MOLWORLD: MOLECULE WORLD MODELS FOR ACTIONABLE MOLECULAR OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Molecular optimization in drug discovery aims to discover molecules with improved target properties, but practical lead optimization often requires more than high predicted scores. A useful candidate should also be actionable: it should be reachable from known molecules through a sequence of local structural modifications, providing explicit structural references for interpreting property changes within an evolving chemical series. Existing de novo and single-molecule optimization methods do not explicitly model such reachability, especially when both the target molecules and the intermediate molecules connecting them to known compounds are unknown. In this work, we formulate actionable molecular optimization as iterative expansion of a molecule-transfer graph, where nodes are molecules and edges encode matched molecular pair (MMP) relations representing localized structural differences. We propose MOLWORLD, a molecule world model-guided framework that uses connected local subgraphs as anchor contexts for graph expansion. Given the current graph and a candidate context, a latent molecule world model predicts the resulting local structural expansion. Its dynamics are shared across property objectives, while a property-specific scorer evaluates predicted expansions to guide context selection. A context-conditioned generator then uses the molecular structures and MMP relations within selected contexts to propose candidates, learning from local graph-completion tasks. Candidates are evaluated, and those connected through verified MMP relations are incorporated into the graph, preserving reachability from the initial set. The updated graph serves as the state for subsequent optimization. Experiments on property optimization and docking-based tasks show that MOLWORLD discovers high-property molecules while maintaining stronger structural connectivity, supporting actionable and sequential molecular design.

1 INTRODUCTION

Molecular optimization aims to design molecules with improved target properties, such as activity, selectivity, docking affinity, or drug-likeness Nicolaou et al. (2007). Recent machine learning methods have made substantial progress by formulating this problem as de novo molecular generation Kusner et al. (2017); Liu et al. (2018); Jin et al. (2018); Zhou et al. (2019); Guimaraes et al. (2017); You et al. (2018) or property-guided optimization from existing molecules He et al. (2021; 2022); Jin et al. (2020); Fu et al. (2020); Wu et al. (2024). These formulations are effective for discovering high-scoring candidates under a property oracle, but practical lead optimization often requires more than high predicted property values. A useful candidate should also be *actionable*: it should be reachable from existing molecules through a sequence of local structural modifications, providing explicit structural references for interpreting property changes and planning further exploration within a chemical series.

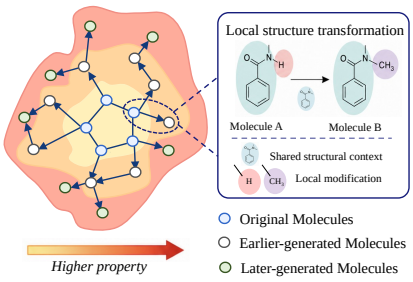


Figure 1: Molecule-transfer graph expansion. Generated molecules are progressively inserted into the graph through verified MMP relations, enabling sequential exploration toward higher-property regions while preserving reachability from the initial molecule series.

In lead optimization, molecules are commonly organized around analogue series, where compounds share conserved structural contexts and differ through local substitutions or modifications Naveja

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

& Vogt (2021); Stacey et al. (2025); Raymond et al. (2009); Yoshimori & Bajorath (2022). Such relationships naturally induce a *molecule-transfer graph*: nodes are molecules, and edges represent matched molecular pair (MMP) relations Dossetter et al. (2013); Griffen et al. (2011). An MMP consists of two molecules that retain the same structural context while differing by a well-defined, localized fragment replacement, as illustrated in the inset of Figure 1. Each edge therefore provides an explicit structural comparison for interpreting property differences, while successive edges make the progression from existing compounds to new candidates traceable. Such a sequence can extend beyond the immediate structural neighborhood of any single starting molecule.

We therefore study *actionable molecular optimization*: as illustrated in Figure 1, given an initial set of related molecules, the goal is to discover molecules with strong properties while ensuring that they are reachable from the initial set through paths of MMP relations. After a new molecule is generated and connected to the current series, it becomes part of the molecular world from which future molecules can be designed. Optimization therefore proceeds through iterative expansion of this molecular world, with the updated graph serving as the state for subsequent exploration.

Achieving this goal presents three challenges. First, the existing graph contains rich information about molecular structures and local transformations, but its size makes directly conditioning generation on the entire graph impractical. A generation context must retain useful structural relationships while remaining manageable as the graph grows. Second, the graph contains many potential expansion regions, and the properties of existing molecules do not directly reveal the quality of new molecules that can be generated from them. Evaluating every possible expansion would require costly generation and property-oracle calls. Third, generating candidates that can extend the graph requires exploiting both the structures of related molecules and the MMP relations linking them. The generator must use these patterns to propose candidates connected to the current graph.

Motivated by this view, we propose MOLWORLD, a molecule world model-guided framework for actionable molecular optimization. We use connected local subgraphs, termed anchor contexts, to provide compact generation contexts that retain related molecular structures and their MMP relations. Building on this representation, MOLWORLD couples two modules. First, molecule world model-guided anchor context selection anticipates the outcomes of candidate expansions before generation is executed. A latent molecule world model predicts the local structural expansion associated with each candidate context, and a property-specific scorer combines this prediction with the current graph representation to rank the contexts. The structural dynamics model is shared across property objectives. Second, a context-conditioned molecule generator jointly uses the molecular structures and MMP relations within a selected context to propose candidates that complete or extend its local molecule-transfer pattern. Candidates are evaluated, and those connected through verified MMP relations are incorporated into the graph. The updated graph serves as the state for the next iteration.

We evaluate MOLWORLD on molecular property optimization and docking-based optimization tasks. The results show that MOLWORLD discovers high-property molecules while maintaining stronger graph connectivity than competing methods. Our contributions are summarized as follows:

- We formulate actionable molecular optimization as a reachability-constrained problem and propose MOLWORLD, a framework that optimizes molecular properties through iterative expansion of a molecule-transfer graph. By maintaining MMP paths to the initial molecular set, the framework makes high-property candidates traceable to the starting molecular series and provides structural references for interpreting property changes along local modifications.
- We introduce a molecule world model that simulates the evolution of a molecular world under context-conditioned generation. Given the current molecule-transfer graph and an anchor context, it anticipates the resulting structural expansion in latent space. Coupled with a property-specific scorer, it guides property-directed search by evaluating expansions before execution.
- We formulate context-conditioned molecular generation as local molecule-transfer graph completion, using multiple related molecules and their MMP relations as a joint generation context. The generator learns to propose candidates from the structural patterns represented by this context, supporting graph expansion through subsequent MMP validation.

2 RELATED WORK

Machine Learning-based Molecular Optimization. De novo molecular optimization generates molecules with desired properties without a specified starting compound. Approaches include continuous latent-space optimization Kusner et al. (2017); Liu et al. (2018); Jin et al. (2018), reinforcement learning Zhou et al. (2019); Guimaraes et al. (2017); You et al. (2018); Olivecrona et al. (2017); Munson et al. (2024); Wang et al. (2025); graph-based autoregressive or flow models

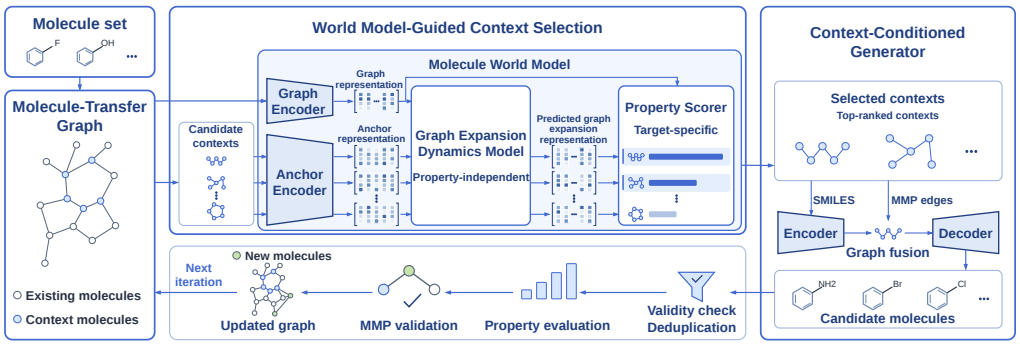


Figure 2: Overview of the MOLWORLD framework. The molecule world model predicts graph expansion conditioned on candidate anchor contexts, and a property-specific scorer ranks these contexts for selection. The context-conditioned generator then proposes new molecules. After validity checks, deduplication, and property evaluation, candidates connected to the current graph through verified MMP relations are incorporated for the next iteration.

Shi et al. (2020); Zang & Wang (2020); and diffusion and structure-aware models that generate 2D or 3D molecules under geometric or target-conditioned constraints Hoogeboom et al. (2022); Xu et al. (2023); Peng et al. (2022); Guan et al. (2023); Schneuing et al. (2024); Huang et al. (2024). In contrast, **pairwise molecular optimization** seeks improved analogs of a lead molecule while preserving similarity, scaffolds, or key substructures. Approaches include molecular translation using paired textual representations He et al. (2021; 2022), graph-to-graph translation (Jin et al., 2019b;a; 2020; Fu et al., 2020; Xie et al., 2021), and LLM-based approaches Wu et al. (2024); Ye et al. (2025). Retrieval-based methods such as RetMol (Wang et al., 2022) also condition generation on multiple exemplar molecules. MOLWORLD instead organizes molecular contexts within an evolving MMP graph and uses a molecule world model to predict graph expansion for context selection.

Additional related work on World Models is given in Appendix A.

3 METHOD

Given an initial molecular set and a target property, we propose MOLWORLD, a framework for actionable molecular optimization through iterative expansion of a molecule-transfer graph. The framework generates high-property candidates while maintaining MMP paths to the initial set, and incorporates accepted molecules into the graph for subsequent optimization. Within this framework, a molecule world model predicts how generation from each candidate context would expand the graph, guiding anchor context selection. A context-conditioned generator then uses the molecular structures and MMP relations within the selected contexts to propose new molecules (Figure 2).

3.1 PROBLEM FORMULATION

Given an initial set of related molecules $\mathcal{M}_0$ and a target property oracle f, we seek new molecules with high property values that can be reached from $\mathcal{M}_0$ through successive local structural modifications. We use matched molecular pair (MMP) relations to specify these modifications: $m \sim m'$ indicates that m' can be obtained from m by a localized structural change that preserves their shared chemical context. A molecule x is *reachable* from $\mathcal{M}_0$ if there exists a finite transformation path

$$m_0 \to m_1 \to \cdots \to m_L = x, \qquad m_0 \in \mathcal{M}_0, \tag{1}$$

where $m_{\ell-1} \sim m_\ell$ for every $\ell = 1, \ldots, L$. Each generated molecule must be connected to the initial set through a path of MMP relations among initial and generated molecules. Our objective is to find a generated set $\mathcal{X}$ that maximizes

$$\max_{\mathcal{X}} \; J_f(\mathcal{X}) \tag{2}$$

subject to the reachability requirement above, where J_f measures the property quality of the generated set, such as its top-k property scores. This formulation supports both direct analogues of initial molecules and more distant candidates connected through successive local modifications.

3.2 OVERALL FRAMEWORK

Graph and anchor contexts. We initialize the molecule-transfer graph as $G_0 = (\mathcal{V}_0, \mathcal{E}_0)$, where $\mathcal{V}_0 = \mathcal{M}_0$ and edges encode MMP relations between molecules. We denote the graph at iteration t by $G_t = (\mathcal{V}_t, \mathcal{E}_t)$. Since the graph can already contain many molecules and continues to grow during

optimization, directly conditioning molecular generation on the entire graph becomes computationally impractical. We therefore condition generation on a connected local subgraph $Z = (V_Z, E_Z)$ of G_t, termed an *anchor context*. The molecular structures and MMP relations within Z capture shared structural features and local transformation patterns, providing a compact generation context. Each anchor context contains five connected molecules. The framework does not require a fixed initial graph size: any initial graph containing a connected component of at least five molecules can supply a valid generation context.

World model-guided context selection. At iteration t, we first construct a candidate pool $\mathcal{C}_t$ using connected-subgraph beam search, combining high-property seeds and neighbors with randomized proposals favoring less-visited molecules. Repeated-use penalties discourage revisiting the same contexts, balancing property-guided search with exploration. Construction details are provided in Appendix B.3. Given the current graph G_t and a candidate anchor context $Z \in \mathcal{C}_t$, the molecule world model, detailed in Section 3.3, predicts the graph expansion resulting from generation conditioned on Z. To assess this expansion for the target property, its property-specific scorer combines the prediction with the current graph representation to assign a score $s_t(Z)$. For a generation batch size B, we select the B highest-scoring contexts:

$$\mathcal{Z}_t = \underset{\mathcal{Z} \subseteq \mathcal{C}_t,\ |\mathcal{Z}|=B}{\arg\max} \sum_{Z \in \mathcal{Z}} s_t(Z). \tag{3}$$

Generation and graph expansion. The context-conditioned generator (detailed in Section 3.4) uses the molecular structures and MMP relations within the selected contexts to produce candidates $\mathcal{X}_t$. After validity checks and deduplication, all remaining candidates are evaluated by the property oracle. RDKit-based MMP matching identifies those connected to at least one molecule in G_t, forming the accepted set $\widetilde{\mathcal{X}}_t$. The graph is then updated as

$$G_{t+1} = \left(\mathcal{V}_t \cup \widetilde{\mathcal{X}}_t,\, \mathcal{E}_t \cup \mathcal{E}_t^{+}\right), \tag{4}$$

where $\mathcal{E}_t^{+}$ contains verified MMP edges linking accepted molecules to existing or other accepted molecules. Each accepted molecule extends an existing MMP path from the initial set. The updated graph supplies new anchor contexts for subsequent generation, completing the feedback loop in Figure 2. Detailed acceptance and edge-construction rules are provided in Appendix B.5.

