МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет "Запорізька політехніка"

Субботін С. О., Олійник А. О., Федорченко Є.М., Рудь М. С. Фердман П.М.,
Філатов С.О., Федорченко Ю.В., Харченко А.С.

МАТЕМАТИЧНІ ТА ПРОГРАМНІ ЗАСОБИ ДЛЯ ПРИЙНЯТТЯ РІШЕНЬ, РОЗПІЗНАВАННЯ ОБРАЗІВ Й ІНТЕЛЕКТУАЛЬНОГО ДІАГНОСТУВАННЯ

Монографія

Запоріжжя
2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет “Запорізька політехніка”

Субботін С. О., Олійник А. О., Федорченко Є.М.,
Рудь М. С. Фердман П.М., Філатов С.О.,
Федорченко Ю.В., Харченко А.С.

МАТЕМАТИЧНІ ТА ПРОГРАМНІ ЗАСОБИ ДЛЯ ПРИЙНЯТТЯ РІШЕНЬ, РОЗПІЗНАВАННЯ ОБРАЗІВ Й ІНТЕЛЕКТУАЛЬНОГО ДІАГНОСТУВАННЯ

Монографія
під загальною редакцією
д. т. н., проф. С. О. Субботіна

Видання підготовлено в межах науково-дослідних робіт «Методи і засоби прийняття рішень для оброблення даних в інтелектуальних системах розпізнавання образів» (№ держ. реєстрації – 0117U003920) та «Інтелектуальні методи та програмні засоби діагностування й неруйнівного контролю якості техніки військового та цивільного призначення» (номер держ. реєстрації – 0119U100360) за часткової підтримки міжнародних проєктів програми Еразмус+ Європейського Союзу “Internet of Things: Emerging Curriculum for Industry and Human Applications” (ALIOT, Ref. no. 573818-EPP-1-2016-1-UK-EPPKA2-CBHE-JP) та “Innovative Multidisciplinary Curriculum in Artificial Implants for Bio-Engineering BSc/MSc Degrees” (BIOART, Ref. no. 586114-EPP-1-2017-1-ES-EPPKA2-CBHE-JP)

Запоріжжя
2020

УДК 004.891.3:004.855.5
М 34

Рекомендовано до друку Вченою радою
Національного університету "Запорізька політехніка"
(протокол №4/20 від 02.03.2020 року)

Колектив авторів:
Субботін С. О., д. т. н., проф. (розділи 1, 3, 4)
Олійник А. О., к. т. н., доц. (розділи 1 – 4)
Федорченко Є.М. (розділи 1 – 4)
Рудь М. С. (розділ 2)
Фердман П.М. (розділ 3)
*Філатов С.О. (*розділ 4)
Федорченко Ю.В. (розділ 1)
Харченко А.С. (розділ 3)

Рецензенти:

Чопоров С.В., професор кафедри програмної інженерії Запорізького національного університету, доктор технічних наук, професор.

Литвиненко В.І., доктор технічних наук, професор, завідувач кафедри інформатики та комп'ютерних наук Херсонського національного технічного університету.

М 34 **Математичні та програмні засоби для прийняття рішень, розпізнавання образів й інтелектуального діагностування**: монографія / С. О. Субботін, А. О. Олійник, Є. М. Федорченко та ін. ; під заг. ред. С. О. Субботіна. – Запоріжжя : Національний університет "Запорізька політехніка", 2020. – 271 с.

ISBN 978–617–529–273–0

Монографія містить аналіз та дослідження методів, моделей та програмних засобів розпізнавання образів. Наведено вирішення задач розпізнавання об'єктів сцени автотранспортних засобів. Удосконалено та досліджено методи вирішення задачі пошуку оптимального шляху для автоматизації процесу прийняття рішень. Вирішено завдання оптимізації та прогнозування економічних показників та медико-екологічного моделювання на основі методів та засобів інтелектуального комп'ютингу. Розглянуті методи можуть використовуватися при вирішенні практичних завдань створення інтелектуальних систем прийняття рішень та аналізу даних різної природи.

Видання призначено для наукових співробітників, аспірантів, студентів комп'ютерних спеціальностей закладів вищої освіти, а також може використовуватися педагогічними працівниками та практичними фахівцями.

УДК 004.891.3:004.855.5

ISBN 978–617–529–273–0

ПЕРЕЛІК СКОРОЧЕНЬ

ALH – Adaptive LED headlight (Адаптивні світлодіодні фари);
API – Application programming interface (прикладний програмний інтерфейс);
BIC – Bayesian information criterion (інформаційний критерії Байеса);
BSM – Blind Spot Monitoring (система контролю «сліпих зон»)
DAA – Driver attention alert (система контролю втомленості водія);
ETL – Extract transform load (модуль завантаження, обробки та зберігання даних);
GLM – Generalized linear model (узагальнена лінійна модель);
IDE – Integrated development environment (інтегроване середовище розробки);
KNN – *k*-nearest neighbor method (Метод *k* найближчих сусідів);
LDWS – Lane departure warning system (система попередження про виїзд за межі полоси руху);
LSTM – Long short-term memory (довга короткочасна пам’ять);
MAE – Mean absolute error (середнє абсолютне відхилення);
MLP – Multilayer perceptron (багатошаровий перцептрон);
MSE – Mean squared error (середнє квадратичне відхилення);
OLS – Ordinary least squares (метод найменших квадратів);
PSO – Particle swarm optimization (метод рою часток);
R^2 – Determination coefficient (коефіцієнт детермінації);
SAE – Society of automotive engineers (організація автомобільних інженерів);
SCBS – Smart city brake support (система безпечного гальмування в місті з функцією розпізнавання пішоходів та машин);

SSIS	–	SQL server integration services (Служби інтеграції SQL Server);
SVM	–	Support vector machine (метод опорних векторів);
UI	–	User interface (інтерфейс користувача);
БД	–	база даних;
ГА	–	генетичний алгоритм;
ГДВ	–	гранично допустимі викиди;
ГДК	–	гранично допустима концентрація;
ДТП	–	дорожньо-транспортна пригода;
ЕА	–	еволюційний алгоритм;
ОМ	–	острівна модель;
ООП	–	об'єктно-орієнтоване програмування;
ПЗ	–	програмне забезпечення;
СЗД	–	система збереження даних;
СУБД	–	система управління базами даних;
ТУВ	–	тимчасово узгоджені нормативи викидів;
ЧПУ	–	числове програмне управління;
ШНМ	–	штучна нейронна мережа.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОСИЛАНЬ 3

ВСТУП 9

РОЗДІЛ 1 МЕТОДИ РОЗПІЗНАВАННЯ ГРАФІЧНИХ ОБРАЗІВ ТА АВТОТРАНСПОРТНИХ ЗАСОБІВ 12

1.1 Задача розпізнавання графічних образів 13

1.2 Дослідження інформаційних систем розпізнавання образів 15

1.2.1 Mazda Smart City Brake Support 15

1.2.2 Автопілот Tesla 18

1.2.3 Waymo 21

1.2.4 Розпізнавання машин Blippar 23

1.2.5 Mobileye 24

1.2.6 Проєкт розпізнавання рукописних цифр набору даних MNIST за допомогою бібліотеки Keras 33

1.2.7 Проєкт розпізнавання об'єктів на зображеннях за допомогою бібліотеки Keras 34

1.2.8 Проєкт підбіру параметрів при навчанні повнозв'язної нейронної мережі 35

1.3 Порівняння підходів реалізації розпізнавання образів 37

1.4 Дослідження реалізації процесу розпізнавання образів методом Віоли-Джонса з використанням бібліотеки OpenCV 38

1.4.1 Підготовка до навчання каскадного класифікатора 38

1.4.2 Навчання каскадного класифікатора 40

1.5 Дослідження реалізації процесу розпізнавання образів за допомогою нейронних мереж з використанням бібліотеки Keras 42

1.5.1 Навчання нейронних мереж 42

1.5.2 Використання простого генетичного алгоритму в навчанні нейронних мереж з використанням бази зображень CIFAR-100 44

1.5.3 Використання простого генетичного алгоритму в навчанні нейронних мереж з використанням власної бази зображень 46

1.6 Розроблення генетичних методів розв'язання задачі розпізнавання графічних образів 47

1.6.1 Модифікація простого генетичного алгоритму Альфа-Бета 47
1.6.2 Модифікація простого генетичного алгоритму Альфа-Бета фіксована 51
1.6.3 Модифікація простого генетичного алгоритму Фіксована 52
1.7 Програмне забезпечення для розпізнавання графічних образів 53
1.7.1 Модулі розробленого програмного забезпечення 56
1.7.2 Характеристики програми 59
1.7.3 Вхідні та вихідні дані 60
1.7.4 Повідомлення 62
1.7.5 Розпізнавання учасників дорожнього руху за допомогою нейронної мережі 65
1.7.6 Призначення і умови виконання програми 65
1.7.7 Виконання програми 66
1.8 Експерименти і результати 70
1.8.1 Експерименти по дослідженню розроблених генетичних методів розв'язання задачі розпізнавання графічних образів 70
1.8.2 Результати роботи методів Віоли-Джонса та гістограм орієнтованих градієнтів 74
1.8.3 Результати навчання каскадного класифікатора за методом Віоли–Джонса 74
1.8.4 Результати навчання нейронних мереж 76
1.8.5 Результати розпізнавання за допомогою навченої нейронної мережі 77
1.9 Висновки за розділом 1 81
1.10 Література до розділу 1 82

РОЗДІЛ 2 МЕТОДИ ВИРІШЕННЯ ЗАДАЧІ ПОШУКУ ОПТИМАЛЬНОГО ШЛЯХУ ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСУ ПРИЙНЯТТЯ РІШЕНЬ 87

2.1 Постановка проблеми 87
2.2 Дослідження методів еволюційного пошуку для розв'язання задачі комівояжера 92
2.3 Модифікації генетичних методів пошуку оптимального шляху 96
2.3.1 Модифікації простого генетичного алгоритму 96
2.3.2 Змішаний метод 96

2.3.3 Змішаний оптимальний метод 97
2.3.4 Розумний метод 97
2.3.5 Метод абсолютного оптимуму 98
2.3.6 Випадковий метод вибору 98
2.3.7 Модифікація методу активації 99
2.3.8 Метод оптимізації розташування ліків 100
2.4 Програмне забезпечення вирішення задачі пошуку оптимального шляху 101
2.4.1 Розроблення інформаційних моделей програмного забезпечення пошуку оптимального шляху 101
2.4.2 Модулі розробленого програмного забезпечення 110
2.4.3 Сутності бази даних програмної системи 117
2.4.4 Алгоритм функціонування програми 117
2.4.5 Виконання програми 119
2.5 Експерименти і результати 124
2.6 Висновки за розділом 2 128
2.7 Література до розділу 2 130

РОЗДІЛ 3 ОПТИМІЗАЦІЯ ТА ПРОГНОЗУВАННЯ ФІНАНСОВИХ ПОКАЗНИКІВ ПІДПРИЄМСТВ ТОРГІВЛІ 134

3.1 Постановка завдання 135
3.2 Аналіз літературних даних та постановка проблеми 136
3.3 Розробка модифікації оператора мутації при розв'язанні задачі оптимізації фінансових показників аптек 141
3.4 Програмне забезпечення для прогнозування фінансових показників аптек 146
3.4.1 Технологічні рішення для оптимізації 147
3.4.2 Структурна схема програми 148
3.4.3 Опис модулів розроблюваної системи 148
3.4.4 Звертання до програми (файл проєкту) 153
3.4.5 Вхідні і вихідні дані 154
3.4.6 Виконання програми 154
3.5 Експерименти та результати методів ініціалізації початкової популяції еволюційного алгоритму 158
3.6 Висновки за розділом 3 176

3.7 Література до розділу 3 _____ 177

РОЗДІЛ 4 МОДЕЛЮВАННЯ ЗАЛЕЖНОСТІ ПОКАЗНИКІВ ЗДОРОВ'Я ВІД ОБСЯГІВ ВИКИДІВ ЗАБРУДНЮЮЧИХ РЕЧОВИН У ПОВІТРЯ _____ 181

4.1 Моделювання процесу забруднення атмосферного повітря _____ 182

4.1.1 Види та джерела забруднення повітря _____ 182

4.1.2 Показники впливу рівня забруднення атмосферного повітря на здоров'я людини _____ 187

4.1.3 Аналіз рівню та динаміки викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел по регіонам України _____ 189

4.1.4 Теплова мапа рівня викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел_____ 191

4.1.5 Постановка завдання _____ 193

4.2 Побудова моделей залежності показників здоров'я від обсягів викидів забруднюючих речовин у повітря_____ 195

4.3 Синтез нейромережевих моделей показників захворюваності населення 229

4.4 Програмне забезпечення для прогнозування залежності захворюваності від обсягів викидів_____ 244

4.5 Висновки за розділом 4_____ 261

4.6 Література до розділу 4 _____ 262

ВСТУП

Завдання розпізнавання образів відноситься до класу важко формалізованих завдань і в даний час є особливо актуальним у зв'язку з необхідністю автоматизації образних процесів комунікації (візуальних, мовних) в інтелектуальних системах. Тому до цього часу продовжується розробка і програмна реалізація ефективних методів передачі розпізнавальної функції людини комп'ютеризованим інтелектуальним системам. Для вирішення завдань цього класу широкого розповсюдження набули методи та технології на основі штучних нейронних мереж та стохастичної оптимізації, зокрема еволюційні та генетичні алгоритми.

У цій монографії розглядаються питання розроблення та дослідження методів і програмних засобів оброблення великих даних в системах діагностування та розпізнавання образів.

Перший розділ присвячено дослідженню та розв’язанню завдання розпізнавання учасників дорожнього руху (автомобілів, велосипедів, пішоходів, мотоциклів, вантажівок). Розпізнавання учасників дорожнього руху – практичне застосування теорії розпізнавання образів, яке пов’язано з необхідністю автоматичної локалізації учасників дорожнього руху на наявній фотографії.

У **другому розділі** розглянуто методи розв’язання транспортної задачі та задачі складської логістики на основі квазіінтелектуальних методів оптимізації, зокрема методу, який базується на генетичних алгоритмах. На сьогодні характерною особливістю переходу людства до інформаційного суспільства є надзвичайно швидкі трансформаційні зміни поколінь технологій, споживчих стандартів, ринків виробництва та збуту товарів. Тому в сучасних умовах підприємства, які займають те або інше місце на економічному ринку праці, являють собою динамічні, нестаціонарні системи. Нині ефективність функціонування більшості підприємств визначається рівнем застосування логістики. Логістика посідає значне місце під час перебудови механізмів господарювання в сучасних ринкових умовах.

У **третьому розділі** досліджено питання пошуку оптимального асортименту для мережевих підприємств (зокрема, мережевих аптек). Оптимізація асортименту призведе до більш ефективного використання площі аптек, зменшення незадоволеного попиту та, в

кінцевому результаті, до підвищення конкурентноздатності підприємства за рахунок зменшення роздрібної вартості товарів шляхом зниження видатків на зберігання та обслуговування неоптимально завантажених площин підприємства.

В ході роботи було розроблено генетичний метод багатокритеріальної оптимізації з модифікацією оператора мутації для дослідження ступеню впливу факторів на фінансові показники аптек та вибору моделі оптимізації. Досліджено проблеми та існуючі методи оптимізації фінансових показників мережевих аптек, розроблено генетичний метод з модифікацією оператора мутації для вирішення проблеми управління асортиментом аптечної продукції, заснований на еволюційних методах. .

Четвертий розділ присвячено моделюванню залежності показників здоров'я від обсягів викидів забруднюючих речовин на основі інтелектуальних обчислень. Досліджено залежність показників захворюваності населення хворобами системи кровообігу, туберкульозом та онкологічними хворобами від обсягів викидів забруднюючих речовин в атмосферне повітря в результаті діяльності стаціонарних джерел забруднення в різних регіонах України. Розроблено програмне забезпечення для дослідження залежності показників захворюваності від обсягів викидів забруднюючих речовин. Це забезпечить можливість прогнозування показників захворюваності, що може бути використано при плануванні регіональних бюджетів в розділі витрат на охорону здоров'я, медикаментозне забезпечення населення, здійснення заходів щодо покращення екологічного становища.

Монографію підготовлено як результат науково-дослідних робіт «Методи і засоби прийняття рішень для оброблення даних в інтелектуальних системах розпізнавання образів (№ державної реєстрації – 0117U003920), «Інтелектуальні методи та програмні засоби діагностування й неруйнівного контролю якості техніки військового та цивільного призначення» (номер державної реєстрації – 0119U100360), що виконуються на кафедрі програмних засобів Національного університету "Запорізька політехніка", за часткової підтримки міжнародних проєктів програми Еразмус+ Європейського Союзу "Internet of Things: Emerging Curriculum for Industry and Human Applications" (ALIOT, Ref. no. 573818-EPP-1-2016-1-UK-EPPKA2-

CBHE-JP) та “Innovative Multidisciplinary Curriculum in Artificial Implants for Bio-Engineering BSc/MSc Degrees” (BIOART, Ref. no. 586114-EPP-1-2017-1-ES-EPPKA2-CBHE-JP).

Монографія містить результати оригінальних досліджень авторів і може бути рекомендована для широкого кола прикладних фахівців в області комп’ютерних наук та інформаційних технологій, технічного діагностування, розпізнавання образів. Видання орієнтоване на наукових співробітників, аспірантів, студентів комп’ютерних спеціальностей закладів вищої освіти, а також може використовуватися педагогічними працівниками та практичними фахівцями.

РОЗДІЛ 1
РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ ТА ІДЕНТИФІКАЦІЯ АВТОТРАНСПОРТНИХ ЗАСОБІВ

Кількість областей, в яких застосовується розпізнавання образів, постійно зростає. Це пов'язано з розвитком технологій, методів, програмних рішень, бібліотек, обчислювальної техніки, а також з необхідністю автоматизації чи контролю процесів без участі людини. Одною з таких областей є дорожня безпека, що включає багато вузьких напрямків: розпізнавання номерних знаків, визначення завантаженості вулиць, розпізнавання учасників руху в безпілотному транспорті та інше.

Однак, на сьогодні, повністю не вирішені проблеми якості розпізнавання. Наприклад, в системі Autopilot Tesla було виявлено вразливість – система у більшості випадків не може розпізнати велосипед, вона визначає велосипед як людину чи невеликий автомобіль, відповідно, система може прийняти рішення внаслідок яких можлива загроза життю велосипедиста [1]. Також відомі вразливості в розпізнаванні учасників дорожнього руху в таких системах як Mazda Smart City Brake Support, автопілот Waymo. Тому, дослідження розв'язання задачі розпізнавання велосипедів та інших учасників дорожнього руху є достатньо актуальною задачею, оскільки може бути використано в системі контролю та/або забезпечення безпеки дорожнього руху чи для дослідження актуальності створення спеціалізованих велосипедних доріжок, тощо.

Метою розділу є визначення оптимального підходу до розв'язання задачі розпізнавання учасників дорожнього руху, який надасть найкращу точность розпізнавання та його програмна реалізація.

Для досягнення поставленої мети були визначені такі завдання:

– проаналізувати існуючі підходи до розвязання задачі розпізнавання образів;

– проаналізувати існуючі системи що містять реалізацію розпізнавання учасників дорожнього руху;

– розробити модифікації генетичного алгоритму;

– використати розроблені модифікації генетичного алгоритму та існуючі підходи для розпізнавання учасників дорожнього руху;

– провести аналіз отриманих результатів та зробити висновки щодо кращого підходу.

Запропоновано модифікацію генетичного алгоритму, використання якої, покликане оптимізувати підбір параметрів при навчанні нейронних мереж, для розв'язання задачі розпізнавання образів..

1.1 Задача розпізнавання графічних образів

Розпізнавання образів належить до області інформаційних технологій та пов'язано з необхідністю ідентифікації об'єктів за певними ознаками. Необхідність в розпізнаванні виникає в багатьох областях – від систем безпеки та медичних систем до розпізнавання штрих-кодів.

Ознака – певна характеристика об'єкта, значення якої або вона сама властива саме цьому об'єкту. Ознаки, спільні для декількох об'єктів, не несуть корисної інформації та не розглядаються як ознаки в задачі розпізнавання.

Ключовим аспектом в задачі розпізнавання є виявлення ознак об'єктів в ході навчання. Зазвичай навчання здійснюється шляхом представлення зображень, з окремими об'єктами з вказанням їх приналежності певному класу. Як результат навчання, розпізнавальна система повинна набути здатності надавати однакову відповідь на вхідні об'єкти, що належать одному класу та відмінну – на об'єкти інших класів. Важливо відмітити, що кожен об'єкт повинен бути наведений в навчальній вибірці у якнайбільш різноманітних видах, це збільшить вірогідність точного розпізнавання, адже невеликі відмінності об'єкта який потрібно розпізнати, від зображень об'єкта за якими проводилося навчання може призвести до неспроможності розпізнавальної системи вірно розпізнати цей об'єкт. Наприклад, будь-яка літера, може бути по-різному написана. Також, можлива ситуація коли навіть об'єкти одного класу можуть достатньо сильно відрізнятися один від іншого.

Класичним випадком умови задачі розпізнавання образів зазвичай є наступна: для множини вхідних об'єктів, яка складається з підмножин – класів, необхідно провести класифікацію – встановити клас об'єкту, якщо задана інформація про класи, є описи всієї множини та інформації про об'єкт [2]. Таке формулювання задачі

більш характерне для нейронних мереж. Деякі методи розпізнавання базуються не на множині класів, а на лише двох класах, наприклад, необхідно визначити, чи відноситься вхідний об'єкт до класу "велосипед" або ні (метод Віоли-Джонса).

Як правило, при реалізації розпізнавання образів виконується обробка вхідного зображення – переведення з кольорового до чорно-білого. Це спрощує процес розпізнавання, бо зображення в такому випадку можна розглядати як функцію на площині [2]. Тобто, для множини точок площини T, де функція $f(x,y)$ у кожній точці зображення описує характеристику – яскравість, прозорість, то така функція є формальним записом зображення. Тоді множина всіх функцій $f(x,y)$ площини T є моделлю множини всіх зображень X [2].

Розрізняють три групи методів розпізнавання образів:

– порівняння зі зразком: методи найближчого середнього та найближчого сусіда;

– статистичні методи: метод байєсівського вирішуючого правила;

– нейронні мережі: персептрон, когнітрон, згорткова нейронна мережа, мережа Хопфілда, тощо.

Однією з багатьох областей де розпізнавання образів знайшло застосування - це системи контролю за дорожньою безпекою, а в останній час активно розвивається і використовується в системах "автопілот" багатьма компаніями (наприклад, Tesla, Mazda, Waymo, BMW, Toyota, Ford).

Дослідження вирішення проблеми розпізнавання учасників дорожнього руху є актуальною задачею, оскільки досі не реалізовано системи здатної безпомлково здійснювати розпізнавання - автопілот Tesla має помилки в розпізнаванні вантажівок, велосипедів та дорожньої розмітки, в тому числі нездатний детектувати статичні транспортні засоби [1], система Mazda Smart City Brake Support нездатна розпізнавати велосипеди та мотоцикли, реалізовано часткове розпізнавання пішоходів[3], автопілот Waymo має проблеми з визначенням місцезнаходження та розпізнаванням об'єктів (стовпи, світлофори, тощо) [4].

Зазвичай, системи що реалізують розпізнавання учасників дорожнього руху, використовують камери змонтовані в салоні та напрямлені на дорогу, допоміжні сенсори, радари, лідари, та певний

бортовий комп’ютер здатний опрацьовувати всю інформацію що надходить та відображати повідомлення користувачу.

В даних системах зображення що надходить з камери, зазвичай попередньо оброблюється (відбувається переведення зображення з кольорового простору до чорно-білого) та далі за допомогою навченої нейронної мережі відбувається розпізнавання обєктів на зображенні та віднесення їх до окремих класів (наприклад, «авто», «вантажівка», «мотоцикл», тощо).

Беручи до уваги той факт, що популярним та поширеним рішенням для вирішення задачі розпізнавання образів є навчання та використання нейронних, було розглянуто низку інформаційних систем, що використовують нейронні мережі. Також, було проведено власне дослідження навчання та застосування нейронних мереж для розпізнавання образів з використанням генетичного алгоритму для оптимізації результатів навчання нейронних мереж.

1.2 Аналіз систем розпізнавання транспортних засобів

Сучасний автомобільний транспорт стає все більш комп’ютеризованим. Автомобільна промисловість впроваджує все більше нових технологій, що дозволяє користувачам полегшити своє життя [5].

Безліч компаній займаються розробками розумних автомобілів. Розумний автомобіль здатен орієнтуватися використовуючи різні прилади і методи, такі як радар, лідар, GPS, одометри, і комп'ютерний зір. Сучасні системи керування здатні інтерпретувати інформацію з сенсорів аби визначити правильний напрямок руху, а також ідентифікувати перешкоди і відповідні покажчики. Розумні автомобілі мають системи керування які здатні аналізувати сенсорні дані і розрізняти інші транспортні засоби на дорозі, що є дуже корисним при плануванні маршруту до бажаної точки призначення [5-18].

Переваги використання розумного автомобіля очевидні:

– підвищення безпеки на дорогах зважаючи на виключення основного джерела аварій – людського фактора;

– взаємодіючи між собою і оточенням, розумнні автомобілі зможуть прокладати оптимальний маршрут, що дозволить скоротити затори;

– економія для бізнесу на оплату роботи водіїв;

– додатковий час, яке раніше витрачалося на керування автомобілями [5-18].

1.2.1 Mazda Smart City Brake Support

Як стверджує японська автобудівна компанія Mazda, безпека водія та пасажирів є головним пріорітетом компанії [5]. Метою безпеки Mazda є розширення «куту зору» водія та надання інформації та захист від неочікуваних перешкод, як візуальної, так і звукової. Тож, певні моделі Mazda, а саме: Mazda 3, Mazda 6, Mazda CX-3, Mazda CX-5, Mazda CX-9, мають вбудований набір технологій безпеки i-ACTIVSENSE, для раннього виявлення потенційних загроз, швидкого реагування, та допомоги водію в керувані авто [5].

До набору i-ACTIVSENSE входять такі системи:

– система адаптивного освітлення LED фар головного світу (ALH – Adaptive LED Headlight): покращує нічну видимість та допомогає водію уникнути небезпечних ситуацій;

– система безпечного гальмування в місті з функцією розпізнавання пішоходів та машин (SCBS – Smart City Brake Support): помічає машини та пішоходів перед авто та автоматично включає гальма;

– система контролю «сліпих зон» (BSM – Blind Spot Monitoring): попереджує водія про наявність машин, що рухаються обабіч та ззаду в тому ж напрямку, але й про наявність перешкод на шляху авто при русі заднім ходом;

– система попередження про виїзд за межі полоси руху (LDWS – Lane Departure Warning System): зчитує розмітку з поверхні дороги та попереджує водія про ненавмисне, без включення вказівника повороту, зміщення зі своєї полоси дороги;

– система контролю втомленості водія (DAA – Driver Attention Alert): використовує інформацію про кут повороту руля, поточну швидкість, дані з камери направленої на дорогу, для оцінки стану водія та допомогти уникнути аварії, до якої може призвести втомленість та брак уваги, та пропонує зробити перерву [5, 6].

Функціонування систем забезпечується таким обладнанням: міліхвильовим радаром (76 ГГц), камерою, інфрачервоним лазером ближньої дистанції, квазі–міліхвильовим радаром (24 ГГц), зображено на рисунку 1.1 [6].

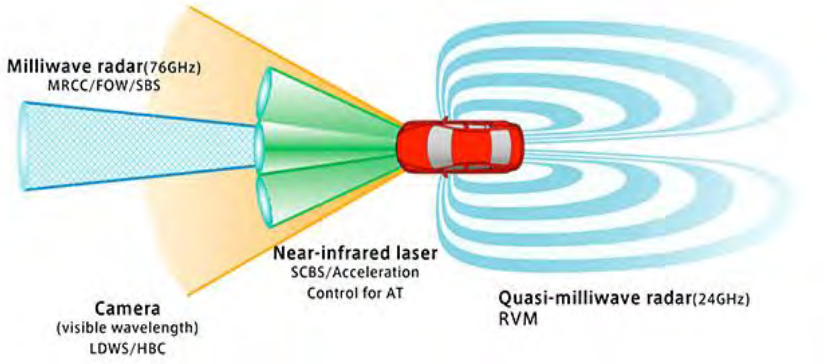


Рисунок 1.1 – Обладнання i-ACTIVSENSE [6]

Система безпечного гальмування в місті з функцією розпізнавання пішоходів та машин (SCBS) здатна розпізнавати автомобілі та пішоходів за рахунок направленої на дорогу камери. Принцип роботи приведено на рисунку 1.2 [3].

При виявленні близької перешкоди система SCBS система попереджує водія звуковим сигналом та світовим індикатором про небезпеку, накопичує тиск в гальмівній системі, обираючи вільний хід педалі тормозу, щоб допомогти водію здійснити екстренне гальмування, а коли небезпека зникне, скине зайвий тиск. Якщо водій не реагує на загрозу аварії та не здійснює гальмування чи не виконує маневри самостійно (не повертає кермо) система SCBS задіює гальмівні механізми автоматично [3].

Відстань реагування системи становить 100 метрів, а верхній ліміт швидкості, за якої система здатна зреагувати на перешкоду залежить від моделі. Для моделей Mazda 2, Mazda CX-3 та Mazda CX-9 такий ліміт становить 30 км/год, а для Mazda 3, Mazda 6 та Mazda CX-5 – 80 км/год. При цьому автоматичне гальмування для уникнення зіткнення з перешкодою можливо в діапазоні швидкості до 30 км/год [3].

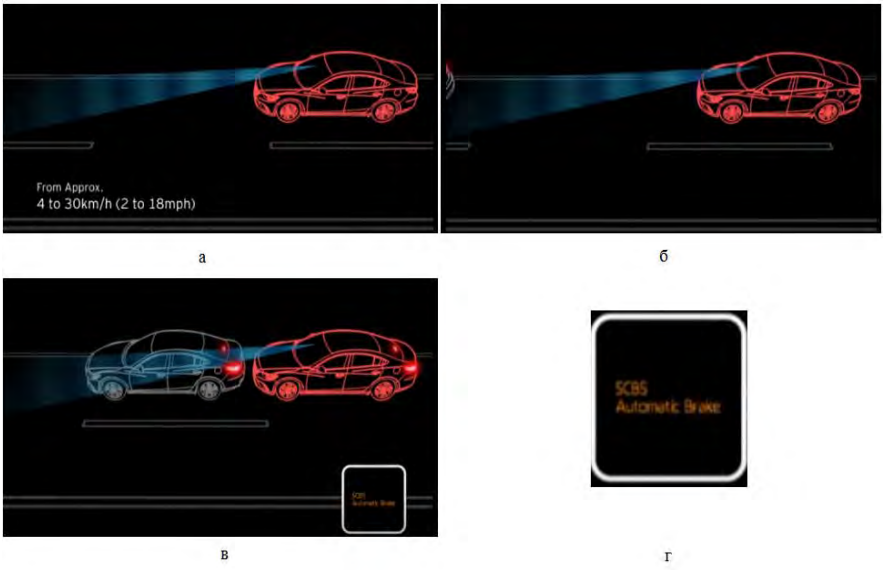


Рисунок 1.2 – Принцип роботи системи Smart City Brake Support:
а – рух автомобіля зі швидкістю 30–80 км/год; б – водій не гальмує; в – система автоматично гальмує; г – повідомлення про автоматичне гальмування [3]

Розпізнавання пішоходів можливе, якщо пішохід ростом від 1 до 2 метрів, достатньо освітлений, без зонту чи великого багажу, не з'являється перед авто раптово. Система не здатна розпізнати двухколісні транспортні засоби – велосипеди та мотоцикли [3].

Детальний опис механізму роботи системи наведено в інструкції експлуатації авто [3].

Загалом, перевагами даного проєкту є широкий набір функцій, забезпечення автоматичного гальмування при небезпеці зіткнення, а серед недоліків можна виділити незадовільну спроможність розпізнавання пішоходів, велосипедів та мотоциклів та застосування лише в певних авто Mazda.

1.2.2 Автопілот Tesla

Автопілот Tesla це система розширеної допомоги водію (advanced driver-assistance system), та є функціональною частиною автомобілів

які виробляє американська компанія Tesla (рис. 1.3). Дана система може самостійно слідувати своєю полосою дороги, змінювати полоси при видимій розмітці, має адаптивний круїз-контроль, здатна паркуватися, може бути викликана для виїзду з гаража та навпаки для заїзду в гараж. Планується, що будуть додані функції зміни автостради, та виїзд з автостради, якщо пункт призначення користувача поблизу [7].

Рисунок 1.3 – Серія автомобілів Tesla [7]

З 2014 року було випущено чотири версії Автопілоту Tesla з різним апаратним забезпеченням, дані версії відомі як: Hardware 1 (2014), Hardware 2 (2016), Hardware 2.5 (2017), Hardware 3 (2018).

Hardware 1 характеризується наступним апаратним забезпеченням: камера змотована на верхній частині лобового скла, направлений на дорогу радар фірми Bosch, поміщений за рещітку, надзвукові акустичні сенсори місцеположення на обох бамперах, що забезпечують кут огляду у 360°, бортовий комп’ютер Mobileye EyeQ3. Дане обладнання дозволяє розпізнавати дорожні знаки, розмітку, перешкоди та інші машини. Оновлення з Hardware 1 до Hardware 2 не передбачено, через великі об’єми роботи та витрати [7].

До складу Hardware 2 входить графічний процесор (graphical processing unit) Nvidia Drive PX 2 для обчислень, що базуються на архітектурі паралельних обчислень CUDA (compute unified device architecture), яка дозволяє значно підвищити обчислювальну продуктивність завдяки використанню графічних процесорів фірми Nvidia [7].

А також, до Hardware 2 входить 8 камер, 12 надзвукових сенсорів та радар з покращеними показниками обробки, напрямлений на дорогу, який може розпізнавати машини у зливу та туман, є можливість змінювати бортовий комп'ютер. Tesla стверджує що даний апаратний набір може обляти 200 кадрів на секунду, а також що забезпечення Hardware 2 робить Автопілот Tesla сумісним зі стандартом 5 рівня (повна автоматизація) організації SAE (Society of Automotive Engineers) [7].

Версія Hardware 2.5 (також відома як Hardware 2.1) відрізняється від попередньої доданим другим модулем (без графічного процесора) для забезпечення більшої обчислювальної продуктивності та зменшенням кількості проводки для підвищення надійності роботи [7].

Версія Hardware 3 включає в себе спеціальний процесор, розроблений Tesla. Було заявлено, що нова система буде обробляти 2000 кадрів в секунду, Tesla назвала це "прискорювачем нейронної мережі" (neural network accelerator), а також компанія стверджує, що дана версія більш націлена на "повне самокерування", а не на функції "розширеного автопілоту" [7].

Автопілот Tesla може передбачати фронтальне чи бокове зіткнення з іншою машиною, велосипедом чи пішоходом, на відстані до 160 метрів, система повідомляє про це звуковим сигналом, та може автоматично застосовувати гальма, якщо вважає можливим зіткнення з перешкодою [7].

На жаль, відомі випадки помилок Автопілоту Tesla, що призвели до аварій та загибелі людей. В 2016 році відбулося три аварії – перша в КНР, Автопілот Tesla не зреагував на маневр машини попереду, що раптово виконала маневр на дорозі, та не застосував гальма. Друга у США – Автопілот Tesla не зреагував на маневр автовоза на дорозі, та не застосував гальма [7]. Третя відбулась в Норвегії у 2016 році, Автопілот Tesla не розпізнав мотоцикліста [1].

В 2017 році в результаті дослідження, було виявлено, що Автопілот Tesla має вразливість – низьку здатність вірно розпізнавати велосипедистів, що є потенційною причиною для аварій.

В 2018 році у США за участю Автопілота Tesla відбулося три аварії. В першій Tesla зіткнувся з припаркованою пожежною машиною. В результаті дослідження цього випадку, було зроблено

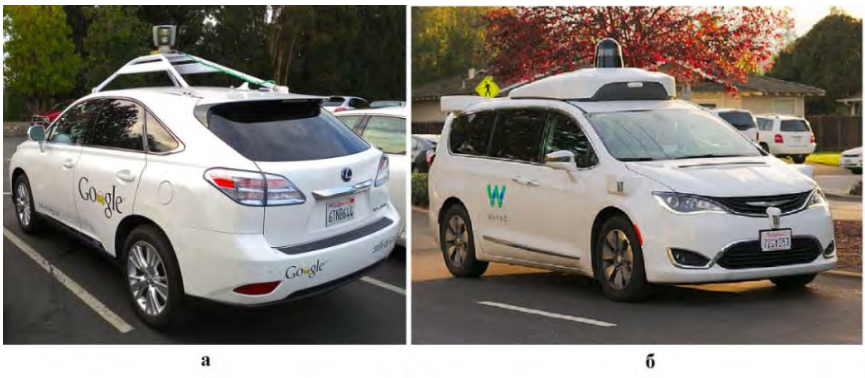
висновок, що Автопілот Tesla може не детектувати статичні машини та об'єкти, рухаючись із автострадною швидкістю. В другій Tesla врізалась у вузький бетонний бар'єр, на розділенні дороги автостради. Причиною стало помилкове розпізнавання розмітки. Третя аварія виникла з пожежною машиною що зупинилась на червоний колір світлофора, та через те, що пожежна машина була статична, Автопілот Tesla не розпізнав її [7].

Перевагами даного проєкту є забезпечення 5 рівня SAE (повної автоматизації), автоматичного гальмування при небезпеці зіткнення, а серед недоліків можна виділити помилки в розпізнаванні вантажівок, велосипедів, дорожньої розмітки, нездатність детектувати статичні транспортні засоби та застосування лише в авто Tesla.

1.2.3 Waymo

Waymo – проєкт автопілоту від одноіменної компанії що належить Alphabet Inc. (з 2016 року), проте був започаткований компанією Google у 2009 році [7, 8].

Автомобілі Waymo використовують інформацію з сервісу Google Street View та обладнані відеокамерами, лідаром який встановлено на криші, радарами в передній частині авто та датчик за допомогою якого визначається поточна позиція авто на карті, який підключено до одного з задніх колес (рис. 1.4) [7].



а б

Рисунок 1.4 – Безпілотні авто: а – Google; б – Alphabet Inc. Waymo [7]

За даними Google, в 2010 році в результаті тестування автомобілів, отримано що в реальних умовах, без участі людини, автомобілі проїхали близько 1600 км, та 225308 км з частковою участю людини. В 2012 році було повідомлено про відмітку у більш ніж 480000 км за мінімальної участі людини. В 2013 році було виявлено, що автомобілі Google не можуть рухатися під зливою чи по засніженій місцевості, через те, що визначення поточної місцевості проводиться за рахунок порівняння фотографій вулиць з результатами візуалізації наколишнього ландшафту скануючими системами авто. Даний підхід дозволяє відрізнити пішохода від стовпа, проте за поганої погоди в системи виникають складнощі. В кінці травня 2014 Google представив новий прототип свого безпілотного автомобіля, що не має керма, педалей, та повністю автономний. Станом на 2014 рік автомобілі Google мали проблеми з розпізнаванням сигналів світлофора, та була відсутня здатність паркуватися. В 2016 році у Google розробили спосіб розпізнавання проблискових маячків машин екстрених служб, що дозволить безпілотному авто відреагувати згідно правил дорожнього руху, на поліцейську машину чи машину швидкої допомоги. З 2016 року проєкт з розробки автономних автомобілів перейшов до Alphabet Inc. та отримав назву Waymo [7].

В 2017 році було оновлено комплектацію Waymo: додано нові сенсори та контролери, які дешеве виготовити, камери для покращення огляду та очисники для лідара. Для Waymo розроблюються власні сенсори, радари та лідари, що зменшує залежність від інших компаній та зменшує собівартість готового автомобіля.

Розробники Waymo стверджують, що сенсори якими обладнані їх авто здатні розпізнавати пішоходів, велосипеди, автівки, закриті на ремонт дороги та інші об’єкти в радіусі що дорівнює трьом футбольним полям на 360°. Лазери на близькій дистанції можуть розпізнавати та фокусуватися на об’єктах що розташовані недалеко від авто, а радар дозволяє оглядати місцевість довкола авто та слідкувати за рухом об’єктів.

Відомі випадки аварій за участю автомобілів Google. Станом на 2015 рік Google повідомила що 23 автівки були учасниками 14 ДТП, проте у Google запевняли, що аварії відбувались не з вини їх автопілота, а через водіїв авто та водіїв інших автівок.

В 2016 році, автомобіль Google протаранив автобус, коли намагався об'їхати перешкоду у вигляді мішків з піском. У Google визнали свою провину, та охарактеризували дану аварію як непорозуміння та недолік навчання автопілота.

Google звітувала щомісяця, щодо ДТП, в яких були задіяні її автомобілі, Waymo продовжили цю традицію [7].

Плюси даного проєкту в наступному: компанія сама займається виготовленням обладнання для авто, розпізнавння об'єктів можливе на великій відстані. Недоліками проєкту є проблеми з визначенням поточного місцезнаходження та проблеми з розпізнаванням об'єктів.

1.2.4 Розпізнавання машин Blippar

Blippar – компанія яка спеціалізується на створенні програм доповненої реальності для смартфонів [9].

У 2017 році дана компанія з використанням глубокого навчання нейронних мереж та комп'ютерного зору створила алгоритм, який дозволяє розпізнати марку машини та отримати таку інформацію, як рік виготовлення моделі, технічні специфікації, приблизну вартість та відгуки, а також є можливість здійснити віртуальний огляд салону. Інформацію можа отримати лише сфотографувавши авто, як припарковане, так і те що в рухається (до 24 км/год). Проте інформація наявна для машин, що були випущені після 2000 року та присутні на ринку США [10].

Компанія стверджує, що її розробка здатна розпізнавати автомобілі з точністю у 97,7% [11].

Крім того, компанія раніше реалізувала розпізнавання різноманітних об'єктів: облич, страв, рослин та тварин [10].

На офіційному сайті компанії [11] є можливість власноруч перевірити роботу розпізнавання, обравши зображення автомобіля (рис. 1.5).

Користувацька версія програми з використанням даного алгоритму доступна для користувачів операційних систем iOS та Android тільки у США, а професійна, платна версія Blippar Car Recognition API доступна по всьому світу [12].

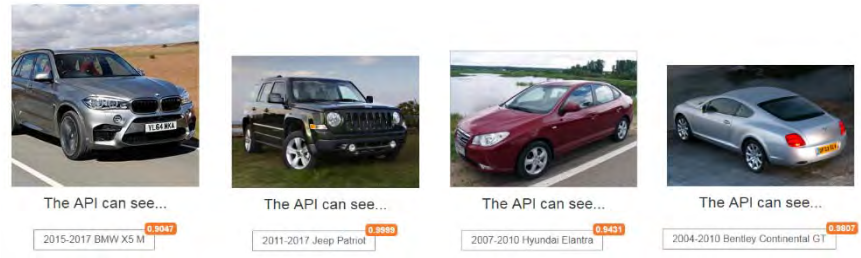


Рисунок 1.5 – Приклади розпізнавання на сайті Blippart [11]

Переваги даного проєкту в наступному: можливість отримати детальну інформацію про авто лише по фото, можливість віртуального огляду салону авто, є можливість використати API алгоритмів Blippar у власних проєктах. Серед недоліків проєкту – нездатність розпізнавання авто випущених раніше 2000 року та доступність додатку лише на території США.

1.2.5 Mobileye

Mobileye – ізраїльська компанія, яка розробляє розвинуті системи допомоги водію (ADAS – advanced driver-assisstance systems), що покликані попередити та зменшити ймовірність аварії [13].

На початку існування компанія займалась розробкою алгоритмів та спеціалізованого нейронного процесору EyeQ, в якому працюють всі розроблені компанією алгоритми обробки зображень. Після років тестувань плата з процесором EyeQ надійшла в продаж, та була використана такими фірмами як BMW, General Motors, Volvo. Спочатку, дана плата пропонувалась в якості додаткової опції при купівлі машини, а згодом стала частиною стандартного комплекту [13].

Електронна система контролю дотримання ряду Mobileye, що слідкує за положенням авто на дорозі та попереджує водія про порушення ряду встановлювалась з 2007 року на машинах BMW, General Motors, Volvo [13].

З 2008 року машини серії BMW 7 доповнюються системою розпізнавання дорожних знаків Mobileye, а також адаптивною системою керування фарами [13].

В 2010 році Mobileye оголосила про впровадження системи екстреного гальмування при наявності небезпеки зіткнення з пішоходом. Розпізнавання пішоходів засноване на роботі відеокамери з використанням обробки образів та застосуванням класифікаторів і аналізу оптичних потоків, та для розпізнавання не важливо, рухається пішохід чи ні. Дистанція на якій система здатна розпізнати пішохода дорівнює 30 метрів [13].

В 2011 році Mobileye продемонстрували першу в світі систему автономного екстреного гальмування авто, яка базувалась лише на аналізі зображення, та встанавлювалась на автовках BMW, General Motors та Opel. Також в 2011 році Mobileye оголосила про створення функціоналу що повідомляє про ризик зустрічного транспорту [13].

В 2015 році компанія Tesla почала використовувати плати Mobileye у своїх автомобілях, починаючи з моделі Tesla Model S. В 2016 році після аварії за участю Tesla та автовоза, у Mobileye заявили, що їх алгоритми зможуть розпізнавати вантажівки (та вантажівки з прицепами) починаючи з 2018 року.

Однак пізніше, у 2016 році, компанії оголосили про зупинення співпраці [13].

В 2017 році Mobileye разом з BMW та Intel оголосили про початок розробки тестового автопарку безпілотних машин. В подальших планах компаній – масовий випуск безпілотних машин близько 2021 року. В тому ж році в Intel оголосили про купівлю Mobileye за 15,3 млрд. доларів [13].

На сьогодні, технологічною основою продуктів компанії є користання алгоритмів розпізнавання руху що апаратно реалізовані в серії плат EyeQ, на відміну від інших, що використовують радари та лідари. Алгоритми аналізу зображень розроблені Mobileye здатні розпізнати автівки, мотоцикли, вантажівки та пішоходів, незалежно від частини доби. Джерелом зображень є відеокамера, закріплена на дзеркалі заднього виду [13].

Станом на 2018 рік компанія випустила чотири плати: EyeQ1, EyeQ2, EyeQ3, EyeQ4. В планах компанії у 2020 році випустити наступну плату – EyeQ5 (табл. 1.1) [14].

Таблиця 1.2 містить пояснення рівнів автономності SAE [15].

Таблиця 1.1 – Характеристики плат серії EyeQ [14]

Плата	Рік	Рівень автономності SAE	Продуктивність	Живлення	Напівпровідникова технологія
EyeQ1	2008	1	0,0044	2,5 Вт	180nm CMOS (1999)
EyeQ2	2010	1	0,026	2,5 Вт	90nm CMOS (2004)
EyeQ3	2014	2	0,256	2,5 Вт	40nm CMOS (2008)
EyeQ4	2018	3	2,5	3 Вт	28nm FD-SOI (2011)
EyeQ5	2020	4/5	24	10 Вт	7nm FinFET (2018)

Таблиця 1.2 – Рівні автоматизації SAE [15]

Рівень автономності	Пояснення
0 (Автоматизація відсутня)	Водій самостійно виконує всі дії пов’язані з кермуванням
1 (Асистент)	Присутня система прискорення чи зменшення швидкості руху, використовується інформація про середовище руху, та очікування, що водій виконує всі дії пов’язані з кермуванням
2 (Часткова автоматизація)	Присутні декілька систем допомоги водію, прискорення та зменшення швидкості руху, та очікування, що водій виконує всі дії пов’язані з кермуванням
3 (Умовна автоматизація)	Присутня система яка здатна керувати процесом руху автомобіля, з очікуванням, що водій зреагує на запит про втручання
4 (Високий рівень автоматизації)	Присутня система яка здатна керувати процесом руху автомобіля, навіть якщо водій не реагує на запити про втручання
5 (Повна автоматизація)	Присутня система яка здатна керувати процесом руху автомобіля при всіх умовах та станах навколишнього середовища

Mobileye стверджує, що системи їх системи:

– попеджують водія за 2 секунди до можливого зіткнення з пішоходом;

– попереджують водія за 2,7 секунди до можливого зіткнення з транспортним засобом що рухається попереду [16].

Компанія пропонує різні рішення систем для допомоги водію: Mobileye 550, Mobileye 560, Mobileye Smartphone Application [16].

Mobileye 550 та Mobileye 560 (рис. 1.6) – системи 5 покоління, що характеризується інтеграцією через Bluetooth, та через спеціальну програму для смартфонів передають водію попередження про різні ситуації та мають наступні функції [17]:

– попередження про можливість зіткнення з авто що рухається попереду (Mobileye FCW);

– попередження про можливість зіткнення з пішоходом (Mobileye PCW);

– попередження про зменшення відстані між авто (Mobileye HMW);

– попередження про порушення ряду руху (Mobileye LDW);

– інтелектуальне керування дальнім світлом (IHC);

– індикатор обмеження швидкості (SLI).

Рисунок 1.6 – Системи Mobileye [17]: а – Mobileye 550; б – Mobileye 560

Відмінністю Mobileye 560 від Mobileye 550 є наявність додаткового пристрою – EyeWatch, основним призначенням якого є візуальне на звукове повідомлення про небезпеку зіткнення з транспортним засобом чи пішоходом, або порушення ряду руху [18].

На рисунках 1.7, 1.8 наведено приклади обробки відеопотоку системами Mobileye. Зелена область – вільний простір дороги, голуба полоса – відстань до транспортного засобу що рухається попереду [19].

Рисунок 1.7 – Приклад обробки відеопотоку системами Mobileye [19]

Рисунок 1.8 – Приклад обробки відеопотоку системами Mobileye [19]

На рисунках 1.9, 1.10 наведено приклади розпізнавання сигналів світлофорів [19].

Рисунок 1.9 – Приклад розпізнавання сигналів світлофорів [19]

Рисунок 1.10 – Приклад розпізнавання сигналів світлофорів [19]

Mobileye Smartphone Application дозволяє отримувати попередження про небезпечні ситуації безпосередньо на смартфон через Bluetooth, та підтримується лише системами 5 покоління [20].

Mobileye Smartphone Application має наступні функції:

– Mobileye FCW – попередження про можливість зіткнення з транспортним засобом, що рухається попереду: відображення червого авто (рис. 1.11а), якщо дистанція в межах допустимої – відображається зелене авто (рис. 1.11б);

Рисунок 1.11 – Функція Mobileye FCW [20]:
а – можливість зіткнення з авто; б – дистанція в допустимих межах

– Mobileye PCW – попередження про можливість зіткнення з пішоходом: відображається червоний пішоход (рис. 1.12а), якщо пішохід в зоні потенційного зіткнення відображається помаранчевий пішоход (рис. 1.12б);

Рисунок 1.12 – Функція Mobileye PCW [20]:
а – можливість зіткнення з пішоходом;
б – пішоход в межах зони потенційного зіткнення

– Mobileye LDW – попередження про порушення ряду руху. Відображаються позначки лівого чи правого ряду відповідно (рис. 1.13);

Рисунок 1.13 – Попередження про порушення ряду руху [20]

– Mobileye HMW – відображає поточну дистанцію по відношенню до транспортного засобу попереду. Якщо авто знаходиться ближче вказаної відстані (обчислюється в секундах), система попередить про небезпеку і відобразиться червоне авто з відстанню (рис. 1.14 а), якщо відстань в межах допустимої відобразиться авто зеленого кольору (рис. 1.14 б).

