Guiding Evolutionary Molecular Design: Adding Reinforcement Learning for Mutation Selection

Gaëlle Milon-Harnois, Chaïmaâ Touhami, Nicolas Gutowski, Benoit da Mota, Thomas Cauchy

Guiding Evolutionary Molecular Design: Adding Reinforcement Learning for Mutation Selection

Gaëlle MILON-HARNOIS
Univ Angers, LERIA, SFR MathSTIC
F-49000 Angers, France
gaelle.milon-harnois@univ-angers.fr

Chaïmaâ TOUHAMI
Univ Angers, LERIA, SFR MathSTIC
F-49000 Angers, France
ctouhami@etud.univ-angers.fr

Nicolas GUTOWSKI
Univ Angers, LERIA, SFR MathSTIC
F-49000 Angers, France
nicolas.gutowski@univ-angers.fr

Benoit DA MOTA
Univ Angers, LERIA, SFR MathSTIC
F-49000 Angers, France
benoit.damota@univ-angers.fr

Thomas CAUCHY
Univ Angers, CNRS, MOLTECH-ANJOU, SFR MATRIX
F-49000 Angers, France
thomas.cauchy@univ-angers.fr

***Abstract*—The efficient exploration of chemical space remains a central challenge, as many generative models still produce unstable or non-synthesizable compounds. To address these limitations, we present *EvoMol-RL*, a significant extension of the *EvoMol* evolutionary algorithm that integrates reinforcement learning to guide molecular mutations based on local structural context. By leveraging Extended Connectivity Fingerprints (ECFPs), *EvoMol-RL* learns context-aware mutation policies that prioritize chemically plausible transformations. This approach significantly improves the generation of valid and realistic molecules, reducing the frequency of structural artifacts and enhancing optimization performance. The results demonstrate that *EvoMol-RL* consistently outperforms its baseline in molecular pre-filtering realism. These results emphasize the effectiveness of combining reinforcement learning with molecular fingerprints to generate chemically relevant molecular structures.**

***Index Terms*—Evolutionary Algorithm, Reinforcement Learning, Sleeping Bandit, Cheminformatics, Molecule Generation**

I. Introduction

The chemical space of possible molecules is extraordinarily vast, rendering exhaustive exploration computationally intractable. Theoretical estimates suggest that the number of drug-like molecules may exceed 10^{60} [1], while even limiting the enumeration to small organic molecules with coherent structures can already yield over 160 billion structures [2]. Yet, databases of experimentally known or commercially available molecules such as ZINC [3] or ChEMBL [4] contain fewer than a billion compounds. This indicates an immense potential for discovery—spanning applications in drug development, materials science, and fundamental chemistry. Exploring this space to identify novel, synthetically feasible, and functionally relevant molecules is one of the central challenges in cheminformatics.

This combinatorial explosion poses fundamental obstacles for rational molecular discovery, especially when candidate evaluation is computationally expensive or experimentally costly. Accordingly, the past decade has seen the development of a wide variety of generative models aimed at the design of novel molecules optimized for target properties. These include: (1) **Rule-based methods**, notably template-based retrosynthetic planning [5], expert-guided reactivity prediction [6], and extraction of graph-based reaction rules from large-scale reaction databases with atom-atom mapping [7]; (2) **Fragment-based approaches**, which assemble molecules by recombining known substructures, with junction-tree Variational Autoencoders as seminal examples [8], and recent comprehensive reviews highlighting their strengths in ensuring chemical validity and interpretability [9]; (3) **Evolutionary algorithms (EAs)**, which use iterative mutation and selection processes to explore chemical space, with growing interest in combining EA frameworks with synthesizability constraints [10]–[12]; (4) **Deep learning models** based on Textual representations such as the Simplified Molecular Input Line Entry System (SMILES) or graph representations, including autoencoders [13], graph neural networks, and diffusion-based generative models, as surveyed comprehensively in [14]; and (5) **Reinforcement learning (RL)** where agents iteratively construct or modify molecules based on sequential decision-making, has emerged as a powerful framework for molecular optimization. In chemistry, RL agents learn to construct or modify molecules by exploring chemical space with the aim of maximizing long-term rewards tied to molecular properties. Foundational work by Olivecrona *et al.* [15] introduced policy-gradient-based SMILES generation, while Zhou *et al.* [16] extended this to graph-based models using deep Q-networks. Recent reviews by Gow *et al.* [17] and Sridharan *et al.* [18] provide comprehensive overviews of RL's theoretical foundations and practical implementations in chemistry, covering applications from molecule generation and retrosynthesis to geometry optimization. These works highlight the increasing relevance of RL as a general-purpose optimization tool in computational molecular science.

Despite significant advances, a persistent limitation across generative models is the production of unrealistic, unstable, or non-synthesizable molecules. This issue arises in various forms: random sampling in *de novo* models often lacks chemical validity [19]; latent space methods may suffer from

overfitting, mode collapse, or poor inverse mapping [20]; fragment-based strategies can recombine motifs in sterically or electronically implausible ways without contextual coherence [9]; and rule-based simulations may generate products outside synthetic feasibility due to incomplete encoding of reaction conditions or selectivity constraints [21]. These limitations typically stem from a lack of chemically informed constraints, insufficient local context during generation, or excessively permissive mutation and transformation rules [14]. In this landscape, *EvoMol* offers a robust alternative by combining evolutionary search with chemically motivated filters, particularly those enforcing realistic atom environments and ring systems [22]. However, like most evolutionary algorithms, it applies stochastic mutations that are agnostic to the local molecular context [12]. As a result, many generated structures remain chemically irrelevant and must still be evaluated and filtered out. This inefficiency becomes particularly problematic in scenarios where molecular evaluation is computationally expensive (e.g., docking, quantum chemistry) or experimentally limited [23].

