UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA AGRONÓMICA, ALIMENTARIA Y DE BIOSISTEMAS

MÁSTER EN BIOLOGÍA COMPUTACIONAL

Biotecnología – Biología Vegetal

Estructura de Macromoléculas

Computational exploration of chemical space using generative models

MASTER FINAL THESIS

Author: Izana Alcalde Cercadillo

Professional tutor: Carlos Óscar Sorzano

Academic tutor: María Garrido Arandia

July, 2026

Table of Contents

List of figures

List of tables

List of abbreviations

Artificial Intelligence (AI)
Basic Local Alignment Search Tool (BLAST)
Curriculum Learning (CL)
Command Line Interface (CLI)
Difference between Augmented and Posterior (DAP)
Deep Reinforcement Learning (DRL)
Graphical User Interface (GUI)
Hydrogen Bond Donors (HBD)
Hydrogen Bond Acceptors (HBA)
High-throughput Screening (HTS)
Large Language Models (LLMs)
Maximum Common Substructure (MCS)
Molecular Weight (MW)
Multi-Objective Optimization (MOO)
Multiple Sequence Alignment (MSA)
Negative Log-Likelihood (NLL)
Natural Language Processing (NLP)
Open Drug Discovery Toolkit (ODDT)
Protein Data Bank (PDB)
Quantitative Estimate Drug-likeness (QED)
Reinforcement Learning (RL)
Recurrent Neural Networks (RNNs)
Synthetic Accessibility Score (SASCore)
Staged Learning (SL)
Simplified Molecular Input Line Entry System (SMILES)
Transfer Learning (TL)
Topological Polar Surface Area (TPSA)
Virtual Drug Screening (VDS)

Abstract

Modern drug discovery is shifting from the passive screening of static chemical libraries toward the active, computational exploration of chemical space. While traditional virtual screening is restricted to predefined databases, generative models such as REINVENT4 treat molecular design as an optimization problem, enabling a transition from a "find-and-filter" to a "design-and-evolve" paradigm. However, its reliance on manual installation, command-line execution and complex configuration files limit its accessibility to non-computational researchers. Furthermore, Scipion-chem, a platform designed to facilitate virtual screening workflows by integrating diverse external programs into reproducible, automated pipelines, lacked any de novo molecular generation capability. This work addresses both limitations through the development of a dedicated plugin that integrates REINVENT4 within Scipion-chem. Three protocols were implemented covering the main functionalities of REINVENT4 automating installation, configuration and execution, while ensuring interoperability with Scipion's existing tools. The plugin was validated through a case study based on human adenovirus type 5 in which a novel molecule set was generated from a reference library of SMILES previously identified as strong docking candidates against the target. The generated set was then docked in parallel with the reference and a control set against the identified binding pockets. Despite a lower docking success rate, the generated molecules achieved the most favourable binding energies in the most relevant pockets, confirming a directed and effective exploration of chemical space. Overall, this work presents the integration of a de novo generative tool within Scipion-chem, completing its virtual screening workflow and making generative molecular design accessible to researchers without advanced computational expertise.

1.Introduction

1.1. Evolution of drug discovery

For decades, the discovery of novel therapeutic agents has been intrinsically bound to High-throughput Screening (HTS) methodologies. The introduction of HTS historically revolutionized the pharmaceutical industry by transitioning drug discovery from a manual process to an industrialized and automated paradigm. By benefiting from techniques like robotics, microplates or microfluidics, HTS enabled researchers to physically evaluate novel compounds in a more efficient way[1]. However, the overall drug discovery pipeline remains facing limitations. While HTS is highly proficient at generating initial hits, the process requires years of downstream optimization, only for most identified candidates to be discarded in later stages due to poor solubility, high toxicity, or lack of efficacy[2].

With the rise of structural biology and the exponential growth of high-performance computing resources, the screening paradigm expanded toward virtual drug screening (VDS) transitioning into a digital environment. By using computational techniques based on either the molecular structure of the receptor or known ligands, VDS enables the rapid evaluation of massive compound databases, drastically reducing the time and cost associated with drug design[3]. However, despite these computational advancements, both techniques still operate under a conceptual framework known as “find-and-filter”, where it is assumed that the optimal solution already exists within a predefined container[4].

1.2. Limitations of predefined chemical libraries

Under this premise, the drug discovery process is structured as a reductive funnel where sequential filtering is applied to exclude molecules from an initial library[4,5]. In practice, these screening workflows rely on molecular databases such as ChEMBL[6], PubChem[7] or ZINC[8], which effectively act as the static boundaries of the searchable space.

Therefore, the fundamental bottleneck of the “find-and-filter” paradigm relies on the disparity between existing databases and the true magnitude of chemical space. In cheminformatics, the chemical space for drug-like molecules (defined as those which comply with empirical filters such as Lipinski’s rule of 5[9]) is theoretically estimated to comprise between 10^{60} and 10^{63} possible molecules[10]. Exploring this immense combinatorial space via static screening, is computationally impossible. Even modern VDS projects that successfully scale workflows to

screen billions of compounds[11] barely scratch an infinitesimal fraction of this universe.

Furthermore, these databases are predominantly populated by compounds that are easily synthesized and already well validated, a phenomenon known as synthetic bias, formalized by Connor Coley[12]. This induces a profound structural poverty or lack of diversity in molecular scaffolds. Consequently, when a screening project confronts a novel or complex biological target, it fails, as chemical solutions do not exist within preexisting catalogues[13].

Bound by both the physical limits of data storage, and the structural redundancy of existing catalogues, the traditional VDS paradigm inevitably reaches a ceiling, and overcoming this restriction requires a shift from static filtering toward generative and dynamic exploration algorithms.

1.3. Paradigm shift to artificial intelligence

To overcome the limitations of static databases, cheminformatics has undergone a reconceptualization driven by artificial intelligence (AI). AI has transitioned from being an optimization tool used to accelerate compound discarding or target validation, into a creative engine for de novo molecular generation[14]. This marks the definitive transition from “find-and-filter” to “design-and-evolve”, where rather than examining existing libraries, generative mathematical models autonomously design novel molecular scaffolds from scratch, ensuring the direct exploration of previously uncharted regions[15].

Within this paradigm, natural language processing (NLP) models and large language models (LLMs) have been adapted to the field of chemistry, treating molecular structures not as graphs or physical coordinates, but as a formal textual language based on the SMILES (Simplified Molecular Input Line Entry System)[16] notation. Under this computational framework, SMILES strings are processed as sentences, while individual atoms and bonds function as tokens within a specialized vocabulary[17].

To process this chemical language, deep learning architectures such as recurrent neural networks (RNNs) and transformers, are trained on massive text data of known compounds expecting them to capture structural dependencies within the sequence[18,19]. However, a challenge in molecular NLP is that minor syntax errors yield structurally impossible molecules. This requires the integration of a mechanism to reward precision and penalize invalid outputs.

1.4. Deep reinforcement learning and multi-objective optimization

In this context, deep reinforcement learning (DRL) provides the mathematical framework to treat drug discovery as a dynamic optimization problem[20]. The system is structured as a closed loop where an agent (the generative neural network) generates SMILES strings with the desired properties which, if valid, grants a reward to the model. This iterative process progressively maximizes the scores of the generated solutions[21].

However, the main challenge within this reward mechanism is multi-objective optimization (MOO). In medical chemistry the properties required for a clinical candidate are excluding or exhibit negative correlation, that is, improving one property usually degrades another exponentially reducing the probability of finding an ideal candidate within traditional libraries[22].

To resolve this conflict, the reward mechanism in DRL implements two main strategies: scalar desirability functions (which aggregates several metrics into a single weighted reward function)[23] and optimization based on Pareto dominance[22]. Consequently, the algorithms do not pursue a single, and usually non-existent ideal structure. Instead, they direct the exploration toward the Pareto front, a set of optimal solutions where it is mathematically impossible to improve one parameter without worsening another, thereby offering a diverse range of optimal candidates for clinical development[24].

To translate these complex frameworks into practical drug discovery pipelines, specialized computational tools are required. Among the generative tools designed to implement this DRL strategies, this work focuses on REINVENT[20,25,26].

1.5. REINVENT

Developed as an open-source tool by AstraZeneca, REINVENT utilizes RNNs and transformer architectures for molecular generation. Its software architecture is highly versatile relying on probabilistic models to capture chemical syntax while supporting diverse learning strategies. However, despite its computational power, implementing REINVENT poses some technical barriers, primarily due to the following software engineering and usability challenges:

1. **Installation:** Setting up the software relies on manual configuration which often triggers dependency conflicts, complicated local installations and cross-platform reproducibility.

2. **Command line complexity:** REINVENT operates entirely via command line interface (CLI) introducing a steep learning curve for non-computational researchers who must manually manage a high volume of execution files, directory paths and terminal commands.

3. **Configuration management:** Executing a basic run requires creating complex, structured configuration files. The user must know a vocabulary of highly specific parameter names, nested schemas and syntax rules where a single error can abort the entire execution.

4. **Data interoperability:** REINVENT outputs molecular data primarily as text SMILES lists or CSV files. To make these results biologically meaningful, researchers must manually handle file format conversion and feed them into external structural biology software (e.g., generating 3d coordinates in SDF or PDB formats).

5. **Loss of traceability:** Running autonomous scripts makes it difficult to track the process and replicate it for another analysis, losing reproducibility across different laboratories.

These technical difficulties are not unique to REINVENT, and they represent a common challenge found throughout the entire VDS workflow and among the different existing software. This precise fragmentation of tools and lack of interoperability is what the Scipion-chem platform aims to solve. By benefiting from Scipion-chem's unified workspace, these software limitations can be mitigated, providing the ideal architectural foundation to integrate REINVENT4.

1.6. Scipion ecosystem

Originally developed for structural biology protocols[27–29], Scipion has extended its architecture to the field of VDS through Scipion-chem[30]. This extension unified the fragmented VDS tools under a single interoperable framework by enforcing modularity, traceability and reproducibility.

This software automatically manages external installations and intermediate file format conversion, registering execution parameters and creating followable VDS pipelines, everything with interconnected protocols within a graphical user interface (GUI)[27,30].

Currently, Scipion-chem includes protocols for traditional VDS tasks, including molecule importing, structural preparation, ligand filtering, molecular docking and molecular dynamics, counting with software like Schrödinger[31], AutoDock[32], LePhar[33]or Fpocket[34]. However, it

lacks any protocol for de novo molecular generation.

Consequently, including REINVENT4 within Scipion-chem addresses a major technical gap, as it simplifies the functionality of an advanced generative AI while directly coupling it with Scipion's preexisting structural validation tools, effectively unifying the entire VDS research workflow.

2. Objectives

The aim of this work is to design, develop and integrate a plugin in the Scipion-chem platform that includes the de novo molecule generation capabilities of REINVENT4's. By doing so, this project seeks to establish an automated reproducible and closed-loop pipeline for VDS. To do so the following objectives have been established.