3.3 Molecule World Model

The properties of molecules in an anchor context indicate promising regions of the graph, but do not directly reveal the quality of new molecules that generation from this context would produce. Evaluating every candidate context through generation and property evaluation would be costly. Following latent world-model approaches (Ha & Schmidhuber, 2018; Hansen et al., 2024), we learn to predict action-conditioned changes to the search state in representation space. Here, the state is the current molecule-transfer graph, an action is generation conditioned on an anchor context, and the predicted outcome represents the resulting structural expansion. Given the current graph G_t and a candidate anchor context $Z \in \mathcal{C}_t$, our molecule world model combines graph expansion prediction and property scoring as follows:

$$\begin{aligned} \text{State and context encoders:} \quad & h_t = E_G(G_t), \qquad a_Z = E_A(Z), \\ \text{Graph expansion dynamics model:} \quad & \widehat{\delta}_Z = \mathcal{W}_\phi(h_t, a_Z), \\ \text{Property scorer:} \quad & s_t(Z) = Q_\psi(h_t, \widehat{\delta}_Z). \end{aligned} \tag{5}$$

Here, E_G encodes the complete current graph G_t as h_t. Each anchor context Z specifies a context-conditioned generation action and is encoded by E_A as a_Z. Conditioned on both representations, the graph expansion dynamics model $\mathcal{W}_\phi$ predicts how the graph would expand under this action, representing the anticipated expansion in latent space as $\widehat{\delta}_Z$. The scorer Q_ψ combines this prediction with the current graph representation to assign the context a score $s_t(Z)$. The dynamics model learns structural expansion independently of the target property and is shared across property objectives, while the scorer is trained for the target property f.

For the anchor encoder E_A, we reuse the molecular encoding and graph-fusion modules of the context-conditioned generator, pretrained on a separate ChEMBL-derived dataset as described in Section 3.4. This connects the world model's action representation to the context used for generation. We train the remaining components of the molecule world model on an offline dataset $\mathcal{D}$ of molecular

graph expansions constructed from ChEMBL Zdrazil et al. (2024) using the pretrained generator. The dataset records molecular graphs, anchor contexts, and their observed expansion outcomes; its construction and train/validation/test splits are described in Appendix B.1.

To support graph expansion prediction, E_G should capture both node-level structural context and graph-level information. We therefore adapt masked latent prediction Baevski et al. (2022); Assran et al. (2023) to the initial graphs in $\mathcal{D}$, combining node and graph prediction with token-diversity regularization. During offline pretraining, a student graph encoder and prediction heads predict representations from graphs with masked node features, while an exponential-moving-average (EMA) target encoder supplies representations of the corresponding complete graphs. The objective is

$$\mathcal{L}_{\mathrm{enc}} = \frac{1}{|M|} \sum_{v \in M} \ell_{\mathrm{H}}(\widehat{u}_v, \mathrm{sg}[\bar{u}_v]) + \lambda_g \ell_{\mathrm{H}}\Big(\widehat{H}, \mathrm{sg}[\bar{H}]\Big) + \lambda_d \mathcal{L}_{\mathrm{div}}. \tag{6}$$

Here, M is the set of masked molecular nodes; $\widehat{u}_v, \widehat{H}$ are node and graph-token predictions, and $\bar{u}_v, \bar{H}$ are the corresponding layer-normalized target representations. The function ℓ_{H} is smooth-L_1 loss, sg denotes stop-gradient, and λ_g, λ_d are loss weights. The diversity term $\mathcal{L}_{\mathrm{div}}$ discourages redundant graph tokens, encouraging complementary structural information. After pretraining, the EMA target encoder is frozen and used as E_G. The graph encoder architecture, training details, and validation results are provided in Appendix B.2.

With both encoders fixed, the dynamics model learns to predict the graph expansion induced by generation conditioned on Z. Since the structural change from a single generation action is small relative to the current graph, we focus supervision on the added structure. The prediction target therefore encodes an expansion graph comprising accepted new molecules, their MMP edges, and the existing molecules to which they connect. The training objective is

$$\mathcal{L}_{\mathrm{dyn}}(\phi) = \mathbb{E}_{(G_t, Z) \sim \mathcal{D}} \left[\ell_{\mathrm{H}} \left(\mathcal{W}_\phi(E_G(G_t), E_A(Z)) \,,\ \frac{1}{R_Z} \sum_{r=1}^{R_Z} E_G\Big(\Delta G_{t,Z}^{(r)}\Big) \right) \right]. \tag{7}$$

Here, $\Delta G_{t,Z}^{(r)}$ is the r-th observed expansion graph, and R_Z is the number of recorded generations for the same state–context pair. The dynamics model architecture, target construction, training details, and validation results are provided in Appendix B.2.

Finally, we train the property-specific scorer Q_ψ using the current graph representations and the dynamics model's predicted expansion representations, keeping both encoders and the dynamics model fixed. Supervision reflects the property quality of the best newly accepted molecules. The training objective combines a best-action loss that favors the highest-quality actions within each training state with a pairwise ranking loss that learns their relative ordering. Supervision and loss definitions are provided in Appendix B.3.

3.4 Context-Conditioned Molecule Generation

Given a selected context $Z \in \mathcal{Z}_t$, the generator proposes candidates that locally complete or extend the molecule-transfer pattern represented by Z. For an anchor context $Z = (V_Z, E_Z)$, the generator models a conditional distribution:

$$p_\theta(x \mid Z) \approx p_{\mathrm{data}}\big(x \mid x \in \mathcal{N}_{\mathrm{local}}(Z)\big), \tag{8}$$

where $\mathcal{N}_{\mathrm{local}}(Z)$ denotes molecules that plausibly complete or extend the local molecule-transfer graph represented by Z. This formulation turns candidate generation from one-to-one molecular editing into local graph completion: instead of modifying a single molecule in isolation, the model learns to propose molecules compatible with the transformation patterns in the selected context.

We use an encoder-decoder architecture to parameterize this distribution, as illustrated in Figure 2. Each molecule $m_i \in V_Z$ is represented by its SMILES string. A shared molecular encoder first maps the molecules in the context into node representations. A graph fusion module then performs message passing over the context edges E_Z to obtain a structure-aware context representation:

$$C_Z = \mathrm{Pool}\left(\left\{ \mathrm{GNN}_\theta\big(\{\mathrm{Enc}_\theta(\mathrm{SMILES}(m_i))\}_{m_i \in V_Z}, E_Z\big)_i \right\}_{m_i \in V_Z} \right). \tag{9}$$

Here, C_Z encodes the molecular identities of the context nodes and their local transformation relations. The molecular encoding and graph-fusion modules are shared with the anchor encoder E_A: the generator uses the fused token sequences for decoding, while E_A averages them within each molecule

to obtain the action representation a_Z. A Transformer decoder conditions on C_Z for autoregressive SMILES generation:

$$p_\theta(x \mid Z) = \prod_{\ell=1}^{L} p_\theta(x_\ell \mid x_{<\ell}, C_Z). \tag{10}$$

Sampling from this distribution yields multiple candidate molecules conditioned on the same selected context. The generator is trained in two stages, first using pairwise molecular transformations to pretrain the Transformer encoder-decoder, then using subgraph-based contexts to train all modules. Training details are provided in Appendix B.4.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We pretrain the context-conditioned generator and the molecule world model on ChEMBL (Zdrazil et al., 2024), which provides a large-scale collection of bioactive molecules for learning local molecular transformation patterns, and evaluate MOLWORLD on ZINC250 (Sterling & Irwin, 2015) and PMV21 (Vu et al., 2021). For each dataset, we construct a molecule-transfer graph whose nodes are molecules and whose edges encode MMP relations identified using RDKit-based molecular fragmentation. Implementation details and hyperparameters are provided in Appendix B.

Task 1: Property Optimization. We adopt the oracle budgets during optimization and running top-k AUC evaluation of the Practical Molecular Optimization (PMO) benchmark (Gao et al., 2022) on both ZINC250 and PMV21. Given a set of molecules, the task is to iteratively generate new molecules under a fixed oracle budget to improve a target property. We consider four target properties: GSK3β (glycogen synthase kinase-3 beta), QED (Bickerton et al., 2012), which measures drug-likeness, JNK3 (c-Jun N-terminal kinase-3), and DRD2 (dopamine receptor D2). For the three bioactivity tasks, pretrained predictive models from TDC are used as oracle functions. The oracle budget is set to 10,000 calls for ZINC250 and 1,000 calls for PMV21. Each experiment is repeated with three random seeds.

Task 2: Docking-Based Optimization. We further consider three protein-ligand docking tasks from TDC Graff et al. (2021), which correspond to structure-based drug design scenarios. The proteins include DRD3 (PDB ID: 3PBL), EGFR (PDB ID: 2RGP), and Adenosine A2A receptor (PDB ID: 3EML). For each task, molecules are evaluated using the TDC docking oracle implemented through the PyScreener AutoDock Vina backend Eberhardt et al. (2021). The oracle returns the best Vina binding affinity among generated poses, where lower docking scores indicate stronger predicted binding. All methods are evaluated under the same docking protocol and oracle budget. Due to the higher computational cost of docking evaluation, we restrict the oracle budget to 1,000 calls. The full docking protocol, including receptor preparation, ligand conformer generation, docking box settings, and Vina parameters, is provided in Appendix C.3.

Compared Methods. We compare MOLWORLD with nine baselines for property optimization: GraphGA (Jensen, 2019), REINVENT (Olivecrona et al., 2017), Augmented Memory (Guo & Schwaller, 2023), GP-BO (Tripp et al., 2021), MOLLEO (Wang et al., 2025), NFBO (Lee et al., 2025b), GenMol (Lee et al., 2025a), RetMol (Wang et al., 2022), and InVirtuoGen (Kaech et al., 2026). For docking-based optimization, we compare against GraphGA and MOLLEO, both previously evaluated on the same three docking targets under a 1,000-call oracle budget (Wang et al., 2025). All methods are evaluated under the same oracle budgets during optimization. Oracle evaluations of candidates that fail the MMP reachability criterion also count toward the budget for every method. Further details on the baseline methods are provided in Appendix C.2.

Evaluation Metrics. For both tasks, a generated molecule is considered reachable if it forms an MMP edge with an initial molecule or a previously accepted molecule. Accepted molecules are added to the graph after each generation batch. For property optimization, we report AUC@10 and AUC@100 under this criterion, together with the isolated node ratio (Iso). Iso is the fraction of valid, scored unique generated molecules that do not connect to the initial graph under this reachability criterion. AUC@k measures the average running top-k property score over the oracle budget, with higher values indicating better optimization performance. For docking-based optimization, we report the same three metrics, computing AUC from docking scores capped at zero, $\bar{s}_{\mathrm{dock}} = \min(0, s_{\mathrm{dock}})$, where s_{dock} is the Vina docking score. Lower docking AUC and lower Iso indicate better performance.

4.2 MAIN RESULTS

Task 1: Property Optimization. Table 1 reports mean property optimization results; results with standard deviations are provided in Appendix C.4.

Table 1: Property optimization on ZINC250 (10,000 oracle calls) and PMV21 (1,000 calls). Bold and underlined values denote the best and second-best results, respectively.

	Method	GSK3β			QED			JNK3			DRD2		
		AUC_{10} ↑	AUC_{100} ↑	Iso↓	AUC_{10} ↑	AUC_{100} ↑	Iso↓	AUC_{10} ↑	AUC_{100} ↑	Iso↓	AUC_{10} ↑	AUC_{100} ↑	Iso↓
ZINC250	GraphGA	0.3963	0.2589	0.8792	0.9304	0.8844	0.8092	0.5056	0.4081	0.8233	0.6163	0.3150	0.8772
	REINVENT	0.6564	0.5338	0.6689	0.9392	0.9062	0.6274	0.7574	0.7097	0.3953	0.8903	0.8381	0.5380
	Aug-Mem	0.6175	0.4400	0.9091	0.9366	0.8931	0.7300	0.7201	0.6445	0.7495	0.8552	0.7426	0.7895
	GP-BO	0.8195	0.7198	0.6106	0.9231	0.8745	0.9330	0.3509	0.3050	0.3230	0.9083	0.8336	0.3163
	MOLLEO	0.8904	0.8090	0.4642	0.9299	0.8751	0.7956	0.7759	0.7106	0.6214	0.7474	0.3679	0.8543
	NFBO	0.8316	0.6026	0.9623	0.9384	0.8856	0.7273	0.6459	0.5559	0.6462	0.9179	0.7823	0.7489
	GenMol	0.3360	0.1083	0.9901	0.9103	0.8266	0.8720	0.0003	0.0000	1.0000	0.9860	0.9522	0.4267
	RetMol	0.5112	0.4687	0.4019	0.9457	0.9396	0.3427	0.5422	0.4656	0.1996	0.9484	0.8536	0.4884
	InVirtuoGen	0.9012	0.7397	0.7102	0.9288	0.8646	0.8655	0.6833	0.5412	0.7918	0.6805	0.1792	0.7820
	MolWorld	**0.9576**	**0.9044**	**0.0460**	**0.9481**	**0.9470**	**0.0438**	**0.8225**	**0.7770**	**0.1105**	**0.9943**	**0.9819**	**0.0202**
PMV21	GraphGA	0.2751	0.1856	0.6057	0.7286	0.5408	0.8007	0.1490	0.0982	0.5527	0.7722	0.4403	0.6247
	REINVENT	None	None	1.0000	None	None	1.0000	None	None	1.0000	None	None	1.0000
	Aug-Mem	None	None	1.0000	None	None	1.0000	None	None	1.0000	None	None	1.0000
	GP-BO	0.2426	0.1512	0.6597	0.6908	0.5429	0.6237	0.1364	0.0858	0.6420	0.7674	0.4145	0.5930
	MOLLEO	0.3008	0.1936	0.6603	0.7880	0.5649	0.7903	0.1744	0.1114	0.5347	0.8019	0.4521	0.7047
	NFBO	0.2733	0.1082	0.8463	0.6115	0.3218	0.8445	0.1662	0.0951	0.6734	0.6783	0.3092	0.7897
	GenMol	0.3138	0.0538	0.9810	0.8010	0.3158	0.9477	0.2315	0.0427	0.9717	0.8659	0.3765	0.9123
	RetMol	0.2037	0.0866	0.8917	0.6290	0.4175	0.9000	0.1122	0.0529	0.8930	0.7774	0.5135	0.6157
	InVirtuoGen	0.2287	0.1284	0.9000	0.6113	0.4287	0.8973	0.1321	0.0741	0.9000	0.4878	0.2112	0.7478
	MolWorld	**0.4225**	**0.3490**	**0.0493**	**0.8892**	**0.8194**	**0.0744**	**0.2710**	**0.2175**	**0.0437**	**0.9018**	**0.8205**	**0.0990**

Across both ZINC250 and PMV21, MOLWORLD achieves the strongest overall performance, obtaining the best or highly competitive AUC scores across most tasks. As de novo generation methods, REINVENT and Augmented Memory do not explicitly enforce connectivity to the initial graph. Although some of their generated molecules satisfy the MMP connectivity constraint on ZINC250, none do on PMV21; their corresponding AUC entries are therefore reported as None. MOLWORLD achieves the lowest isolated node ratio across all eight dataset–property combinations, indicating that the generated molecules remain well integrated into the molecule-transfer graph through valid local transformations.

