Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Модель штучної імунної системи для обробки зображень

(тема)

Виконав:

студент II курсу, групи СПм-22-6

Бурда А.С.
(прізвище, ініціали)

Спеціальність

123 «Комп’ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма

Системне програмування
(повна назва освітньої програми)

Керівник: ст. викл. Фомічов О.О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп'ютерна інженерія»
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Бурді Анатолію Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Модель штучної імунної системи для обробки зображень

затверджена наказом по університету від " 01 " квітня 2024 р. № 257 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 15 червня 2024 р.

3. Вхідні дані до роботи
штучна імунна система
ШІС
модель
класифікація та кластеризація

4. Перелік питань, що потрібно опрацювати у роботі
Аналіз предметної області та постановка завдання
Обґрунтування роботи розробленох моделі ШІС
Реалізація програмних засобів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 15 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання та аналіз літератури	01.04.2024 – 06.04.2024	
2	Огляд існуючих рішень та методів	07.04.2024 – 12.04.2024	
3	Розробка моделі	13.04.2024 – 18.04.2024	
4	Вибір програмних засобів	19.04.2024 – 25.04.2024	
5	Програмна реалізація	26.04.2024 – 02.05.2024	
6	Аналіз отриманих результатів	03.05.2024 – 16.05.2024	
7	Оформлення записки	17.05.2024 – 14.06.2024	
8	Представлення роботи в ЕК	15.06.2024	

Дата видачі завдання 01 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ ст. викл. Фомічов О.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 58 с., 19 рис., 2 дод., 11 джерел.

НЕГАТИВНИЙ ВІДБІР, ШТУЧНА ІМУННА СИСТЕМА, МОДЕЛЬ, КОМП’ЮТЕРНА СИСТЕМА.

Метою кваліфікаційної роботи є аналіз існуючих методів та моделей штучної імунної системи для обробки зображень.

В ході підготовки кваліфікаційної роботи проведено аналіз існуючих методів та моделей штучної імунної системи для обробки зображень. Проведено аналіз найбільш поширених класичних методів та моделей штучних імунних систем для кластеризації даних. Виділено набір універсальних імунних операторів та проаналізовано можливості їх модіфікації задля підвищення якості кластеризації. Сформовано модель деревовидної штучної імунної мережі та алгоритм Dendric-aiNet для кластеризації даних. Розроблено програмний застосунок для модулювання алгоритмів кластеризації

ABSTRACT

Master’s thesis: 58 pages, 19 figures, 2 appendices, 11 sources.

NEGATIVE SELECTION, ARTIFICIAL IMMUNE SYSTEM, MODEL, COMPUTER SYSTEM.

The major goal of this thesis is to analyze the existing methods and models of the artificial immune system for image processing.

In order to analysis of existing methods and models of the artificial immune system for image processing was carried out. An analysis of the most common classical methods and models of artificial immune systems for data clustering was carried out. A set of universal immune operators was identified and the possibilities of their modification were analyzed in order to improve the quality of clustering. A tree-like artificial immune network model and Dendric-aiNet algorithm for data clustering have been developed. A software application for modulating clustering algorithms has been developed

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ 7
ВСТУП 8
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ 10
1.1 Біологічні основи штучних імунних систем 10
1.2 Аналіз ШІС 13
1.3 Обчислювальні моделі ШІС 16
1.3.1 Модель негативного відбору 16
1.3.2 Модель позитивного відбору 17
1.3.3 Модель клонального відбору 18
1.3.4 Модель мережевої взаємодії 19
1.3.5 Інші моделі 21
1.4 Програмне забезпечення для роботи з ШІС 21
2 ОБҐРУНТУВАННЯ РОБОТИ РОЗРОБЛЕНОХ МОДЕЛІ ШІС 25
2.1 Огляд технологій розробки 26
2.2 Аналіз технології Docker 30
3 РЕАЛІЗАЦІЯ ПРОГРАМНИХ ЗАСОБІВ 35
3.1 Реалізація алгоритмів 35
3.2 Використання ШІС 39
3.3 Програмна реалізація 41
3.4 Використання БД 42
ВИСНОВКИ 47
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 48
ДОДАТОК А Графічний матеріал кваліфікаційної роботи 49
ДОДАТОК Б Апробація 58

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних

ІАД – інтелектуальний аналіз даних

ОС – операційна система

ПЗ – програмні засоби

ПІС – природна імунна система

ШІС – штучна імунна система

ШНМ – штучна нейронна мережа

ВСТУП

Штучні імунні системи (ШІС) є парадигмою обчислювального інтелекту, яка заснована на тих концепціях, які спостерігаються в природній імунній системі кожної людини (ШІС). Імунна система виконує обробку зовнішніх сигналів, використовуючи високопаралельний процес. Імунна система є чудовим прикладом системи, що реалізує адаптивні процеси на глобальному рівні на основі локальних взаємодій.

Для вирішення завдань розпізнавання та класифікації імунна система використовує механізми навчання, пам'яті та асоціативного пошуку. SIS розглядає різні механізми імунітету та їхнє ставлення до обробки інформації та вирішення проблем. У порівнянні з іншими моделями та методами, заснованими на біологічних уявленнях, такими як нейронні мережі та еволюційні алгоритми, штучні імунні системи досі привертали дуже мало уваги. Імунна система має великий інтерес як система, здатна ефективно обробляти великі обсяги даних. Зокрема, він виконує велику кількість складних високопаралельних розподілених обчислень [1].

Поведінка імунної системи загалом визначається всією сукупністю локальних взаємодій. Відповідні методи успішно використовуються для вирішення завдань розпізнавання образів, діагностики та усунення несправностей, а також для створення систем комп'ютерної безпеки та деяких інших програм.

У ШІС використовуються обчислювальні моделі, засновані на принципі роботи ПІС, такі як модель клональної селекції, взаємодії, негативної селекції та ін. Модель негативного відбору – одна з найвідоміших моделей ШІС.

Його принцип роботи полягає у генерації детекторів, що відповідають аномальній поведінці досліджуваного об'єкта. Метод негативного відбору складається з двох етапів: навчання та класифікації. Огляд наявного

програмного забезпечення показує наявність кількох окремих розрізнених програм для роботи з ШІС, тому важливо розробити програмне забезпечення для реалізації ШІС для виявлення аномалій у вихідних даних з урахуванням обчислювальної моделі негативної селекції.

Метою кваліфікаційної роботи є розробка програмних засобів аналізу даних з використанням генетичних алгоритмів на основі штучних імунних систем.

Завдання:

- аналіз існуючих програмних засобів;
- аналіз принципів та методів побудови штучних імунних систем;
- розробка класів об’єктно-орієнтованої моделі ШІС;
- вибір програмних засобів для реалізації розглянутих алгоритмів;
- реалізація та тестування програмних модулів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Біологічні основи штучних імунних систем

Природна імунна система є складною системою, що складається з кількох функціонально різних компонентів. Імунна система використовує багаторівневий захист проти зовнішніх антигенів, включаючи дію неспецифічних (вроджених) та специфічних (придбаних) захисних механізмів.

У процесі еволюції імунна система виявиться розрізняти зовнішні антигени (бактерії та віруси) та спільні клітини або молекули організму. Встановлено, що в людини підтримується велика кількість імунокомпетентних клітин, які циркулюють по всьому тілу.

Основним типом клітин, що беруть участь у імунній відповіді та володіють властивостями специфічності, різноманітності, пам'яті та адаптивності, є лімфоцити. В організмі є два основних типи лімфоцитів – Т-(рисунок 1.1) та В-лімфоцити(рисунок 1.2). Основною їх функцією є захоплення антигенів та забезпечення взаємодії з ними лімфоцитів для стимуляції імунної відповіді. Зазначені два типи лімфоцитів відіграють в імунній відповіді різну роль, хоч і можуть взаємодіяти, контролюючи таким чином свої функції.

При попаданні антигену в організм лише мала частина клітин імунної системи здатна розпізнавати його пептидів. Таке розпізнавання стимулює процеси розмноження та диференціювання лімфоцитів, що призводять до утворення клонів ідентичних клітин (або антитіл). Цей процес, званий розмноженням клону формує численну популяцію специфічних до антигену антитілопродукуючих клітин. Розмноження клону імунокомпетентних клітин призводить до руйнування чи нейтралізації антигену.

Антиген – це ініціатор та рушійна сила всіх реакцій набутого імунітету. Імунна система виникла для розпізнавання та руйнування чужорідних антигенів, а також усунення джерела їх утворення – бактерій, інфікованих вірусом клітин тощо. Коли антиген елімінovано, імунна відповідь припиняється. Частина клітин, що утворилися, зберігається для імунної пам'яті. В результаті, подальша дія схожого антигену призводить до більш швидкої імунної реакції (вторинної відповіді). Реакція В-лімфоциту на антиген представлена рисунку 1.1.

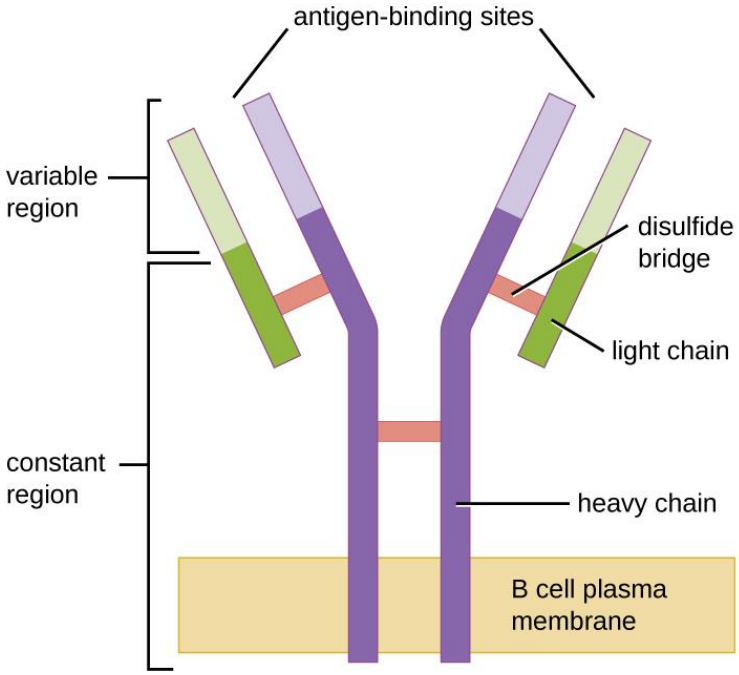

Рисунок 1.1 – Взаємодія В-лімфоцитів та антигенів

Процес циркуляції В- і Т-лімфоцитів у первинних та вторинних лімфоїдних органах ретельно контролюється, що забезпечує попадання відповідних клітинних популяцій – ненавчених клітин, а також клітин пам'яті – у різні місця призначення. Клітини пам'яті надають вибіркове перевагу тому типу тканин, у якому вони вперше зустрілися з антигеном. Ймовірно, це забезпечує повернення кожної конкретної клітини пам'яті до тієї ділянки тіла, де вона, швидше за все, знову зустрінеться з антигеном. Реакція Т-лімфоциту на патоген представлена рисунку 1.2.