To overcome the limitations of context-agnostic mutations in evolutionary algorithms, we introduce *EvoMol-RL*, an extension of the *EvoMol* framework that integrates *reinforcement learning* (RL) to guide the molecular mutation process. By incorporating local molecular environments encoded via Extended Connectivity Fingerprints (ECFPs) [24], *EvoMol-RL* learns context-aware mutation policies that favor chemically meaningful and synthetically plausible transformations. This approach combines the stochastic exploration strength of evolutionary algorithms with the policy learning capabilities of RL, thereby improving both the efficiency and realism of chemical space navigation. The main contributions of this work are as follows: (1) we enhance *EvoMol* with a reinforcement learning agent that selects mutations conditioned on the molecular context (*EvoMol-RL*); (2) we introduce a dynamic and adaptive action space, structured around ECFP-derived features, which restricts chemically invalid or unproductive mutations; (3) we show that *EvoMol-RL* significantly outperforms the original *EvoMol* in terms of chemical validity, synthesizability, and property-based optimization performance.

The paper is organized as follows. Section II reviews relevant background on molecular generation and policy learning. Section III presents the problem formulation and describes the *EvoMol-RL* architecture in detail. Section IV reports experimental results and comparative benchmarks. Section V concludes with key findings and outlines directions for future research.

II. Background

A. Molecular Basics

A molecule is a discrete chemical entity composed of atoms held together by covalent bonds. Atoms are characterized by their atomic number or symbol and valence—defined as the number of covalent bonds they can form—while bonds may vary in order (single, double, triple). Molecular design efforts in organic chemistry typically focus on a subset of "heavy" atoms, including carbon (C), nitrogen (N), oxygen (O), fluorine (F), sulfur (S), phosphorus (P), chlorine (Cl), and bromine (Br), in addition to hydrogen (H).

B. Molecular Representation

A molecule can be represented in various formats designed to be compact, reproducible, unambiguous, flexible, and suitable for computational analysis. Among these, two are particularly popular.

a) Molecular graphs: they form the backbone of many cheminformatics approaches. Atoms, except H, are represented as vertices with atomic symbol, and bonds as edges labeled by bond types [25]. Hydrogen atoms are usually implicit and inferred from standard valence rules. Mathematically, a molecular graph can be defined as in equation (1).

$$G = (V, E, f_{\alpha\tau}, f_\beta, f_\gamma) \tag{1}$$

where V is a set of vertices representing the atoms of the molecule; $E \subseteq V \times V$ is a set of edges representing the chemical bonds between atoms; $f_{\alpha\tau} : V \to \{C, N, O, F, P, S, Cl, Br\}$ is the vertex labeling function, associating each vertex with an atomic symbol; $f_\beta : E \to \{1, 2, 3\}$ is the edge labeling function, associating each edge with a bond type; $f_\gamma : V \to \mathbb{Z}$ is the formal charge labeling function, associating a formal charge to each vertex.

While this discrete representation is well-suited to computation, challenges remain: exact graph matching (isomorphism) is a NP-complete decision problem, and comparing graphs via edit distance is a NP-hard optimization problem. Nevertheless, chemists have developed rules (e.g., canonicalization procedures) to provide consistent and unique graph traversals in most common cases.

Classical molecular drawings have some additional particularities: vertices without letters correspond to carbon atoms, and hydrogens bonded to carbons are implicit, while other hydrogens are explicit. Fig. 1 presents on the left, an example of the graph representation for the acetylsalicylic acid molecule.

b) Textual Representations: Textual representations such as the SMILES provide a linear encoding of molecular graphs with canonical variants, allowing string-based comparisons [26]. Among the drawbacks of this representation, SMILES discrete syntax does not always align with chemical similarity, and edit distances such as Levenshtein may fail to capture structural features. The top left of fig. 1 shows an example of the SMILES representation for the acetylsalicylic acid molecule.

C. Molecular Context

The graph-based approach is the foundation for defining structural descriptors like Extended Connectivity FingerPrints (ECFP) [24]. ECFPs are topological, circular fingerprints that iteratively encode substructures around each atom in a molecule, capturing local environments within a specified diameter (see Fig. 1).

At diameter $d = 0$ (noted ECFP_0), only intrinsic atomic features—such as atomic number, formal charge, number of

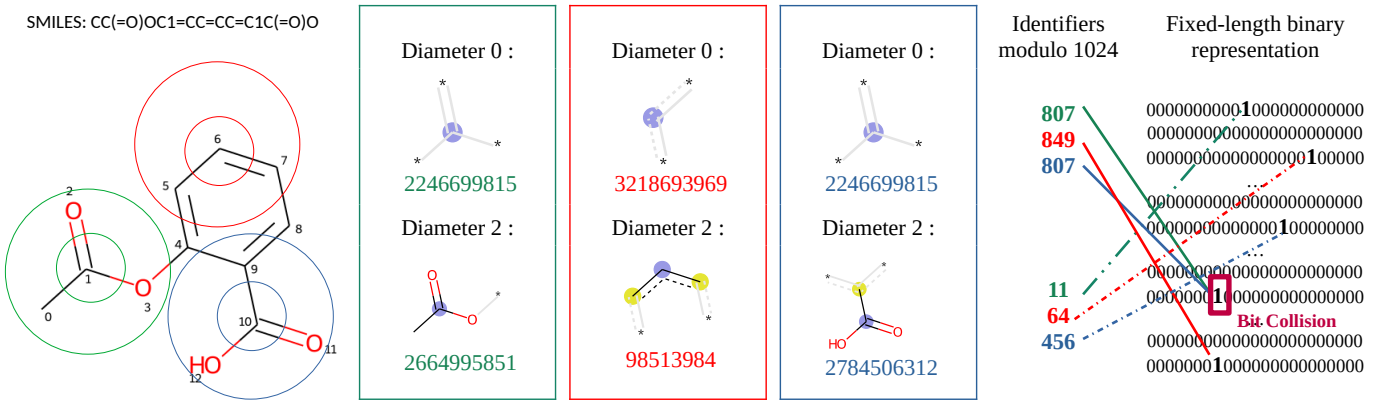

Fig. 1. ECFP generation process on Acetylsalicylic acid molecule and generation of the fixed-length fingerprint. *Blue circles correspond to the central atoms.*

bonded heavy atoms, and number of attached hydrogens—are considered. These features are hashed into unique integer identifiers. At $d = 2$ ($ECFP_2$), the process incorporates first-neighbor environments; for each atom, the hashed identifiers of its immediate neighbors are concatenated with its own, then re-hashed. This iterative procedure continues until a target diameter is reached (typically $d = 4$; $ECFP_4$). The resulting identifiers describe unique atom-centered environments across the molecule. These are collected and folded (e.g., modulo 1024 or 2048) into a fixed-length binary vector that encodes the presence (1) or absence (0) of specific environments. In our work, ECFPs are used to define the molecular context on which mutation decisions are conditioned.