Task 2: Docking-Based Optimization. Table 2 reports docking-based optimization results on A2AR, DRD3, and EGFR. MOLWORLD achieves the lowest AUC@10 and AUC@100 across all three targets among the compared methods, demonstrating effective discovery of MMP-connected molecules with favorable docking scores over the search budget. It also consistently achieves the lowest isolated node ratio, indicating that its generated molecules are better integrated into the molecule-transfer graph. Together, these results show that MOLWORLD combines effective docking optimization with structurally connected molecular generation.

Table 2: Docking optimization with 1,000 oracle calls (lower is better).

Method	A2AR			DRD3			EGFR		
	AUC@10↓	AUC@100↓	Iso↓	AUC@10↓	AUC@100↓	Iso↓	AUC@10↓	AUC@100↓	Iso↓
GraphGA	-9.473	-8.237	0.765	-8.644	-7.515	0.828	-9.069	-7.794	0.799
MOLLEO	-9.952	-8.296	0.779	-9.055	-7.663	0.807	-9.209	-7.826	0.823
MOLWORLD	**-10.351**	**-8.908**	**0.121**	**-9.687**	**-8.646**	**0.119**	**-9.502**	**-8.330**	**0.159**

4.3 UNDERSTANDING MOLWORLD

Score Distribution and Optimization Dynamics. To further analyze the quality of generated molecules, we visualize the score distributions of reachable molecules under a fixed oracle budget on PMV21 (Figure 3). The plotted distributions show that MOLWORLD generates more molecules in high-score regions, particularly on QED and DRD2. This illustrative view complements top-k AUC metrics by showing molecule counts across the score range. We further analyze the optimization process using the top-10 running average score as a function of oracle calls (Appendix C.5). MOLWORLD converges faster on several tasks, particularly GSK3β and JNK3 on ZINC, while remaining comparable to strong baselines on the remaining tasks.

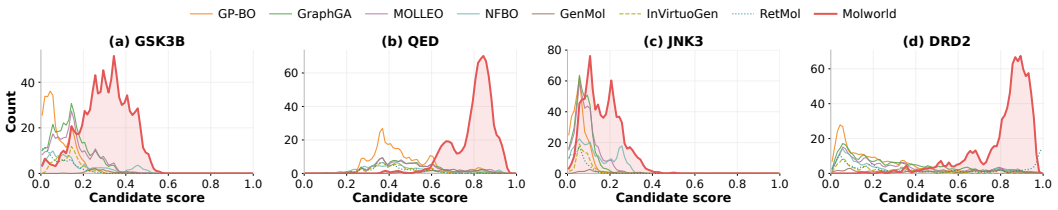


Figure 3: Property score distributions of MMP-connected molecules on PMV21. Curves show smoothed molecule counts across score values under a 1,000-call oracle budget.

Ablation Study. We examine the contributions of molecule world model-guided anchor context selection, context-conditioned generation, and iterative graph update through five variants (Figure 4). **Random Anchor Selection** randomly selects contexts from the same candidate pool, while **Mean-Property Selection** ranks them by the mean observed property value of their constituent molecules. These variants test whether the candidate construction procedure and existing property observations are sufficient for effective selection. **Direct Context Ranker** replaces the molecule world model and its property scorer with a learned ranker that directly takes the current graph and anchor representations as input. It uses the same supervision and data split, with a comparable trainable head size, testing whether directly learning to rank contexts is sufficient without a molecule world model. These selection variants share the context-conditioned generator. **Fixed Search Graph** retains the learned selection and generation modules but prevents newly generated molecules from becoming part of subsequent generation contexts. **STONED Generator** replaces the context-conditioned generator with random SELFIES mutation of a molecule sampled from the selected context. To examine the influence of candidate-pool construction, we additionally evaluate MOLWORLD, Random Anchor Selection, Mean-Property Selection, and Direct Context Ranker using a pool of randomly sampled connected five-molecule contexts. This replaces the original heuristic-guided pool construction while retaining each method's context-selection strategy. All variants use the same setup.

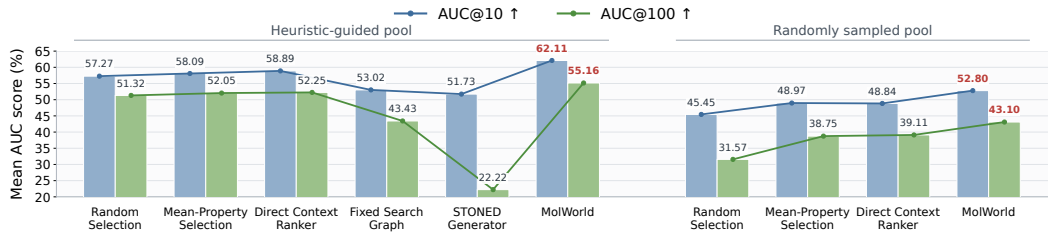


Figure 4: Property optimization ablations on PMV21, averaged across GSK3β, QED, JNK3, and DRD2, with 1,000 oracle evaluations per task.

As shown in Figure 4, MOLWORLD achieves the highest average AUC@10 and AUC@100 across the four properties. Its improvement over Direct Context Ranker supports predicting structural expansion before assessing property quality, rather than ranking contexts directly. The performance drop with Fixed Search Graph highlights the importance of reusing newly accepted molecules as anchors for further exploration. Replacing our generator with STONED reduces average AUC@100 from 0.5516 to 0.2222, supporting the contribution of the learned context-conditioned generator to optimization performance. MOLWORLD also achieves the highest average AUC@10 and AUC@100 among the compared selection strategies under randomly sampled candidate pools, indicating that its selection advantage persists when heuristic pool construction is removed.

Generated Molecule Distribution. Figure 5 shows a UMAP projection of initial and generated molecules (left) and oracle scores over generation order (right) on GSK3β.

In the UMAP projection, early generated molecules largely overlap with the initial set, while later molecules gradually expand into new regions of chemical space. The 10-step moving-average curve shows improving oracle scores over generation order. Appendix C.6 shows similar patterns on QED, JNK3, and DRD2. Anchor contexts quickly shift from initial to newly generated molecules after the early iterations, enabling MOLWORLD to reuse its discoveries for further graph expansion toward higher-scoring regions (Appendix C.7).

Qualitative Visualization of Sequential Molecular Optimization. Figure 6 presents initial molecules ($O1$–$O3$) and generated candidates across four iterations of QED optimization. The upper panels compare the selected sequence $O1$, $G1$, $G4$, $G7$, and $G9$, with QED scores of 0.827,

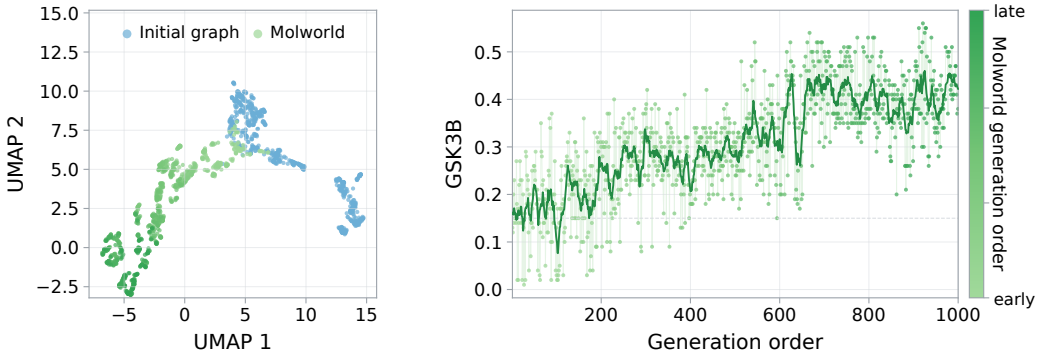


Figure 5: Generated molecule distribution and oracle-score progression on GSK3β.

0.835, 0.895, 0.922, and 0.936, respectively. The selected molecules retain an indole core and an N-methylpiperidine moiety, while differing in their terminal groups, N-substituents, and side-chain attachment patterns. For the depicted structures, the descriptor contributions in the RDKit QED calculation help explain the score progression within this composite objective (Bickerton et al., 2012). Relative to $O1$, $G1$ replaces the terminal amide group with a trifluoromethyl group. The hydrogen-bond donor count decreases from two to one, improving its QED desirability term; less favorable changes in other terms partly offset this gain, yielding a modest net increase. $G4$ differs from $G1$ at several positions, including the substituent at the indole nitrogen and the amine-side-chain attachment. The absence of the alkyne motif removes a match to the structural-alert patterns used by RDKit QED, providing the largest positive contribution to this score increase. In $G7$, the indole N-substituent is a methyl group rather than the trifluoroethyl group in $G4$. The calculated logP decreases from 4.35 to 3.33, and the corresponding improvement in the QED lipophilicity term is the main contributor to the higher score. $G9$ additionally differs from $G7$ in aryl fluorination and the attachment position and linker length of the aminoalkyl side chain. Its further QED increase is driven primarily by a more favorable molecular-weight term, with a smaller favorable contribution from calculated logP. The score increases reflect the QED objective and do not, by themselves, establish improvements in biological activity or experimental developability.

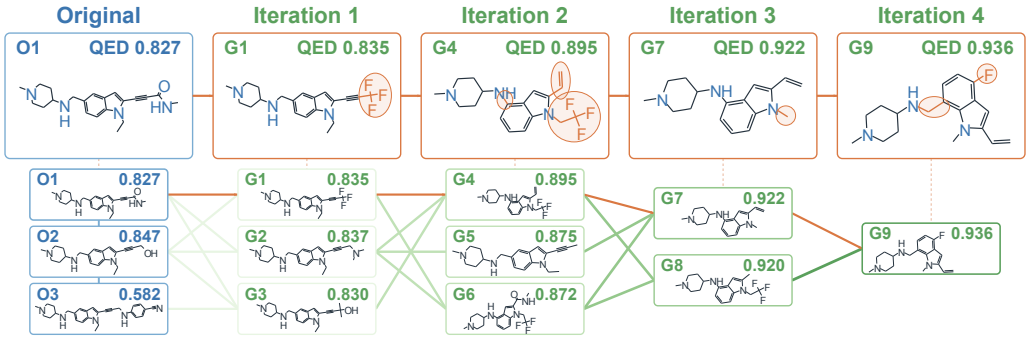


Figure 6: Sequential QED optimization with MOLWORLD: the molecular graph (bottom) and a selected sequence (top). Orange circles highlight selected structural differences.

5 CONCLUSION

We introduced MOLWORLD, a molecule world model-guided framework that optimizes molecular properties by expanding a molecule-transfer graph while preserving MMP reachability from the initial molecular set. Its molecule world model predicts graph expansion from candidate anchor contexts in latent space, with structural dynamics shared across properties. A property-specific scorer evaluates these predictions to guide anchor context selection. The context-conditioned generator uses molecular structures and MMP relations in the selected contexts to propose molecules, which enter the graph through verified MMP connections. Experiments on property optimization and docking-based optimization show that MOLWORLD discovers high-scoring molecules while maintaining stronger connectivity than competing methods. Ablation studies and further analyses support the effectiveness of its components.

AI USE STATEMENT

We used generative AI tools for manuscript editing and coding assistance, including debugging and scripts for processing and visualizing experimental results.

ETHICS STATEMENT

This research aims to support drug discovery through computational molecular optimization. The experiments evaluate molecular candidates using computational property predictors and docking oracles. These evaluations support candidate prioritization for further investigation, while practical applications require appropriate experimental validation and safety assessment.

REPRODUCIBILITY STATEMENT

Section 3 describes the method. Appendix B details the structural transition dataset construction and splits, model architectures, training objectives, hyperparameters, and anchor context selection. The Experimental Setup and Additional Experimental Details sections specify evaluation datasets, oracle budgets, the reachability criterion, metrics, and the docking protocol.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629. IEEE, 2023.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. In *International conference on machine learning*, pp. 1298–1312. PMLR, 2022.

G Richard Bickerton, Gaia V Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L Hopkins. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98, 2012.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Wing Yin Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. *OpenAI Technical Report*, 2024.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024.

Jingtao Ding, Yunke Zhang, Yu Shang, Yuheng Zhang, Zefang Zong, Jie Feng, Yuan Yuan, Hongyuan Su, Nian Li, Nicholas Sukiennik, et al. Understanding world or predicting future? a comprehensive survey of world models. *ACM Computing Surveys*, 58(3):1–38, 2025.