1. Design and implement python-based protocols inside the Scipion-chem environment simplifying the complexity of REINVENT4 and making it accessible for the final researcher.

2. Ensure compatibility between the newly developed plugin and the pre-existing chemoinformatic tools within Scipion-chem allowing its integration into full VDS workflows.

3. Validate the developed tool by the study of a real practical case against a therapeutic target demonstrating viability, efficacy and robustness of the new integrated system.

3. Materials and Methods

3.1 REINVENT4

The REINVENT framework has established a solid record in de novo molecular design, evolving from its initial proposals to its current release[20,25,26]. The current version is fully grounded in its predecessors, rewritten as a more unified version, being defined as the most advanced and efficient variant to date. Consequently, this version was selected as the central generative engine for the development of the Scipion plugin in this work.

REINVENT4 is an open-source platform developed in Python that generates new molecules and trains models through generative AI. Its primary objective is to iteratively propose novel structures that satisfy user defined biological, chemical and biophysical properties.

3.1.1 Molecular representation

REINVENT treats drug design as a natural language generation task. To achieve this, it utilizes SMILES linear notation, where molecular structures are encoded as sequences of chemical characters or tokens. The probability of generating syntactically valid sequences is modelled using two deep learning architectures.

a. **Recurrent neural networks**: Compute the probability of the next token conditioned exclusively on the preceding characters of the string[35,36].

b. **Transformer models:** Evaluate long range correlations within the SMILES chain allowing the model to capture complex structural patterns[37,38].

These models are referred to as agents and describe a distribution over sequences of tokens from a fixed vocabulary learned during training, ending each sequence with a special termination token. The probability of generating a sequence is calculated as the product of the conditional probabilities of each token given the preceding ones. From this, REINVENT4 computes the Negative Log-Likelihood (NLL), which serves as a measure of how expected a generated molecule is according to the model.

To generate new molecules, REINVENT4 supports two decoding strategies. Multinomial sampling selects each token stochastically according to its probability controlled by a

temperature parameter, whereas beam search in contrast, is a deterministic strategy that retains the most probable sequences at each step, ensuring unique outputs at a higher computational cost.

3.1.2 Generative methods

To address different purposes, REINVENT4 organizes learning architecture into four distinct generative methodologies, each of them with its own topological constraints and a specific workflow.

1. **De novo generation (Reinvent):** Builds molecular structures from scratch. It is an unconstrained method focused on discovering entirely novel scaffolds[20].

2. **Scaffold decoration (Libinvent):** Focuses on generating R-groups (side chains) at a predefined attachment point from a rigid core scaffold provided by the user[39].

3. **Fragment Linking (LinkInvent):** Designs linkers (structural bridges) to interconnect two independent given warheads[40].

4. **Molecule to molecule optimization (Mol2Mol):** Based on transformer architectures, takes a target structure as reference to generate structural analogues within a given similarity radius by optimizing properties[37].

3.1.3 REINVENT4 software architecture

REINVENT4 operates as a CLI tool with no continuous memory between runs. Every execution is initialized by a configuration file provided by the user in TOML or JSON format. This file contains generator selection, parameter names and values, and absolute paths, all organized within a specific structure. Throughout the simulation, the software will provide real-time logs about the evolution of the run and will record final outputs into structured CSV files. These files contain diverse information depending on the run mode but overall will contain generated SMILES together with their respective NLL.

The execution of REINVENT4 is centred around a Prior model, which is a static generative base model that has been pre-trained on millions of compounds extracted from ChEMBL[6]. Its main function is to act as a linguistic model which knows basic chemistry rules, ensuring that generated sequences are valid. Each of the four generative methods has its own dedicated

Prior, trained specifically for that generation task and therefore differing in their vocabulary. To utilize this model, REINVENT4 structures its workflow into three distinct running modes that will be explained in the following sections.

3.1.3.1. Transfer Learning (TL)

TL is a traditional supervised learning approach used for model fine-tuning, where the model's probability distribution is shifted to favour the generation of compounds structurally like an input dataset.

The employed technique is teacher-forcing[41] in which the algorithm presents a real molecule from the input dataset to guide the learning process. The neural network weights are modified so during subsequent generations the model is more likely to produce molecules like those from the training set. Since TL is prone to overfitting, a validation set can be provided to monitor validation loss independently from the training loss.

The output of this mode consists of the resulting fine-tuned model, as well as any requested checkpoints saved during the process.

3.1.3.2. Staged Learning (SL)

SL mode is used to fine-tune a model toward the generation of molecules that fulfil a given set of criteria. This process is based on reinforcement learning (RL)[21,42] and curriculum learning (CL)[40]. Under this framework, the software uses two distinct components, a static reference model or Prior and a dynamic model under training or Agent, which is initially cloned from the Prior.

The Agent model undergoes iterative RL cycles. In each epoch, the agent samples a batch of SMILES strings which are then evaluated by a user-defined scoring function. This scoring function can unify criteria such as physicochemical descriptors, toxicity filters, or molecular docking binding affinities, into a single scalar reward using weighted arithmetic or geometric means. Each scoring component produces a raw value that is converted into a normalized value between 0 and 1 through a transformation function such as a sigmoid, reverse sigmoid, or step function. Users assign a weight to each component based on its relative importance. These components are then combined into a single scalar score using either a weighted arithmetic or geometric mean.

This scalar score is combined with the likelihood of the static Prior model to define the augmented likelihood, where the user can control the relative weight given to the reward signal against the regularization imposed by the original Prior. In this case, the Prior will act as a constraint to keep the Agent from staying too far from possible sequences.

The updating of the neural network weights is handled by the DAP (Difference between Augmented and Posterior) strategy[39]. This update is performed by minimizing a loss function based on the difference between the Agent's and Prior's likelihoods. Specifically, the loss for each generated sequence is the squared difference between the augmented likelihood and the Agent's current likelihood. This loss is averaged over the batch and minimized at each epoch using stochastic gradient descent, progressively shifting the Agent's likelihood toward the augmented distribution.

Furthermore, all this architecture operates under a CL strategy. This enables the concatenation of sequential training phases, gradually increasing the complexity of the reward criteria and making it easier for the Agent to adapt to MOO. In practice, this allows the user to provide multiple sequential stages within a single run, each one defining its own scoring profile. A stage is considered complete, and the run proceeds to the next one, once a maximum score threshold or a maximum number of steps has been reached. A checkpoint file is saved after each stage allowing it to be resumed or inspected later.

In addition, two optional mechanisms can be incorporated. First the diversity filter that promotes the generation of structurally diverse molecules by keeping memory of previously generated scaffolds. Once a specific scaffold is sampled a user defined number of times, any subsequent identical structures receive a score of zero, preventing the Agent from collapsing onto a small number of repeated structures. Complementarily, the inception mechanism, maintains a separate memory of the highest scoring molecules generated so far, which users can also define with specific compounds of interest. This reinforces successful solutions and accelerates the convergence towards the desired chemical space.

The output of this mode includes the final trained model, intermediate checkpoints and a detailed CSV file containing the molecules generated throughout the run, along with their respective scores and both Agent and Prior likelihoods.

3.1.3.3 Sampling

In this run mode, no learning or modifications to the model weights are performed. The algorithm

acts strictly as a text generator that evaluates its vocabulary and selects next token based on the model's learned probabilities.

As an output, a CSV file is obtained containing the generated SMILES along with their corresponding NLL values serving as a measure of the model's confidence in generating each sequence.

3.2. Scipion-chem infrastructure

The Scipion framework is built upon an object-oriented Python architecture that ensures data traceability, execution tracking and reproducibility across heterogeneous software. To integrate an external tool such as REINVENT4, any newly developed model must comply with Scipion's native architecture which relies on a plugin-based ecosystem.

Scipion is modular and functions through independent plugins. To extend the framework's capabilities, developers follow a standardized template structure. This allows a straightforward installation of each plugin via a standardized command, provided the user has the base Scipion software installed. Once a new plugin is installed, it is registered ensuring that dependencies, graphical components and execution binaries are bound to the host instance without conflicting other pre-existing tools.

To ensure reproducibility and avoid software conflicts, Scipion enforces environment isolation. Every plugin manages its own isolated environment (typically handled through Conda). Scipion automatically maps the execution paths ensuring no manual user intervention. When a user executes any task, Scipion generates a unique, isolated directory. All intermediate files, generated configurations, and standard logs are written inside this directory, preventing data from overwriting and preserving a historical record of every run within the project.

3.2.1 Protocol class

Rather than forcing users to manually execute disconnected scripts, Scipion-chem is built around the concept of interconnected protocols, each representing a computational task. Any task must be implemented by subclassing the protocol class. This requires a structured template based on three primary methods that define how the protocol operates.

- **_defineParams**: This method is responsible for building the GUI. Input parameters,

numerical thresholds, file paths, or options selectors are defined within this function. To prevent overwhelming the user with technical details, parameters can be categorized into two distinct visibility modes, normal for essential variables required and advanced, which remains hidden by default and exposed when explicitly toggled by the user.

- **_insertAllSteps:** Defines the internal execution pipeline, dividing the entire protocol into discrete, sequential steps. These steps are registered by Scipion, enabling real time progress tracking and recovery if the process is interrupted.

- **_validate:** Automated pre-execution validation layer. It verifies that the provided parameters are within acceptable boundaries, required inputs are present and input files correctly formatted. If any check fails, execution is halted, preventing erroneous runs.

An important part of the execution flow is the final registration of results. At the end of the pipeline defined in _insertAllSteps, the protocol invokes a specific method to register its outputs. This mechanism binds the resulting data objects to the project run, ensuring they are visible and available to other tools within the project. To ensure data compatibility between different software packages, Scipion enforces interoperability through standardized object models.

In addition, Scipion utilizes auxiliary tools called wizards and viewers. A wizard is an interactive helper bound to a specific parameter to handle complex inputs that cannot be easily represented in a flat form. Conversely, viewers provide specialized graphical interfaces for the visualization of the defined output objects.

3.3 REINVENT4 implementation in Scipion

To incorporate REINVENT4 into the Scipion-chem ecosystem, a dedicated plugin was developed. The complete source code is publicly available on GitHub and can be accessed at [https://github.com/izanalcer/scipion-chem-reinvent].

3.3.1 Plugin installation

To prevent version conflicts with host system packages, the plugin automates environment isolation. Utilizing Scipion's binary management system, the installation creates a dedicated Conda environment running Python 3.10. The plugin clones the official REINVENT4 source code repository[43] and triggers its native setup scripts, ensuring critical dependencies like

TOML[44] and RDKit[45] libraries are installed.

To prevent duplicate or corrupted environments, the installation workflow implements persistent status checkpoints. These files act as indicators for Scipion, verifying that each setup stage is completed successfully. Furthermore, since all REINVENT4 operations rely on the mentioned priors, an automated downloader script was created. Instead of requiring users' intervention to download the required files, the plugin communicates directly with the dedicated Zenodo repository, automatically downloading the prior models and placing them within the internal directory of the plugin.