Рисунок 1.14 – Відображення відстані до транспортного засобу [20]:
а – небезпечна відстань; б – допустима відстань

– Mobileye SLI – розпізнає дорожні скоростні знаки та попереджає водія, якщо шкидкість авто перевищує допустиму. Знак буде мерехтіти, якщо зафіксоване перевищення дозволеної швидкості (рис. 1.15).

Рисунок 1.15 – Повідомлення про обмеження швидкості руху [20]

– Mobileye TSR – розпізнає різноманітні дорожні знаки та попереджає про них водія (рис. 1.16). Для функцій TSR та SLI нещодавно розпізнані знаки відображаються та поступово бліднуть, поки повністю не зчезнуть. Яскраві кольори знаку свідчать про їх актуальність.

Рисунок 1.16 – Попередження про дорожній знак [20]

– Mobileye IHC – здійснює контроль над світовими фарами, автоматично змінюючи світло з дальнього чи ближнього в залежності від наявності зустрічного руху. Про активність дальнього світла свідчить індикатор – синя фара.

Проєкти Mobileye мають такі переваги: надають широкий набір функцій та можливість використання у будь-якому авто, реалізовано автоматичне гальмування при небезпеці зіткнення. Серед недоліків можна виділити такі: наявні проблеми з розпізнаванням вантажівок, низька практичність надсилання сповіщень на смартфон при виникненні небезпечної ситуації.

1.2.6 Проєкт розпізнавання рукописних цифр набору даних MNIST за допомогою бібліотеки Keras

Даний проєкт є практичною частиною до вільних інтернет-курсів спеціаліста з програмування глибоких нейронних мереж А.Созикіна [21]. В даному проєкті розглядається навчання повнозв'язної нейронної мережі для розпізнавання рукописних цифр, за набором даних MNIST, який містить 60000 навчальних зображень та 10000 тестових зображень (рис. 1.17) [22], з використанням бібліотеки Keras.

Рисунок 1.17 – Зразки зображень MNIST [22]

У цьому проєкті повнозв'язна нейронна мережа містить два шари: перший містить 800 нейронів з функцією активації relu, та другий – вихідний, містить 10 нейронів (по нейрону на клас) з функцією активації softmax. Функція обчислення втрат – categorical crossentropy, функція оптимізації – SGD, метрика – точність. Навчання відбувалось

на протязі 100 епох, та розмір вибірки навчання складав 200 зображень [21].

Після навчання повнозв'язної нейронної мережі, її було протестовано на зображеннях що не використовувались в навчанні. В результаті тестування було отримано, що точність розпізнавання рукописних цифр навченої нейронної мережі дорівнює 96,34% [21].

Даний проєкт є суто навчальним та демонстративним, проте його ідеї можна використати у дослідженні навчання нейронних мереж для розпізнавання образів відмінних від цифр.

1.2.7 Проєкт розпізнавання об'єктів на зображеннях за допомогою бібліотеки Keras

Даний проєкт є практичною частиною до вільних інтернет-курсів спеціаліста з програмування глибоких нейронних мереж А.Созикіна [23]. В даному проєкті розглядається навчання згорткової нейронної мережі для розпізнавання образів, за набором даних CIFAR-10, що складається з 60000 кольорових зображень розділених на 10 класів (літак, авто, птиця, кіт, олень, пес, жаба, кінь, корабель, вантажівка): 50000 навчальних зображень та 10000 тестових зображень (рис. 1.18) [24], з використанням бібліотеки Keras.

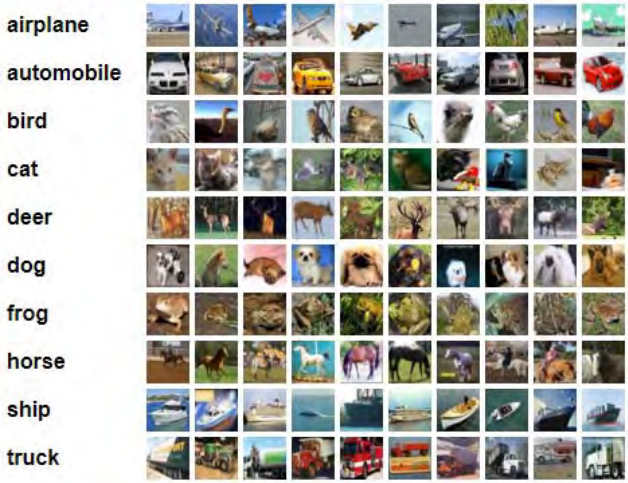


Рисунок 1.18 – Зразки зображень CIFAR-10 [24]

В даному проєкті згорткова нейронна мережа має наступну архітектуру: 2 шари згортки, що мають по 32 нейрони та функцію активації relu, шар підвибірки, 2 шари згортки що мають по 64 нейрони та функцію активації relu, шар підвибірки, повнозв'язний шар для класифікації, що має 512 нейронів, вихідний шар з 10 нейронів, з функцією активації softmax. Функція обчислення втрат – categorical crossentropy, функція оптимізації – SGD, метрика – точність. Навчання відбувалось на протязі 25 епох, а розмір вибірки навчання складав 32 зображення [23].

Після навчання згорткової нейронної мережі, її було протестовано на зображеннях що не використовувались в навчанні. В результаті тестування було отримано, що точність розпізнавання рукописних цифр навченої нейронної мережі дорівнює 76,29% [23].

Даний проєкт є суто навчальним та демонстративним, проте його ідеї можна використати у дослідженні навчання нейронних мереж для розпізнавання образів відмінних від набору даних CIFAR-10.

1.2.8 Проєкт підбіру параметрів при навчанні повнозв'язної нейронної мережі

Даний проєкт є дослідженням впливу значень параметрів на навчання повнозв'яної нейронної мережі, та їх підбору за рахунок використання простого генетичного алгоритму, з використанням бібліотеки Keras [25].

Автор даного проєкту відмічає що за набором даних MNIST, завдяки підбору параметрів, досягається точность розпізнавання у 98%, проте за набором CIFAR-10 вдається досягти лише 56% [25].

В простому генетичному алгоритмі було використано 20 особин в популяції та 10 популяцій, також було підвищено ймовірність мутації з 1% до 20%. Параметрами нейронної мережі що підбиралися були: кількість нейронів (32 – 1024), кількість шарів (1 – 4), функція активації (relu, elu, tanh, sigmoid), функція оптимізації (rmsprop, adam, sgd, adagrad, adadelta, adamax, nadam) [25].

Для навчання повнозвязної нейронної мережі за набором даних CIFAR-10 розмір вибірки для навчання складав 64 зображення, для набору MNIST – 128 зображень. Вихідний шар нейронної мережі містить функцією активації softmax. Функція обчислення втрат

нейронної мережі – categorical crossentropy, метрика – точність. Особливістю проєкту є використання необмеженої кількості епох, навчання зупиться тоді, коли показник точності на тестових зображеннях зупинить покращуватися [25 - 41].

Даний проєкт є суто демонстративним, проте його ідеї можна використати у дослідженні підбору параметрів для навчання нейронних мереж за допомогою простого генетичного алгоритму, для розпізнавання образів відмінних від набору даних CIFAR-10 та MNIST.

У таблиці 1.3 наведено порівняльну характеристику розглянутих у пункті 1.2 систем, що вирішують задачу розпізнавання образів.

Таблиця 1.3 – Порівняння систем розпізнавання образів

Система	Переваги	Недоліки
Mazda Smart City Brake Support	Широкий набір функцій; автоматичне гальмування при небезпеці зіткнення	Часткова реалізація розпізнавання пішоходів; нездатність розпізнавати велосипеди та мотоцикли; застосовується лише в певних авто Mazda
Автопілот Tesla	5 рівень SAE (повна автоматизація); автоматичне гальмування при небезпеці зіткнення	Помилки в розпізнаванні вантажівок, велосипедів та дорожньої розмітки; нездатність детектувати статичні транспортні засоби; застосовується лише в авто Tesla
Waymo	Виготовлення обладнання для авто компанією; велика дальність дії розпізнавння об’єктів	Проблеми з визначенням місцезнаходження та розпізнаванням об’єктів
Blippar	Детальна інформація про авто лише по фото; можливість віртуального огляду салону авто; API для використання алгоритмів Blippar у власних проєктах	Нездатність розпізнавання авто випущених раніше 2000 року; доступ до додатку можливий лише на території США
Mobileye	Широкий набір функцій; можливість використання у будь–якому авто; автоматичне гальмування при небезпеці зіткнення	Проблеми з розпізнаванням вантажівок; низька практичність надсилання сповіщень на смартфон

Виходячи з таблиці 1.3, можна зробити висновок, що на даний момент розв'язання задачі розпізнавання образів (пішоходів, велосипедів, вантажівок) є доволі актуальною задачею.

1.3 Порівняння підходів реалізації розпізнавання образів

У таблиці 1.4 наведено порівняння підходів до рішення задачі розпізнавання образів.

Таблиця 1.4 – Порівняння підходів до рішення задачі розпізнавання образів

Підхід	Переваги	Недоліки
Метод Віоли-Джонса [42]	Стійкість до викривлень зображень, швидкість розпізнавання, висока точність	Низька швидкість навчання, результати розпізнавання залежать від ресурсів комп'ютера, помилки в розпізнаванні
Метод HOG [43]	Прийнятна точність, незалежність від масштабу зображень	Складність реалізації, результати розпізнавання залежать від ресурсів комп'ютера
Згорткова НМ [44]	Висока швидкість навчання, точність розпізнавання	Стійкість до змін зображення забезпечується частково, для навчання необхідні великі обчислювальні ресурси, складність підбору значень параметрів мережі при навчанні
Повнозв'язна НМ [45]	Висока швидкість навчання, проста реалізація	Зображення розглядається як вектор, необхідність навчання великої кількості параметрів навіть для невеликих зображень, низькі показники точності

1.4 Дослідження реалізації процесу розпізнавання образів методом Віоли-Джонса з використанням бібліотеки OpenCV

1.4.1 Підготовка до навчання каскадного класифікатора

Етап підготовки до навчання каскадного класифікатора включає в себе підготовку вибірок зображень, «позитивної» – із зображеннями велосипедів, та «негативної» – зображення без велосипедів, складання файлів-описів зображень цих вибірок та створення вектору із зображень «позитивної» вибірки [46 - 51].

Для навчання каскадного класифікатору було зібрано 3100 «позитивних» зображень і 6200 «негативних» зображень. На рисунках 1.19 та 1.20 наведено приклади зображень. Для досягнення високої точності розпізнавання співвідношення «позитивних» до «негативних» має бути 1:2.

Рисунок 1.19 – Приклади «позитивних» зображень [46 - 50]

Також були складені файли-описи для обох типів зображень. Для «позитивних» зображень файл–опис має містити таку інформацію: шлях і ім'я файлу, кількість об'єктів, *x* координата початку об'єкта, *y* координата початку об'єкта, *x* координата кінця об'єкта, *y* координата кінця об'єкта (рис. 1.21).

Для «негативних» зображень файл–опис має містити шлях і ім'я файлу.

Приклад структури зберігання файлів необхідних для навчання представлено на рисунку 1.22.

Рисунок 1.20 – Приклади «негативних» зображень [46 - 50]

```
bike\bike_1.jpg 1 0 0 728 429
bike\bike_2.jpg 1 0 0 220 186
bike\bike_3.jpg 1 0 0 210 126
bike\bike_4.jpg 1 0 0 590 383
bike\bike_5.jpg 1 0 0 1200 800
bike\bike_6.bmp 1 0 0 274 431
bike\bike_7.jpg 1 0 0 155 92
bike\bike_8.jpg 1 0 0 424 253
```

Рисунок 1.21 – Приклад файлу-опису «позитивних» зображень

```
\Positive
        001.png
        002.png
        ...
\Negative
        001.png
        002.png
positive.dat
negative.dat
```

Рисунок 1.22 – Приклад структури зберігання файлів

Перед навчанням класифікатора було сформовано вектор «позитивних» зображень. Для цього була використана утиліта бібліотеки OpenCV «opencv_createsamples»:

```
opencv_createsamples –info /home/anton/bicycles/positive.dat –num 3100 –vec /home/anton/bicycles/samples.vec –w 50 –h 25,
```

де info /home/anton/bicycles/positive.dat – шлях до файлу опису «позитивних» зображень, num 3100 – число «позитивних» зображень, vec /home/anton/bicycles/samples.vec – шлях для створюваного файлу вектору, w 50 – ширина шаблону для пошуку, h 25 – висота шаблону для пошуку.

1.4.2 Навчання каскадного класифікатора

Після того як вектор зображень був сформований, було виконано виклик з бібліотеки OpenCV утиліти навчання класифікатора «opencv_traincascade»:

```
opencv_traincascade –data ~/bicycles/training –vec ~/bycicles/samples.vec –bg ~/bycicles/negative.dat –numStages 20 –numPos 2790 –numNeg 6200 –w 50 –h 25 -mode ALL –precalcValBufSize 1024 –precalcIdxBufSize 1024 > log.txt,
```

де data ~/bicycles/training – шлях для збереження проміжних та підсумкового результатів навчання, vec ~/bycicles/samples.vec – шлях до вектору «позитивних» зображень, bg ~/bycicles/negative.dat – шлях до файлу опису «негативних» зображень, numStages 20 – кількість рівнів навчання, numPos 2790 – 90% від кількості «позитивних» зображень, numNeg 6200 – кількість «негативних» зображень, w 50 – ширина шаблону для пошуку, h 25 – висота шаблону для пошуку, mode ALL – визначається, використовувати чи ні повний комплект ознак Хаара, precalcValBufSize 1024 та precalcIdxBufSize 1024 – пам'ять що виділяється під процес навчання, > log.txt – запис повідомлень у файл.

Варто зазначити, що для шаблону розміром 50x25 пікселів кількість можливих ознак Хаара становить 1176509.

Після виклику даної утиліти починається процес навчання. Приклад наведено на рисунку 1.23.

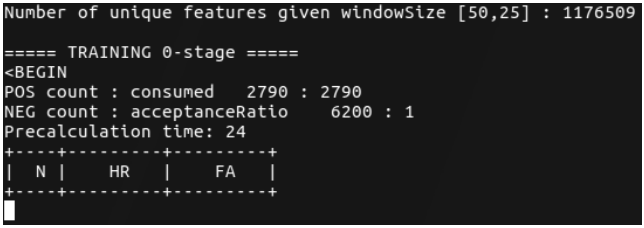


Рисунок 1.23 – Початок навчання

Значення деяких параметрів навчання які були використані за замовчуванням:

– featureType (ознаки які використовуються при навчанні) – HAAR (також можливе використання LBP);

– boostType (алгоритм навчання) – GAB (також можливе використання DAB, RAB, LB) оскільки в процесі порівняння[50] визначено що він має найбільш прийнятне співвідношення «час–продуктивність»;

– minHitRate (відсоток «позитивних» зображень розпізнаних вірно) – 0,995;

– maxFalseAlarmRate (відсоток «негативних» зображень невірно розпізнаних як «позитивні») – 0,5.

Підсумком навчання є файл «cascade.xml», який в подальшому можна включити в програму і використовувати для розпізнавання велосипедів.

Навчання каскадного класифікатору з ознаками Хаара за допомогою утиліт бібліотеки OpenCV вимагає великих обчислювальних ресурсів, особливо у випадках великих об’ємів вхідних вибірок.

Даний підхід навчання класифікатору чутливий до вхідних параметрів, наступні параметри безпосередньо впливають на якість розпізнавання:

– параметр numStages: визначає кількість рівнів каскадного класифікатору. Рекомендовані значення від 16 до 25;

– параметр minHitRate: відсоток «позитивних» зображень розпізнаних вірно. За замовчування має значення 0,995. Означає, що у вихідній вибірці буде не більше ніж 1-0,995=0,005% пропусків. Чим

вище даний параметр, тим вище рівень хибних спрацювань. Залежить від вхідної вибірки;

– параметр maxFalseAlarmRate: максимальний бажаний рівень хибних спрацювань для кожного етапу класифікатору;

– параметр numPos: певний відсоток від загальної кількості «позитивних» зображень, як правило, 80%-90% від загального числа «позитивної» вибірки. Пов'язаний з параметром minHitRate: чим нижче значення minHitRate, тим більше зображень «позитивної» вибірки буде вважатися непридатними;

На якість розпізнавання також впливають кількість (хорошими є позитивні вибірки від 10000 файлів) та різноманітність зображень (об'єкт на зображеннях повинен бути присутнім в різних ракурсах).

При використанні каскадного класифікатора, підвищити точність розпізнавання можна використовуючи максимально жорстке значення параметру перекриття розпізнавання функції non_max_suppression, бібліотеки imutils, а швидкість можна підвищити використовуючи зміни масштабу зображення. Наприклад, використовуючи значення параметру перекриття розпізнавання 0,15 та зменшення зображення на 50% було отримано, що швидкість виконання майже в 4 рази швидше, ніж при 100% масштабі зображень, проте було отримане незначне погіршення точності розпізнавання.

1.5 Дослідження реалізації процесу розпізнавання образів за допомогою нейронних мереж з використанням бібліотеки Keras

1.5.1 Навчання нейронних мереж

Для навчання нейронних мереж було використано бібліотеку Keras та як допоміжний засіб для обчислень бібліотеку TensorFlow. У ролі вхідних даних було використано базу зображень CIFAR-100 [24], яка містить 100 класів об'єктів та 20 суперкласів об'єктів. Навчання відбувалось на 45000 зображеннях, а після навчання була виконана автоматична перевірка навченої нейронної мережі на 5000 зображеннях, та визначено точність розпізнавання для кожної навченої нейронної мережі. Після навчання для подальшого використання було збережено моделі нейронних мереж до файлів *.json та ваги нейронних мереж до файлів *.h5. При навчанні

нейронних мереж були розвинуті ідеї проєктів розглянутих у пунктах 1.2.6 та 1.2.7.

Навчання повнозв'язної нейронної мережі відбувалось за такими параметрами: 1 шар – 800 нейронів, 100 нейронів на вихідному шарі, активація першошо шару – relu, активація вихідного шару – softmax, функція втрат categorical crossentropy, оптимізатор – SGD, метрика – точність, розмір вибірки навчання – 500 зображень, 400 епох. На рисунку 1.24 зображено графік навчання повнозв'язної нейронної мережі. Тут для кожної епохи: loss – втрати навчання, acc – точність навчання, val_loss – втрати перевірки , val _acc – точність перевірки. Максимальне значення acc – 0,4014 (40,14%). Максимальне значення val_acc – 0,2416 (24,16%). Точність після автоматичної перевірки повністю натренованої нейронної мережі – 24,93%.

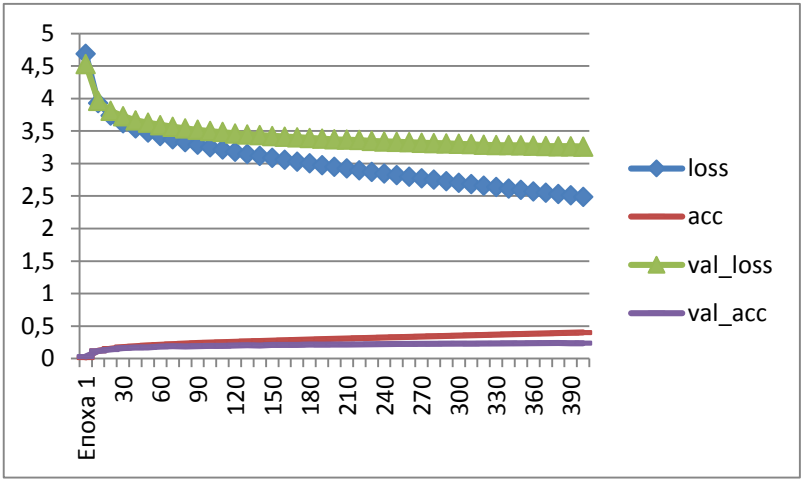


Рисунок 1.24 – Графік навчання повнозв'язної нейронної мережі

Навчання згорткової нейронної мережі відбувалось за такими параметрами: 4 шари – по 32 нейрони на перших двох шарах, та по 64 на останніх двох, відступ same, активація всіх шарів – relu, повнозв'язний шар для класифікації на 512 нейронів з активацією relu, вихідний шар на 100 нейронів з активацією softmax, функція втрат – categorical crossentropy, оптимізатор – SGD з показником learning_rate=0,01, метрика – точність, розмір вибірки навчання – 128

зображень, 100 класів, 35 епох. На рисунку 1.25 зображено графік навчання згорткової нейронної мережі. Тут для кожної епохи: loss – втрати навчання, acc – точність навчання, val_loss – втрати перевірки , val _acc – точність перевірки. Максимальне значення acc – 0,6238 (62,3%). Максимальне значення val_acc – 0,4684 (46,8%). Точність після автоматичної перевірки повністю натренованої нейронної мережі – 47,14%.

Для розпізнавання образів результуючий файл *.h5 слід підключити в програмному коді програмного забезпечення.

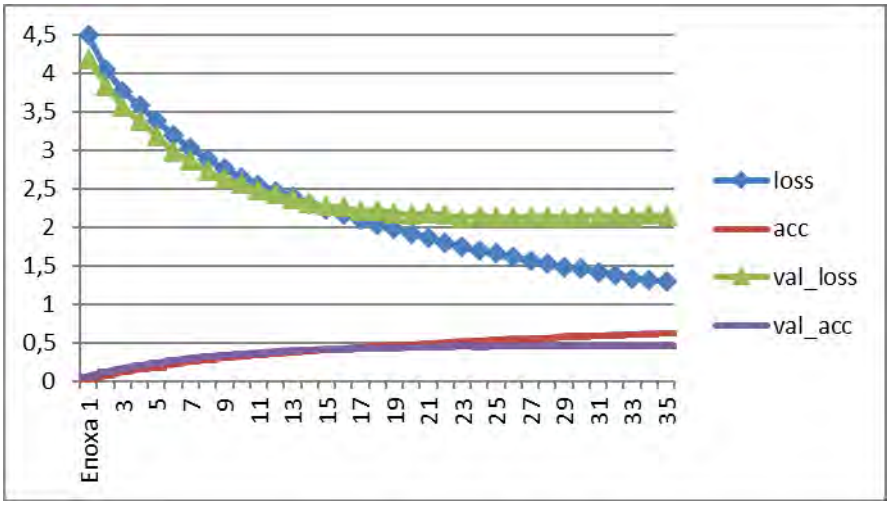


Рисунок 1.25 – Графік навчання загорткової нейронної мережі

1.5.2 Використання простого генетичного алгоритму в навчанні нейронних мереж з використанням бази зображень CIFAR-100

Для підвищення точності розпізнавання, для повнозв’язної та загорткової нейронних мереж було виконано підбір параметрів за допомогою простого генетичного алгоритму. При підборі параметрів для навчання нейронних мереж була розвинута ідея проєкту розглянутого в пункті 1.2.8.

Для нейронних мереж параметрами що підбираються були:

– кількість нейронів у шарах (64–1024 для повнозв’язної та 32–128 для згорткової);

– кількість шарів (1–4 для повнозв ’язної мережі, 4 та 6 – для згорткової);

– функція активації (relu, elu, tanh, sigmoid);

– оптимізатор (rmsprop, adam, sgd, adagrad, adadelta, adamax, nadam).

Генетичий алгоритм для повнозв’язної мережі мав такі параметри: 7 поколінь з популяцією 20 особин, а для згорткової: 5 поколінь з популяцією 20 особин.

Задання більшої кількості поколінь для обох мереж спричиняє зупинку процесу перебору, через брак обчислювальних ресурсів комп’ютера з процесором Intel Core i5-7400 з тактовою частотою 3 ГГц.

Для повнозв’язної нейронної мережі були задані такі параметри: 100 класів, розмір вибірки навчання – 128 зображень, вихідний шар зі 100 нейронів та функцією активації softmax, функція втрат – categorical crossentropy, метрика – точність, без обмеження кількості епох.

У результаті підбору параметрів для повнозв’язної мережі отримано, що за 7 поколінь, з наступними параметрами досягається точність 30,52%: 2 шари по 512 нейронів, активація – elu, оптимізатор – adamax.

Для згорткової нейронної мережі були задані такі параметри: 100 класів, розмір вибірки навчання – 128 зображень, відступи – same, вихідний шар зі 100 нейронів та функцією активації softmax, функція втрат – categorical crossentropy, метрика – точність, 35 епох.

У результаті підбору параметрів для згорткової мережі (з використанням 4 шарів) отримано, що за 5 поколінь з наступними параметрами досягається точність 48%: шари по 32 нейрони, активація – elu, оптимізатор – adamax.

У результаті підбору параметрів для згорткової мережі (з використанням 6 шарів) отримано, що за 5 поколінь з наступними параметрами досягається точність 53,69%: 1 і 2 шари – 32 нейрони, всі інші шари – по 128 нейронів, активація – relu, оптимізатор – adamax.

1.5.3 Використання простого генетичного алгоритму в навчанні нейронних мереж з використанням власної бази зображень

Для дослідження впливу кількості класів та якості зображень навчальної вибірки на підсумковий показник точності розпізнавання навченої нейронної мережі було обрано 5 класів: пішохід, велосипед, мотоцикл, авто, вантажівка та створено три вибірки які використовувались в процесі навчання нейронних мереж: навчальна, перевірочна, тестова [25] (рис. 1.26). Кожна з навчальних вибірок складається з 1000 зображень, кожна перевірочна зі 400 зображень, кожна тестова вибірка з 200 зображень. Загалом на один клас припадає по 1600 зображень.

Рисунок 1.26 – Приклади зображень задіяних у навчанні нейронних мереж [25]

Підбор параметрів здійснювався для навчання згорткової нейронної мережі, з використанням розвитку ідеї проєкту розглянутого в пункті 1.2.8.

Параметрами, що підбираються, були:

– кількість нейронів на шарах (32–128);

– функція активації (relu, elu, tanh, sigmoid);

– оптимізатор (rmsprop, adam, sgd, adagrad, adadelta, adamax, nadam).

Генетичий алгоритм мав такі параметри: 5 поколінь з популяцією 20.

Задання більшої кількості поколінь та більших значень кількості нейронів спричиняє зупинку процесу перебору, через брак обчислювальних ресурсів комп'ютера з процесором Intel Core i5-7400 з тактовою частотою 3 ГГц.

Для згорткової нейронної мережі були задані такі параметри: 5 класів, 3 шари, розмір вибірки навчання – 16 зображень, вихідний шар з 5 нейронами та функцією активації softmax, функція втрат – categorical crossentropy, метрика – точність, 35 епох.

У результаті підбору параметрів для згорткової мережі отримано, що за 5 поколінь, з наступними параметрами досягається точність 85.89%: на першому шарі – 32 нейрони, на другому – 128 нейронів, на третьому – 32 нейрони, функція активації – relu, оптимізатор – sgd. Час підбору становив 8 днів, 8 хвилин, 39 секунд.

1.6 Розроблення генетичних методів розв'язання задачі розпізнавання графічних образів

Як відзначено вище, використання відомих методів розпізнавання образів при обробленні даних, що характеризуються низькою якістю зображень, поганими погодними умовами та недостатнім освітленням, є недосить ефективними на практиці та в деяких випадках не забезпечує прийнятної точності розпізнавання. Крім того, такі методи є досить складними та вимагають використання великих обсягів обчислювальних та часових ресурсів, що впливає на отримувані результати та ефективність використання таких методів на практиці. Тому у цьому розділі виконується розробка модифікацій простого генетичного методу навчання нейромереж для підвищення ефективності процесу розпізнавання образів, зокрема для збільшення точності та швидкості розпізнавання.

1.6.1 Модифікація простого генетичного алгоритму Альфа-Бета

В порівнянні з оригінальною версією простого генетичного алгоритму, в даній модифікації було додано можливість виникнення

двох мутацій, а також, змінено відбір особин для схрещення. Це дозволяє пришвидшити виконання підбору параметрів навчання нейронних мереж, та підвищити результуючий показник точності порівняно з базовою версією простого генетичного алгоритму.

Дана модифікація полягає в наступному: для схрещення в кожній генерації обирається різна кількість пар для схрещення, при чому в парі одна особина відноситься до найпристосованіших, а друга до найменш пристосованих особин, а також випадковим чином може виникнути дві мутації (базова та подвоююча) або одна мутація (базова): за методом Монте-Карло – генерується випадкове число, 0 або 1, якщо випадає 0, то виникає одна мутація, якщо випадає 1 – виникає дві мутації.

Враховуючи все вище сказане, послідовність дій запропонованої модифікації простого генетичного алгоритму можна описати таким чином: відбувається генерація початкової популяції P_s , що складається з N особин (1.1):

$$P_S = (I_1,...,I_N), \tag{1.1}$$

де I – особина.

Кожна особина I (її гени) кодується певними значеннями з множини всіх параметрів множини Q. Нехай для підбору параметрів нейронних мереж маємо множину парамерів Q, що визначається (1.2):

$$Q = (n_j, m, f_{act}, f_{opt}), \tag{1.2}$$

де n – кількість нейронів певного шару, j=1,...,m, m – кількість шарів, f_{act} – функція активації, f_{opt} – функція оптимізації.

Тоді особина I_N початкової популяції P_s може бути закодована за формулою (1.3):

$$\begin{gathered} Q(n_i) = R[32,64,...,n_{\max}], \;\; Q(m) = R[1,2,...,m_{\max}], \\ Q(f_{act}) = R[relu, elu, tanh, sigmoid], \\ Q(f_{opt}) = R\begin{bmatrix} adam, sdg, ada\max, nadan, rmsprop, adagrad, \\ adadelta \end{bmatrix}, \end{gathered} \tag{1.3}$$

де i=1,...,m, R – функція вибору випадкової величини, n_{max} – максимальна кількість нейронів, m_{max} – максимальна кількість шарів.

Особину можна описати за (1.4):

$$I_N = R(Q), \tag{1.4}$$

Далі для оцінки оптимальності кожної особини I популяції P_S проводиться обчислення фітнес-функції f_{fit} за оформулою (1.5):

$$f_{fit}(P_S) = \max\{I_i\}, \tag{1.5}$$

де i=1,...,N.

Наступним кроком є перевірка, чи не виконалися умови закінчення пошуку оптимального рішення. Такими умовами можуть бути кількість поколінь N, значення фітнес-фінкції f_{fit}, час виконання, тощо. На прикладі підбору параметрів для навчання нейронних мереж, значенням фітнес-функції є показник точності навченої нейронної мережі. Якщо умови виконалися – то необхідно завершити виконання алгоритму та вивести отримані результати, найоптимальніші особини покоління.

Далі виконується відбір особин до нової популяції P_n: особини ранжуються за показником пристосованості, далі випадковим чином визначається кількість пар – певна кількість найпристосованіших особин, та стільки ж найменш пристосованих. Найпристосованіші особини формують підмножину B, найменш пристосовані – підмножину W. Обидві підмножини входять в множину пар V. Число особин що можуть бути обрані в пари знаходиться в діапазоні 20-60% від загальної кількості особин. Решта нової популяції P_n отримується за рахунок схрещення обраних особин (K).

Схрещення (C) особин має наступний алгоритм: з підмножин B та W випадково вибирається по особині M та F які мають певний набір генів (1.6):

$$\begin{aligned} M &= \left(\left(n_1^1,...,n_m^1\right), m^1, f_{act}^1, f_{opt}^1\right), \\ F &= \left(\left(n_1^2,...,n_m^2\right), m^2, f_{act}^2, f_{opt}^2\right). \end{aligned} \tag{1.6}$$

Тоді схрещення можна виразити як (1.7):

$$C = (M \times F). \tag{1.7}$$

Результатом схрещення особин є два нащадки K_1, K_2, які можна записати як (1.8):

$$\begin{aligned} K_1 &= \left(R\left[\left(n_1^1,...,n_m^1\right),\left(n_1^2,...,n_m^2\right)\right],R\left[m^1,m^2\right],R\left[f_{act}^1,f_{act}^2\right],R\left[f_{opt}^1,f_{opt}^2\right]\right), \\ K_2 &= \left(R\left[\left(n_1^1,...,n_m^1\right),\left(n_1^2,...,n_m^2\right)\right],R\left[m^1,m^2\right],R\left[f_{act}^1,f_{act}^2\right],R\left[f_{opt}^1,f_{opt}^2\right]\right), \end{aligned} \tag{1.8}$$

де R – функція вибору випадкової величини.

Даний підхід схрещення особин називається рівномірним схрещенням, оскільки при кодуванні нащадка випадковим чином обираються значення для кожного гена від кожного предка.

Для отриманих нащадків застосовується оператор мутації μ згідно з формулою (1.9):

$$\mu(K) = R\left[R(n_i), R(m), R(f_{act}), R(f_{opt})\right]. \tag{1.9}$$

В базовій версії алгоритму, випадковим чином виникає одна мутація, тоді як в запропонованій модифікації випадковим чином може виникнути дві (μ_1, μ_2) або одна мутація (μ). При чому, в ситуації коли виникає дві мутації – один ген може мутувати двічі.

Тоді після застосування оператору мутацій нащадка можна записати як (1.10):

$$K_\mu = ((n_1,...,n_m), m, f_{act}, f_{opt}). \tag{1.10}$$

Після застосування оператору мутації нащадки включаються до нового покоління P_n. Схрещення та мутації проводяться до тих пір, поки не буде створено нове покоління P_n розміру N:

$$P_n = (V, K_j), \tag{1.11}$$

де j=1, ..., (N-V).

Далі обчислюється фітнес-функція f_{fit} для кожної особини нового покоління P_n (формула(1.5)).

Перевіряється, чи не виконалися умови закінчення пошуку оптимального рішення. Якщо умови виконалися, тоді необхідно завершити виконання алгоритму та вивести отримані результати, найоптимальніші особини покоління.

Якщо умови не виконалися, тоді продовжити виконання алгоритму, перейти до формування наступного покоління P_{n+1} заснованого на поколінні P_n.

При використанні запропонованої модифікації можна пришвидшити виконання підбору параметрів навчання нейронних мереж, та підвищити показник точності, оскільки підвищення кількості мутацій та підбір в пари різних особин надає більшу різноманітність комбінацій генів, що призводить до кращих показників за менший час.

1.6.2 Модифікація простого генетичного алгоритму Альфа-Бета фіксована

В порівнянні з оригінальною версією простого генетичного алгоритму, в даній модифікації було додано можливість виникнення двох мутацій, додано фіксовану точку схрещення, а також, змінено відбір особин для схрещення. Це дозволяє дещо підвищити результуючий показник точності порівняно з базовою версією простого генетичного алгоритму.

Дана модифікація полягає в наступному: для схрещення в кожній генерації обирається різна кількість пар для схрещення, при чому в парі одна особина відноситься до найпристосованіших, а друга до найменш пристосованих особин, випадковим чином може виникнути дві мутації (базова та подвоююча) або одна мутація (базова): за методом Монте-Карло – генерується випадкове число, 0 або 1, якщо випадає 0, то виникає одна мутація, якщо випадає 1 – виникає дві мутації, та встановлено фіксовану точку схрещення – в схрещенні приймає участь перша половина генів, тобто гени що відповідають за кількості нейронів на шарах, а значення інших генів – завжди передаються нащадкам від однієї з особин.

Послідовність дій даної модифікації схожа на послідовність дій попередньої модифікації, проте, операція схрещення відмінна: згідно формули 1.6 з підмножин *B* та *W* випадково вибирається по особині *M* та *F* які мають певний набір генів. Їх схрещення можна виразити як (1.12):

$$C^* = ((0{,}5M \times 0{,}5F), 0{,}5F). \tag{1.12}$$

Результатом схрещення особин є два нащадки K_1, K_2., які можна записати наступним чином:

$$\begin{aligned} K_1 &= (R[(n_1^1,...,n_m^1),(n_1^2,...,n_m^2)], m^2, f_{act}^2, f_{opt}^2), \\ K_2 &= (R[(n_1^1,...,n_m^1),(n_1^2,...,n_m^2)], m^2, f_{act}^2, f_{opt}^2), \end{aligned} \tag{1.13}$$

де *R* – функція вибору випадкової величини, m^2, f_{act}^2, f_{opt}^2 – значення генів що передались від особини F.

1.6.3 Модифікація простого генетичного алгоритму

У порівнянні з оригінальною версією простого генетичного алгоритму, в даній модифікації було додано можливість виникнення двох мутацій, додано фіксовану точку схрещення. Це дозволяє пришвидшити виконання підбору параметрів навчання нейронних мереж порівняно з базовою версією простого генетичного алгоритму.

Дана модифікація полягає в наступному: встановлено фіксовану точку схрещення – в схрещенні приймає участь перша половина генів, тобто гени що відповідають за кількості нейронів на шарах, а значення інших генів – завжди передаються нащадкам від однієї з особин та на етапі мутації випадковим чином виникає дві мутації (базова та подвоююча) або одна мутація (базова): за методом Монте-Карло – генерується випадкове число, 0 або 1, якщо випадає 0, то виникає одна мутація, якщо випадає 1 – виникає дві мутації.

Послідовність дій даної модифікації схожа на послідовність дій першої модифікації, проте має певну відмінність – відбір особин до нової популяції відбувається як у базовій версії генетичного

алгоритму (формула (1.6)), та схрещення відбувається як у другій модифікації (формули (1.12)–(1.13)).

1.7 Програмне забезпечення для розпізнавання графічних образів

Розроблене програмне забезпечення включає в себе різні програми: розпізнавання велосипедів з використанням каскадного класифікатора, навчання нейронної мережі, тестування розпізнавання нейронної мережі, підбору параметрів при навчанні нейронних мереж з використанням модифікацій простого генетичного алгоритму (рис. 1.27).

Програма реалізація розпізнавання велосипедів з використанням каскадного класифікатора (метода Віоли-Джонса) складається з кількох модулів, пов’язаних методами, тому структурну схему програми можна зобразити наступним чином (рис. 1.28).

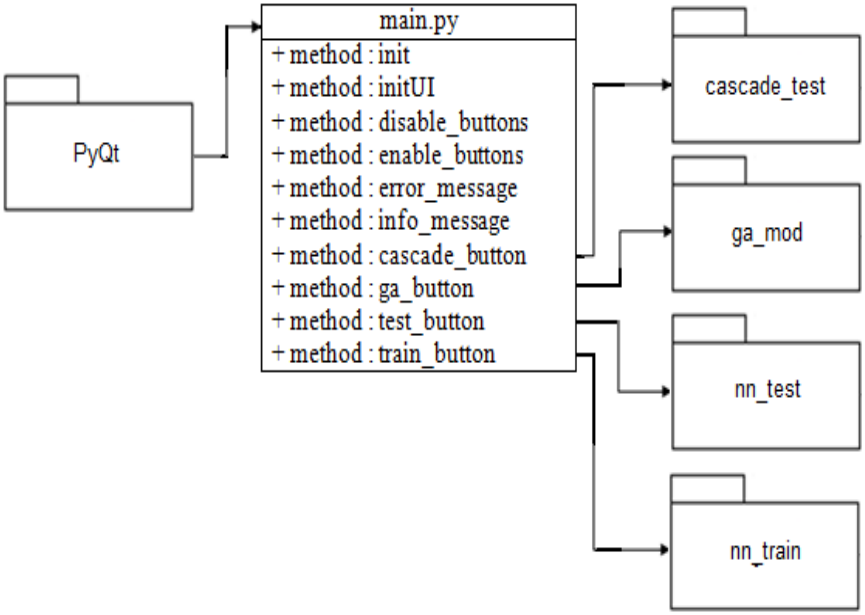


Рисунок 1.27 – Схема розробленого програмного комплексу

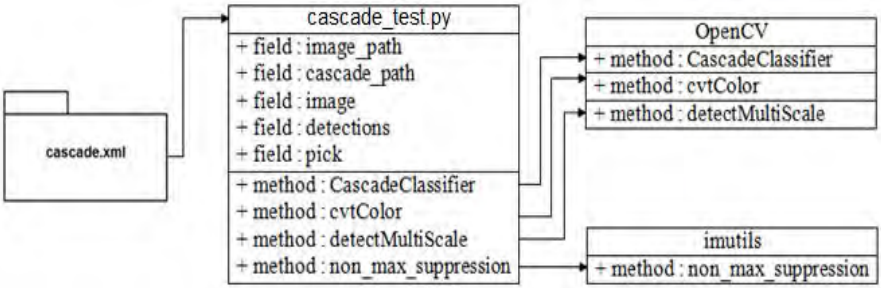


Рисунок 1.28 – Схема програми тестування каскадного класифікатора

Програмну реалізацію підбору параметрів навчання нейронних мереж за допомогою модифікацій генетичного алгоритму можна зобразити наступним чином (рис. 1.29). Файли даного проєкту містяться в окремій теці – ga_mod.

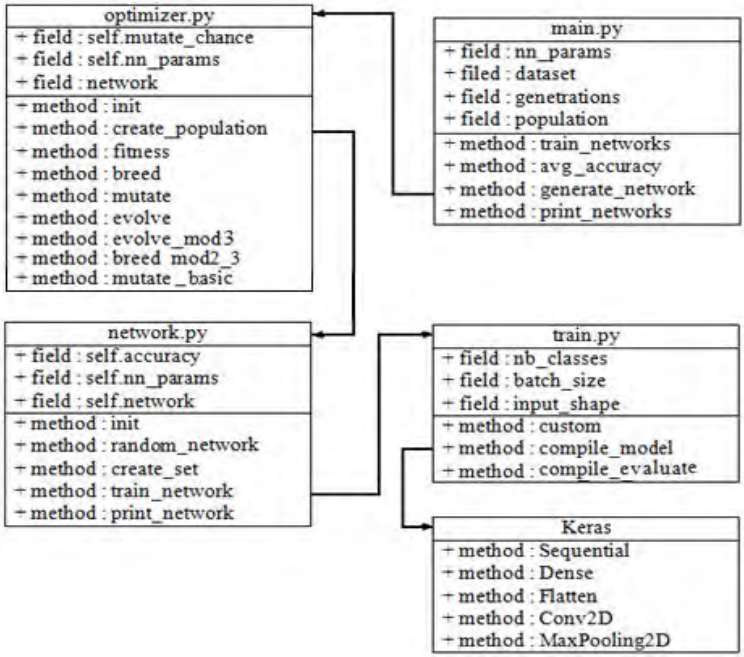


Рисунок 1.29 – Схема програми підбору параметрів для навчання нейронних мереж

Програмну реалізацію розпізнавання образів з використанням навченої нейронної мережі за допомогою модифікації генетичного алгоритму можна зобразити наступним чином (рис. 1.30).

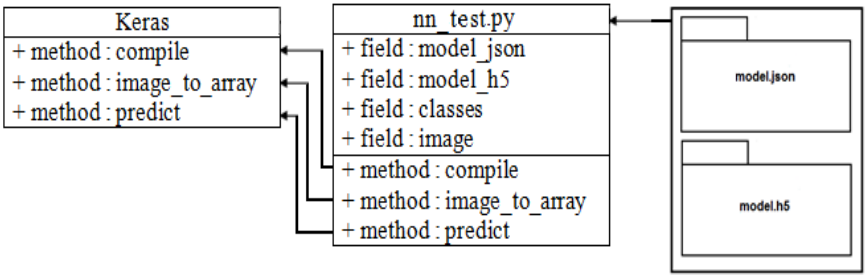


Рисунок 1.30 – Схема програми тестування нейронної мережі

Програмну реалізацію навчання згорткової нейронної мережі можна зобразити наступним чином (рис. 1.31).

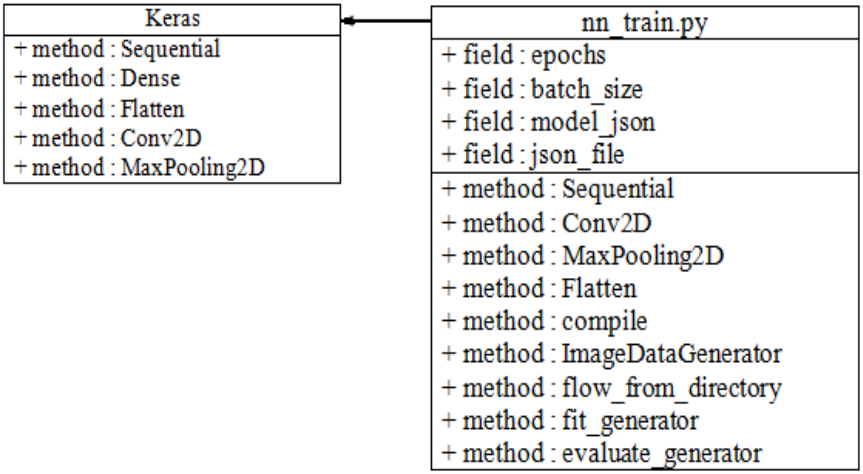


Рисунок 1.31 – Схема програми навчання нейронної мережі

1.7.1 Модулі розробленого програмного забезпечення

Розроблене програмне забезпечення являє собою набір програм для розпізнавання образів, який містить файли (скрипти, модулі) для розпізнавання образів за методом Віоли-Джонса та з використанням загорткової нейронної мережі, та скрипти навчання нейронної мережі та підбору параметрів при навчанні нейронних мереж.

Головний модуль програми – main.py, з якого виконується виклик програм перелічених вище. Даний модуль містить такі методи:

– initUI – метод створення вікна програми;

– disable_buttons – метод в якому обмежується можливість виклику програм;

– enable_buttons – метод в якому надається можливість виклику програм;

– error_message – метод виведення вікна помилки;

– info_message – метод виведення вікна повідомлення;

– cascade_button – метод виклику програми тестування детектування каскадного класифікатора;

– ga_button – метод виклику програми підбору параметрів при навчання нейронних мереж;

– test_button – метод виклику програми тестування розпізнавання нейронною мережею;

– train_button – метод виклику програми навчання нейронної мережі.

Програма розпізнавання образів з використанням каскадного класифікатора складається з двох файлів: cascade_test.py та cascade.xml.

Файл cascade_test.py – головний файл програми що використовує каскадний класифікатор для розпізнавання, в якому відбувається зчитування зображення та файлу каскадного класифікатора, відбувається обробка зображення та розпізнавання, і вивід результуючого зображення. Скрипт використовує такі методи:

– CascadeClassifier – метод зчитування каскадного класифікатора з бібліотеки OpenCV;

– imread – метод зчитування зображення з бібліотеки OpenCV;

– datetime.now – метод визначення поточного часу. Використовується для обчислення часу витраченого на розпізнавання;

– cvtColor – метод переведення зображення в інший колірний простір з бібліотеки OpenCV;

– detectMultiScale – метод розпізнавання об’єкту з бібліотеки OpenCV;

– non_max_suppression – метод що дозволяє зменшити похибку розпізнавання з бібліотеки imutils.

Файл cascade.xml – навчений каскадний класифікатор для розпізнавання об’єкту.

Програма підбору параметрів при навчанні нейронної мережі складається з таких файлів: main.py, network.py, optimizer.py, train.py, які містяться в теці ga_mod.

Файл main.py – головний файл програми, де відбувається ініціалізація значень параметрів що будуть використовуватися в процесі навчання нейронних мереж, виконується виклик навчання нейронних мереж. Скрипт має такі методи:

– train_networks – викликає метод train скрипту network.py та відображає прогрес навчання нейронних мереж;

– avg_accuracy – підраховується середній показник точності навчених мереж;

– generate_network – викликає методи навчання нейронних мереж та методи підбору параметрів зі скрипту optimizer.py;

– print_networks – виконується вивід інформації про нейронні мережі з кращими показниками;

– main – відбувається ініціалізація значень параметрів що будуть використовуватися в процесі навчання нейронних мереж, та виклик методу generate.

Файл network.py призначений для зберігання інформації про нейронну мережу. Містить такі методи:

– init – виконується ініціалізація мережі;

– random_network – виконується заповнення генів мережі випадковими значеннями;

– create_set – встановлює відповідній мережі значення згенерованих генів;

– train_network – викликає метод навчання нейронних мереж train_and_score зі скрипту train.py;

– print_network – виконує вивід інформації про показники навченої нейронної мережі.

Файл optimizer.py містить реалізацію модифікованого простого генетичного алгоритму для підбору оптимальних параметрів навчання нейронної мережі. Містить такі методи:

– init – виконується ініціалізація параметрів для підбору параметрів;

– create_population – створення популяції нейронних мереж;

– breed – виконується схрещення двох нейронних мереж (модифікація Альфа-Бета);

– breed_mod2_3 – виконується схрещення двох нейронних мереж (модифікації Альфа-Бета фіксована та Фіксована);

– mutate – виконується оператор мутації до генів нейронної мережі (дві мутації - базова та подвоююча);

– mutate_basic – виконується оператор мутації до генів нейронної мережі (одна мутація - базова);

– evolve – виконується відбір нейронних мереж в наступне покоління (модифікації Альфа-Бета та Альфа-Бета фіксована);

– evolve_mod3 – виконується відбір нейронних мереж в наступне покоління (модифікація Фіксована).

Файл train.py призначений для завантаження навчальної вибірки та безпосередньо навчання нейронної мережі. Містить наступні методи:

– custom – виконується завантаження навчальної вибірки;

– compile_model – виконується створення структури нейронної мережі;

– compile_evaluate – виконується навчання нейронної мережі.

Програма навчання нейронної мережі складається з файла nn_train.py. В даному скрипті відбувається завантаження навчальної вибірки та навчання нейронної мережі та її збереження до пам’яті комп’ютера. Архітектура навченої нейронної мережі зберігається до файлу model.json, а ваги мережі записуються до файлу model.h5. Скрипт використовує такі методи:

– compile – відбувається збірка нейронної мережі;

– ImageDataGenerator – відбувається генерація навчального набору з навчальної вибірки;

– flow_from_directory – відбувається перетворення зображень для подальшого використання;

– fit_generator – відбувається робота із зображеннями в процесі навчання;

– evaluate_generator – відбувається робота із зображеннями в процесі авто-тестування розпізнавання;

– save_weights – запис ваг нейронної мережі до файлу model.h5.

Програма розпізнавання образів з використанням нейронної мережі складається з файлу nn_test.py, який використовує файли архітектури мережі model.json та файл її ваг model.h5. В даному скрипті виконується зчитування вхідного зображення, переведення його у відповідний формат, та подання нейронній мережі для розпізнавання. Скрипт використовує такі методи:

– model_from_json – відбувається завантаження архітектури нейронної мережі з файлу model.json;

– load_weights – відбувається завантаження ваг нейронної мережі з файлу model.h5;

– compile – відбувається збірка нейронної мережі;

– load_img – завантаження зображення для розпізнавання;

– predict – виконується розпізнавання по вхідному зображенню.

В процесі навчання нейронних мереж було задіяно три вибірки, які містяться на комп'ютері: навчальна, перевірочна, тестова. Кожна вибірка містить 5 класів: пішохід, велосипед, мотоцикл, авто, вантажівка. Всі вибірки представляють собою набори зображень, що містять відповідний об'єкт. Кожна з навчальних вибірок складається з 1000 зображень, кожна перевірочна зі 400 зображень, кожна тестова вибірка з 200 зображень.

1.7.2 Характеристики програми

Програмний комплекс складається з чотирьох підпрограм – навчання нейронної мережі, тестування нейронної мережі, тестування каскадного класифікатора, підбору параметрів навчання нейронних мереж за допомогою простого генетичного алгоритму та його модифікаціями.

