Scientific Data Extraction App

Cel aplikacji

Scientific Data Extraction App to narzędzie przeznaczone do przeglądania i analizy danych biologicznych zawierających komórki raka piersi. Aplikacja umożliwia wyszukanie związku na wykresie i podgląd jego szczegółów. Każdy punkt na wykresie przedstawia parę: związek i jego skoncentrowanie, który jest pokolorowany w zależności od poziomu skoncentrowania lub wartości MOA. Po wybraniu punktu wyświetlają się szczegółowe dane takie jak: nazwa związku, wartość SMILES, wartość MOA, stężenie.

Funkcjonalności aplikacji

1. **Interaktywna wizualizacja danych**
   Użytkownicy mogą przeglądać dane w formie interaktywnych wykresów w przeglądarce, gdzie punkty są kolorowane w zależności od poziomu stężenia lub wartości MOA.
2. **Szczegółowe informacje o związkach**
   Po wybraniu punktu na wykresie wyświetlane są szczegółowe dane dotyczące wybranego związku, takie jak: nazwa, wartość SMILES, wartość MOA oraz stężenie.

Uogólniona struktura projektu

Strukturę projektu można podzielić na **frontend** i **backend**. Głównym zadaniem backendu jest analiza plików .csv oraz zdjęć w formacie .tif zgromadzonych na stronie: [BBBC021](). Zdjęcia przed analizą programu należy przetworzyć przez aplikację **Cell Profiller** z użyciem pipeline'u dostępnego na wyżej wymienionej stronie.

Główne elementy oprogramowania

Klasa Program

Główna klasa odpowiedzialna za:

- utworzenie bazy danych wraz z tabelami,
- modyfikowanie plików .csv i przystosowanie ich do analizy,
- wypełnienie bazy początkowymi danymi,
- wyliczenie wektorów rozmieszczenia punktów na podstawie danych zebranych z plików powstałych po analizie przez program **Cell Profiller**.

Wyliczone dane są przetwarzane i przekazywane do modułu aplikacji (app), który, dzięki wykorzystaniu technologii **FastAPI**, udostępnia je w warstwie front-endowej.

Opis klas zajmujących się przetwarzaniem danych

1. **Klasa DatabaseCreator**
   Zarządza połączeniem z bazą danych oraz tworzeniem tabel. Obsługuje cztery rodzaje tabel (**compounds**, **images**, **color_by_concentration**, **color_by_moa**) i zatwierdza zmiany po każdej operacji. Automatycznie zamyka połączenie z bazą danych po zakończeniu pracy.

2. **Klasa DatabaseFiller**
   Wypełnia bazę danych informacjami o związkach chemicznych, ich stężeniach oraz wartościach MOA na podstawie danych z plików CSV. Tworzy unikalne kolory dla stężeń i MOA, przypisując je do odpowiednich rekordów w bazie. Działa w oparciu o interfejs bazy danych, zapewniając spójność i integralność danych.

3. **Klasa DatabaseInterface**
   Interfejs definiujący zestaw abstrakcyjnych metod do operacji na bazie danych, takich jak:
   - łączenie się z nią,
   - zamykanie połączenia,
   - dodawanie danych do tabel,
   - aktualizowanie rekordów,
   - pobieranie danych.
     Wszystkie te operacje muszą być zaimplementowane w klasie dziedziczącej. Celem jest zapewnienie jednolitego sposobu obsługi bazy danych w aplikacji.

4. **Klasa SQLiteDatabase**
   implementuje wymieniony wyżej interfejs

5. **Klasa Paths**
   Zawiera stałe przechowujące ścieżki do różnych zasobów w projekcie. Używa modułu **Path** do tworzenia dynamicznych ścieżek, zależnych od bieżącego katalogu roboczego. Przechowywane są tu ścieżki do folderów bazy danych, zasobów oraz plików CSV.

6. **Klasa CsvFormatter**
   Służy do przetwarzania plików CSV, usuwając określoną liczbę kolumn i zapisując przetworzone dane w nowym folderze. Umożliwia dynamiczne tworzenie folderów wyjściowych oraz modyfikowanie ścieżek plików, dodając do nich "_formatted". Pliki CSV są przetwarzane w częściach, co pozwala na oszczędność pamięci przy pracy z dużymi danymi. Główna metoda **run_formatter** wykonuje całą operację, analizując pliki z katalogu wejściowego i zapisując przetworzone wersje w katalogu wyjściowym.

7. **Klasa CalculateVectors**
   Wykonuje obliczenia wektorów średnich na podstawie plików CSV, a następnie przekształca je do przestrzeni 2D za pomocą algorytmu **UMAP**. Umożliwia iterację po folderach, przetwarzanie danych z plików oraz powiązanie obliczonych wektorów z obrazami zapisanymi w bazie danych. Obliczone współrzędne (x, y) są zapisywane w bazie danych, a obrazy związane z danymi są przypisywane do odpowiednich związków chemicznych. Proces obejmuje wczytywanie danych z CSV, tworzenie wektorów, konwersję do 2D oraz zapis wyników do bazy danych.

API do zarządzania związkami

Sekcja kodu zajmująca się wystawianiem endpointów w aplikacji **FastAPI** definiuje cztery różne trasy:

1. **/compounds**
   Pobiera wszystkie związki chemiczne z bazy danych za pomocą funkcji **get_all_compounds()** z klasy **Repository**.
2. **/compounds/colored_by_concentration**
   Pobiera wszystkie związki chemiczne z bazy danych, uwzględniając kolorowanie według stężenia, dzięki funkcji **get_all_compounds_colored_by_concentration()**.
3. **/compounds/colored_by_moa**
   Pobiera związki chemiczne z kolorowaniem według wartości MOA za pomocą **get_all_compounds_colored_by_moa()**.
4. **/compound/details/{compound_name}/{compound_concentration}**
   Pobiera szczegóły dotyczące konkretnego związku na podstawie nazwy i stężenia, używając **get_compound_details()**.

Klasa Repository odpowiada za komunikację z bazą danych, pobieranie danych o związkach oraz ich szczegółów, zwracając je w odpowiedniej strukturze.