3.3.2 Protocol architecture

To implement all REINVENT4's running modes, three independent protocols were developed.

Non-essential parameters required, such as directory paths or intermediate file naming were automated internally and removed from the GUI. The remaining parameters were segmented using the normal and advanced visibility mode. Once this stage is completed, the plugin will intercept these values, writing them as required in the nested TOML configuration file, finally invoking the command line execution in the background.

While sharing these common features, each protocol was customized to simplify its requirements.

- **ReinventSampling**: It simplifies the user interface by focusing strictly on essential variables, such as the target Prior path and the desired sample size.

- **ReinventTransferLearning**: As an advanced feature, the protocol automatically handles data partitioning, splitting the input molecular dataset into distinct training and validation subsets.

- **ReinventStagedLearning**: As the most complex protocol, it uses the implementation of wizards later described in section 3.3.2.3.

To support this execution architecture, resolve constraints related to SMILES vocabularies and ensure full data traceability, several data structures and auxiliary utilities were implemented as detailed in the following subsections.

3.3.2.1 SMILES preprocessing

REINVENT's Priors support a restricted token vocabulary. They cannot process complex stereochemical notations or uncommon elements that have not been exposed to often enough, leading to runtime errors. To overcome this restriction a preprocessing and validation function was implemented in the protocol before execution.

The script, while going through input sequences, immediately discards those containing forbidden elements unsupported by the Prior. The remaining structures are evaluated using RDKit to verify chemical and valence validity, discarding those invalids. Finally valid structures undergo a transformation into canonized SMILES.

3.3.2.2 Objects

Data interoperability is based on transforming intermediate files into Scipion objects that the entire workflow can understand.

The plugin relies on SetOfSmallMolecules class as its primary data objects, allowing users to import this starting dataset from zero or from other protocols previously executed.

Two new custom data objects were defined by subclassing the EMObject class.

- **ReinventModel:** Created for binary neural network models and checkpoints generated during training.

- **ReinventToml:** Registers TOML configuration files during runs.

At the end of a generative protocol, the output CSV is read and transformed into a SmallMoleculesLibrary, introducing it back to the project workflow and making it compatible with other validation tools within Scipion-chem.

3.3.2.3. Wizards

Due to the complex nested structure demanded in the SL configuration files, presenting all parameter options in a flat GUI form was impractical. To resolve this, a wizard function was developed.

Launched via interactive buttons from within the protocol interface, this wizard allows users to build stage and scoring configurations dynamically. The workflow operates as a reusable template where the user fills out a single set of fields for either a training stage or a scoring component, and upon clicking the wizard button, those values are captured and appended into the nested TOML configuration. This process can be repeated iteratively as many times as needed, allowing the user to configure multiple different steps sequentially creating the desired SL structure.

To provide clear visibility of the process, both a detailed information box and a summary box are included, ensuring the user can track exactly which steps and components have been added already. Additionally, two extra wizards are implemented to delete either entire stages or individual components from the defined workflow by introducing their position number.

3.4 Case of study

To demonstrate the full integration and reproducibility of the developed plugin within Scipion-chem, a VDS pipeline was designed as a validation case study.

3.4.1 SMILES library construction

The target selected for this study was the human adenovirus type 5, utilizing its cryo-EM structure deposited in the Protein Data Bank (PDB)[46] under accession code 6B1T[47]. This structure corresponds to the fibre protein of the viral capsid, which mediates the initial attachment to the host cell. Given that this interaction is the first step of the infection cycle, compounds capable of binding this target could compete with the cell receptor, thereby blocking attachment and preventing the infection.

To target functional regions less susceptible to mutation, a Basic Local Alignment Search Tool (BLAST) was executed against the NCBI database[48]. Subsequently, a Multiple Sequence Alignment (MSA) was performed on these sequences to map and isolate the most conserved structural regions within the viral protein, which were designated as the primary binding pockets for subsequent docking.

Concurrently, an initial library of 1615 ligands was obtained from the ZINC[8] database. To ensure clinical relevance, these compounds were filtered to include only FDA approved drugs,

guaranteeing their safety and viability for human use. Molecular docking simulations were then executed across the previously identified conserved regions using the filtered ligand library. To extract the most promising molecules, a scoring function was implemented. First, compounds demonstrating a docking binding affinity higher than -8 kcal/mol were discarded and the remaining were rescored using the Open Drug Discovery Toolkit (ODDT) to minimize false positives by capturing non-linear interactions that may have been overlooked during the initial docking. Top-ranking compounds were subsequently mapped to their corresponding PubChem[7] identifiers. Only compounds registered in the database were kept resulting in a total of 300 SMILES.

3.4.2. New SMILES generation

The resulting list of SMILES was used in Scipion as a SetOfSmallMolecules, serving as a foundational input for the generative pipeline. The REINVENT4 procedure was performed using the newly developed plugin, sequentially executing three interconnected protocols.

The Reinvent molecule generator was employed in this study. To bias the generation toward the desired chemical scaffolds, a TL protocol was executed to fine-tune the default prior model using the mentioned SMILES dataset. To prevent overfitting, the input was partitioned into training (90%) and validation (10%) sets. Once finished the most optimal checkpoint model was selected based on the lowest validation error. Detailed parameter settings are summarized in Table 1.

Table 1. Parameter configuration for the REINVENT4 transfer learning protocol.

PARAMETER	VALUE	NOTES
Molecule generator	Reinvent	Uses default reinvent.prior
SMILES file	298 SMILES input file	Later divided into 268 for training and 30 for validation
Number of epochs	50	
Save checkpoint	10	A checkpoint model file was saved every 10 epochs
Batch size	64	Number of molecules processed in each epoch.
Number of reference molecules	0	Uses the entire dataset to calculate similarity
Sample batch size	100	Number of generated molecules to compute sample loss

Prior to defining the SL stages, eleven molecular descriptors were calculated using RDKit: molecular weight (MW), Topological Polar Surface Area (TPSA), lipophilicity (SlogP), hydrogen bond donors (HBD), hydrogen bond acceptors (HBA), number of rotatable bonds, total number of aromatic and aliphatic rings, fraction of sp3 carbons (Csp3), quantitative estimate drug-likeness (QED) and synthetic accessibility score (SASCore). The statistical distribution for each descriptor is presented in Table 2.

Table 2. Statistical distribution of molecular descriptors for the reference chemical library. The table presents the mean, 10th percentile (P10), 90th percentile (P90), and standard deviation (Stdev) calculated via RDKit.

DESCRIPTOR	Mean	P10	P90	Stdev
MW (Da)	377.10	237.70	543.50	126.30
TPSA (Å²)	65.60	12.50	120.00	47.80
SlogP	3.07	1.05	5.18	1.74
HBD	1.90	0.00	4.00	1.72
HBA	4.39	2.00	8.00	2.59
NumRotBond	5.30	1.00	9.00	3.06
NumAromaticRings	1.72	1.00	3.00	0.95
NumAliphaticRings	1.70	0.00	4.00	1.39
Csp3	0.44	0.25	0.67	0.17
QED	0.62	0.30	0.86	0.20
SASCore	3.27	2.21	4.74	0.97

The best performing TL model was then used in a SL protocol to restrict molecule generation to a specific range of physicochemical characteristics, consistent with the resulting profile of the input dataset. A three-stage protocol was designed: an initial stage applied permissive constraints to filter out impossible molecules and establish a basic drug-likeness threshold, a second stage to introduce stricter bioavailability criteria and a final stage to replicate the structural profile mentioned before. General configuration is summarized in Table 3, and parameters defined for each individual stage are detailed in Table 4.

Table 3. Parameter configuration for the REINVENT4 staged learning protocol.

PARAMETER	VALUE	NOTE
Molecule generator	Reinvent	
Prior model file	best model resulting from TL protocol	
Purge memories	No	Model maintains memory across stages ensuring it does not repeat previously found structures.
Learning strategy	DAP	Parameters selected to conservatively preserve chemical knowledge acquired during TL.
Sigma of the reward function	128	
Learning rate	0.0001	
Diversity Filter	IdenticalMurckoScaffold	Promotes scaffold diversity. Assigns a score penalty to any molecule whose scaffold has been conserved in more than 25 high scoring molecules, limiting repetitive generation of similar compounds.
Bucket size	25	
Minimum score	0.4	
Scoring function	Geometric mean	Selected for all stages. Penalizes multiplicatively any component with a low score enforcing satisfaction of all objectives.

Table 4. Stage and scoring components configurations for the staged learning protocol. This table includes descriptors ranges, weight of each component and execution criteria defined for each stage. (-) indicates components that were inactive during that specific stage.

SCORING COMPONENT	STAGE 1		STAGE 2		STAGE 3	
	Range	Weight	Range	Weight	Range	Weight
QED	0.30–0.85	0.40	0.30–0.85	0.20	≥ 0.50	0.10
MW	150–550	0.30	220–520	0.20	250–480	0.15
SASCore	≤ 5.5	0.30	≤ 4.5	0.10	≤ 4.5	0.10
TPSA	—	—	10–120	0.15	—	—
SlogP	—	—	0.5–5.5	0.15	1.0–5.0	0.10
HBD	—	—	0–5	0.10	—	—
HBA	—	—	1–8	0.10	—	—
NumRotBond	—	—	0–10	0.10	—	—
Csp3	—	—	—	—	0.25–0.67	0.25
NumAromaticRings	—	—	—	—	1–3	0.15
NumAliphaticRings	—	—	—	—	1–4	0.15
Termination score	0.65		0.72		0.78	
Max steps	400		600		1000	
Min steps	100		150		200	

Finally, a sampling protocol was executed using the optimized model, generating a final dataset of 300 SMILES.

3.4.3 Final Analysis

To evaluate the performance of the generated molecules, three different subsets of compounds were established for a comparative analysis.

A. **Reference:** Original SMILES used for model training.
B. **Generated:** New sampled SMILES obtained from REINVENT4.
C. **Random:** Random selection of FDA-approved compounds.

To ensure methodological consistency and validate the compatibility of the newly developed plugin, the entire VDS workflow was executed within the Scipion framework. All three subsets went through an identical pipeline against the same agent.

Ligands were prepared using RDKit ligand preparation protocol and concurrently, the structure of the adenovirus protein (6B1T) was obtained and processed with the target preparation protocol. Active sites were subsequently identified using the P2Rank protocol. Following, a filtering process based on the highest prediction scores, selected the top 25 pockets for further analysis. Finally, an AutoDock docking protocol was executed evaluating the three compound datasets in one docking position across the 25 binding pockets.

4. Results

4.1. Plugin implementation

A dedicated plugin was successfully developed and integrated into the Scipion-chem framework, incorporating the full REINVENT4 architecture. The plugin adheres to Scipion's native ecosystem standards, ensuring full compatibility with pre-existing protocols and data objects within the platform.