Підпрограма навчання нейронної мережі зчитує вказані параметри (кількість нейронів для кожного з трьох шарів, функція активації, функція оптимізації, кількість епох навчання) та зображення навчальної вибірки, та починає процес навчання нейронної мережі. Вивід деталей процесу навчання відображається в Terminal (Ubuntu), cmd (Windows). Після завершення навчання нейронної мережі, її ваги

та архітектура зберігаються до файлів model.h5 та model.json відповідно. Дана підпрограма складається з файла nn_train.py.

Підпрограма тестування нейронної мережі зчитує вказаний файл зображення, виконує обробку, та відображає вхідне зображення і надає в повідомленні результат розпізнавання: клас обєкту, впевненість та час виконання розпізнавання. Дана підпрограма складається з файлу nn_test.py, який використовує файли архітектури мережі model.json та файл її ваг model.h5.

Підпрограма тестування каскадного класифікатора зчитує вказаний файл зображення, виконує обробку, та відображає зображення з виділеним розпізнаним велосипедом у зеленій рамці, а якщо розпізнавання не вдалося – відображається вхідне зображення, також в повідомленні наводиться час виконання розпізнавання. Дана підпрограма складається з двох файлів: cascade_test.py та cascade.xml.

Підпрограма підбору параметрів навчання нейронних мереж за допомогою простого генетичного алгоритму та його модифікацій зчитує вказані параметри (версія ГА, кількість епох навчання) та починає процес навчання нейронної мережі. Вивід деталей процесу навчання відображається в Terminal (Ubuntu), cmd (Windows) та відбувається запис до файлу log.txt. Дана підпрограма складається з таких файлів: main.py, network.py, optimizer.py, train.py, які містяться в теці ga_mod.

1.7.3 Вхідні та вихідні дані

Вхідними даними для підпрограм програмного комплексу пов’язаних з розпізнаванням образів є файли зображень на яких необхідно розпізнати учасника дорожнього руху, для підпрограми навчання нейронної мережі вхідними даними є кількості нейронів на шарах, функції активації та оптимізації, кількість епох навчання, а також навчальна вибірка зображень(табл. 1.5). Для підпрограми підбору параметрів навчання за допомогою генетичного алгоритму та його модифікацій, вхідними даними є версія генетичного алгоритму (базова чи одна з трьох модифікацій), кількість епох навчання та навчальна вибірка зображень(табл. 1.6).

Також, у файлах nn_train.py та train.py (з каталогу ga_mod) у змінних train_data_dir (навчальна вибірка), validation_data_dir

(перевірочна вибірка), test_data_dir (тестова вибірка) повинні бути вказані вірні шляхи до навчальних вибірок зображень.

Таблиця 1.5 – Вхідні дані для навчання нейронних мереж та підбору параметрів

Параметр	Тип даних
Кількість нейронів у шарі	Ціле число (int) (можливі значення: 32, 64, 128)
Функція активації	Строка (string) (можливі значення: relu, elu, tanh, sigmoid)
Функція оптимізації	Строка (string) (можливі значення: rmsprop, adam, sgd, adagrad, adadelta, adamax, nadam)
Кількість епох навчання	Ціле число (int) (діапазон можливих значень: 30-100)
Версія генетичного алгоритму	Ціле число (int) (діапазон можливих значень: 0-3)

Таблиця 1.6 – Версії генетичного алгоритму та їх коди

Версія генетичного алгоритму	Код версії генетичного алгоритму
Базова	0
Модифікація Альфа-Бета	1
Модифікація Альфа-Бета фіксована	2
Модифікація Фіксована	3

Вихідними даними програмного комплексу є дані про поточні стани процесу навчання нейронної мережі та процесу підбору параметрів при навчанні нейронної мережі (рис. 1.32), файл log.txt, в який записуються проміжні та кінцевий результати (рис. 1.33), та

файли model.json і model.h5, що отримуються в результаті навчання нейронної мережі.

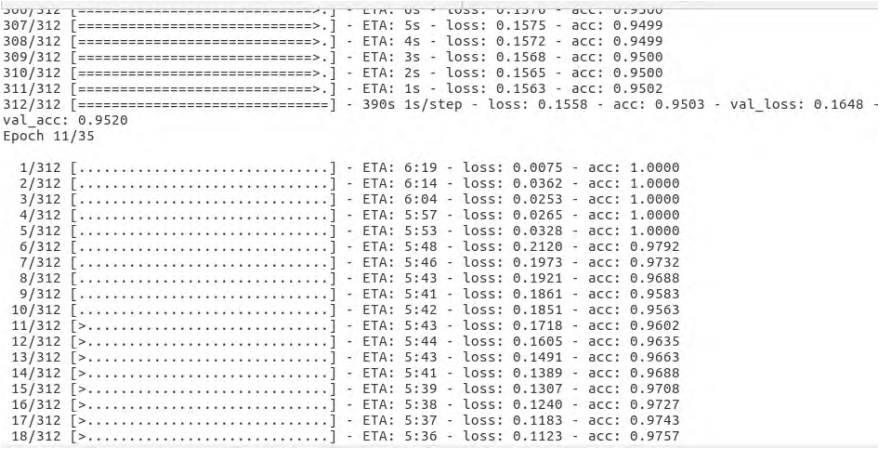


Рисунок 1.32 – Приклад виводу підпрограм навчання нейронної мережі

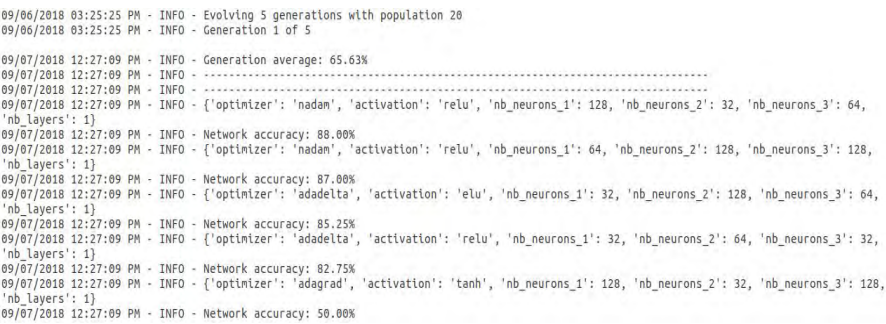


Рисунок 1.33 – Приклад вмісту файлу log.txt

1.7.4 Повідомлення

Програмний комплекс містить різноманітні інформаційні повідомлення та повідомлення про помилки. Також, слід відмітити, що в процесі роботи програмного комплексу варто звертати увагу на повідомлення що наводяться у утилітах cmd та Terminal (залежно від

операційної системи). Перелік всіх повідолень що містяться у програмному комплексі наведено у таблиці 1.7.

Таблиця 1.7 – Перелік повідомлень програмного комплексу

Повідомлення	Опис	Приклад
Об’єкт класу: _. Впевненість: [_]. Час розпізнавання: _.	Вивід інформації про розпізнавання об’єкта за допомогою нейронної мережі	Рис. 1.34
Час детектування: _ секунд.	Вивід інформації про розпізнавання за допомогою каскадного класифікатора	Рис. 1.35
Запуск навчання НМ. Деталі в терміналі.	Повідомлення про початок навчання нейронної мережі	Рис. 1.36
Запуск підбору параметрів навчання НМ за допомогою модифікації ГА. Деталі в терміналі та файлі log.txt.	Повідомлення про початок підбору параметрів параметрів навчання нейронної мережі за допомогою генетичного алгоритму	Рис. 1.37
Помилка відкриття файлу. Перевірте наявність файлів model.json, model.h5.	Повідомлення про помилку в ході виконання розпізнавання за допомогою нейронної мережі	Рис. 1.38
Помилка відкриття файлу. Перевірте наявність файлу cascade.xml	Повідомлення про помилку в ході виконання розпізнавання за допомогою каскадного класифікатора	Рис. 1.39
Перевірте правильність завдання параметрів нейронної мережі	Повідомлення про помилку в ході завдання параметрів навчання нейронної мережі	Рис. 1.40

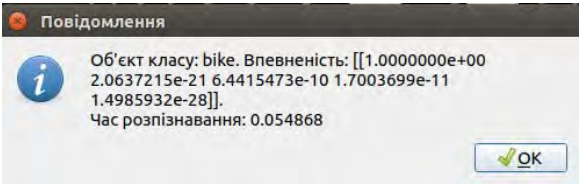


Рисунок 1.34 – Приклад повідомлення розпізнавання за допомогою нейронної мережі

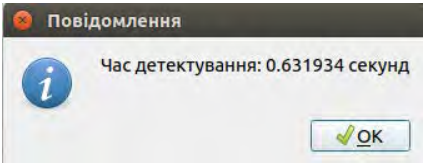


Рисунок 1.35 – Приклад повідомлення про розпізнавання
за допомогою каскадного класифікатора

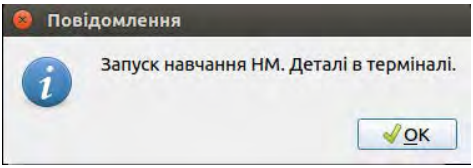


Рисунок 1.36 – Повідомлення про початок навчання нейронної мережі

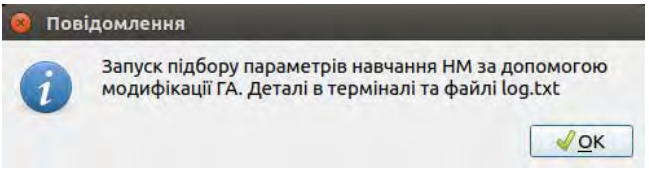


Рисунок 1.37 – Повідомлення про початок підбору параметрів параметрів навчання
нейронної мережі за допомогою генетичного алгоритму

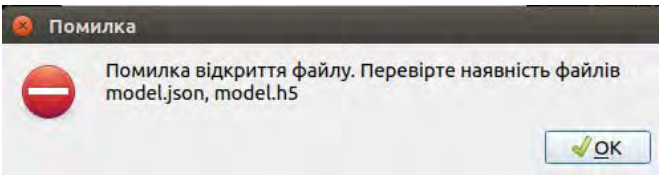


Рисунок 1.38 – Помилка розпізнавання за допомогою нейронної мережі

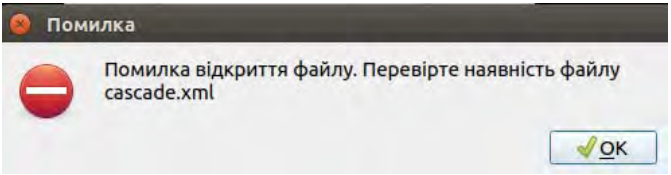


Рисунок 1.39 – Помилка розпізнавання за допомогою каскадного класифікатора

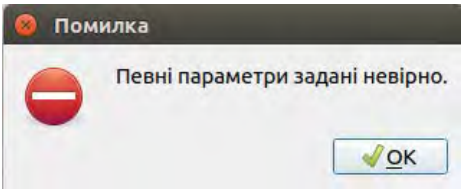


Рисунок 1.40 – Помилка у заданих параметрах навчання нейронної мережі

1.7.5 Розпізнавання учасників дорожнього руху за допомогою нейронної мережі

Алгоритм розпізнавання учасників дорожнього руху реалізовано в підпрограмі nn_test.py.

На вхід підпрограмі надходить зображення, обране користувачем. Далі завантажуються ваги натренованої нейронної мережі та її архітектура (файли model.h5, model.json), та виконується збірка нейронної мережі. Здійснюється зміна розміру зображення обраного користувачем, відповідно до значення за яким було навчено нейронну мережу (150×150). Далі зображення переводиться у тривимірний масив, та проводиться його нормалізація. Далі за вхідними вагами виконується розпізнавання нейронною мережею. Як результат отримується ім’я класу об’єкту, та масив дійсних чисел (наприклад, 9,99, що відповідає 99,9%), який описує ймовірнісне відношення об’єкту на зображенні до певного класу.

1.7.6 Призначення і умови виконання програми

Розроблений програмний комплекс призначений для проведення досліджень та для роботи з підходами до розпізнавання образів – навчання нейронних мереж, підбору параметрів при навчанні

нейронних мереж за допомогою генетичного алгоритму та його модифікацій, тестування розпізнавання учасників дорожнього руху навченим нейронними мережами та каскадного класифікатора навченого детектувати велосипеди.

1.7.7 Виконання програми

Звертання до програмного комплексу можливе за допомогою утиліти cmd у Windows або Terminal у Unix – необхідно перейти до каталогу з файлами проєкту, та виконати команду python3 main.py.

Перед викликом програми слід переконатися, що у файлах nn_train.py та ga_mod/train.py (відповідають програмам «Навчання НМ» та «Підбір параметрів НМ з ГА») у змінних train_data_dir (навчальна вибірка), validation_data_dir (перевірочна вибірка), test_data_dir (тестова вибірка), вказані вірні шляхи до навчальних вибірок зображень.

В результаті виконання команди відобразиться вікно програмного комплексу (рис. 1.41), яке містить чотири компоненти «кнопка», які мають наступне значення:

– тестування НМ – запуск підпрограми тестування розпізнавання навченої нейронної мережі. Необхідно обрати зображення для розпізнавання;

– навчання НМ – запуск підпрограми навчання нейронної мережі.

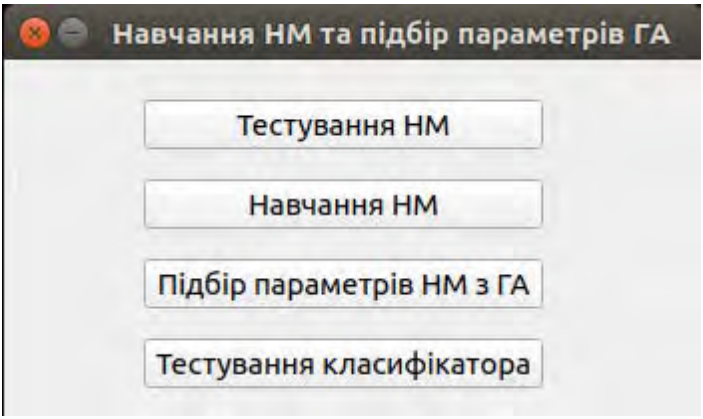


Рисунок 1.41 – Вікно програмного комплексу

Необхідно ввести кількості нейронів для трьох шарів, функції активації та оптимізації, кількість епох навчання;

– підбір параметрів НМ з ГА – запуск підпрограми підбору параметрів навчання нейронної мережі за допомогою простого генетичного алгоритму чи його модифікації. Необхідно ввести бажану версію генетичного алгоритму та кількість епох навчання;

– тестування класифікатора – запуск підпрограми тестування розпізнавання за допомогою каскадного класифікатора. Необхідно обрати зображення для розпізнавання.

Після натиснення кнопки «Тестування НМ» відобразиться стандартне вікно вибору зображення, слід перейти до каталогу з бажаним файлом та обрати його. Після цього підпрограма тестування нейронної мережі зчитує вказаний файл зображення, виконує обробку, та відображає вхідне зображення (рис. 1.42) і надає в повідомленні результат розпізнавання: клас обєкту, впевненість та час виконання розпізнавання (рис. 1.34).

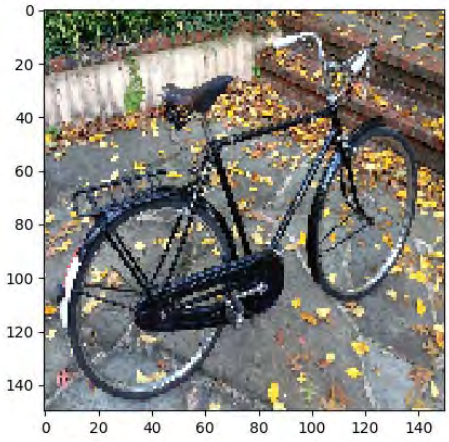


Рисунок 1.42 – Приклад зображення для розпізнавання

Після натиснення кнопки «Навчання НМ» послідовно оператору будуть відображатися вікна введення кількості нейронів для трьох шарів (рис. 1.43), функцій активації (рис. 1.44) та оптимізації (рис. 1.45), а також кількості епох навчання (рис. 1.46).

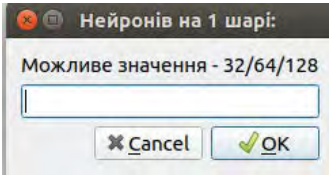


Рисунок 1.43 – Введення кількості нейронів 1 шару

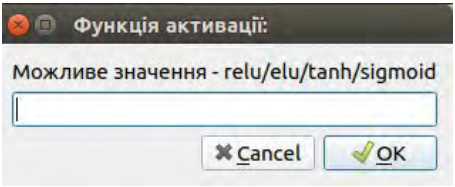


Рисунок 1.44 – Введення функції активації

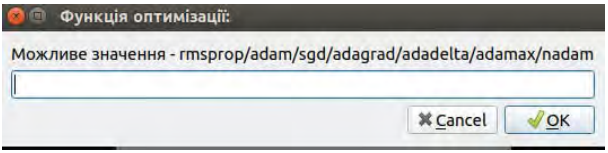


Рисунок 1.45 – Введення функції оптимізації

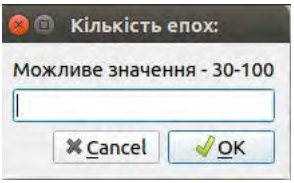


Рисунок 1.46 – Введення кількості епох навчання

Після того, як всі параметри було введено без помилок, відобразиться повідомлення про початок навчання нейронної мережі (рис. 1.36). Інформація про хід процесу навчання буде відображатися в cmd чи Terminal (залежно від ОС) (рис. 1.32). Після завершення навчання в каталозі проєкту з'являться файли model.json та model.h5, файл архітектури та ваг нейронної мережі відповідно.

Після натиснення кнопки «Підбір параметрів НМ з ГА» послідовно оператору відобразяться вікна введення версії генетичного алгоритму (рис. 1.47) та кількості епох навчання (рис. 1.48).

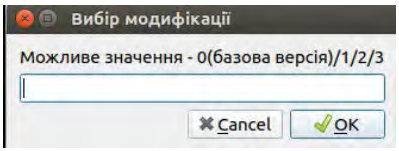


Рисунок 1.47 – Введення версії генетичного алгоритму

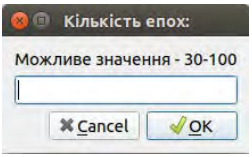


Рисунок 1.48 – Введення кількості епох навчання

Після того, як всі параметри було введено без помилок, відобразиться повідомлення про початок підбору параметрів навчання нейронної мережі (рис. 1.37). Інформація про хід процесу підбору буде відображатися в cmd чи Terminal (залежно від ОС) (рис. 1.32) та проміжні і кінцеві результати будуть записуватися у файл log.txt (рис. 1.33). Після завершення підбору наприкінці файлу log.txt буде наведено показники кращої особини – нейронної мережі: функції оптимізації та активації, кількості нейронів на трьох шарах.

Після натиснення кнопки «Тестування класифікатора» відобразиться стандартне вікно вибору зображення, слід перейти до каталогу з бажаним файлом та обрати його. Після цього підпрограма тестування каскадного класифікатора зчитує вказаний файл зображення, виконує обробку, та відображає вхідне зображення з виділеним зеленим прямокутником у разі успішного розпізнавання (рис. 1.49), інакше відображається вхідне зображення без прямокутника, і надає в повідомленні час виконання розпізнавання (рис. 1.35).

Рисунок 1.49 – Результат тестування каскадного класифікатора

Програмний комплекс містить різноманітні інформаційні повідомлення та повідомлення про помилки. Також, слід відмітити, що в процесі роботи програмного комплексу варто звертати увагу на повідомлення що наводяться у утилітах cmd та Terminal (залежно від операційної системи). Перелік всіх повідолень що містяться у програмному комплексі наведено в таблиці 1.7.

1.8 Експерименти і результати

1.8.1 Експерименти по дослідженню розроблених генетичних методів розв'язання задачі розпізнавання графічних образів

Для навчання нейронних мереж використовувалась навчальна вибірка, яка складалася із 1000000 зображень, розділених на 5 класів.

Було проведено 4 експерименти, в яких було використано розроблені модифікації генетичного алгоритму та безпосередньо простий генетичний алгоритм. Всі експерименти були проведені з однаковими параметрами: кількість поколінь – 5, розмір популяції – 20, коефіцієнт мутації – 0,01. Вхідні дані – вибірка зображень 5 класів (пішохід, велосипед, мотоцикл, авто, вантажівка).

Результати експериментів з порівняння простого генетичного методу та розроблених модифікацій наведено у табл. 1.8.

Таблиця 1.8 – Результати підборів параметрів за допомогою простого генетичного алгоритму

Версія ГА	Час навчання		Точність на тестових даних	Точність авто-тестування
	Фактичний	У відсотках до базової версії ГА		
Базова	8 днів 8 хвилин 39 секунд	100 %	89,45 %	93,89 %
Модифікація Альфа-Бета	5 днів 22 години 27 хвилин 5 секунд	73,9 %	92,12 %	96,90 %
Модифікація Альфа-Бета фіксована	8 днів 12 годин 4 хвилини 58 секунд	106,2 %	90,02 %	95,89 %
Модифікація фіксована	7 днів 7 годин 16 хвилин 43 секунди	91,1 %	92,48 %	95,72 %

Виходячи результатів, наведених в табл. 1.8, можна зробити висновок, що використання модифікації Альфа-Бета генетичного алгоритму є кращим підходом для досягнення вищої точності розпізнавання за менший час.

У табл. 1.9 наведено результати розпізнавання образів при використанні різних підходів для навчання розпізнавальних моделей.

Виходячи з таблиці 1.9, можна зробити висновок, що зменшення кількості класів для навчання зі 100 до 5 значно підвищило показник точності, проте час виконання підбору дещо збільшився. Також, порівнюючи результати отримані при виконанні підбору параметрів з використанням простого генетичного алгоритму та його модифікацій, можна зробити висновок, що кращим підходом до оптимізації процесу підбору параметрів навчання нейронної мережі є використання модифікації генетичного алгоритму Альфа-Бета.

Як видно з таблиць 1.8 та 1.9, розроблені модифікації простого генетичного методу дозволяють підвищити швидкість синтезу розпізнавальних моделей відносно базової версії ГА

Таблиця 1.9 – Порівняння результатів розпізнавання образів при використанні різних підходів для навчання розпізнавальних моделей

Характеристика Підхід хід	Час навчання	Точність на тестових даних	Точність автотестування
Каскадний класифікатор	8 днів 15 годин 36 хвилин	90,95 %	–
Повнозв'язна нейронна мережа (100 класів, без використання ГА)	1 година 4 хвилини 19 секунд	1,2 %	24,93 %
Згорткова нейронна мережа (100 класів, без використання ГА)	1 година 59 хвилин 34 секунди	1,8 %	47,14 %
Повнозв'язна нейронна мережа (100 класів, з використанням ГА)	22 години 47 хвилин 33 секунди	1,4 %	30,52 %
Згорткова нейронна мережа (100 класів, з використанням ГА, 4 шари)	4 дні 16 годин 59 хвилини 21 секунда	2,1%	48%
Згорткова нейронна мережа (100 класів, з використанням ГА, 6 шарів)	6 днів 12 годин 51 хвилина 42 секунди	2,39%	53,69%
Згорткова нейронна мережа (100 класів, з використанням ГА, 6 шарів)	6 днів 12 годин 51 хвилина 42 секунди	2,39%	53,69%
Згорткова нейронна мережа (5 класів, з використанням ГА, 3 шари)	8 днів 8 хвилин 39 секунд	89,45 %	93,89 %
Згорткова нейронна мережа (5 класів, з використанням модифікації ГА Альфа-Бета, 3 шари)	5 днів 22 години 27 хвилин 5 секунд	92,12 %	96,90 %
Згорткова нейронна мережа (5 класів, з використанням модифікації ГА Альфа-Бета фіксована, 3 шари)	8 днів 12 годин 4 хвилини 58 секунд	90,02 %	95,89 %
Згорткова нейронна мережа (5 класів, з використанням модифікації ГА фіксована, 3 шари)	7 днів 7 годин 16 хвилин 43 секунди	92,48 %	95,72 %

Зокрема, при використанні модифікації Альфа-Бета витрачається лише 73,9 % часу базового методу, при використанні модифікації фіксована – 91,1 % часу базового генетичного методу).

Це дозволило зменшити час оброблення експериментального масиву даних за допомогою модифікованого алгоритму Альфа-Бета на 58 годин у порівнянні з базовим методом.

Як видно з таблиці 1.8, найкращим методом за швидкістю рішення обраної задачі є повнозв'язна нейронна мережа, яка має наступні характеристики: 1 година 4 хвилини 19 секунд, проте точність розпізнання у неї складає лише 24,93%. Повнозв’язна нейронна мережа з використанням ГА має наступні характеристики: час навчання 22 години 47 хвилин 33 секунди, проте точність розпізнавання складає лише 30,52 %. Розроблені модифікації простого генетичного методу дозволили отримати точність розпізнавання на тестових даних 95–96 %, що є досить прийнятним результатом.

Такі результати обумовлюються використанням розроблених евристичних процедур, зокрема мутацій з використанням методу Монте-Карло, модифікованих операторів відбору та схрещування. Це дозволило підвищити показник точності та зменшити час оптимізації за допомогою розроблених модифікацій простого генетичного методу у порівнянні з його базовою версією.

Виходячи з результатів розпізнавання можна зробити висновок, що навчена нейронна мережа здатна з досить високою точністю визначати належність об’єкта до певного класу. Таким чином, розроблені модифікації простого генетичного алгоритму дозволяють підвищити точність розпізнавання та зменшити час навчання. Це досягається за рахунок використання у розроблених модифікованих методах нових евристичних процедур, зокрема додано можливість мутацій з використанням методу Монте-Карло, модифіковано оператори відбору та схрещування. Це дозволило підвищити показник точності у порівнянні з базовою версією простого генетичного алгоритму.

Недоліком модифікацій простого генетичного алгоритму, розроблених та досліджених у цій роботі, є необхідність витрачання великого часу (декілька діб) при обробці великих масивів даних, що при розв’язанні деяких практичних завдань є неприпустимим. Таким чином обмеженнями на використання розроблених модифікацій є невеликі обсяги оброблюваних даних.

1.8.2 Результати роботи методів Віоли-Джонса та гістограм орієнтованих градієнтів

В ході виконання роботи на початковому етапі було порівняно результати роботи методів Віоли-Джонса та гістограм орієнтованих градієнтів на прикладі розпізнавання людей, оскільки в бібліотеці OpenCV містяться вже реалізовані каскадний класифікатор та функції для розпізнавання людей. Тестування проводилось на 200 зображеннях різних розмірів, на комп'ютері з процесором Intel Core i5-7400 з тактовою частотою 3 ГГц. Порівняння результатів розпізнавання методами Віоли-Джонса та гістограм орієнтованих градієнтів наведено у таблиці 1.10.

Таблиця 1.10 – Порівняння результатів розпізнавання методів Віоли-Джонса та гістограм орієнтованих градієнтів

	Віоли-Джонса		HOG	
Розпізнано	162	81%	152	76%
Не розпізнано	38	19%	48	24%
Середній час розпізнавання (секунд)	0,22606		0,40783	

За результатами наведеними у таблиці 1.10 можна зробити висновок, що використання методу Віоли-Джонса є ефективнішим з точки зору часу та точності розпізнавання, тому перевагу в подальшому дослідженні реалізації розпізнавання образів було надано саме цьому методу.

1.8.3 Результати навчання каскадного класифікатора за методом Віоли–Джонса

В ході виконання курсового проєкту були проведені експерименти з реалізації навчання та тестування каскадного класифікатора за методом Віоли-Джонса за ознаками Хаара та оператором LBP. Результати експериментів навчання та тестування наведено в таблицях 1.11 та 1.12. Тестування проводилось на 210 зображеннях різних розмірів, що не використовувались в процесі навчання, на комп'ютері з процесором Intel Core i5-7400 з тактовою частотою 3 ГГц.

Таблиця 1.11 – Показники навчання каскадних класифікаторів з ознаками HAAR та оператором LBP

Рівень	Етапи		Час	
	HAAR	LBP	HAAR	LBP
0	20	11	2 год. 14 хв.	2 хв. 30 с.
5	32	13	9 год. 56 хв.	8 хв. 45 с.
10	42	13	3 доби 13 год. 8 хв.	18 хв. 13 с.
15	48	13	6 діб 1 год. 20 хв.	33 хв. 47 с.
19	65	15	8 діб 15 год. 36 хв.	2 год. 3 хв. 32 с.

Таблиця 1.12 – Порівняння результатів тестування каскадів з ознаками HAAR та оператором LBP

Характеристика	HAAR		LBP		HAAR (50%)		LBP (50%)	
Розпізнано	191	90,95%	182	86,66%	187	89,05%	168	80%
Не розпізнано	19	9,05%	28	13,33%	23	10,95%	42	20%
Середній час розпізнавання (с)	0,3269		0,1826		0,0818		0,0486	

Виходячи з таблиць 1.11 та 1.12, можна зробити висновок, що використання ознак Хаара є більш доцільним, оскільки надає більшу точність. Для використання у вбудовуваних пристроях (embedded device), можливо доцільніше буде використовувати оператор LBP, оскільки час опрацювання зображення менший, отже точність може буде вища, ніж при використанні каскаду з ознаками Хаара.

Залежно від параметрів, а також від кількості зображень, тривалість навчання за ознаками Хаара може варіюватися від декількох днів до тижнів. Навчання за заданими параметрами і з використанням 3100 «позитивних» і 6200 «негативних» зображень тривало 8 днів, в той час як за цими ж вибірками тривалість навчання класифікатору з оператором LBP склала близько 2 годин.

1.8.4 Результати навчання нейронних мереж

В експериментах було розглянуто реалізацію навчання згорткової та повнозв’язної нейронних мереж на базі зображень CIFAR100. В таблиці 1.13 наведено порівняння показників навчання повнозв’язої та згорткової нейронних мереж.

Для підвищення точності розпізнавання, для повнозв’язної та згорткової нейронних мереж було виконано підбір параметрів за допомогою простого генетичного алгоритму.

Результати підбору параметрів наведено в таблиці 1.14.

Таблиця 1.13 – Порівняння показників навчання нейронних мереж

Характеристика	Повнозв'язна НМ	Згорткова НМ
Час навчання	1 год. 4 хв. 19 с.	1 год. 59 хв. 34 с.
Точність	24,93%	47,14%

Таблиця 1.14 – Порівняння показників підбору параметрів навчання нейронних мереж

	Повнозв'язна НМ	Згорткова НМ (4 шари)	Згорткова НМ (6 шарів)
Час підбору	22 год. 47 хв. 33 с.	4 доби 16 год. 59 хв. 21 с.	6 діб 12 год. 51 хв. 42 с.
Точність	30,52%	48%	53,69%

На рисунку 1.50 наведено графік середніх показників точності для кожного покоління підбору параметрів навчання нейронних мереж. На рисунку 1.51 наведено графік показників 5 кращих особин останнього покоління підбору параметрів для кожної нейронної мережі.

Порівнюючи показники навчання нейронних мереж з таблиць 1.13 та 1.14, можна зробити висновок, що простий генетичний алгоритм допоміг поліпшити показники точності для обох нейронних мереж, але тим не менш, даний показник лишається незадовільно низьким для використання в системах розпізнавання велосипедів.

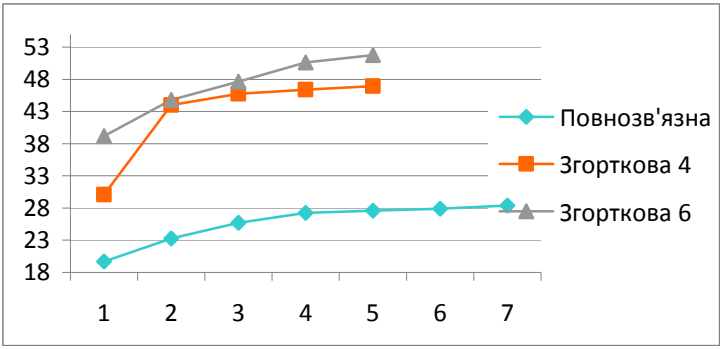


Рисунок 1.50 – Графік середніх показників точності при підборі параметів для навчання нейронних мереж

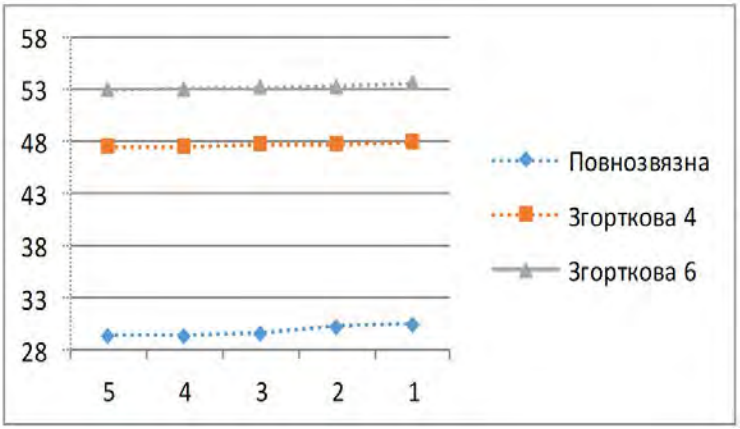


Рисунок 1.51 – Графік показників точності останнього покоління при підборі параметів для навчання нейронних мереж

Досягти більш високих показників точності можливо при використанні більших значень параметрів генетичного алгоритму та нейронних мереж, для чого потрібні більші обчислювальні можливості, та, можливо, при використанні навчальної вибірки з меншою кількістю класів [47-50].

1.8.5 Результати розпізнавання за допомогою навченої нейронної мережі

Після завершення підбору параметрів навчання нейронної мережі за допомогою генетичного алгоритму та його модифікацій, було визначено що кращі показники точності та часу досягаються при використанні модифікації Альфа-Бета. За підібраними параметрами окремо було навчено згорткову нейронну мережу, та було проведено ряд експериментів з розпізнавання образів учасників дорожнього руху. Приклади розпізнавання наведено на рисунках 1.52–1.55.

Рисунок 1.52 – Приклад розпізнавання об’єкту «велосипед»

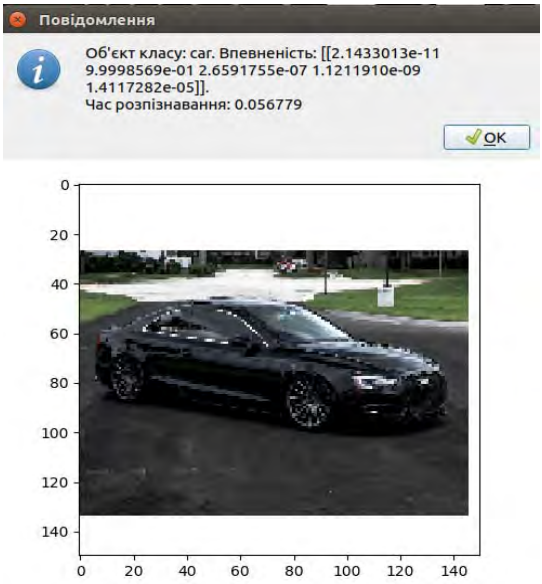


Рисунок 1.52 – Приклад розпізнавання об’єкту «авто»

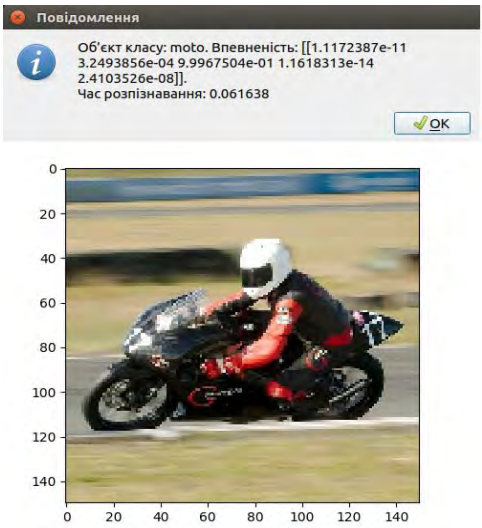


Рисунок 1.53 – Приклад розпізнавання об’єкту «мотоцикл»

Рисунок 1.54 – Приклади розпізнавання об'єкту «пішохід»

Рисунок 1.55 – Приклад розпізнавання об'єкту «вантажівка»

1.9 Висновки за розділом 1

Досліджено методи та засоби розпізнавання образів: метод Віоли-Джонса для навчання каскадного класифікатору, метод гістограм орієнтованих градієнтів, повнозв'язна та згорткова нейронні мережі та комбінація методів – нейронні мережі і генетичні алгоритми. Виконано порівняння показників вказаних підходів, та зроблено висновок щодо доцільності використання кожного підходу. Для підвищення показників нейронних мереж було застосовано простий генетичний алгоритм.

У результаті порівняння показників точності навчання нейронних мереж, було зроблено висновок, що найбільш ефективною, з точки зору точності розпізнавання є згорткова нейронна мережа (convolutional neural network).

Розроблено модифікації простого генетичного методу. В запропонованій модифікації Альфа-Бета було додано можливість виникнення двох мутацій, а також модифіковано оператор відбору особин для схрещення. В модифікації «Альфа-Бета фіксована» було додано можливість виникнення двох мутацій, додано фіксовану точку схрещення, а також змінено відбір особин для схрещення. В модифікації Фіксована було додано можливість виникнення двох мутацій та додано фіксовану точку схрещення. Запропоновані модифікації "Альфа-Бета" та "Альфа-Бета фіксована" дозволяють підвищити результуючий показник точності, в порівнянні з базовою версією простого генетичного алгоритму. В той же час "Альфа-Бета" та "Фіксована" дозволяють отримати результати підбору за менші витрати часу порівняно з базовою версією простого генетичного алгоритму.

У результаті порівняння показників точності отриманих за допомогою використання модифікацій генетичного алгоритму, зроблено висновок що найкращий показник отримано при використанні модифікації Альфа-Бета, яку запропоновано використовувати для вирішення задачі оптимізації підбору параметрів навчання нейронних мереж.

В ході дослідження було проведено ряд експериментів. За результатами експериментів виявлено що, застосування простого генетичного алгоритму при підборі параметрів навчання згорткової

нейронної мережі є менш ефективним, а набільш ефективним є використання його запропонованої модифікації Альфа-Бета, яка дозволила пришвидшити виконання процесу підбору параметрів навчання нейронних мереж (кількість нейронів на шарах, функції активації та оптимізації), та підвищити результуючий показник точності порівняно з базовою версією простого генетичного алгоритму.

1.10 Література до розділу 1

1. Эксперт робототехники: «Никогда не используйте автопилот Tesla рядом с велосипедистами!» [Электронный ресурс]. – Режим доступа: https://itc.ua/news/ekspert-robototehniki-nikogda-ne-ispolzuyte-avtopilot-tesla-ryadom-s-velosipedistami/.
2. Фомин, Я. А. Распознавание образов: теория и применения. – 2-е изд. / Я. А. Фомин – М.: ФАЗИС, 2012. – 429 с.
3. Системи Mazda i-Activsense [Електронний ресурс]. – Режим доступу: http://mazda.ua/ua/showroom/cx-5/i-activsense/.
4. Waymo [Electronic resource]. – Access mode: https://waymo.com.
5. Mazda Innovation [Електронний ресурс]. – Режим доступу: http://mazdamotors.kh.ua/ua/showroom/mazda3/innovation/.
6. Mazda i-Activsense [Electronic resource]. – Access mode: http://www.mazda.com/en/innovation/technology/safety/i-activsense/.
7. Tesla Autopilot [Electronic resource]. – Access mode: https://www.tesla.com/autopilot.
8. Waymo Technology [Electronic resource]. – Access mode: https://waymo.com/tech/.
9. Blippar [Electronic resource]. – Access mode: https://www.blippar.com.
10. Blippar создала аналог Shazam для автомобилей [Электронный ресурс]. – Режим доступа: http://autonews.autoua.net/novosti/16224-blippar-sozdala-analog-shazam-dlya-avtomobilej.html.
11. Blippar Computer Vision API [Electronic resource]. – Access mode: https://www.blippar.com/car-recognition-api.
12. Blippar выпустил AR-сервис для распознавания автомобилей [Электронный ресурс]. – Режим доступа: https://apparat.cc/news/identify-cars-blippar/.

13. Mobileye [Electronic resource] – Access mode: https://www.mobileye.com/.
14. The Evolution of EyeQ [Electronic resource]. – Access mode: https://www.mobileye.com/our-technology/evolution-eyeq-chip/.
15. Self-driving car [Electronic resource]. – Access mode: https://www.theverge.com/autonomous-cars.
16. Mobileye Продукти [Електронний ресурс]. – Режим доступу: http://mobile-eye.com.ua/?page_id=297.
17. Mobileye 5-та серія [Електронний ресурс]. – Режим доступу: http://mobile-eye.com.ua/?page_id=2548.
18. Mobileye 560 [Електронний ресурс]. – Режим доступу: http://mobile-eye.com.ua/?page_id=2558.
19. Mobileye Our Technology [Electronic resource]. – Access mode: https://www.mobileye.com/our-technology/.
20. Mobileye Smartphone Application [Електронний ресурс]. – Режим доступу: http://mobile-eye.com.ua/?page_id=2647.
21. Алгоритмічно-програмні засоби розпізнавання рукописних символів на зображенні [Електронний ресурс]. – Режим доступу: http://ena.lp.edu.ua:8080/bitstream/ntb/42829/2/2017n881_Paramud_Y-Algorithmic_and_software_98-106.pdf.
22. The MNIST Database of handwritten digits [Electronic resource]. – Access mode: http://yann.lecun.com/exdb/mnist/.
23. Распознавание объектов на Python / Глубокое машинное обучение [Електронний ресурс]. – Режим доступу: https://itproger.com/news/174.
24. CIFAR-10 and CIFAR-100 datasets [Electronic resource]. – Access mode: https://www.cs.toronto.edu/~kriz/cifar.html.
25. Repository neural-network-genetic-algorithm користувача harvitronix [Electronic resource]. – Access mode: https://github.com/harvitronix/neural-network-genetic-algorithm.
26. Viola P. Rapid object detection using a boosted cascade of simple features / P. Viola, M. Jones // Computer Vision and Pattern Recognition (CVPR 2001): 2001 IEEE Computer Society Conference, Kauai, 8-14 Dec. 2001: proceedings. – Los Alamitos: IEEE, 2001. – Vol. 1. – P. 511–518. DOI: 10.1109/CVPR.2001.990517.
27. Viola P. Robust real-time face detection / P. Viola, M.J. Jones // International Journal of Computer Vision. – 2004. – Vol. 57. – № 2. – P. 137–154. DOI: 10.1023/B:VISI.0000013087.49260.fb

28. Balali V. Segmentation and recognition of roadway assets from car-mounted camera video streams using a scalable non-parametric image parsing method / V. Balali, M. Golparvar-Fard // Automation in Construction. – 2015. – Vol. 49. – P. 27-39. doi: 10.1016/j.autcon.2014.09.007

29. Saini M. Multiwavelet transform based license plate detection / M. Saini, S. Saini // Journal of Visual Communication and Image Representation. – 2017. – Vol. 44. – P. 128-138. doi: 10.1016/j.jvcir.2017.01.003

30. Gavrila D.M. Real-time object detection for “smart” vehicles / D.M. Gavrila, V. Philomin // International Conference on Computer Vision, Seventh, 20-27 Sept. 1999: proceedings. – Los Alamitos: IEEE, 1999. – Vol. 1. – P. 87–93. DOI: 10.1109/ICCV.1999.791202.

31. Wang P. Reading car license plates using deep neural networks / H. Li, P. Wang, M. You, C. Shen // Image and Vision Computing. – 2018. – Vol. 72. – P. 14-23. doi: 10.1016/j.imavis.2018.02.002

32. Schapire R.E. A decision-theoretic generalization of on-line learning and an application to boosting / R.E. Schapire, Y. Freund // Journal of Computer and System Sciences. – 1997. – Vol. 55. – P. 119-139.

33. Schapire R.E. Improved Boosting Algorithms Using Confidence-rated Predictions / R.E. Schapire, Y. Singer // Machine Learning. – 1999. – Vol. 37. – P. 297-336.

34. Wei Y. Efficient Histogram-Based Sliding Window / Y. Wei, L. Tao // IEEE CVPR. – 2010. – P. 3003-3010.

35. Kusuma G. A Review of Recent Advancements in Appearance-based Object Recognition / G. Kusuma, E. Wigati, E. Chandra // Procedia Computer Science. – 2019. –Vol. 157. – P. 613-620. doi: 10.1016/j.procs.2019.08.227

36. Ren S. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks / S. Ren, K. He, R. Girshick, J. Sun // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2017. – Vol. 39. – P. 1137-1149. doi: 10.1109/tpami.2016.2577031

37. Zeiler M. D. ADADELTA: An Adaptive Learning Rate Method [Electronic resource] – Access mode: https://arxiv.org/pdf/1212.5701.pdf.

38. Sangineto E. A Self Paced Deep Learning for Weakly Supervised Object Detection / E. Sangineto, M. Nabi, D. Culibrk, N. Sebe // IEEE

Transactions on Pattern Analysis and Machine Intelligence. – 2019. – Vol. 41. – P. 712-725. doi: 10.1109/tpami.2018.2804907

39. Incorporating Nesterov Momentum Into Adam [Electronic resource]. – Access mode: https://openreview.net/pdf?id= OM0jvwB8jIp57ZJjtNEZ.

40. Convolution Neural Networks vs Fully Connected Neural Networks [Electronic resource]. – Access mode: https://medium.com/datadriveninvestor/convolution-neural-networks-vs-fully-connected-neural-networks-8171a6e86f15.

41. Loussaief S. Convolutional neural network hyper-parameters optimization based on genetic algorithms / S. Loussaief, A. Abdelkrim // International Journal of Advanced Computer Science and Applications. – 2018. – Vol. 9, Issue 10 – P. 252 – 266. doi: 10.14569/IJACSA.2018.091031.

42. Метод Виолы-Джонса [Электронный ресурс]. – Режим доступа: https://oxozle.com/2015/04/11/metod-raspoznavaniya-lic-violy-dzhonsa-viola-jones/.

43. Kassani P. H. A new sparse model for traffic sign classification using soft histogram of oriented gradients / P. H. Kassani, Teoh, B. J. // Applied Soft Computing Journal. – 2017. – Vol. 52 – P. . 231-246. doi: 10.1016/j.asoc.2016.12.037

44. Правило навчання ШНМ [Електронний ресурс]. – Режим доступу:http://opticstoday.com/katalog-statej/stati-na-ukrainskom/ nejromerezhi/ navchannya-shtuchnix-nejronnix-merezh.html.

45. Szandala T. A Comparison of Different Learning Algorithms for Pattern Recognition with Hopfield's Neural Network / T. Szandala // Procedia Computer Science. – 2015. – Vol. 71. – P. 68-75. doi: 10.1016/j.procs.2015.12.205

46. Кононюк А. Е. Дискретно-непрерывная математика. (Алгоритмы) / А. Е. Кононюк. – Кн. 10, Ч.3 – К.: «Освіта України», 2017. – 444 с.

47. Cascade Classifier Training - OpenCV Documentation [Electronic resource]. – Access mode: https://docs.opencv.org/2.4.13/doc/user_guide/ug_traincascade.html

48. Ojala T. A Comparative Study of Texture Measures with Classification Based on Feature Distributions / T. Ojala, M. Pietikainen, D. Harwood // Pattern Recognition. – 1996. – Vol. 29. – P. 51-59.

49. Zhang C. Ensemble Machine Learning. Methods and Applications [Electronic resource]. – Access mode: http://pzs.dstu.dp.ua/DataMining/boosting/bibl/Ensemble%20Machine%20Learning.pdf

50. Fedorchenko I. Development of the modified methods to train a neural network to solve the task on recognition of road users / I. Fedorchenko, A. Oliinyk, A. Stepanenko, T. Zaiko, S. Shylo, A. Svyrydenko // EasternEuropean Journal of Enterprise Technologies. – 2019. – Vol. 2, Issue 9/98. – P. 46–55.

51. Dougherty G. Pattern recognition and classification / G. Dougherty. – New York: Springer, 2013. – 196 p.

РОЗДІЛ 2
МЕТОДИ ВИРІШЕННЯ ЗАДАЧІ ПОШУКУ ОПТИМАЛЬНОГО ШЛЯХУ ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСУ ПРИЙНЯТТЯ РІШЕНЬ

При розробці інтелектуальних систем прийняття рішень виникають потреби розв'язання задач дискретної оптимізації. Вирішення задач оптимізації, зокрема задачі комівояжера, має багато прикладів практичного використання: прокладання мереж, оптимальне розташування газового трубопроводу, дизайн антени системи подачі, конфігурація транзисторів на платах, або сортування об'єктів для отримання найбільш відповідної конфігурації [1].

У загальному випадку задача формулюється так: відвідати усі точки із заданого списку, при умові що цей шлях буде найменшим серед усіх можливих шляхів. Тобто існує проблема знаходження оптимального шляху, використовуючи як можна менше обчислювальних потужностей комп'ютеру.

Стандартним шляхом рішення задачі є метод брутфорсу, або грубої сили, коли ми розраховуємо кожен з можливих шляхів та обираємо найкращий. Для n точок існує (n-1)! можливих шляхів. Наприклад, для 10 точок кількість шляхів дорівнює 3628800, а для 20 величезне число 2432902008176640000 [1].

Існують методи, що дозволяють знайти близькі до оптимуму рішення, наприклад, алгоритм найближчого сусіда та метод рою часток. Ці методи дозволяють знайти "непогане" рішення задачі комівояжера з довільною швидкістю. У цьому розділі для розв'язання задачі пошуку оптимального шляху запропоновано використовувати модифікований метод генетичного пошуку. Генетичні методи отримані в процесі узагальнення та імітації в штучних системах таких властивостей живої природи, як природний відбір, пристосованість до змінюваних умов середовища, спадкоємність нащадками життєво важливих властивостей від батьків і т.п. [2].

2.1 Постановка проблеми

У наш час відомі різні методи практичного використання рішення задачі комівояжера. Один з них це використання аптечного роботу

для автоматичної подачі ліків потрібних клієнту. При використанні цих роботів у момент подачі ліків маніпулятором роботу виникає проблема вирішення оптимізаційної задачі комівояжера. Ця задача відома через разючий контраст між простотою її формулювання та важкості рішення, вона відноситься до розряду NP-складних задач та демонструє всі аспекти комбінаторної оптимізації, слугувала й продовжує слугувати як опорна позначка для нових алгоритмічних ідей як то імітований відпал, табу пошук, нейронні мережі та еволюційні методи [2].

Запропоновано такий загальний шлях вирішення задачі комівояжера за допомогою генетичних алгоритмів [2, 3]:

– надати правильну репрезентацію для кодування шляху комівояжера через пакунки ліків;

– вигадати прийнятні генетичні оператори та їх модифікації для збереження працездатності та структури алгоритму;

– запобігти передчасному сходженню, тобто неможливості генерації нащадків, які кращі за своїх батьків.

Існує багато засобів для автоматизації обслуговування клієнтів, наприклад, засоби фірми Consis (рис. 2.1).

Рисунок 2.1 – Аптечний робот Consis [4]

Ці аптечні роботи німецької фірми або як їх ще називають роботи-фармацевти ідеально підходять для аптек, які бажають роботизувати частину свого асортименту ліків у діапазоні 1000-1500 найменувань. Доступні у 4-х можливих варіантах, роботи Consis можуть вмістити у себе від 8000 до 25000 упаковок. Одна з його особливостей це висока швидкість видачі упаковок усього за 7 секунд, що досягається за рахунок продуманої конструкції робота й швидкісному маніпулятору, таке швидке виконання замовлень дозволяє вільно обслуговувати 4 робочих місця у часи пік. Також дуже важливою є здатність компактного зберігання 3500 упаковок на кожному $м^2$ – це досягається завдяки використанню схильних полиць при середньому розмірі упаковок 100х30х60 [4].