D. Unrealistic environments: Silly Walks

Silly Walks (SW) is a computational metric that quantifies structural implausibility (or molecular abnormality) on a scale from 0, the least *silly* molecule, to 1, the most *silly* one. Following the implementation available in Patrick Walters' GitHub repository [27], the metric evaluates the proportion of *silly* bits in a molecule, i.e. the $ECFP_d$ (with $d \in \{0, 2, 4\}$) that never appear in a reference dataset, usually, the ChEMBL [28] or Zinc [3] databases.

E. The EvoMol Algorithm

In the field of de novo generation of molecules, *EvoMol* is an evolutionary algorithm that optimizes an arbitrary objective function (OF) by performing atomic level mutation on a set of molecules [11]. At each step, the algorithm selects up to ten best molecules based on this OF. Each selected molecule is then processed individually: random actions are applied to explore its neighborhood to find a better solution. For each molecule, a maximum of 50 mutations are attempted to find an improver.

In this study, among all *EvoMol*'s actions, we will only consider the most elementary ones, which alone allow us to cover the entire molecular space. Three primary atomic level actions are available: adding an atom (*AddA*), removing an atom (*RmA*) and changing bonds between 2 atoms (*ChB*), which includes creation and deletion of a bond.

During each iteration, *EvoMol* systematically evaluates the validity of every available action ($a \in \mathcal{A}$) for each vertex in the molecular graph, where $\mathcal{A}$ is the set of potential actions defined by the user within $\{AddA, RmA, ChB\}$. Therefore, all actions are not valid for each vertex as illustrated by Fig. 2 which presents all the *AddA* and *RmA* valid mutations on acetylsalicylic acid molecule. For the sake of space and clarity, valid *ChB* mutations are not reported. The valid mutations list ($Valid_{(a,idx_p)}$) contains all combinations of an action $a \in \mathcal{A}$ paired with an integer idx_p that denote the positional index of each valid a within the molecular structure, optionally crossed with additional parameters such as atom to add for *AddA* or type of new bond for *ChB*. Selection of a mutation (a, idx_p) is then performed through random sampling from $Valid_{(a,idx_p)}$.

Following the execution of the selected mutation on the molecule to improve, filters are implemented to ensure that the resulting molecule does not contain any *silly* bits (see Sec. II-D) and represents a genuinely novel compound that is not present within the existing molecular population. When a generated molecule successfully passes the filters, the OF is evaluated and compared to that of the parent molecule. Only new molecules showing an improved OF relative to their initial counterpart are inserted into the pool of candidates. This iterative process continues until a stopping criterion is met, usually a predetermined number of steps. This depiction of *EvoMol* is simplified in order to focus on the elements necessary in our study, more details are available in the article describing the method [11].

F. Reinforcement Learning and Sleeping Bandit Problem

Since *EvoMol-RL* relies on both a probability matching like method and the adversarial sleeping bandit problem, this section reminds all the related key concepts that underlie our approach.

a) Multi-armed bandits: The k-armed bandit [29] is a reinforcement learning [30] instance with a single state and a set of k independent actions ("arms") $i \in k$. Each

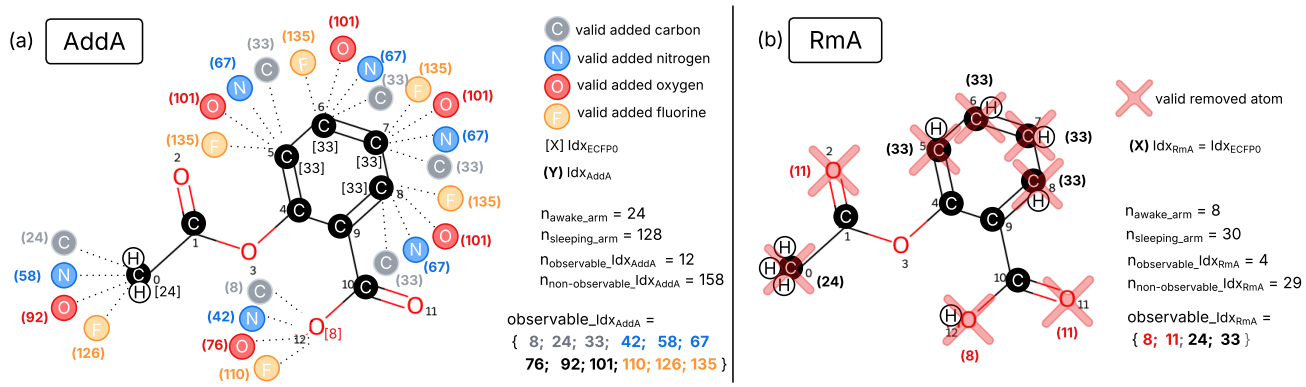

Fig. 2. Valid mutations starting from the acetylsalicylic acid molecule and corresponding Idx_{ECFP_0} and Idx_a. (a) *AddA* valid mutations with (C, N, O, F) candidate atom set. (b) *RmA* valid mutations.

arm is associated with a fixed reward distribution of mean $\mu_i \in [0,1]$. The vector of expected rewards is denoted by $D_r = (\mu_1, \ldots, \mu_k) \in [0,1]^k$. The problem is sequential: let $T \in \mathbb{N}^*$ be the time horizon, at each iteration $t \in 1, \ldots, T$, the learner chooses an arm $i_t \in K$ and receives a loss $\ell_{t,i_t} \in [0,1]$ (or reward $r_{t,i_t} = 1 - \ell_{t,i_t}$) (i.e. a reward $r_{i_t} \in \{0,1\}$ is drawn from a Bernoulli distribution with parameter μ_i). The goal is to maximize the cumulative reward (or minimize the regret) by learning to favor arms with higher expected returns.