4.1.1. Installation

Plugin installation was completed through a single standardized command, requiring no additional manual intervention from the user. The installation pipeline automatically created the necessary directory structure and a dedicated Conda environment, isolating all dependencies to prevent conflicts with the host system. The REINVENT4 source code was successfully cloned from its official repository, and all required libraries, including RDKit and TOML, were installed within the isolated environment. Prior model files were automatically retrieved from the designated Zenodo repository[49] and placed in the appropriate internal directories. Persistent checkpoint files were generated at each stage of the installation, confirming successful completion of every setup step. All installation events were recorded in log files, ensuring traceability of the setup process.

4.1.2. Protocols

Three independent protocols were implemented and registered within the Scipion-chem interface. Together, these protocols cover the full REINVENT4 operational workflow and can be interconnected sequentially, with the output of one protocol serving directly as the input of the next as shown in Figure 1.

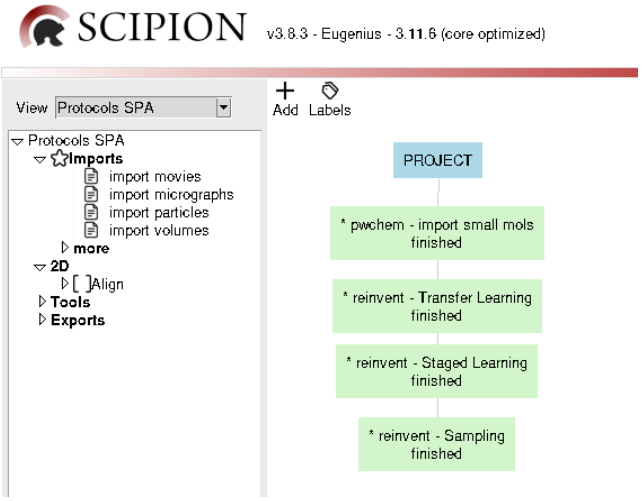

Figure 1. Integration of REINVENT4 protocols in Scipion. The Scipion interface displays the pwchem import protocol integrated with the three core protocols implemented in this project: Transfer Learning, Staged Learning, and Sampling.

Parameter classification into standard and advanced visibility levels was validated across all three protocols, with default values successfully predefined. The comparison between two modes is shown in Figure 2. Users were only required to supply situational parameters with no additional configuration needed. All parameters were confirmed to display integrated help text upon request. An example of displayed help is shown in Figure 3. Input and output file handling was verified to operate through Scipion's objects, with the correct object types automatically recognized and propagated between protocols.

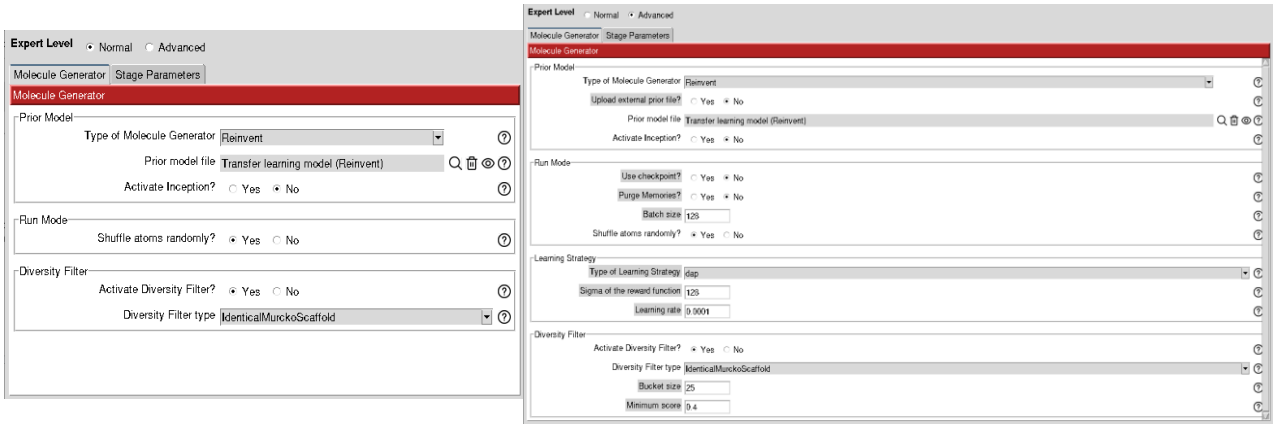

Figure 2. Protocol interface comparison across expert levels. The Staged Learning interface is shown under different user modes: (A) standard parameters displayed under the normal expert level, and (B) additional configuration options visible in the advanced expert level.

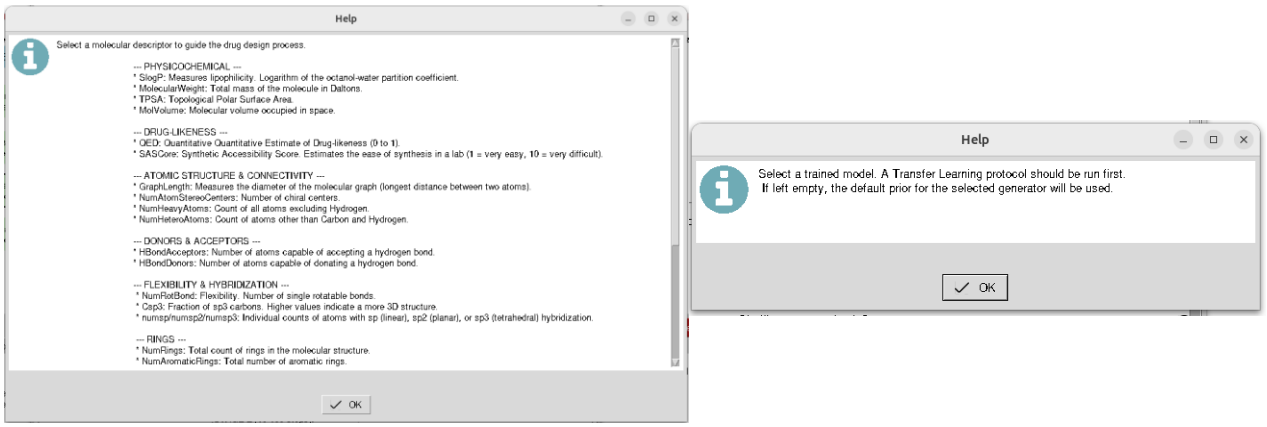

Figure 3. Contextual help text documentation within Scipion. Examples of the integrated Scipion help system within the Staged Learning protocol interface, illustrating documentation for: (A) the select scoring component parameter, and (B) the prior model file parameter.

The validation layer successfully intercepted configuration errors across all tested scenarios. Attempts to initiate runs with missing inputs or out of range parameter values were blocked prior to execution, with descriptive error messages displayed to the user in all cases. This mechanism guides users through the configuration process, preventing execution failures and providing clear error explanations with examples shown in Figure 4.

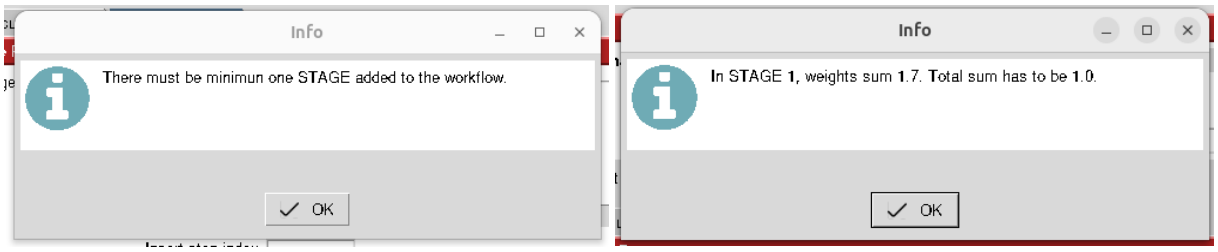

Figure 4. Validation error messages in Scipion. Interface alerts triggered during invalid Staged Learning executions: (A) error generated when no stages are defined in the pipeline, and (B) error displayed when the cumulative weight of scoring components in a single stage exceeds the maximum allowed limit.

The generation and management of REINVENT4's required configuration TOML files is handled entirely internally. Users are not required to create, edit or interact with these files at any point during execution. The configuration files are assembled automatically from the GUI inputs and executed in the background. In Figure 5 a resulting TOML from an SL run is shown.

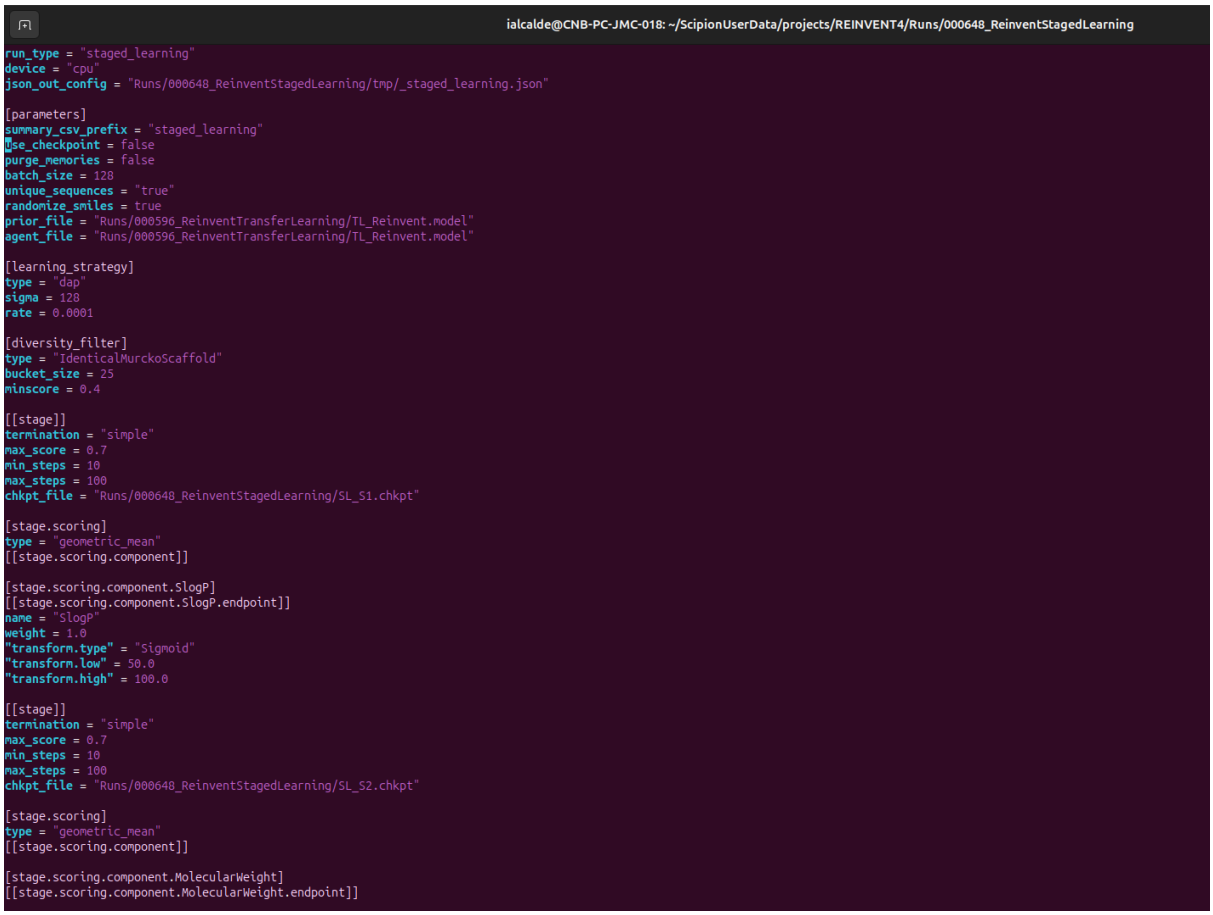

Figure 5. Automatically generated TOML configuration file. The resulting backend TOML file produced upon the successful execution of the Staged Learning protocol.