Ще один засіб вартий для згадування це роботизований склад італійської фірми Pharmathek (рис. 2.2) – дозволяє роботизувати максимально можливий асортимент з різною швидкістю обігу. Доступний у 2-х стандартних розмірах, й опціонально – за індивідуально спроєктованими параметрами він може вмістити від 10 до 50 тисяч упаковок. Такий апарат забезпечує гнучкість й швидку адаптація до змін, оскільки у середині апарату зберігається різний аптечний асортимент з різною кількістю упаковок кожної одиниці. Аптечний робот самостійно обирає необхідне місце для зберігання кожної упаковки. Компактний маніпулятор робоскладу забезпечує щільне зберігання упаковок різної форми до 3500 упаковок на кожному $м^2$. При зміні асортименту, робот автоматично знайде кращі місця для зберігання упаковок з різним періодом обігу. Крім того забезпечується автоматичне завантаження упаковок – кожен пристрій має вбудовану функцію напівавтоматичного завантаження упаковок. Також його особливістю є ефективне використання площі та інтегрована інтелектуальна система робоскладів відслідковує внутрішні запаси, терміни придатності препаратів та по запиту може провести інвентаризацію у режимі реального часу [5].

Важливим елементом функціонування засобу є алгоритм подачі потрібного клієнту товару, тобто вирішення оптимізаційної задачі комівояжера, яка у даному випадку полягає у тому щоб дістатись потрібного товару певним шляхом, при умові що цей шлях буде найменшим серед усіх можливих шляхів. Тобто існує проблема знаходження оптимального шляху, використовуючи як можна менше обчислювальних потужностей.

Рисунок 2.2 – Роботизований склад Pharmatek [5]

Механізм переміщення (маніпулятор) включає каретку з закріпленим робочим органом і привід. Маніпулятор забезпечує переміщення товарів від місця прийому до місця зберігання, а потім до місця видачі. Плоска вертикальна робоча зона, утворена вертикальною площиною етажерки для зберігання товарів, визначає використання декартової системи координат переміщення робота вздовж площині етажерки (2 ступеня рухливості) і зміщення каретки з робочим органом в горизонтальній площині всередину етажерки (третя ступінь рухливості). Така схема використовується в плотерах планшетного типу або ріжучих верстатах з ЧПУ (лазерних, фрезерних і т.д.). Якщо етажерки розташовані з двох сторін від механізму переміщення, то каретка повинна розгортатися на 180 градусів (четверта ступінь рухливості). Для прискорення роботи можуть використовуватися 2 механізми переміщення або більше. Деякі виробники аптечних роботів використовують маніпулятор у кутовій системі координат з 6 ступенями рухливості, недоліком якого є обмежена робоча зона, доступна маніпулятору, розташована навколо нього. Надмірність ступеня рухливості (6 замість трьох або чотирьох) невигідна у фінансовому плані [6].

В якості робочого органу робота зазвичай застосовується встановлене на каретці захоплення.

Щоб отримати високу точність позиціонування каретки механізму переміщення зазвичай використовують електричний привід з кроковими двигунами для кожного ступеня рухливості, а також сенсорну систему, що дозволяє системі управління розраховувати і компенсувати помилки переміщення. Кроковий двигун повертається на кут відповідно до кількості поданих на нього імпульсів, що надходять від електронних блоків (драйверів, контролерів), що входять в систему управління. Зусилля крокового двигуна, необхідне для переміщення товару і деталей механізму переміщення, залежить від його потужності, а також амплітуди і тривалості (точніше прогальності) поданих на нього імпульсів. Завдяки програмі системи управління, що змінює параметри імпульсів крокових двигунів, досягається плавний розгін, швидке переміщення каретки і плавне її зупинення.

Досліджені технології використовують неефективний механізм маніпулятора для пошуку потрібних ліків. Метою роботи є розробка методів та програмних засобів пошуку оптимального шляху при виборі набору ліків, розташованих у апараті за вимогами користувача.

Серед вже існуючих публікацій за обраною тематикою можна виділити такі статті:

– Jean-Yves Potvin у роботі “Genetic algorithms for the traveling salesman problem” (1996) проводить аналіз методів вирішення оптимізаційної задачі комівояжера за допомогою евристичних алгоритмів та дає детальний опис способу використання генетичних алгоритмів для вирішення комбінаторних задач, у тому числі спосіб вирішення проблеми перестановки. У висновку статті зазначається що генетичні алгоритми здатні конкурувати з найкращими евристичними алгоритмами для задач середнього розміру(кількасот точок). Однак вони потребують багато часу та не можуть бути успішно примінятися до проблем розмірністю приблизно мільйон точок. З іншої сторони вони надають середу для паралельної реалізації, тому що вони працюють на популяції рішень [7];

– Woan Shin Tan, Siang Li Chua, Keng Who Yong, Tuck Seng Wu у роботі “Impact of Pharmacy Automation on Patient Waiting Time: An Application of Computer Simulation” (2009) намагаються показати можливості використання комп’ютерної симуляції для розрахунку впливу автоматизованої аптечної системи розподілення на час

очікування покупців та взагалі потенціал цих систем як рутинного засобу у аптечному управлінні. У висновку статті зазначається що використання автоматизованої системи розподілення призводить до збільшення швидкості обслуговування та дозволяє зменшити кількість робітників на ~28% [8–33].

З метою усунення виявлених недоліків відомих методів розв'язання задачі комівояжера та їх практичного застосування в даній роботі розроблено метод розв'язання задачі комівояжера за допомогою еволюційного пошуку.

2.2 Дослідження методів еволюційного пошуку для розв'язання задачі комівояжера

Задача комівояжера являє собою класичну оптимізаційну проблему що важко вирішується традиційними техніками. Її ціль – знайти найкоротший шлях для комівояжера який повинен навістити N точок. Цей тип задачі існує у багатьох формах, з кількома інженерними застосуваннями що включають оптимальне розташування газового трубопроводу, дизайн антени системи подачі, конфігурація транзисторів на платах, або сортування об'єктів для отримання найбільш відповідної конфігурації. Ейлер сформулював задачу комівояжера у 1759 році, і вона була формально названа й представлена корпорацією "Rand" у 1948 році [1-5].

Функція вартості для найпростішої форми задачі – це відстань яку проходить комівояжер у заданому порядку (x_n, y_n), $n = 1, \ldots, N$ за формулою відстані між двома точками, яка розраховується за допомогою теореми Піфагора:

$$\cos t = \sum_{n=0}^{N} \sqrt{(x_n - x_{n+1})^2 + (y_n - y_{n+1})^2}, \qquad (2.1)$$

де (x_n, x_{n+1}), (y_n, y_{n+1}) – координати n-го пакунку ліків.

Для рішення окресленої вище задачі у цій роботі буде використовуватися еволюційний пошук. Щоб провести аналогію між термінологією ЕА (еволюційних алгоритмів) та задачі комівояжера, треба дати визначення головним термінам ЕА:

– ген – одиниця спадковості, саме вони змінюються у процесі схрещування та мутації. Може приймати бінарні та числові значення – це визначає тип ЕА. При бінарному кодуванні ген може займати змінну кількість бітів;

– хромосома – послідовність генів;

– локус – позиція гена у хромосомі;

– алель – значення гена;

– популяція – сукупність хромосом.

Коли ми вирішуємо задачу комівояжера ми вважаємо:

– ген – індекс ліків у шляху;

– хромосома – шлях, що є послідовністю точок подачі ліків.

Дуже доречним є питання про різницю роботи з бінарними або числовими еволюційними методами, а саме який з методів дає кращий результат. Практика показує що більш результативними є числові методи. В них не потребується перетворення значень змінної до бінарних чисел та потреба стежити за числом бітів потрібних для відображення змінної. Числові генетичні алгоритми також є більш сумісними з іншими алгоритмами оптимізації, що робить їх здатними до комбінування або гібридизації [1,2].

Після великої кількості порівнянь бінарних та числових ГА, Міхалевіц (1992) сказав, “Проведені експерименти показують що числова репрезентація швидша, більш послідовна від запуску до запуску та надає більшу точність (особливо з великою множиною де бінарне кодування потребує набагато довшої репрезентації)”. Винахідники ГА у Європі у багатьох випадках використовували числові змінні у алгоритмі [1].

Кожен з методів повинен був вирішити задачу мінімізації двомірної функції:

$$f(x_1, x_2, x_3, x_4) = 100 \times (x_2 - x_1^2) + (1 - x_1)^2 + 90 \times (x_4 - x_3^2) + 10{,}1 \times (x_2 - 1)^2 + (x_4 - 1)^2, \quad (2.2)$$

Початкові дані кожного з обмежених методів:

– розмір популяції – 50,

– величина зупинки – 0,1, при $-20 \le x_1, x_2, x_3, x_4 \le 20$.

У бінарному методі використовувався пропорційний метод відбору, одноточкове схрещування та випадкова мутація. У числовому була використана мутація Гауса. У процесі виконання 100 експериментів були отримані результуючі дані. На рис. 2.3, 2.4 зображено результуючі дані для бінарного алгоритму.

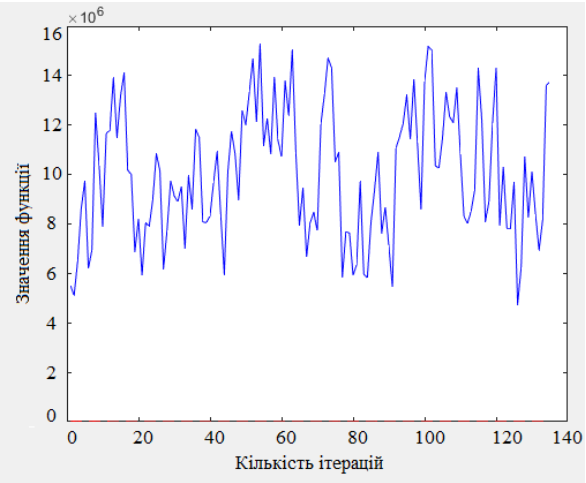


Рисунок 2.3 – Графік значень функції бінарного методу

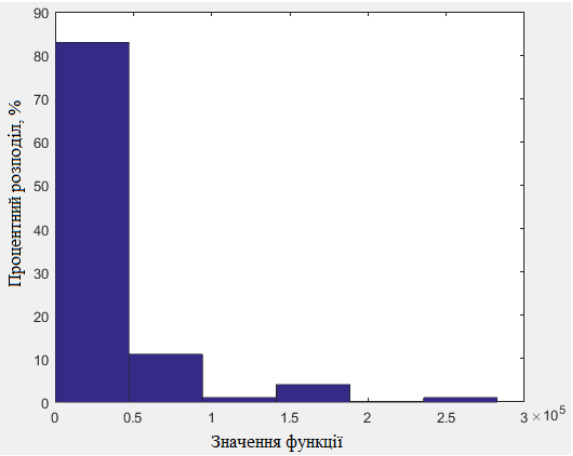


Рисунок 2.4 – Гістограма розподілення значень бінарного методу

На рис. 2.5, 2.6 зображено результуючі дані для числового алгоритму.

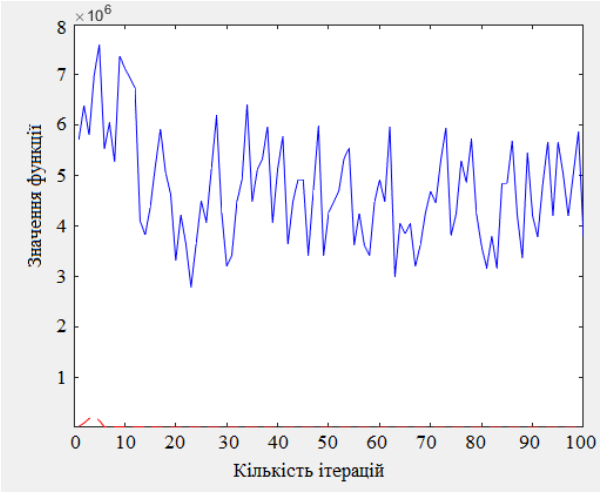


Рисунок 2.5 – Графік значень функції числового методу

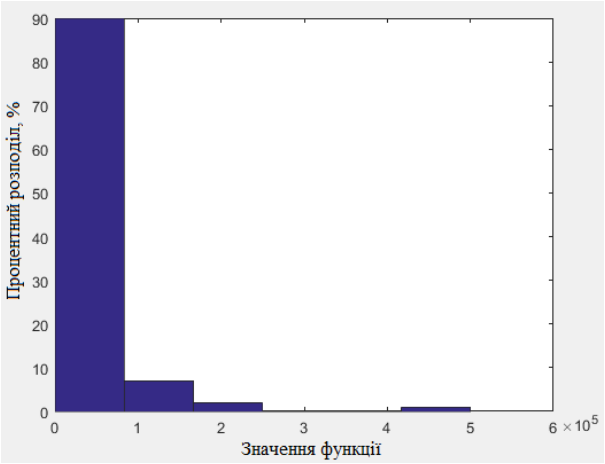


Рисунок 2.6 – Гістограма розподілення значень числового методу

2.3 Модифікації генетичних методів пошуку оптимального шляху

2.3.1 Модифікації простого генетичного алгоритму

Для даної роботи було зроблено наступні модифікації - 3 способи ініціалізації початкової популяції простого ГА. На відміну від існуючих методів, модифікована версія ГА дозволяє обирати метод ініціалізації початкової популяції при вирішенні задачі комівояжера, що, у свою чергу, дозволяє на етапі ініціалізації генерувати більш пристосовані хромосоми (хромосоми з кращими значеннями функції пристосованості) та таким чином покращити результати роботи алгоритму.

Робота алгоритму зупиняється у випадку досягнення максимального числа ітерацій або у випадку виродження популяції може використовуватися відсоток поліпшення значення кращої хромосоми. Після чого обчислюється коефіцієнт поліпшення за формулою:

$$\rho = \frac{f_{best\ t} - f_{best\ t\text{-}1}}{f_{best\ t\text{-}1}}, \tag{2.3}$$

де t – кількість ітерацій алгоритму; $f_{best\,t}$ – найкраще значення цільової функції поточної ітерації; $f_{best\,t\text{-}1}$ – найкраще значення цільової функції попередньої ітерації [5–12].

2.3.2 Змішаний метод

Змішаний метод (Random) [12–14] – метод ініціалізації початкової популяції ГА при використанні якого під час генерації початкової популяції кожен індивід буде складатися з різних послідовностей генів – номерів відвідуваних пакунків ліків розташованих у випадковому порядку. Він складається з чотирьох кроків.

1. Знайти максимальний індекс пакунку ліків (MAX_IDX).

2. Змінити розмір вектору пакунків ліків на знайдений вище індекс.
3. За допомогою функції std::iota() ініціалізувати вектор індексів пакунків ліків – результатом буде вектор типу – 1, 2, 3 … MAX_IDX.
4. За допомогою функції std::shuffle() перемішати зміст вектору індексів пакунків ліків – результатом буде вектор типу – 4, 15, 5 … .

2.3.3 Змішаний оптимальний метод

Змішаний оптимальний (Random optimal) [15–18] – метод ініціалізації початкової популяції ГА при використанні якого під час генерації початкової популяції кожен індивід буде складатися з однакових послідовностей генів, кожна з яких е оптимальним шляхом серед вказаної користувачем кількості згенерованих змішаних шляхів. Він складається з п’яти кроків:

1. Знайти максимальний індекс пакунку ліків (MAX_IDX).
2. Змінити розмір вектору пакунків ліків на знайдений вище індекс.
3. За допомогою функції std::iota() ініціалізувати вектор індексів пакунків ліків – результатом буде вектор типу – 1, 2, 3, … MAX_IDX.
4. За допомогою функції std::shuffle() перемішати зміст вектору індексів пакунків ліків – результатом буде вектор типу – 4, 15, 5, … .
5. Обрати найоптимальніший шлях серед згенерованих та ініціалізувати їм усю популяцію.

2.3.4 Розумний метод

Розумний метод (Smart method) [19] – метод міграції індивідів ОМ (острівна модель) при використанні якого з кожного острова обираються найкращі та найгірші індивіди. Серед тих і інших обчислюється середнє значення фітнес функції. За отриманими результатами обрані індивіди з найкращими середніми значеннями фітнес функції мігрують до островів з найгіршими значеннями. Обчислене середнє значення фітнес функції використовується для порівняння обраних груп індивідів і знаходиться за формулою:

$$A = \frac{\sum_{1}^{N} F_i}{N}, \tag{2.4}$$

де F_i – значення фітнес функції окремого індивіда; N – кількість індивідів у групі міграції (тобто розмір міграції) [20–24].

Метод складається з трьох кроків.

1. Зібрати з усіх островів групи найгірших та найкращих індивідів.
2. Шляхом порівняння величини знайденої за формулою 2.4 віднайти індекси островів з найгіршими групами міграції.

Замінити найгірші індивіди на островах знайдених на попередньому кроці найкращими групами міграції. При тому що група з найбільшим середнім значенням фітнес функції повинна замінити групу з найменшим.

2.3.5 Метод абсолютного оптимуму

Метод абсолютного оптимуму (Absolute optimum method) – метод міграції індивідів ОМ при використанні якого за середнім значенням фітнес функції обирається найкраща група індивідів, яка замінює собою найгірші індивіди на усіх островах моделі. Він складається з трьох кроків [25–29].

1. Зібрати з усіх островів групи найкращих елементів.
2. Шляхом сортування груп індивідів за величиною знайденою за формулою 2.4 знайти групу з максимальним значенням.
3. Замінити найгірші індивіди на усіх островах групою знайденою на попередньому кроці.

2.3.6 Випадковий метод вибору

Випадковий метод вибору (Random migration selection method) – метод вибору індивідів ОМ під час міграції, при використанні якого обирається підпопуляція серед найкращих індивідів (за допомогою відсотків, у інтервалі 5%-90%) для відправки на інший острів, а потім вже серед неї випадковим чином обирається потрібне для міграції

число індивідів. Кількість індивідів з яких обирається група міграції визначається за формулою:

$$N = \frac{P}{100} \cdot I, \tag{2.5}$$

де P – відсоток найкращих індивідів серед яких буде відбиратися група міграції; I – загальна кількість індивідів у популяції [30].

Метод складається з чотирьох кроків.

1. Відсортувати вектор індивідів за значенням фітнес функції.
2. Обчислити за формулою 2.5 кількість найкращих елементів серед яких буде обрано групу міграції, та додати перших N елементів до вектору.
3. За допомогою функції std::shuffle() перемішати зміст вектору індивідів отриманого на попередньому кроці.

Випадковим чином обрати групу міграції з отриманого на попередньому кроці вектору [31–35].

2.3.7 Модифікація методу активації

Для даної роботи було розроблено наступну модифікацію методу активізації – перехресний метод.

Перехресний метод (Cross method) [36–38] – полягає у тому, що під час початку виродження популяції робиться заміна індивідів шляхом розподілу популяції вертикально (розподіл на частини окремих індивідів) та розподіл усієї популяції горизонтально. Кількість елементів при вертикальному розбитті (популяції) знаходиться за формулою:

$$N_v = P \cdot \frac{V}{100}, \tag{2.6}$$

де P – кількість індивідів у популяції; V – відсоток індивідів для вертикального розбиття (популяції).

Кількість елементів при горизонтальному розбитті (індивідів) знаходиться за формулою:

$$N_h = I \cdot \frac{H}{100}, \tag{2.7}$$

де I – кількість генів у індивіді; H – відсоток генів для горизонтального розбиття (індивідів).

Метод складається з трьох кроків.

1. При досяганні критерію зупинки, за допомогою функції std::shuffle() перемішати зміст вектору індивідів.

2. За допомогою значень N_v та N_h робиться горизонтальне та вертикальне розбиття популяції. Потім до індивідів популяції і до генів індивідів які знаходилися лівіше від N_v та N_h відповідно застосовується функція std::shuffle().

3. Для кожного індивіду новоствореної популяції перераховується значення фітнес функції.

2.3.8 Метод оптимізації розташування ліків

Під час подачі ліків маніпулятором аптечного робота швидше за все будуть подаватися ліки які знаходяться ближче до отвору видачі ліків. Тому провівши статистичний аналіз споживчого кошика аптеки, ми можемо виділити категорії ліків які є найпопулярнішими серед клієнтів. Тож присвоївши кожній категорії рейтинг і розмістивши ліки у аптечному роботі в залежності від цього рейтингу ми можемо зменшити час подачі найпопулярніших товарів. Метод складається з 3 кроків:

1. Провести статистичний аналіз споживчого кошика аптеки за певний період.

2. Виділити категорії ліків і присвоїти кожній рейтинг в залежності від її популярності.

3. Розрахувати на основі даних з попереднього кроку позиції ліків у автоматі.

2.4 Програмне забезпечення вирішення задачі пошуку оптимального шляху

2.4.1 Розроблення інформаційних моделей програмного забезпечення пошуку оптимального шляху

ПЗ (програмне забезпечення) розроблене з використанням принципів ООП та складається з кількох взаємопов'язаних модулів – класів з їх складовими – методами й даними, тому можна зобразити структуру програми за допомогою UML-діаграми класів (рис. 2.7).

На діаграмі класів видно, що одним з головних класів є CGeneticAlgorithm. З ним, асоціативним зв'язком (композиція) пов'язаний клас CPopulation. Нащадком цього класу є клас CIsland, який використовує функціонал простого ГА для застосування його у острівній моделі CIslandModel.

Також на діаграмі позначений QML-клас Display – графічна форма візуалізації алгоритму, який пов'язаний з C++ класами графічного відображення процесу роботи простого ГА та острівної моделі – CStandardGAScene та CIslandModelScene відповідно.

Головні класи також наслідують класи бібліотеки Qt як то QObject, QRunnable, QQuickPaintedItem.

Для більш детального огляду функціональної складової системи доречно навести UML-діаграми послідовностей, кооперації, станів, діяльності. На рис. 2.8-2.9 зображені UML-діаграми послідовності. На діаграмі послідовностей (рис. 2.8) зображено головні об'єкти розробленої програми, їх лінії життя та існуючі між ними зв'язки. Ця діаграма дозволяє побачити у хронологічному порядку послідовність створення вказаних об'єктів та обміну повідомлень між ними. З рис. 2.9 видно, що першим створюється вікно параметрів – об'єкт класу Parameters, разом з яким створюються об'єкти класів CIslandModelScene, CIslandModel, CIsland, CPopulation. Об'єкт класу CIsland виконує основну роботу з вирішення задачі оптимізації (операції відбору, схрещування та мутації), об'єкт класу CIslandModel відповідає за виконання міграції та оновлення інтерфейсу користувача. По завершенні роботи надаються результати, об'єкту класу Result, який містить дані для побудови результуючих графіків та відображення важливої інформації (час роботи, кращі значення фітнес функції та відстані). На рис. 2.10 зображено UML-діаграму кооперації.

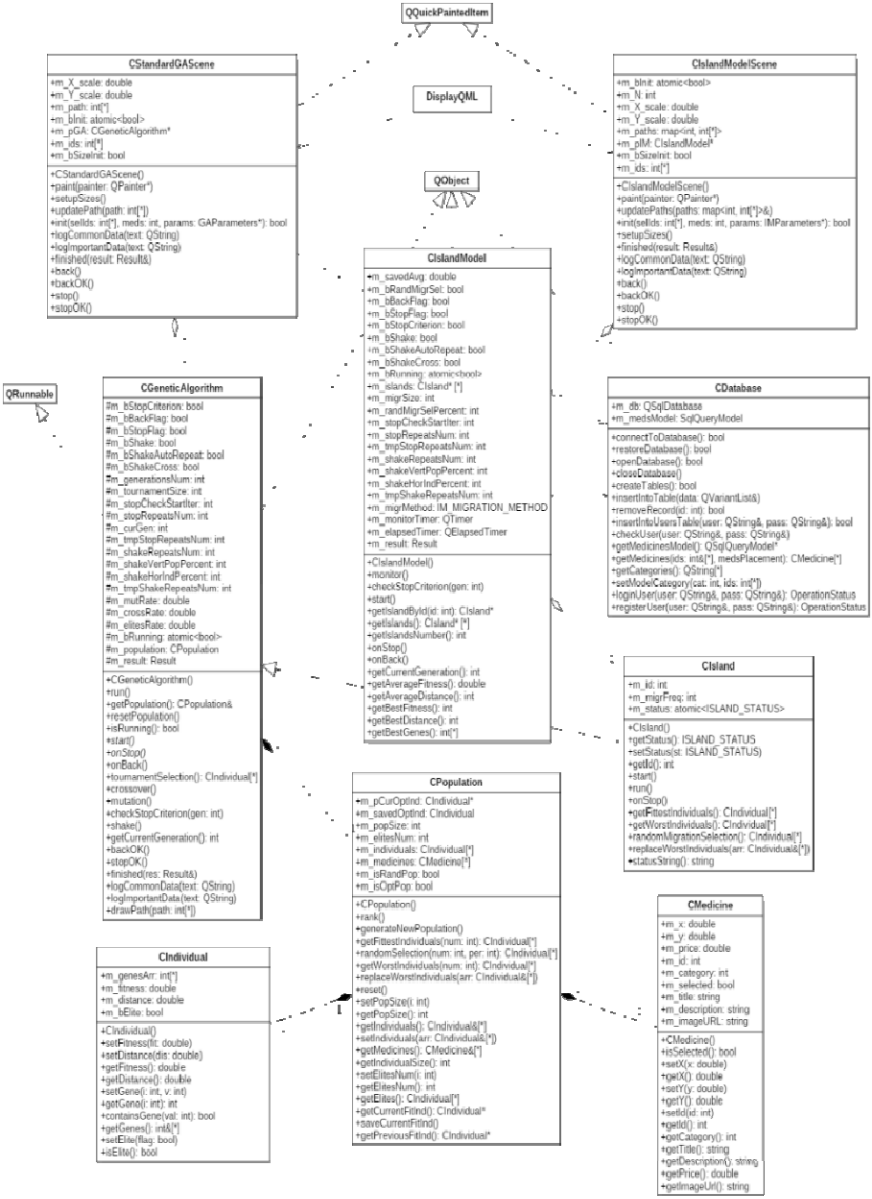


Рисунок 2.7 – UML-діаграма класів

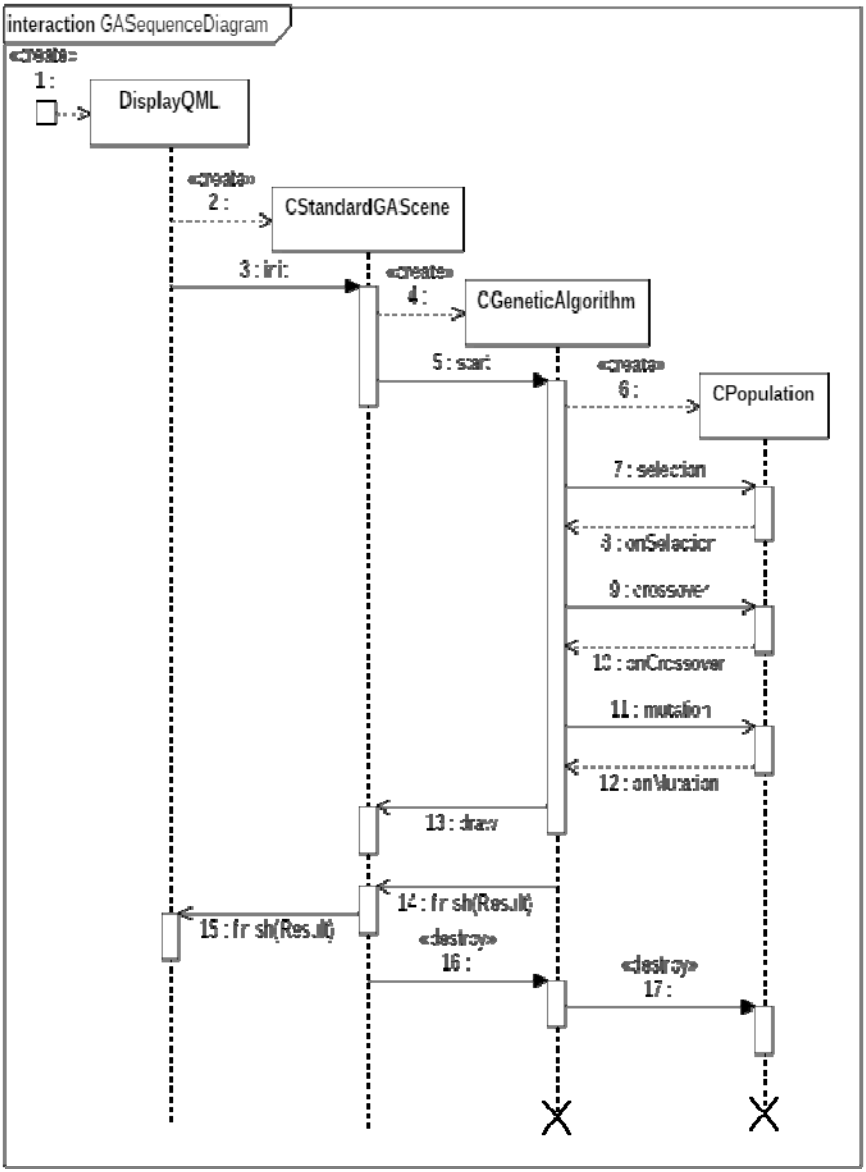


Рисунок 2.8 – UML-діаграма послідовності простого ГА

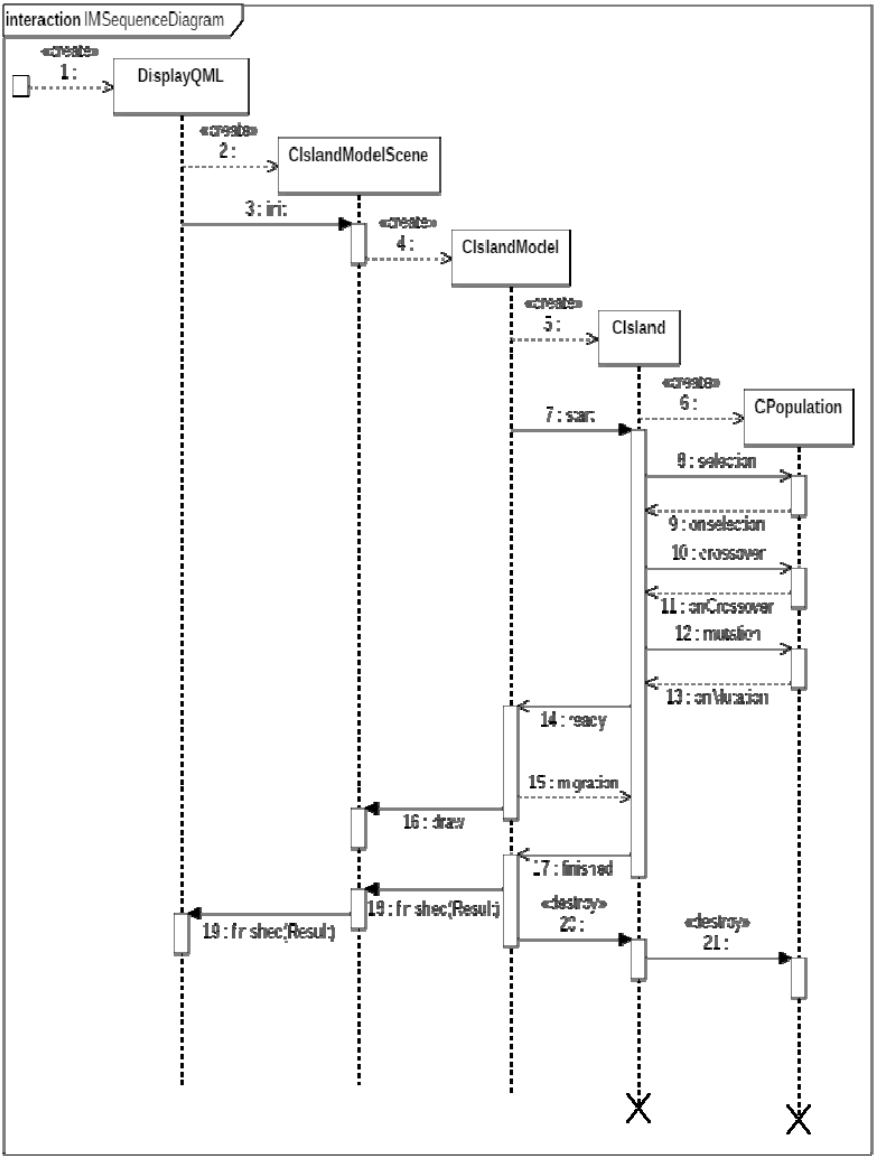


Рисунок 2.9 – UML-діаграма послідовності острівної моделі

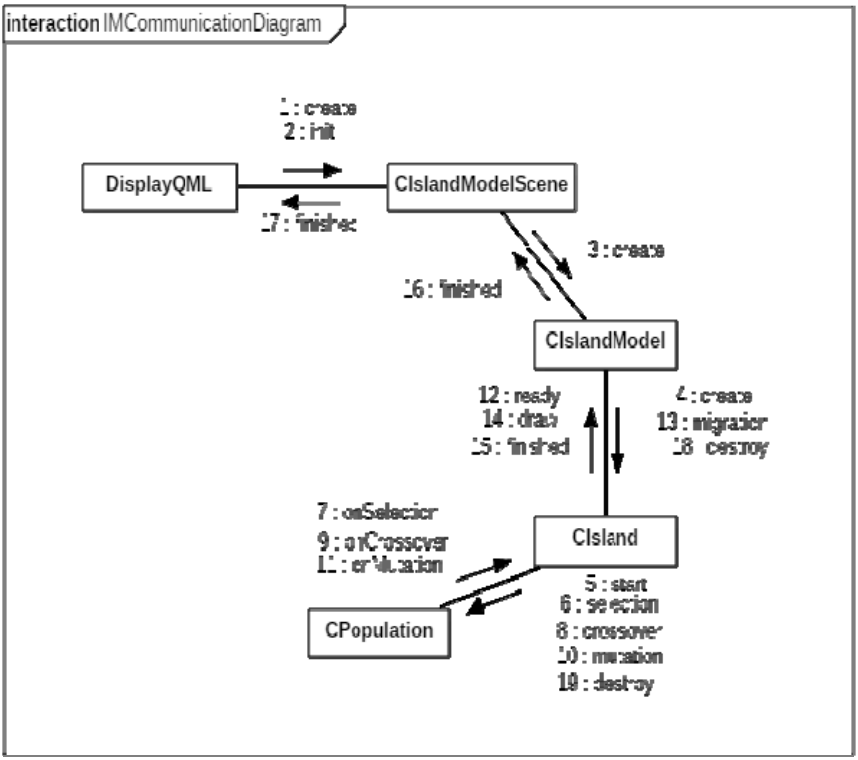


Рисунок 2.10 – UML-діаграма кооперації острівної моделі

На діаграмі кооперації (рис. 2.10) зображено основні об’єкти розробленої програми та функціональний зв’язок між ними. Вона надає можливість прослідкувати за порядком та напрямком повідомлень між об’єктами. Ця діаграма не включає у себе шкалу життя об’єкта, тому є компактнішою і оптимальна для того щоб окинути поглядом взаємодію зазначених об’єктів. Діаграма є миттєвим знімком процесу роботи острівної моделі – після виконання методу start(). На ній показано такі події як оновлення інтерфейсу користувача (draw), відображення результуючої інформації (finish(Result)), міграція при використанні острівної моделі (migrate), проведення відбору, схрещування, мутації (selection, crossover, mutation).На рис. 2.11, 2.12 зображено UML-діаграми станів.

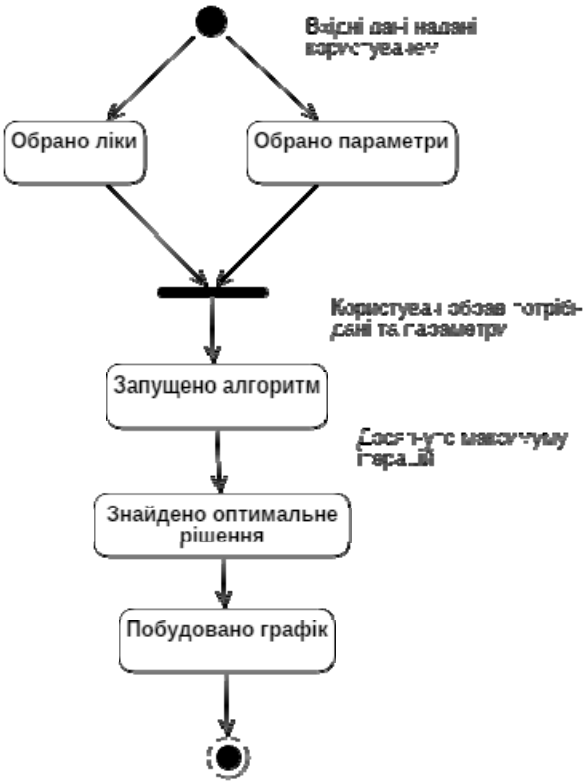


Рисунок 2.11 – UML-діаграма станів простого ГА

На рис. 2.13, 2.14 зображено UML-діаграми діяльності.

На діаграмі діяльності зображено послідовність етапів, які відображають процес роботи ГА, вона акцентує увагу на операціях та показує потік управління при розв'язанні задачі пошуку оптимального шляху при виборі ліків роботом зі складу. Операція можуть бути показані паралельно як то "Пошук оптимуму", "Оновлення інтерфейсу". Також на ній відображається вибір – можливість повернутись до початку, якщо результат не влаштовую. Ітеративність процесу наявна на рис. 2.14 і відбувається після проведення міграції переходячи на попередній етап "Пошук оптимуму".

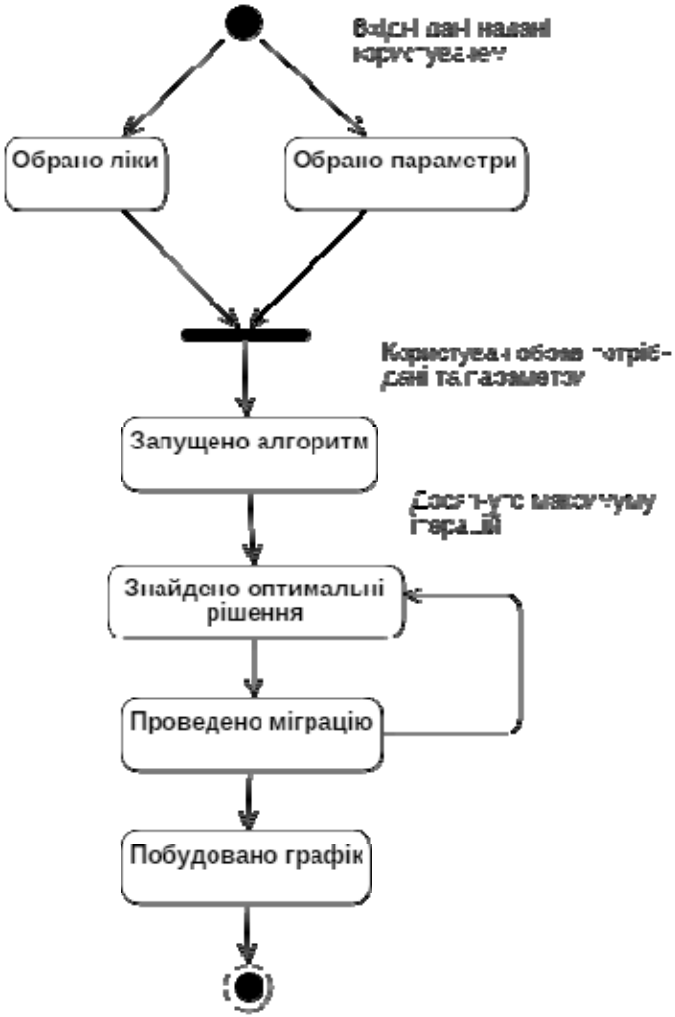


Рисунок 2.12 – UML-діаграма станів острівної моделі

На діаграмі компонентів відображено основні складові створеної системи: як зовнішні залежності системи (БД) так і модульні складові її структури: ГА, острівна модель, популяція.

На рис. 2.15 зображено UML-діаграму компонентів.

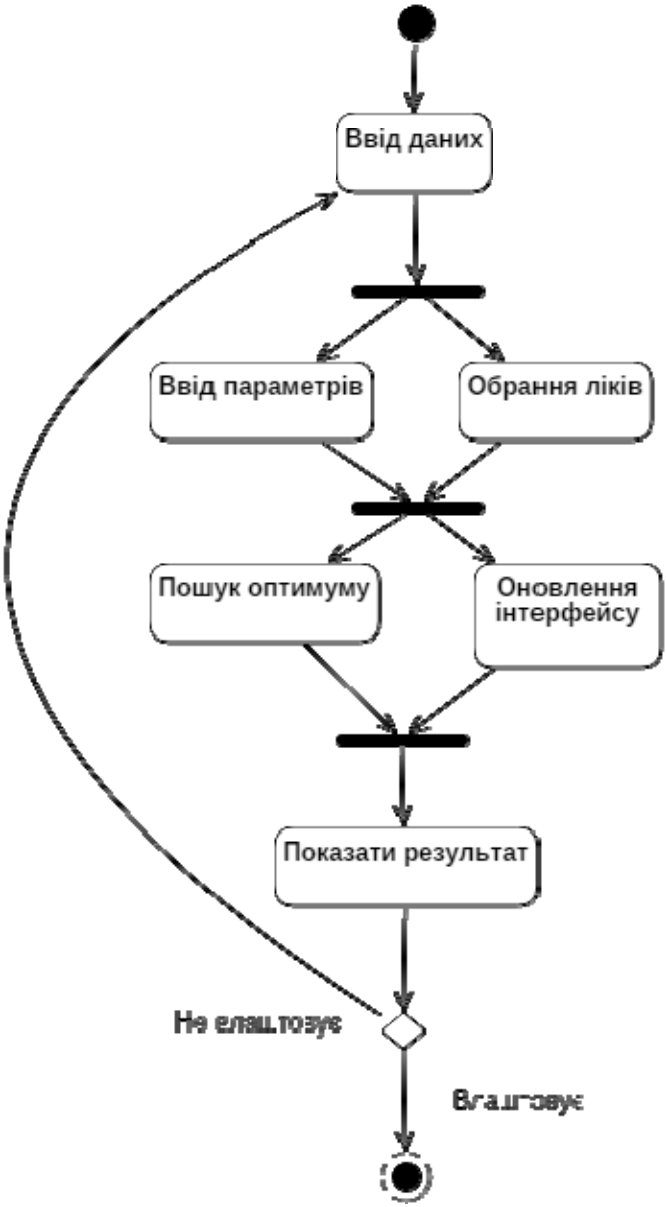


Рисунок 2.13 – UML-діаграма діяльності простого ГА

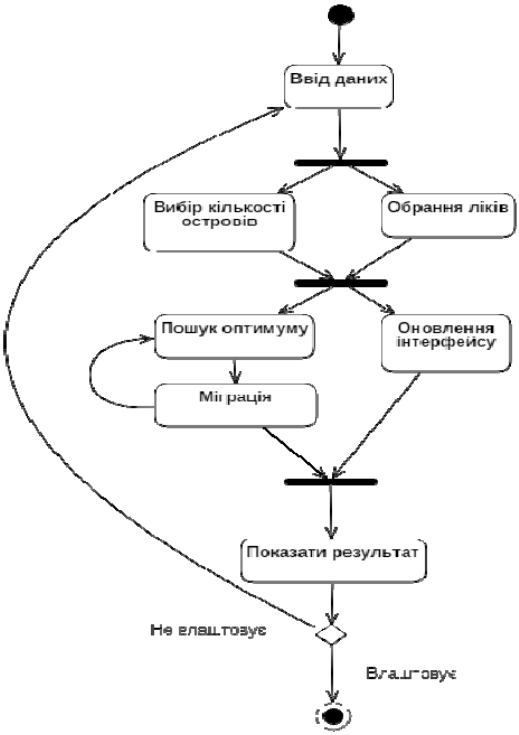


Рисунок 2.14 – UML-діаграма діяльності острівної моделі

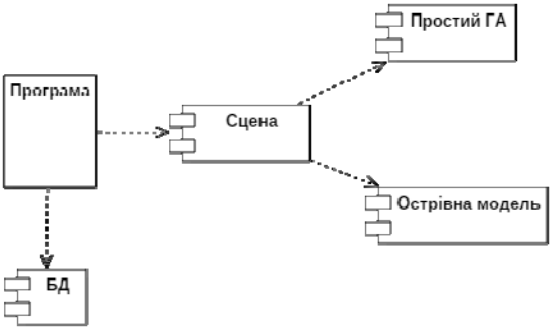


Рисунок 2.15 – UML-діаграма компонентів

2.4.2 Модулі розробленого програмного забезпечення

Клас CGeneticAlghoritm є реалізацією простого ГА. Він є нащадком класів QObject та QRunnable. Наслідування QObject є необхідним для роботи системи сигналів/слотів. Наслідування QRunnable – високорівневого API, надає можливість запускати алгоритм у окремому потоці, щоб не блокувати інтерфейс користувача під час роботи ГА. Він дозволяє отримати та скинути популяцію (методи getPopulation та resetPopulation), виконувати оцінку популяції, відбір, схрещування батьківських хромосом, мутацію нових хромосом (відповідно методи rank, tournamentSelection, crossover, mutation). У лістингу 2.1 наведено визначення класу CGeneticAlghoritm.

Клас CPopulation реалізує собою популяцію яка є важливим елементом таких модулів як класичний ГА та острівна модель. У лістингу 2.2 наведено визначення класу CPopulation.

Клас CIsland є нащадком класу простого ГА, тому виконує схожі з ним функції орієнтовані на роботу у острівній моделі, наприклад, підтримка механізму міграції. У лістингу 2.3 наведено визначення класу CIsland.

Клас CIslandModel містить у собі всі необхідні елементи для використання острівної моделі у системі: острова, кожен з яких виконує роботу простого ГА. Завдання цього класу керувати обміном індивідами між островами – міграцією та загальний контроль процесу роботи острівної моделі. У лістингу 2.4 наведено визначення класу CIslandModel.

Клас CStandardGAScene для візуалізації та відображення інформації під час роботи простого ГА. У лістингу 2.5 наведено визначення класу CStandardGAScene.

Клас CIslandModelScene для візуалізації та відображення інформації під час роботи острівної моделі. У лістингу 2.6 наведено визначення класу CIslandModelScene.