Alexander G Dossetter, Edward J Griffen, and Andrew G Leach. Matched molecular pair analysis in drug discovery. *Drug Discovery Today*, 18(15-16):724–731, 2013.

Jerome Eberhardt, Diogo Santos-Martins, Andreas F Tillack, and Stefano Forli. Autodock vina 1.2. 0: new docking methods, expanded force field, and python bindings. *Journal of chemical information and modeling*, 61(8):3891–3898, 2021.

Tianfan Fu, Cao Xiao, and Jimeng Sun. CORE: Automatic molecule optimization using copy & refine strategy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 638–645, 2020.

Wenhao Gao, Tianfan Fu, Jimeng Sun, and Connor Coley. Sample efficiency matters: a benchmark for practical molecular optimization. *Advances in neural information processing systems*, 35: 21342–21357, 2022.

David E Graff, Eugene I Shakhnovich, and Connor W Coley. Accelerating high-throughput virtual screening through molecular pool-based active learning. *Chemical science*, 12(22):7866–7881, 2021.

Ed Griffen, Andrew G Leach, Graeme R Robb, and Daniel J Warner. Matched molecular pairs as a medicinal chemistry tool: miniperspective. *Journal of medicinal chemistry*, 54(22):7739–7750, 2011.

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. In *International Conference on Learning Representations*, 2023.

Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843*, 2017.

Jeff Guo and Philippe Schwaller. Augmented memory: Capitalizing on experience replay to accelerate de novo molecular design. *arXiv preprint arXiv:2305.16160*, 2023.

Wes Gurnee and Max Tegmark. Language models represent space and time. In *International Conference on Learning Representations*, 2024.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Nick Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, volume 2024, pp. 47376–47405, 2024.

Jiazhen He, Huifang You, Emil Sandström, Eva Nittinger, Esben Jannik Bjerrum, Christian Tyrchan, Werngard Czechtizky, and Ola Engkvist. Molecular optimization by capturing chemist's intuition using deep neural networks. *Journal of Cheminformatics*, 13:26, 2021. doi: 10.1186/s13321-021-00497-0.

Jiazhen He, Eva Nittinger, Christian Tyrchan, Werngard Czechtizky, Atanas Patronov, Esben Jannik Bjerrum, and Ola Engkvist. Transformer-based molecular optimization beyond matched molecular pairs. *Journal of cheminformatics*, 14(1):18, 2022.

Emiel Hoogeboom, Víctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8867–8887. PMLR, 2022.

Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

Lei Huang, Tingyang Xu, Yang Yu, Peilin Zhao, Xingjian Chen, Jing Han, Zhi Xie, Hailong Li, Wenge Zhong, Ka-Chun Wong, et al. A dual diffusion model enables 3d molecule generation and lead optimization based on target pockets. *Nature Communications*, 15(1):2657, 2024. doi: 10.1038/s41467-024-46569-1.

Alan Ianeselli, Jewon Im, Eddie Cavallin, and Mark B. Gerstein. Generative world models to compute protein folding pathways. *bioRxiv*, 2025. doi: 10.1101/2025.03.26.645554.

Jan H Jensen. A graph-based genetic algorithm and generative model/monte carlo tree search for the exploration of chemical space. *Chemical science*, 10(12):3567–3572, 2019.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Hierarchical graph-to-graph translation for molecules. *arXiv preprint arXiv:1907.11223*, 2019a.

Wengong Jin, Kevin Yang, Regina Barzilay, and Tommi Jaakkola. Learning multimodal graph-to-graph translation for molecular optimization. In *International Conference on Learning Representations*, 2019b.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Hierarchical generation of molecular graphs using structural motifs. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4839–4848. PMLR, 2020.

Bowen Jing, Hannes Stärk, Tommi Jaakkola, and Bonnie Berger. Generative modeling of molecular dynamics trajectories. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Benno Kaech, Luis Wyss, Karsten Borgwardt, and Gianvito Grasso. Refine drugs, don't complete them: Uniform-source discrete flows for fragment-based drug discovery. In *International Conference on Learning Representations*, volume 2026, pp. 120938–120963, 2026.

Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *International conference on machine learning*, pp. 1945–1954. PMLR, 2017.

Seul Lee, Karsten Kreis, Srimukh Prasad Veccham, Meng Liu, Danny Reidenbach, Yuxing Peng, Saee Paliwal, Weili Nie, and Arash Vahdat. Genmol: A drug discovery generalist with discrete diffusion. *arXiv preprint arXiv:2501.06158*, 2025a.

Seunghun Lee, Jinyoung Park, Jaewon Chu, Minseo Yoon, and Hyunwoo J Kim. Latent bayesian optimization via autoregressive normalizing flows. *arXiv preprint arXiv:2504.14889*, 2025b.

Jessy Lin, Yuqing Du, Olivia Watkins, Danijar Hafner, Pieter Abbeel, Dan Klein, and Anca Dragan. Learning to model the world with language. *arXiv preprint arXiv:2308.01399*, 2023.

Qi Liu, Miltiadis Allamanis, Marc Brockschmidt, and Alexander Gaunt. Constrained graph variational autoencoders for molecule design. *Advances in neural information processing systems*, 31, 2018.

Brenton P. Munson, Michael Chen, Audrey Bogosian, Jason F. Kreisberg, Katherine Licon, Ruben Abagyan, Brent M. Kuenzi, and Trey Ideker. De novo generation of multi-target compounds using deep generative chemistry. *Nature Communications*, 15(1):3636, 2024. doi: 10.1038/s41467-024-47120-y.

José J Naveja and Martin Vogt. Automatic identification of analogue series from large compound data sets: methods and applications. *Molecules*, 26(17):5291, 2021.

Christos A Nicolaou, Nathan Brown, and Constantinos S Pattichis. Molecular optimization using computational multi-objective methods. *Current Opinion in Drug Discovery and Development*, 10 (3):316, 2007.

Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):48, 2017.

Xingang Peng, Shitong Luo, Jiaqi Guan, Qi Xie, Jian Peng, and Jianzhu Ma. Pocket2Mol: Efficient molecular sampling based on 3d protein pockets. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17644–17655. PMLR, 2022.

John W Raymond, Ian A Watson, and Abdelaziz Mahoui. Rationalizing lead optimization by associating quantitative relevance with molecular structure modification. *Journal of chemical information and modeling*, 49(8):1952–1962, 2009.

Arne Schneuing, Charles Harris, Yuanqi Du, Kieran Didi, Arian Jamasb, Ilia Igashov, Weitao Du, Carla Gomes, Tom L. Blundell, Pietro Lio, Max Welling, Michael Bronstein, and Bruno Correia. Structure-based drug design with equivariant diffusion models. *Nature Computational Science*, 4: 899–909, 2024. doi: 10.1038/s43588-024-00737-x.

Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. Graphaf: a flow-based autoregressive model for molecular graph generation. *arXiv preprint arXiv:2001.09382*, 2020.

Jessica Stacey, Baptiste Canault, Stephen D Pickett, and Valerie J Gillet. Visualising lead optimisation series using reduced graphs. *Journal of Cheminformatics*, 17(1):60, 2025.

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

Austin Tripp, Gregor NC Simm, and José Miguel Hernández-Lobato. A fresh look at de novo molecular design benchmarks. In *NeurIPS 2021 AI for Science Workshop*, 2021.

Binh Vu, Romyr Dominique, Hongju Li, Bruce Fahr, and Andrew Good. Methods and compounds for restoring mutant p53 function, 2021.

Haorui Wang, Marta Skreta, Cher Tian Ser, Wenhao Gao, Lingkai Kong, Felix Strieth-Kalthoff, Chenru Duan, Yuchen Zhuang, Yue Yu, Yanqiao Zhu, Yuanqi Du, Alan Aspuru-Guzik, Kirill Neklyudov, and Chao Zhang. Efficient evolutionary search over chemical space with large language models. In *International Conference on Learning Representations*, 2025.

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *European conference on computer vision*, pp. 55–72. Springer, 2024.

Zichao Wang, Weili Nie, Zhuoran Qiao, Chaowei Xiao, Richard Baraniuk, and Anima Anandkumar. Retrieval-based controllable molecule generation. *arXiv preprint arXiv:2208.11126*, 2022.

Zhenxing Wu, Odin Zhang, Xiaorui Wang, Li Fu, Huifeng Zhao, Jike Wang, Hongyan Du, Dejun Jiang, Yafeng Deng, Dongsheng Cao, Chang-Yu Hsieh, and Tingjun Hou. Leveraging language model for advanced multiproperty molecular optimization via prompt engineering. *Nature Machine Intelligence*, 6:1359–1369, 2024. doi: 10.1038/s42256-024-00916-5.

Yutong Xie, Chence Shi, Hao Zhou, Yuwei Yang, Weinan Zhang, Yong Yu, and Lei Li. MARS: Markov molecular sampling for multi-objective drug discovery. In *International Conference on Learning Representations*, 2021.

Minkai Xu, Alexander S. Powers, Ron O. 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*, pp. 38592–38610. PMLR, 2023.

Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023.

Geyan Ye, Xibao Cai, Houtim Lai, Xing Wang, Junhong Huang, Longyue Wang, Wei Liu, and Xiangxiang Zeng. DrugAssist: A large language model for molecule optimization. *Briefings in Bioinformatics*, 26(1):bbae693, 2025. doi: 10.1093/bib/bbae693.

Atsushi Yoshimori and Jürgen Bajorath. Deepas – chemical language model for the extension of active analogue series. *Bioorganic & Medicinal Chemistry*, 66:116808, 2022.

Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in neural information processing systems*, 31, 2018.

Chengxi Zang and Fei Wang. Moflow: an invertible flow model for generating molecular graphs. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 617–626, 2020.

Barbara Zdrazil, Eloy Felix, Fiona Hunter, Emma J Manners, James Blackshaw, Sybilla Corbett, Marleen De Veij, Harris Ioannidis, David Mendez Lopez, Juan F Mosquera, et al. The chembl database in 2023: a drug discovery platform spanning multiple bioactivity data types and time periods. *Nucleic acids research*, 52(D1):D1180–D1192, 2024.

Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific reports*, 9(1):10752, 2019.

A Additional Related Work

World Models. World models aim to build compact and actionable abstractions of the external world, enabling agents to reason about hidden dynamics, predict future states, and make decisions under uncertainty Ding et al. (2025). Existing studies can be broadly divided into two lines. The first line treats world models as latent representations for decision-making, especially in model-based reinforcement learning. Early neural world models learn compressed visual states and recurrent dynamics to support policy learning in imagination (Ha & Schmidhuber, 2018), while subsequent methods formulate control as learning latent dynamics and optimizing policies through imagined rollouts (Hafner et al., 2020; 2023; Hansen et al., 2024). With the rise of large language and multimodal models, this view has been extended to embedding world knowledge in foundation models (Gurnee & Tegmark, 2024). The second line views world models as future simulators. Recent video generation models show that large-scale generative models can capture temporal coherence, object persistence, and some physical regularities in visual sequences (Brooks et al., 2024). Recent work also moved to interactive simulators and embodied environments, where models generate future states conditioned on actions, language instructions, or multimodal observations (Yang et al., 2023; Bruce et al., 2024; Hu et al., 2023), especially in autonomous driving and robotics, where world models serve both as implicit scene-understanding modules and as simulators for counterfactual prediction, planning, and policy learning (Wang et al., 2024; Lin et al., 2023). Incorporating world models into scientific discovery is a promising yet under-explored area. MDGen models full molecular trajectories as a time series of 3D structures and supports forward simulation (Jing et al., 2024). Recent work further formulates protein-folding pathway prediction as a generative world-modeling problem (Ianeselli et al., 2025). This direction suggests that the scope of world models can extend from visual and embodied prediction to scientific simulation, where learning controllable future dynamics may accelerate biomedical discovery.

B Implementation Details

We provide implementation details for the structural transition dataset, molecule world model training, anchor context selection, property scoring, and generator training. Molecules are represented as nodes in a molecule-transfer graph, with MMP relations defining its edges. Offline collection uses reference-graph edges and RDKit MMP checks; online expansion uses RDKit MMP checks.

B.1 Structural Transition Dataset

Overview. We construct an offline dataset of molecular graph expansions from the ChEMBL molecule-transfer graph. The dataset supports graph-encoder pretraining, action-conditioned dynamics learning, and subsequent property-scorer training. Each state-action pair records the current graph G_t, an anchor context Z, and one or more observed generation outcomes. An outcome includes the raw candidates $\mathcal{X}_{t,Z}$, accepted new molecules $\widetilde{\mathcal{X}}_{t,Z}$, verified MMP edges, and the resulting graph G_{t+1}. Structural data collection is independent of the target property: property labels are attached after the graph-expansion trajectories have been collected.

Initial graphs and trajectories. We sample initial graphs from ChEMBL at four scales to expose the encoder and dynamics model to different graph sizes. The configured collection contains 150 root trajectories:

Initial graph size	Depth-ten trajectories	Depth-twenty trajectories
1,000 nodes	48	12
10,000 nodes	24	6
100,000 nodes	36	9
250,000 nodes	12	3

Each root defines an initial graph and a branching expansion trajectory. At every depth, a bounded frontier retains at most four states for further expansion. The retained states cover structurally productive, intermediate, randomly exploratory, and structurally diverse outcomes. This provides varied transitions without exhaustively expanding the trajectory tree.

Action sampling for data collection. At each active state, we construct a pool of 1,000 unique connected five-molecule anchor contexts. Seed proposals alternate between uniform sampling, low-visit sampling, and low-, medium-, and high-degree sampling. Each seed is expanded through existing graph edges until five distinct connected nodes are obtained. The generator input uses a breadth-first spanning tree of the selected nodes, with four undirected edges supplied bidirectionally.