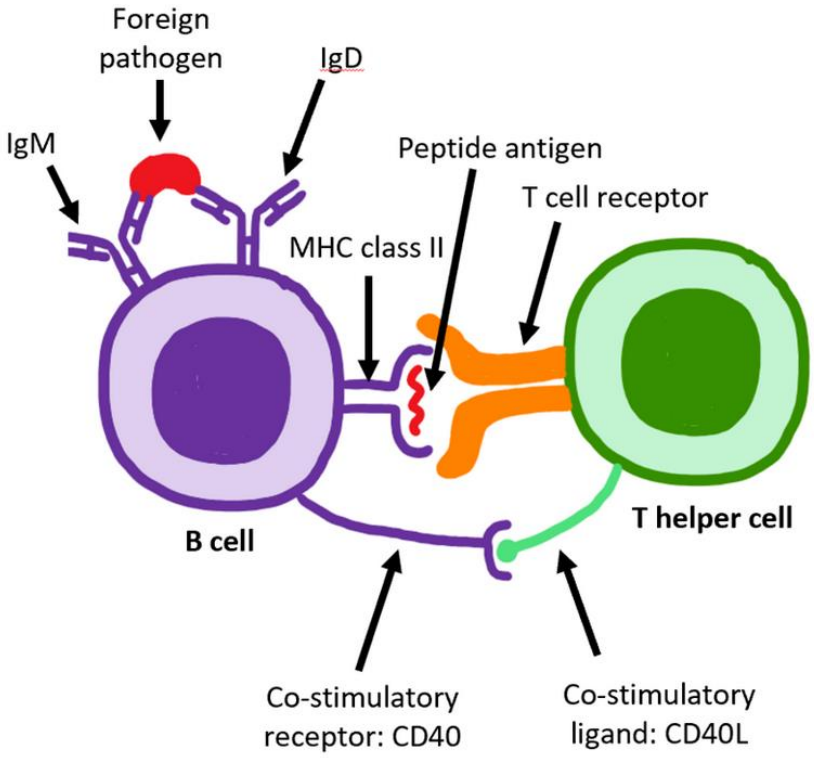

Рисунок 1.2 – Взаємодія Т-лімфоцитів та антигенів

Існує два основних варіанти імунної відповіді: гуморальний, опосередкований В-клітинами та їх продуктами, та клітинний, у якому беруть участь Т-клітини. Обом варіантам імунних реакцій відповідають подібні послідовності етапів захисту організму: активація, поділ, диференціювання, секреція, імунна атака, супресія та пам'ять, проте вони здійснюються у різний спосіб. У регуляції як гумарального, так і клітинного імунітету беруть участь популяції Т-клітин, які називаються хелперними або супресорними клітинами, які посилюють або, відповідно, пригнічують імунну відповідь. Таким чином, основна роль імунної системи полягає в розпізнаванні всіх клітин організму та класифікації їх як «своїх» чи «чужих». Чужорідні клітини зазнають подальшої класифікації з метою стимуляції захисного механізму відповідного типу.

1.2 Аналіз ШІС

Штучні імунні системи – це адаптивні системи, основу яких лежить теоретична імунологія і спостережувані принципи, функції та моделі імунітету, застосовувані на вирішення завдань у різних проблемних областях. ШІС є напрямом штучного інтелекту, що емулює природну імунну систему. В ШІС використовується здатність природної імунної системи виробляти нові типи антитіл і відбирати найбільш підходящі з них для взаємодії з антигенами, що потрапили в організм.

ШІС використовують обчислювальні моделі обробки інформації в імунологічних взаємодіях з практичними застосуваннями для вирішення багатьох проблем, таких як оптимізація та відновлення функцій, розпізнавання образів, пошук даних, комп'ютерна безпека, виявлення помилок, класифікація, оптимізація та ін [7]. Імунна система має великий інтерес як система, здатна ефективно обробляти значні обсяги даних. Зокрема, вона виконує великий обсяг складних високопаралельних розподілених обчислень.

Поведінка імунної системи загалом визначається всією сукупністю локальних взаємодій. Щодо можливих додатків для обробки інформації перспективні такі властивості імунної системи:

- розпізнавання;
- виділення особливостей;
- різноманітність;
- навчання;
- пам'ять;
- розподілений пошук;
- саморегуляції;
- пороговий механізм;
- динамічний захист;
- ймовірнісне виявлення.

При розпізнаванні імунна система здатна розпізнавати та класифікувати різні молекулярні структури та вибірково на них реагувати. Розпізнавання відбувається в ході міжклітинних контактів, при цьому сила зв'язків, що утворюються, визначається формою молекул і величиною електростатичного заряду. Розпізнавання свого та чужого є одним із основних завдань, яке вирішує імунна система.

У властивостях виділення особливостей антиген-презентуючі клітини інтерпретують антигенне оточення та виділяють особливості шляхом обробки антигенів та представлення антигенних пептидів на своїй поверхні. При різноманітності імунна система використовує комбінаторний механізм (генетично-обумовлений процес) для утворення безлічі різних рецепторів лімфоцитів, щоб гарантувати, що хоча б один лімфоцит з усієї сукупності зможе провзаємодіяти з будь-яким наперед заданим (відомим або невідомим) антигеном.

У властивостях навчання імунна система оцінює структуру конкретного антигену, використовуючи його випадкові контакти з складовими цієї системи клітинами. Навчання полягає у зміні концентрації лімфоцитів, що відбувається за первинної відповіді (в результаті першого контакту з антигеном). Отже, здатність імунної системи до навчання закладена головним чином у механізмі поповнення клонів, що призводить до утворення нових імунокомпетентних клітин з урахуванням поточного стану системи (цей процес має назву розмноження клону). У категорії пам'ять невелика частина лімфоцитів, що знаходяться в активованому стані, стає клітинами пам'яті (асоціативна пам'ять).

Вважається, що час життя клітин пам'яті є динамічною величиною та визначається частотою стимуляції антигенами. Використовуючи короткочасні та довгострокові механізми імунної пам'яті, імунна система підтримує ідеальний баланс між економією ресурсів та виконанням функції за рахунок збереження мінімально необхідної, але достатньої пам'яті про

попередні контакти з антигеном. При розподіленому пошуку імунна система – це розподілена система.

Клітини імунної системи, головним чином лімфоцити, безперервно рециркулюють через кров, лімфу, лімфоїдні органи та інші тканини. У разі зустрічі з антигеном вони здійснюють специфічну імунну відповідь. У саморегуляції імунний захист має властивість саморегуляції. Центрального органу, який контролює функції імунної системи, не існує. Залежно від способу проникнення в організм та інших властивостей антигену, регуляція імунної відповіді може бути як локальною, так і системною. Як пороговий механізм імунна відповідь і розмноження імунокомпетентних клітин відбуваються лише після подолання деякого порога, що залежить від сили хімічних зв'язків.

При динамічному захисті клональне розмноження та соматичне гіпермутування дозволяють імунній системі продукувати високоафінні імунокомпетентні клітини, що створює динамічний баланс між вивчальною та захисною функцією адаптивного імунітету. Наявність динамічного захисту поступово призводить до розширення зони спостереження, контрольованої імунною системою. У імовірнісному виявленні перехресні реакції в ході імунної відповіді – це стохастичний процес. До того ж, виявлення антигену завжди неминуче відбувається приблизним чином; отже лімфоцит може взаємодіяти з кількома подібними структурно антигенами.

В імунній відповіді на антиген важливу роль відіграють інші характеристики імунної системи, такі як адаптованість, специфічність, самотолерантність, диференціювання та інші. Всі ці властивості, що стосуються обробки інформації, створюють ряд цікавих можливостей з обчислювальної точки зору.

Області, де застосовуються ШІС:

- методи обчислень;
- когнітивні моделі;
- розпізнавання образів;

- методи виявлення аномалій та несправностей;
- мультиагентні системи;
- моделі самоорганізації;
- моделі колективного інтелекту;
- системи пошуку та оптимізації;
- моделі автономних розподілених систем;
- моделі штучного життя; – системи комп'ютерної та інтернет – безпеки;
- моделі систем, що навчаються;
- методи отримання інформації;
- штучні імунні системи для виявлення підробок;
- методи обробки сигналів та зображень;
- оптимізація;
- розпізнавання образів;
- комп'ютерна безпека;
- інтелектуальний аналіз та вибірка даних;
- навчання адаптивних систем.

1.3 Обчислювальні моделі ШІС

Основними обчислювальними моделями ШІС є модель клонального відбору, мережевої взаємодії, негативного та позитивного відбору.

1.3.1 Модель негативного відбору

Модель негативного відбору є однією з найвідоміших моделей ШІС (рисунок 1.3). Її принцип роботи полягає у генерації детекторів, що відповідають аномальній поведінці досліджуваного об'єкта. Метод негативного відбору складається з двох фаз: навчання та класифікації.