Клас CDatabase для організації доступу до БД. Використовується для отримання інформації щодо користувачів, ліків, категорій ліків. У лістингу 2.7 наведено визначення класу CDatabase.

```
class CGeneticAlgorithm: public QObject, public QRunnable
{
        Q_OBJECT // Макрос для роботи сигналів/слотів Qt
    public:
        CGeneticAlgorithm(std::vector<CMedicine>& medicines, const
GAParameters& params); // конструктор класу
        void run(); // перезаписаний метод базового класу
        const CPopulation& getPopulation() const;
        void resetPopulation();
        bool isRunning();
        virtual void start(); // віртуальний метод початку роботи
        virtual void onStop(); // віртуальний метод зупинки роботи
        void onBack();
        std::vector<CIndividual> tournamentSelection(); // відбір
        void crossover(); // метод схрещування
        void mutation(); // метод мутації
        void checkStopCriterion(std::size_t gen); // критерій зупинки
        void shake(); // метод активізації
        std::size_t getCurrentGeneration() const;
    signals: // сигнали класу
        void backOK();
        void stopOK();
        void finished(Result& result);
        void logCommonData(QString text);
        void logImportantData(QString text);
        void drawPath(std::vector<std::size_t> path);
    protected: // захищені члени класу (параметри ГА та популяція)
        bool m_bStopCriterion,
             m_bBackFlag{false}, m_bStopFlag{false}, m_bShake,
m_bShakeAutoRepeat, m_bShakeCross;
        std::size_t m_generationsNum, m_tournamentSize,
m_stopCheckStartIter,
            m_stopRepeatsNum, m_curGen{1}, m_tmpStopRepeatsNum{},
            m_shakeRepeatsNum, m_shakeVertPopPercent,
m_shakeHorIndPercent, m_tmpShakeRepeatsNum{};
        double m_mutRate, m_crossRate, m_elitesRate;
        std::atomic<bool> m_bRunning{false};
        CPopulation m_population;
        Result m_result;
};
```

Лістинг 2.1 - Визначення класу CGeneticAlghoritm

```
class CPopulation: public QObject // Нащадок класу QObject
{
        Q_OBJECT
    public:
        CPopulation(std::vector<CMedicine>&  medicines,  std::size_t
popSize, double eRate, bool randPop, bool optPop);
        void rank(); // оцінює індивідів популяції
        void generateNewPopulation(); // створює нову популяцію
        std::vector<CIndividual>    getFittestIndividuals(std::size_t
number); // отримує найкращих індивідів
        std::vector<CIndividual>   randomSelection(std::size_t   num,
std::size_t percent); // випадковим чином обирає індивідів
        std::vector<CIndividual>      getWorstIndividuals(std::size_t
number); // отримує найгірших індивідів
        void  replaceWorstIndividuals(const  std::vector<CIndividual>&
arr); // замінює найгірших індивідів
        void reset();
        void setPopSize(std::size_t i);
        std::size_t getPopSize() const;
        std::vector<CIndividual>& getIndividuals();
        void setIndividuals(std::vector<CIndividual>& arr);
        const std::vector<CMedicine>& getMedicines() const;
        std::size_t getIndividualSize(); // отримує розмір індивіда
        void setElitesNum(std::size_t i);
        std::size_t getElitesNum() const; // отримує кількість еліт
        std::vector<CIndividual> getElites();
        const CIndividual* getCurrentFitInd() const;
        void saveCurrentFitInd(); // зберігає найкращий індивід
        CIndividual* getPreviousFitInd();
    private:
        std::unique_ptr<CIndividual> m_pCurOptInd; // кращий індивід
        CIndividual m_savedOptInd;
        std::size_t m_popSize{}, m_elitesNum{}; // кількість еліт
        std::vector<CIndividual> m_individuals; // індивіди
        std::vector<CMedicine> m_medicines; // ліки
        bool m_isRandPop{false}, m_isOptPop{false};
};
```

Лістинг 2.2 – Визначення класу CPopulation

```
class CIsland: public CGeneticAlgorithm
{
       Q_OBJECT
    public:
     CIsland(std::size_t  id,  std::vector<CMedicine>&  medicines,  const
IMParameters& params);
     ISLAND_STATUS getStatus() const; // отримати статус острова
     void setStatus(ISLAND_STATUS status); // встановити статус
     std::size_t getId() const;
     void start(); // старт ГА острова
     void run();
     void onStop();
     std::vector<CIndividual> getFittestIndividuals(std::size_t num);
     std::vector<CIndividual> getWorstIndividuals(std::size_t num);
     std::vector<CIndividual>   randomMigrationSelection(std::size_t   num,
std::size_t percent);
     void replaceWorstIndividuals(const std::vector<CIndividual>& arr);
     std::string statusString(); // отримати статус-строку
    private:
     std::size_t m_id{}, m_migrFreq{};
     std::atomic<ISLAND_STATUS> m_status{ISLAND_STATUS::INITIAL};
};
```

Лістинг 2.3 – Визначення класу CIsland

```
class CIslandModel: public QObject
{
        Q_OBJECT
    public:
        CIslandModel(std::vector<int>&   selectedIds,   unsigned   int
medsPlacement, const IMParameters& params);
        void monitor(); // моніторинг островів
        void checkStopCriterion(std::size_t gen);
        void start();
        CIsland* getIslandById(std::size_t id); // отримати острів
        const   std::vector<std::unique_ptr<CIsland>>&   getIslands()
const;
        std::size_t getIslandsNumber() const;
        void onStop();
        void onBack();
        std::size_t getCurrentGeneration(); // номер ітерації
        double getAverageFitness(); // отримати середній фітнес
        double getAverageDistance(); // отримати середню відстань
        double getBestFitness(); // отримати найкращий фітнес
        double getBestDistance(); // отримати найкращу відстань
        const std::vector<std::size_t>& getBestGenes() const;
    signals: // сигнали класу
        void finished(Result& result); // сигнал по завершенні
        void logCommonData(QString text); // повідомлення №1
        void logImportantData(QString text); // повідомлення №2
        void backOK();
        void stopOK();
        void                          drawPaths(std::map<std::size_t,
std::vector<std::size_t>>& paths); // сигнал для малювання шляху
    private:
        double m_savedAvg{-1};
        bool m_bRandMigrSel, m_bBackFlag{false}, m_bStopFlag{false},
             m_bStopCriterion,      m_bShake,      m_bShakeAutoRepeat,
m_bShakeCross;
        std::atomic<bool> m_bRunning{false};
        std::vector<std::unique_ptr<CIsland>> m_islands;
        std::size_t       m_migrSize{},       m_randMigrSelPercent{},
m_stopCheckStartIter, m_stopRepeatsNum, m_tmpStopRepeatsNum{},
            m_shakeRepeatsNum,                  m_shakeVertPopPercent,
m_shakeHorIndPercent, m_tmpShakeRepeatsNum{};
        IMParameters::IM_MIGRATION_METHOD m_migrMethod{};
        QTimer m_monitorTimer;
        QElapsedTimer m_elapsedTimer;
        Result m_result;
};
```

Лістинг 2.4 – Визначення класу CIslandModel

```
class CStandardGAScene: public QQuickPaintedItem
{
        Q_OBJECT
    public:
        explicit CStandardGAScene(QQuickItem* parent = nullptr);
        ~CStandardGAScene();
        void paint(QPainter* painter); //  метод малювання
        void setupSizes(); // налаштування розмірів
        void updatePath(std::vector<std::size_t> path);
        Q_INVOKABLE bool init(std::vector<int> selectedIds, unsigned int
medsPlacement, GAParameters* paramsGA); // ініціалізація
    signals:
        void logCommonData(QString text);
        void logImportantData(QString text);
        void finished(Result& result);
        void back();
        void backOK();
        void stop();
        void stopOK();
    private:
        double m_Xmax, m_Xmin, m_X_scale{}, m_X_offset{},
               m_Ymax, m_Ymin, m_Y_scale{}, m_Y_offset{};
        std::vector<std::size_t> m_path;
        std::atomic<bool> m_bInit{false};
        std::unique_ptr<CGeneticAlgorithm> m_pGA;
        std::vector<int> m_ids;
        bool m_bSizeInit{false};
};
```

Лістинг 2.5 – Визначення класу CStandardGAScene

```
class CIslandModelScene: public QQuickPaintedItem
{
        Q_OBJECT
    public:
        explicit CIslandModelScene(QQuickItem* parent = nullptr);
        ~CIslandModelScene();
        void paint(QPainter* painter);
        void  updatePaths(std::map<std::size_t,  std::vector<std::size_t>>&
paths); // оновлення шляхів на сцені
        void setupSizes();
        Q_INVOKABLE bool init(std::vector<int> selectedIds, unsigned int
medsPlacement, IMParameters* paramsIM); // ініціалізація
    signals:
        void finished(Result& result);
        void logCommonData(QString text);
        void logImportantData(QString text);
        void back();
        void backOK(); // сигнал при натисканні кнопки “Назад”
        void stop();
        void stopOK();
    private:
        std::atomic<bool> m_bInit{false};
        std::size_t m_N{};
        double m_Xmax, m_Xmin, m_X_scale{}, m_X_offset{},
               m_Ymax, m_Ymin, m_Y_scale{}, m_Y_offset{};
        std::map<std::size_t, std::vector<std::size_t>> m_paths;
        std::unique_ptr<CIslandModel> m_pIM;
        bool m_bSizeInit{false};
        std::vector<int> m_ids;
};
```

Лістинг 2.6 – Визначення класу CIslandModelScene

```
class CDatabase: public QObject
{
        Q_OBJECT
    public:
        enum class OperationStatus // Статуси операцій з БД
        {
            FIELD_EMPTY,
            SUCCESS,
            INTERNAL_DB_ERROR,
            NOT_EXISTS,
            ALREADY_EXISTS
        };
        Q_ENUM(OperationStatus)
        bool connectToDatabase(); // підключення до БД
        bool restoreDatabase();
        bool openDatabase();
        void closeDatabase(); // закриття БД
        bool createTables();
        bool inserIntoTable(const QVariantList& data);
        bool removeRecord(const int id); // видалення запису з БД
        bool  insertIntoUsersTable(const  QString&  username,  const
QString& password);
        bool  checkUser(const  QString&  username,  const  QString&
password = "");
        QSqlQueryModel* getMedicinesModel();
        std::vector<CMedicine>          getMedicines(std::vector<int>&
selectedIds, MEDS_PLACEMENT medsPlacement);
        Q_INVOKABLE std::vector<QString> getCategories();
        Q_INVOKABLE void setModelCategory(unsigned int categoryNum,
std::vector<int> selectedIds);
        Q_INVOKABLE    OperationStatus    loginUser(const    QString&
username, const QString& password);
        Q_INVOKABLE   OperationStatus   registerUser(const   QString&
username, const QString& password);
    private:
        static  const  std::string  DatabaseName,  DatabaseHostname,
DatabaseDriver, UsersTable, UT_Username, UT_Password, MedicinesTable,
MT_Title,   MT_Description,   MT_Price,   MT_ImageURL,   MT_Category,
CategoriesTable, CT_Name, CT_Rating, CT_Description;
        QSqlDatabase m_db;
        SqlQueryModel m_medicinesModel;
};
```

Лістинг 2.7 – Визначення класу CDatabase

Клас CIndividual описує індивіда популяції. При рішенні задачі комівояжера індивід це один із можливих маршрутів. Методи setFitness(), setDistance() зберігають інформацію щодо значення функції вартості та відстані індивіда. У лістингу 2.8 наведено визначення класу CIndividual.

```
class CIndividual
{
    public:
        explicit CIndividual();
        explicit CIndividual(std::size_t size, bool random = false);
        CIndividual(const std::vector<std::size_t>& genes);
        bool operator<(const CIndividual& other);
        void setFitness(double fit); // встановити значення фітнесу
        void setDistance(double dis);
        double getFitness() const;
        double getDistance() const; // отримати значення відстані
        void setGene(std::size_t idx, std::size_t val);
        std::size_t getGene(std::size_t idx) const;
        bool containsGene(int val); // перевірити на наявність гену
        const std::vector<std::size_t>& getGenes() const;
        void setElite(bool flag);
        bool isElite() const; // чи є індивід елітою
    private:
        std::vector<std::size_t> m_genesArr; // масив генів
        double m_fitness{-1}, m_distance{-1};
        bool m_bElite{false};
};
```

Лістинг 2.8 – Визначення класу CIndividual

Клас CMedicine описує пакунок ліків. Інформація для ініціалізації об’єктів цього класу надходить з БД. Містить у собі координати пакунку ліків у двовимірному просторі, назву, опис, шлях до зображення, ціну, категорію. У лістингу 2.9 наведено визначення класу CMedicine.

```
class CMedicine{
    public:
        explicit CMedicine(std::size_t id, bool selected);
        CMedicine(std::size_t  id,  const  std::string&    title,  const
std::string&   description,  double  price,  const  std::string&  urlPath,
std::size_t category, bool selected);
        bool isSelected(); // чи обрані ліки
        void setX(double x);
        double getX() const; // отримати координату X
        void setY(double y);
        double getY() const;
        void setId(int id);
        std::size_t getId() const; // отримати ID ліків
        std::size_t getCategory() const;
        std::string getTitle() const;
        std::string getDescription() const;
        double getPrice() const; // отримати ціну ліків
        std::string getImageUrl() const;
    private:
        double m_x{}, m_y{}, m_price;
        std::size_t m_id, m_category;
        bool m_selected{false};
        std::string m_title, m_description, m_imageURL;
};
```

Лістинг 2.9 – Визначення класу CMedicine

2.4.3 Сутності бази даних програмної системи

У процесі розробки було створено загальний опис моделі системи, наведений у табл. 2.1. У табл. 2.2, 2.3, 2.4 наведено опис сутностей БД.

Таблиця 2.1 – Загальний опис моделі системи

Назва сутності	Назва (укр.)	Опис
Users	Користувачі	Містить дані зареєстрованих користувачів
Medicines	Ліки	Містить інформації щодо ліків
Categories	Категорії	Містить інформацію щодо категорій ліків

Таблиця 2.2 – Опис сутності “Користувачі” (“Users”)

Назва поля	Тип значення	Опис
ID	Integer	Ідентифікатор користувача
Username	Varchar(64)	Логін користувача
Password	Varchar(64)	Пароль користувача

2.4.4 Алгоритм функціонування програми

На рис. 2.16 зображено схему функціонування системи.

На початковому етапі користувач повинен обрати необхідні параметри та вхідні дані.

До параметрів відносяться – кількість ітерацій, розмір популяції, розмір турніру, коефіцієнт еліт, схрещування та мутації. Також можна вибрати один з доступних методів ініціалізації початкової популяції.

Таблиця 2.3 – Опис сутності “Ліки” (“Medicines”)

Назва поля	Тип значення	Опис
ID	Integer	Ідентифікатор ліків
Title	Varchar(64)	Назва ліків
Description	Varchar(128)	Опис ліків
Price	Real	Ціна ліків
ImageURL	Varchar(128)	Посилання на зображення
Category	Integer	Категорія ліків

Таблиця 2.4 – Опис сутності “Категорії” (“Categories”)

Назва поля	Тип значення	Опис
ID	Integer	Ідентифікатор категорії
Name	Varchar(64)	Назва категорії
Description	Varchar(128)	Опис категорії
Rating	Integer	Рейтинг категорії

Даними у цьому випадку є текстовий файл з координатами пакунків ліків для яких треба вирішити задачу комівояжера за допомогою ГА.

Після завантаження даних і запуску алгоритму, почнеться пошук оптимального рішення. Пошук буде виконуватися у новому потоці, тому користувач може змінювати параметри для підготовки до нового запуску ГА.

Після закінчення роботи ГА користувач може змінити параметри або дані та знову продовжити роботу.

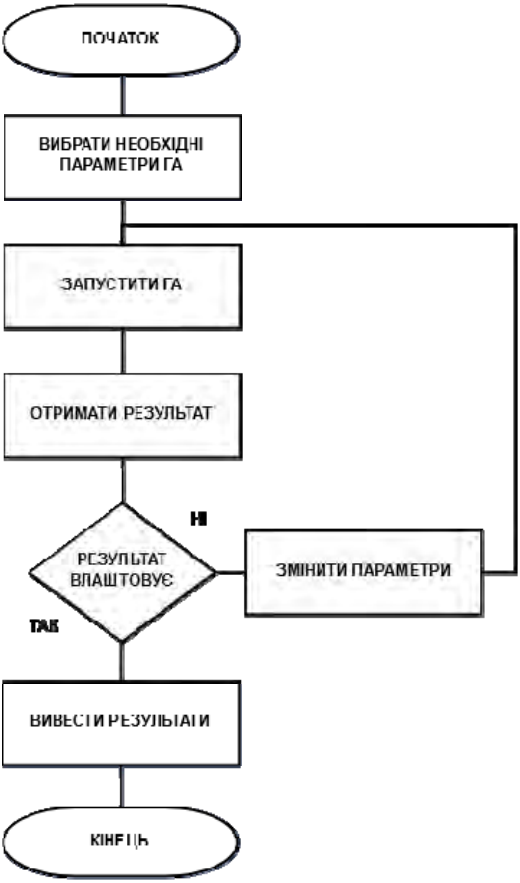


Рисунок 2.16 – Схема функціонування системи

2.4.5 Виконання програми

Екран входу (рис. 2.17) призначений для авторизації та реєстрації користувачів які збираються використовувати систему.

Для авторизації необхідно ввести логін і пароль та натиснути кнопку “Авторизація” (Login).

Для реєстрації треба ввести незареєстрований логін і пароль та натиснути кнопку “Реєстрація” (Register).

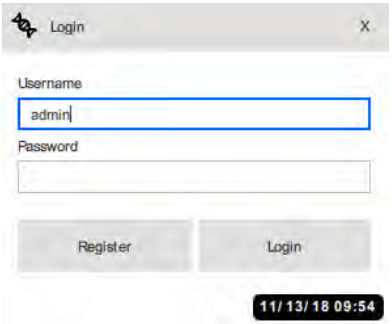


Рисунок 2.17 – Екран входу

Екран параметрів (рис. 2.18) призначений для обрання потрібних ліків та параметрів алгоритму. Щоб обрати ліки потрібно натиснути на вкладку потрібної категорії і обрати потрібні ліки.

Рисунок 2.18 – Екран параметрів

Екран роботи алгоритму призначений для візуалізації роботи алгоритму, для інформування оператора щодо прогресу роботи та надання можливості оператору впливати на роботу алгоритму. На рис. 2.19–рис. 2.21 зображено екрани роботи алгоритму.

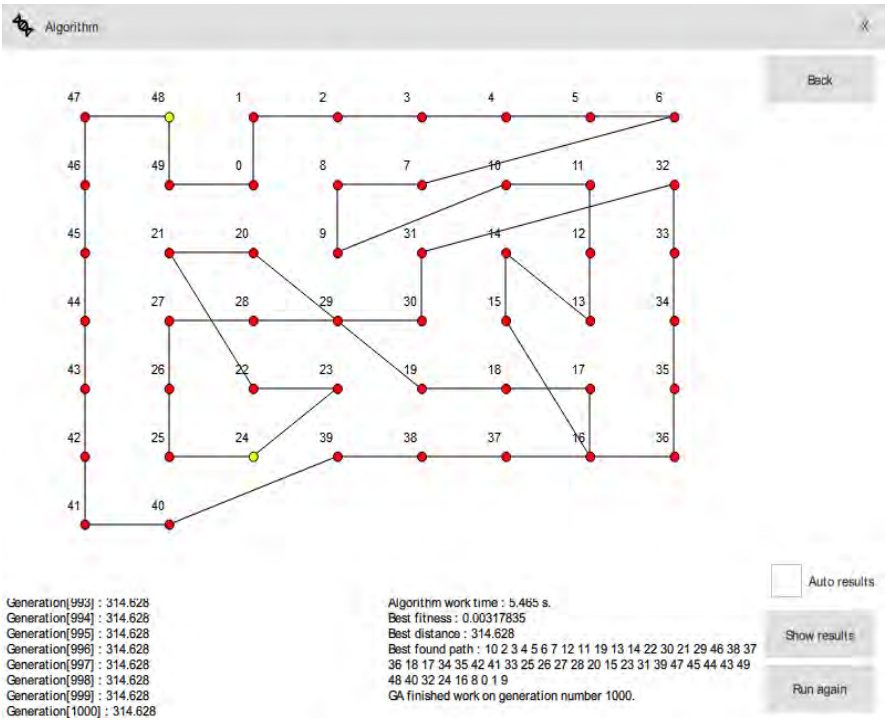


Рисунок 2.19 – Екран роботи простого ГА

На рис. 2.19 зображено момент закінчення роботи простого ГА, зліва внизу видно консоль у якій виводиться інформація щодо поколінь ГА та значень їх фітнес функції. Справа внизу видно результуючу інформацію як то час роботи алгоритму, найкращу знайдену відстань, найкращий шлях.

На рис. 2.20 зображено роботу острівної моделі, а саме візуалізація пошуку оптимуму на кожному з островів. Зліва внизу відображається інформації щодо еволюційного пошуку на кожному з островів.

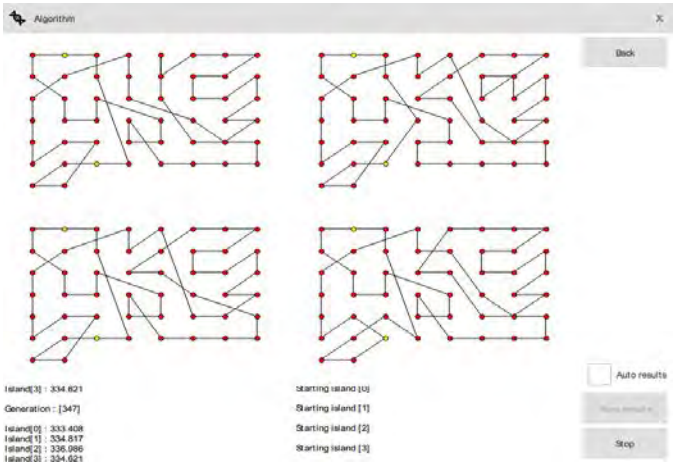


Рисунок 2.20 – Екран роботи острівної моделі

На рис. 2.21 зображено результати роботи алгоритму, які включають у себе графіки зміни фітнес функції та відстані. Також можна подивитися на обрані параметри алгоритму та результуючу інформацію.

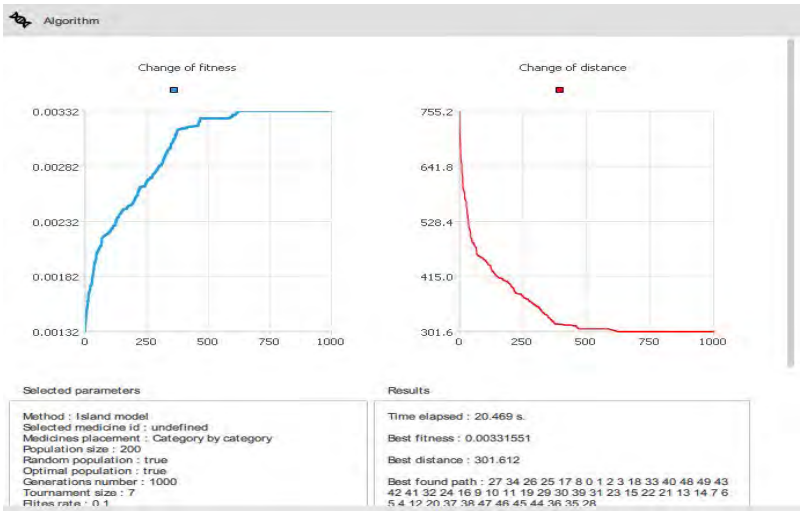


Рисунок 2.21 – Екран результатів острівної моделі

Повідомлення програми з’являються посеред графічного інтерфейсу користувача (підказки, зауваження) або вони надходять через консоль (інформація щодо роботи алгоритму, результати).

На рис. 2.22–рис. 2.24 зображено повідомлення оператору.

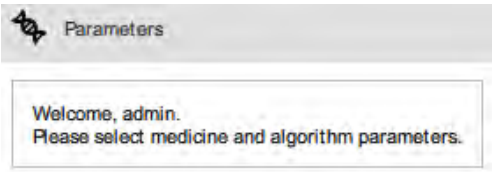


Рисунок 2.22 – Повідомлення оператору

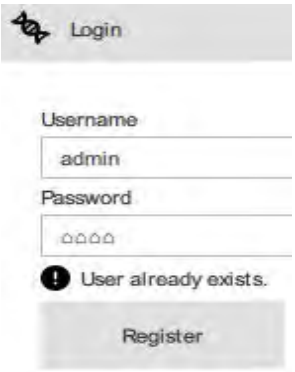


Рисунок 2.23 – Повідомлення при реєстрації вже існуючого користувача

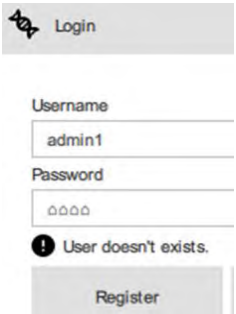


Рисунок 2.24 – Повідомлення при авторизації за незареєстрованими даними

2.5 Експерименти і результати

Були проведені два експерименти: перший описує ефективність методів ініціалізації початкової популяції генетичного алгоритму, другий ефективність методу активізації острівної моделі.

Перший експеримент був проведений з такими параметрами: кількість ітерацій (поколінь) – 1000, розмір популяції – 200, розмір турніру – 7, коефіцієнт еліти – 0,1, коефіцієнт схрещування – 0,9, коефіцієнт мутації – 0,1. Вхідні дані – 50 координат пакунків ліків. При запусках відрізнялись лише способи ініціалізації початкової популяції.

Другий експеримент використовував параметри з попереднього за винятком доданих параметрів острівної моделі: 4 острови. При запусках різницею був лише активований метод активізації.

З приведених нижче рисунків, які відображають результати першого експерименту, видно візуалізацію знайденого шляху (рис. 2.25), графіки залежності кількості ітерацій від функції вартості, дистанції (рис. 2.26). Видно, що при використанні стандартного методу ініціалізації початкової популяції, початкове значення функції дорівнює 0,0029, дистанції – 3,38. Час виконання експерименту – 5,406 секунд.

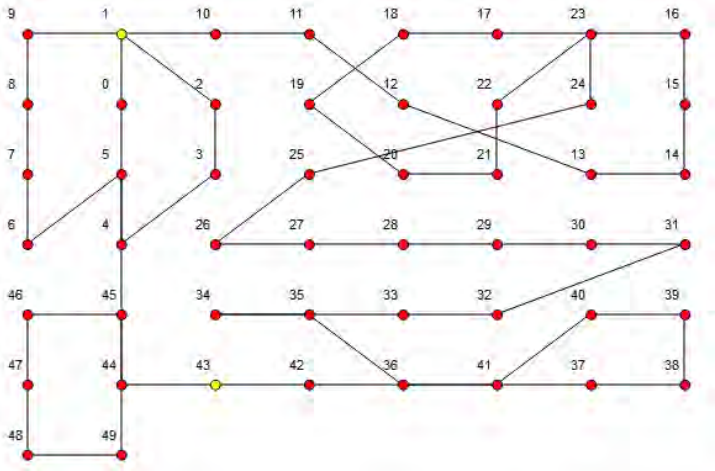


Рисунок 2.25 – Результуючий шлях 1-го експерименту

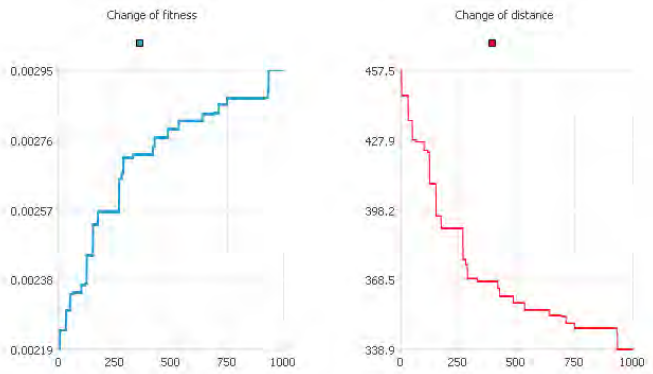


Рисунок 2.26 – Результуючий графік 1-го експерименту

Після проведення першого з експериментів доречно відмітити, що найкращий результат у стандартного методу ініціалізації початкової популяції. Це пояснюється тим, що у цій вирішуваній задачі використання початкової популяції виду 1, 2, 3, 4 і т. д. дозволяє мати менше початкове значення розрахованої функції пристосованості та відповідно дистанції.

На рис. 2.27 зображено результуючу інформацію 1-го експерименту.

Рисунок 2.27 – Результуюча інформація 1-го експерименту

Перший стовпчик табл. 2.5 дає нам можливість переконатися у доречності використання стандартного методу ініціалізації популяції. Усі 3 модифікації показали приблизно однакові показники часу, результати наведені у табл. 2.5.

Таблиця 2.5 – Результати першого експерименту

Назва методу	Початкове значення відстані, м	Результуюче значення відстані, м	Час виконання, с
Стандартний	4,57	3,38	5,406
Змішаний	3,784	3,341	5,530
Змішаний випадковий	3,55	3,349	5,544

На рис. 2.28, 2.29 зображено результати проведення другого експерименту.

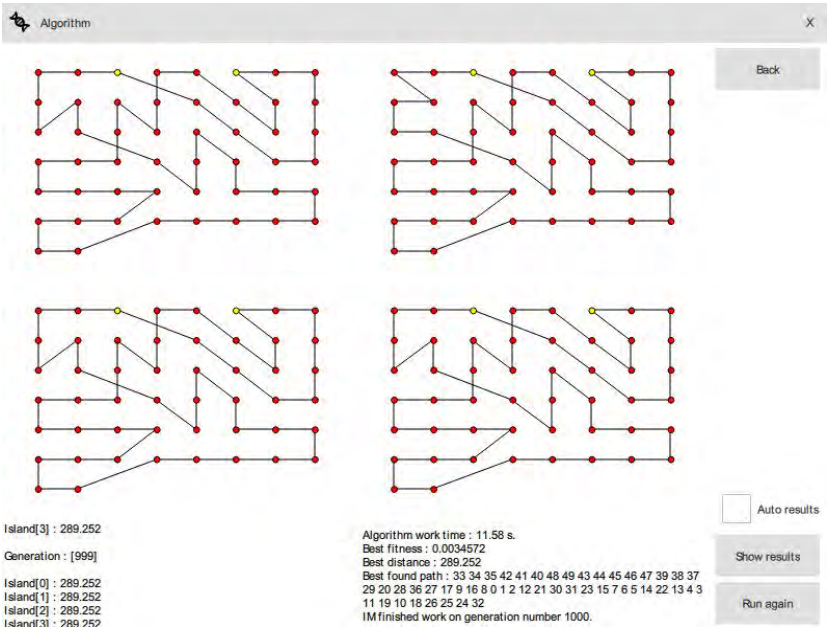


Рисунок 2.28 – Результат другого експерименту

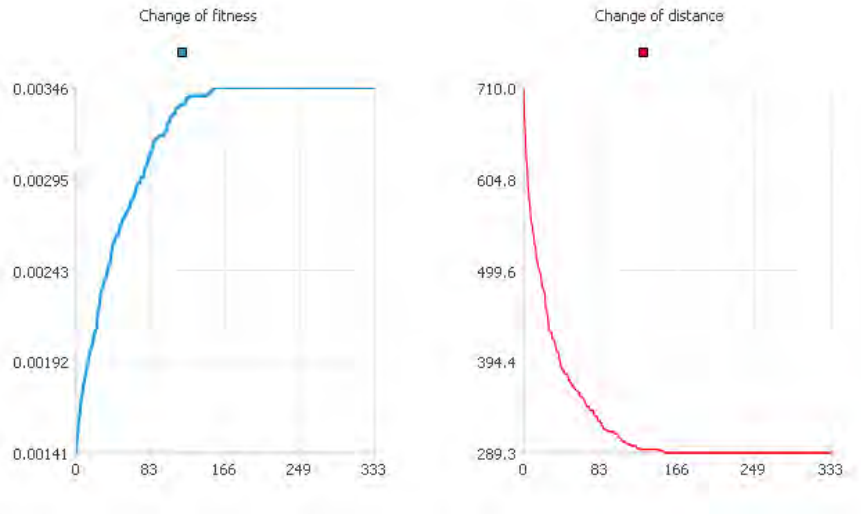


Рисунок 2.29 – Результуючий графік другого експерименту

На рис. 2.30 зображено результуючу інформацію другого експерименту.

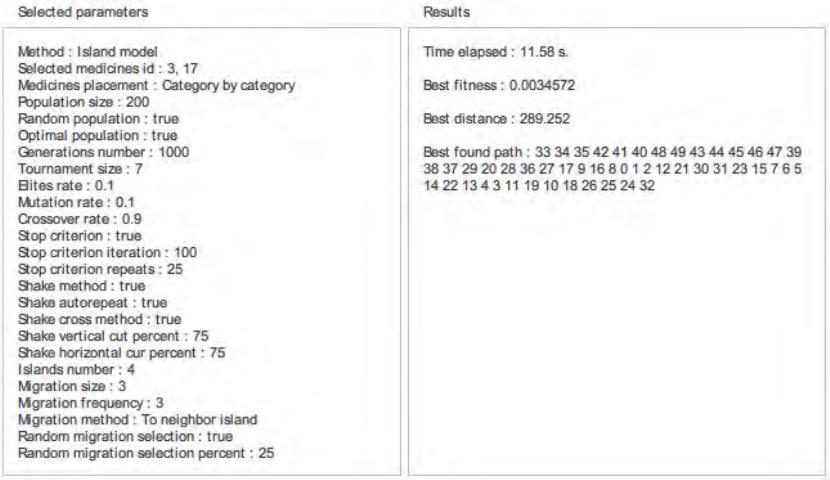


Рисунок 2.30 – Результуюча інформація другого експерименту

Після проведення другого з експериментів доречно відмітити, що кращий результат показав метод активізації острівної моделі. Це пояснюється тим, що він дозволив продовжити пошук мінімуму навіть після досягнення критерію зупинки, а саме це від нього й очікувалося. Другий та третій стовпчики табл. 2.6 дають нам можливість переконатися у доречності використання методу активізації острівної моделі: хоча обидва запуски показали приблизний час, метод активізації показав кращий результат. Результати експерименту наведені у табл. 2.6.

Таблиця 2.6 – Результати другого експерименту

Назва методу	Початкове значення відстані, м	Результуюче значення відстані, м	Час виконання, с
Стандартна ОМ	7,07	2,96	11,5
Метод активізації	7,1	2,89	11,58

2.6 Висновки за розділом 2

Задача комівояжера являє собою класичну оптимізаційну проблему що важко вирішується традиційними техніками. Її ціль – знайти найкоротший шлях що проходить через всі N точок. Цей тип задачі існує у багатьох формах, з кількома інженерними застосуваннями, тому розробка програмних засобів для її вирішення є дуже доречною. Один із варіантів практичного застосування це використання аптечного роботу для автоматичної подачі ліків потрібних клієнту. При використанні цих роботів у момент подачі ліків маніпулятором роботу виникає проблема вирішення оптимізаційної задачі комівояжера.

У даному розділі було використано еволюційний пошук як засіб оптимізації. Було проведено теоретичний аналіз предметної області ГА та вже існуючих методів ГА. Також було проведено порівняльний експеримент між існуючими методами для виявлення

найоптимальнішого набору генетичних операторів при вирішенні задачі оптимізації.

Програма була написана за допомогою високорівневої мови C++, а саме була використана бібліотека Qt. При написанні програми використовувалися принципи об'єктно-орієнтованого програмування, а саме використання класів, принципу інкапсуляції даних та наслідування. Для візуалізації структури та функціональний відносин модулів розробленого програмного забезпечення використовувалися UML-діаграми класів, послідовностей, діяльності, станів та кооперації.

Окремо було розглянуто алгоритм функціонування програми – дії які повинен виконати користувач для рішення задачі комівояжера.

Було запропоновано модифікацію еволюційного алгоритму – а саме 3 методи генерації початкової популяції:

– стандартний метод (Default) – при використанні цього методу при генерації початкової популяції кожен індивід буде складатися з однакової послідовності генів – індексів відвідуваних пакунків ліків розташованих у порядку зростання від першого до останнього;

– змішаний метод (Random) – при використанні цього методу при генерації початкової популяції кожен індивід буде складатися з різних послідовностей генів – індексів відвідуваних пакунків ліків розташованих у випадковому порядку;

– змішаний оптимальний (Random optimal) – при використанні цього методу при генерації початкової популяції кожен індивід буде складатися з однакових послідовностей генів, кожна з яких є оптимальним шляхом серед вказаної користувачем кількості згенерованих змішаних шляхів.

Для острівної моделі було реалізовано такі модифікації: 2 методи міграції (розумний та абсолютного оптимуму), випадковий метод вибору індивідів при міграції. Також було реалізовано метод активізації та його модифікацію – перехресний метод. Було додано метод оптимізації розташування ліків.

Наукова новизна роботи полягає у тому, що запропоновано модифікацію еволюційних методів, яка передбачає використання одного з розроблених методів ініціалізації початкової популяції.

Запропонована модифікація дозволяє суттєво підвищити швидкість за рахунок того що при використанні стандартного методу

ініціалізації початкової популяції, початкове значення розрахованої функції пристосованості та відповідно дистанції буде меншим ніж при використанні інших методів ініціалізації, що дозволить більш точніше визначити оптимум задачі. Також використання розподіленого алгоритму – острівної моделі дозволяє отримати приріст у швидкості та у якості результату. Метод активізації надає можливість продовжити еволюційний пошук навіть при досягненні критерію зупинки.

2.7 Література до розділу 2

1. Haupt, R. Practical genetic algorithms / R.L. Haupt, S. E. Haupt.— New Jersey: John Wiley & Sons, 2004. – 261 p.
2. Gen M. Genetic algorithms and engineering design / M. Gen, R. Cheng. – New Jersey: John Wiley & Sons, 1997. – 352 p.
3. Олійник, А. О. Еволюційні обчислення та програмування: навчальний посібник / А. О. Олійник, С. О. Субботін, О. О. Олійник. – Запоріжжя: ЗНТУ, 2010. – 324 с.
4. Willach Pharmacy Solutions – UK [Electronic resource]. Access mode: https://www.willach-pharmacy-solutions.com/EN/index.php.
5. Pharmathek – Robot e magazzini automatizatti [Electronic resource]. Acess mode: http://www.pharmathek.com.
6. Gan, Y. Human-like Manipulation Planning for Articulated Manipulator / Y. Gan, X. Dai // Journal of Bionic Engineering. – vol. 9, 2012. – no. 4. – pp. 434-445.
7. Potvin, J. Genetic algorithms for the traveling salesman problem / J. Potvin // Annals of Operations Research. – 1996. –Vol. 63, № 3. – P. 337-370.
8. Tan W. Impact of Pharmacy Automation on Patient Waiting Time: An Application of Computer Simulation / W. Tan, S. Chua, K. Yong, T. Wu // Ann Acad Med Singapore. – 2009. – Vol. 38. – P. 501-507.
9. Goldberg, D. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading / D. Goldberg. – MA: Addison-Wesley, 1989. – 432 p.
10. Holland, J. Adaptation in Natural and Artificial Systems / J. Holland. – Ann Arbor: University of Michigan Press, 1975. – 232 p.

11. Субботін, С. О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень: Навчальний посібник / С. О. Субботін. – Запоріжжя: ЗНТУ, 2008. – 341 с.

12. Burkhovetskiy V. Parallelizing an exact algorithm for the traveling salesman problem / V. Burkhovetskiy, B. Steinberg // Procedia Computer Science. – 2017. –Vol. 119. – P. 97-102. doi: 10.1016/j.procs.2017.11.165

13. Dopico J. Encyclopedia of artificial intelligence / J. Dopico, J. de la Calle, A. Sierra. – New York : Information Science Reference, 2009. – Vol. 1-3. – 1677 p.

14. Емельянов, В. В. Теория и практика эволюционного моделирования / В. В. Емельянов, В. В. Курейчик, В. М. Курейчик. – М. : Физматлит, 2003. – 432 с.

15. Курейчик, В. М. Генетические алгоритмы: монография / В. М. Курейчик. – Таганрог: ТРТУ, 1998. – 242 с.

16. Bäck, T.: I Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. Published in: Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence, 27-29 June 1994, pp. 57-62 (1994). doi: 10.1109/ICEC.1994.350042.

17. Ротштейн, А. П.Интеллектуальные технологии идентификации: нечеткие множества, генетические алгоритмы, нейронные сети / А. П. Ротштейн. – Винница: Універсум-Вінниця, 1999. – 320 с.

18. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л.Рутковский ; пер. с польск. И. Д. Рудинского. – М.: Горячая линия – Телеком, 2004. – 452 с.

19. Chambers L. The practical handbook of genetic algorithms / L. Chambers. – Florida : CRC Press, 2000. – Vol. I: Applications. – 520 p.

20. Chambers L. The practical handbook of genetic algorithms / L. Chambers. – Florida : CRC Press, 2000. – Vol. II: New frontiers. – 421 p.

21. Chambers L. The practical handbook of genetic algorithms. / L. Chambers. – Florida: CRC Press LLC, 2000. – Vol. III: Complex coding systems. – 659 p.

22. Syswerda G. Uniform crossover in genetic algorithms / G. Syswerda. –Los Altos, CA: Morgan Kaufmann, 1989. – 7 p.

23. Altenberg, L. The evolution of evolvability in genetic programming / K. L. Altenberg, E. Kinnear // Advances in Genetic Programming. MIT Press, 1994. – 27 p.

24. Davis L. Handbook of Genetic Algorithms / L. Davis. – Van Nostrand Reinhold, 1991. – 385 p.

25. Вороновский, Г.К. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности / Г. К. Вороновский, К. В. Махотило, Н. Петрашев, С. А. Сергеев. – Х.: Основа, 1997. – 111 с.

26. Grefenstette, J. Optimization of control parameters for genetic algorithms / J. Grefenstette. – San Francisco, CA: Morgan Kaufmann, 1986, – 112 p.

27. Foneseca, C. Genetic algorithms for multi-objective optimization: Formulation, discussion, and generalization / C. Foneseca. –San Francisco, CA: Morgan Kaufmann, 1993. – 7 p.

28. Gong Y. Distributed island-model genetic algorithms using heterogeneous parameter settings / Y. Gong, A. Fukunaga // 2011 IEEE Congress of Evolutionary Computation (CEC), New Orlean, 5-8 June 2011: proceedings. – Los Alamitos: IEEE, 2011. – P. 820.

29. Cantu-Paz E. Are multiple runs of genetic algorithms better than one? / E. Cantu-Paz // Genetic and Evolutionary Computation : 5th Annual Conference (GECCO 2003), Chicago, 12–16 of July 2003: proceedings. – P. 801-812. – (Lecture Notes in Computer Science, 2003. – Vol: 2723). DOI: 10.1007/3-540-45105-6_94.

30. Cantu-Paz E. Efficient and accurate parallel genetic algorithms / E. Cantu-Paz. – Massachusetts: Kluwer Academic Publishers, 2001. – 162 p.

31. Stroustrup, B.The C++ programming language [Electronic resource] / B., Stroustrup. – Acess mode: https://www.academia.edu/13295044/The_C_Programming_Language_4th_Edition.

32. Шлее, М. Qt 5.5 Профессиональное программирование на C++ / М. Шлее. – СПб.: “БХВ-Петербург”, 2015. — 896 с.

33. Kreibich, J. Using SQLite: Small. Fast. Reliable. Choose Any Three [Electronic resource]. - Acess mode: http://www.freeoa.net/attachments/2014/using.sqlite.oreilly.2010.pdf.

34. Buduma, N. Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms / N. Buduma, N. Locascio // O'Reilly Media. – 2017. – 298 с.

35. Kaspi M. Maximizing the profit per unit time for the travelling salesman problem / M. Kaspi, M. Zofi, R. Teller // Computers & Industrial Engineering. – 2019. –Vol. 135. – P. 702-710. doi: 10.1016/j.cie.2019.06.050

36. Cardenas-Montes, M. Creating hard-to-solve instances of travelling salesman problem / M. Cardenas-Montes // Applied Soft Computing. – 2018. – Vol. 71. – P. 268-276. doi: 10.1016/j.asoc.2018.07.010

37. Stanek, R. Geometric and LP-based heuristics for angular travelling salesman problems in the plane / R. Stanek, P. Greistorfer, K. Ladner, U. Pferschy // Computers & Operations Research. – 2019. –Vol. 108. – P. 97-111. DOI: 10.1016/j.cor.2019.01.016

38. Mestria M. New hybrid heuristic algorithm for the clustered traveling salesman problem / M. Mestria // Computers & Industrial Engineering. – 2019. –Vol. 116. – P. 1-12. doi: 10.1016/j.cie.2017.12.018

РОЗДІЛ 3
ОПТИМІЗАЦІЯ ТА ПРОГНОЗУВАННЯ ФІНАНСОВИХ ПОКАЗНИКІВ ПІДПРИЄМСТВ ТОРГІВЛІ

У сучасному конкурентному середовищі швидкість та правильність прийняття рішень є ключовим фактором успішності рітейлера, яким є і аптечна мережа. Головними індикаторами успішності роботи аптеки є її фінансові показники: прибуток та оборот.

Впливати на ці показники можна різними методами, але одним з найефективніших методів є оптимізація асортименту складу аптеки.

Питання оптимального асортименту є важливим як для аптеки, що працює тривалий час, так і для аптеки, що тільки буде відкриватись.

На вибір асортименту впливають такі фактори, як площа аптеки, собівартість медикаментів, оборотність асортиментних позицій, маркетингові фактори, фактори сезонності, екологічні фактори, фактори географічного положення аптеки відносно місцевої інфраструктури і т. ін.

Оптимізація асортименту призведе до більш ефективного використання площі аптек, зменшення незадоволеного попиту та, в кінцевому результаті, до зменшення роздрібної вартості ліків за рахунок зменшення видатків на зберігання та обслуговування неоптимальне завантажених площин аптеки.

В ході роботи було розроблено генетичний метод багатокритеріальної оптимізації з модифікацією оператора мутації для дослідження ступеню впливу факторів на фінансові показники аптек та вибору моделі оптимізації. У роботі досліджуються проблеми та існуючі методи оптимізації фінансових показників мережевих аптек, розробляється генетичний метод з модифікацією оператора мутації для вирішення проблеми управління асортиментом аптечної продукції заснований на еволюційних методах. В результаті використання запропонованих методів і засобів виявляються найбільш значущі, з точки зору підвищення ефективності, процеси прийняття рішень в системі управління асортиментом.

У цьому розділі виконано дослідження методів генетичної оптимізації асортименту підприємств аптечної торгівлі.

3.1 Постановка завдання

Фінансові показники аптек, такі, як прибуток і оборот, залежать від цілої низки факторів: об'єм продаж, наявний асортимент, розвиток інфраструктури, географічне положення аптеки і т.ін.

Це означає, що врахувавши всі статистично значимі фактори, можна впливати на значення фінансових показників, а також будувати прогнозні моделі, використовуючи методи математичної статистики.

Завдання оптимізації – це знаходження таких значень змінних функції, при яких функція досягає екстремуму. Поставлене завдання є частинним випадком класичного завдання оптимізації функції .

Метою розділу є дослідження методів оптимізації показників ефективності аптечних мереж з використанням штучних нейронних мереж (ШНМ) та генетичних алгоритмів.

В ході проєктування необхідно розробити модель та програмне рішення для дослідження ступеню впливу факторів на фінансові показники аптек та вибору моделі оптимізації.

Прибуток аптеки залежіть від таких базових факторів:

– сума продажу;

– валовий прибуток;

– довжина складу;

– показники захворюваності населення хворобами.

Математично це можна виразити як функцію від багатьох змінних:

$$F(x) = f(x_1, x_2, x_3, x_4),$$

де $F(x)$ – умовний фінансовий показник; x_1 – значення суми продажів; x_2 – значення валового прибутку; x_3 – значення довжини складу аптеки; x_4 –значення показників здоров’я від обсягів викидів забруднюючих речовин у повітря.

У цій роботі синтезується модель залежності фінансових показників від впливу асортименту продукції аптечної мережі. Перевіряється гіпотеза щодо впливу процентного співвідношення між групами товарів в асортименті на прибутковість аптеки.

Дані для аналізу пропонується зберігати у СУБД або у CSV, вихідні дані – відображати в додатку з можливістю експорту у текстовий файл.

3.2 Аналіз літературних даних та постановка проблеми

Оптимізація асортименту в аптеці – це комплекс заходів, спрямований на кількісну і якісну зміну його структури з метою підвищення раціональності і ефективності. Основна мета – гармонізувати асортимент, що приведе до мінімізації витрат і збільшення прибутку аптеки.

Широко поширеним способом оптимізації асортименту є аналіз дефектури, що дозволяє виявити попит на відсутні препарати шляхом фіксації попиту покупців первостольником в обліковій системі або на паперових носіях (журнал незадоволеного попиту, або обліку дефектури). Проте він має і свої недоліки, оскільки можливе спотворення даних за рахунок недостатньо ефективної системи відстежування дефектури. Окрім цього, не завжди можна отримати об'єктивні дані, оскільки дослідження споживчих переваг тільки в одній торговій точці часто не є репрезентативним [1].

У роботі [2] запропоновано АВС-аналіз для проведення повноцінного аналізу асортименту. Досить часто в аптеках для дослідження асортименту використовують АВС-аналіз. Ідея методу АВС аналізу будується на підставі принципу Парето: «за більшість можливих результатів відповідає відносно невелике число причин», зараз більш відомого як «правило – 20 на 80». Даний метод аналізу отримав великий розвиток, завдяки своїй універсальності і ефективності.

За допомогою цього аналізу товари розбиваються за ступенем впливу на загальний результат. Причому принципом групування може бути величина виручки, що отримується від конкретної групи продуктів, обсяг продажів або будь-які інші параметри. Часто виручка більш показова в якості критерію групування. Групування за обсягом продажів може бути адекватна в тому випадку, якщо аналізовані товари однорідні за складом і ціною.

Таким чином, вивчаючи роздрібні продажі, ми виділяємо групу «А» (позиції, сума часток з накопичувальним підсумком яких, становить перші 50% від загальної суми параметрів), групу «В» (товари, сума часток з накопичувальним підсумком яких становить від 50 до 80 від загальної суми параметрів) і групу «С» (решта товари, сума часток з накопичувальним підсумком яких становить від 80% до 100% від загальної суми параметрів). Після проведення АВС аналізу

по групах продукції, той же аналіз проводиться всередині груп, але не всіх, а вибірково, наприклад, тільки що входять до груп «А» і «В» або вибірково.

Очевидно, що необхідно контролювати наявність в асортименті товарних позицій класу «А». По відношенню до товарних позицій класу «В» контроль може бути поточним, а по відношенню до позицій класу «С» – періодичним. Таким чином, в ході АВС-аналізу формується АВС-рейтинг товарів.

Робота [3] надає опис використаного XYZ-аналізу, який допомагає оцінювати і порівнювати стабільність продажів товарних груп або окремих товарів різного типу попиту або різних цінових категорій. Застосовується для оптимізації товарних запасів і визначення частоти замовлення товару.

XYZ-аналіз використовує показник, що характеризує потреби аптеки в товарному запасі.

Алгоритм проведення XYZ-аналізу включає наступні етапи:

1) вибір об'єктів аналізу (товарна група, товарна одиниця, постачальники, клієнти і т.п.);

2) визначення параметрів аналізу (одиниці продажів, обсяг продажів, дохід, середній запас, кількість замовлень і т.п.) і періоду аналізу (тиждень, місяць, квартал, півріччя, рік);

3) визначення коефіцієнтів варіації для аналізованих ресурсів;

4) групування ресурсів відповідно до зростання коефіцієнта варіації;

5) розподіл за категоріями X, Y, Z;

6) графічне представлення результатів аналізу.

Результатом XYZ-аналізу є виділення 3 груп товарів:

– категорії X – групи товарів зі стабільною величиною споживання і, отже, високими можливостями прогнозування попиту;

– категорії Y – групи товарів з відомими сезонними коливаннями і середніми можливостями прогнозування;

– категорії Z – група товарів з нестабільним попитом і, як наслідок, низькою точністю прогнозування попиту.

Розподіл товару на категорії X, Y і Z визначається значенням коефіцієнта варіації. Відповідно до класичного підходу до категорії X відносяться товари з коефіцієнтом варіації 0-10%, до категорії Y – 0-25%, а до категорії Z – більше 25%. Однак в аптеках доцільно

використовувати великі інтервали, тому що використання класичного підходу часто призводить до того, що значна кількість товарів невиправдано потрапляє в категорію Z. Тобто стосовно до аптечного асортименту категорія X повинна включати товари з коефіцієнтом варіації 0-15%, категорія Y – 15-40%, категорія Z – більше 40 %.

Робота [4] надає опис використаного аналізу асортименту за методом Дібба-Симкина. Аналіз асортименту за методом Дібба-Симкина здійснюється для класифікації товарів і дозволяє визначити основні напрямки розвитку окремих товарних груп, виявити пріоритетні позиції асортименту, оцінити ефективність структури асортименту та шляхи її оптимізації. Для аналізу використовуються дані про динаміку продажів і собівартості продукції. На основі співвідношення обсягу продажів у вартісному вираженні і вкладу в покриття витрат товар відноситься до однієї з 4-груп.

Класифікація виділених груп в даному методі показує наступне:

1.Група А. Це найбільш значуща для аптеки група. Товари, що входять до цієї групи, можуть служити еталонами при виборі нового товару для включення в асортимент. Необхідно прагнути до збільшення числа товарних позицій в даній групі, так як приріст продажів саме цих товарів має найбільший вплив на прибуток підприємства.

2.Група В1. Це група товарів для яких слід виявляти шляхи підвищення прибутковості цієї продукції (можливості зростання цін, пошук більш вигідних постачальників для зниження собівартості і т.п.).

3.Група В2. Необхідно шукати можливості для збільшення продажів продукції даної товарної групи (проведення промоакцій, реклама і т.п.). Завдяки високій рентабельності продукції даної групи темпи зростання прибутку підприємства будуть вищими за темпи зростання продажів цих товарів.

4.Група С. Це найменш цінні для підприємства товари, тому необхідно розглянути можливості заміни ряду товарів з даної групи, а також оцінити ефективність виключення найменш прибуткових товарів.

Робота [5] надає опис використаного методу аналізу чеку. Завдання аналізу структури чеків – це отримання інформації, необхідної для прийняття рішень щодо корекції або зміни структури асортименту в

залежності від розташування аптеки, сезонності продажів та інших факторів, що впливають на асортиментну політику.

В ході проведення аналізу структури чеків виявляють зміни в структурі і сумі чека в залежності від часу доби, дня тижня і сезону. Основний показник при такому аналізі – сума середнього чеку, яку розраховують як відношення загального обсягу продажів до кількості чеків за певний період часу.

Крім розміру середнього чека, аналізуючи структуру чеків, слід розглянути зміна кількості покупців, суми покупок і товарообігу аптеки в середньому по днях тижня, а також проаналізувати зміну кількості позицій в чеках по різним діапазонам суми тощо.

За допомогою диференційованого аналізу чеків виявляють: найчастіші позиції (товари) в чеках; найбільші чеки; чеки різних груп покупців; чеки в різний час доби. Диференційований аналіз також дозволяє розділити покупців аптеки на групи і оцінювати їх купівельні кошики (склад чеків різних груп покупців); виявляти спільно придбані товари. Розміри максимального, середнього і мінімального чека в аптеці є показником платоспроможності основного купівельного контингенту. Результати аналізу чеків дозволяє судити про асортимент і цінову політику аптеки, проводити порівняння з конкурентами (порівнюючи розмір середнього чека).

Безпосередньо вивчення структури чеків проводиться з метою аналізу поточної діяльності аптеки для визначення основних тенденцій в її роботі, які допоможуть підготуватися до сезонних коливань попиту і зміни в зв'язку з цим споживчих переваг. Аналіз структури чеків дозволяє виявити сильні і слабкі сторони аптечного закладу, успішно конкурувати і задовольняти потреби покупця.

