Developing a Reactive Fragment Database for Lead Optimization

Introduction

The primary objective of this project is to streamline the lead optimization process by automating two key tasks:

1. **Reaction Visualization**: Create an RDKit-based script to generate simple images representing a set of chemically tractable reactions (sourced from a GitHub repository). Each reaction should be labeled with a text-based name as well.

2. **Reactive Fragment Database Construction**: Develop a Python script to query PubChem for small molecules (<150 Da) containing reactive moieties that can participate in the listed reactions. If feasible, filter compounds to those that are commercially available.

Eventually, this work will be integrated into MolModa, allowing researchers to efficiently explore and implement synthetically accessible fragment-growing strategies for lead optimization. In MolModa, users will be able to provide starting molecules, and the MolModa plugin will add fragments to those starting molecules according to chemically tractable reactions. The project is designed to be completed in approximately 10 weeks.

Timeline and Phases

Week 1–2: Project Setup and Familiarization

- **Objective**
  Familiarize with the project scope, the RDKit library, and the GitHub repository of chemical reactions.

- **Tasks**

  - **Task 1**: Review the GitHub repository containing the list of chemically tractable reactions.
  - **Task 2**: Install and test RDKit (or confirm existing setup).
  - **Task 3**: Explore RDKit's reaction functionality and documentation (e.g., `rdChemReactions` and reaction SMARTS).
  - **Task 4**: Gather references and tutorials on reaction depiction (e.g., `MolToImage`).

- **Goals**

- Understand the structure and format of the reaction data from GitHub.
- Acquire proficiency in RDKit's methods for visualizing molecules and reactions.

- **Resources**

  - GitHub repository with reaction definitions.
  - RDKit Documentation.
  - Tutorials on RDKit reaction handling (e.g., official RDKit blog, community examples).

Week 3–4: Reaction Depiction Script Development

- **Objective**
  Develop a Python script using RDKit that generates simple reaction scheme images and extracts reaction names. These images and names will eventually be used in MolModa so users can select reactions.

- **Tasks**

  - **Task 1**: Parse the reaction definitions (SMIRKS/SMARTS) from the GitHub list.
  - **Task 2**: Implement a function to produce image files (e.g., SVG) for each reaction.
  - **Task 3**: Record the reaction name or label in an associated text file.

- **Goals**

  - Automate the creation of reaction images for easy integration into the MolModa interface.
  - Ensure each reaction image is accurate, clear, and labeled with the correct name.

- **Resources**

  - RDKit's `rdChemReactions.ReactionFromSmarts` or `ReactionFromSmirks` documentation.

- **Output**

  - **Reaction Depiction Script**: A standalone Python script (e.g., `reaction_visualizer.py`) that reads reactions from the GitHub file and outputs labeled reaction images.
  - A directory containing generated images and a corresponding CSV or JSON file mapping each reaction to its name and image path.

Week 5–6: PubChem Query Script — Basic Implementation

- **Objective**
  Create a Python script to query PubChem for molecules with specific reactive moieties matching each

reaction. These compounds will eventually be used to perform chemical reactions *in silico*, all within the MolModa interface.

- **Tasks**
  - **Task 1**: Research PubChem's API (REST or PUG-REST) and identify relevant endpoints for searching by substructure.
  - **Task 2**: Implement a function to construct a query string or advanced search that identifies molecules containing the required reactive groups (e.g., substructure search queries).
  - **Task 3**: Incorporate basic filters, such as molecular weight (< 150 Da).
  - **Task 4**: Explore feasibility of filtering by commercial availability (if PubChem provides relevant metadata).
- **Goals**
  - Develop a proof-of-concept script that can retrieve relevant small molecules from PubChem.
  - Validate the returned compounds to ensure they have the correct reactive functionalities.
- **Resources**
  - PubChem PUG-REST Documentation.
- **Output**
  - **PubChem Query Script** (e.g., `pubchem_search.py`), capable of returning sets of candidate molecules in SMILES format (with CIDs).
  - Preliminary dataset of small molecules (<150 Da) for at least one or two reactions as a proof-of-concept.

Week 7–8: Refinement and Expansion of PubChem Queries

- **Objective**
  Expand and refine the PubChem query script to handle multiple reactions.
- **Tasks**
  - **Task 1**: Automate the process so each reaction from the GitHub list triggers a relevant substructure search.
  - **Task 2**: Store fetched molecule data in a structured format (JSON).
- **Goals**
  - Build a pre-compiled library for each reaction, mapping each reactive moiety to a collection of

commercially available, small molecular fragments.

- **Resources**
  - Scripts from prior weeks and examples of advanced PubChem queries.
  - Additional substructure search references or community-driven solutions for filtering.
- **Output**
  - **Refined PubChem Query Script** with multi-reaction support.
  - Structured “Reactive Fragment Libraries” dataset for each reaction, ready for direct integration into MolModa.

Week 9–10: Final Phase: Documentation and Presentation

- **Objective**
  Wrap up the project by finalizing documentation and preparing final deliverables.
- **Tasks**
  - **Task 1**: Write a clear README file detailing script usage, parameters, and output formats.
  - **Task 2**: Prepare final deliverables (scripts, data, instructions).
- **Goals**
  - Ensure the codebase is polished, well-documented, and easily maintainable.
  - Provide a comprehensive guide to future users or collaborators.
- **Resources**
  - Online or internal documentation best practices (e.g., README).
  - Feedback on final script functionality.
- **Output**
  - **Finalized Codebase**: Finalized versions of the reaction depiction and PubChem query scripts.
  - **User Documentation**: README describing each script’s usage, required dependencies, and examples.

Overall Deliverables

1. **Reaction Depiction Script**

- Generates images for a set of chemically tractable reactions.
- Labels each reaction with its name, suitable for integration into MolModa.

2. **PubChem Query Script**

    - Identifies small molecules (<150 Da) with reactive moieties suitable for each reaction.
    - Filters for commercial availability, if feasible.

3. **Reactive Fragment Libraries**

    - Pre-compiled sets of commercially available molecules for each reaction type, ready for *in silico* reaction modeling.

4. **Documentation and User Guide**

    - A README file describing script usage, installation instructions, dependencies, and examples.