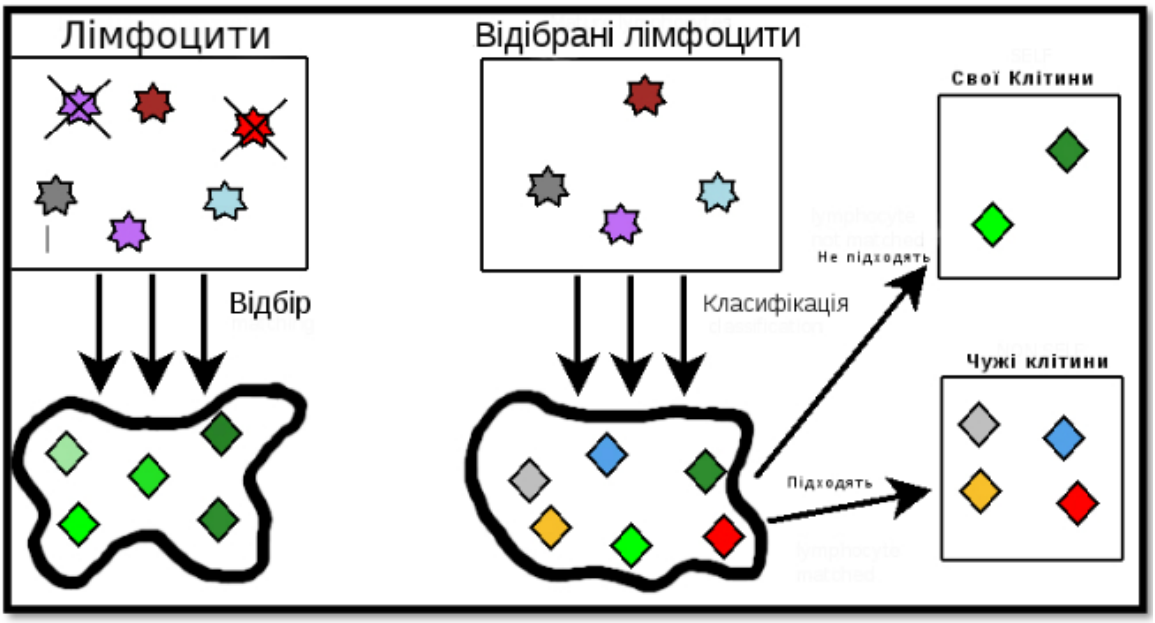

Рисунок 1.3 – Модель негативного відбору

Під час навчання на вхід моделі надходять "свої" антигени (self-antigen). При цьому в ШІС виробляються випадкові клітини, які стають детекторами лише в тому випадку, якщо вони не реагують на жодний з антигенів, а також на вже існуючі імунні клітини (антитіла). Таким чином, отримана під час навчання популяція клітин значною мірою визначає простір передбачуваних «чужих» клітин. У процесі розпізнавання клітина вважається патогенною («чужою»), якщо її розпізнає хоч один із існуючих детекторів. Як недоліки моделі негативного відбору можна відзначити великі тимчасові витрати на роботу. Однак деякі його модифікації, зокрема V-детектор [7], показали добрі результати та визначили нові шляхи розвитку моделі.

1.3.2 Модель позитивного відбору

Позитивний відбір заснований на схемі роботи моделі негативного відбору, за винятком того, що детектором вважають випадкову клітину, що реагує хоча б один антиген з навчальної вибірки. Вочевидь, що негативний відбір проти позитивним ефективніший у разі, коли область «чужих» клітин

значно менше області «своїх». Таким чином, для правильного вибору моделі необхідно мати апріорні знання про розподіл «своїх» і «чужих» клітин, а це є можливим далеко не завжди.

1.3.3 Модель клонального відбору

Клональний відбір є найпоширенішою моделлю позитивного відбору (рисунок 1.4). Її особливість у тому, що після генерації випадкового позитивного детектора він піддається клонування, а потім здійснюється мутація кожного з клонів. Таким чином, з популяції клонів можна визначити найбільш відповідний антигену детектор, який і замінить батьківський детектор у загальній популяції. Цей підхід дозволяє прискорити процес збіжності способу, у своїй розмір популяції значно збільшується, що призводить до економії обчислювальних ресурсів.

Характеристики клонального відбору:

- розпізнавання антитілами специфічних антигенів; – відбір антитіл з високою афінністю та їх клонування;
- клонування антитіл пропорційно до афінності даного антитіла з антигеном – тобто, чим вище афінність, тим більша кількість клонів проводиться;
- генерація випадкових генетичних змін, шляхом внесення мутацій, для підтримки різноманітності популяції антитіл;
- мутація, якій піддаються антитіла обернено-пропорційна афінності антитіл до антигену;
- видалення новостворених антитіл, якщо вони мають низьку афінність;
- наявність клітин пам'яті для збереження знайдених рішень.

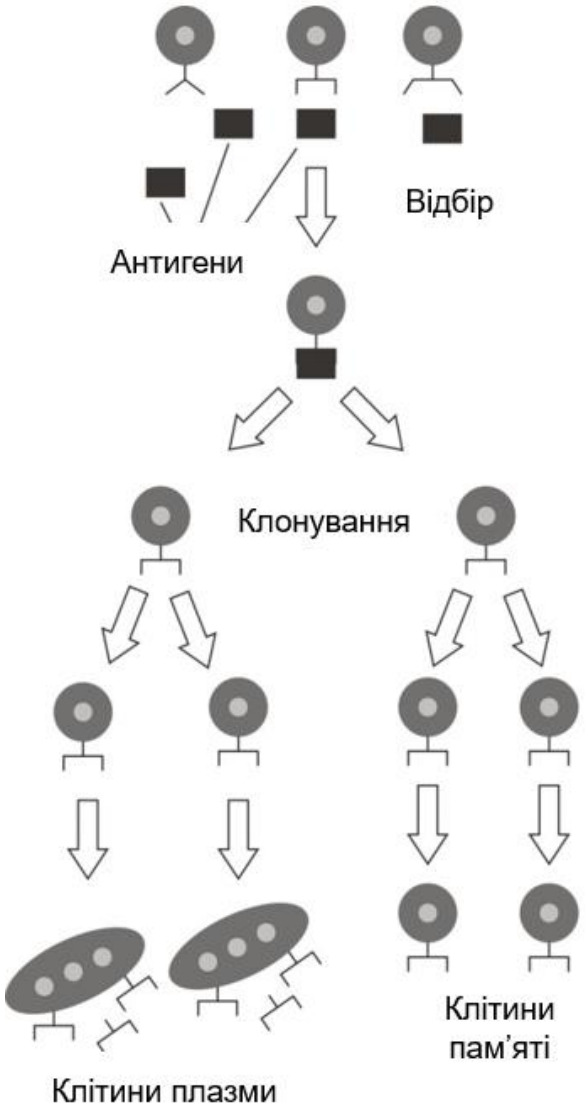

Рисунок 1.4 – Модель клонального відбору

1.3.4 Модель мережевої взаємодії

Імунна система являє собою регульовану мережу молекул і клітин, які розпізнають один одного навіть за відсутності антигену (рисунок 1.5). Такі структури часто називають ідіотіпіческіх мережами, вони служать математічною основою для вивчення поведінки імунної системи. Така теорія інтерпретується у вигляді системи диференціальних рівнянь, яка описує динаміку концентрації клонів лімфоцитів і відповідних молекул імуноглобулінів.

Теорія ідіотіпіческіх регуляції заснована на припущенні, що різні клони лімфоцитів один від одного не ізольовані, а підтримують зв'язок шляхом взаємодій між своїми рецепторамі і антитілами. Отже, розпізнавання антигену здійснюється не поодиноким клоном клітин, а скоріше на системному рівні, за участю різних клонів, взаємодіючих за типом реакцій антиген-антитіло як єдина мережа.

В ході імунної відповіді сам антиген викликає лише вироблення першого набору антитіл Ab_1. Потім ці антитіла, діючи в якості антигену, викликають вироблення другого набору «анти-ідіотіпіческіх» антитіл Ab_2, розпізнаних ідіотопи на антитіла Ab_1. Аналогічно здійснюється вироблення третього набору антитіл, Ab_3, які розпізнають антитіла Ab_2, і так далі. Ключовий постулат теорії полягає в тому, що одинична клітина продукує лише один тип антитіл; звідси слідує кілька висновків:

- існує аллельному виключення;
- всі антитіло-подібні рецептори на поверхні лімфоцита повинні бути ідентичними або, принаймні, мати ідентичні легкі ланцюги і варіабельні області важких ланцюгів;
- всі антитіла, які продукують даної клітиною, і її нащадки повинні мати однаковий ідіотипів.

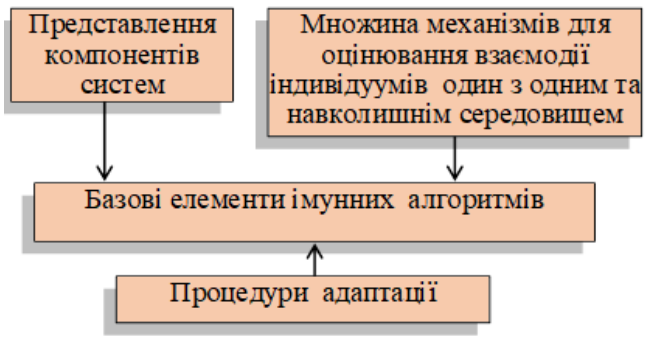

Рисунок 1.5 – Компоненти ШІС

Формулюючи основи своєї теорії, були введені поняття формальних та функціональних мереж. Формальні мережі служать для вивчення питань

репертуару, дуалізму і супрессии. При розгляді функціональних мереж представлена кількісна картина теорії. Даний підхід гранично математизированной і, в основному, пов'язаний з описом фазових переходів. Надалі площину фазових змінних даної системи рівнянь була розділена на докритическим область, область переходу і посткритическую область.

1.3.5 Інші моделі

Існують і інші обчислювальні моделі, що імітують різні імунологічні явища, наприклад здатність імунної системи знаходити спільні структури в зашумленной середовищі, можливість виявляти і підтримувати охоплення структур, що належать різним класам, а також здатність до ефективного навчання навіть у тому випадку, якщо експресувати не всі молекули імуноглобулінів і не всі антигени присутні.

У декількох роботах для моделювання процесу соматичних мутацій (при якому утворюються антитіла для розпізнавання конкретного антигену) застосовувалися генетичні алгоритми. Хоффман [22] зіставив імунну систему з нервовою системою, показав ряд аналогій, наявних на рівні загальної поведінки цих систем (що відрізняються на рівні структурних елементів), і теореті¬ческі допустив існування симетричною моделі нейронної мережі з поведінкою, аналогічним імунної реакції у відповідь на дію стимулу. Фармер і ін. [13], а також Берсін і Варела [12] зіставили імунну систему з самонавчанням класифікаторів.

1.4 Програмне забезпечення для роботи з ШІС

В даний час ШІС представлені багатьма різнорідними інструментами програмного забезпечення, які реалізують евристичні алгоритми, еволюційні обчислення і т.д.

Фірма IBM реалізувала проект створення імунної системи для кіберпростору, яка включає в себе наступні етапи:

- виявлення невідомого вірусу на машині користувача;
- виділення примірника вірусу і відправка його центрального комп'ютера;
- автоматичний аналіз вірусу і вироблення приписи для його розпізнавання і видалення з будь-якого пристрою, підключеного до мережі;
- доставка приписи комп'ютера користувача, вбудовування його в антивірусні бази даних і запуск програми для виявлення і видалення всіх копій вірусу;
- поширення антивірусного приписи в локальній мережі користувача і потім по всій мережі. З часів появи електронної пошти з'явилася система реклами (часто небажано) під назвою спам-розсилки того чи іншого товару або послуги, аналог паперової реклами безкоштовно розповсюджується по поштових скриньках житлових будинків. В даний час становить 70-90% від усіх поштових повідомлень.