We score training-data candidates using structural information:

$$b_{\text{struct}}(Z) = \frac{1}{|V_Z|} \sum_{v \in V_Z} \log(1 + \deg(v)) + \frac{0.35}{1 + \bar{c}_Z} + 0.15 \, \frac{\sigma_{\deg,Z}}{1 + \overline{\deg}_Z}, \tag{11}$$

where $\bar{c}_Z$ is the mean visit count, and $\overline{\deg}_Z$ and $\sigma_{\deg,Z}$ are the mean and standard deviation of node degrees within the context.

The pool is divided into low-, medium-, and high-score groups. We select three actions from the high group, two from the medium group, two from the low group, one random action, and two additional actions favoring diversity. Within the structural groups, selection combines normalized structural scores with diversity relative to already selected contexts. Diversity combines node-set Jaccard distance and a difference in mean degree, with weights 0.8 and 0.2, respectively.

This procedure collects ten actions per state without target-property information, unlike downstream property-guided candidate construction (Appendix B.3).

Generation and structural acceptance. The pretrained context-conditioned generator samples 16 sequences per action using nucleus sampling with $p = 0.95$, temperature 1.0, and maximum sequence length 64. Twenty percent of state-action pairs are executed with three random seeds; the remaining pairs use one seed. All seed outcomes of the same action share a state-action identifier.

Generated strings are canonicalized, checked for molecular validity, and deduplicated. Molecules already present in the current graph are excluded from the accepted-new-molecule set. For generated molecules belonging to the ChEMBL reference graph, insertion relations are retrieved from the reference edges. Outside that graph, MMP relations are computed with RDKit `rdMMPA`, using at most three cuts and 20 cut bonds, and a variable-fragment heavy-atom ratio capped at 0.33.

We apply the same connectivity-based acceptance rule used during online graph expansion (Appendix B.5). A valid new molecule is accepted if it forms a verified MMP edge with a molecule in the current graph. Accepted molecules are then incorporated to obtain the next graph state. Verified edges linking accepted new molecules to existing or other accepted molecules are retained. No learned link predictor participates in data collection. The stored records retain the structural evidence associated with added edges.

For actions with multiple seeds, one outcome is randomly chosen as the action's continuation representative. The other outcomes remain available for learning the distribution of results associated with that state-action pair.

Data splits and training cache. Root trajectories are assigned to training, validation, and test splits using a deterministic hash-based rule with nominal proportions $80/10/10$. All states, actions, and seed outcomes descended from a root inherit its split. Thus, transitions within one root trajectory cannot appear in different splits. The split is trajectory-based; it does not enforce molecule-disjoint or scaffold-disjoint partitions between independently sampled roots.

The cache used for dynamics and property-scorer training contains 124 training roots and 11 validation roots, totaling 6,035 action groups, 60,350 state-action pairs, and 84,490 action-seed outcomes. Test roots are excluded from this cache and from checkpoint selection. Initial graphs provide graph-encoder pretraining examples, while the recorded state-action outcomes provide dynamics supervision. Property scores of accepted new molecules provide scorer labels.

B.2 Molecule World Model Training

Training stages. We first pretrain a graph encoder using masked latent prediction on the initial graphs of the structural dataset. Its target encoder is then frozen and reused as E_G for both current graphs and local expansion outcomes. The anchor encoder E_A is extracted from the pretrained context-conditioned generator and remains frozen. We train the dynamics model $\mathcal{W}_\phi$ on recorded expansions, then freeze it when training the property scorer.

B.2.1 Graph Encoder

Molecular features and message passing. Each molecule is represented by a radius-two, 2,048-bit Morgan fingerprint and 23 RDKit descriptors. Active fingerprint bits are embedded and averaged to obtain a 192-dimensional fingerprint representation. A separate MLP maps the descriptors to 128 dimensions. Their concatenation is projected to a 256-dimensional molecular representation.

The descriptors are molecular weight, LogP, molar refractivity, TPSA, Labute ASA, heavy-atom count, heteroatom count, hydrogen-bond donor and acceptor counts, rotatable-bond count, total ring count, aromatic, aliphatic, and saturated ring counts, fraction of sp^3 carbons, NHOH count, NO count,

valence-electron count, Bertz complexity, Hall–Kier alpha, and the three Kappa indices. Descriptor preprocessing uses the fixed statistics stored with the reusable molecular-feature cache.

Two sparse message-passing layers incorporate molecule-transfer graph connectivity. Each layer combines a transformed self-representation with a transformed mean-neighbor representation, followed by a SiLU activation, dropout, a residual connection, and layer normalization.

Graph tokens. The encoder summarizes a graph with 16 tokens of dimension 256. Four tokens contain the mean, standard deviation, elementwise maximum, and degree-weighted mean of the node representations. The degree-weighted summary uses weights proportional to $\log(1 + \deg(v))$.

Twelve additional tokens are obtained by learned competitive attention pooling. Each node is assigned across the learned queries using a softmax with temperature 0.25; assignments are then normalized over nodes for each pooled token. The resulting tokens receive layer normalization. This combines global statistics with learned summaries of different regions of the graph.

Masked latent prediction. Following teacher–student latent prediction methods Baevski et al. (2022); Assran et al. (2023), we maintain an online encoder E_η and an exponential-moving-average target encoder $E_{\bar{\eta}}$. The online encoder receives a graph with the molecular content of 15% of nodes masked by a learned token, retaining edges. The target encoder receives the unmasked graph.

Let M denote the masked node set, u_v and H the online node and graph representations, and $\bar{u}_v$ and $\bar{H}$ their target counterparts. Separate heads P_n and P_g predict the node and graph representations. To discourage redundant graph tokens, we also penalize similarity between normalized predicted graph tokens q_i, with $K = 16$. The prediction losses, diversity penalty, and combined objective are

$$\begin{aligned} \mathcal{L}_{\text{node}} &= \frac{1}{|M|} \sum_{v \in M} \ell_{\text{H}}(P_n(u_v), \text{sg}[\text{LN}(\bar{u}_v)]) \,, \\ \mathcal{L}_{\text{graph}} &= \ell_{\text{H}}\big(P_g(H), \text{sg}[\text{LN}(\bar{H})]\big) \,, \\ \mathcal{L}_{\text{div}} &= \frac{1}{K(K-1)} \sum_{i \neq j} (q_i^\top q_j)^2, \\ \mathcal{L}_{\text{enc}} &= \mathcal{L}_{\text{node}} + 0.25\, \mathcal{L}_{\text{graph}} + 0.05\, \mathcal{L}_{\text{div}}. \end{aligned} \tag{12}$$

Here, sg denotes stop-gradient and LN denotes layer normalization. The smooth-L_1 loss averages the following penalty over representation coordinates:

$$\ell_{\text{H}}(x, y) = \text{mean}_k \begin{cases} \frac{1}{2}(x_k - y_k)^2, & |x_k - y_k| < 1, \\ |x_k - y_k| - \frac{1}{2}, & \text{otherwise.} \end{cases} \tag{13}$$

Only the online encoder and prediction heads receive gradient updates. The target encoder follows the exponential-moving-average update $\bar{\eta} \leftarrow m\bar{\eta} + (1 - m)\eta$, with momentum increasing from 0.9995 toward 1.0 during training.

Optimization and initialization. The final competitive-pooling encoder is initialized from an earlier graph JEPA checkpoint by transferring compatible parameters. The earlier checkpoint uses the same molecular encoder and two graph layers and is trained with node and graph prediction losses. The competitive-pooling stage adds multi-token pooling and token-diversity regularization.

Both stages use AdamW with learning rate 2×10^{-4}, weight decay 0.04, 20 epochs, and two warmup epochs followed by cosine learning-rate decay. Training balances the sampling of initial graphs across graph-size groups. Checkpoint selection uses the combined validation objective. The selected target encoder from the competitive-pooling stage is frozen and used as E_G.

Validation. We evaluate whether the trained encoder uses MMP relations by removing or shuffling the input edges on the validation graphs. We keep the model parameters, masked nodes, and target representations fixed across settings; the target encoder always receives the unmasked graph with its original edges. We report the masked-node prediction loss $\mathcal{L}_{\text{node}}$ and graph representation prediction loss $\mathcal{L}_{\text{graph}}$, averaged over validation graphs. Both measure smooth-L_1 error against the target encoder's representations, with lower values indicating more accurate predictions.

Removing or shuffling the edges increases masked-node prediction error to 3.45 and 3.23 times that obtained with the original edges, respectively (Table 3). This indicates that the encoder uses the observed MMP relationships to predict masked molecular representations.

B.2.2 Anchor Encoder

The anchor representation is extracted from the same molecular encoding and graph-fusion modules used by the context-conditioned generator. Molecular token representations receive graph-conditioned

Table 3: Graph encoder validation under input-edge perturbations.

Input edges	Node prediction loss ↓	Graph prediction loss ↓
Original MMP edges	**0.1048**	**0.0365**
Removed edges	0.3617	0.1269
Shuffled edges	0.3382	0.0429

updates through the generator's fusion operation. Averaging over valid sequence tokens separately for each anchor molecule gives $a_Z = E_A(Z) \in \mathbb{R}^{5\times 768}$, with one action token per molecule preserving the generator's learned contextual representation. The encoder is not retrained for dynamics learning. See Section 3.4 and Appendix B.4 for generator architecture and pretraining.

B.2.3 Dynamics Model

Local expansion target. For state-action pair (G_t, Z) and generation seed r, let $\Delta G^{(r)}_{t,Z}$ denote the recorded local outcome graph. Its nodes consist of all accepted new molecules and all existing molecules incident to the saved insertion edges. Its edges contain every saved new-to-old and new-to-new relation. Co-occurrence in this local graph does not add old-to-old edges.

The target and prediction are

$$\delta_Z = \frac{1}{R_Z}\sum_{r=1}^{R_Z} E_G\left(\Delta G^{(r)}_{t,Z}\right), \qquad \widehat{\delta}_Z = \mathcal{W}_\phi(E_G(G_t), E_A(Z)), \tag{14}$$

where $R_Z \in \{1, 3\}$. An outcome with no accepted molecules receives an all-zero token representation. The target is an encoding of the local expansion, rather than the difference between current-graph and next-graph embeddings.

The complete current graph and local outcome graph use exactly the same frozen encoder. New-node identity is used to construct the local outcome graph but is not supplied as an additional encoder feature. For each distinct current graph, message passing and graph pooling are recomputed; only graph-independent molecular features are cached.

Architecture. The dynamics model receives 16 current-graph tokens of dimension 256 and five anchor tokens of dimension 768. A learned projection maps the anchor tokens to dimension 256. These inputs are concatenated with 16 learned output queries, with positional and type embeddings distinguishing the three token groups.

A four-layer Transformer with eight attention heads and dropout 0.1 processes the sequence. An output MLP and layer normalization transform the output-query representations into the predicted expansion tokens $\widehat{\delta}_Z \in \mathbb{R}^{16\times 256}$.

Training. We minimize

$$\mathcal{L}_{\text{dyn}} = \frac{1}{|\mathcal{D}|}\sum_{(G_t,Z)\in\mathcal{D}} \ell_{\text{H}}(\widehat{\delta}_Z, \delta_Z), \tag{15}$$

where $\mathcal{D}$ contains training state-action pairs. Both encoders remain frozen.

The base configuration uses AdamW with learning rate 3×10^{-4}, weight decay 0.04, 30 epochs, and eight state groups per batch. Training uses a 5% warmup followed by cosine decay, gradient clipping at norm 1, and mixed precision. Checkpoint selection minimizes validation dynamics loss.

Validation. We evaluate the selected dynamics checkpoint by comparing its predicted expansion representations $\widehat{\delta}_Z$ with the observed expansion targets δ_Z on the validation set. Prediction error is measured by smooth-L_1 loss, averaged over representation coordinates, tokens, and state–context pairs. As a constant baseline, we predict the mean expansion target computed from the training set. We also evaluate the trained model with mismatched contexts by cyclically shifting anchor contexts within each validation batch while keeping graph inputs and targets unchanged.

Table 4: Validation of latent graph expansion predictions.

Prediction setting	Smooth-L_1 loss ↓
Mean training-set target	0.2083
Dynamics model with mismatched contexts	0.2148
Dynamics model with matched contexts	**0.0488**

With matched contexts, the dynamics model reduces prediction error by 76.57% relative to the mean-target baseline (Table 4). Its mean cosine similarity is 0.9421, computed between corresponding

predicted and target tokens and averaged over tokens and validation pairs. Higher error under context mismatch indicates that accurate expansion prediction depends on the anchor context.

B.3 Anchor Context Selection and Property Scoring

B.3.1 Candidate Construction During Optimization

The online candidate pool uses current property observations to focus search, while visit-aware random proposals and repeated-use penalties promote exploration. This differs from the property-independent action sampling used to collect the structural training dataset.

Search region and seeds. Only connected components containing at least five nodes are eligible. When more than 3,000 nodes are eligible, we attempt to restrict search to the subgraph induced by the 3,000 highest-property nodes. If no connected component in this subgraph can support a five-node anchor, we retain the original eligible search region.

Seed selection combines up to 100 highest-property nodes with up to 200 additional nodes sampled preferentially from less-visited candidates. To sample n such nodes, candidates are sorted by cumulative visit count with randomized tie-breaking. The first $5n$ candidates, or all available candidates if fewer, form a shortlist that is shuffled before selecting n nodes. Previously selected seeds are excluded from this additional sample.