The SL wizard was verified to correctly capture stage and scoring component definitions to the TOML configuration file. The information and summary panels accurately reflected the configuration and deletion wizards successfully removed stages and components by position indexes.

The preprocessing function performed without errors across all tested inputs. Invalid SMILES containing unsupported notations or unrecognized atomic symbols were successfully identified and discarded. The remaining structures were confirmed to be converted to canonical SMILES format prior to being passed to the Prior model.

All three protocols completed their respective runs without requiring any manual command line interaction or file modification. Real time execution logs were accessible throughout all runs via Scipion's integrated output console. An example of the output log of a run is shown in Figure 6.

Figure 6. Real-time standard output log execution in Scipion. Snippet of the Scipion standard output log (run.stdout) displaying execution details, steps, and real-time progress during an active Transfer Learning run.

4.2. Case of study

4.2.1. Generative pipeline

For the initial (TL) run, the reference dataset consisted of 298 valid SMILES that were partitioned into training (90%, 268 SMILES) and validation (10%, 30 SMILES) sets. The training was completed in approximately 13 minutes over 50 epochs, with the model achieving its optimal validation loss of 34.572 at epoch 20. Beyond this checkpoint, further training led to overfitting, therefore, the model output from epoch 20 was selected for the subsequent SL protocol.

The staged learning protocol was executed over approximately 2 hours. In all three stages the run terminated before reaching the maximum number of steps defined in the configuration, indicating that the termination scores were reached before. The reported score (average of individual component scoring) increased within each stage. The clearest learning behaviour was observed in Stage 2, where the average score increased nearly five points. Concurrently, the NLL of the agent increased with respect to the prior, indicating that the model specialized toward a different space region. Besides, this divergence led to a progressive decrease in the percentage of valid SMILES, dropping from an average of 98.3% to 93.9% with individual steps obtaining as low as 87%.

Finally, the sampling protocol generated 300 molecules, from which 16 smiles were identified as invalid and subsequently removed, resulting in a final set of 284 novel compounds.

4.2.2. Docking results

The docking results obtained for the three molecule sets (Reference, Generated and Random)

were first filtered to remove outliers and invalid docking energy values. To provide an overall comparison of docking performance across the three sets, two metrics were calculated relative to the total number of valid inputs, the percentage of molecules that yielded at least one valid docking pose in any of the 25 pockets screened, and the percentage of hit molecules, defined as those achieving a docking energy ≤ -8 kcal/mol. Figure 7 summarises these results for each dataset.

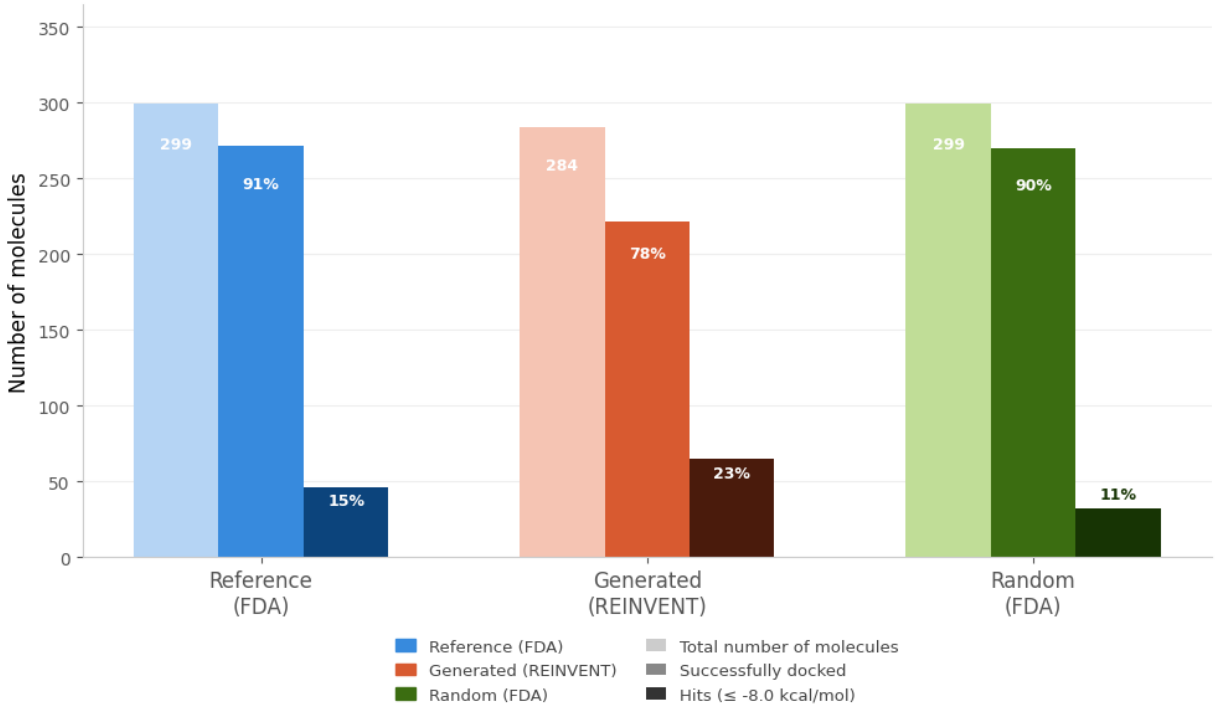

Figure 7. Docking success rates across molecule sets. Comparison of docking outcomes for the three molecule sets screened against 25 binding pockets of 6B1T: Reference set used for training, REINVENT4 generated set and a randomly selected control. For each set, the light coloured bar indicates the total number of valid molecules used as input for docking, the intermediate coloured bar indicates the number and percentage of molecules that achieved at least one successful docking pose, and the dark bar indicates the number and percentage of hit molecules (docking energy ≤ -8.0 kcal/mol).

The reference and random FDA set showed comparative docking success rates (91% and 90% respectively), while the generated set showed a lower docking success rate (78%) but the highest hit rate (23%) among the three groups.

For a clearer view of the docking results, the distribution of binding energies was visualized for each molecule set using violin plots as shown in Figure 8.

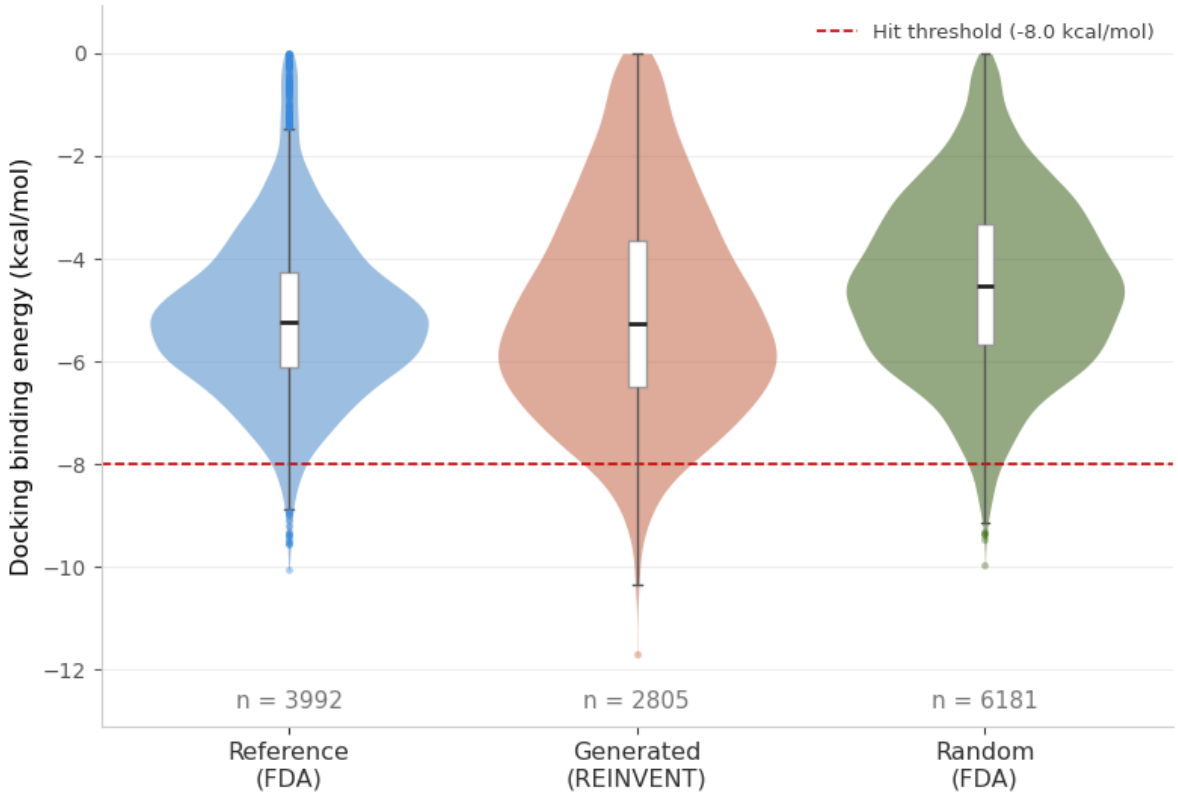

Figure 8. Distribution of docking binding energies across molecule sets. Violin plots showing the distribution of docking binding energies (kcal/mol) for the Reference, Generated, and Random molecule sets across 25 binding pockets of 6B1T. The embedded box plot indicates the median (black line) and interquartile range (white box), with whiskers extending to 1.5 × IQR. Individual points beyond this range are shown as outliers. The dashed red line marks the hit threshold (-8.0 kcal/mol). n indicates the number of valid docking poses included in each distribution.