Механізми боротьби зі спамом:

- чорні списки відправників;
- сірі списки, що вимагають повторного звернення поштового сервера для відправки;
- контекстні фільтри. Так само розроблена Байєсова фільтрація спаму – метод для фільтрації спаму, заснований на застосуванні байєсівського класифікатора, в основі якого лежить застосування теореми Байєса.

Переваги: простота і зручність реалізації, ефективність. Принциповий недолік: метод базується на припущенні, що одні слова частіше зустрічаються в спам, а інші – в звичайних листах, і неефективний, якщо це припущення невірно. Існує метод Байесови отруєння, що дозволяє додати багато зайвого тексту, іноді ретельно підібраного, щоб «обдурити» фільтр. Чи не принциповий недолік - метод працює тільки з текстом.

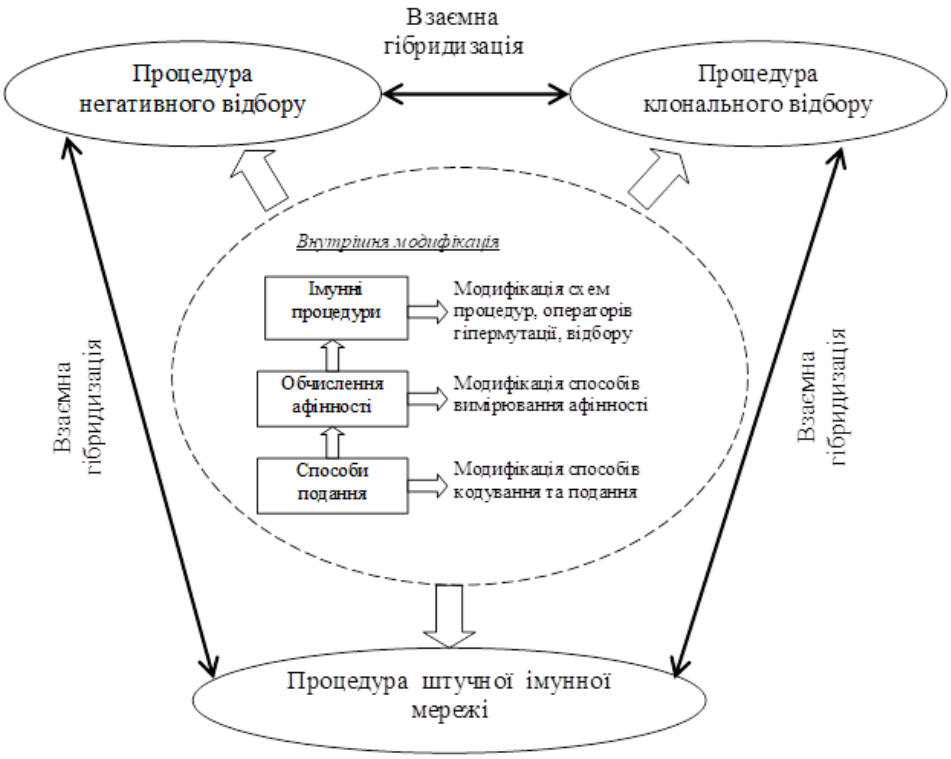

Рисунок 1.6 – Типи гібридизації ШНМ

Одним з наслідків впровадження засобів боротьби зі спамом стала ймовірність «помилково позитивного» рішення щодо спаму, тобто частина листів, які не є спамом, стала позначатися як спам. У разі агресивної антиспам-політики (знищення листів, які здаються спамом, в автоматичному режимі без повідомлення відправника / одержувача) це призводить до труднообнаруживаемим проблем з проходженням пошти. Для подолання зазначених недоліків існуючих методів боротьби зі спамом була реалізована штучна імунна система для боротьби зі спамом на основі моделі негативного відбору і відповідне програмне забезпечення. Існуюче програмне забезпечення для роботи з ШНМ дозволяє вирішувати тільки вузьке коло завдань, для вирішення яких вона розроблена, що не охоплюючи всю сферу застосування моделі негативного відбору.

Рисунок 1.7 – Модель імунної системи

Формулюючи основи своєї теорії, було введено поняття формальних та функціональних мереж. Формальні мережі служать вивчення питань репертуару, дуалізму і супресії. При розгляді функціональних мереж представлена кількісна картина теорії. Даний підхід максимально математизований і, в основному, пов'язаний з описом фазових переходів. Надалі площина фазових змінних системи рівнянь, що розглядається, була розділена на докритичну область, область переходу і посткритичну область.

2 ОБҐРУНТУВАННЯ РОБОТИ РОЗРОБЛЕНОХ МОДЕЛІ ШІС

Враховуючи необхідність отримати якомога більше розповсюджений серед користувачів застосунок, логічним є рішення обрати платформу, яка цьому найбільш сприятиме. В такому випадку логічним є обрати платформу, яка буде сумісна, з якомога більшою кількістю можливих систем. Такими в сучасному світі безсумнівно є вебзастосунки.

Серед усіх мов програмування найбільш пристосованими до розробки вебзастосунків можна назвати наступні: JavaScript, PHP, Python. Аналізуючи переваги вищеназваних мов програмування, спираючись на потреби, було обрано мову програмування JavaScript (а саме похідну від нього TypeScript), за наступні переваги: можливість використовувати одну і ту саму мову програмування як на клієнтські, так і на серверній стороні застосунку;

JavaScript працює у всіх сучасних браузерах.

Під час розробки програмного забезпечення також ключовими є такі фактори як сумісність технологій, їх вартість (оскільки у випадку їх високої вартості підтримка проєкту може потребувати надмірної кількості ресурсів). Також важливими є навички якими володіє команда розробки, оскільки на опанування нових для неї технологій може піти значний відрізок часу.

Тож в результаті, враховуючи усі вище вказані параметри, для розробки проєкту були обрані наступні технології: мова програмування TypeScript:

- клієнтський фреймворк React;
- серверний фреймворк NodeJs;
- серверний фреймворк NestJs;
- система збірки проєкту Nx;
- реляційна база даних PostgreSQL;
- мова запитів даних і маніпуляцій для API GraphQL;
- об'єктно-реляційне відображення TypeORM;

- інструментарій для управління ізольованими Linux-контейнерами Docker.

2.1 Огляд технологій розробки

Головна ідея мови програмування TypeScript це надати типізації мові програмування JavaScript, таким чином значно зменшивши кількість помилок, які роблять розробники. Це також має зробити простішою роботу над великими проектами. Ця мова це синтаксична надбудова над мовою JavaScript, в якій присутні такі речі, як: оголошення типів, інтерфейсів, анотування типів, змінних та інших синтаксичних одиниць. Також TypeScript надає можливість підтвердити тип виразу. Компілятор мови програмування TypeScript працює таким чином, що на виході компіляції ми отримуємо код мовою JavaScript стандарту EcmaScript 5. Ось ще декілька особливостей TypeScript [12]:

- типи з мови програмування TypeScript не залишають після себе жодних слідів вже в відкомпільованому коді, тобто перевірка на відповідність типів відбувається тільки на етапі розробки, і вже по його закінченні типи не «навантажують» код застосунку;
- наявна така річ, як «виведення типу», що означає те, що програміст може вказувати не усі типи. Невідомі типи можуть бути обраховані на основі попередніх. Це не тільки надає можливість робити менше помилок розробнику, але й поліпшує продуктивність коду після компіляції.

TypeScript має динамічну типізацію, тобто деякі типи можуть бути введені не статично React це JavaScript фреймворк (англ. Framework, каркас, платформа – інфраструктура, що полегшує розробку складних систем [13]). React був створений інженерами компанії Facebook. Він був створений для розв'язання проблем, які виникають в застосунках зі складними інтерфейсами, в яких є необхідність зберігати дані, які до того ж мають змінюватись с часом. React має широкий набір інструментів, який надає

можливість як початківцю, так і спеціалісту у сфері створення вебінтерфейсів, створювати односторінкові застосунки.

React використовує Flux архітектуру, яка була розроблена спеціально для цього фреймворку. Ця концепція створена для уникнення ситуації, коли архітектура застосунку може перейти у багатонапрямний потік даних. Це досягається завдяки тому, що React влаштований таким чином, що дані в ньому йдуть по колу, тобто зміна даних приводить до зміни стану, а зміна стану призводить до зміни даних., а вводитись під час роботи.

NodeJs це не лише фреймворк для JavaScript, насправді це середовище в якому виконується JavaScript. Він побудований на рушії V8, який належить корпорації Google. Таким чином він надає можливість виконати JavaScript не лише в окремому браузері, з усіма обмеженнями які це накладає, натомість це дає можливість виконати JavaScript-код на будь-якій платформі, на яку можливо встановити NodeJs. NodeJs був представлений спільноті розробників на конференції JSConf у 2009 році Райаном Далом і змінив цим уявлення про JavaScript. NodeJs найбільш всього підходить для програм, які мають обробляти велику кількість одночасних підключень.

NestJs або просто Nest це фреймворк для створення серверних застосунків на платформі NodeJs. Він написаний на TypeScript та рекомендованою мовою також є TypeScript, хоч він підтримує розробку і на JavaScript. Насправді Nest це лише абстракція, «прошарок» над звичайними серверними фреймворками для NodeJs такими як Express та Fastify. По замовчування він якраз працює на Express. Розробка застосунків на NestJs не змінюється в залежності від фреймворку який слугує йому у якості основи, оскільки він надає однаковий інтерфейс в усіх випадках. Це і є філософією NestJs, і це дійсно робить всі написані на ньому застосунки дуже схожими, в чого є багато переваг, при цьому він не накладає ніяких обмежень на те, які застосунки ви хочете на ньому писати, він лише спрощує процес написання.

Nx це швидка система збірки з підтримкою монорепозиторіїв (англ. monorepo, стратегія програмування, коли код багатьох проєктів зберігається в

одному репозиторії). Nx це інструмент для створення підтримки та розширення модульної структури проєкту. Використовуючи nx можна налаштувати його лише один раз і використовувати на всіх етапах розробки саме у такому вигляді, при чому весь цей час питання того де, та як, створити новий модуль вже матимуть відповідь.

Рисунок 2.1 – Огляд технологій розробки

PostgreSQL це реляційна база даних з відкритим вихідним кодом. Не зважаючи на те, що вона безплатна і розвивається волонтерами з усього світу, вона утримує конкурентоспроможність, яка ні чим не поступається таким власницьким базам даних як Sybase, Oracle, та DB2. PostgreSQL як пропонує усі стандартні для реляційної бази даних можливості, так і пропонує багато нових. Серед них створення нових фундаментальних типів, використання структур індексування, що, наприклад прискорює роботу з геометричними типами даних. Також PostgreSQL в деякому сенсі надає

можливість зберігати дані як в нереляційній базі даних, так, наприклад, є можливість зберігати дані у форматі JSON (JavaScript Object Notation - Нотація об'єктів JavaScript). PostgreSQL побудований на основі клієнт-серверної моделі, що означає, що можливо створити власний клієнт для PostgreSQL. На стороні сервера можливо додати власні процедурні мови.

