Similarity Searching in PubChem Using SMILES

Similarity searching is an essential tool in computer-aided drug discovery, enabling researchers to identify compounds that are structurally similar to a query molecule. Such compounds may have improved interactions with a target protein, so similarity searching is vital for lead optimization. This project focuses on developing a Python script to perform Tanimoto similarity searches, as well as substructure and superstructure searches, using the PubChem API.

Week 1-2: Tanimoto Similarity Search

- Objective:
  - Implement Tanimoto similarity searching functionality.
- Tasks:
  - Write a Python function to query the PubChem API for compounds similar to a given SMILES.
  - Allow users to set a customizable Tanimoto coefficient threshold.
  - Parse the JSON response to extract relevant details, such as compound identifiers and SMILES.
- Goals:
  - Retrieve a list of similar compounds based on the Tanimoto similarity metric.
- Resources that might help (not carefully checked):
  - PubChem Similarity Search API: Similarity Search
  - Python HTTP Requests Library: Requests Documentation
- Output:
  - A Python function that returns a JSON object containing compounds similar to the query molecule.

Week 3-4: Substructure and Superstructure Searches

- Objective:
  - Extend the script to perform substructure and superstructure searches.
- Tasks:

    - Write Python functions to query the PubChem API for substructures and superstructures of the input SMILES.
    - Parse JSON responses to extract compound identifiers and structural data (CID, SID, SMILES).

- Goals:

    - Retrieve compounds that are substructures or superstructures of the query molecule.

- Resources that might help (not carefully checked):

    - PubChem Substructure Search API: Substructure Search

- Output:

    - Python functions that return JSON objects for substructure and superstructure search results.

Week 5: Refinement and Error Handling

- Objective:

    - Enhance the script's robustness and usability.

- Tasks:

    - Add error handling for failed queries or invalid inputs.
    - Ensure the JSON outputs are well-structured and clear.
    - Test the script with a variety of SMILES inputs to confirm reliability.

- Goals:

    - Deliver a robust script that handles errors gracefully and produces reliable results.

- Resources that might help (not carefully checked):

    - Python Error Handling Guidelines: Error Handling

- Output:

    - A refined and reliable Python script.

Week 6: Documentation and Finalization

- Objective:
  - Document the script and prepare it for delivery.
- Tasks:
  - Write a README file with clear instructions for using the script, including examples.
  - Package the script and documentation for submission.
- Goals:
  - Ensure the script is easy to use and well-documented.
- Resources that might help (not carefully checked):
  - Python Documentation Best Practices: Real Python Docs
- Output:
  - A finalized Python script and README file.

Overall Deliverables:

1. A Python script with functions for Tanimoto similarity, substructure, and superstructure searches using the PubChem API.
2. A README file documenting the script's usage and functionality.