Its performance is thus measured by the (pseudo-)regret defined in equation (2).

$$\mathrm{Reg}_T = \sum_{t=1}^{T} \ell_{t,i_t} - \min_{i \in [K]} \sum_{t=1}^{T} \ell_{t,i}. \tag{2}$$

b) Adversarial sleeping bandits: In the adversarial sleeping bandit (ASB) [31] and [32], losses $\{\ell_{t,i}\}_{i=1}^{k}$ may be chosen by an adversary and only a subset $K_t \subseteq K$ of "available" arms is offered at each round. The learner must pick $i_t \in K_t$. As defined in equation (3), regret is computed against the best fixed arm that was "awake" whenever it was needed:

$$\mathrm{Reg}_T^{\mathrm{sleep}} = \max_{i \in [K]} \sum_{t=1}^{T} \big(\mathbb{I}[i \in K_t] \cdot \ell_{t,i_t} - \ell_{t,i} \big). \tag{3}$$

Note that classical algorithms like EXP3-IX guarantee $\widetilde{O}(\sqrt{T})$ regret [33].

c) Connection with EvoMol-RL: In *EvoMol-RL*, each chemical mutation is modeled as an arm. Because the set of valid mutations depends on the current molecular graph, arms naturally sleep. Moreover, the reward acceptance by the SW filter is non-stochastic and potentially adversarial, motivating the ASB formulation. Our empirical probability matching heuristic strategy (See Sec. III-A) is a roulette wheel implementation adapted to the huge, sparse contextual state space induced by ECFP contexts (see Sec. III-A1). This formal link justifies the exploration-exploitation machinery introduced later.

G. Motivation

Since *EvoMol* is context-blind, it performs mutations randomly on the molecular graph. Without feedback on which mutations are efficient and which are ineffective, the agent lacks a reference point to decide which arm to select. We hypothesize that mutation selection can be improved through a heuristic based on RL that leverages molecular context using ECFP for learning. Since not all mutations and not all ECFP$_d$ indices are available at each step, and rewards (consisting in success counts) are computed iteratively, the adversarial sleeping bandit framework appears well-suited to adress this problem. Learning is guided by a reward function based on the molecule's ability to pass the *SW* filters.

III. Problem Setting

In this section, we state our problem and describe our method: *EvoMol-RL*, *EvoMol* enhanced with Reinforcement Learning.

A. Problem setup

At each iteration, the algorithm operates under a probability matching adversarial sleeping bandit-inspired heuristic approach, where the set of valid mutations (arms) $\mathcal{M}_t$ at time step $t \in [1,T]$ varies dynamically depending on the molecular graph structure.

Namely, let $T \in \mathbb{N}^*$ be the time horizon. Let $\mathcal{M} = \{m_1, ..., m_{k_T}\} \subseteq \Omega$ be a given set of k_T available independent valid mutations encountered during an entire *EvoMol* molecular generation process. Ω represents the whole universe of possible mutations that can be encountered during any molecular generation process. At each time step $t \in [1,T]$, a subset of k_t distinct valid mutations $\mathcal{M}_t = \{m_1^{(t)}, ..., m_{k_t}^{(t)}\} \subseteq \mathcal{M} \subseteq \Omega$ is available, which depends on the current state of the molecular graph G_t, thus resulting in non-stationary and context-dependent dynamics. Each valid mutation $m_j \in \mathcal{M}_t$ ($j \in \{1, ..., k_t\}$) is characterized by the action a applied at positional index idx_p on G_t, leading to a local context c derived from an ECFP-like molecular encoding. Let $p_{m_j}^{(t)}$

denote the empirical success probability for mutation m_j at time t, computed as the success-to-trial ratio for action a within the local context c. A stochastic policy $\pi_t(m) \propto p_{m_j}^{(t)}$ is then employed to select an action $m_t \in \mathcal{M}_t$, applying a dynamic proportional selection mechanism similar to a roulette wheel sampling strategy. This method inspired from probability matching relies on p_{m_j} build upon past observations. Upon executing mutation m_t, a reward $r_{t,m_t} = 1 - \ell_{t,m_t}$ is observed based on the *SW* metric (see Sec. II-D and Table I). Finally, $p_{(a,c)}$ for the selected action a within the selected local context c, is updated according to the observed binary reward $r_{t,m_t} \in \{0, 1\}$. The detailed update mechanism for $p_{(a,c)}$ (denoted $p_{Idx_a(c)}$) is presented in the next section III-A3.

According to this problem setup, the subsequent subsections will examine the key point formalization of molecular context c, describe the concept of sleeping mutations, and explicit the evaluation of molecule realism.

1) Molecular Context: For each action, the molecular context must encode both the ECFP$_d$ identifier (ID_{ECFP_d}) and, when applicable, the specific option index for actions with multiple variants. Specifically, to fully characterize the contextual state space, *AddA* contexts require the consideration of the index of the atom type to be added ($Idx_{\alpha\tau}$) within the candidate atom set, while *ChB* contexts require the bond type index (Idx_β). Because the number of possible contexts varies across different action types, molecular context analysis must be performed on an action-specific basis.

The function $Idx_a(c)$ generates a unique identifier for each known and realistic context c of each action a. It is computed from the positional index of ID_{ECFP_d} within the valid ECFP$_d$ list (Idx_{ECFP_d}) combined with either $Idx_{\alpha\tau}$ for *AddA* or Idx_β for *ChB*. Fig. 2 illustrates this encoding scheme using acetylsalicylic acid molecular structure, with diameter 0 ECFP list containing 33 ID_{ECFP_0}. Fig. 2 shows the Idx_{ECFP_0} values and the corresponding Idx_{AddA} and Idx_{RmA} indices calculated for all *AddA* and *RmA* valid mutations applicable to this molecular system. For example, carbon atoms where *AddA* operations are valid (Fig. 2a) correspond to two distinct ECFP$_0$ environments (Idx_{ECFP_0} = 24 and 33). Since four atom types (C, N, O, F) can be added to each environment, this results in 8 different Idx_{AddA} values (24, 33, 58, 67, 92, 101, 126, 135). Similarly, the oxygen atoms where *AddA* operations are valid exhibit only one Idx_{ECFP_0} value (8), resulting in 4 potential Idx_{AddA} values (8, 42, 76, 110). In total, 12 Idx_{AddA} values are valid for step 1. An inverse function applied to these Idx_a enables the recovery of the contextual ID_{ECFP_0}. Finaly, as shown, identical index values may occur multiple times within a single algorithmic iteration.