GraphQL це мова запитів до веб-API, а також одночасно середовище для виконання запитів і концепція яка це все обумовлює. Щоб запропонувати користувачам (тим які взаємодіють з відкритим API застосунку, або розробникам клієнтської частини застосунку) API GraphQL, розробник визначає модель, в якій вказує сутності, та операції, які можуть бути проведені над ними. Взаємодія з таким API відбувається наступним чином: користувач (клієнт) запитує в сервера певну множину даних, описуючи необхідні йому дані у запиті за допомогою спеціального синтаксису GraphQL, і у відповідь отримує лише ті дані, що запитав. В першу чергу GraphQL вирізняється швидкістю своєї роботи (завдяки економії трафіку), та зручності використання. Найголовнішою його перевагою як раз і є цей механізм повернення лише потрібної інформації, оскільки завдяки цьому часом можна зекономити як на довжині відповіді, так і на кількості запитів. Також оскільки GraphQL вимагає досить скрупульозного описання схеми, то більшість його реалізацій також наділені можливістю автоматично генерувати документацію.

TypeORM це ORM (англ. Object-Relational Mapping, об’єктно-реляційне відображення, технологія програмування, задачею якої є пов’язати реляційну базу даних з концепціями об'єктноорієнтованих мов програмування) а також бібліотека для TypeScript, яка надає можливість поєднати TypeScript-код з базою даних найпростішим способом. TypeORM надає можливість створювати, модифікувати, видаляти та проводити інші зміни з базами даних не пишучи при цьому SQL-коду (Structured Query Language - Мова структурованих запитів), повністю, або частково при цьому абстрагуючись від обраної бази даних.

TypeORM надає наступні можливості:

- створювати класи, які будуть собою уособлювати певні таблиці в базах даних;
- описувати поля для цих класів, таким чином окреслюючи колонки таблиць;
- створювати зв’язки різних видів, спеціалізуючи значення вище означених полів;
- наслідувати класи-таблиці, таким чином значно економлячи об'єм вихідного коду;
- автоматично генерувати і робити міграції;
- ще багато різноманітних можливостей.

2.2 Аналіз технології Docker

Docker – це контейнерно-віртуалізаційна технологія. Тобто, це дуже легковага технологія для створення і роботи з легковагими віртуальними машинами. На додачу до цього Docker це також дуже зручний і сучасний розробницький інструмент, який дозволяє створювати, змінювати, та навіть ділитись з іншими розробниками, образами операційних систем разом з необхідним програмним інструментарієм. Досягається це за допомогою того, що цим образом є лише невеликий (зазвичай) файл, в якому описується яка необхідна операційна система, які налаштування для неї, а також які мають бути встановлені програми та з якими налаштуваннями. Це рішення дозволяє дуже просто ділитись цими образами. Також оскільки більшість з таких образів є у вільному доступі, то інші розробники можуть їх використовувати у якості джерела для своїх власних образів.

Найважливіша проблема, яку Docker створений розв'язувати, це надмірна вимогливість до ресурсів від звичайних віртуальних машин. Те що вони використовують значну кількість ресурсів навіть сучасних персональних комп’ютерів можна легко пояснити тим, що їм для роботи

треба бути запущеними на пристроях з вже інстальованими операційними системами, куди треба встановити програми для розгортання VM (Virtual Machine - Віртуальна машина), і вже в розгорнутій VM відкрити потрібний застосунок. Docker створений розв'язати цю проблему в тих випадках, коли розробнику необхідно швидко розгорнути певну програму, а також зберегти цю можливість для інших членів команди. Також Docker значно полегшує тест програми на різних платформах.

2.3 Процес проектування ШІС

Проектування цифрових інтегральних схем (ЧІС) – це складний процес, що включає кілька основних етапів. Ось загальний огляд цього процесу:

- визначення вимог і специфікацій: Першим кроком є визначення вимог до схеми, включаючи функціональність, продуктивність, енергоспоживання, вартість, розмір і інші параметри. Це етап, на якому визначаються ключові технічні характеристики майбутньої схеми;

- розробка архітектури: Нна цьому етапі розробляється загальна структура ЧІС, визначаються основні компоненти і їх взаємодія. Це включає вибір топології схеми, типів транзисторів, розташування компонентів тощо;

- логічне проектування: логічне проектування включає створення функціональних блоків і їх взаємодії на логічному рівні. Включає створення логічних схем, використання мов опису апаратури (наприклад, VHDL або Verilog) для опису поведінки схеми;

- синтез і оптимізація: логічна схема перетворюється в фізичну форму, яка може бути реалізована в кремнії. На цьому етапі проводиться оптимізація за різними параметрами, такими як швидкість, площа, енергоспоживання;

- фізичне проектування: на цьому етапі схема розміщується і маршрутизується на кристалі. Це включає розміщення компонентів на фізичному рівні, прокладку з'єднань між ними, врахування обмежень за сигналом і енергоспоживанням;

- перевірка і верифікація: проводиться перевірка правильності роботи схеми за допомогою симуляцій і верифікаційних методів. Включає тестування на різних рівнях абстракції, включаючи логічну і фізичну перевірку;

- виготовлення і тестування: після завершення проектування схема передається на виготовлення в кремнієвому вигляді. Після виготовлення проводиться тестування реальної ЧІС на відповідність специфікаціям;

- виведення на ринок: якщо всі етапи успішно пройдені, готова ЧІС може бути випущена на ринок і інтегрована в кінцеві продукти.

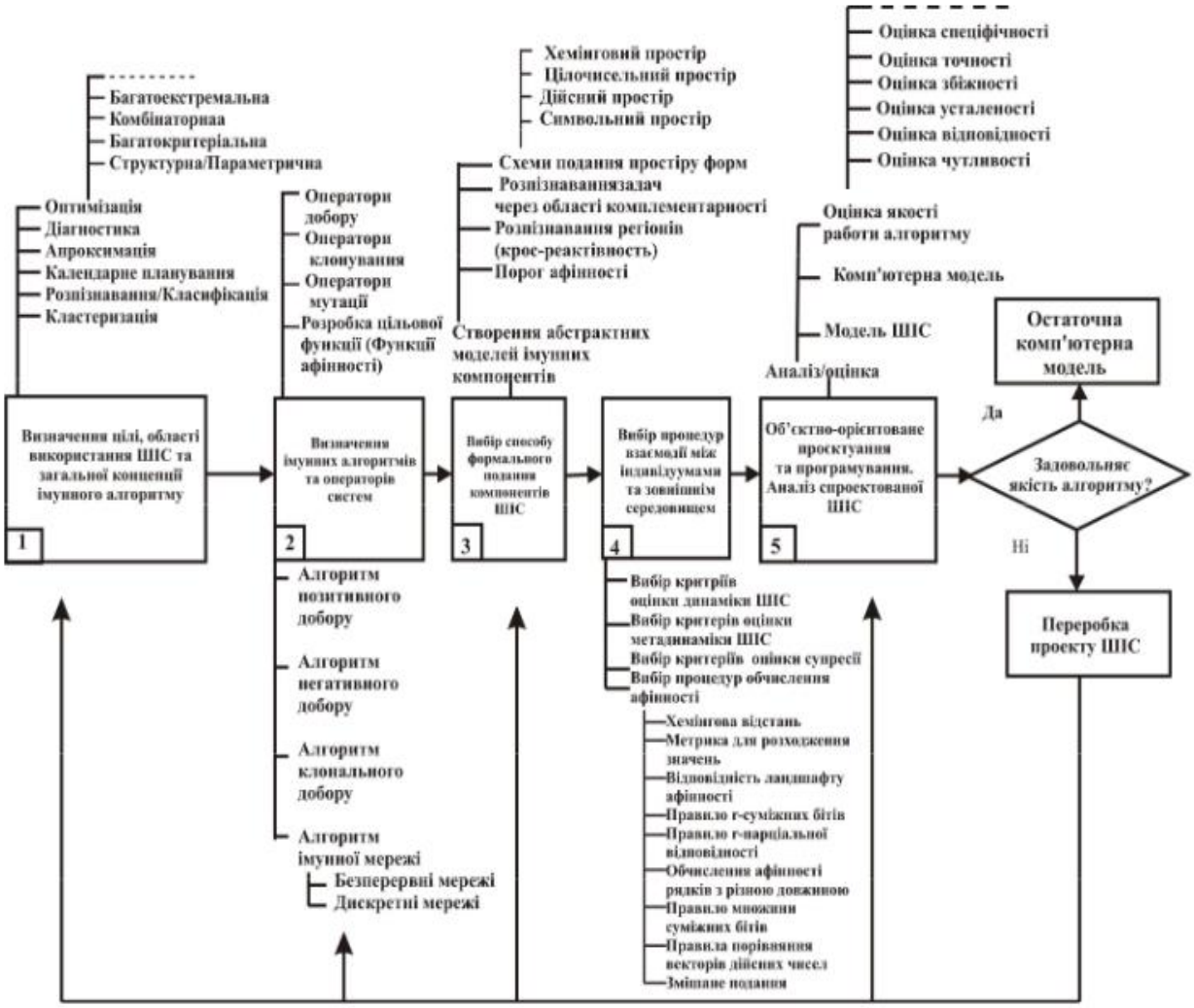

Рисунок 2.2 – Процес проектування й оцінки ефективності ШІС

Кожен з цих етапів вимагає спеціальних інструментів і технологій, а також глибоких знань і досвіду в області електроніки і комп'ютерного проектування.