Connected-subgraph beam search. Each partial context is expanded by adding an adjacent node. The expansion candidates consist of up to ten highest-property frontier nodes and ten additional frontier nodes selected through the low-visit sampling procedure. Equivalent node sets are deduplicated. Let $c_t(v)$ be the number of previously selected anchors containing v, and let $r_t(v)$ count its occurrences in selected anchors from the previous five iterations. For cached property values $f(v)$, define the context mean $\overline{f}_Z = |V_Z|^{-1} \sum_{v \in V_Z} f(v)$. Partial contexts are scored by

$$b_t(Z) = \overline{f}_Z - \lambda_v P_v(Z) - \lambda_o P_o(Z) - \lambda_r P_r(Z). \tag{16}$$

Writing $R_t(Z) = \{v \in V_Z : r_t(v) > 0\}$ for recently used nodes, the visit, overlap, and repeated-use penalties are

$$\begin{aligned} P_v(Z) &= \frac{1}{|V_Z|} \sum_{v \in V_Z} \log(1 + c_t(v)), \\ P_o(Z) &= \max(0,\, |R_t(Z)| - 3)\,, \\ P_r(Z) &= \frac{1}{\max\{1, |R_t(Z)|\}} \sum_{v \in R_t(Z)} \log(1 + r_t(v)). \end{aligned} \tag{17}$$

The empty sum gives $P_r(Z) = 0$ when no context node was used recently. The default coefficients are $\lambda_v = 0.02$, $\lambda_o = 0.03$, and $\lambda_r = 0.01$. At each depth, the beam retains at most 300 partial contexts. Search stops growing a context once it contains five nodes. Complete anchors already executed in earlier iterations are excluded from the candidate pool.

Adaptive penalties. The implementation adjusts penalties according to recorded generation yield. For an iteration with nonempty output but fewer than ten newly recorded candidates, each coefficient is multiplied by 1.25, up to respective maxima of 0.08, 0.12, and 0.04. For nonempty output of at least ten candidates, coefficients are multiplied by 0.95, bounded below by their defaults. Empty-output iterations do not update these coefficients.

Final selection. Every completed candidate is scored by the property scorer using its current-graph and predicted-expansion representations. Final selection takes the highest-scoring B candidates without additional MMR, property-based, or repeated-use reranking. Thus, heuristic scores determine candidate availability, while the learned scorer determines selection within the resulting pool.

The runner defaults to $B = 20$. If no explicit batch size is supplied, an adaptive setting can increase it to 50 when the preceding five iterations produce fewer than 200 recorded candidates. Explicit batch-size settings override this adaptation. Candidate-pool and batch sizes can also be set per run.

B.3.2 Property Scorer

Training labels. For generation seed r, let $\widetilde{\mathcal{X}}_{t,Z}^{(r)}$ denote the accepted new molecules for action Z, and let $f_{(k)}^{(r)}$ be their k-th highest property score. Missing entries are assigned zero. We define the seed-level label and aggregated action label as

$$y_t^{(r)}(Z) = \frac{1}{3} \sum_{k=1}^{3} f_{(k)}^{(r)}, \qquad y_t(Z) = \frac{1}{R_Z} \sum_{r=1}^{R_Z} y_t^{(r)}(Z). \tag{18}$$

For an action producing no accepted molecules, the label is zero. Keeping the denominator fixed at three makes the label sensitive to both property quality and accepted-molecule yield. The averaging is performed after computing each seed's label, rather than pooling molecules across seeds. Labels use only newly accepted molecules; properties of molecules already present in G_t do not contribute. Property-specific scorers share the structural encoder and dynamics model.

Architecture. The property scorer receives 16 current-graph tokens and 16 predicted-expansion tokens, each of dimension 256. These are concatenated with one learned query token. Type and positional embeddings distinguish the current graph, predicted expansion, and query. A three-layer Transformer with eight attention heads and dropout 0.1 processes the sequence, and an MLP maps the query output to the scalar action score $s_t(Z) = Q_\psi(h_t, \widehat{\delta}_Z)$. Scorer training uses dynamics predictions $\widehat{\delta}_Z$, not the observed outcome targets δ_Z. The raw anchor representation is not provided as a separate scorer input. The graph encoder, anchor encoder, and dynamics model remain frozen.

Ranking objectives. Losses compare actions belonging to the same current state. For a group with labels y_i and predicted scores s_i, define the action probabilities $p_i = \exp(s_i)/\sum_j \exp(s_j)$, the tied best actions $\mathcal{B} = \{i : y_i = \max_j y_j\}$, and the pairs with distinct labels $\mathcal{P} = \{(i,j) : i < j,\ y_i \neq y_j\}$. The best-action objective assigns equal target probability to tied best actions, while the pairwise objective weights comparisons by their label differences:

$$\begin{aligned}
\mathcal{L}_{\text{best}} &= -\frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} \log p_i, \\
\mathcal{L}_{\text{pair}} &= \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} w_{ij}\log(1+\exp[-\operatorname{sign}(y_i-y_j)(s_i-s_j)])\,, \\
w_{ij} &= \operatorname{clip}\left(\frac{|y_i-y_j|}{\sigma_y},\, 0.25,\, 4\right).
\end{aligned} \tag{19}$$

Here, σ_y is the standard deviation of training action labels, so larger label differences receive greater weight within fixed bounds. The two losses are averaged over eligible state groups and added with equal weights, $\mathcal{L}_{\text{score}} = \mathcal{L}_{\text{best}} + \mathcal{L}_{\text{pair}}$. Groups with fewer than two actions or no distinguishable label differences are omitted. Small tolerances handle ties and near-zero label differences.

Optimization and checkpoint selection. The base training configuration uses AdamW with learning rate 3×10^{-4}, weight decay 0.04, and $(\beta_1, \beta_2) = (0.9, 0.95)$. Training uses 24 epochs, 32 state groups per batch, a 5% warmup followed by cosine decay, and gradient clipping at norm 1.

Checkpoint selection uses validation normalized selection regret. For state t, let $i_t^\star = \arg\max_i s_i$ be the selected action. Its normalized regret is

$$\text{Regret}_t = \frac{\max_i y_i - y_{i_t^\star}}{\max_i y_i - \min_i y_i}. \tag{20}$$

States with no label variation are excluded. We select the checkpoint with the lowest mean validation regret to prioritize useful action selection over property-regression accuracy.

B.4 Generator Training Details

The context-conditioned generator is trained to complete local molecule-transfer patterns. Its goal is to generate a target molecule conditioned on a small context of related molecules, thereby learning chemically valid molecular edits and local transformation patterns within analogue series.

We train the generator in two stages. In the first stage, we initialize the sequence-to-sequence backbone from a molecular Transformer pretrained on pairwise molecular transformation data. This pretraining stage exposes the model to a broad range of molecular structures and local edits, and provides a general molecular generation prior. In the second stage, we insert a graph neural network fusion module and fine-tune the resulting hybrid model on graph-based subgraph completion tasks constructed from a ChEMBL-derived molecule-transfer graph. Each training instance consists of a local molecular context as input and one held-out related molecule as the target. This training scheme encourages the generator to combine broad molecular-edit knowledge from pairwise pretraining with local analogue-series pattern modeling from graph-based completion.

To construct the molecule-transfer graph, we first identify matched molecular pairs from ChEMBL. We then connect molecules appearing in these pairs into an undirected molecule-transfer graph. From this graph, we sample connected local subgraphs as training examples. Each sampled subgraph contains six molecules: five molecules are used as the input context, and the remaining molecule is

held out as the target. We construct 300k such subgraph-completion examples and train the generator with supervised teacher forcing by maximizing the conditional likelihood of the target molecule given the context.

Architecturally, we use T5Chem as the molecular sequence-to-sequence backbone. A two-layer GNN fusion module is inserted after the encoder to aggregate information across molecules in the input context before decoding the target molecule. Unless otherwise stated, the context length is five molecules, the GNN dropout rate is 0.1, and the GNN hidden dimension is set to match the hidden dimension of the Transformer backbone. The model is optimized with AdamW using a learning rate of 10^{-4}, weight decay of 10^{-5}, batch size 16, and maximum input and target lengths of 96 and 64 tokens, respectively. We train for three epochs, selecting the checkpoint by lowest validation loss.

We consider three fine-tuning regimes to assess the contribution of different model components: updating only the GNN module, updating the GNN and decoder, and updating the full model. In all cases, optimization is performed only over trainable parameters. Validation losses are averaged across workers, with a separate best checkpoint for each fine-tuning regime.

B.5 Molecule Graph Expansion

This subsection details the acceptance and edge-construction rules for the graph update in Eq. 4. Generated candidates are canonicalized, checked for validity, and deduplicated against the current graph and the generated batch. Let $\mathcal{X}_t^{\text{valid}}$ denote the resulting valid, unique candidates not already present in $\mathcal{V}_t$. These candidates are evaluated by the target property oracle regardless of MMP acceptance. The current graph G_t contains the initial molecules and molecules accepted in earlier batches. We accept a candidate if RDKit-based MMP matching identifies an edge to any molecule in $\mathcal{V}_t$, then insert the accepted molecules and their verified MMP edges using Eq. 4. The accepted set and added edges are

$$\begin{aligned} \widetilde{\mathcal{X}}_t &= \left\{ x \in \mathcal{X}_t^{\text{valid}} \;\middle|\; \exists u \in \mathcal{V}_t,\ x \sim u \right\}, \\ \mathcal{E}_t^{+} &= \left\{ \{x, u\} \;\middle|\; x \in \widetilde{\mathcal{X}}_t,\ u \in \mathcal{V}_t \cup \widetilde{\mathcal{X}}_t,\ x \neq u,\ x \sim u \right\}. \end{aligned} \tag{21}$$

The acceptance decisions are completed against G_t before the graph is updated. Here, $\{x, u\}$ denotes an undirected edge between molecules x and u, with $x \sim u$ indicating an MMP relation. In this set, u may be either an existing molecule or another accepted new molecule. Each accepted molecule therefore extends an existing MMP path from the initial set. The expanded graph and new property observations guide subsequent anchor selection, closing the loop between world model-guided generation and graph expansion.

B.6 Optimization Protocol

At each iteration, MolWorld uses the molecule world model and property scorer to select anchor contexts from the current molecule-transfer graph, generates candidate molecules from each selected context, evaluates valid, deduplicated candidates with the target oracle, and expands the graph using RDKit-based MMP validation. Repeated candidates do not consume additional oracle budget. Evaluations of new candidates count toward the budget regardless of whether they satisfy the MMP reachability criterion. The updated molecule-transfer graph is then used for the next iteration. For property optimization, the oracle budget is 10000 for ZINC250 and 1000 for PMV21. For docking-based optimization, the oracle budget is 1000. All experiments are repeated with 3 random seeds and conducted on a server equipped with an AMD EPYC 7413 CPU, 125 GiB system memory, and an NVIDIA RTX A6000 GPU with 48 GB of memory.

C Additional Experimental Details

C.1 Evaluation Scope

Oracle budgets. The reported budgets count new oracle evaluations during optimization. For property optimization, the scorer is pretrained using target-property labels on ChEMBL-derived expansion outcomes and reused across searches for the same property. We report oracle query efficiency during optimization separately from this reusable training investment. The docking scorer uses only docking evaluations collected within the reported 1,000-call budget.

Actionability. Actionability here refers to structural-series actionability: verified MMP paths connect generated candidates to the initial series and provide structural references for subsequent optimization.

C.2 Baseline Details

We provide brief descriptions of the baseline methods used in our experiments. All baselines use the same oracle budgets during optimization and task settings as MolWorld.

We compare methods under fixed budgets of new oracle evaluations during optimization, with each method using its pretrained models and previously prepared resources. Several baselines incorporate prior information through pretrained molecular generators, learned latent representations, or property-informed fragment vocabularies. In particular, GenMol uses target-property scores computed before optimization to construct its initial fragment vocabulary. The reported budgets measure additional oracle evaluations during optimization; the cost of pretraining and offline resource construction is considered separately.

GraphGA. GraphGA (Jensen, 2019) is a graph-based genetic algorithm for molecular optimization. It maintains a population of molecules and iteratively generates new candidates through molecular mutation and crossover operations. The population is updated according to oracle scores, allowing the search process to favor molecules with improved properties over successive generations.

MOLLEO. MOLLEO (Wang et al., 2025) is a recent molecule optimization method that combines evolutionary search with large language model-based molecular editing. We use its BioT5-based variant, which incorporates a model pretrained on molecular, biological, and textual data and subsequently fine-tuned for text-guided molecule generation. It uses LLM-guided mutation and crossover operations to propose chemically meaningful candidates, and then selects molecules based on oracle feedback. This supports search under limited oracle budgets.

REINVENT. REINVENT (Olivecrona et al., 2017) is a reinforcement-learning-based molecular generation method built on a SMILES generative model. Starting from a pretrained prior model, it updates the generator using oracle-based rewards so that generation favors desired properties. The prior learns molecular sequence patterns before the target-specific optimization begins.

Augmented Memory. Augmented Memory (Guo & Schwaller, 2023) is a reinforcement-learning-based molecular optimization method that improves sample efficiency by reusing high-scoring molecules observed during optimization. It starts from a pretrained molecular prior and reuses stored oracle scores across augmented SMILES representations to improve learning without additional evaluations of the same molecules. The memory mechanism helps preserve promising molecular patterns and stabilizes oracle-guided generator updates.

GP-BO. GP-BO (Tripp et al., 2021) applies Bayesian optimization to molecular optimization. It trains a Gaussian process surrogate model to approximate the oracle function over molecular representations and uses the surrogate to guide the selection of promising candidate molecules.

NFBO. NFBO (Lee et al., 2025b) performs Bayesian optimization in the latent space of SeqFlow, an autoregressive normalizing flow for molecular sequences. The released SeqFlow checkpoint is pretrained on 1.27 million molecules from GuacaMol. Its encoding and decoding mappings are designed to eliminate the reconstruction gap between a molecule and its latent representation. Candidate sampling adjusts exploration at each token according to its importance, while oracle observations guide the latent-space search.