2) Dynamic mutations set and sleeping arms: Since the molecular graph undergoes structural modifications across successive steps, the valid mutations set exhibits dynamic variations that correspond to the evolving configuration of atoms and bonds within the molecular graph. For example, as shown in Fig. 2a, the carbon atom at position 1 of the acetylsalicylic acid molecule has already formed 4 bonds and therefore cannot undergo any *AddA* mutation. The 4 theoretical arms (namely, the addition of carbon, nitrogen, oxygen, and fluorine atoms at position 1) therefore remain sleeping. Similarly, the removal of the same atom (Fig. 2b) would necessarily result in molecular cleavage into two separate entities; consequently, the mutation $(RmA, 1)$ is sleeping at this step. Based on the total mutations frequency calculated from the default maximum heavy atom count per molecule (38), each action exhibits stepwise variation in sleeping and awake mutations frequencies, with sleeping mutations theoretically predominating (refer to $n_{awake-arm}$ and $n_{sleeping-arm}$ in both panels of Fig. 2).

Given that molecular contexts are intrinsically linked to mutations, the set of observable ID_{ECFP_d} varies accordingly, rendering the collection of observable mutation contexts inherently non-stationary in nature. At any given time step, the vast majority of theoretical $Idx_a(c)$ remains unavailable and is therefore classified as non-observable ID_{ECFP_0}, as demonstrated by the comparison of observable and non-observable Idx_a frequencies in Fig. 2. For instance, Fig. 2 shows that the acetylsalicylic acid molecule lacks both F and N atoms. Consequently, all ID_{ECFP_0} related to these atoms remain inactive at step 1. If mutation chosen adds a N atom at position 12 (Fig. 2a), ID_{ECFP_0} related to this atom will be activated at Step 2. Since the O atom at position 12 no longer maintains a bond with hydrogen, the *AddA* action on this atom will become inactive at this step and related context indices 8, 76, and 110 will consequently become non-observable Idx_{AddA}.

3) Evaluation of Molecule Realism and Probability Matching : In II-D, *SW* metric was introduced as a measure for evaluating the proportion of *silly* bits in molecular representations. In this study, *SW* serve a dual analytical purpose both detailed in this subsection: they are employed both as a criterion for selecting the most relevant actions during the molecular generation process and as a comparative metric for evaluating the realism of molecules generated by *EvoMol-RL* against those produced by the standard *EvoMol* approach.

At each step, mutation m_j (tuple (a, idx_p)) selection is probability-matching based, using empirical success rates of corresponding $Idx_a(c)$ to pass the *SW* filter as implicit rewards, defined in equation (4). let $n_{Use}(Idx_a(c))$ be the frequency of $Idx_a(c)$ selection throughout the process and

TABLE I
NOTATION USED IN THE SLEEPING BANDIT FORMULATION OF *EvoMol-RL*.

Symbol	Meaning
Ω	Whole universe of distinct mutations
k	Total number of distinct mutations
$\mathcal{M}$	Set of distinct mutations encountered
k_T	Total number of distinct mutations encountered
$\mathcal{M}_t$	Set of mutations valid ("awake") at step t
k_t	Number of distinct mutations valid ("awake") at step t
m_t	Mutation actually applied at step t
ℓ_{t,m_t}	Loss of m_t at step t (1 if mutant fails SW filter, 0 otherwise)
$\text{Reg}_T^{\text{sleep}}$	Adversarial sleeping-bandit regret up to step T

$n_{Success}(Idx_a(c))$, its frequency to generate a new molecule passing *SW* filter (without *silly* bit), $Idx_a(c)$ success rate, $p_{Idx_a(c)}$ is computed as follow:

$$p_{Idx_a(c)} = \frac{n_{Success}(Idx_a(c))}{n_{Uses}(Idx_a(c))} \tag{4}$$

For each valid mutation $m_j \in \mathcal{M}_t$ ($j \in \{1, ..., k_t\}$) (tuple (a, idx_p) in $Valid_{(a,idx_p)}$); m_j success rate, p_{m_j}, is the success rate of the corresponding action context $p_{Idx_a(c)}$. To provide opportunities for unselected contexts, a minimum success rate, denoted as p_{min}, is initially defined by the user. Subsequently, to ensure data smoothing, valid mutations success rates undergo normalization according to equation (5):

$$w_{m_j} = \frac{max(p_{min}, p_{m_j})}{\sum\limits_{m_j \in \mathcal{M}_t; j \in \{1,...,k_t\}} max(p_{min}, p_{m_j})} \tag{5}$$

The list of weights w_{m_j} is then used to perform weighted randomization of a mutation (a, idx_p). This weighting scheme enables probabilistic selection where each mutation's likelihood of being chosen is proportional to its corresponding weight value, thereby implementing a stochastic sampling mechanism that favors tuples with higher associated weights while still maintaining the possibility of selecting those with lower weights.

By incorporating existing ID_{ECFP_d} within $Idx_a(c)$ and leveraging their empirical efficacy in generating chemically realistic molecules, the proposed method is expected to reduce the number of newly generated molecules rejected by the *SW* filter compared to those produced by *EvoMol* algorithm.

B. Method to balance exploration and exploitation

The exploitation-exploration trade-off balances using known high-reward actions versus testing new potentially better actions. Three strategies were tested in this protocol. Constant strategy maintains the same exploration rate ε throughout learning to randomly explore, with low values favoring exploitation and high values promoting exploration. Moderate ε values (e.g., 0.1, 0.2) provide balanced exploration-exploitation trade-off whereas higher values (from 0.3) prioritize exploration with frequent random action selection. Epsilon-greedy and Power Law (PL) strategies start with high exploration ($\varepsilon_0 = 1$) that gradually decrease over time, allowing initial broad search followed by focused exploitation. In formulation of exploration rate following a power law distribution (7); α value determine how rapidly exploration decreases over time. This function is done by λ in (6) equation.

$$\varepsilon_{Greedy}(t) = max(\varepsilon; \varepsilon_0.e^{-\lambda.t}) \tag{6}$$

$$\varepsilon_{PL}(t) = max(\varepsilon; \frac{\varepsilon_0}{(1+t)^{\alpha}}) \tag{7}$$

At each step t, exploration versus exploitation is determined by comparing the current epsilon value, ε_{curr} (computed with (6) for Epsilon-greedy approach or (7) for PL strategy) against a random value $alea \in [0; 1]$.