Процес формування деревовидної штучної імунної мережі починається з визначення аффінностей між антитілами (рисунок 2.3)

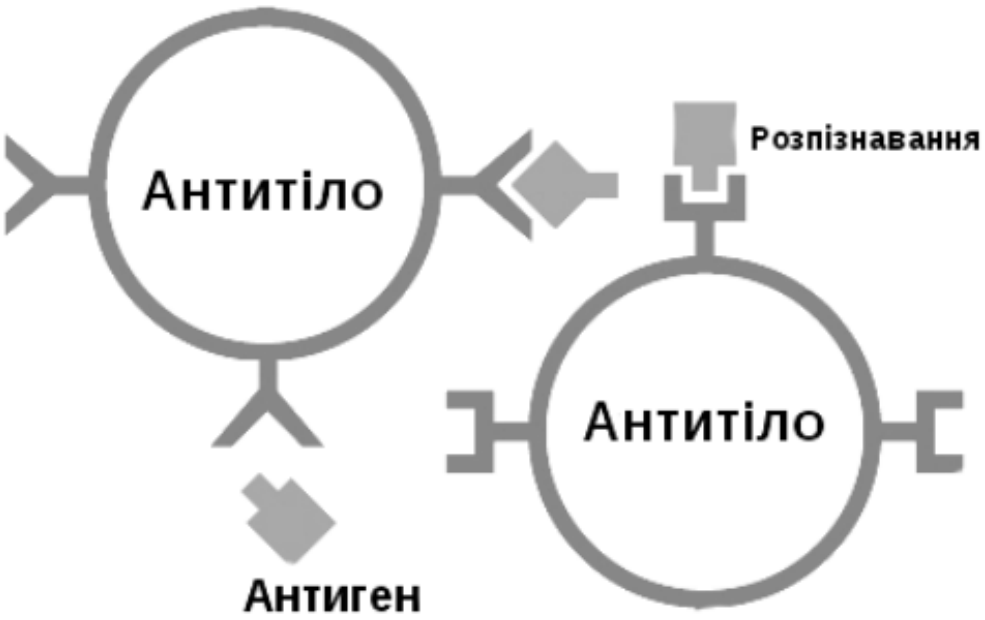

Рисунок 2.3 – Модель ШІС. Принцип негативного відбору

2.4 Схема роботи імунної мережі

Модель негативного відбору є однією з найвідоміших моделей ШІС. Її принцип роботи полягає у генерації детекторів, що відповідають аномальній поведінці досліджуваного об'єкта. Метод негативного відбору складається з двох фаз: навчання та класифікації. Під час навчання на вхід моделі надходять "свої" антигени. При цьому в ШІС виробляються випадкові клітини, які стають детекторами лише в тому випадку, якщо вони не реагують на жодний з антигенів, а також на вже існуючі імунні клітини (антитіла).

$$DaiNet(AB,k,c) = {}^{PRP}\begin{bmatrix} Scaling(AB)\rightarrow \\ Presentation(AB)\rightarrow \\ NATCalculation(AB) \end{bmatrix} \Rightarrow$$

$$\Rightarrow {}^{DKN}\begin{bmatrix} DKnetCreation(AB,k)\rightarrow \\ CalcStimulation(AB)\rightarrow \\ CentersSelection(AB,c,NAT)\rightarrow \\ DendricClustering(AB') \end{bmatrix} \Rightarrow$$

$$\Rightarrow {}^{NET}\begin{bmatrix} Cloning(AB'',CL)\rightarrow \\ Mutation(CL)\rightarrow \\ Presentation(CL,AB',AB'')\rightarrow \\ CLSupression(CL,AB',AB'')\rightarrow \\ NetSupression(CL,AB'')\rightarrow \\ AvCalculation(AB'')\rightarrow \\ ClusterSelection(AB'') \end{bmatrix},$$

Рисунок 2.4 – Модель ШІС

Етапи загальної схеми роботи моделі деревовидної імунної мережі при кластеризації даних для обробки зображень представлені на рисунку 2.4.

3 РЕАЛІЗАЦІЯ ПРОГРАМНИХ ЗАСОБІВ

3.1 Реалізація алгоритмів

Загалом виокремлюються чотири типи запозичень вихідного коду (кожен наступний наслідує від попереднього його можливості):

- повне копіювання без змін (окрім, наприклад, зміни коментарів);
- «неінтелектуальні» зміни, тобто зміни для яких не потрібно знати яку саме задачу виконує програма. До таких можна віднести: зміну назв змінних та констант, додавання та видалення пробільних символів, зміна типів даних на схожі, та інші прості зміни які дозволяє мова програмування;
- додавання та видалення елементів вихідного коду, яке не несе за собою вагомої зміни у роботі програми. Зміна порядку розташування та запуску елементів вихідного коду. Не значне редагування вихідного коду;
- значне редагування програми, головним критерієм того, що це все ще є запозиченням, це збереження логіки та алгоритмів, які були використані в оригіналі.

Саме зараз важливо зазначити: жодна з сучасних систем визначення схожості вихідного коду не може ефективно розв'язувати дві наступні проблеми:

- ситуації, коли перевірки потребують фрагменти вихідного коду надзвичайно малої величини, так, наприклад навіть людині буде важко у звичайних умовах визначити ступінь схожості двох реалізацій алгоритму сортування бульбашкою, реалізованих двома учнями;
- редагування типу, який був вище описаний під номером чотири. Така ситуація на відміну від описаної вище виглядає як та, яка може бути вирішена, однак її вирішення має сумнівну цінність. По -перше, слід повернутись до питання розглянутого на самому початку цієї роботи: «Що є запозиченням? І схожість тексту якої міри має ним вважатись?». Оскільки

значне редагування вихідного коду якоюсь мірою вже є новизною, яку вніс автор, а також це редагування може бути пов’язане з покращенням яких-небудь параметрів як вихідного коду, так і роботи програми, то в певних випадках запозичення може бути якщо не виправдане, то вважатись таким, рівень якого допустимий. По-друге, навіть розв'язання цього питання може поставити іншу проблему. Оскільки таке прискіпливе виокремлювання подібності може призвести до збільшення кількості помилок такого типу, коли дві ніяк не пов’язані роботи визначаються як ті, що мають високий рівень схожості, хоча саме в цьому випадку нас цікавить саме вірогідність запозичення одного автора в іншого.

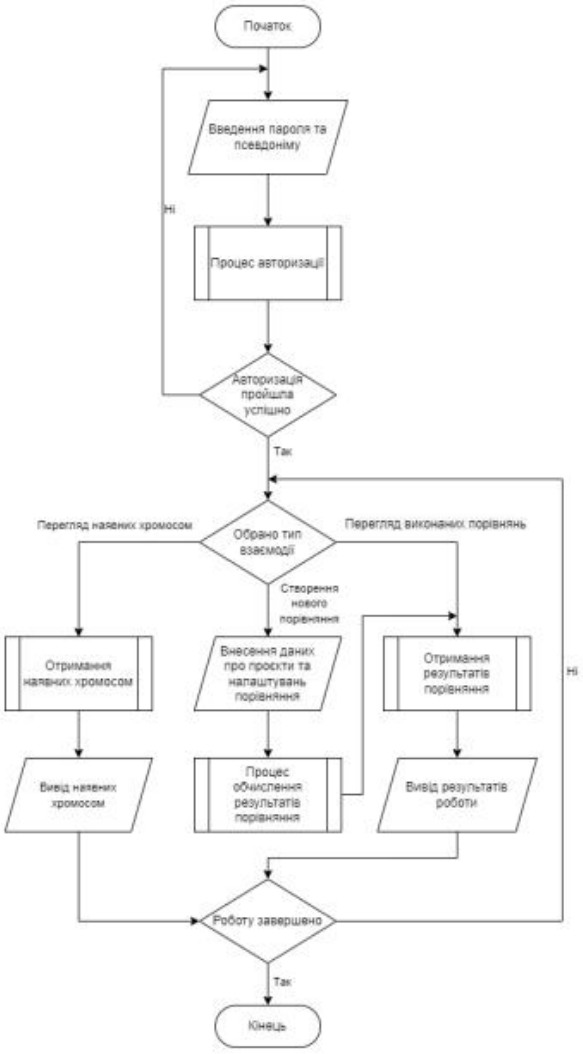

Рисунок 3.1 – Блок-схема роботи системи

Тепер перейдемо до методів порівняння програм, серед них зазвичай виокремлюють основні п’ять:

- «Text-based» – цей метод полягає у визначені текстових метрик, при цьому в відриві від того, чим цей текст є, найкращим прикладом методу такого типу є алгоритм Левенштейна. Цей спосіб дуже простий. Очевидно, що такий метод ефективний при невеликій відсотковій зміні тексту програми, і навпаки вразливий до навіть найпростіших методів приховання плагіату;

- «Token-based» – методи, які розбивають текст на так звані «токени», а потім різним чином аналізують отримані дані. Цей метод вже більш стійкий ніж попередній. Однак звісно він все ще дуже чутливий до вагомих реформацій вихідного коду;

- Metric-based» – методи цього виду зазвичай аналізують дані отримані завдяки описаним вище. Суть цих методів це обрахування кількості синтаксичних одиниць в коді, тобто може, наприклад бути підраховано кількість циклів або змінних у вихідному коді. Даний метод більш надійний ніж попередні, але все ще досить просто може бути обманутий, наприклад додаванням або видаленням певних елементів вихідного коду;

- «Tree-based» – цей метод полягає у створені AST-дерева (Abstract Syntax Tree – Абстрактне синтаксичне дерево) для вихідного коду, а також порівняння отриманих AST-дерев будь-якими способами. Даний метод вважається досить точним, але і складним в реалізації;

- «Binary-based» – методи цього виду потребують компіляції вихідного коду, що одразу накладає на них обмеження того, що вони не можуть працювати з певними мовами програмування. Однак перевагами методів цього типу є те, що вони можуть повністю ігнорувати такі зміни як зміна назв констант або яких-небудь інших синтаксичних одиниць. Також для певних мов програмування актуальним також буде те, що завдяки оптимізації компілятора, ще деякі зміни будуть проігноровані. Однак очевидним є недолік таких методів. Для кожної мови програмування потрібно буде мати

компілятор, особливі проблеми можуть виникнути з мовами які мають багато різноманітних діалектів.

Звісно найточніше значення схожості двох вихідних кодів може дати гібридний метод, тобто той, в якому будуть використані усі переваги вище описаних методів. Але в такому разі виникає логічне питання, а як слід трактувати різні значення схожості отримані від різних методів? Зазвичай обирають серед двох варіантів - це середнє та максимальне значення.

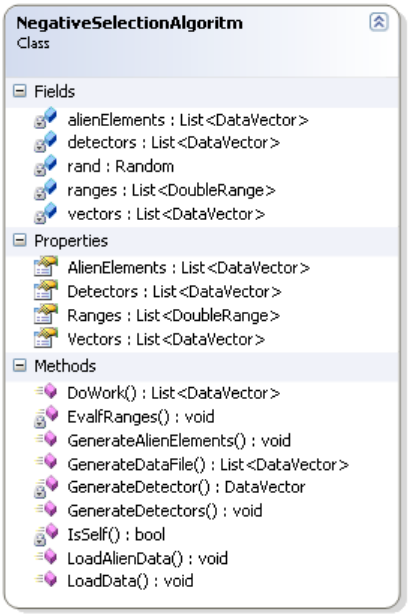

Рисунок 3.2 – Класи об’єктно-орієнтованої моделі ШІС

І саме тут доцільно використати генетичний алгоритм, який би мав певні початкові значення, по яких би від початку і визначав загальний відсоток подібності текстів. А надалі спираючись на помічених користувачем недоліків визначені цього відсотку, він може адаптуватись до певного роду робіт, і до певних авторів.