Стохастичні (імовірнісні) моделі [6] широко застосовуються в тих випадках, коли ті або інші чинники носять невизначений характер. Такі ситуації характерні для самих різних областей людської діяльності. Прикладами можуть служити погодні умови через декілька років, попит на яку-небудь продукцію і т. п. Стохастичні (імовірнісні) моделі управління запасами припускають, що інтенсивність споживання відповідного виду матеріальних ресурсів – це випадкова величина, розподіл якої може бути описаний тим або іншим статистичним законом.

Необхідно представити задачу так, щоб її рішення можна було б записати у вигляді генотипу, тобто вектору значень (генів). Оптимальною буде така стратегія управління асортиментом, яка мінімізує суму всіх витрат, пов'язаних із створенням, зберіганням і нестачею запасів, в одиницю часу або за певний (у тому числі і нескінченний) проміжок часу. У найзагальнішому вигляді завдання управління асортиментом зводиться до знаходження такого розміру асортименту *xt* у момент часу *t*, який мінімізує загальну функцію витрат: $F(xt) \rightarrow \min$.

Управління асортиментом полягає у відшукуванні такої стратегії поповнення і витрати запасами, при якому функція витрат набуває мінімального значення. Проста модель управління запасами представлена таким чином. Нехай функції $A(t)$, $B(t)$ виражають відповідно поповнення запасів, їх витрату за проміжок часу $[0, t]$. Рівень запасу у момент t визначається основним рівнянням запасів:

$$F(t) = F_0 + A(t) - B(t),$$

де F_0 – початковий запас в момент t=0 [7].

Аналіз [2–7] дозволяє стверджувати, що розробка методів оптимізації фінансових показників є доволі актуальною задачею. Досліджено проблеми та існуючі методи управління асортиментом мережевих аптек. На основі досліджених методів аналізу асортименту можна зробити висновок про те, що комплексні методи аналізу асортименту показують необхідність розгляду цілої групи показників ефективності асортименту. Склад цих показників і їх вплив на підсумкову оцінку відрізняється в залежності від особливостей асортименту, самої аптеки і сформованої ринкової кон'юнктури. Тому методи [2–5] повинні не тільки вибирати, але і адаптуватися виходячи їх поточної ситуації на конкретній аптеці. Проаналізувавши методи [6–7] управління асортиментом, можна зробити висновок що лінійні моделі не володіють повним набором можливостей для вибору варіантів структури асортименту, оскільки вони дозволяють отримати оптимальне рішення тільки на один плановий період і не розглядають його зв'язок з показниками попереднього і наступного періоду. Тому для оптимізації фінансових показників мережевих аптек та підвищення ефективності їх діяльності доцільно використовувати

більш складні нелінійні, зокрема нейромережеві, моделі, які дозволять апроксимувати складні багатовимірні нелінійні залежності з високою точністю.

3.3 Розробка модифікації оператора мутації при розв’язанні задачі оптимізації фінансових показників аптек

Задача оптимізації, що виникла і яка розв’язується в даній роботі, характеризуються великою кількістю змінних, і, як наслідок, великим обсягом простору пошуку, що не дає можливості дослідити все різноманіття рішень за прийнятний час. У зв'язку з цим виникла проблема практичної можливості розв'язання даної задачі оптимізації: знайти ефективний або хоча б досить простий в практично важливих випадках алгоритм її вирішення. Для розв’язуання даного завдання було прийнято рішення використовувати еволюційні методи [8], які у порівнянні з методами повного перебору дозволять скоротити обчислювальні витрати і вирішити задачу оптимізації швидше і ефективніше.

Генетичні алгоритми [8] на даний момент є найбільш відомим представником еволюційних методів оптимізації.

Генетичні алгоритми (ГА) є напрямком теорії еволюційних алгоритмів, заснованої на принципі: «кожен біологічний вид цілеспрямовано розвивається і змінюється для того, щоб найкращим чином пристосуватися до навколишнього середовища».

Один з недоліків відомих еволюційних алгоритмів [8] полягає у відсутності механізму врахування обмежень оптимізаційного завдання.

Тому для усунення зазначеного недоліку у цій роботі запропоновано генетичний метод багатокритеріальної оптимізації з модифікацією оператора мутації для вирішення завдання оптимізації. Розглянемо деяку систему, що складається з двох підсистем, які описуються багатоекстремальними рівняннями:

$$z_1 = z_1(x_1, y_1), \tag{3.4}$$

$$z_2 = z_2(x_2, y_2), \tag{3.5}$$

де x_1, y_1, x_2, y_2 – параметри системи; z_1, z_2 – цільові функції її функціонування [9].

Сформулюємо проблему розв'язання задачі багатоцільової оптимізації такої системи:

$$P^* = P(z_1^*, z_2^*) = \max P(x_1, y_1, x_2, y_2), \tag{3.6}$$

де P – комплексна цільова функція; $x_{1min} \le x_1 \le x_{1max}$; $x_{2min} \le x_2 \le x_{2max}$; $y_{1min} \le y \le y_{1max}$; $y_{2min} \le y_2 \le y_{2max}$.

В даному випадку функція P є, по суті, компонентом багатоцільового показника якості $P\{z_1,z_2\}$ і перетворює сукупність таких компонент в скалярний цільовий показник [10].

Одним з найбільш поширених підходів до врахування обмежень є метод штрафних функцій [9], основна ідея якого полягає в тому, що придатність індивіда обчислюється не тільки в залежності від відповідного йому значення цільової функції, а й від міри порушення обмежень:

$$fitness(x) = f(x) + \delta \cdot \lambda(t) \cdot \sum_{j=1}^{m} f_j^{\beta}(x), \tag{3.7}$$

де t – номер покоління; δ=1, якщо вирішується завдання мінімізації; δ= -1, якщо вирішується завдання максимізації; $f_j(x)$ – штраф за порушення j-го обмеження; β – дійсне число $\lambda(t)=(C \cdot t)^{\alpha}$ [11].

У вибраному методі штрафних функцій обчислення значення $f_j(x)$ відбувається динамічно, залежно від міри порушення обмежень, по формулі для t-й ітерації, а значення $\lambda(t)=(C \cdot t)^{\alpha}$:

$$f_i(x) = \begin{cases} \max\{0, g_i(x)\}, j = \overline{1,r}, \\ |h_i(x)|, j = \overline{r+1,m}, \end{cases}. \tag{3.8}$$

де $g_i(x) \le 0$; $h_j(x)=0$ – обмеження задачі [12].

Отже, формула (3.7) придатності індивіда має вигляд:

$$fitness(x) = f(x) + \delta \cdot \lambda (C \cdot t)^{\alpha} \cdot \sum_{j=1}^{m} f_j^{\beta}(x), \qquad (3.9)$$

Перевагою методу динамічних штрафів [13] є те, що він вимагає набагато менше параметрів, ніж інші методи штрафних функцій. Замість вибору з набору фіксованих рівнів порушення обмежень в даному методі штраф розраховується динамічно.

Для покращення якості роботи алгоритму та розширення його можливостей було розроблену модифіковані оператори мутації.

Перша модифікація оператора мутації складається у наступному: нові значення генів для модифікації обирається не як випадкове число, а з ряду випадкових чисел, що підпорядковуються закону нормального розподілення [14].

Алгоритм когнітивно-стильової детермінації виконується у наступній послідовності:

– обрання хромосоми для мутації;

– генерація умовно-випадкового масиву чисел за законом нормального розподілення, що за розміром дорівнює розміру хромосоми та медіанна точка та середньоквадратичне відхилення розподілення співпадає з медіанною точкою ряду;

– вибір генів для мутації випадковим чином;

– заміна генів на значення з нормалізованого випадкового ряду;

– повернення хромосоми у популяцію.

У хромосомі $A=a_1a_2\ldots a_n$ випадковим чином вибирається k позиція (біт) $1 \le k \le n$. Далі проводиться інверсія значення гена в k позиції: $a_k^{'}=a_k$ [15].

У когнітивно-стильової детермінації значення гена після оператора мутації розраховується за формулою:

$$c_i^* = \begin{cases} c_i + \delta(t, b_i - c_i), \text{ при } x = 0; \\ c_i - \delta(t, b_i - a_i), \text{ при } x = 1, \end{cases} . \qquad (3.10)$$

$$\delta(t,y) = y\left(1 - r^{\left(1-\frac{t}{\varepsilon_{max}}\right)^b}\right), \tag{3.11}$$

де x – ціле випадкове число, що приймає значення 0 або 1; $r \in [0,1]$ – випадкове дійсне число; ε_{max} – максимальна кількість епох алгоритму; b – параметр, що задається дослідником [16].

Крім того, якщо протягом досить великого числа поколінь не відбувається збільшення пристосованості, то застосовуються «мала» і «велика» мутації покоління. При «малої» мутації покоління до всіх особинам, крім 10% кращих, застосовується оператор мутації. При «великий» мутації кожна особина або мутує, або замінюється на випадково згенерувала [17].

Алгоритм Noetic (інтелектуальної) мутації полягає у використанні ШНМ в процесі мутації. Однією з цілей запропонованої модифікації є забезпечення тільки "позитивної" мутації, тобто такої, яка покращує фенотип хромосоми. Такий оператор виконується у наступній послідовності:

– обрання хромосоми для мутації;

– застосування стандартної мутації;

– використання раніше навчену ШНМ для прогнозування прибутковості, передавши на вхід мережі "оригінальну" хромосому та "мутовану";

– порівняти відповіді ШНМ і, тільки якщо "мутована" хромосома забезпечує кращу прибутковість, додати її в популяцію. В іншому випадку в популяцію повертається "оригінальна" хромосома [18].

Математично, штучний нейрон зазвичай представляють, як деяку нелінійну функцію від єдиного аргументу – лінійної комбінації всіх вхідних сигналів. Цю функцію називають функцією активації або функцією спрацьовування, передавальною функцією [19].

Функціонування нейрону можна описати формулою:

$$y = \begin{cases} 1, & \sum_{i=1}^{N} w_i u_i \geq V \\ 0, & \sum_{i=1}^{N} w_i u_i < v, \end{cases} . \tag{3.12}$$

де y – вихідний сигнал нейрона; w_1, …, w_N – синаптичні вагові коефіцієнти; u_1, …, u_N – вхідні сигнали ШН; v – порогове значення [20].

Модель (3.13) може буде подана у вигляді:

$$y = f(\sum_{i=0}^{N} w_i u_i), \tag{3.13}$$

де $w_0 = v$, $u_o = 1$.

Третім типом модифікації оператора мутації є комбінація попередніх двох методів.

Алгоритм Merger (об'єднавчої) мутації виконується у наступній послідовності:

– обрання хромосоми для мутації;

– застосування "нормалізуючої" мутації;

– використання раніше навчену ШНМ для прогнозування прибутковості, передавши на вхід мережі "оригінальну" хромосому та "мутовану";

– порівняти відповіді ШНМ і, тільки якщо "мутована" хромосома забезпечує кращу прибутковість, додати її в популяцію.

В іншому випадку в популяцію повертається "оригінальна" хромосома [21]. В якості функції активації f, що наведена у формулі (3.13), можна використовувати функцію (3.14):

$$f(x) = \begin{cases} 1, x \geq 0, \\ 0, x < 0, \end{cases} . \tag{3.14}$$

Окрім виразу (3.14) у якості функції активації f також можуть застосовуватися порогові функції [22] вигляду (3.15) та (3.16)

$$f(x)=\begin{cases} 1, x \geq 0, \\ -1, x < 0, \end{cases}. \tag{3.15}$$

$$y=\begin{cases} 1, & x>1, \\ -1, & x<-1, \\ x, & |x| \leq 1, \end{cases}. \tag{3.16}$$

Розроблено генетичний метод багатокритеріальної оптимізації з модифікацією оператора мутації, який переважає по надійності і швидкості у порівнянні з методами повного перебору. Окрім цього, модифікований генетичний алгоритм, наділений способами врахування обмежень, є ефективним інструментом для вирішення задачі оптимізації асортименту в аптеці. Це, у свою чергу, призведе до більш ефективного використання площі аптек, зменшення незадоволеного попиту та, в кінцевому результаті, до зменшення роздрібної вартості ліків за рахунок зменшення видатків на зберігання та обслуговування неоптимальне завантажених площин аптеки.

3.4 Програмне забезпечення для прогнозування фінансових показників аптек

Розроблене програмне забезпечення складається з таких модулів:

– модуль зберігання даних;

– розрахунковий модуль;

– модуль введення-виведення даних;

– модуль зв'язку всіх елементів системи.

Також елементи системи повинні забезпечувати достатню швидкодію та здатність до масштабування.

Основними об'єктами вибору про проєктуванні є мова програмування системи, система управління базами даних (СУБД) та середовище розробки (Integrated Development Environment – IDE).

3.4.1 Модуль зберігання данних

Дані, які будуть використані для навчання та тестування ШНМ, а також для ГА, потребують вирішення питання збереження.

У якості навчальних наборів даних вирішено використовувати дані продажів мережі аптек за 12 місяців. Ці дані можна зберігати у таких типах сховищ:

– текстові файли (CSV, txt, xml,...);
– бінарні файли даних;
– не реляційні бази даних (NoSQL сховища);
– реляційні бази даних (MySql, MS SQL Server, Firebird,...);
– багатовимірні сховища даних (OLAP сховища).

Обраний тип зберігання повинен забезпечувати достатню гнучкість форматів, високу швидкість отримання даних, здатність зберігати великі об'єми даних, здатність до розширення функціоналу.

Всім цим вимогам відповідають реляційні СУБД та OLAP-сховища.

На даний момент існує великий вибір типів РСУБД: Microsoft SQL Server, Oracle RDBMS, Firebird, MySql, MariaDB та ін.

Серед цих виробників лише Microsoft та Oracle мають інтегровані рішення для створення OLAP сховищ. При цьому СУБД Microsoft SQL Server забезпечує високий рівень розширюваності для програміста та велику кількість інструментів керування БД.

З огляду на попередні аргументи, у якості системи зберігання даних обрано MS SQL Server 2016.

3.4.2 Технологічні рішення для оптимізації

Мова програмування R призначена, в основному, для роботи з даними. Однією з переваг даної мови програмування є наявність багатого репозиторію пакетів розширення, що розміщені у вільному доступі. Пакети розширення, що використовувались у даній роботі, наведені в таблиці 3.1.

Використання пакетів значно спрощує розробку та захищає від можливих помилок. Особливістю використаних пакетів є те, що більшість з них дозволяє значну кастомізацію як з залученням

оригінального програмного коду, так і за рахунок інтерфейсів розширення.

Таблиця 3.1 – Використані пакети R

Назва пакету	Призначення
GA	Робота з еволюційними алгоритмами
Keras	Робота з ШНМ
Xts	Робота з часовими рядами
Forecast	Робота з регресійними моделями
Stats	Бібліотека статистичних функцій
Nortest	Набір тестів нормального розподілення

3.4.2 Структурна схема програми

Стек програми складається з трьох головних рівнів: рівня даних, рівня застосунку та рівня інтерфейсу.

На рівні даних розміщено модулі первинного завантаження даних, а також бази даних, що використовуються в якості джерел даних для розрахунків та зберігання результатів.

На рівні застосунку розташовано модулі, які безпосередньо здійснюють обчислення: модулі ШНМ, лінійних моделей, та еволюційних алгоритмів.

Модулі рівня інтерфейсу забезпечують графічний інтерфейс взаємодії користувача та розробленої системи.

Обмін даними між рівнями здійснюється через програмну шину обміну даними.

Структурна схема розроблюваної програми представлений на рисунку 3.1.

3.4.3 Опис модулів розроблюваної системи

Рівень даних складається з двох основних модулів – модуль завантаження, обробки та зберігання даних (ETL(англ.) – Extract Transform Load) , та модуль зберігання даних СЗД (система зберігання даних).

Також окремим підмодулем рівня є модуль для збереження даних моделей ШНМ та ГА (БД MODEL).

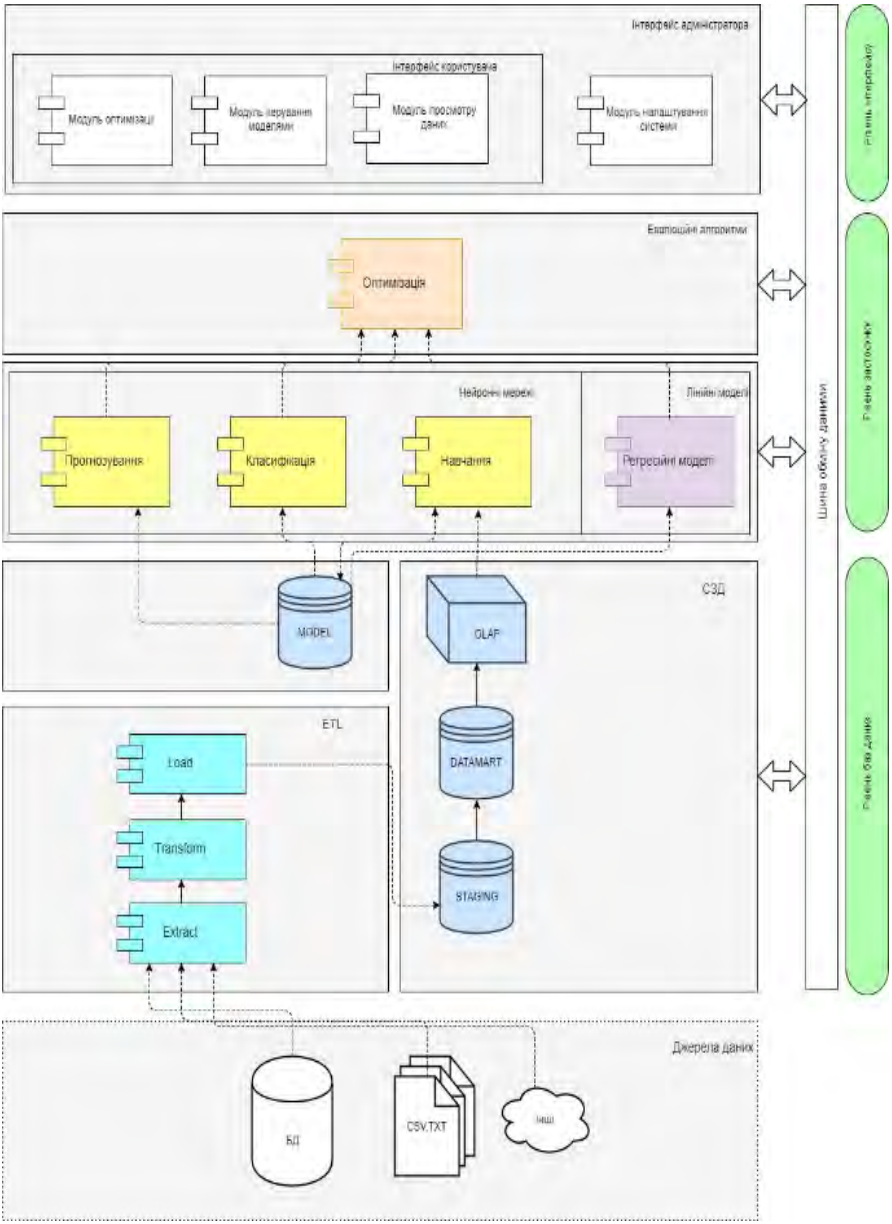


Рисунок 3.1 – Структурна схема програми

Усі модулі рівня даних реалізовано з використанням стеку технологій Microsoft SQL Server 2016.

Модуль ETL реалізовано за допомогою технології SQL Server Integration Services (SSIS).

Модуль СЗД містить 2 реляційні БД STAGING та DATAMART.

БД STAGING використовується для зберігання даних, завантажених із зовнішніх джерел, у вигляді, придатному для подальшої трансформації та завантаження до основної БД DATAMART.

БД DATAMART призначена для зберігання очищених та упорядкованих даних. Дані у цій БД нормалізовані та пов'язані ключами. БД DATAMART слугує джерелом даних для багатовимірної БД OLAP.

БД OLAP являє собою загрегований OLAP - куб даних. Використання кубів даних в якості джерела даних для розрахунку моделей у порівнянні з реляційними БД дає значний приріст у продуктивності розрахунків, а також значно пришвидшує розробку за рахунок попередньої агрегації всіх можливих розрізів даних.

Для реалізації модуля СЗД використовуються технології SQL Server Database Engine для реляційних БД STAGING та DATAMART, та SQL Server Analysis Services (SSAS) для кубу даних OLAP.

На рівні застосунку розташоване розрахункове ядро системи - модулі ШНМ, ГА та модуль регресійних моделей.

Модуль ШНМ містить три субмодулі:

– субмодуль тренування (навчання) ШНМ призначений для тренування ШНМ на даних, отриманих з БД OLAP рівня даних. Після тренування ваги тренованої моделі зберігаються у БД MODEL рівня даних.

– субмодуль класифікації використовує ШНМ для рішення завдання оптимізації за допомогою класифікації. Він використовує дані ваг та моделей з БД MODEL.

– субмодуль прогнозування використовує ШНМ для рішення завдання оптимізації через розрахунок прогнозу. Він використовує дані ваг та моделей з БД MODEL.

Модуль регресійних моделей призначений для рішення завдання оптимізації через розрахунок прогнозу методами регресійного аналізу. Дані для розрахунку беруться з БД OLAP.

Результати роботи модулів прогнозування та класифікації використовуються модулем оптимізації блоку "Еволюційні алгоритми". Цей модуль за допомогою генетичних алгоритмів обирає найбільш підходящу модель з доступних розрахункових для вирішення завдання оптимізації за набором параметрів.

Технічно модулі рівню застосунку реалізовані у вигляді програмних модулів мови програмування R.

Рівень інтерфейсу являє собою набір візуальних форм, створених на платформі .NET мовою програмування C#. Основні функції модулів рівня інтерфейсу:

– модуль оптимізації – призначений для запуску розрахунку оптимізації та виведення результатів;

– модуль керування моделями – призначений для керування розрахунковими моделями – ручного коригування вагів ШНМ, коефіцієнтів регресії тощо;

– модуль просмотру даних – призначений для візуалізації вихідних даних та результатів розрахунків;

– модуль налаштувань системи – частина інтерфейсу адміністратора, що призначена для виконання системних операцій та глобальних налаштувань системи.

Шина обміну даними. Важливою складовою системи є шина обміну даними, за допомогою якої здійснюється зв'язок між різними модулями рівнями системи. У даному проєкті шина обміну даними реалізована в якості DLL–бібліотеки, розробленої на платформі .NET за допомогою мови програмування C#.

Бази даних. У розробленій системі використовується декілька БД, кожна з яких має своє функціональне призначення. Використання декількох БД замість однієї дає змогу знизити функціональну зв'язність та покращує можливості масштабування системи.

Також це позитивно впливає на швидкодію системи в цілому.

В проєкті задіяні 3 основні БД:

– STAGING;

– DATAMART;

– MODEL.

Також у проєкті задіяна нереляційна OLAP БД "OLAP".

БД STAGING призначена для зберігання "сирих" даних, отриманих після завантаження з первинних систем.

На рисунку 3.2 подана ER–діаграма БД STAGING.

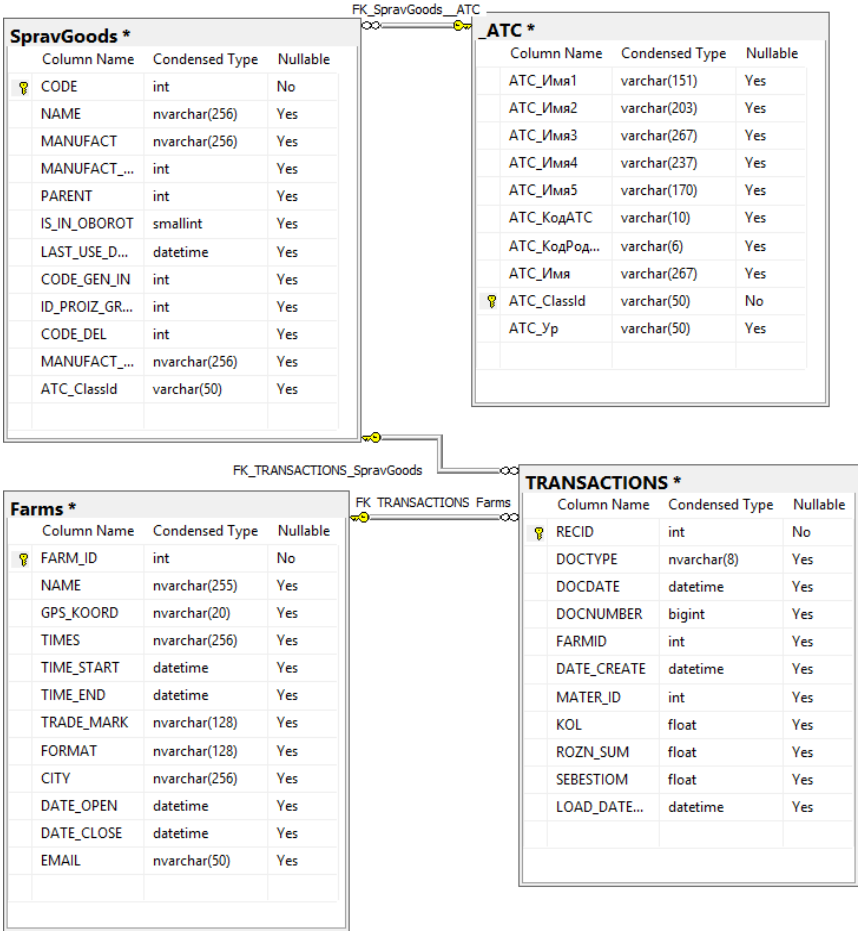


Рисунок 3.2 – ER–діаграмма БД STAGING

БД DATAMART призначення для зберігання т.з. "вітрини даних". В ній зберігаються дані, очищені, нормалізовані та предагреговані для побудови агрегованого OLAP-кубу. До даних у цій БД пред'являються підвищені вимоги з якості.

ER-діаграма БД представлена на рисунку 3.3.

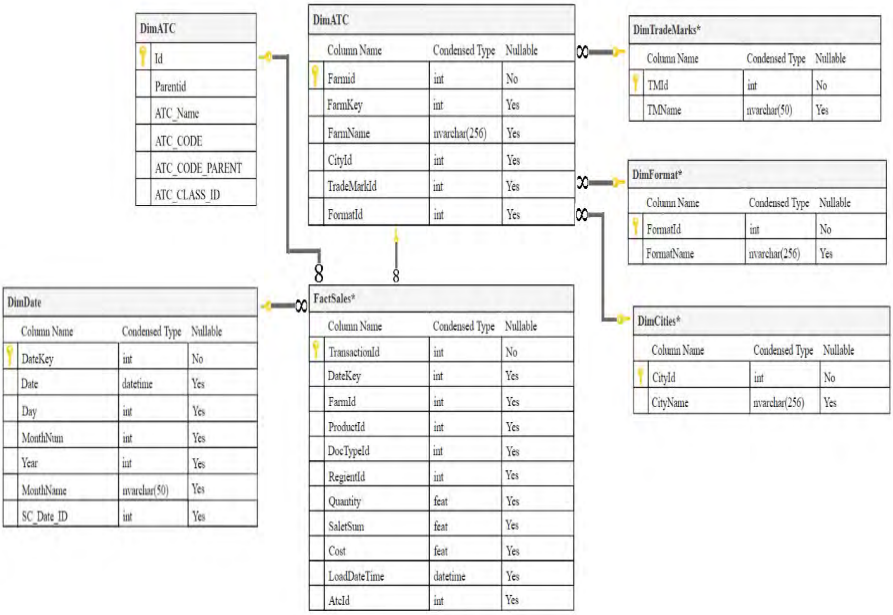


Рисунок 3.3 – ER–діаграмма БД DATAMART

БД MODEL зберігає дані, отримані у процесі навчання ШНМ, такі, як топологія мережі та розраховані ваги.

3.4.4 Звертання до програми (файл проєкту)

Проєкт на мові програмування R містить наступні основні блоки:

– ProcessDatasetFarms.r – модуль для завантаження даних з БД і обробки їх для передачі у розрахункове ядро;

– LoadDataFromCSV.r – модуль для завантаження даних з CSV і обробки їх для передачі у розрахункове ядро;

– Regressions.R – розрахунки регресійних моделей;

– MLP.R – побудова і навчання ШНМ;

– AssortmentOptimizationGA.R – оптимізація методами ГА;

– genOps.R – розроблені модифікації генетичних операторів;

– functions.R – допоміжні функції.

3.4.5 Вхідні і вихідні дані

У якості вхідних даних для програми виступають дані продаж, залишків та прибутку.

Ці дані можуть зберігатися в БД, або можуть бути надані в форматі CSV.

Файл даних продаж повинен мати наступу структуру: обов'язковий стовпчик Period, в якому зберігається дата звітного місяця, та стовпчики, названі іменами змінних (наприклад, "ATC406"), в яких містяться дані продаж.

Файл даних прибутку повинен мати наступу структуру: обов'язковий стовпчик Period, в якому зберігається дата звітного місяця, та стовпчики, названі іменами змінних (наприклад, "ATC406"), в яких містяться дані приросту прибутку по відношенню до попереднього періоду.

Файл даних залишків повинен мати наступу структуру: обов'язковий стовпчик Period, в якому зберігається дата звітного місяця, та стовпчики, названі іменами змінних (наприклад, "ATC406"), в яких містяться усереднені дані залишків.

В якості вихідних даних слугують дані запропонованого розподілу груп товарів, що відображаються в екранній формі.

Повідомлення програми містяться у системних логах та виводяться на екранну форму. Система генерує повідомлення у наступних випадках:

– дані завантажено успішно;
– помилка завантаження даних;
– розрахунок почато;
– розрахунок закінчено;
– помилка розрахунку;
– необроблена помилка.

3.4.6 Виконання програми

Головний екран програми містить 3 вкладки: "Завантаження даних", "Оптимізація даних", "Тренування ШНМ".

Для запуску програми потрібно запустити на виконання файл FarmAssoGui.exe.

Екран "Завантаження даних" призначений для завантаження та початкового профілювання даних.

Для завантаження даних з БД необхідно натиснути кнопку "Завантажити дані з БД".

Для завантаження даних з текстового файлу необхідно натиснути вибрати файл в діалогу вибору файла та натиснути кнопку "Завантажити з файлу".

Після завантаження відображаються профілюючі дані (кількість строк, початок та кінець періоду даних).

В полі "Набір даних" відображається ім'я завантаженого файлу, в нижній частині форми відображаються завантажені файли.

Приклад екрану завантаження наведено на рисунку 3.4.

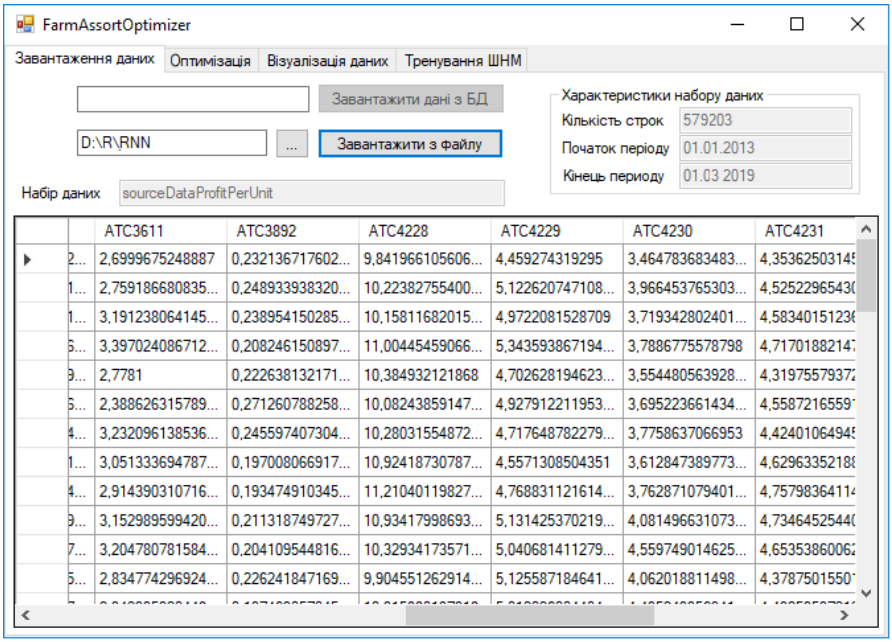


Рисунок 3.4 – Екран завантаження даних

Вкладка "Оптимізація" призначена для запуску розрахунку оптимізації.

Для старту розрахунку необхідно натиснути кнопку "Запуск оптимізації".

По закінченню розрахунку виводиться інформація:

– прогнозований приріст прибутку;

– прогнозована середня довжина складу;

– процентні співвідношення груп товарів в асортименті.

Приклад екрану оптимізації наведено на рисунку 3.5.

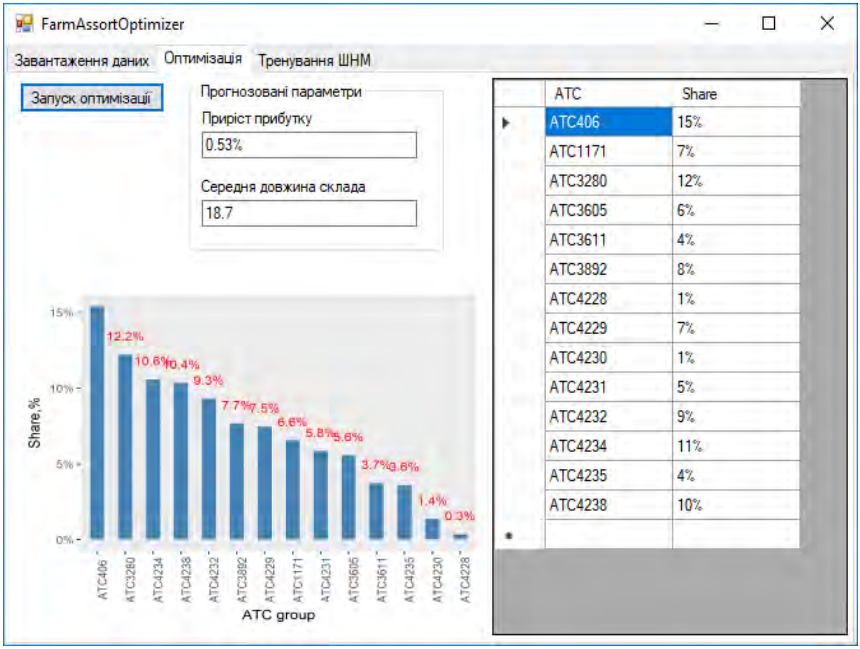


Рисунок 3.5 – Екран "Оптимізація"

В процесі роботи програми виникає необхідність періодично повторювати навчання ШНМ. Це може бути пов'язано з появою нових даних або з коригуванням даних, на яких раніше проводилося навчання.

Для цього призначено екран "Тренування ШНМ". Для старту процесу тренування необхідно натиснути кнопку "Запуск навчання ШНМ". По закінченню розрахунку на формі буде відображено параметри розрахунку:

– середня похибка мережі;
– середньо-квадратична похибка мережі.
Також буде відображено графік тренування.
Приклад екрану тренування ШНМ відображено на рисунку 3.6.

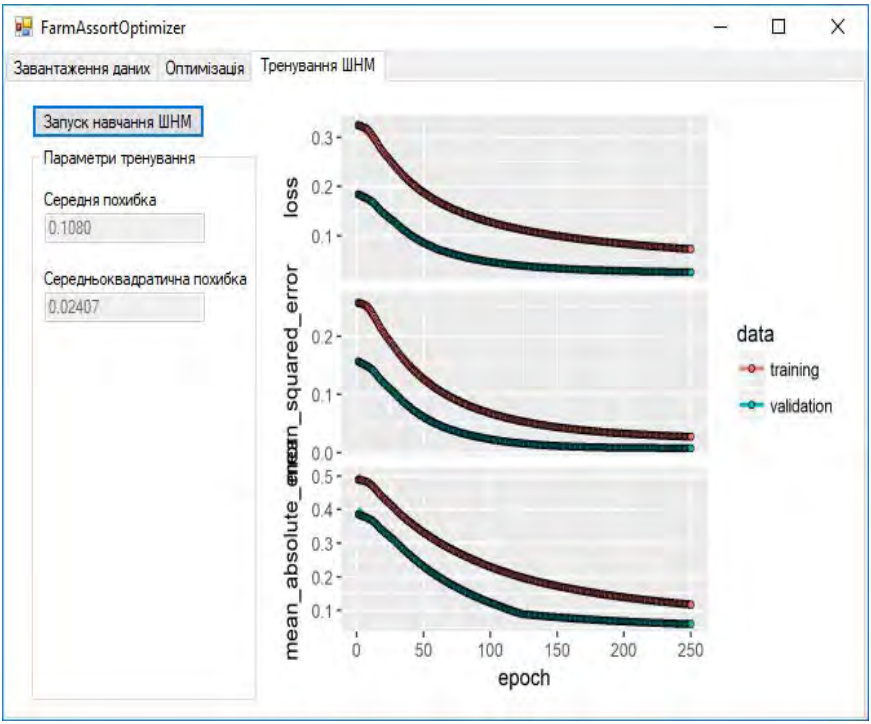


Рисунок 3.6 – Екран "Тренування ШНМ"

Програма повідомляє оператора у наступних випадках:
– розрахунок запущено;
– розрахунок завершено;
– помилка завантаження даних;
– помилка під час виконання розрахунку;
– необроблена помилка.
Повідомлення відображаються у вигляді стандартних вікон повідомлень Windows.

3.5 Експерименти та результати методів ініціалізації початкової популяції еволюційного алгоритму

Вибірка даних містила інформацію про структуру чека клієнтів аптечної мережі. Структура чека характеризується інформацією про наявність певних товарів в ньому. Основними ознаками (атрибутами), що характеризують структуру чека покупця є: x_1 – ідентифікатор клієнта (KeyCustomer) – унікальний номер, що дозволяє однозначно визначити конкретного клієнта аптечної мережі; x_2 – SKUQTY – кількість найменувань товарів у чеку; x_3 – ATCQty – кількість ATC груп; x_4 – SalesSum – сумарна вартість товарів у чеку; x_5 – MarginSum – торгова точка, яка здійснює продаж товару; x_6 – OrderQty – кількість чеків певного клієнта на момент поточної покупки; x_7 – AvrOrder- середній чек (грошова оцінка) певного клієнта; x_8 – OrderRowsQty – розмір знижки; x_9 – AvrPositionsQty – середня кількість товарів в чеку певного клієнта; x_{10} – структура товарів в чеку, що подана у вигляді:

$$x_{10} = \left\{ \left\langle t_i, C_i, S_i \right\rangle \right\}, \tag{3.17}$$

де t_i – найменування i-го товару в чеку; C_i – кількість i-го товару в чеку; S_i – вартість одиниці i-го товару в чеку.

Для вирішення завдання оптимізації фінансових показників аптек, як функції від розподілу товарних груп в асортименті аптеки, необхідно визначити такий процентний розподіл груп товарів, при якому будуть виконуватись наступні умови:

– максимізація прибутку аптеки;

– мінімізація часу перебування товару на складі ("довжини складу");

– покращення прогнозних показників прибутку аптеки;

– варіабельність асортименту аптеки, що забезпечує представленість максимальної кількості товарних груп в асортименті аптеки.

Типовий розподіл товарних груп в асортименті аптеки представлений на рис. 3.7 та рис. 3.8.

На рис. 3.7 наведено розподіл середньомісячної долі групи товару в асортименті аптеки у розрізі довжини складу на якому можна

побачити, що основний відсоток товару зосереджений в інтервалі від 20 до 24 довжини складу.

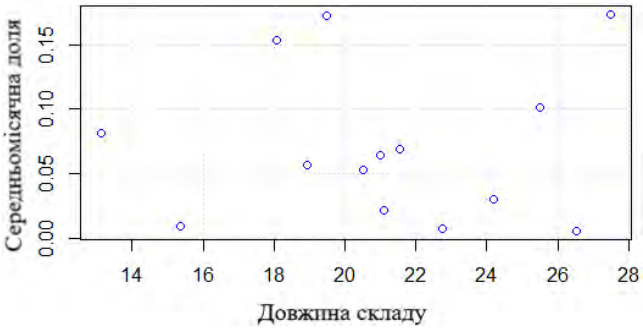


Рисунок 3.7 – Розподіл середньомісячної долі групи товару в асортименті аптеки у розрізі довжини складу

На рисунку 3.8 зображено статистичний розподіл груп товарів за довжиною складу, на якому можна побачити, що найчастіший розмір площі, яку займає товар знаходиться в діапозоні від 20 до 25 м.

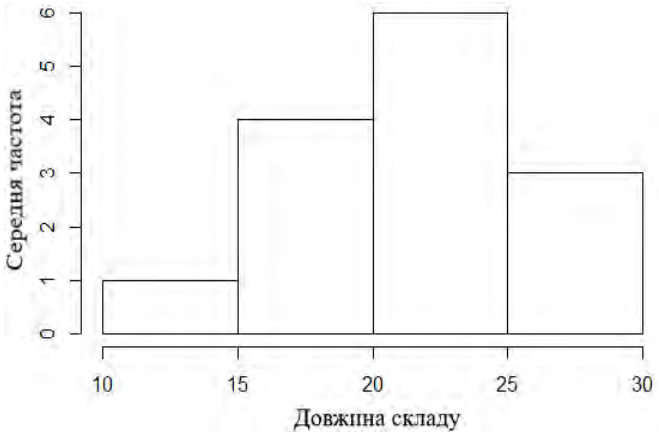


Рисунок 3.8 – Статистичний розподіл груп товарів за довжиною складу

Розподіл товарних груп за прибутковістю представлений на рисунку 3.9, на якому можна побачити залежність, що той товар у якого найбільший середньомісячний прибуток, має найменший час перебування на складі.

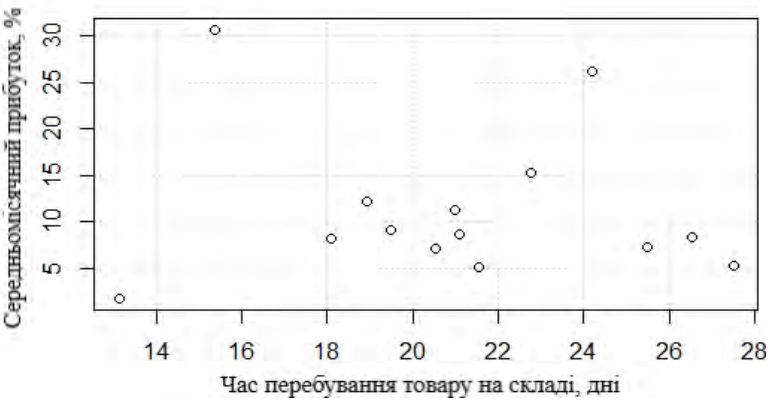


Рисунок 3.9 – Розподіл товарних груп за прибутковістю у розрізі довжини складу

У якості одиниці групи товару було обрано групу класифікації довідника лікарських засобів Моріон за діючим засобом (Анатомо-терапевтично-хімічна класифікація – міжнародна система класифікації лікарських засобів. Використовуються скорочення: латиницею АТС (від Anatomical Therapeutic Chemical)).

Для вирішення поставленого завдання можливе використання комбінації методів, суперпозиція результатів яких представляє узагальнений результат оптимізації.

Вихідними даними для усіх розрахунків слугують такі основні набори даних [23]:

– дані продаж помісячно за період 2013 - 2017 років;
– дані прибутку помісячно за період 2013-2017 років;
– дані залишків помісячно за період 2013-2017 років.

Ці набори даних є базовими, на їх основі формуються такі набори даних [24]:

– зміни прибутку по відношенню до попереднього періоду помісячно;
– пропорції розподілу груп товарів у залишку;
– "довжини складу" груп товарів;
– різноманітні усереднені дані.

Важливою вимогою до даних для моделювання є їх якість. Якщо дані містять т.з. "шум", сезонну компоненту, викиди, пропуски – це негативно впливає на точність прогнозів та якість моделей.

Також дані, призначені для використання в якості навчальних вибірок для ШНМ, повинні бути нормалізовані для зменшення похибки та покращення якості тренування.

Обробка сирих даних перед поданням моделі проходить у наступній послідовності [25]:

– очищення вибірок з невизначеними або пустими ключовими полями;

– обробка пропусків в даних предикторів;

– обробка аномалій даних предикторів;

– видалення сезонної компоненти з часових рядів;

– приведення даних до типів, що використовуються у розрахунках;

– нормалізація даних.

При обробці пропусків даних порожні значення замінюються на медіанне значення, розраховане за формулою (3.18):

$$Me = X_{Me} + i_M \frac{\frac{\sum f}{2} - S_{Me\ 1}}{f_{Me}}, \tag{3.18}$$

де X_{Me} – нижнє значення медіанного інтервалу; i_M – медіанний інтервал; S_{Me} – сума спостережень, що була накоплена до начала медіанного інтервалу; f_{Me} – кількість спостережень в медіанному інтервалі [26].

Таким чином, забезпечується мінімальна статистична погрішність від значень ряду.

Обробка аномалій в даних являє собою очистку набору змінних від аномально високих або низьких значень. Така очистка відбувається за допомогою функції InterQuartile Range [27].

Очистка від сезонної компоненти часового ряду відбувається за допомогою методу декомпозиції.

Для нормалізації даних використовується метод нормалізації MinMax [28]. Нормалізоване значення змінної x розраховується за формулою (3.19):

$$z = \frac{x\text{-}\min(x)}{\max(x)\text{-}\min(x)}. \tag{3.19}$$

Одним із методів оцінки ефективності сформованої корзини асортименту є забезпечення прибутку у прогнозному періоду. Для надання такої оцінки доцільно використовувати поширені методи прогнозування.

Прогнозування можна здійснювати за абсолютним значенням або за напрямком тренда.

До методів прогнозування за абсолютним значенням відносяться [29]:

- лінійна регресія;
- поліноміальна регресія;

Для прогнозування за напрямком тренду можна представити напрямок тренда, як набор дискретних класів [30]. В найпростішому випадку бінарної класифікації виконується умова (3.20).

$$P(x) = \begin{cases} 0, & \Delta f(x) \leq 0, \\ 1, & \Delta f(x) > 0, \end{cases} \tag{3.20}$$

де $P(x)$ – прогнозований клас; $\Delta f(x)$ – зміна значення по відношенню до попереднього періоду.

Для збільшення точності прогнозу за класифікаціє можна збільшити кількість класів. Це дає змогу більш гнучко використовувати дані прогнозування.

Для прогнозування за класифікацією використані наступні методи [31-34]:

- логістична регресія;
- ШНМ.

Для оцінки якості прогнозних моделей використовується Інформаційний критерій Байеса (Bayesian information criterion (BIC)), середня абсолютна помилка (Mean Absolute Error (MAE)) та середньоквадратична помилка (Mean Square Error (MSE)) [35 - 37].

Для прогнозування використано модель логістичної регресії для бінарної класифікації. В якості навчальних даних моделі використовувались дані о співвідношенні груп АТС в асортименті аптеки, в якості класифікатора – значення 1, якщо спостерігалося зростання прибутку, 0 – у зворотньому випадку.

Також для прогнозування використано класифікатор на базі ШНМ. В якості ШНМ було реалізовано MLP з двома скритими шарами та одним вихідним нейроном [38].

Для класифікації весь ряд значень приросту прибутку було розбито на 10 рівних класів, які були позначені числами із множини [0;1].

В якості навчальних даних моделі використовувались дані о співвідношенні груп АТС в асортименті аптеки.

Характеристики шарів ШНМ наведені в таблиці 3.2

Таблиця 3.2 – Шари ШНМ для прогнозування руху тренду

Тип шару	Кількість нейронів	Функція активації
Вхідний(перший)	14	ReLU
Скритий(другий)	27	ReLU
Скритий(третій)	20	ReLU
Скритий(четвертий)	10	ReLU
Вихідний(п'ятий)	1	ReLU

Параметри навчання ШНМ наведені у таблиці 3.3.

На рисунку 3.10 відображено графік тренування ШНМ на навчальних даних та представлена залежність абсолютної похибки, середньоквадратичної похибки та збитку від епохи, а також представлено порівняння навчальних та прогнозованих даних.

В таблиці 3.4 наведені порівняльні характеристики моделей прогнозування, розраховані на даних продажів аптеки за період 2013 - 2017 років.

Складність оптимізації у даному випадку полягає в тому, що оптимізувати треба не один параметр, а два (прибуток та оборотність товару), і при цьому забезпечити присутність всіх груп товарів в асортименті аптеки. Якщо прибуток виразити, як P, а довжину складу через L, то у загальному вигляді функція для оптимізації прийме вигляд [39]:

$$f(P,L) = F = \frac{P}{L}, \tag{3.21}$$

де F – фітнес функція.

Таблиця 3.3 – Параметри навчання ШНМ для прогнозування руху тренду

Параметр	Значення
Кількість нейронів у вхідному шарі	14
Кількість нейронів у вихідному шари	1
Оптимізатор	SGD
Метрики	MAE, MSE
Розмір тренувальної вибірки	56
% тестових даних	15%
Розмір батчу	25
Кількість епох	250

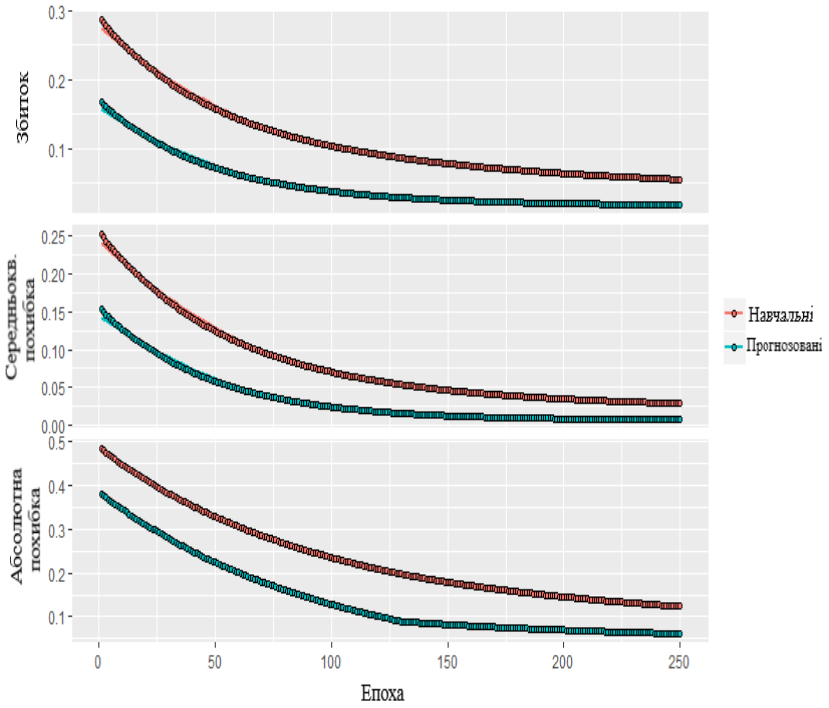


Рисунок 3.10 – Графік навчання ШНМ для прогнозування

Таблиця 3.4 – Якісні характеристики прогнозних моделей

Модель	BIC	MAE	MSE
Лінійна регресія	-102,81	-	-
Поліноміальна регресія	-63,57	-	-
Логістична регресія	94,99	-	-
ШНМ	-	0,024	0,1067

На рисунку 3.11 подано типовий розподіл АТС-груп товарів в розрізі довжини складу (оборотності товару). На цьому рисунку зображено залежність середньомісячного прибутку на одиницю продукції від середньої довжини склада. Можна побачити, що за обсягом аптечних продажів лідирує група АТС406, зокрема має такі показники, як середньомісячний прибуток, який складає 0,60 та середня довжина, що дорівнює 0,4.