The three distributions showed similar median binding energies, with the random set showing the lowest. However, the REINVENT generated set displayed a wider interquartile range and a longer lower tail, extending to -11.7 kcal/mol. In addition, the generated set showed a broader spread of values below the hit threshold relative to the other two sets.

The mean binding energy and the number of hit molecules per pocket were analysed to compare differences between the three molecule sets, as shown in Figure 9 and Figure 10 respectively.

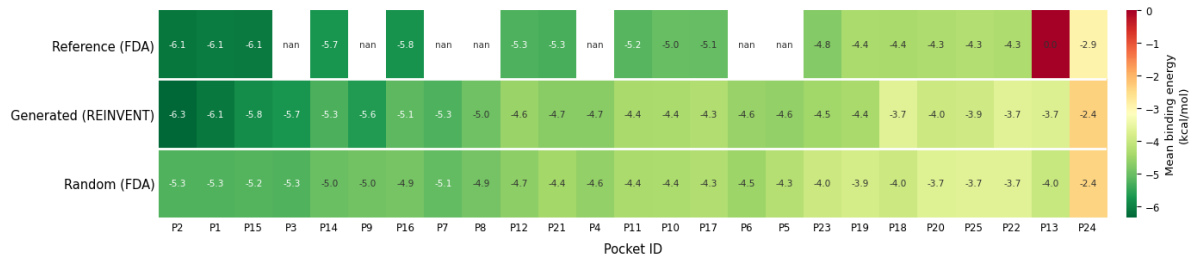

Figure 9. Mean docking binding energy per pocket across molecule sets. Heatmap showing the mean binding energy obtained for each of the 25 binding pockets of 6B1T, for the Reference, Generated, and Random molecule sets. Pockets are ordered from highest to lowest mean binding energy based on the Reference set. Darker green indicates more favourable (more negative) binding energies. nan indicates pockets for which no valid docking poses were obtained for that set.

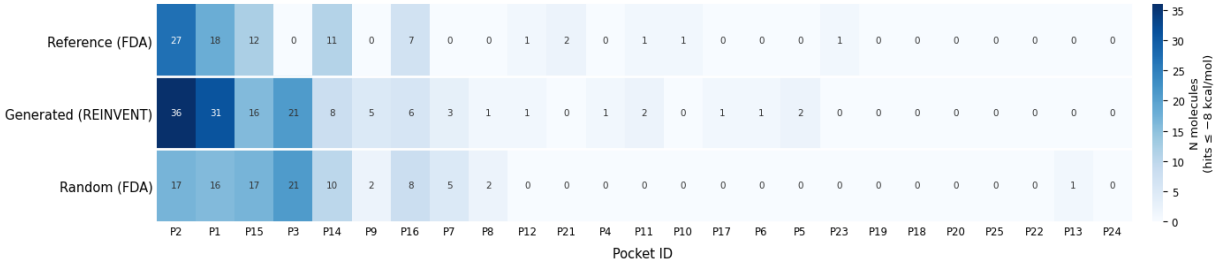

Figure 10. Number of hit molecules per pocket across molecule sets. Heatmap showing the number of hit molecules (docking energy ≤ -8.0 kcal/mol) obtained for each of the 25 binding pockets of 6B1T, for the Reference, Generated, and Random molecule sets. Pockets are ordered from highest to lowest number of hits.

Across all three sets, the pockets with the most favourable mean binding energies (P2, P1, P15, P14, P16) were also those concentrating the largest numbers of hit molecules, indicating consistency. The Generated set showed mean binding energies comparable to, and in several pockets matching, those of the Reference set, both of which were more favourable than those of the Random set. Consistently, the Random set also showed fewer hit molecules than the Reference and Generated sets across the top ranked pockets. In terms of hit counts, the Generated set exceeded the Reference set in the top-ranked pockets (e.g., P2: 36 vs. 27 hits, P1: 31 vs. 18 hits).

Overall, the docking analysis indicates that, despite a lower docking success rate, the Generated set produced a higher proportion and number of hits than the Reference and Random sets, particularly in the most relevant pockets.

Finally, Table 5 lists the five molecules with the best docking energies from the Reference and Generated sets, with their two-dimensional structures included in Figure 11. To identify shared structural motifs, a Maximum Common Substructure (MCS) analysis was performed across all pairs using RDKit. The most relevant results are included in Table 6.

Table 5. Docking energies of the top ranked molecules in Reference and Generated sets. The pocket in which the best energy was found is also included.

SET	PUBCHEM CID / MOLECULE ID	POCKET	ENERGY (kcal/mol)
Reference	CID:60795	1	-10.06
	CID:146156652	1	-9.55
	CID:73759769	2	-9.52
Generated	MOL_173	1	-11.70
	MOL_175	2	-10.34
	MOL_015	2	-10.32

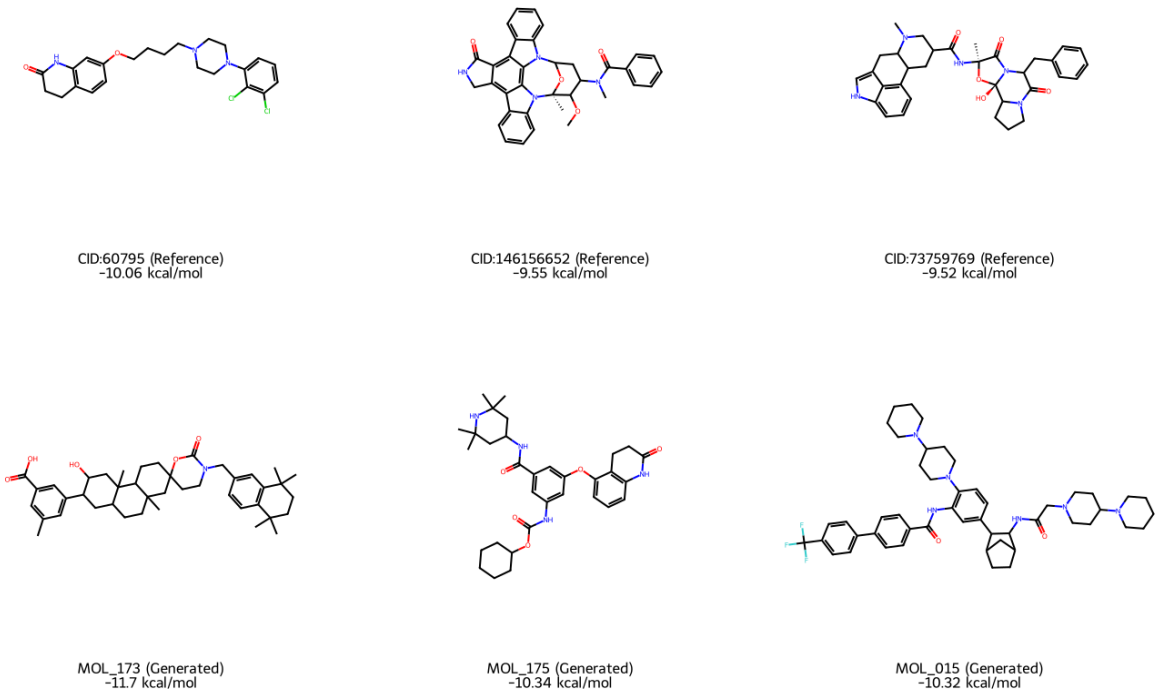

Figure 11. Two-dimensional structures of the top ranked Reference and Generated molecules. Representations were obtained with RDKit. Labels the molecule ID, the corresponding set, and the docking energy.

Table 6. Maximum Common Substructure analysis results. The analysis was performed between top ranked molecules in Reference and Generated datasets. Shared atoms, bonds and common substructure were obtained using RDKit's MCS analysis.

REFERENCE MOLECULE	GENERATED MOLECULE	SHARED ATOMS	SHARED BONDS	COMMON SMILES
CID:60795	MOL_173	7	7	OC1:C:C:C:C:C:1
CID:60795	MOL_175	11	12	O=C1CCC2:C:C:C:C:C:2N1
CID:60795	MOL_015	7	7	NC1:C:C:C:C:C:1
CID:146156652	MOL_173	8	8	O=CC1:C:C:C:C:C:1
CID:146156652	MOL_175	10	10	CNC(=O)C1:C:C:C:C:C:1
CID:146156652	MOL_015	10	10	CNC(=O)C1:C:C:C:C:C:1
CID:73759769	MOL_173	10	11	C1:C2:C:C:C:C:C:2CCC1
CID:73759769	MOL_175	10	11	C1:C:C2:C(:C:C:1)CCCN2
CID:73759769	MOL_015	7	7	CC1:C:C:C:C:C:1

Consistent with previously identified binding pockets, 1 and 2 retained the majority of the best scoring poses across both datasets. However, the found structural matches correspond to small and chemical unspecific fragments such as isolated benzenes or benzamide rings involving 7 to 10 shared atoms.

5. Discussion

The fragmentation of computational tools across the VDS workflow has historically represented one of the primary barriers to its adoption by non-computational researchers. As outlined in the introduction, REINVENT4 specifically concentrates several of these barriers: a complex manual installation, exclusive CLI execution, complex configuration files, limited data interoperability, and poor traceability. Beyond REINVENT4, the absence of any de novo molecular generation capability within Scipion-chem represented a fundamental gap in its VDS pipeline. This work addressed both problems simultaneously through the development of a dedicated plugin. No equivalent integration of a generative AI tool within a unified VDS platform existed prior to this work, which further highlights the relevance of the contribution presented here.

5.1. Resolution of technical barriers

The collective effect of the implemented simplifications represents a qualitative shift in how REINVENT4 can be used in practice. Individually, eliminating manual TOML creation, automating installation, or exposing parameters through a structured GUI are incremental improvements. Together, however, they remove the threshold that has historically restricted generative molecular design to non-computational researchers.

This is better illustrated by the reduction in the number of files and manual interactions required to complete an equivalent run in both environments. What natively requires navigating a CLI, managing directory structures, and manually converting outputs, is reduced within the plugin to supplying a SMILES dataset and selecting parameters through a graphical form. The learning curve associated with the tool itself is largely reduced, allowing the researcher to focus on biological and chemical decisions rather than the operational ones. It is worth noting that visual workflow platforms such as KNIME[50] have similarly simplified specific stages of the classical VDS pipeline, including library preparation and screening, however, none of these platforms currently incorporate de novo generative tools.

The error handling design reinforces this shift. Rather than encountering silent failures or cryptic runtime exceptions, users receive feedback before execution begins, a contrast that becomes relevant given the computational cost of generative runs, where a misconfigured job can waste hours of processing time.

Summarising the developed plugin successfully addressed all five technical barriers identified.

Taken together, these properties suggest that its primary contribution is mainly translational, making a powerful tool accessible to its primary target audience.

5.2. Completion of the VDS pipeline

Beyond resolving REINVENT4's specific limitations, the integration of this plugin addresses a broader strategic gap. Prior to this work, Scipion-chem covered the traditional VDS workflow, including molecule importing, structural preparation, docking, and dynamics, but lacked any de novo generation capability. The addition of this plugin completes the pipeline, from generative design through structural validation within a single environment.