Тож серед обраних методів та алгоритмів, які слугуватимуть функціями для визначення параметрів подібності можна перерахувати наступні: у якості методів типу «Text-based» були обрані алгоритм Дамерау-Левенштейна та бібліотека string-mismatch, яка реалізує алгоритм «Greedy»; у якості методу

типу «Token-based» був обраний коефіцієнт Жаккара; задля реалізації методу типу «Tree-based» була використана бібліотека uglify-js, завдяки якій було отримано AST-дерево вихідного коду; у якості методу типу «Metric-based» був розроблений метод countReservedKeywords, який маючи дані про зарезервовані слова в мові програмування TypeScript обраховує кількість синтаксичних одиниць кожного типу використаних у вихідному коді.

Метод countReservedKeywords працює наступним чином: спочатку видаляються усі символи, що не є літерами латинського алфавіту, окрім пробільних символів; текст розділяється на масив строк, таким чином, що кожне слово розділене з іншими пробільним символом стає елементом масиву.

3.2 Використання ШІС

Задля спрощення розуміння концепції роботи генетичного алгоритму серед користувачів розроблюваної системи прийнято рішення назвати вище описану сутність «robot». Також в рамках цієї роботи ця сутність може бути згадана як хромосома генетичного алгоритму.

Процес обрахування загального ступеня схожості досить простий. Сутність robot є набором ключ-значення, в якому ключем є назва певного методу обрахування ступеня схожості текстів, значенням якого є коефіцієнт, на який має бути помножений результат роботи цього методу над обраними проєктами.

Таким чином, кожен користувач має бути забезпечений принаймні одним екземпляром сутності robot, яка і буде використана під час обрахування загального ступеня схожості. Прийнято рішення брати за базове значення кожного елемента (гена) хромосоми величину визначену за формулою *1/N*, де N – це кількість методів визначення запозичень, які використовує система. Таким чином, перше використання (до зміни хромосоми завдяки генетичному алгоритму) цієї сутності буде мати такі ж

результати, як ніби загальний ступінь схожості був визначений як середнє серед усіх методів використаних в гібридному. Також прийнято рішення надати користувачу можливість «заморозити» значення цієї хромосоми. Таким чином, якщо користувач використає цю можливість на новоствореній хромосомі, то кінцевий відсоток запозичення завжди буде визначатись як середнє значення результатів усіх використаних методів.

Також це надає користувачу наступну можливість: якщо користувач визначив, що якийсь з методів особливо якісно визначає схожість проектів, які він вніс, він може в сутності robot поставити значення для цього методу рівним одиниці, а для всіх інших нуль. Такі зміни призведуть до того, що в наступних порівняннях, в яких він обере саме цю хромосому, загальне значення схожості буде рівне результату роботи того методу, якому він поставив одиницю в хромосомі.

Як вже було сказано раніше, дані отримані завдяки різним методам визначення схожості текстів зводяться до одного значення завдяки знанню поточного стану сутності robots. Зміна цього стану якраз відбувається на запит користувача, в якому він вказує такий параметр, як «правильна імовірність плагіату», який він визначає на власний розсуд спираючись на ті дані, які надає система для аналізу, або, в крайньому разі, власноруч провівши аналіз проєктів наданих на перевірку.

Була розроблена проста реалізація генетичного алгоритму, який і забезпечує еволюцію тіла сутності robots. Важливо зазначити, що для цього алгоритму хромосомою є екземпляр сутності robot, а генами відповідно ті коефіцієнти, які ця сутність зберігає відповідно до кожного методу визначення схожості, який існує у системі.

Робота генетичного алгоритму відбувається наступним чином:

Визначення дельта-значення, тобто різниці в абсолютних величинах між очікуваним користувачем значенням схожості та тим, яке отримується після вирахування загального значення схожості за допомогою поточного

стану сутності robot та n (кількість може бути змінена, базовим значенням є чотири попередніх стани) попередніх її станів;

Після попереднього пункту ми отримуємо n-ну кількість хромосом та дельта-значень, які собою уособлюють те, наскільки дана хромосома відповідає очікуванням користувача. В разі, якщо якесь з дельта-значень дорівнює нулю, хромосома яка йому відповідає визначається, як та, що є задовільною, і алгоритм на цьому завершується, хромосома повертаєтьсяяк результат роботи алгоритму (ця хромосома стає новим значенням сутності robot, а попереднє значення записується в таблицю попередніх станів сутності robot). У разі, якщо існує лише один стан цього екземпляру сутності robot (таке можливо, якщо сутність жодного разу не була змінена ні вручну, ні за допомогою генетичного алгоритму) проводиться мутація (тобто для таких випадків наступний пункт цього списку ігнорується).

Відбір батьків відбувається методом рулетки, при цьому імовірність хромосоми стати такою вимірюється тим наскільки мале її дельта - значення в порівнянні з іншими хромосомами. Після відбору батьків відбувається схрещування. Схрещування відбувається таким чином, що сусідні хромосоми по колу діляться генами, при цьому розділення відбувається по лінії кросовера (вона може бути налаштована, за замовчуванням, це кількість генів поділена два).

Усі отримані хромосоми з певною імовірністю проходять мутації, для одного з генів. Число, яке відповідає цьому гену зменшується або збільшується на випадкову величину. Як можна спостерігати під час мутації багато дій мають певну імовірність, ця імовірність визначається константними значеннями встановленими в системі.

3.3 Програмна реалізація

Згідно з архітектурою, яку пропонує використовувати Nx, в директорії apps зберігаються компоненти загального застосунку, які мають бути

запущені окремо один від одного. В директорії libs зберігаються бібліотеки, сутності з яких можуть використовуватись в усіх компонентах програми. Таке розділення вихідного коду дозволяє дуже гнучко і зручно пере використовувати вихідний код застосунку.

На рисунку 3.3 можна спостерігати директорію libs/common/artificial-intelligence, яка вміщає розроблений генетичний алгоритм, а також деякі допоміжні функції. Також можна побачити директорію libs/common/source-codes-comparing-methods, яка вміщає розроблений метод знаходження схожості вихідних кодів програм, а також функції обгортки для інших методів, які були взяті зі сторонніх бібліотек.

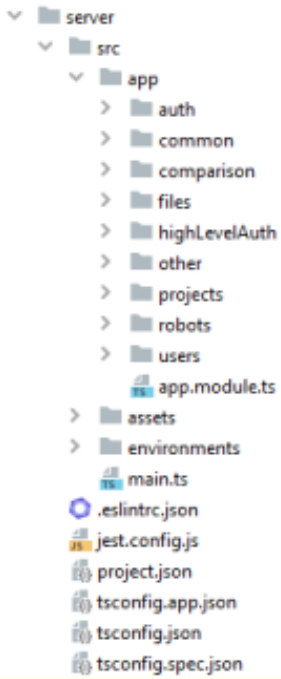

Рисунок 3.3 – Структура

3.4 Використання БД

Оскільки при розробці застосунку було використано бібліотеку TypeORM, створення бази даних та додавання таблиць та обмежень відбувається шляхом написання класів та додавання до них декораторів.

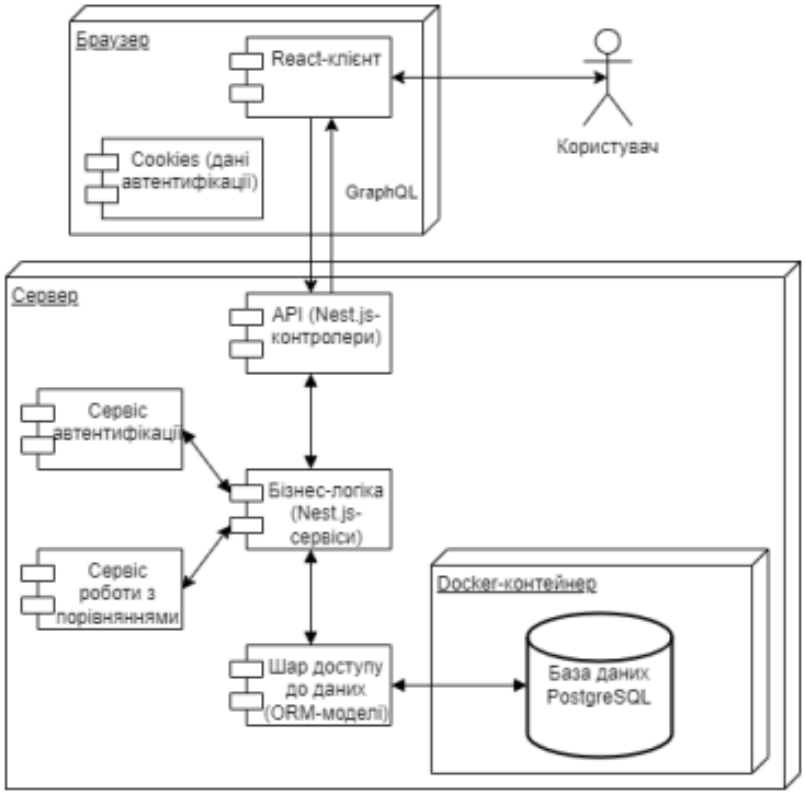

Рисунок 3.4 – БД

Роботу клієнт-серверного застосунку, основними перевагами якого є можливості, які надає саме серверна частина обумовлює серверний застосунок, та його інтерфейси.

Якщо СУБД (система управління базами даних) повертає певну помилку (наприклад про те, що вже є запис з такою електронною поштою), то сервер поверне помилку з повідомленням про те, що операція не завершилась успішно. В іншому випадку від СУБД повернеться екземпляр створеної сутності користувача, який буде відправлений у відповідь на запит клієнтського застосунку.

Спочатку програма перевіряє чи взагалі існує користувач з такою поштою. Далі програма бере хеш-код від пароля, який ввів користувач, та порівнює його з хеш-кодом, який був збережений в базі даних. У разі якщо хеш- коди паролів не будуть рівні сервер поверне відповідну помилку. Якщо хеш- коди будуть рівні, це буде означати, що користувач автентифікований.

