.4 (Paired feature error bounds Wasserstein discrepancy) *Let $(\widehat{\mathcal{M}}, \mathcal{M})$ have the joint distribution of a denoising prediction and its clean target. Fix the pooled feature map g, and assume its two induced distributions have finite second moments. Then*

$$W_2^2\left(g_{\#}P_{\widehat{\mathcal{M}}}, g_{\#}P_{\mathcal{M}}\right) \le \mathbb{E}\|g(\widehat{\mathcal{M}}) - g(\mathcal{M})\|^2. \tag{35}$$

If the time weighting satisfies $w(t) \ge w_{\min} > 0$ over the considered time distribution, the right-hand side is at most $\mathbb{E}\mathcal{L}_{\mathrm{PERC}}/w_{\min}$.

Proof. Push the prediction–target coupling forward through (g, g). The result is a valid coupling of the two feature distributions. Since W_2^2 is the infimum of the expected squared distance over all such couplings, it is no greater than the cost of this particular pairing. The weighted statement follows pointwise from $w_{\min}\|g(\widehat{\mathcal{M}}) - g(\mathcal{M})\|^2 \le \mathcal{L}_{\mathrm{PERC}}$ and then taking expectations. □

Corollary D.5 (Geometric error under local non-collapse) *If, for the paired molecules in a specified local region, g additionally satisfies*

$$\|g(\mathcal{M}) - g(\mathcal{M}')\| \ge c\, d_{\mathrm{mol}}(\mathcal{M}, \mathcal{M}'), \qquad c > 0, \tag{36}$$

then

$$\mathbb{E}d_{\mathrm{mol}}(\widehat{\mathcal{M}}, \mathcal{M})^2 \le c^{-2}\mathbb{E}\|g(\widehat{\mathcal{M}}) - g(\mathcal{M})\|^2. \tag{37}$$

Proof. Square the assumed lower bound for each pair and take expectations. □

The corollary relies on a lower (non-collapse) bound on g rather than an upper Lipschitz bound, which is why the effective-rank and pairwise-distance diagnostics in Sections B.2.1 and B.3 are relevant: they measure whether the refined feature map avoids collapse.

D.6 A Local Quadratic Optimization Surrogate

Proposition D.6 (Isotropic regularization of a quadratic surrogate) *Consider direct optimization of a feature vector u with local quadratic objective $F(u) = \frac{1}{2}(u - u_*)^T H(u - u_*)$, where H is positive definite. Let $m = \lambda_{\min}(H) > 0$ and $L = \lambda_{\max}(H)$. Adding $\frac{\lambda}{2}\|u - e\|^2$, with a fixed target e and $\lambda \ge 0$, changes the Hessian to $H + \lambda I$ and gives*

$$\kappa(H + \lambda I) = \frac{L + \lambda}{m + \lambda} \le \frac{L}{m} = \kappa(H). \tag{38}$$

Proof. Each eigenvalue increases by λ. The inequality follows from $m(L + \lambda) \le L(m + \lambda)$. It is strict when $L > m$ and $\lambda > 0$. □

The proposition is stated in feature space. In parameter space, with $J = \nabla_\theta u_\theta$, the alignment term contributes $\lambda J^T J$ to the Gauss–Newton Hessian, so the same isotropic effect holds to the extent that $J^T J$ is well conditioned on the relevant subspace.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction match the method and evaluations in Sections 3.2, 4 and 4.4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Scope, computational overhead, and theoretical limitations are discussed in Sections B.9, D and 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Assumptions and complete proofs are provided in Section D.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The method and data splits are described, and the key hyperparameters are described in the main paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The datasets are public, but the manuscript provides no code repository or executable reproduction instructions.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: They have been described.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Main results are computed under multi seed.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Main generator costs are reported in Section B.9.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The study uses public molecular data and discusses potential misuse in Section C.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Benefits, dual-use risks, and mitigation measures are discussed in Section C.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The work does not release high-risk datasets or models.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: –

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new dataset or pretrained model is released.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The study involves no crowdsourcing or human participants.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The study involves no human participants.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not components of the proposed method.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.