These approaches were tested with different values for ε; α and λ parameters to determine optimal balance between immediate rewards and long-term performance discovery.

C. EvoMol-RL algorithm

EvoMol-RL algorithm uses the *EvoMol* algorithm as detailed in [11] (source code available at https://github.com/jules-leguy/EvoMol). The primary modification involves replacing the SearchNeighbour function with the following implementation[1]:

Compute $Valid_{(a,idx_p)}$ in $CurrMol$ (Sec. II-E)
$\mathcal{M}_t = Valid_{(a,idx_p)}$
$m_t =$
- **if** random$(alea) > \varepsilon_{Curr}$ (6) (7):
  random$(m_j) \in \mathcal{M}_t$
- **else for** each $m_j \in \mathcal{M}_t$; $j \in 1, ..., k_t$:
  - Get $Idx_a(c)$
  - Compute $p_{Idx_a(c)}$ (4)
  - Compute w_{m_j} (5)
  Weighted random$(m_j) \in \mathcal{M}_t$

Apply m_t to $CurrMol$ generating $NewMol$
Increment $n_{Uses}(Idx_a(c)_t)$

EvoMol-RL algorithm subsequently verifies whether the $NewMol$ satisfies the filtering criteria and observes the reward, r_{t,m_t}, which depends on whether $NewMol$ represents an OF improvement. The success frequency of $Idx_a(c)_t$ is then updated according to the definition provided in (8).

$$n_{Success}(Idx_a(c)_t) = n_{Success}(Idx_a(c)_t) + r_{t,m_t} \tag{8}$$

IV. Empirical Evaluation

A. Experimental Protocol

To systematically evaluate the impact of key algorithmic parameters, an exhaustive enumeration of hyperparameters combinations was performed over ECFP diameter $d \in \{0, 2\}$, $\varepsilon \in \{0.1, 0.2, 0.3\}$, and exploration strategies: epsilon-greedy ($\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}\}$), Power Law ($\alpha \in [0.25, 0.4]$ in steps of 0.05) and constant epsilon value fixed at the corresponding ε throughout the run.

A series of ten experimental runs per configuration was conducted using predetermined constant seeds to ensure reproducible comparisons between different script configurations. Each run, started from acetylsalicylic acid molecule, was executed for a duration of 500 steps, based on the assumption of fast model convergence, with the *SW* objective function serving as the evaluation metric. Only one action is performed between each evaluation to avoid having to deal with delayed rewards. The optimization process involves to maximize the occurrence of realistic molecules (those passing the *SW* filter). Primary criterion is the rate of generated molecules passing the *SW* filter throughout the run, *i.e.* percentage of realism. Secondary criterion is the rate of generated molecules not already in population P, *i.e.* percentage of novelty. For all

[1]The comprehensive Python program is hosted at https://github.com/gmilha/EvoMol_ICTAI

TABLE II
PERCENTAGE REALISM AND PERCENTAGE NOVELTY MEANS AND STANDARD DEVIATIONS OF MOLECULES GENERATED BY *EvoMol* AND *EvoMol-RL*.

ε	**% realism mean ± std**		**% novelty mean ± std**	
EvoMol	0.51 ± 0.01		0.58 ± 0.01	
EvoMol-RL	***ECFP*$_0$**	***ECFP*$_2$**	***ECFP*$_0$**	***ECFP*$_2$**
0.1	0.78 ± 0.01	0.82 ± 0.01	0.36 ± 0.01	0.32 ± 0.01
0.2	0.73 ± 0.01	0.76 ± 0.01	0.41 ± 0.01	0.38 ± 0.01
0.3	0.68 ± 0.01	0.69 ± 0.01	0.46 ± 0.01	0.44 ± 0.01

configurations, both criteria obtained with ECFP$_0$ and ECFP$_2$ are compared to actual *EvoMol* results.

We would like to emphasize that by initializing the population with the same single molecule and by not activating any of *EvoMol*'s specific parameters to favor the diversity of solutions, we are deliberately setting the algorithm in such a way that the two evaluated criteria are only influenced by the choice of mutations. This special case allows us to better assess the impact of our method. We therefore expect to observe an increase in realism, the objective function, at the cost of a decrease in novelty.

Among all tested parameters, only ε meaningfully influences realism: variability remains 1% regardless of exploration method, λ, or α. Consequently, Table II and results analysis section (IV-B) only shows the best-performing configuration for each (ε, ECFP diameter) pair. The peak realism is achieved by Power-Law ($\alpha = 0.35$) and Greedy ($\lambda = 0.10$). Moreover, because Power-Law is slightly faster and more stable, it serves as the reference for reporting means and inter-run standard deviations.

B. Results Analysis

Over the 10 runs for each method we compute and report in Table II: 1) The realism percentage mean of generated molecules (before filtering) along with its standard deviation; 2) The novelty percentage mean along with its standard deviation.

Despite the limited structural information provided by ECFP$_0$, a substantial increase in pre-filtering realism percentage is observed, achieving 17% ($\varepsilon = 0.3$) to 27% ($\varepsilon = 0.1$) higher realism relative to *EvoMol*. When the richer ECFP$_2$ fingerprint is used instead, the best configuration reaches an average of 82% realistic molecules for $\varepsilon = 0.1$. For this optimal ε configuration, a *Kruskal-Wallis* test is subsequently conducted to highlight inequalities among the mean realism percentages across *EvoMol*, *EvoMol-RL* ECFP$_0$, and ECFP$_2$. Herein, the *Kruskal-Wallis* test demonstrates significant statistical differences in realism performance between the three methods under investigation (p-value < 0.001). Thus, a *Wilcoxon* signed rank tests is performed over the realism to determine whether the results between each pair of methods differ significantly . The significance will be further given with the p-value.