5.3. Case of study analysis

The complete pipeline was successfully executed within Scipion, verifying the interoperability and compatibility of the three newly developed protocols with the existing software in Scipion. Additionally, compared to classical and non-automated experimental screening methods, which would require years to evaluate a comparable number of compounds, the complete pipeline was executed in a matter of days for the three datasets, reinforcing the time and cost advantages of computational VDS approaches.

The results obtained during the learning stages of the generative pipeline show that learning was successful, with the model progressively converging towards the desired chemical space. The early termination observed in all three stages further suggests that the scoring thresholds were reached efficiently. It is worth highlighting the substantial increase in score obtained during Stage 2, which proved to be the most decisive stage, as it captured most of the characteristics defined for the target chemical space.

The increase in the agent's NLL relative to the prior was expected, as was the decline in the percentage of valid SMILES. This behaviour is consistent with the fact that the more restrictive the desired characteristics, the more difficult it becomes to generate molecules that satisfy them, resulting in a higher proportion of invalid structures. This represents an acceptable cost given the specificity of the generated molecules as only 16 out of 300 sampled SMILES were invalid, and this loss in validity is compensated with the docking outcomes discussed below.

Regarding the docking results, a lower success rate was observed for the Generated set (78%) compared to the Reference and Random sets (91% and 90%, respectively), suggesting that

REINVENT4 produced structurally novel or complex molecules that could not be processed by AutoDock. However, this reduction in docking success was compensated by the highest hit rate among the three sets (23%), indicating that the molecules that did successfully dock did it with higher affinity. This confirms that the model was biased towards generating molecules similar to those in the initial reference set, while also being able to improve them by exploring new regions of that chemical space.

The distribution of docking binding energies supports this interpretation, showing a clear difference in the number of molecules below the hit threshold, even increasing that of the reference compounds. It is also worth noting that the wider interquartile range and the longer tails observed in the Generated set, point to greater structural diversity among the sampled molecules. This is consistent with the diversity filter applied during the staged learning stages, which discouraged repeated scaffolds and thereby promoted the exploration of novel compounds.

Finally, the pocket analysis aligns with these interpretations. The Generated set exceeded the Reference set in hit counts within the same pockets, rather than showing improvements distributed indiscriminately across all of them. This pattern indicates that the generated molecules were not simply improved in a general sense but specifically improved relative to the characteristics of the initial reference set, with molecule generation effectively guided towards the same structurally relevant regions. Notably, this convergence does not appear at the level of molecular scaffolds, as the structural comparison between both sets did not result relevant results. This suggests that the staged learning process successfully directed molecule generation towards relevant regions of the target through diverse chemical strategies, rather than simply producing molecules by replicating the reference scaffolds.

Overall, the case of study confirms that REINVENT4 can be effectively integrated within Scipion-chem to support generative VDS, with the Generated set showing a favourable balance between structural novelty and binding performance, achieving higher hit rates and stronger affinities than both the reference and random controls.

5.3.1. Limitations of obtained results.

Despite these promising results, several limitations should be considered. First, the loss of a substantial number of compounds due to SMILES invalidity should not be unnoticed. While acceptable in this case, this limitation becomes more critical if a more computationally expensive analysis, such as molecular dynamics, were performed, as the time lost on processing invalid

molecules would become considerably more costly.

Second, the results presented here remain computational predictions, as no in vitro validation was performed. Experimental confirmation of the observed binding trends should therefore be a priority in future work. Beyond binding affinity, the synthetic accessibility of the generated molecules should also be assessed, since structurally novel compounds may be difficult or even unfeasible to synthesise in practice, regardless of their predicted binding performance. Additionally, no toxicity or ADMET filtering was applied to the successful candidates, which represents a relevant step to be addressed in further studies before prioritising them for experimental testing.

Furthermore, the generative results are inherently constrained by the composition of the initial reference dataset. As the scoring functions and physicochemical thresholds used during staged learning were derived from the reference molecules, the model's exploration of chemical space was implicitly biased towards properties already represented in this set, rather than constituting an unbiased sampling of chemical diversity.

5.4. Future directions.

5.4.1. Expanding REINVENT4's plugin features.

While the plugin successfully integrates REINVENT4's core generative mode, the broader REINVENT ecosystem offers additional running modes that remain outside the current implementation. As future work, the three other alternative generative modes (LibInvent, LinkInvent and Mol2Mol) within the REINVENT ecosystem could be implemented. Given the complexity of the required input structures and constraints, Scipion currently lacks a specific protocol capable of parsing them, requiring the development of new protocols for this specialized import. Although the underlying architecture, execution modes, and required parameters for these running modes are already implemented within the plugin, they will remain hidden until the input interface is fully developed. This represents a clear pathway for future work to expand the tool's capabilities.

Additionally, all current protocols were executed locally using CPU power. A valuable future improvement would be implementing GPU support to significantly accelerate execution times. This enhanced performance would allow users to introduce stricter model constraints, run more training epochs, and generate substantially larger volumes of output SMILES.

5.4.2. Alternatives to REINVENT

Although this work focuses on implementing REINVENT as a solution for de novo molecular generation, alternative tools could also be integrated in the future. As a direction for future development, new software within this domain should be considered, for example MolSnapper[51] and DrugEX[52]. While REINVENT4 stands out as a robust framework optimized for 1D SMILES, these other tools could offer other possibilities.

MolSnapper provides significant advantages for structure-based drug design, by operating directly with 3D structures. This capability enables the generation of molecules explicitly tailored to fit a specific protein binding site, ensuring the reproduction of biophysical constraints directly in the generated pose without requiring an additional redocking step. Furthermore, MolSnapper incorporates specific spatial constraints to prevent steric clashes with the target protein pocket, ultimately yielding approximately twice as many valid molecular structures compared to alternative methods[51].

On the other hand, while the two previously mentioned tools work with SMILES strings, DrugEX introduces graph representations. This approach ensures that the organization of molecular fragments remains invariant, guaranteeing an absolute validation rate.

Overall, this future integration should not only focus on new independent software but also on implementing consensus tools featured in other stages of the workflow, allowing users to easily compare results across different methods.

6. Conclusions

This work has successfully achieved the objectives initially proposed. In accordance with those objectives, the following conclusions are presented:

1. Three different protocols were designed and implemented within the Scipion-chem environment covering the full complexity of REINVENT4 behind a structured GUI. The manual configuration of TOML files, CLI execution, and dependency management were fully automated, making generative molecular design accessible to researchers without computational expertise.

2. Full compatibility between the newly developed plugin and the pre-existing chemoinformatic tools within Scipion-chem was ensured. The plugin integrates seamlessly into the platform's modular architecture, enabling the direct connection of generated outputs with structural validation within a single unified workflow.

3. The developed tool was validated through a real practical case study against a therapeutic target, demonstrating the viability, efficacy and robustness of the integrated system. The generated molecules were successfully processed through the VDS pipeline, confirming the functional integrity of the plugin under realistic research conditions.

Overall, this work presents the first integration of a de novo molecule generative tool within the Scipion-chem platform, effectively filling a significant gap in its VDS pipeline. By uniting de novo molecular generation, structural preparation, docking, and dynamics under a single reproducible environment, the plugin transforms what was previously a fragmented, expert process into an accessible and traceable workflow. Ultimately, this integration reduces one of the biggest limiting factors in modern therapeutic research, ensuring that computational expertise is no longer a prerequisite for utilizing its most advanced methodologies.

Future efforts may focus on the integration of additional REINVENT4 running modes, GPU acceleration and the incorporation of alternative generative tools. Taken together, these future developments would establish Scipion-chem as a complete and comprehensive platform for computational drug discovery, where the full cycle from generative molecular design to structural validation can be conducted within a single, unified and reproducible environment.

ANNEX 1: Declaration of use of Artificial Intelligence

Artificial intelligence tools were used during the development of this work. Their role was strictly supportive, specifically assisting in code development fixing minor errors. Additionally, AI was employed to translate selected sections of the text, refine the language, and ensure correct grammar and an appropriate academic writing style.

Crucially, no figures, graphs, or experimental results presented in this work were generated by AI, nor were any AI tools involved in formulating conclusions or performing tasks of an intellectual or scientific nature. The integration of AI was purely instrumental and did not interfere with independent research.

All content generated, modified, or produced with the assistance of these tools was carefully reviewed, validated, and edited by the author, who remains fully responsible for the accuracy, academic integrity, and final content of this thesis.

7. Bibliography

1. Khushal Khambhati, Deepak Siruka, Suresh Ramakrishna, Vijai Singh. Current progress in high-throughput screening for drug repurposing. Prog Mol Biol Transl Sci. 2024 Jan 1;205:247–57. doi:10.1016/bs.pmbts.2024.03.013

2. Oke A, Sahin D, Chen X, Shang Y. High Throughput Screening for Drug Discovery and Virus Detection. Comb Chem High Throughput Screen. 2022;25(9):1518–33. doi:10.2174/1386207324666210811124856 PubMed PMID: 34382507.

3. Leelananda SP, Lindert S. Computational methods in drug discovery. Beilstein J Org Chem. 2016 Dec 12;12(1):2694–718. doi:10.3762/bjoc.12.267

4. Lin X, Li X, Lin X. A Review on Applications of Computational Methods in Drug Screening and Design. Molecules. 2020 Jan;25(6):1375. doi:10.3390/molecules25061375

5. Mullard A. The drug-maker's guide to the galaxy. Nature. 2017. doi:10.1038/549445a

6. Gaulton A, Bellis LJ, Bento AP, Chambers J, Davies M, Hersey A, et al. ChEMBL: a large-scale bioactivity database for drug discovery. Nucleic Acids Res. 2012 Jan 1;40(Database issue):D1100-7. doi:10.1093/nar/gkr777 PubMed PMID: 21948594; PubMed Central PMCID: PMC3245175.

7. Kim S, Thiessen PA, Bolton EE, Chen J, Fu G, Gindulyte A, et al. PubChem Substance and Compound databases. Nucleic Acids Res. 2016 Jan 4;44(D1):D1202–13. doi:10.1093/nar/gkv951

8. Irwin JJ, Shoichet BK. ZINC − A Free Database of Commercially Available Compounds for Virtual Screening. J Chem Inf Model. 2005 Jan 1;45(1):177–82. doi:10.1021/ci049714+

9. Lipinski C, Lombardo F, Dominy BW, Feeney PJ. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. Adv Drug Deliv Rev. 1997 Jan 1;3:23–5.

10. Reymond JL. The Chemical Space Project. Acc Chem Res. 2015 Mar 17;48(3):722–30. doi:10.1021/ar500432k

11. Gorgulla C, Boeszoermenyi A, Wang ZF, Fischer PD, Coote P, Das KMP, et al. An open-source drug discovery platform enables ultra-large virtual screens. Nature. 2020 Apr;580(7805):663–8. doi:10.1038/s41586-020-2117-z PubMed PMID: 32152607; PubMed Central PMCID: PMC8352709.