GenMol. GenMol (Lee et al., 2025a) is a discrete diffusion model operating on Sequential Attachment-based Fragment Embedding (SAFE) sequences. Its generator is pretrained on the SAFE dataset containing molecules from ZINC and UniChem. It generates molecules through parallel denoising and supports molecular optimization through fragment remasking: selected fragments are regenerated while the remaining molecular context is retained. This allows local structural modifications to be explored using oracle feedback. Before optimization, its initial fragment vocabulary is ranked using the average target-property scores of molecules containing each fragment. These scores are computed offline and are separate from the oracle calls counted during optimization.

RetMol. RetMol (Wang et al., 2022) guides a pretrained molecular generator with retrieved exemplar molecules that partially or fully satisfy the desired property criteria. Its retrieval fusion module is trained using a self-supervised molecular similarity objective, while the pretrained encoder and decoder remain frozen. It fuses representations of the input molecule and retrieved exemplars to condition generation. Iterative refinement updates molecules and the retrieval database, using new discoveries to guide proposals without task-specific fine-tuning.

Table 5: Task 2 docking settings using TDC boxes (coordinates and sizes in Å).

Target	PDB ID	Box center (x, y, z)	Box size (x, y, z)	Exhaustiveness	Poses
DRD3	3PBL	$(9,\ 22.5,\ 26)$	$(15,\ 15,\ 15)$	8	9
EGFR	2RGP	$(16.2921,\ 34.8708,\ 92.0353)$	$(15,\ 15,\ 15)$	8	9
A2AR	3EML	$(-9.0636,\ -7.1446,\ 55.8626)$	$(15,\ 15,\ 15)$	8	9

InVirtuoGen. InVirtuoGen (Kaech et al., 2026) is a discrete flow model for fragmented SMILES that refines sequences starting from a uniform token distribution. Optimization starts from a pretrained generative checkpoint. For property optimization, it combines genetic search with Proximal Property Optimization, a fine-tuning procedure adapted to discrete flows. Oracle feedback guides both candidate selection and updates to the generative model.

C.3 Docking Protocol

Task 2 evaluates generated molecules on DRD3, EGFR, and A2AR using TDC docking oracles through the PyScreener AutoDock Vina backend. The corresponding receptor structures are 3PBL, 2RGP, and 3EML. For each molecule, the oracle returns the best Vina binding affinity among the generated poses, with lower scores indicating stronger predicted binding.

PyScreener converts the prepared receptor PDB files to PDBQT format using the `prepare_receptor` utility. For ligand preparation, generated SMILES strings are parsed with RDKit, hydrogens are added, and 3D conformers are generated using distance-geometry embedding and minimized with the MMFF force field. The ligands are then processed with OpenBabel/Pybel, assigned Gasteiger charges, and written in PDBQT format.

The docking box centers and sizes follow the TDC metadata (Table 5). Vina uses an exhaustiveness of 8, up to 9 output poses, an energy range of 3.0 kcal/mol, and 4 CPU threads per docking call. The Vina random seed is not fixed.

Docking AUC is computed over the first 1,000 oracle calls using $\bar{s}_{\text{dock}} = \min(0, s_{\text{dock}})$. For all compared methods, molecule eligibility follows the batch-wise reachability criterion defined in the experimental setup. Evaluations remain on the oracle-call axis regardless of molecule eligibility. When batch records are unavailable, molecules are processed individually in the saved oracle-call order. The running top-k score sums the k lowest eligible scores and divides by $\min(k, t)$, with unfilled positions assigned zero. AUC@10 and AUC@100 average these running scores across calls, so lower values are better. Within the running top-k set, duplicate molecules occupy one position, using their best eligible observed score; failed calls retain their budget positions. Only scores available during the original run are included. Iso is the fraction of valid, scored unique generated molecules that are not accepted under the same reachability criterion.

C.4 Additional Property Optimization Results

This section provides detailed property optimization results with standard deviations. Tables 6 and 7 report the MMP-constrained results on ZINC250 and PMV21, respectively, using the batch-wise reachability criterion defined in the experimental setup. ZINC250 results are evaluated with 10,000 oracle calls, while PMV21 results are evaluated with 1,000 oracle calls. Results are reported as mean $\pm$ sample standard deviation across seeds. None denotes the absence of generated molecules satisfying the MMP connectivity constraint.

InVirtuoGen exhibits substantial variation on ZINC–JNK3, with AUC@10 values of 0.8513, 0.3890, and 0.8096 across three seeds. The lowest-AUC run eventually achieves a final top-10 mean score of 0.917 among reachable molecules, but its first reachable molecule scoring at least 0.8 appears only after 6,637 oracle evaluations, compared with 1,185 and 1,302 evaluations in the other runs. This delayed discovery substantially reduces its AUC despite strong final scores. In another seed, InVirtuoGen achieves a maximum reachable score of 0.98, exceeding the maximum of 0.96 observed across the three MolWorld runs. These results illustrate the distinction between finding an exceptionally high-scoring molecule and consistently discovering high-scoring, reachable molecules early within a fixed oracle budget.

Training-molecule overlap. We examine exact molecular overlap between the molecules used to construct property-scorer training labels and the initial and generated molecules in MolWorld. Molecules are matched using canonical isomeric SMILES. The initial PMV21 graph contains no overlapping molecules among 722 molecules, while the ZINC250 graph contains 341 among 109,500 (0.3114%). Generated-molecule overlap is computed over the union of unique molecules across three

Table 6: MMP-constrained property optimization on ZINC250. Results are reported as mean ± sample standard deviation. Iso denotes the fraction of generated molecules that do not connect to the initial graph under the reachability criterion.

Method	GSK3β			QED		
	AUC@10↑	AUC@100↑	Iso↓	AUC@10↑	AUC@100↑	Iso↓
GraphGA	0.3963 ± 0.3076	0.2589 ± 0.3407	0.8792 ± 0.1763	0.9304 ± 0.0046	0.8844 ± 0.0095	0.8092 ± 0.0208
REINVENT	0.6564 ± 0.2467	0.5338 ± 0.2734	0.6689 ± 0.1972	0.9392 ± 0.0007	0.9062 ± 0.0025	0.6274 ± 0.0483
Aug-Mem	0.6175 ± 0.2580	0.4400 ± 0.3527	0.9091 ± 0.0676	0.9366 ± 0.0011	0.8931 ± 0.0047	0.7300 ± 0.0421
GP-BO	0.8195 ± 0.1652	0.7198 ± 0.2186	0.6106 ± 0.3663	0.9231 ± 0.0056	0.8745 ± 0.0060	0.9330 ± 0.0096
MOLLEO	0.8904 ± 0.0425	0.8090 ± 0.0503	0.4642 ± 0.1389	0.9299 ± 0.0087	0.8751 ± 0.0335	0.7956 ± 0.0287
NFBO	0.8316 ± 0.0235	0.6026 ± 0.0187	0.9623 ± 0.0082	0.9384 ± 0.0014	0.8856 ± 0.0063	0.7273 ± 0.0320
GenMol	0.3360 ± 0.2550	0.1083 ± 0.1172	0.9901 ± 0.0094	0.9103 ± 0.0093	0.8266 ± 0.0057	0.8720 ± 0.0089
RetMol	0.5112 ± 0.4455	0.4687 ± 0.4099	0.4019 ± 0.1239	0.9457 ± 0.0015	0.9396 ± 0.0035	0.3427 ± 0.2195
InVirtuoGen	0.9012 ± 0.0310	0.7397 ± 0.0522	0.7102 ± 0.0737	0.9288 ± 0.0136	0.8646 ± 0.0452	0.8655 ± 0.0865
MolWorld	**0.9576 ± 0.0216**	**0.9044 ±0.0385**	**0.0460 ± 0.0043**	**0.9481 ± 0.0001**	**0.9470 ± 0.0005**	**0.0438 ± 0.0066**
Method	JNK3			DRD2		
	AUC@10↑	AUC@100↑	Iso↓	AUC@10↑	AUC@100↑	Iso↓
GraphGA	0.5056 ± 0.1334	0.4081 ± 0.1085	0.8233 ± 0.1054	0.6163 ± 0.3437	0.3150 ± 0.3110	0.8772 ± 0.0408
REINVENT	0.7574 ± 0.0303	0.7097 ± 0.0320	0.3953 ± 0.0572	0.8903 ± 0.0134	0.8381 ± 0.0113	0.5380 ± 0.1838
Aug-Mem	0.7201 ± 0.0944	0.6445 ± 0.0961	0.7495 ± 0.1489	0.8552 ± 0.0375	0.7426 ± 0.0441	0.7895 ± 0.0800
GP-BO	0.3509 ± 0.1783	0.3050 ± 0.1919	0.3230 ± 0.2317	0.9083 ± 0.0621	0.8336 ± 0.0679	0.3163 ± 0.2214
MOLLEO	0.7759 ± 0.0471	0.7106 ± 0.0519	0.6214 ± 0.0395	0.7474 ± 0.3493	0.3679 ± 0.4014	0.8543 ± 0.0849
NFBO	0.6459 ± 0.0257	0.5559 ± 0.0379	0.6462 ± 0.1412	0.9179 ± 0.0540	0.7823 ± 0.0266	0.7489 ± 0.0770
GenMol	0.0003 ± 0.0006	0.0000 ± 0.0001	1.0000 ± 0.0001	0.9860 ± 0.0027	0.9522 ± 0.0122	0.4267 ± 0.0266
RetMol	0.5422 ± 0.1701	0.4656 ± 0.1678	0.1996 ± 0.0321	0.9484 ± 0.0359	0.8536 ± 0.1695	0.4884 ± 0.2729
InVirtuoGen	0.6833 ± 0.2557	0.5412 ± 0.3364	0.7918 ± 0.1663	0.6805 ± 0.2432	0.1792 ± 0.1431	0.7820 ± 0.0600
MolWorld	**0.8225 ± 0.1293**	**0.7770 ± 0.1229**	**0.1105 ± 0.0951**	**0.9943 ± 0.0010**	**0.9819 ± 0.0014**	**0.0202 ± 0.0020**

Table 7: MMP-constrained property optimization on PMV21. Results are reported as mean ± sample standard deviation. Iso denotes the fraction of generated molecules that do not connect to the initial graph under the reachability criterion.

Method	GSK3β			QED		
	AUC@10↑	AUC@100↑	Iso↓	AUC@10↑	AUC@100↑	Iso↓
GraphGA	0.2751 ± 0.0217	0.1856 ± 0.0134	0.6057 ± 0.0558	0.7286 ± 0.0362	0.5408 ± 0.0238	0.8007 ± 0.0205
REINVENT	None	None	1.0000 ± 0.0000	None	None	1.0000 ± 0.0000
Aug-Mem	None	None	1.0000 ± 0.0000	None	None	1.0000 ± 0.0000
GP-BO	0.2426 ± 0.0046	0.1512 ± 0.0015	0.6597 ± 0.0040	0.6908 ± 0.0516	0.5429 ± 0.0230	0.6237 ± 0.0393
MOLLEO	0.3008 ± 0.0162	0.1936 ± 0.0134	0.6603 ± 0.0902	0.7880 ± 0.0194	0.5649 ± 0.0233	0.7903 ± 0.0202
NFBO	0.2733 ± 0.0484	0.1082 ± 0.0433	0.8463 ± 0.0764	0.6115 ± 0.1316	0.3218 ± 0.1050	0.8445 ± 0.0683
GenMol	0.3138 ± 0.0394	0.0538 ± 0.0095	0.9810 ± 0.0017	0.8010 ± 0.0482	0.3158 ± 0.1516	0.9477 ± 0.0294
RetMol	0.2037 ± 0.0291	0.0866 ± 0.0168	0.8917 ± 0.0080	0.6290 ± 0.0336	0.4175 ± 0.0152	0.9000 ± 0.0020
InVirtuoGen	0.2287 ± 0.0069	0.1284 ± 0.0031	0.9000 ± 0.0020	0.6113 ± 0.0041	0.4287 ± 0.0038	0.8973 ± 0.0012
MolWorld	**0.4225 ± 0.0159**	**0.3490 ± 0.0220**	**0.0493 ± 0.0060**	**0.8892 ± 0.0119**	**0.8194 ± 0.0098**	**0.0744 ± 0.0359**
Method	JNK3			DRD2		
	AUC@10↑	AUC@100↑	Iso↓	AUC@10↑	AUC@100↑	Iso↓
GraphGA	0.1490 ± 0.0148	0.0982 ± 0.0074	0.5527 ± 0.0389	0.7722 ± 0.0023	0.4403 ± 0.0225	0.6247 ± 0.0567
REINVENT	None	None	1.0000 ± 0.0000	None	None	1.0000 ± 0.0000
Aug-Mem	None	None	1.0000 ± 0.0000	None	None	1.0000 ± 0.0000
GP-BO	0.1364 ± 0.0159	0.0858 ± 0.0050	0.6420 ± 0.0272	0.7674 ± 0.0632	0.4145 ± 0.0642	0.5930 ± 0.1127
MOLLEO	0.1744 ± 0.0102	0.1114 ± 0.0048	0.5347 ± 0.0331	0.8019 ± 0.0173	0.4521 ± 0.0369	0.7047 ± 0.0400
NFBO	0.1662 ± 0.0543	0.0951 ± 0.0639	0.6734 ± 0.2556	0.6783 ± 0.1008	0.3092 ± 0.2019	0.7897 ± 0.0709
GenMol	0.2315 ± 0.0698	0.0427 ± 0.0150	0.9717 ± 0.0040	0.8659 ± 0.0313	0.3765 ± 0.0735	0.9123 ± 0.0244
RetMol	0.1122 ± 0.0083	0.0529 ± 0.0047	0.8930 ± 0.0030	0.7774 ± 0.1405	0.5135 ± 0.2820	0.6157 ± 0.2333
InVirtuoGen	0.1321 ± 0.0025	0.0741 ± 0.0014	0.9000 ± 0.0010	0.4878 ± 0.0693	0.2112 ± 0.0356	0.7478 ± 0.0072
MolWorld	**0.2710 ± 0.0286**	**0.2175 ± 0.0224**	**0.0437 ± 0.0090**	**0.9018 ± 0.0063**	**0.8205 ± 0.0047**	**0.0990 ± 0.0249**

seeds. Final Top-10 and Top-100 overlap counts are summed across the three seeds, considering only generated molecules that satisfy the MMP reachability criterion.