As illustrated in Table II and in Fig. 3, *EvoMol-RL* significantly outperforms the baseline (*EvoMol*) in pre-filtering realism percentage for both ECFP$_0$ (p-value < 0.002) and ECFP$_2$ (p-value < 0.002). When comparing the two fingerprint granularities (ECFP$_0$ and ECFP$_2$), *EvoMol-RL* significantly delivers higher realism when rewards are computed with ECFP$_2$ rather than ECFP$_0$ (p-value < 0.002) (See Table II). Since the larger diameter ($d = 2$) captures chemical context beyond the focal atom that ECFP$_0$ ignores, this richer and more discriminative feature set gives the *RL* policy clearer signals about which mutations are chemically plausible. This allows it to converge more quickly and to surpass the ECFP$_0$ variant across the entire ε range, with the largest margin at $\varepsilon = 0.1$.

Furthermore, Fig. 3 shows that the new *RL* module rapidly identifies effective mutations within the first $\approx$ 20 steps, achieving a success rate that quickly surpasses and consistently outperforms the baseline.

Finally, the simultaneous drop in novelty and rise in realism (See Table II) is an expected indicator of effective reinforcement learning. Once the policy discovers mutation patterns that consistently produce realistic structures, it shifts from extensive exploration to targeted exploitation, revisiting those chemotypes more often. This deliberate focus lowers novelty but confirms that the agent is focusing on reliable, chemically plausible regions of the search space.

V. CONCLUSION

In this article, the concept of context linked to an action on a molecular graph was introduced, paving the way for the use of reinforcement learning in molecular generation. The main results of this article is the experimental demonstration that the addition of reinforcement learning for mutation selection in an evolutionary algorithm, here *EvoMol*, allows the algorithm to be better guided towards solutions satisfying an objective function, the realism of molecules in this study. For this crucial objective for chemists, a significant improvements regardless

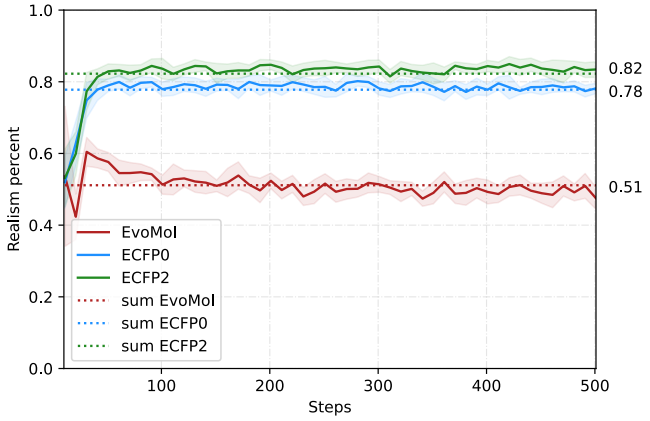

Fig. 3. Percentage of realistic molecules generated across testing steps from the Power Law configuration with parameters α=0.35 and ε=0.1. The solid line represents the mean values, with standard deviations indicated by the highlighted regions, computed using a sliding window of 10 steps. Results obtained with *EvoMol* and *EvoMol-RL* with consideration of ECFP$_0$ or ECFP$_2$ are respectively depicted in red, blue and green.

of the methods and their hyperparameters was measured. Furthermore, the method converges rapidly and exhibits low variability between the different executions for a given set of parameters. It has also been shown that a larger context, using $ECFP_2$, allows a better policy learning.

The definition of the molecular context and this first application make it possible to consider direct extensions of this work in the future. Other reinforcement algorithms, such as EXP3 [33] to name just one, could be studied. In this study, the exploration capabilities of the original algorithm were inhibited in order to measure the specific contribution of reinforcement learning to mutation selection. A more exhaustive study, using all the mechanisms and varying the parameters, should be carried out. Other objective functions could also be tested, especially if their computational cost is very high, as in this case avoiding unnecessary evaluations may be critical. The next step in this work could be to look at deferred rewards, by letting the algorithm perform several actions before evaluating the molecule and calculating the reward to be assigned to each action. This extension is important in order to consider hard optimization problems requiring temporary non-improving our invalid solutions.

REFERENCES

[1] R. S. Bohacek, C. McMartin, and W. C. Guida, "The art and practice of structure-based drug design: a molecular modeling perspective," Medicinal Research Reviews, vol. 16, no. 1, pp. 3–50, 1996.

[2] L. Ruddigkeit, R. van Deursen, L. C. Blum, and J.-L. Reymond, "Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17," Journal of Chemical Information and Modeling, vol. 52, no. 11, pp. 2864–2875, Nov. 2012.

[3] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman, "ZINC: a free tool to discover chemistry for biology," Journal of Chemical Information and Modeling, vol. 52, no. 7, pp. 1757–1768, Jul. 2012, publisher: American Chemical Society.

[4] A. Gaulton, A. Hersey, M. Nowotka, A. P. Bento, J. Chambers, D. Mendez, P. Mutowo, F. Atkinson, L. J. Bellis, E. Cibrián-Uhalte, M. Davies, N. Dedman, A. Karlsson, M. P. Magariños, J. P. Overington, G. Papadatos, I. Smit, and A. R. Leach, "The ChEMBL database in 2017," Nucleic Acids Research, vol. 45, no. D1, pp. D945–D954, Jan. 2017.

[5] M. H. S. Segler, M. Preuss, and M. P. Waller, "Planning chemical syntheses with deep neural networks and symbolic AI," Nature, vol. 555, no. 7698, pp. 604–610, Mar. 2018.

[6] C. W. Coley, W. Jin, L. Rogers, T. F. Jamison, T. S. Jaakkola, W. H. Green, R. Barzilay, and K. F. Jensen, "A graph-convolutional neural network model for the prediction of chemical reactivity," Chemical Science, vol. 10, no. 2, pp. 370–377, Jan. 2019.