12. Gao W, Coley CW. The Synthesizability of Molecules Proposed by Generative Models [Internet]. 2020 Feb 17. doi:10.48550/arXiv.2002.07007

13. Hoffmann T, Gastreich M. The next level in chemical space navigation: going far beyond enumerable compound libraries. Drug Discov Today. 2019 May;24(5):1148–56. doi:10.1016/j.drudis.2019.02.013 PubMed PMID: 30851414.

14. Schneider G. Mind and machine in drug design. Nat Mach Intell. 2019. doi:10.1038/s42256-019-0030-7

15. Vamathevan J, Clark D, Czodrowski P, Dunham I, Ferran E, Lee G, et al. Applications of machine learning in drug discovery and development. Nat Rev Drug Discov. 2019

Jun;18(6):463–77. doi:10.1038/s41573-019-0024-5 PubMed PMID: 30976107; PubMed Central PMCID: PMC6552674.

16. Weininger D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci. 1988 Feb 1;28(1):31–6. doi:10.1021/ci00057a005

17. Gómez-Bombarelli R, Wei JN, Duvenaud D, Hernández-Lobato JM, Sánchez-Lengeling B, Sheberla D, et al. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. ACS Cent Sci. 2018 Feb 28;4(2):268–76. doi:10.1021/acscentsci.7b00572

18. Schwaller P, Hoover B, Reymond JL, Strobelt H, Laino T. Extraction of organic chemistry grammar from unsupervised learning of chemical reactions. Sci Adv. 2021 Apr 7;7(15):eabe4166. doi:10.1126/sciadv.abe4166 PubMed PMID: 33827815; PubMed Central PMCID: PMC8026122.

19. Segler MHS, Kogej T, Tyrchan C, Waller MP. Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks. ACS Cent Sci. 2018 Jan 24;4(1):120–31. doi:10.1021/acscentsci.7b00512

20. Olivecrona M, Blaschke T, Engkvist O, Chen H. Molecular de-novo design through deep reinforcement learning. J Cheminformatics. 2017 Sep 4;9(1):48. doi:10.1186/s13321-017-0235-x

21. Popova M, Isayev O, Tropsha A. Deep reinforcement learning for de novo drug design. Sci Adv. 2018 Jul;4(7):eaap7885. doi:10.1126/sciadv.aap7885 PubMed PMID: 30050984; PubMed Central PMCID: PMC6059760.

22. Nicolaou CA, Brown N. Multi-objective optimization methods in drug design. Drug Discov Today Technol. 2013 Sep 1;10(3):e427–35. doi:10.1016/j.ddtec.2013.02.001

23. Nguyen HM, Vu NH, Lam HT, Nguyen HD. Pareto-Guided Reinforcement Learning for Multi-Objective ADMET Optimization in Generative Drug Design. 2025 Dec 2.

24. Liu Y, Zhu Y, Wang J, Hu R, Shen C, Qu W, et al. A Multi-Objective Molecular Generation Method Based on Pareto Algorithm and Monte Carlo Tree Search. Adv Sci. 2025 Apr 4;12(20):2410640. doi:10.1002/advs.202410640 PubMed PMID: 40183315; PubMed Central PMCID: PMC12120794.

25. Blaschke T, Arús-Pous J, Chen H, Margreitter C, Tyrchan C, Engkvist O, et al. REINVENT 2.0: An AI Tool for De Novo Drug Design. J Chem Inf Model. 2020 Dec 28;60(12):5918–22. doi:10.1021/acs.jcim.0c00915

26. Loeffler HH, He J, Tibo A, Janet JP, Voronov A, Mervin LH, et al. Reinvent 4: Modern AI–driven generative molecule design. J Cheminformatics. 2024 Feb 21;16(1):20. doi:10.1186/s13321-024-00812-5

27. Conesa P, Fonseca YC, Jiménez de la Morena J, Sharov G, de la Rosa-Trevín JM, Cuervo A, et al. Scipion3: A workflow engine for cryo-electron microscopy image processing and structural biology. Biol Imaging. 2023;3:e13. doi:10.1017/S2633903X23000132 PubMed PMID: 38510163; PubMed Central PMCID: PMC10951921.

28. de la Rosa-Trevín JM, Quintana A, Del Cano L, Zaldívar A, Foche I, Gutiérrez J, et al. Scipion: A software framework toward integration, reproducibility and validation in 3D electron microscopy. J Struct Biol. 2016 Jul;195(1):93–9. doi:10.1016/j.jsb.2016.04.010

PubMed PMID: 27108186.

29. Krieger JM, Sorzano COS, Carazo JM. Scipion-EM-ProDy: A Graphical Interface for the ProDy Python Package within the Scipion Workflow Engine Enabling Integration of Databases, Simulations and Cryo-Electron Microscopy Image Processing. Int J Mol Sci. 2023 Jan;24(18):14245. doi:10.3390/ijms241814245

30. Del Hoyo D, Salinas M, Lomas A, Ulzurrun E, Campillo NE, Sorzano CO. Scipion-Chem: An Open Platform for Virtual Drug Screening. J Chem Inf Model. 2023 Dec 25;63(24):7873–85. doi:10.1021/acs.jcim.3c01085

31. Repasky MP, Shelley M, Friesner RA. Flexible Ligand Docking with Glide. Curr Protoc Bioinforma. 2007;18(1):8.12.1-8.12.36. doi:10.1002/0471250953.bi0812s18

32. Forli S, Huey R, Pique ME, Sanner MF, Goodsell DS, Olson AJ. Computational protein–ligand docking and virtual drug screening with the AutoDock suite. Nat Protoc. 2016 May;11(5):905–19. doi:10.1038/nprot.2016.051

33. Liu N, Xu Z. Using LeDock as a docking tool for computational drug design. IOP Conf Ser Earth Environ Sci. 2019 Feb 23;218:012143. doi:10.1088/1755-1315/218/1/012143

34. Le Guilloux V, Schmidtke P, Tuffery P. Fpocket: An open source platform for ligand pocket detection. BMC Bioinformatics. 2009 Jun 2;10(1):168. doi:10.1186/1471-2105-10-168

35. Gupta A, Müller AT, Huisman BJH, Fuchs JA, Schneider P, Schneider G. Generative Recurrent Networks for De Novo Drug Design. Mol Inform. 2018 Jan;37(1–2):1700111. doi:10.1002/minf.201700111 PubMed PMID: 29095571; PubMed Central PMCID: PMC5836943.

36. Perron Q, Mirguet O, Tajmouati H, Skiredj A, Rojas A, Gohier A, et al. Deep generative models for ligand-based de novo design applied to multi-parametric optimization. J Comput Chem. 2022 Apr 15;43(10):692–703. doi:10.1002/jcc.26826 PubMed PMID: 35218219.

37. He J, Nittinger E, Tyrchan C, Czechtizky W, Patronov A, Bjerrum EJ, et al. Transformer-based molecular optimization beyond matched molecular pairs. J Cheminformatics. 2022 Mar 28;14(1):18. doi:10.1186/s13321-022-00599-3

38. Ross J, Belgodere B, Chenthamarakshan V, Padhi I, Mroueh Y, Das P. Large-Scale Chemical Language Representations Capture Molecular Structure and Properties. arXiv.org. 2021 Jun 17. doi:10.48550/arXiv.2106.09553

39. Fialková V, Zhao J, Papadopoulos K, Engkvist O, Bjerrum EJ, Kogej T, et al. LibINVENT: Reaction-based Generative Scaffold Decoration for in Silico Library Design. J Chem Inf Model. 2022 May 9;62(9):2046–63. doi:10.1021/acs.jcim.1c00469

40. Guo J, Knuth F, Margreitter C, Janet JP, Papadopoulos K, Engkvist O, et al. Link-INVENT: generative linker design with reinforcement learning. Digit Discov. 2023 Apr 11;2(2):392–408. doi:10.1039/D2DD00115B

41. Williams RJ, Zipser D. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. Neural Comput. 1989 Jun;1(2):270–80. doi:10.1162/neco.1989.1.2.270

42. Richard S. Sutton, Andrew G. Barto. Reinforcement learning: an introduction. Cambridge (Massachusets): MIT Press; 2018.

43. MolecularAI/REINVENT4 [Python] [Internet]. AstraZeneca - Molecular AI; 2026 [cited 2026

Jun 13]. Available from: https://github.com/MolecularAI/REINVENT4

44. Pearson W. uiri/toml [Python] [Internet]. 2026 [cited 2026 Jun 13]. Available from: https://github.com/uiri/toml

45. Landrum G, Tosco P, Kelley B, Rodriguez R, Cosgrove D, Vianello R, et al. rdkit/rdkit: 2026_03_3 (Q1 2026) Release [Internet]. Zenodo; 2026 [cited 2026 Jun 13]. Available from: https://zenodo.org/records/20446949 doi:10.5281/zenodo.20446949

46. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, et al. The Protein Data Bank. Nucleic Acids Res. 2000 Jan 1;28(1):235–42. doi:10.1093/nar/28.1.235 PubMed PMID: 10592235; PubMed Central PMCID: PMC102472.

47. Dai X, Wu L, Sun R, Zhou ZH. Atomic Structures of Minor Proteins VI and VII in Human Adenovirus. J Virol. 2017 Nov 30;91(24):10.1128/jvi.00850-17. doi:10.1128/jvi.00850-17

48. NCBI Resource Coordinators. Database resources of the National Center for Biotechnology Information. Nucleic Acids Res. 2018 Jan 4;46(D1):D8–13. doi:10.1093/nar/gkx1095

49. Löffler H. REINVENT4 priors [Internet]. Zenodo; 2025 [cited 2026 Jun 13]. Available from: https://zenodo.org/records/15641297 doi:10.5281/zenodo.15641297

50. Berthold MR, Cebron N, Dill F, Gabriel TR, Kötter T, Meinl T, et al. KNIME: The Konstanz Information Miner. Preisach C, Burkhardt H, Schmidt-Thieme L, Decker R, editors. Data Anal Mach Learn Appl. 2008;319–26. doi:10.1007/978-3-540-78246-9_38

51. Ziv Y, Imrie F, Marsden B, Deane CM. MolSnapper: Conditioning Diffusion for Structure-Based Drug Design. J Chem Inf Model. 2025 May 12;65(9):4263–73. doi:10.1021/acs.jcim.4c02008

52. Šícho M, Luukkonen S, van den Maagdenberg HW, Schoenmaker L, Béquignon OJM, van Westen GJP. DrugEx: Deep Learning Models and Tools for Exploration of Drug-Like Chemical Space. J Chem Inf Model. 2023 Jun 26;63(12):3629–36. doi:10.1021/acs.jcim.3c00434