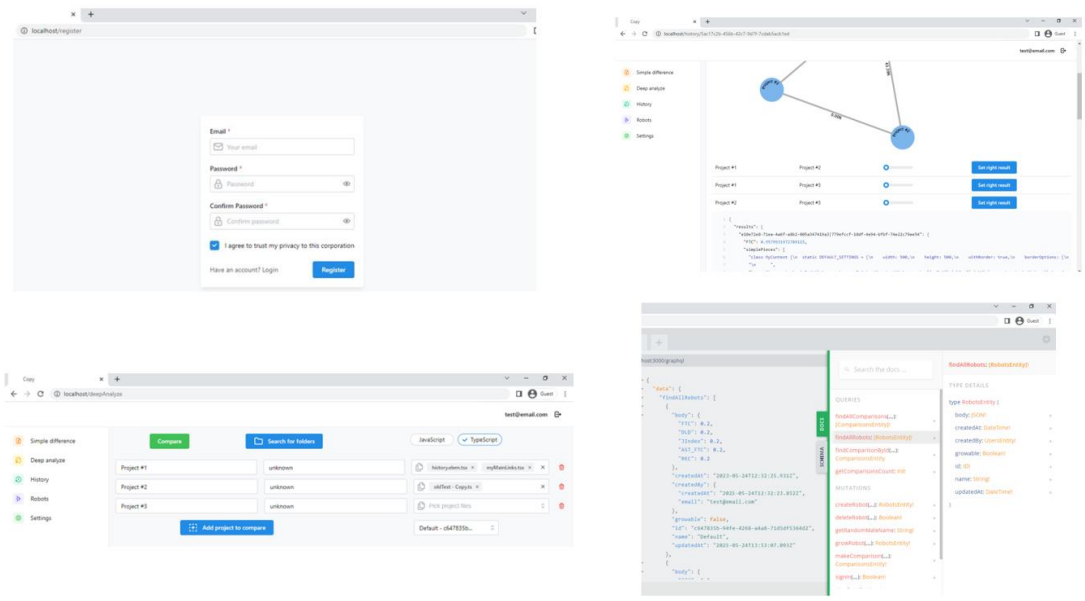

Рисунок 3.5 – Результати роботи

Результати представлені на рисунку 3.5. Проілюстровано сторінку, на якій користувач може переглянути, змінити та створити нові екземпляри сутності robots, прапор «growable», визначає чи є тіло цієї сутності статичним, тобто чи можуть бути його параметри оновлені шляхом еволюції на сторінці отримання результатів порівняння.

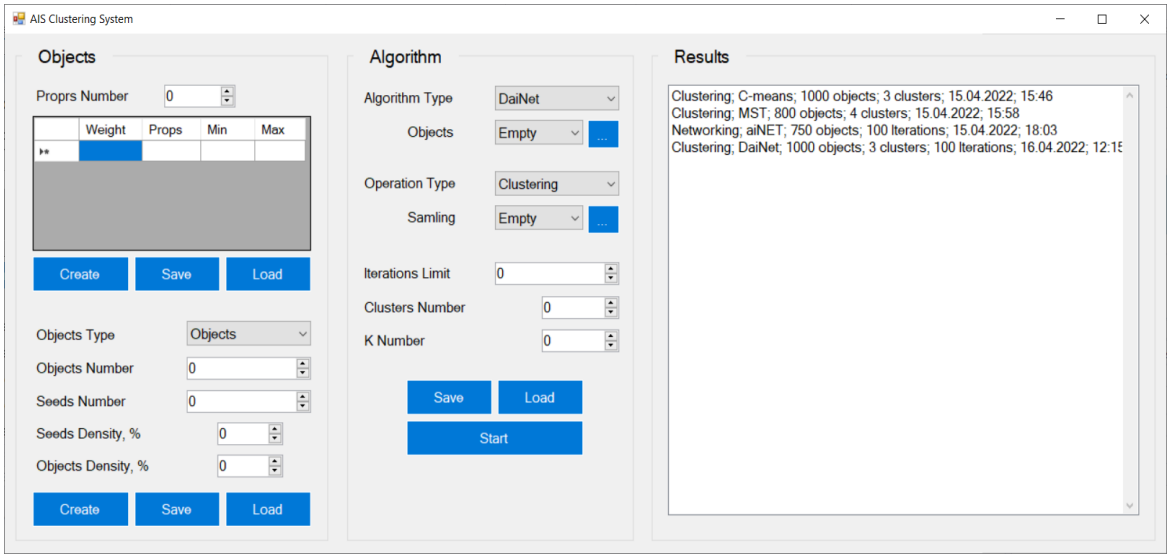

Рисунок 3.6 – Результати роботи моделі

Для моделювання алгоритмів кластеризації було розроблено застосунок за допомогою мови C#, платформи .NET та патерну Model-View-Controller. Цей застосунок дозволяє обирати на налаштовувати алгоритм кластеризації, формувати набори об’єктів та досліджувати процес їх роботи.

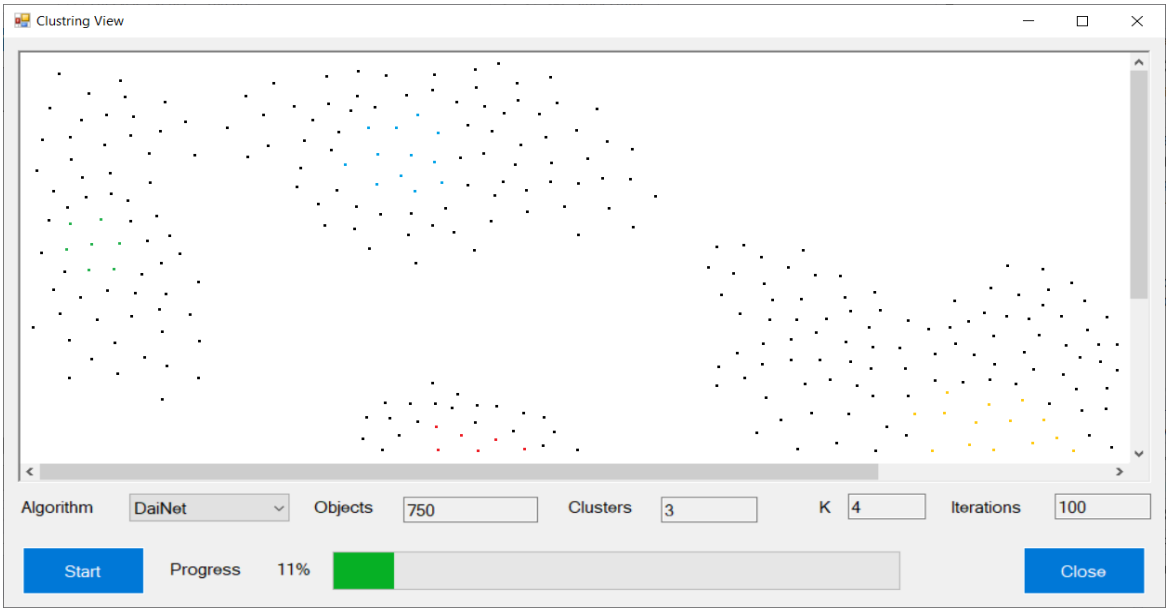

Рисунок 3.7 – Результати роботи моделі

Також користувач може отримати доступ до відкритого API системи. На цій сторінці представлена вичерпна документація на всі доступні користувачу методи якими він може взаємодіяти з системою. Оскільки серверна частина взаємодіє з клієнтською за допомогою технології GraphQL, то документація також представлена у вигляді схеми даних, яку користувач може використовувати для створення власних запитів. Дослідження роботи методу Dendric-aiNet відбувалося у порівнянні із обраними еталонними методами кластеризації. У таблиці на слайді використовуються два умовних позначення: параметр T або Time характеризує швидкість кластериації, а параметр A або Accuracy характеризує точність кластеризації. Відповідно до цього найбільшою швидкістю характеризується метод MST, але запропонований метод Dendric-aiNet поступається йому на 10-15% за цим показником.

Алгоритм		Набір 1	Набір 2	Набір 3
MST	T	38%	36%	39%
	A	88%	85%	82%
C-means	T	72%	74%	72%
	A	100 %	100 %	100 %
Clonalg	T	100 %	100 %	100 %
	A	80%	83%	81%
aiNet	T	98%	95%	93%
	A	52%	50%	50%
Dendric-aiNet	T	48%	46%	47%
	A	95%	93%	96%

Рисунок 3.8 – Аналіз отриманих результатів

Найбільшою точністю характеризується метод C-means, але запропонований метод Dendric-aiNet поступається йому лише не 5-6%. Таким чином, запропонований метод відрізняється достатньою збалансованістю порівняно в іншими методами кластеризації.

ВИСНОВКИ

Проведено аналіз існуючих методів та моделей штучної імунної системи для обробки зображень. Проведено аналіз найбільш поширених класичних методів та моделей штучних імунних систем для кластеризації даних. Виділено набір універсальних імунних операторів та проаналізовано можливості їх модіфікації задля підвищення якості кластеризації. Сформовано модель деревовидної штучної імунної мережі та алгоритм Dendric-aiNet для кластеризації даних. Розроблено програмний застосунок для модулювання алгоритмів кластеризації

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бурда А.С., Прудіус М.А., Стефанюк Я.Г., Фомічов О.О. Методи обробки та інтелектуального аналізу даних з використанням штучних імунних систем // Системи управління, навігації та зв'язку. 2024. № 3. С.100-103.

2. D. Dasgupta (Ed.) Artificial Immune Systems and Their Applications. // Springer, 1999.

3. L.N. de Castro and J. Timmis Artificial Immune Systems: A New Computational Intelligence Approach // Springer, 2002.

4. A.O.Tarakanov, V.A. Skormin and S.P. Sokolova Immunocomputing: Principles and Applications //Springer, 2003.

5. The Online Home of Artificial Immune Systems www.artificial-immune-systems.org

6. L.N. de Castro and J. Timmis Artificial Immune Systems: A New Computational Intelligence Approach // Springer, 2002.

7. L. N. de Castro and F. J. Von Zuben Learning and Optimization Using the Clonal Selection Principle // IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems, vol. 6, n. 3, pp. 239-251, 2002.

8. L. N. de Castro and F. J. Von Zuben The Clonal Selection Algorithm with Engineering Applications // Proceedings of GECCO'00, Workshop on Artificial Immune Systems and Their Applications, Las Vegas, USA, July, pp. 36-37, 2000.

9. E. Clark, A. Hone and J. Timmis A Markov Chain Model of the B-Cell Algorithm // ICARIS 2005, Springer-Verlag, LNCS 3627, pp. 318 – 330, 2005.

10. L.N. de Castro and J. Timmis .Artificial Immune Systems: A New Computational Intelligence Approach // Springer, 2002.

11. D.Dasgupta A comparison of negative and positive selection algorithms in novel pattern detection.