To evaluate the contribution of overlapping molecules to optimization scores, we set their score contributions to zero and recompute AUC while preserving oracle-call positions, budgets, and reachability labels. We report the mean difference across seeds, defined as AUC after exclusion minus

AUC before exclusion (Table 8). Final Top-10 and Top-100 overlap describe the molecules retained at the end of optimization, whereas AUC changes also reflect overlapping molecules that entered the top-ranked sets earlier in the search.

Table 8: Training-molecule overlap and AUC sensitivity of MOLWORLD. Input and generated overlap are percentages. Final Top-10 and Top-100 overlap are molecule counts summed across three seeds. ΔAUC reports the mean change after excluding overlapping score contributions.

Dataset	Property	Input overlap (%)	Generated overlap (%)	Final Top-10 overlap	Final Top-100 overlap	ΔAUC@10	ΔAUC@100
ZINC250	GSK3β	0.3114	0.0156	0	0	−0.000016	−0.000075
	QED		3.5317	0	1	0.000000	−0.000656
	JNK3		0.3385	0	0	−0.000469	−0.000344
	DRD2		0.0489	0	0	0.000000	−0.000041
PMV21	GSK3β	0.0000	0.0000	0	0	0.000000	0.000000
	QED		0.0000	0	0	0.000000	0.000000
	JNK3		0.0000	0	0	0.000000	0.000000
	DRD2		0.0000	0	0	0.000000	0.000000

Across the analyzed trajectories, none of the final Top-10 molecules overlaps with the property-scorer training molecules. The final Top-100 sets contain only one overlapping molecule, found in the ZINC250 QED task. Excluding overlapping score contributions reduces mean AUC@10 by at most 0.000469 and mean AUC@100 by at most 0.000656, indicating that these molecules contribute little directly to the AUC scores in this analysis.

Training-scaffold overlap. We further compare generated molecules with the property-scorer label training set using Bemis–Murcko scaffolds computed with RDKit, ignoring stereochemistry. A generated molecule counts as overlapping only if its scaffold occurs in the training set but is absent from the initial input graph. Scaffolds already present in the input graph are excluded from the overlap counts, while the denominators remain unchanged. We consider only MMP-reachable generated molecules. The output percentage is computed over the union of unique molecules across three seeds; final Top-10 and Top-100 counts sum the ranked entries across seeds.

Table 9: Scaffold overlap with the property-scorer label training set after excluding scaffolds present in the initial graph. Generated overlap is the percentage of output molecules satisfying this condition. Top-10 and Top-100 entries report overlapping entries out of the total across three seeds.

Dataset	Property	Generated overlap (%)	Top-10 overlap	Top-100 overlap
ZINC250	GSK3β	1.60	0/30	0/300
	QED	9.66	0/30	15/300
	JNK3	49.96	20/30	198/300
	DRD2	39.23	18/30	158/300
PMV21	GSK3β	0.00	0/30	0/300
	QED	0.05	0/30	0/300
	JNK3	0.00	0/30	0/300
	DRD2	0.00	0/30	0/300

Table 9 shows that overlap varies across tasks, with higher rates on ZINC250 JNK3 and DRD2. Our objective is to improve molecular properties while preserving MMP reachability from the initial set; scaffold novelty is not an optimization requirement. Sharing a scaffold with a training molecule does not imply reproducing that molecule, so scaffold overlap complements the exact-molecule analysis above. On PMV21, the initial graph shares no scaffolds with the property-scorer label training set, and generated-molecule overlap is zero for GSK3β, JNK3, and DRD2, including their final Top-10 and Top-100 sets. Nevertheless, MOLWORLD achieves the highest AUC@10 and AUC@100 among the compared methods on each of these three tasks (Table 1). Strong optimization performance is therefore also observed on tasks with no measured scaffold overlap with this training set.

C.5 OPTIMIZATION DYNAMICS

We analyze the optimization dynamics of different methods by plotting the top-10 running average oracle score as a function of oracle calls. Figure 7 shows how efficiently each method discovers high-quality molecules under the same oracle budget.

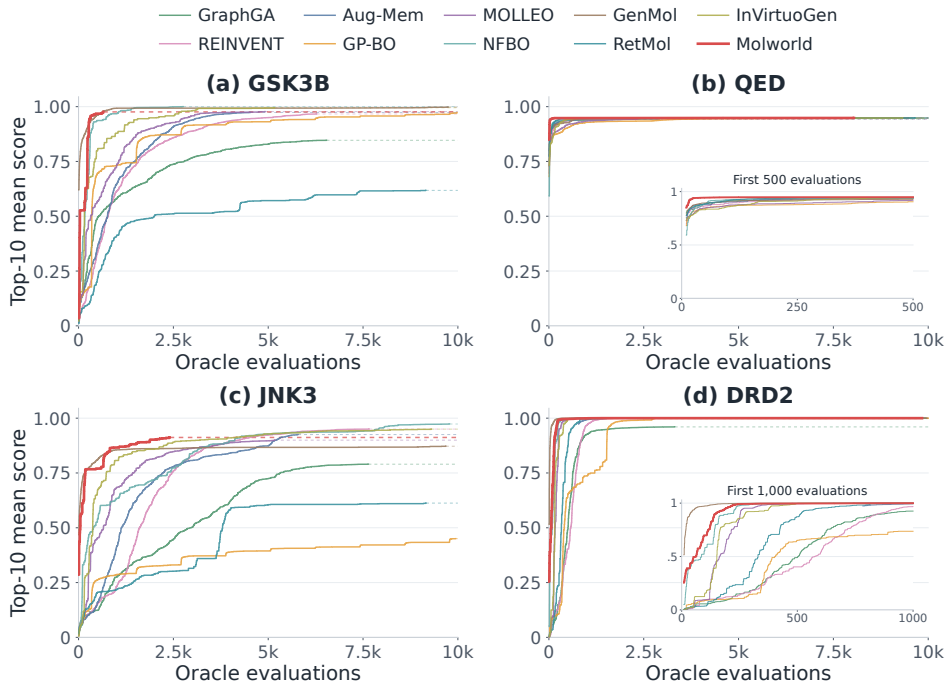


Figure 7: Optimization dynamics on ZINC250. Top-10 running average oracle score over all valid scored molecules as a function of oracle calls across four tasks.

MOLWORLD shows faster improvement on more challenging tasks such as GSK3β and JNK3, reaching strong top-10 performance earlier than most baselines. On relatively easier tasks such as QED and DRD2, most methods improve quickly, while MOLWORLD remains competitive throughout the search process. These results suggest that MOLWORLD efficiently guides search toward high-quality regions under fixed oracle budgets.

C.6 ADDITIONAL GENERATED MOLECULE DISTRIBUTION RESULTS

We provide additional generated molecule distribution visualizations on QED, JNK3, and DRD2 in Figures 8, 9, and 10. For each task, the left panel shows the UMAP projection of initial and generated molecules, and the right panel shows oracle-score progression over generation order with a 10-step moving-average curve. Consistent with the GSK3β result in the main text, MOLWORLD expands from the initial molecule-transfer graph toward higher-scoring regions.

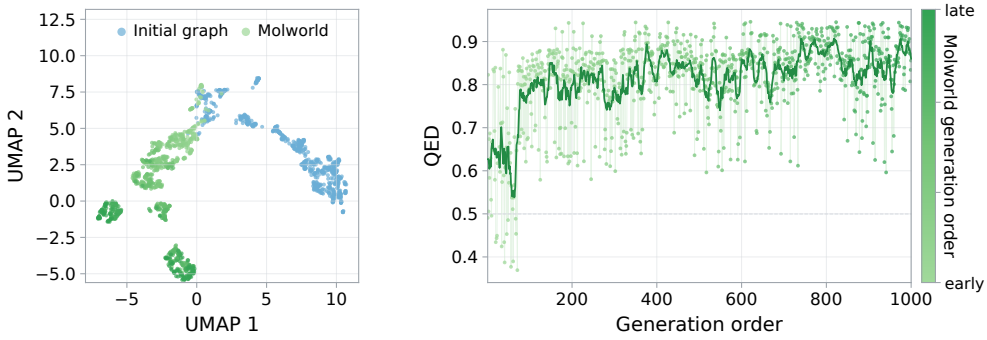


Figure 8: Generated molecule distribution and oracle-score progression of MOLWORLD on QED.

C.7 ANCHOR CONTEXT EVOLUTION

We further analyze how the selected anchor contexts evolve during optimization, as shown in Figure 11. Each anchor context contains five molecules that define the local generation region. Blue bars denote initial-graph molecules; light, medium, and dark green bars denote molecules generated in early, middle, and late stages, respectively.

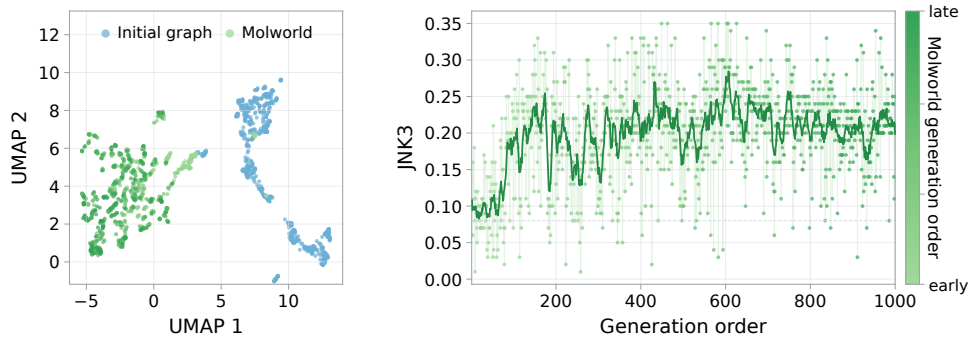


Figure 9: Generated molecule distribution and oracle-score progression of MOLWORLD on JNK3.

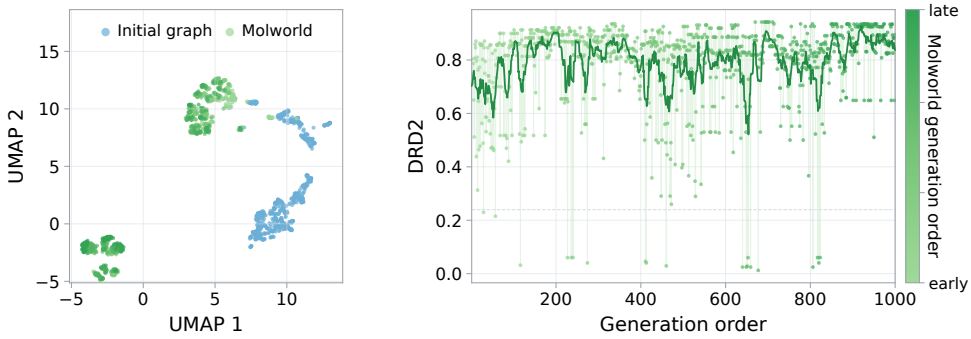


Figure 10: Generated molecule distribution and oracle-score progression of MOLWORLD on DRD2.

Across all tasks, anchor contexts quickly shift from initial molecules to newly generated molecules after the early iterations. The number of initial molecules used in anchor contexts increases only at the beginning and then stabilizes, while generated molecules become the dominant source of later anchor contexts. This confirms the self-expanding behavior of MOLWORLD: newly generated molecules are incorporated into the evolving molecule-transfer graph and reused to guide subsequent generation. As a result, MOLWORLD can progressively build upon its own generated molecules rather than repeatedly sampling from the initial graph.

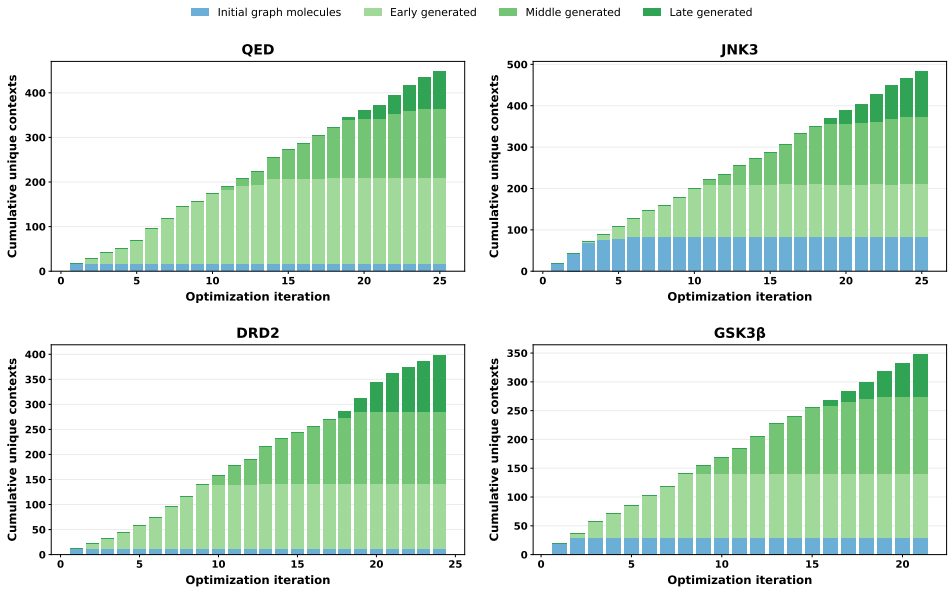


Figure 11: Evolution of anchor context sources across iterations, shown as cumulative counts. Blue denotes initial-graph molecules; light, medium, and dark green denote molecules from early, middle, and late stages.