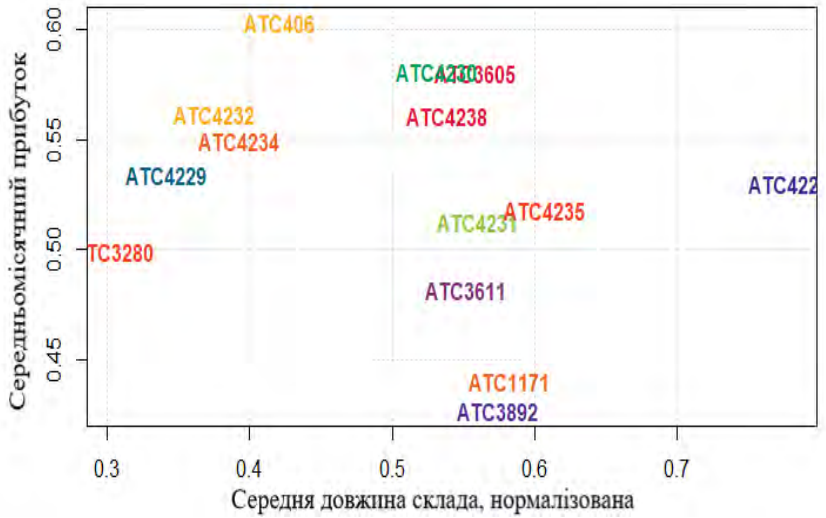


Рисунок 3.11 – Типовий розподіл АТС-груп товарів в розрізі довжини складу (оборотності товару)

При проведенні дослідження ефективності роботи генетичних методів були визначені наступні початкові умови та параметри: максимальна кількість ітерацій=1000, розмір популяції=50, мінімальне значення=0,01, максимальне значення = 0,99, ініціалізація – випадкова, селекція – пропорційна, схрещування – локальне арифметичне, мутація – пропорційна випадкова, ймовірність мутації – 0,2, ймовірність схрещування – 0,8.

Можна припустити, що класичний ГА швидко зійдеться на очевидному рішенні розподілити більший процент на групи, яка має найбільшу прибутковість та невелике значення довжини складу. На рисунку 3.12 зображено процентний розподіл груп товарів класичного ГА. Як можна побачити, ГА розподілив 94 % на групи АТС406 та АТС 4232, а на інші групи всього 6 %.

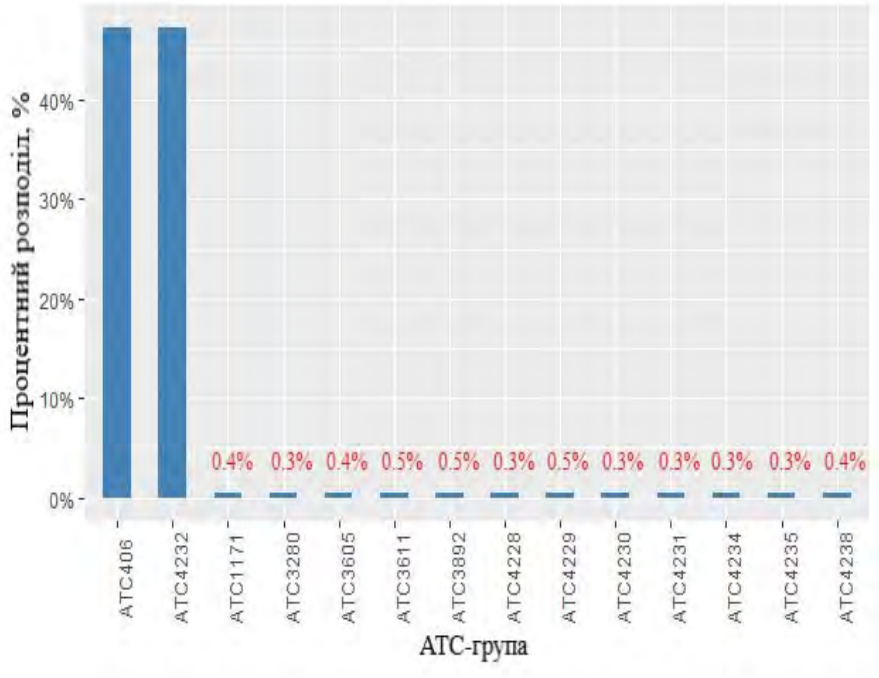


Рисунок 3.12 – Результат роботи класичного ГА

На рисунку 3.13 зображено діаграму зростання фітнес-функції для класичного ГА, на якому видно, що зростання фітнес функції є доволі повільним 1000с та середнє значення фітнес-функції складає 8,0.

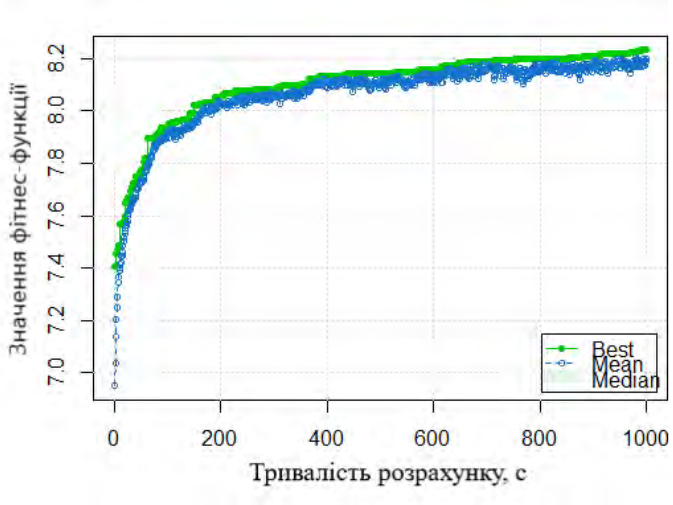


Рисунок 3.13 – Діаграма зростання фітнес-функції для класичного ГА

Але для аптеки такий розподіл є неприйнятним, тому що при цьому сформується незадоволений запит покупців, що призведе до зниження кількості відвідувачів та, як наслідок ,зниження прибутку.

Тому модифікований ГА повинен виконувати додаткову умову – розподіл груп товарів має бути близький до нормального розподілу (функція Гауса) [40].

Для вирішення цього завдання в цільову функцію було введено поняття штрафу алгоритму. Це означає, що якщо оцінювана популяція не відповідає нормальному розподілу, то значення фітнес-функції падає пропорційно. Також були введені вагові коефіцієнти для прибутку і довжини складу, що дозволило гнучко керувати пріоритетом показників [41].

Рівняння 3.21 прийняло вигляд (3.22):

$$f(P,L)=\frac{2P+e^{A}}{e^{L}}, \tag{3.22}$$

де P – очікуваний прибуток; L – довжина складу; A – ступінь близькості розподілення до нормального.

Величина A – це результат тесту Андерсона – Дарлінга, що показує, наскільки розподіл вибірки співпадає з нормальним розподілом. Чим більша ця величина, тим швидше зростає фітнес-функція.

Після коригування фітнес-функції ГА показав наступні результати (рисунки 3.14, 3.15). На рисунку 3.14 зображено процентний розподіл груп товарів класичного ГА з використанням штраф алгоритму, як можна побачити, розподіл долі товарних груп в асортименті став більш варіабельним.

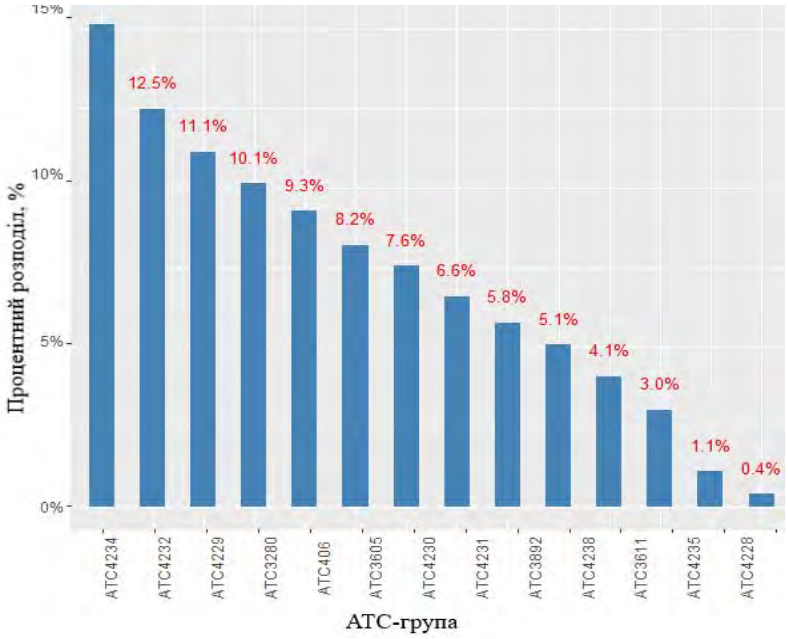


Рисунок 3.14 – Розподіл груп товарів (штраф алгоритму)

На рисунку 3.15 зображено залежність значення фітнес-функції від тривалості розрахунку, на якому видно, що з використанням штраф алгоритму, сам ГА швидше вийшов на точку насичення, але тривалість розрахунку залишилась незмінною 1000 с.

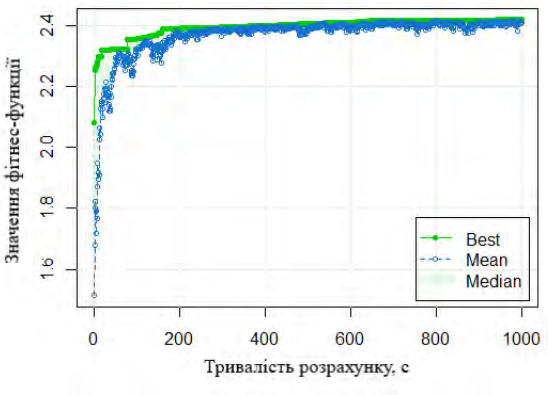


Рисунок 3.15 – Діаграма роботи ГА(штраф алгоритму)

На рисунку 3.16 представлено результати роботи запропонованого методу когнітивно-стильової детермінації, а саме процентний розподіл груп товарів. Як можна побачити, розподіл долі товарних груп в асортименті став більш близький до нормального розподілу (функція Гауса).

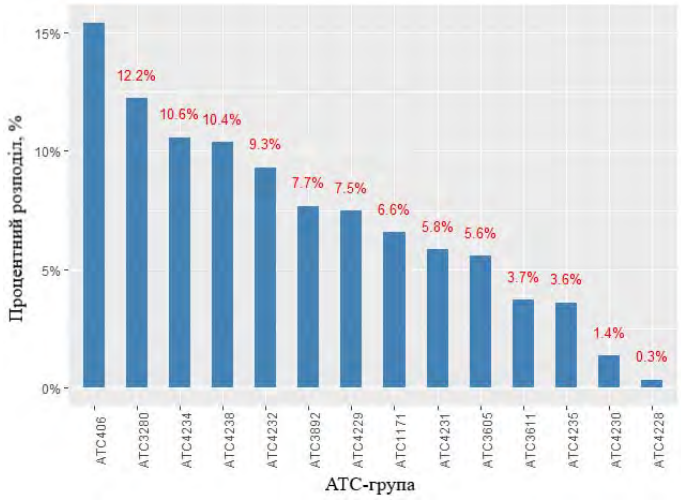


Рисунок 3.16 – Розподіл груп товарів (когнітивно-стильова детермінація)

На рисунку 3.17 представлено результат роботи ГА, що підкоряється закону нормального розподілу. Як можна побачити, нормалізація оператора мутації при деяких умовах позитивно вплинула на швидкість збіжності алгоритму (зокрема, тривалість розрахунку ітерацій при використані когнітивно-стильової детермінації складає 994 с, що значно менше у порівнянні з часом класичного ГА, час виконання якого складає 1024 с).

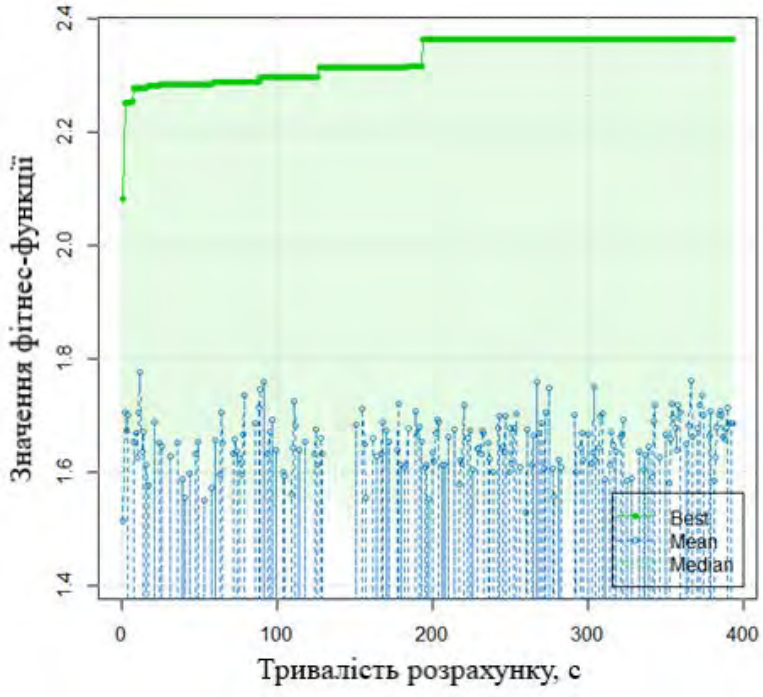


Рисунок 3.17 – Діаграма роботи ГА(когнітивно-стильова детермінація)

Наступна модифікація оператора мутації полягає у внесенні в роботу оператору мутації "інтелекту". Ця модифікація служить для визначення доцільності мутації хромосоми, спираючись на знання ретроспективних та прогнозних даних з використанням у якості прогнозної моделі ШНМ. Результати роботи модифікованого оператору представлені на рисунках 3.18 та 3.19.

Рисунок 3.18 – Розподіл груп товарів (Noetic мутація)

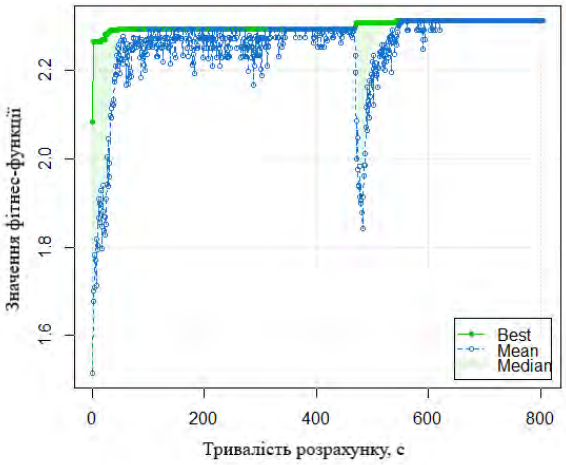


Рисунок 3.19 – Діаграма роботи ГА (Noetic мутація)

На рисунку 3.18 зображено процентний розподіл груп товарів Noetic мутації, на якому можна побачити , що у порівнянні з когнітивно-стильовою детермінацією, розподіл долі товарних груп в

асортименті Noetic мутації не підкорюються закону нормального розподілу. Але при цьому модифікований ГА забезпечує досить прийнятні параметри вихідної множини.

На рисунку 3.19 представлена діаграма роботи ГА(Noetic мутації), за допомогою якого можна побачити, що у порівнянні з когнітивно-стильовою детермінацією, значення фітнес-функції Noetic мутація залишається незмінним, але час синтезу моделі суттєво збільшився з 400 с до 800 с, це пояснюється використанням ШНМ.

Третя модифікація полягає у комбінації двох зазначених модифікацій. Результати роботи модифікованого оператору представлені на рисунках 3.20 та 3.21. На рисунку 3.20 зображено процентний розподіл товару при об'єднанні когнітивно-стильової детермінації та Noetic мутації, на якому видно, що розподіл долі товарних груп в асортименті став більш близький до нормального розподілу у порівнянні з Noetic мутацією, але менш близький, ніж у когнітивно-стильової детермінації.

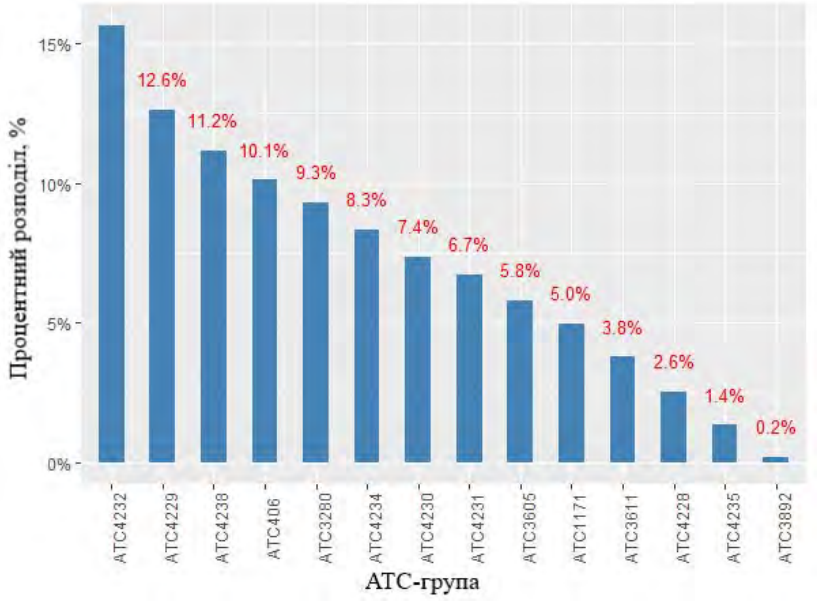


Рисунок 3.20 – Розподіл груп товарів (Merger мутація)

На рисунку 3.21 представлена діаграма роботи ГА(Merger мутації), за допомогою якої можна побачити, що алгоритм Merger мутації найшвидше вийшов на точку насичення у порівнянні з вищезазначеними алгоритмами.

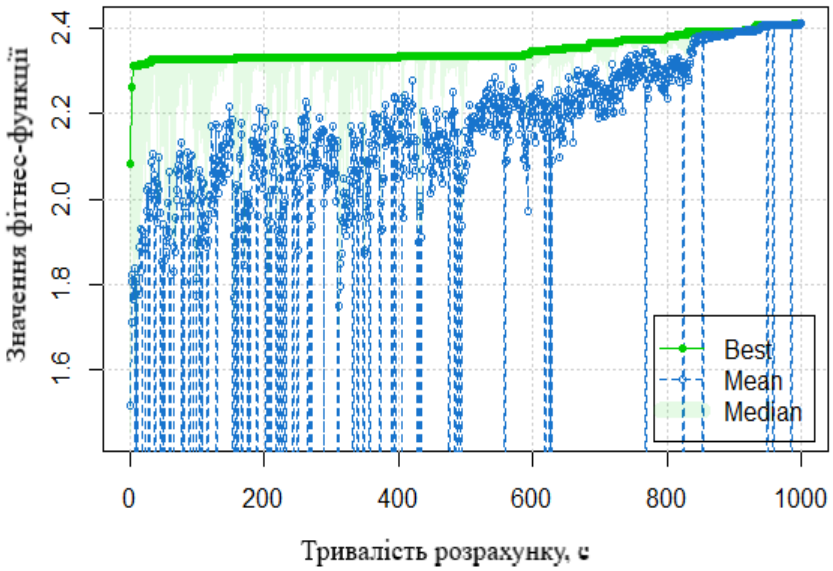


Рисунок 3.21 – Діаграма роботи ГА (Merger мутація)

Оцінка ефективності генетичних алгоритмів проводилася за параметрами: тривалість розрахунку ітерацій, тест Андерсона-Дарлінга, середньоквадратична похибка, очікуваний середній приріст прибутку, середня довжина складу нормалізована. Під тестом Андерсона-Дарлінга [42] вважається критерій, який призначений для перевірки, наскільки розподіл вибірки співпадає з нормальним розподілом. Середньоквадратична похибка є середньоквадратичною відстанню між прогнозованим і фактичним значенням. Критерій прибутковості розраховувався як різниця між рівнем валового доходу торгівлі та рівнем витрат обігу.

В таблиці 3.5 наведено порівняльні дані роботи класичного ГА та модифікованих версій.

Таблиця 3.5 – Порівняльний аналіз модифікацій ГА

Тип моделі	Тривалість розрахунку, ітерацій, с	Тест Андерсона - Дарлінга	Середньо-квадратична похибка	Очікуваний середній приріст прибутку, %	Середня довжина склада нормалізована
Лінійна регресія	875	0,8878315	0,8080	0,50525923	0,4955
Поліноміальна регресія	983	0,8971466	0,998	0,5212312	0,5397
Логістична регресія	787	0,8772425	0,977	0,509569	0,46345
ШНМ	991	0,8878415	0,967	0,5345789	0,39445
Класичний ГА	994	$3{,}67 \times 10^{-7}$	0,9358	0,5217804	0,3964
ГА зі штрафом	974	0,9978415	0,754	0,5407766	0,44987
Когнітивно-стильова детермінація	393	0,9837505	0,782	0,5564371	0,4587188
Noetic мутація	804	0,9986964	0,638	0,5397302	0,4946176
Merger мутація	1024	0,9976742	0,685	0,5438465	0,4544578

Можна побачити, що при використанні Noetic та Merger мутацій збільшується тривалість розрахунку (зокрема, при використанні Merger мутації час синтезу нейромоделі складає 1024 с, при використанні Noetic мутації – 804 с у порівнянні з часом 787 с при використанні логістичної регресії). Таке збільшення зумовлене використанням ШНМ, яка тренована на реальних співвідношеннях, що не підкорюються закону нормального розподілу. Але при цьому ГА забезпечує достатньо оптимальні параметри вихідної множини.

Метод «Merger мутація» показує більш успішні результати в порівнянні з логістичною регресією, а саме за показником тесту Андерсона-Дарлінга (при використанні Merger мутації показник тесту

складає 0,9976742 на відміну від логістичної регресії, у якої показник тесту Андерсона-Дарлінга дорівнює 0,8772425) можна стверджувати, що розподіл вибірки є більш прийнятним до нормального розподілу. Отже фітнес функція Merger мутації зростає швидше в порівнянні з логістичною регресією. А також даний метод забезпечує кращу прибутковість (при Merger мутації прибутковість становить 0,5438465%, а при логістичній регресії 0,509569%), проте вимагає більше обчислювальних та часових витрат (при використанні Merger мутації час синтезу нейромоделі складає 1024 с) з причини покладеної в його основу ідеї об'єднання Когнітивно-стильової детермінації та Noetic мутації.

При порівнянні Noetic мутації з класичним ГА можна зробити висновок, що Noetic краще за показниками тривалості розрахунку (зокрема, при використанні Noetic мутації час синтезу нейромоделі складає 804 с у порівнянні з часом 994 с при використанні класичного алгоритму) та середнім приростом прибутку (показник прибутку у Noetic мутації складає 0,5397302%, а у класичного ГА становить 0,5217804%), але поступається за показниками довжини складу (зокрема, при використанні Noetic мутації складає 0,4946176, а у класичного ГА становить 0,3964). Порівнюючи Noetic мутацію з ГА зі штрафом, що є також розвитком класичного ГА, можна сказати, що дані алгоритми рівноправні за ефективністю.

Найкращі результати показала когнітивно-стильова детермінація, тобто вона дає виграш в тривалості розрахунку, що становить 393 с та середньому прирості прибутку, який складає 0,5564371%, що набагато більше своїх аналогів.

Отримані результати дозволяють зробити висновок про те, що запропонований підхід до вирішення завдань оптимізації фінансових показників мережевих аптек дає змогу збільшити середній приріст прибутку аптеки на 0,53% та мінімізувати довжину складу до 0,45 з урахуванням середньо-квадратичної похибки 0,685.

Таким чином, в роботі запропоновано і обґрунтовано новий підхід до вирішення завдання оптимізації процесу роботи відділу закупівлі ліків, основна ідея якого полягає у використанні модифікованого генетичного методу для оптимізації параметрів моделі з контролем математичного розподілення значень вихідної хромосоми, з метою підвищення ефективності (стійкості) ГА як еволюціонуючої системи.

Розроблений алгоритм по ефективності вирішення задачі оптимізації на множині тестових даних перевершує методи регресії та класичний ГА.

3.6 Висновки за розділом 3

Було проаналізовано дані продаж, залишків та прибутковості мережі аптек, кількісні співвідношення груп товарів в асортименті аптек, закономірності розвитку та сезонності продаж, закономірності формування асортиментної корзини товарів.

Було досліджено та побудовано певні математичні моделі прогнозування та оптимізації показників прибутковості та оптимальності формування асортименту .

Було проаналізовано типову аптеку мережі та досліджена пристосованість побудованих математичних моделей до реальних умов продаж та формування асортименту.

В ході дослідження виявлено, що оптимальним методом формування асортиментної корзини аптеки є використання генетичного алгоритму з модифікованим оператором мутації.

Розроблені нові генетичні оператори мутації, що дозволяють враховувати в процесі еволюційного пошуку фактори математичного розподілення значень генів у хромосомі та керувати процесом мутації, коригуючи доцільність мутації в кожному конкретному випадку. Для цього в роботу оператору мутації було включено запит до штучної нейронної мережі прямого розповсюдження.

Розроблено програмну систему виконання розрахунків і оптимізації мовою програмування R з використанням бібліотек GA для реалізації алгоритму еволюційного пошуку та Keras для реалізації штучної нейронної мережі.

Також до програмного пакету входить Windows-додаток, розроблений мовою програмування C#. Додаток надає дружній інтерфейс користувача для виконання розрахунків та відображення результатів та рекомендацій щодо формування асортиментної корзини.

Також було проаналізовано економічний аспект проєкту і обґрунтовано доцільність впровадження розробленого програмного забезпечення.

3.7 Література до розділу 3

1. Haşim Sak Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition [Electronic resource].– Access mode: https://www.arxiv-vanity.com/papers/1402.1128/.
2. Cornell University Library. [Electronic resource]. – On some extensions to GA package, 2017. – Access mode: https://arxiv.org/pdf/1605.01931.pdf.
3. Swingler, Kevin Applying Neural Networks: A Practical Guide / Keven Swingler. – Published by Morgan Kaufmann, 2001. – 301 p.
4. Neural-Network-Biased Genetic Algorithms for Materials Design: Evolutionary Algorithms That Learn / [T.K. Patra, V. Meenakshisundaram, J.H. Hung, D.S. Simmons] // ACS Combinatorial Science. – 2017. – Vol. 19, issue 2. – P. 96 – 107. Doi: 10.1021/acscombsci.6b00136.
5. A multiobjective genetic algorithm for the localization of optimal and nearly optimal solutions which are potentially useful: nevMOGA [A. Pajares, X. Blasco, J.M. Herrero, G. Reynoso-Meza] // Hindawi. Complexity. – 2018. – Vol. 11 – P. 1 – 22. Doi: 10.1155/2018/1792420.
6. Using Evolutionary Algorithms and Machine Learning to Explore Sequence Space for the Discovery of Antimicrobial Peptides / [Mari Yoshida, Trevor Hinkley, SoichiroTsuda et al.] // Chem. – 2018. – Vol. 4, Issue 3. – P. 533-543. doi:10.1016/j.chempr.2018.01.005.
7. Bourlard, H. Auto-Association by Multilayer Perceptrons and Singular Value Decomposition / H. Bourlard and Y. Kamp // Biologycal Cybernetics, 1988. – 3 p.
8. Baker J. Reducing Bias and Inefficieny in the Selection Algorithm. Genetic Algorithms and Their Applications / Lawrence Erlbaum – Inc. Hillsdale, 1987. – 7 p.
9. Schoenauer M. Divide-and-Evolve: a Sequential Hybridization Strategy Using Evolutionary Algorithms / M. Schoenauer, P.Savéant, V. Vidal // Advances in Metaheuristics for Hard Optimization. Natural Computing Series. – Berlin: Springer, 2007. – Section 9. – P. 179-198 doi: 10.1007/978-3-540-72960-0_9
10. Vlasic I. Improving genetic algorithm performance by population initialisation with dispatching rules / I. Vlasic, M. Durasevic, D. Jakobovic // Computers & Industrial Engineering. – 2019. – Vol. 137. – P. 106030. doi: 10.1016/j.cie.2019.106030

11. Gen, M. Genetic algorithms and engineering design / M. Gen, R. Cheng. – New Jersey : John Wiley & Sons, 1997. – 352 p.

12. Haupt, R.L. Practical genetic algorithms / R. L. Haupt, S. E. Haupt. – New Jersey : John Wiley & Sons. Inc, 2004. – 261 p.

13. Willach Pharmacy Solutions [Electronic resource]. – Access mode : https://www.willach-pharmacy-solutions.com/EN/index.php

14. Stepanenko O. Development of the method for decomposition of superpositions of unknown pulsed signals using the secondorder adaptive spectral analysis / O. Stepanenko, A. Oliinyk, L. Deineha, T. Zaiko // Eastern-European Journal of Enterprise Technologies. – 2018. – Vol. 92, Issue 2/9. – P. 48–54. DOI: 10.15587/1729-4061.2018.126578.

15. Oliinyk, A. O. Parallel Method of Production Rules Extraction Based on Computational Intelligence / A. Oliinyk, S. Skrupsky, S. Subbotin, I. Korobiichuk // Automatic Control and Computer Sciences. – 2017. – Vol. 51, Issue 4. – P. 215–223. DOI: 10.3103/S0146411617040058

16. Oliinyk A. A Evolutionary method for solving the traveling salesman problem / A. Oliinyk, I. Fedorchenko, A. Stepanenko, M. Rud, D. Goncharenko // Problems of Infocommunications. Science and Technology: 5th International Scientific-Practical Conference PICST2018, Kharkiv, 9–12 October 2018 : proceedings. – Kharkiv: Kharkiv National University of Radioelectronics, 2018 – P. 331 – 339.

17. Alsayaydeh, J. A. Stratified Model of the Internet of Things Infrastructure / J. A. Alsayaydeh, V. Shkarupylo, M. S. Hamid, S. Skrupsky, A. Oliinyk // Journal of Engineering and Applied Sciences. – 2018. – Vol. 13, Issue 20. – P. 8634-8638. DOI: 10.3923/jeasci.2018.8634.8638.

18. Dopico, J. Encyclopedia of artificial intelligence / Eds.: J. R. Dopico, J. D. de la Calle, A. P. Sierra. – New York : Information Science Reference, 2009. – Vol. 1-3. – 1677 p.

19. Nagata Y. A Powerful Genetic Algorithm Using Edge Assembly Crossover for the Traveling Salesman Problem / Y. Nagata, S. Kobayashi. – 2012. – P. 346-363. doi: 10.1287/ijoc.1120.0506

20. Hoffman, K. Traveling Salesman Problem / K. Hoffman, G. Rinaldi // Advertising Response, Encyclopedia of Operations Reseach. — 2013. – P. 1573-1578. doi: 10.1007/978-1-4419-1153-7_1068

21. Wang, Y. The hybrid genetic algorithm with two local optimization strategies for traveling salesman problem // Computers & Industrial

Engineering. – 2014. – Vol. 70. – P. 124-133. doi: 10.1016/j.cie.2014.01.015

22. Sanches D. Improving an exact solver for the traveling salesman problem using partition crossover / D. Sanches, D. Whitley // Proceedings of the Genetic and Evolutionary Computation Conference. – 2017. – P. 337-344. doi: 10.1145/3071178.3071304

23. Hussain, A. Genetic Algorithm for Traveling Salesman Problem with Modified Cycle Crossover Operator / A. Hussain, Y. Muhammad // Computational Intelligence and Neuroscience. – 2017. – P. 7. doi: 10.1155/2017/7430125

24. Tsai, C. A High-Performance Genetic Algorithm: Using Traveling Salesman Problem as a Case. / C. Tsai, S. Tseng // The Scientific World Journal. – 2014. – P. 14. doi: 10.1155/2014/178621

25. Fedorchenko I. Development of the modified methods to train a neural network to solve the task on recognition of road users / I. Fedorchenko, A. Oliinyk, A. Stepanenko, T. Zaiko, S. Shylo, A. Svyrydenko // EasternEuropean Journal of Enterprise Technologies. – 2019. – Vol. 2, Issue 9/98. – P. 46–55.DOI: 10.15587/1729-4061.2019.164789

26. Shkarupylo V. Development of stratified approach to software defined networks simulation / V. Shkarupylo, S. Skrupsky, A. Oliinyk, T. Kolpakova // Eastern-European Journal of Enterprise Technologies. – 2017. – Vol. 89, Issue 5/9. – P. 67–73.DOI: 10.15587/1729-4061.2017.110142.

27. Kolpakova T. Improved method of group decision making in expert systems based on competitive agents selection / T. Kolpakova A. Oliinyk, V. Lovkin // 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON), Kyiv, May 29 – June 2, 2017 : conference proceedings. – Kyiv: Institute of Electrical and Electronics Engineers, 2017. – P. 939–943. DOI: 10.1109/UKRCON.2017.8100388

28. Sarac V. Calculation of Electromagnetic Fields in Electrical Machines using Finite Elements Method [Electronic resource]. – Access mode: https://core.ac.uk/download/pdf/35335797.pdf.

29. Winston D. Novel Energy Conservation Scheme for Three Phase Induction Motor Drives Employed in Constant Speed Applications [Electronic resource]. – Access mode: https://pdfs.semanticscholar.org/dd39/ce6fc7f0abd2341849bc807f9bd15562b305.pdf.

30. Oliinyk A. Factor analysis of transaction data bases / A. Oliinyk, T. Zaiko, S. Subbotin // Automatic Control and Computer Sciences. – 2014. – vol.48, no.2 – pp. 87-96.

31. Oliinyk A. A stochastic approach for association rule extraction / A. Oliinyk, S. Subbotin // Pattern Recognition and Image Analysis. – 2016. – vol.26, no.2. – pp. 419-426.

32. Oliinyk A. Synthesis of Neuro-Fuzzy Networks on the Basis of Association Rules / A. Oliinyk, T. Zayko, S. Subbotin // Cybernetics and Systems Analysis. – 2014. – vol.50, no.3. – pp. 348-357.

33. Goldberg, D. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley, 1989. – 432 p.

34. Holland, J. Adaptation in Natural and Artificial Systems. Ann Arbor: University of Michigan Press, 1975., – 232 p.

35. Chambers L. The practical handbook of genetic algorithms / ed. L. D. Chambers. – Florida : CRC Press, 2000. – Vol. I: Applications. – 520 p.

36. Chambers L. The practical handbook of genetic algorithms / ed. L. D. Chambers. – Florida : CRC Press, 2000. – Vol. II: New frontiers. – 421 p.

37. Chambers L. The practical handbook of genetic algorithms. / ed. L. D. Chambers. – Florida: CRC Press LLC, 2000. – Vol. III: Complex coding systems. – 659 p.

38. Cantu-Paz E. Efficient and accurate parallel genetic algorithms / E. Cantu-Paz. – Massachusetts: Kluwer Academic Publishers, 2001. – 162 p.

39. Bao Lin Xiaoyan Sun, Salous S. Solving Travelling Salesman Problem with an Improved Hybrid Genetic Algorithm / Bao Lin Xiaoyan Sun, S. Salous // Journal of Computer and Communications. – 2016. – № 4. – P. 98-06/

40. Adibi M. Single and multiple outputs decision tree classification using bi-level discrete-continues genetic algorithm / M. Adibi // Pattern Recognition Letters. – 2019. – Vol. 128. – P. 190-196. doi: 10.1016/j.patrec.2019.09.001

41. Lapan M. Deep Reinforcement Learning Hands-On: Apply modern RL methods, with deep Q-networks, value iteration, policy gradients, TRPO, AlphaGo Zero and more / M Lapan. – Packt Publishing. – 2018. –P. 546.

42. Baloch G. Capacity and assortment planning under one-way supplier-driven substitution for pharmacy kiosks with low drug demand / G. Baloch, F. Gzara // European Journal of Operational Research. – 2019. doi: 10.1016/j.ejor.2019.09.007

РОЗДІЛ 4
МОДЕЛЮВАННЯ ЗАЛЕЖНОСТІ ПОКАЗНИКІВ ЗДОРОВ'Я ЛЮДЕЙ ВІД ОБСЯГІВ ВИКИДІВ ЗАБРУДНЮЮЧИХ РЕЧОВИН У ПОВІТРЯ

Розв'язання завдань медичного діагностування пов'язано з необхідністю побудови моделей залежності показників здоров'я пацієнтів від різних факторів. До таких факторів в багатьох випадків можуть відноситися кількість викидів забруднюючих речовин в районі проживання конкретних пацієнтів.

У даному розділі виконано дослідження залежності показників захворюваності населення хворобами системи кровообігу, туберкульозом та онкологічними хворобами від обсягів викидів забруднюючих речовин в атмосферне повітря в результаті діяльності стаціонарних джерел забруднення в різних регіонах України, а також розробка програмного пакету для дослідження залежності показників захворюваності від обсягів викидів забруднюючих речовин. Це забезпечить можливість прогнозування показників захворюваності, що може бути використано при плануванні регіональних бюджетів в розділі витрат на охорону здоров'я, медикаментозне забезпечення населення, здійснення заходів щодо покращення екологічного становища.

Проблематика досліджуваної предметної області є досить актуальною і привертає багато уваги як з боку світової наукової спільноти у галузі охорони здоров'я, так і з боку дослідників у сфері Data Science. Згідно Статуту Всесвітньої Організації охорони здоров'я, мати найвищий досяжний рівень здоров'я є одним з основних прав будь-якої людини, а охорона навколишнього середовища (зокрема, забруднення повітря і води) є одним з 70 основних напрямків діяльності ООН [1].

Водночас, побудова моделі такої залежності не є простою задачею, адже рівень забруднення не є єдиним фактором, що впливає на рівень захворюваності. Складність цього завдання обумовлена також тим, що залежність захворюваності від обсягів викидів не є лінійною. Тому таке моделювання потребує використання сучасних методів

обчислювального інтелекту, зокрема штучних нейронних мереж, генетичних алгоритмів тощо.

У цьому розділі наведено аналіз динаміки обсягів викидів забруднюючих речовин і діоксиду вуглецю в атмосферне повітря від стаціонарних джерел забруднення, а також дослідження динаміки показників захворюваності по такими хворобам, як туберкульоз, хвороби органів кровообігу і онкологічні хвороби в різних регіонах України за останні роки. Розглянуто відомі математичні моделі, які використовуються для аналізу залежності (зокрема моделі, що в якості базису використовують логістичну регресію, метод опорних векторів, метод найменших квадратів, випадковий ліс, метод найближчого сусіда), а також моделювання з застосуванням таких методів і засобів, як штучні нейронні мережі, генетичні алгоритми, мультиагентні системи (алгоритм рою часток) і їх комбіновані варіанти. Побудовано моделі залежності показників захворюваності від обсягів викидів забруднюючих речовин, а також створено програмну реалізацію алгоритму, що дозволяє прогнозувати зазначені вище показники захворюваності в залежності від обсягів викидів.

4.1 Моделювання процесу забруднення атмосферного повітря

4.1.1 Види та джерела забруднення повітря

Для побудови теплової мапи використано дані з офіційного сайту Державної служби статистики України [1] (Статистична інформація / Багатогалузева статистична інформація / Регіональна статистика / Навколишнє природне середовище / Викиди забруднюючих речовин в атмосферне повітря / Викиди забруднюючих речовин в атмосферне повітря від стаціонарних джерел забруднення за регіонами (1990-2017)).

Зображення побудоване за допомогою API Google Map [2]

Оскільки, в офіційній статистиці наведені дані по областях України, центрами розповсюдження градієнту на тепловій мапі є приблизні географічні центри областей.

Загальна мапа забрудненням атмосферного повітря України зображена на рисунку 4.1 [3-9].

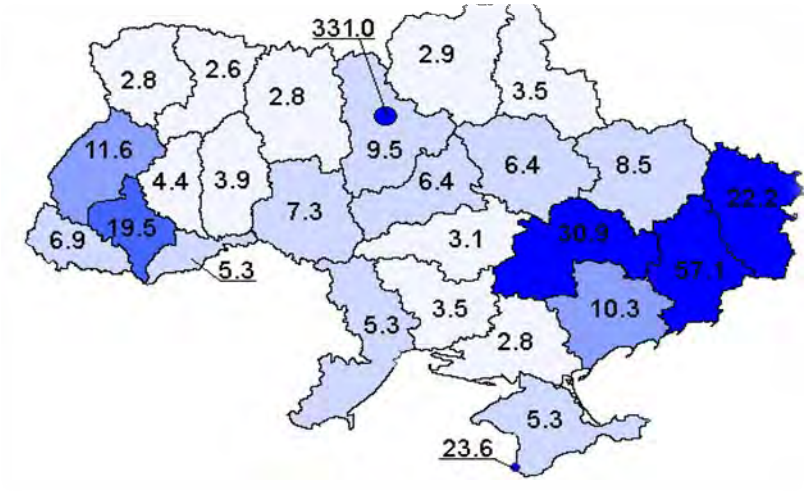


Рисунок 4.1 - Викиди шкідливих речовин в атмосферне повітря (тонн на 1м^2) [3–9]

Враховуючи постійне зростання кількість промислових підприємств (які є головними джерелами викидів різноманітних домішок в атмосферне повітря), на сьогодні, проблема забруднення атмосферного повітря стоїть дуже гостро.

Існує декілька класифікацій забруднюючих речовин [10]. Зокрема, за джерелом утворення, атмосферні домішки можна поділити природні та антропогенного походження.

До природних джерел атмосферного забруднення відносять ті, що пов'язані з різноманітними метеорологічними явищами: пилові бурі, виверження вулканів, космічний пил, тощо. Зазвичай, склад суміші речовин природного забруднення атмосфери на три четверті складається з речовин неорганічного походження – частинок ґрунту, продуктів вивітрювання гірських порід, попелу, неорганічних солей, тощо.

В атмосферному повітрі присутні також і органічні речовини, які є продуктами життєдіяльності організмів. Це вуглеводи, органічні кислоти, ефіри, альдегіди, низькомолекулярні органічні сполуки, тощо. Багато організмів використовують їх для життєвих потреб. Органічні сполуки, що мають згубну дію на найпростіші організми,

бактерії, гриби,називають фітонцидами. Встановлено, що щорік до атмосферного повітря надходить приблизно від 0,7 до 1,5 мільярдів тон морських солей і близько 700 мільйонів тон ґрунтового пилу. Внаслідок лісових пожеж утворюються суміші твердих часток продуктів згоряння і повітря – аерозоль. По різним оцінкам щорічно утворюється близько 35 – 360 мільйонів тон аерозолів, а сумарне надходження аерозолів природного походження від усіх джерел в атмосферне повітря складає до 2,3 мільярдів тон. Зазвичай, якщо природні джерела забруднення не спричинюють істотних змін атмосферного повітря(якщо не перевищуються граничнo-припустимих значень). У той же час, інтенсивне поширення природного джерела забруднення на певній території (виверження вулканів, що супроводжуються викидами вулканічного попелу і газів, масивні лісові і степові пожежі, тощо) може спричинити серйозне забруднення атмосферного повітря на достатньо великій території. У деяких випадках, такі явища навіть можуть бути причиною утворення світлонепроникного екрана навколо Землі і, як наслідок,зміни теплового балансу. Проте, слід зауважити, що зазвичай, природні джерела забруднення атмосферного повітря здебільшого не завдають великої шкоди людині, бо відбуваються за певними біологічними законами, виявляються періодично, достатньо легко прогнозуються та не мають глобального характеру.

Штучне, або антропогенне, забруднення атмосферного повітря, ще забруднення, що пов’язане з діяльністю людини, і спричиняє зміни складу і властивостей атмосферного повітря. Джерела антропогенного забруднення атмосферного повітря можна поділити на стаціонарні і пересувні.

За походженням забруднення атмосферного повітря також можна поділити на фізичні і хімічні. До фізичних забруднень можна віднести пилове забруднення, або механічне, радіоактивне (випромінювання або стійки радіоактивні ізотопи), електромагнітне, шумове і теплове.

Всі антропогенні джерела забруднення атмосферного повітря можна поділити на дві великі групи – стаціонарні джерела і пересувні джерела забруднення.

До стаціонарних джерел забруднення атмосферного повітря можна віднести – підприємство, цех, агрегат, установка або інший нерухомий об’єкт, що здійснює викиди забруднюючих речовин в

атмосферне повітря і зберігає своє місце знаходження протягом певного часу. Стаціонарні джерела забруднення атмосферного повітря, в свою чергу, можна поділити на організовані й неорганізовані [11, 12].

Організовані джерела – це джерела, які мають спеціальні технологічні спорудження, через які здійснюють викиди забруднюючих речовин (труби, газоходи, тощо).

Неорганізовані джерела викидів забруднюючих речовин в атмосферне повітря утворюється, зазвичай раптово, непередбачувано, наприклад, унаслідок порушення герметичності устаткування, порушення певних технологічних процесів, відсутності або незадовільної роботи очищувального обладнання, у місцях зберігання, завантаження або розвантаження продукту. Також, до неорганізованих джерел можна віднести автомобільні стоянки, склади паливно-мастильних матеріалів або інших летючих речовин, тощо.

Стаціонарні джерела забруднення атмосферного повітря. Склад викидів в атмосферне повітря від стаціонарних джерел забруднення, насамперед, залежить від речовин та технологій, які використовує те чи інше джерело забруднення. Так, наприклад, теплові електростанції забруднюють атмосферне повітря викидами, що містять сірчистий ангідрид (SO_2),який у суміші із водою утворює сірчисту кислоту, окисли азоту (NO_n), сажу, смолисті речовини, пил, солі важких металів [10].

Для виробництв, які відносяться до галузі чорної металургії, мають доменне, сталеплавильне та прокатне виробництва, а також гірничорудних цехів, заводів коксохімічної промисловості, теплоенергетичних установок характерними у складі викидів є діоксид вуглецю (CO_2), сірчистий ангідрид (SO_2),сірчаний ангідрид (SO_3), пил, окисли азоту різного ступеню окислення (NO_n), сірководень (H_2S), аміак (NH_3), сірковуглець (CS_2), аерозолі хрому (Cr) і марганцю (Mn), бензол (C_6H_6), феноли та інші гетероциклічні органічні сполуки.

Для підприємств кольорової металургії притаманні викиди в атмосферне повітря насичені такими речовинами, як сполуки кольорових і важких металів, фтору, пари ртуті,окисли азоту різного ступеню окислення (NO_n), сірчистий ангідрид (SO_2), окисли вуглецю, смолисті речовини, поліметалічний пил, вуглеводні (та інші

низькомолекулярні органічні сполуки). Однією з найнебезпечніших для здоров'я людини речовин, яка має канцерогену дію і міститься у викидах підприємств кольорової металургії, є бензпірен.

Схожий склад викидів в атмосферне повітря мають підприємства металообробки та машинобудівні підприємства. Їх викиди містять аерозолі сполук кольорових і важких металів(зокрема, парів ртуті).

Такі речовини, як сірководень (H_2S), сірчистий та сірчаний ангідриди (SO_2 і SO_3 відповідно), різні окисли вуглецю, аміак (NH_3), низькомолекулярні та гетероциклічні вуглеводні (зокрема, бензпірен) містяться у викидах підприємств нафтохімічної і нафтопереробної промисловості.

Хімічна промисловість збагачує атмосферне повітря такими небезпечними речовинами, як окисли сірки (зокрема, сірчаний і сірчистий ангідриди), хлор, окисли азоту, аміак (NH_3), сірководень (H_2S), з'єднання фосфору, високомолекулярні органічні сполуки, хлороводень (HCl), сполуки важких металів, пил і сажа.

Виробництво оздоблювальних і будівельних матеріалів пов'язане з викидами в атмосферне повітря пилу, двоокису кремнію (SiO_2), гіпсового та азбестового пилу, фтору, сполук важких металів.

Пересувні джерела забруднення атмосферного повітря.

Головним джерелом забруднення атмосферного повітря серед пересувних джерел є транспорт. Взагалі, частка, яка приходиться на транспорт серед як стаціонарних, так і пересувних джерел, в Україні складає коло тридцяти восьми відсотків. За видами транспорту – автомобільний, залізничний, повітряний і водний, розподіл кількості викидів забруднюючих речовин в атмосферне повітря виглядає наступним чином. Найбільша частка, п'ятдесят вісім відсотків, припадає на автомобільний транспорт. На залізничний транспорт припадає двадцять п'ять відсотків. Найменш атмосферне повітря забруднюють повітряний транспорт і річковий та морський транспорт – два відсотки і один відсоток, відповідно. Ще чотирнадцять відсотків припадає на дорожньо-будівельний комплекс.

Насамперед, це забруднення продуктами спалювання палива в двигунах внутрішнього згоряння, виділення тепла, шумове забруднення. Якісний склад викидів залежить від виду та якості паливних матеріалів, які використовуються в двигунах, конструктивних особливостей двигунів, їх темничного стану.

При повному згорянні палива (за умов належного стану двигуна), зазвичай, виділяється вуглекислий газ, вода (пара) і двоокис сірки. При недостатньому надходженні кисню вуглеводні, що складають основну частку палива, окислюються неповністю. Як наслідок, утворюється чадний газ (CO), який є дуже токсичним і, навіть у невеликій концентрації в атмосферному повітрі, здатен викликати тяжкі отруєння.

Взагалі, відпрацьовані гази містять близько двохсот компонентів, період розпаду яких у природних умовах складає від декількох хвилин, до декількох років.

4.1.2 Показники впливу рівня забруднення атмосферного повітря на здоров'я людини

Усі забруднюючі атмосферне повітря речовини в більшому чи меншому ступені впливають на здоров'я людини. Ці речовини потрапляють в організм людини переважно через систему дихання, шкіру та слизові оболонки. Органи дихання безпосередньо страждають від забруднення. Домішки часток розміром 0,01 – 0,1 мкм здатні приникати у легені з повітрям, яке вдихає людина, і, приблизно, п'ятдесят відсотків цих часток здатні в них осідати. Токсичний ефект домішок може бути обумовлений такими чинниками:

– речовини є токсичними по своїй хімічній чи фізичній природі;

– речовини можуть перешкоджати фізіологічним процесам очищення респіраторного тракту;

– речовини можуть абсорбувати інші отруйні речовини і попадаючи в легені вивільняти їх.

У деяких випадках, забруднюючи речовини, надходячи до організму людини, можуть потенціювати токсичну дію інших речовин. Таким чином, комбінована токсична дія декількох таких речовин перебільшую їх можливу сумарну дію. До уваги також слід приймати і тривалість впливу отруйної речовини, її експозиційну кількість.

Статистичні дослідження дозволять напевно стверджувати про наявність залежності між рівнем забруднення атмосферного повітря і рівнем захворюваності на такі хвороби, як захворювання органів дихання, хвороби системи кровообігу, хвороби ока.

Значне підвищення концентрації домішок забруднюючих речовин, що зберігається впродовж декількох днів, збільшує смертність людей літнього віку від хвороб органів дихання і захворювань системи кровообігу.

Відомо, що серед професійних захворювань, наприклад, людей, які працюють с азбестом, є ракові захворювання бронхів і діафрагм (мембрана, що розділяє грудну і черевну порожнини). Тривалий контакт з берилієм в наслідок свого шкідливого впливу підвищує імовірність виникнення онкологічних захворювань дихальних шляхів, шкіри і очей.