[7] T.-L. Phan, K. Weinbauer, M. E. G. Laffitte, Y. Pan, D. Merkle, J. L. Andersen, R. Fagerberg, C. Flamm, and P. F. Stadler, "SynTemp: Efficient extraction of graph-based reaction rules from large-scale reaction databases," Journal of Chemical Information and Modeling, vol. 65, no. 6, pp. 2882–2896, Mar. 2025, publisher: American Chemical Society.

[8] W. Jin, R. Barzilay, and T. Jaakkola, "Junction tree variational autoencoder for molecular graph generation," in International Conference on Machine Learning. PMLR, Jul. 2018, pp. 2323–2332.

[9] S. Voloboev, "A review on fragment-based de novo 2d molecule generation," arXiv preprint, 2024, 2405.05293.

[10] J. H. Jensen, "A graph-based genetic algorithm and generative model/Monte Carlo tree search for the exploration of chemical space," Chemical Science, vol. 10, no. 12, pp. 3567–3572, 2019.

[11] J. Leguy, T. Cauchy, M. Glavatskikh, B. Duval, and B. Da Mota, "EvoMol: a flexible and interpretable evolutionary algorithm for unbiased de novo molecular generation," Journal of Cheminformatics, vol. 12, no. 1, p. 55, Sep. 2020.

[12] E. S. Henault, M. H. Rasmussen, and J. H. Jensen, "Chemical space exploration: how genetic algorithms find the needle in the haystack," PeerJ Physical Chemistry, vol. 2, p. e11, Jul. 2020.

[13] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, "Automatic chemical design using a data-driven continuous representation of molecules," ACS Central Science, vol. 4, no. 2, pp. 268–276, Feb. 2018.

[14] D. C. Elton, Z. Boukouvalas, M. D. Fuge, and P. W. Chung, "Deep learning for molecular design—a review of the state of the art," Mol. Syst. Des. Eng., vol. 4, no. 4, pp. 828–849, 2019.

[15] Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen, "Molecular de-novo design through deep reinforcement learning," Journal of Cheminformatics, vol. 9, no. 48, p. 14, Sep. 2017.

[16] Z. Zhou, S. Kearnes, L. Li, R. N. Zare, and P. Riley, "Optimization of molecules via deep reinforcement learning," Scientific Reports, vol. 9, no. 1, p. 10752, Jul. 2019, bandiera_abtest: a Cc_license_type: cc_by Cg_type: Nature Research Journals Number: 1 Primary_atype: Research Publisher: Nature Publishing Group Subject_term: Cheminformatics Subject_term_id: cheminformatics.

[17] S. Gow, M. Niranjan, S. Kanza, and J. G. Frey, "A review of reinforcement learning in chemistry," Digital Discovery, vol. 1, no. 5, pp. 551–567, Oct. 2022, publisher: RSC.

[18] B. Sridharan, A. Sinha, J. Bardhan, R. Modee, M. Ehara, and U. D. Priyakumar, "Deep reinforcement learning in chemistry: A review," Journal of Computational Chemistry, vol. 45, no. 22, pp. 1886–1898, 2024, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.27354.

[19] D. Polykovskiy, A. Zhebrak, B. Sanchez-Lengeling, S. Golovanov, O. Tatanov, S. Belyaev, R. Kurbanov, A. Artamonov, V. Aladinskiy, M. Veselov, A. Kadurin, S. Johansson, H. Chen, S. Nikolenko, A. Aspuru-Guzik, and A. Zhavoronkov, "Molecular sets (MOSES): a benchmarking platform for molecular generation models," Frontiers in Pharmacology, vol. 11, 2020, publisher: Frontiers.

[20] P. Renz, D. Van Rompaey, J. K. Wegner, S. Hochreiter, and G. Klambauer, "On failure modes in molecule generation and optimization," Drug Discovery Today: Technologies, vol. 32-33, pp. 55–63, Dec. 2019.

[21] P. Schwaller, D. Probst, A. C. Vaucher, V. H. Nair, D. Kreutter, T. Laino, and J.-L. Reymond, "Mapping the space of chemical reactions using attention-based neural networks," Nature Machine Intelligence, vol. 3, no. 2, pp. 144–152, Feb. 2021, publisher: Nature Publishing Group.

[22] T. Cauchy, J. Leguy, and B. D. Mota, "Definition and exploration of realistic chemical spaces using the connectivity and cyclic features of ChEMBL and ZINC," Digital Discovery, vol. 2, no. 3, pp. 736–747, 2023, publisher: Royal Society of Chemistry.

[23] C. Bannwarth, S. Ehlert, and S. Grimme, "Extended tight-binding quantum chemistry methods," WIREs Computational Molecular Science, vol. 11, no. 2, p. e1493, 2021.

[24] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," Journal of Chemical Information and Modeling, vol. 50, no. 5, pp. 742–754, May 2010, publisher: American Chemical Society.

[25] J. Leguy, "Combinatorial search lead by machine learning for molecular chemistry," Ph.D. dissertation, Université d'Angers, 2022.

[26] D. Weininger, "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules," Journal of Chemical Information and Computer Sciences, vol. 28, no. 1, pp. 31–36, 1988.

[27] P. Walters, "silly_walks," Jan. 2022, original-date: 2020-12-26T17:25:07Z. [Online]. Available: https://github.com/PatWalters/silly_walks

[28] S. Bühlmann and J.-L. Reymond, "ChEMBL-likeness score and database GDBChEMBL," Frontiers in Chemistry, vol. 8, Feb. 2020.

[29] T. Lattimore and C. Szepesvári, Bandit algorithms. Cambridge University Press, 2020.

[30] R. S. Sutton, A. G. Barto et al., Reinforcement learning: An introduction. MIT press Cambridge, 1998, vol. 1, no. 1.

[31] R. Kleinberg, A. Slivkins, and E. Upfal, "Bandits and experts in metric spaces," Journal of the ACM (JACM), vol. 66, no. 4, pp. 1–77, 2019.

[32] R. Combes, M. S. Talebi Mazraeh Shahi, A. Proutiere et al., "Combinatorial bandits revisited," Advances in neural information processing systems, vol. 28, 2015.

[33] G. Neu, "Explore no more: Improved high-probability regret bounds for non-stochastic bandits," Advances in Neural Information Processing Systems, vol. 28, 2015.