Надмірно токсичними є пари ртуті, які викликають порушення роботи центральної нервової системи і нирок, відомі як «синдром ртутної нирки». Також, слід враховувати, що ртуть може накопичуватися в організмі люди і наслідки отруєння також можуть бути обумовленні кумулятивним ефектом.

Особливе значення має постійне зростання обсягів викидів забруднюючих речовин в містах. Як наслідок, у більшості розвинених країн світу і в Україні спостерігається стабільне зростання чисельності хворих на хронічний бронхіт, емфізема легень, алергійні і онкологічні захворювання.

При систематичному чи періодичному надходженні в організм порівняно невеликих кількостей токсичних речовин відбувається хронічне отруєння. Ознаками хронічного отруєння є порушення нормального поводження, звичок, а також нейропсихічні відхилення: швидке стомлення чи почуття постійної втоми, сонливість, чи навпаки, безсоння, апатія, ослаблення уваги, неуважність, безпам'ятність, сильні коливання настрою.

Отруєння шкідливими речовинами повітря може бути гострим чи хронічним. Якщо концентрація такої речовини в повітрі досить висока, а час впливу не великий, то, зазвичай, спостерігається гостре отруєння. При тривалому впливі отруйних (токсичних) речовин може виникнути хронічне отруєння.

Перебіг хронічного отруєння залежить від індивідуальних властивостей організму людини. Так одні і ті ж самі токсини у різних людей можуть згодом викликати різні хвороби. Подібні ознаки спостерігаються і при радіоактивному забрудненні навколишнього середовища.

Відомо, що у районах, які постраждали від радіоактивного забруднення в результаті Чорнобильської катастрофи, захворюваність серед населення, особливо дітей, збільшилася у багато разів.

Високоактивні в біологічному відношенні хімічні сполуки можуть викликати ефект віддаленого впливу на здоров'я людини: хронічні запальні захворювання різних органів, зміну нервової системи, дію на внутрішньоутробний розвиток плоду, що призводить до різних відхилень у немовлят.

Наразі встановлено пряму залежність між збільшенням числа людей, що хворіють на алергію, бронхіальну астму, онкологічні хвороби, і погіршенням екологічної обстановки у певному регіоні. Доведено канцерогенну дію таких складових викидів, як хром, нікель, берилій, азбест.

Якщо ще в першій половині двадцятого століття рак в педіатричній практиці був дуже рідкісним явищем, то зараз він зустрічається все частіше й частіше. Крім цього, з'являються нові, невідомі раніше хвороби, причини яких буває дуже важко встановити.

4.1.3 Аналіз рівню та динаміки викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел по регіонам України

Взагалі, майже по всіх регіонах України в динаміці обсягів викидів забруднюючих речовин в атмосферне повітря спостерігається майже однаковий тренд. Візуальне відображення динаміки рівнів захворюваності хвороб по регіонах України [13, 14] представлена у вигляді діаграм, наведене на рис. 4.2 та рис. 4.3.

Аналізуючи наведену статистику [13] можна спостерігати доволі різке зменшення рівня викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря по всіх регіонах починаючи з 1990 року (початок спостережень) і до приблизно 1996 – 1997 років. Вочевидь, це пов'язано з загальним спадом виробництва в різних галузях промисловості України в дев'яностих роках. Як відомо, спад виробництва в Україні в дев'яностих роках був пов'язаний з глибокою економічною кризою. Відомо, що З 1990 року по 1994 рік валовий національний продукт скоротився на 44%, обсяг промислової продукції – на 41%, національний доход - на 54%.

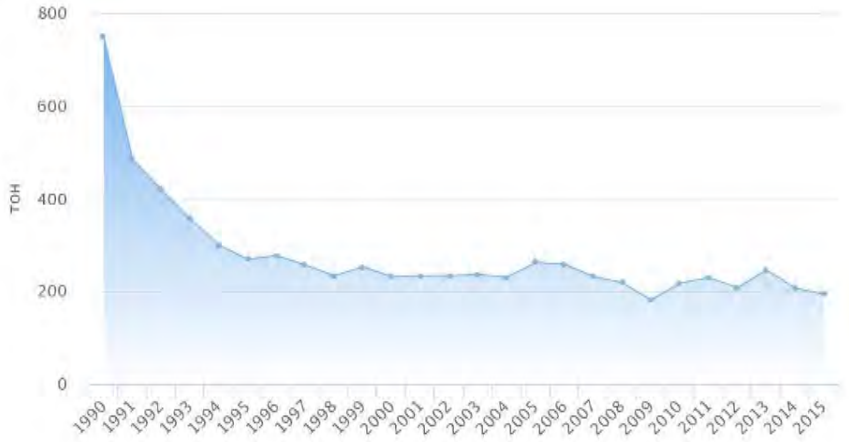


Рисунок 4.2 - Викиди забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел забруднення по Запорізькій області

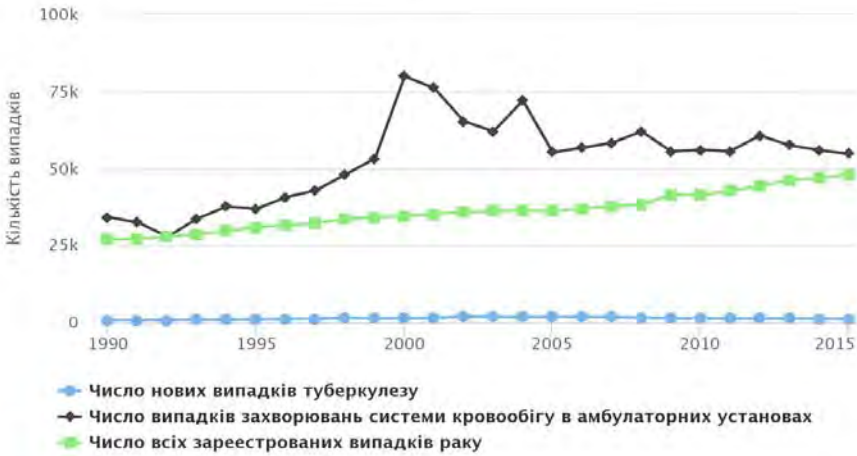


Рисунок 4.3 – Динаміка рівню захворюваності по Запорізькій області

Починаючи з двотисячних років можна спостерігати підвищення, а в деяких регіонах (наприклад, Волинська область, Закарпатська область, Івано-Франківська область та ін.) доволі різке підвищення рівня викидів забруднюючих речовин в атмосферне повітря. На цей період припадає початок економічного зростання в Україні,

нарощування темпів виробництва, поступове збільшення національного валового продукту. Загалом в теперішній час можна спостерігати тенденцію збільшення обсягу викидів у більшості регіонів України. У деяких регіонах рівень викидів залишається стабільним і коливається в межах показників набутих в двотисячних роках.

За даними Державної служби статистики, обсяги викидів за останні 28 років суттєво зменшилися. Якщо у 1992 році Україна входила до 7-ки найбільших країн-забруднювачів у світі, викидаючи в атмосферу 15,5 млн тон забруднюючих речовин, то у 2017 році їхня кількість зменшилася до 2,58 млн тон. Серед областей найбільшими забруднювачами довкілля є Кривий Ріг (30,4%), Дніпропетровська (25,4%) та Івано-Франківська (7,7%) області. Перші – за рахунок концентрації енергетичних та металургійних підприємств, а Франківщина – через наявність Бурштинського енергетичного острову. Найменше викидів зафіксовано у Волинській – 0,2%, Чернівецькій та Закарпатській областях – по 0,1%.

Загалом, найбільшу частку всіх забруднюючих речовин в Україні виділяють енергетика та промисловість – більше 60%. Значна частка транспорту. Наприклад, у Києві саме транспортні засоби є причиною майже 90% всіх небезпечних викидів. Найбільшу роль у забрудненні повітря Києва та наявності постійного смогу відіграють приватні автомобілі, які утворюють 80% усіх небезпечних викидів у місті, 10% – всі інші види транспорту. Решта джерел забруднення повітря столиці – ТЕЦ та комунальна сфера.

Таким чином, можна сказати, що в Україні основним чинником, який впливає на рівень викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел є стан економічного середовища в тим, чи іншім регіоні і в країні взагалі.

4.1.4 Теплова мапа рівня викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел

Чисте повітря необхідно людині для збереження і підтримання нормального здоров'я. Довгий час питання його забруднення не приділялося потрібної уваги. Проте з розвитком промисловості,

зростанням транспорту на душу населення, атмосфера в містах стрімко забруднюється, люди дихають повітрям, отруєним різними хімічними сполуками.

Постійні атмосферні забруднення несприятливо впливають на загальну захворюваність населення. Доведено прямий зв'язок між інтенсивністю забруднення повітря і станом здоров'я, а також зростанням хронічних неспецифічних захворювань, зокрема таких, як атеросклероз, хвороби серця, рак легень тощо. Забруднене повітря значно знижує імунітет та негативно впливає на органи дихання, сприяючи виникненню респіраторних захворювань, катарів верхніх дихальних шляхів, ларингіту, ларинготрахеїту, фарингіту, бронхіту, пневмонії. Забруднення спричиняють серцево-судинні та інші захворювання, зумовлюють виникнення віддалених наслідків, тобто мутагенну, канцерогенну, гонадотоксичну, тератогенну, алергенну, ембріотоксичну і атеросклеротичну дію.

В Україні негативного впливу атмосферних забруднень зазнає близько 17 млн. осіб, або 34% всього населення. Основу структури первинної захворюваності в 2011 році традиційно формували: хвороби органів дихання (43,7%), хвороби системи кровообігу (7,25%), травми, отруєння та деякі інші наслідки дії зовнішніх чинників (6,6%), хвороби сечостатевої системи (6,5%), хвороби кістково-м’язової системи (4,6%) та органів травлення (4,04%).

Проведені у Львові дослідження показали, що у водіїв автомобілів, регулювальників руху спостерігаються наявність карбоксигемоглобіну в крові, що спричиняє зниження рефлекторних реакцій, зрушення з боку активності деяких ферментів.

Отже, практично майже все міське населення, особливо діти, які дуже чутливі до токсичних речовин, вимушені дихати повітрям, що здатне отруювати організм.

Нижче подана теплова мапа (рис. 4.4) рівня викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел на 2015 рік.

Для побудови теплової мапи використано дані з офіційного сайту Державної служби статистики України - http://ukrstat.gov.ua (Статистична інформація / Багатогалузева статистична інформація / Регіональна статистика / Навколишнє природне середовище / Викиди забруднюючих речовин в атмосферне повітря / Викиди

забруднюючих речовин в атмосферне повітря від стаціонарних джерел забруднення за регіонами (1990-2017)).

Зображення побудоване за допомогою API Google Map [2].

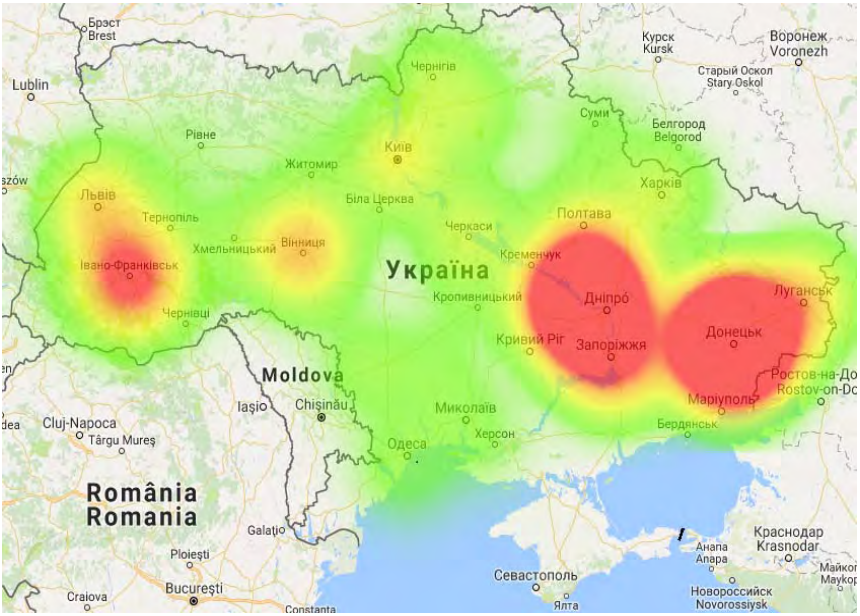


Рисунок 4.4 – Теплова мапа рівня викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел забруднення на 2015 рік

4.1.5 Постановка завдання

Математична залежність показників захворюваності від обсягів викидів забруднюючих речовин можна визначити як функцію, в якій незалежною змінною буде обсяг викидів забруднюючих речовин, а залежною – показник захворюваності:

$$K_{morb} = f\left(x_{emiss}\right),$$

де K_{morb} – показник захворюваності; x_{emiss} – вплив обсягів викидів.

Виходячи з наведених даних і аналізу статистичних даних, можна дійти висновку, що шукана математична модель буде не детермінованою, а скоріш стохастичною.

Більшість експертів сходяться на думці, що окрім обсягів викидів забруднюючих речовин на рівень захворюваності, вочевидь, оказують вплив і безліч інші факторів, точну кількість яких визначити досить проблематично. Якщо позначити ці фактори як x_1, x_2, …, x_n, то узагальнену модель залежності можна записати як:

$$K_{morb} = f\left(x_{emiss}, x_1, x_2, ..., x_n\right).$$

В даному випадку робиться припущення, що обсяг викидів забруднюючих речовин в атмосферне повітря від стаціонарних джерел є домінуючим фактором впливу, і його вплив на показник захворюваності в регіоні буде мати найбільшу вагу.

Під час аналізу предметної області встановлено, що основним фактором впливу викидів на здоров'я людини є наявність в їх складі токсичних речовин [10]. В свою чергу, характер і ступінь впливу токсичних речовин, їх здатність провокувати патологічні стани в організмі людини варіюють в залежності від комбінації метеорологічних і кліматичних факторів таких, наприклад, як температура повітря і кількість опадів.

Крім цього, звичайно, на показники захворюваності впливає якість медичного обслуговування населення. У якості основних метрик, які доцільно враховувати при будуванні моделі залежності показників захворюваності, використано показники кількості лікарів (усіх спеціалізацій) у регіоні і кількість лікарняних ліжок у стаціонарних відділеннях медичних закладів регіону, як кількісний показник обсягів медичного обслуговування.

Нарешті, оскільки розподілення захворюваності в різних регіонах є статистичним, для моделювання такої залежності варто враховувати і кількість населення в регіоні.

Оскільки, згідно медичній статистиці, загальна захворюваність населення має різні показники у різних вікових групах (зазвичай, збільшується з віком), доцільно також враховувати середній вік населення у регіоні.

Таким чином, узагальнену модель залежності показників захворюваності від обсягів викидів з деяким припущенням можна привести до виду:

$$K_{morb} = f\left(x_{emiss}, x_{popul}, x_{temp}, x_{rain\,fall}, x_{docs}, x_{beds}, x_{age}\right),$$

де K_{morb}– показник захворюваності;x_{emiss} – вплив обсягів викидів; x_{popul} – вплив кількості населення; x_{temp} – вплив середньої температури повітря; $x_{rainfall}$ – вплив кількості опадів; x_{docs} – вплив кількості лікарів; $x_{beds.}$– вплив загальної кількості ліжок у стаціонарних відділеннях; x_{age} – вплив середнього віку населення.

Для досягнення мети роботи, необхідно провести аналіз динаміки показників викидів забруднюючих речовин в атмосферне повітря від стаціонарних джерел забруднення та динаміки показників захворюваності по таким хворобам, як хвороби системи кровообігу, туберкульоз, та захворювання на рак (онкологічні захворювання) в різних регіонах України, виявити залежності між ними та створити програмний засіб для моделювання цієї залежності, який надавав би можливість прогнозувати динаміку показників захворюваності від обсягів викидів забруднюючих речовин в атмосферне повітря у майбутньому.

4.2 Побудова моделей залежності показників здоров'я від обсягів викидів забруднюючих речовин у повітря

Діаграми залежностей рівня захворюваності від обсягів викидів забруднюючих речовин і діоксиду вуглецю в атмосферне повітря від стаціонарних джерел за роки з 1990 по 2016 представлені на рисунках 4.5 – 4.7.

З наведених діаграм можна зробити висновок, що залежність показників захворюваності від обсягів викидів забруднюючих речовин в атмосферне повітря є нелінійною, а модель такої залежності є стохастичною. Вочевидь, в даному випадку доцільним є використання більшої кількості параметрів, що мають вплив на значення показників захворюваності в регіоні, і розглядати рівень захворюваності як результат впливу сукупності цих факторів.

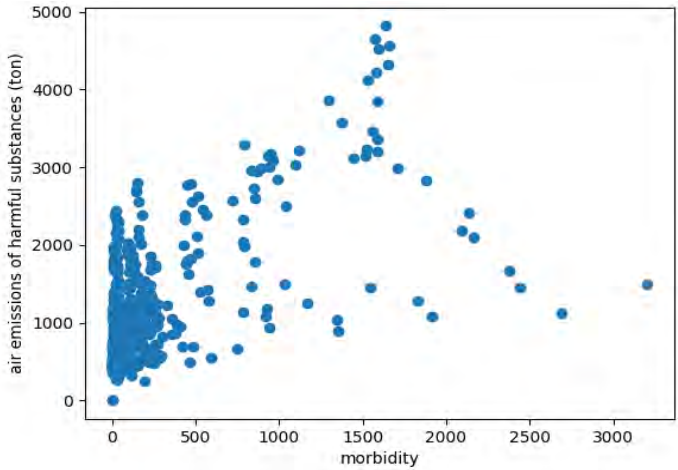


Рисунок 4.5 - Діаграма розподілення рівня захворюваності туберкульозом в залежності від рівня викидів забруднюючих речовин в атмосферне повітря. Діаграма відображає залежність кількості випадків захворювання (morbidity) від обсягу викидів забруднюючих речовин в тонах (air emission of harmful substances)

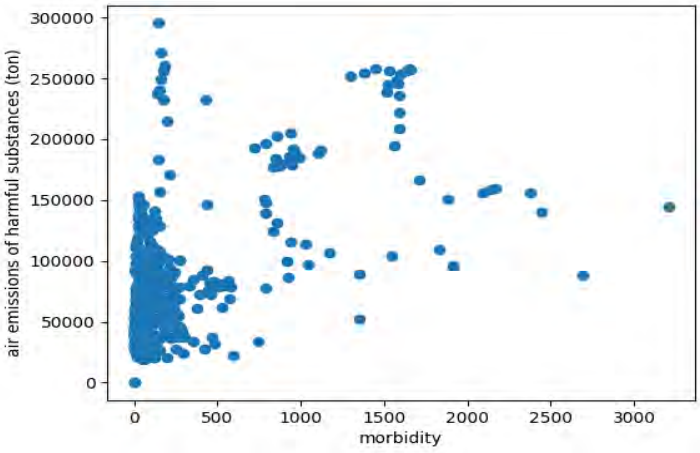


Рисунок 4.6 - Діаграма розподілення рівня захворюваності хворобами системи кровообігу в залежності від рівня викидів забруднюючих речовин в атмосферне повітря. Діаграма відображає залежність кількості випадків захворювання (morbidity) від обсягу викидів забруднюючих речовин в тонах (air emission of harmful substances)

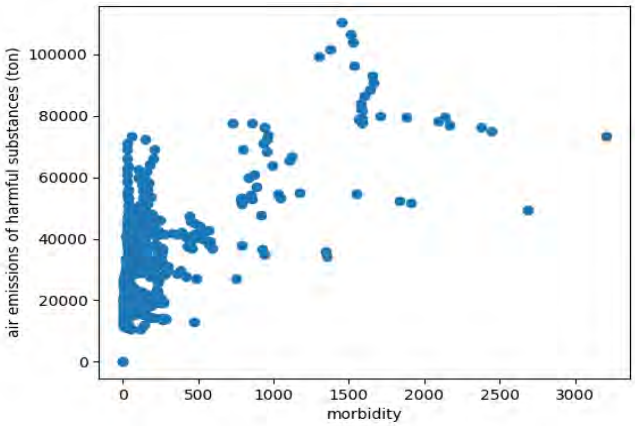


Рисунок 4.7 - Діаграма розподілення рівня захворюваності раком в залежності від рівня викидів забруднюючих речовин в атмосферне повітря. Діаграма відображає залежність кількості випадків захворювання (morbidity) від обсягу викидів забруднюючих речовин в тонах (air emission of harmful substances)

Для навчання і тестування моделей було створено вибірку, що включає в себе такі показники: рівень викидів, кількість населення, середню річну температуру повітря, річний рівень опадів, кількість лікарів в регіоні, кількість лікарняних ліжок в закладах охорони здоров'я в регіоні і середній вік населення.

У якості метрик оцінювання моделей використано коефіцієнт детермінації і середнє абсолютне відхилення.

Коефіцієнт детермінації – це частка дисперсії залежної змінної, яка пояснюється розглянутої моделлю залежності. Коефіцієнт детермінації моделі визначається за формулою:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}},$$

де $SS_{res} = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ – сума квадратів залишків регресії; $y_i, \hat{y}_i$ – фактичні і розрахункові значення змінної, що пояснюється; $SS_{tot} = \sum_{i=1}^{n} (y_i - \hat{y})^2$ –загальна сума квадратів; $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$.

Середнє абсолютне відхилення (Mean Absolute Error, MAE) – є відносною мірою якості прогнозів, тобто може бути використана для порівняння двох (або більше) різних прогнозів одного і того ж показника між собою: найкращим вважається той прогноз, у якого MAE менше. Середнє абсолютне відхилення прогнозу виражається в абсолютних одиницях вимірювання об'єкту, що прогнозується. Якщо визначити фактичне значення як $y_{\text{факт}}$, а прогнозне значення як $y_{\text{прогноз}}$, то середнє абсолютне відхилення прогнозу для серії з n випробувань, можна виразити як:

$$MAE = \frac{\sum_{i=1}^{n} \left| y_{\text{факт},i} - y_{\text{прогноз},i} \right|}{n}.$$

На основі різних методів побудови моделей залежностей для вирішуваної задачі будувалися моделі, результати випробування яких наведено у табл. 4.1 – 4.3. Тут позначено: Linear Regression – лінійна регресія, Random Forest Regressor – регресія випадкового лісу дерев, KNeighbors Regressor – регресія K найближчих сусідів, SVR – регресія на основі методу опорних векторів, Logistic Regression – логістична регресія.

Таблиця 4.1 – Результати випробування моделей залежностей для показника «Кількість нових випадків туберкульозу»

Модель	R^2	*MAE*
Linear Regression	0,721011	22,957569
Random Forest Regressor	0,964696	7,671577
KNeighbors Regressor	0,965791	8,236485
SVR	-0,111545	40,270765
Logistic Regression	0,499439	29,764103

Таблиця 4.2 – Результати випробування моделей залежностей для показника «Кількість випадків захворювань системи кровообігу»

Модель	R^2	*MAE*
Linear Regression	0,623239	1789,727716
Random Forest Regressor	0,950724	571,018151
KNeighbors Regressor	0,953287	573,004327
SVR	-0,097711	2794,433333
Logistic Regression	-0,510438	3814,174359

Таблиця 4.3 – Результати випробування моделей залежностей для показника «Кількість всіх зареєстрованих випадків раку»

Модель	R^2	*MAE*
LinearRegression	0,866728	367,381240
RandomForestRegressor	0,974921	156,387862
KNeighborsRegressor	0,961540	210,157265
SVR	-0,080782	1050,690702
LogisticRegression	-0,205942	1400,357372

До того ж, для зазначених параметрів було розраховано коефіцієнти важливості (згідно обраних метрик) для моделі випадкового лісу рішень (Random Forest Regression), які наведені у таблицях 4.4 – 4.6.

Таблиця 4.4 – Коефіцієнти важливості параметрів для моделі випадкового лісу при визначенні показника «Кількість нових випадків туберкульозу»

Параметр	R^2	*MAE*
Обсяг викидів	0,13425758	0,13387665
Кількість населення	0,33551644	0,3463629
Середня річна температура повітря	0,01404272	0,01356626
Річна кількість опадів	0,02482378	0,02529506
Загальна кількість лікарів (всі спеціалізації)	0,22707743	0,19951564
Загальна кількість лікарняних ліжок	0,12874393	0,14918443
Середній вік населення	0,13553811	0,13219907

Таблиця 4.5 – Коефіцієнти важливості параметрів для моделі випадкового лісу при визначенні показника «Кількість випадків захворювань системи кровообігу»

Параметр	R^2	*MAE*
Обсяг викидів	0,11206173	0,1268073
Кількість населення	0,28892009	0,37365804
Середня річна температура повітря	0,01355616	0,01415118
Річна кількість опадів	0,02759653	0,0258892
Загальна кількість лікарів (всі спеціалізації)	0,26044356	0,1885401
Загальна кількість лікарняних ліжок	0,14068157	0,12273914
Середній вік населення	0,15674035	0,14821503

Таблиця 4.6 – Коефіцієнти важливості параметрів для моделі випадкового лісу при визначенні показника «Кількість всіх зареєстрованих випадків раку»

Параметр	R^2	*MAE*
Обсяг викидів	0,16426391	0,1495061
Кількість населення	0,38449663	0,33411377
Середня річна температура повітря	0,01246084	0,01118696
Річна кількість опадів	0,01792925	0,01652693
Загальна кількість лікарів (всі спеціалізації)	0,20455071	0,269674
Загальна кількість лікарняних ліжок	0,12335029	0,1223544
Середній вік населення	0,09294837	0,09663785

Значення коефіцієнту детермінації випробуваних моделей при визначенні показника «Кількість нових випадків туберкульозу», зображено на рисунку 4.8, значення середнього абсолютного відхилення випробуваних моделей при визначенні показника «Кількість нових випадків туберкульозу», зображено на рисунку 4.9, значення коефіцієнту детермінації випробуваних моделей при визначенні показника «Кількість випадків захворювань системи кровообігу», зображено на рисунку 4.10, Значення середнього абсолютного відхилення випробуваних моделей при визначенні показника «Кількість випадків захворювань системи кровообігу» зображено на рисунку 4.11 та Значення середнього абсолютного

відхилення випробуваних моделей при визначенні показника «Кількість випадків захворювань системи кровообігу» зображено на рисунку 4.12.

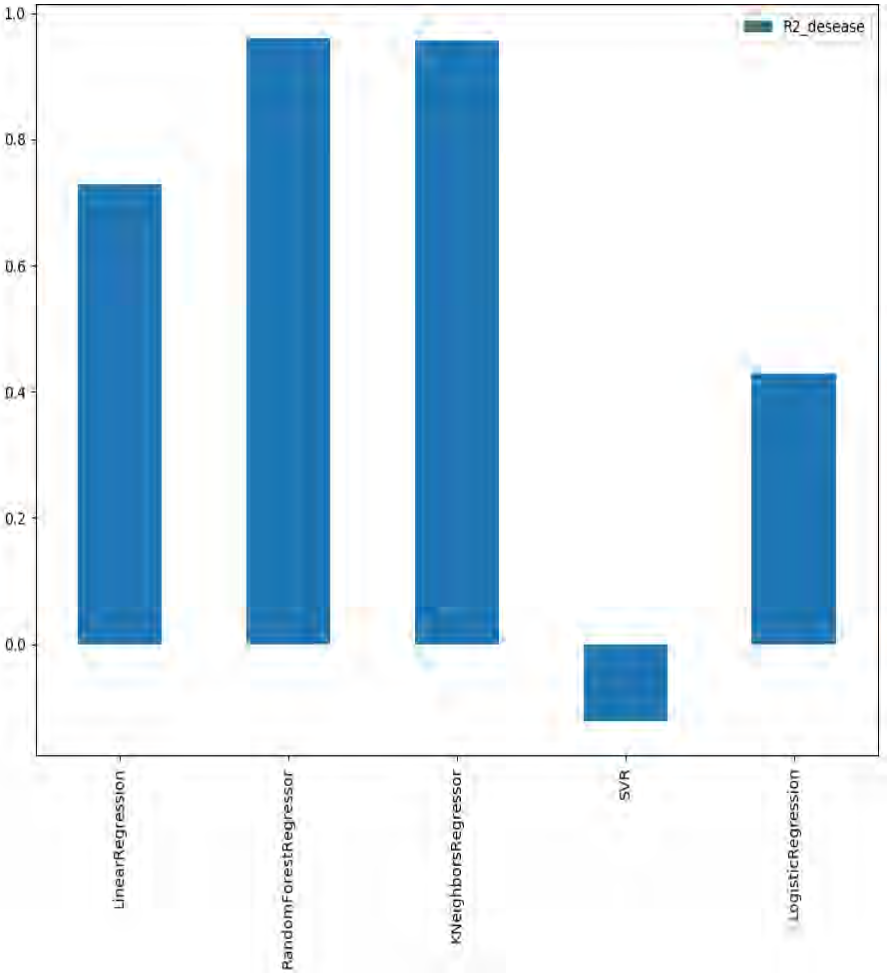


Рисунок 4.8 – Значення коефіцієнту детермінації випробуваних моделей при визначенні показника «Кількість нових випадків туберкульозу»,

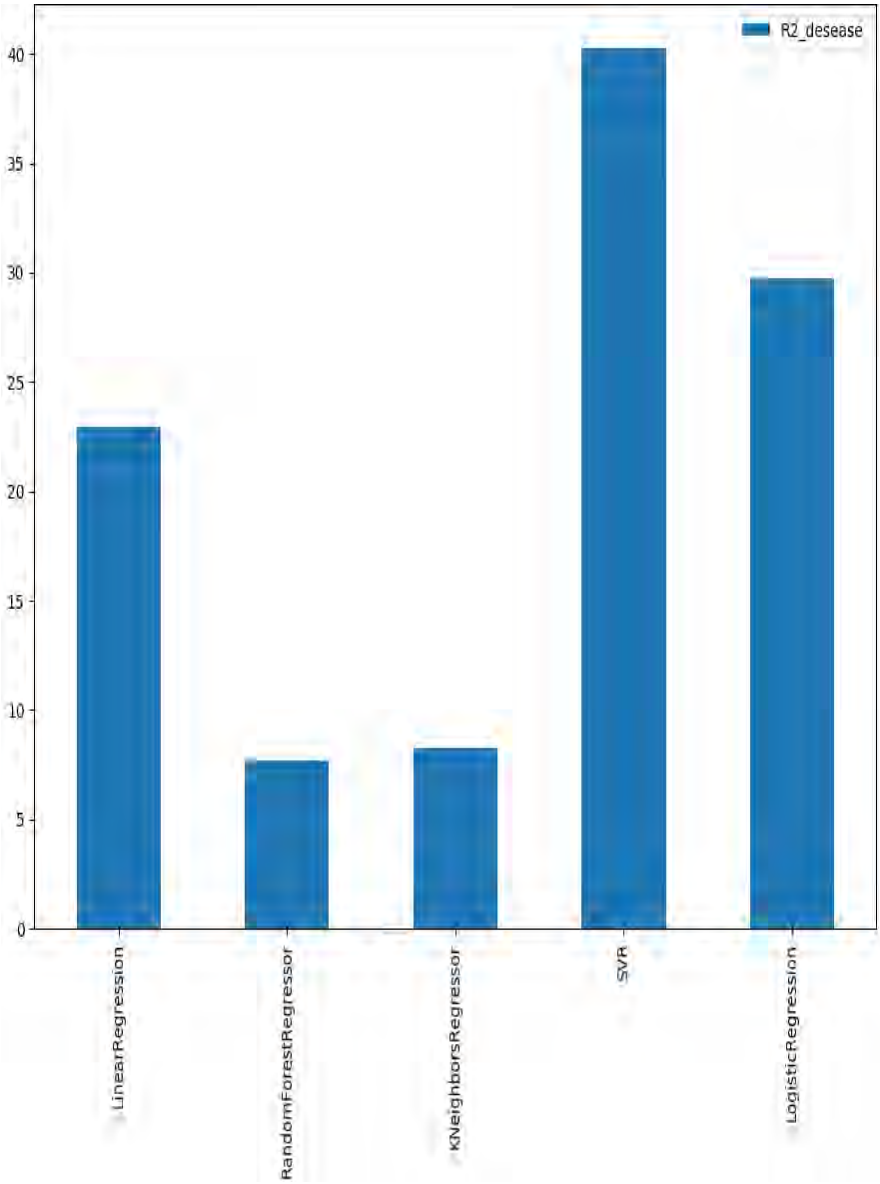


Рисунок 4.9 – Значення середнього абсолютного відхилення випробуваних моделей при визначенні показника «Кількість нових випадків туберкульозу»

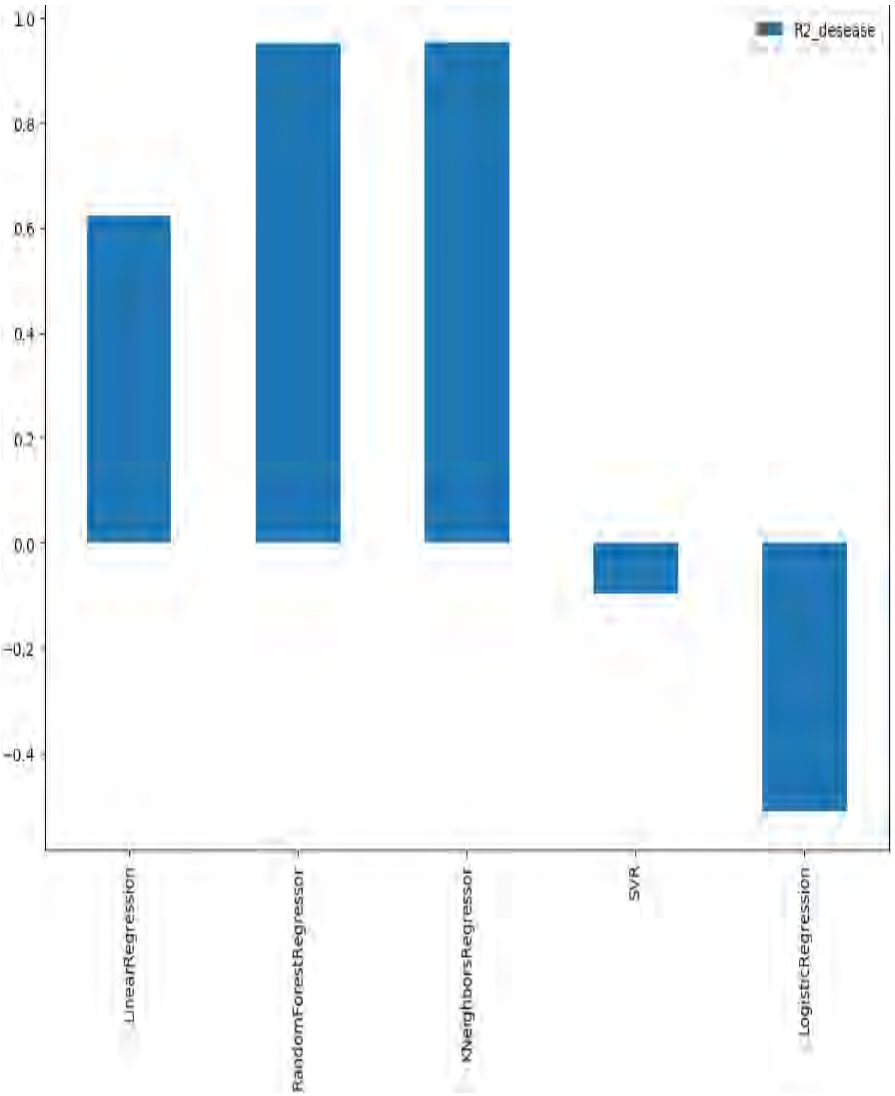


Рисунок 4.10 – Значення коефіцієнту детермінації випробуваних моделей при визначенні показника «Кількість випадків захворювань системи кровообігу»

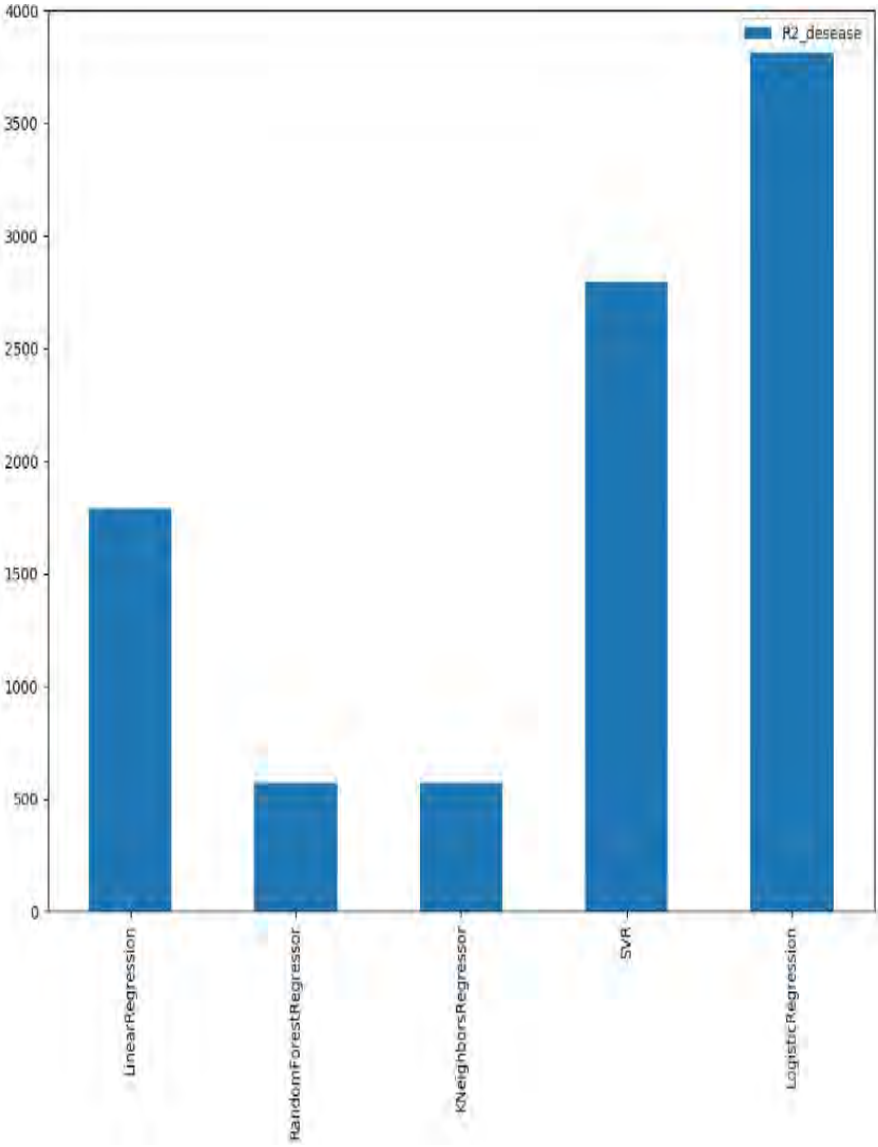


Рисунок 4.11 – Значення середнього абсолютного відхилення випробуваних моделей при визначенні показника «Кількість випадків захворювань системи кровообігу»

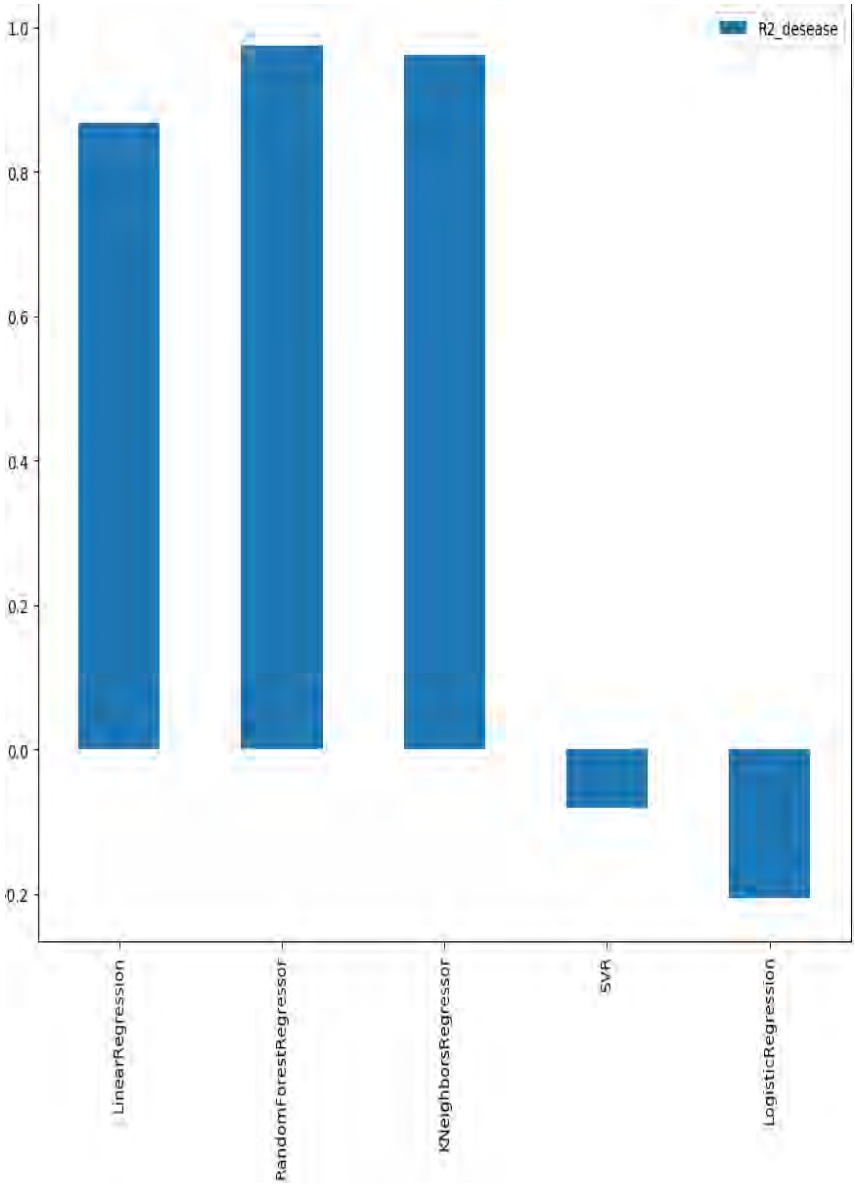


Рисунок 4.12 – Значення коефіцієнту детермінації випробуваних моделей при визначенні показника «Кількість всіх зареєстрованих випадків раку»

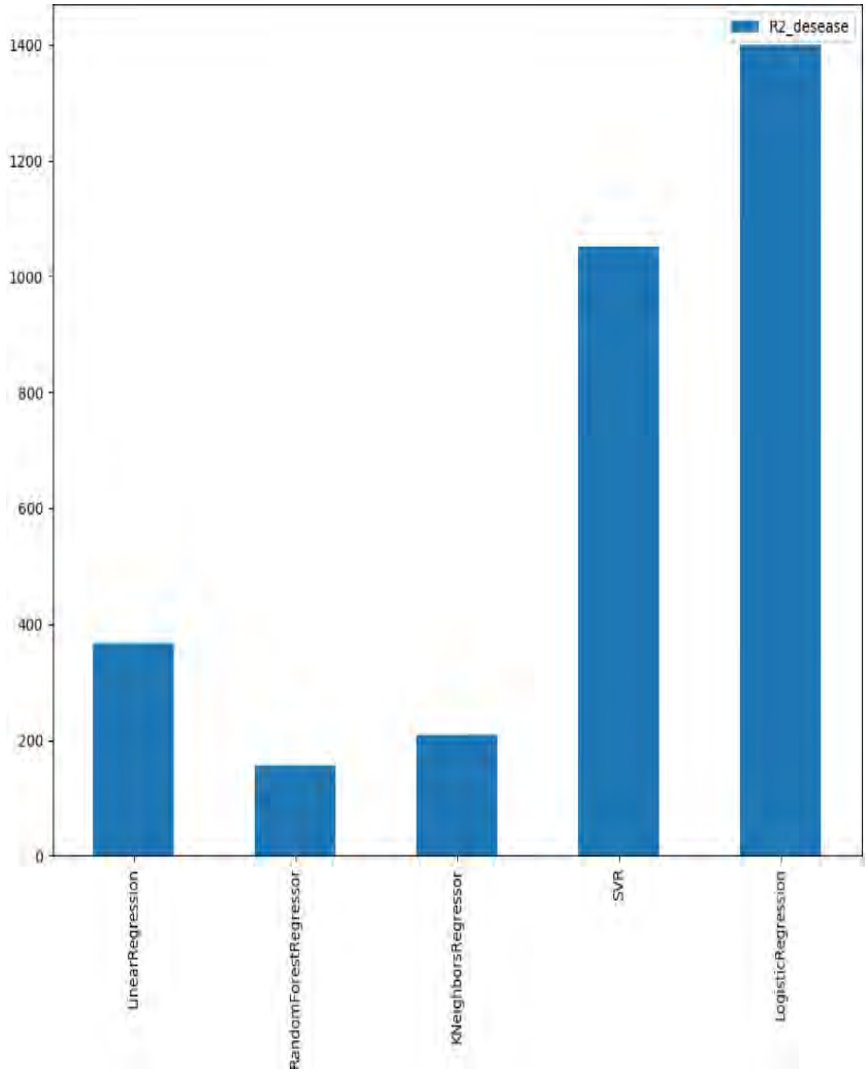


Рисунок 4.13 – Значення середнього абсолютного відхилення випробуваних моделей при визначенні показника «Кількість всіх зареєстрованих випадків раку»

Було створено і досліджено декілька моделей створених з залученням методу штучних нейронних мереж.

Для побудови моделі залежності показників захворюваності від обсягів викидів забруднюючих речовин використано тип штучної мережі – багатошаровий перцептрон.

У якості вхідних параметрів використовується показники обсягів викидів забруднюючих речовин в атмосферне повітря, кількості населення, середнього віку населення, середньої температури, кількості опадів, кількості лікарів у регіоні і кількості ліжок у стаціонарних відділеннях закладів охорони здоров'я у регіоні.

Спочатку, було побудовано просту модель, що має зовнішній шар з сімома нейронами (по кількості вхідних параметрів), один прихований повнозв'язаний шар з 12 нейронами і вихідний шар з одним нейроном (рис. 4.14):

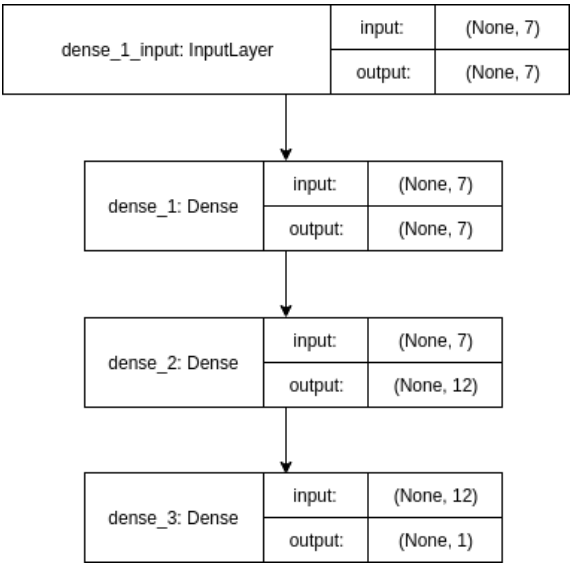
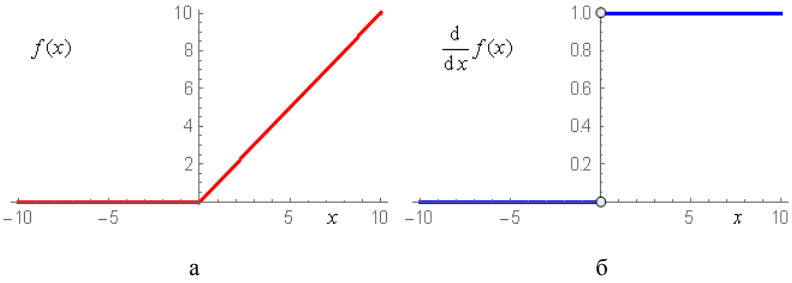


Рисунок 4.14 – Схема MLP з трьома повнозв'язними шарами нейронів (в нотації бібліотеки Theano). Вхідний шар мережі містить 7 нейронів (на кожний з них подається вхідний сигнал), прихований шар (12 нейронів) і вихідний шар, що складається з одного нейрону. У якості функції активації використано ReLU. Первинна ініціалізації синаптичних ваг відповідає нормальному розподіленню

Вибір параметрів нейронної мережі, зокрема кількості нейронів прихованого шару, в більшості випадків є творчою задачею і виконується на підставі експертної оцінки. Однак, існує декілька рекомендацій з цього приводу. Так Hecht-Neilson [36] для обчислення верхньої межі кількості прихованих елементів використовував теорему Колмогорова, яка стверджує, що будь-яка функція n змінних може бути представлена як суперпозиція $2n+1$ одновимірних функцій. Ця межа h дорівнює подвоєній кількості вхідних елементів плюс одиниця [35-37]: $h \leq 2i+1$.

Таким чином, вхідний шар мережі містить сім нейронів (по кількості вхідних параметрів), прихований шар містить 12 нейронів ($12 < 2 \times 7$), вихідний шар містить один нейрон. Нейрони вхідного і прихованого шару у якості функції активації використовують ReLU [37] (Rectifier activation function, рис. 4.15):

$$f(x) = \max(0, x).$$



а б

Рисунок 4.15 – Графіки: а – функції ReLU, б –першої похідної функції ReLU

Таким чином, мережа має 160 параметрів (синаптичних ваг), що можуть бути треновані.

Для первинної ініціалізації синаптичних ваг використано нормальний розподіл (середнє значення дорівнює 0, а стандартне відхилення дорівнює 0,05).

У якості алгоритму оптимізації при навчанні мережі використано алгоритм ADAM. Adam – це алгоритм оптимізації, який може використовуватися замість класичної процедури зниження

випадкового градієнта, щоб оновити ітераційну вагу мережі, засновану на навчальних даних [36]. Алгоритм поєднує у собі переваги таких розширень класичного градієнтного спуску, як адаптивний градієнтний алгоритм (AdaGrad) і розповсюдження середнього квадрату (RMSProp).

Головна відмінність алгоритму полягає в середніх значень як градієнтів, так і других моментів градієнтів. Оновлення синаптичних ваг мережі при використанні алгоритму adam здійснюється таким чином:

$$m_{\omega}^{(t+1)} = \beta_1 m_{\omega}^{(t)} + (1 - \beta_1)\nabla_{\omega} L^{(t)},$$

$$v_{\omega}^{(t+1)} = \beta_2 v_{\omega}^{(t)} + (1 - \beta_2)\left(\nabla_{\omega} L^{(t)}\right)^2,$$

$$\hat{m}_{\omega} = \frac{m_{\omega}^{(t+1)}}{1 - \beta_1^t}, \hat{v}_{\omega} = \frac{v_{\omega}^{(t+1)}}{1 - \beta_2^t}, \omega^{(t+1)} = \omega^{(t)} - \eta \frac{\hat{m}_{\omega}}{\sqrt{\hat{v}_{\omega} + \varepsilon}},$$

де β_1, β_2 – гіперпараметри, що позначають експоненціальні темпи розпаду на момент оцінки; η – початковий рівень навчання; ε – це невелика константа, введена для чисельної стабільності; m_{ω} – експоненціальне рухоме середнє градієнта; ν_{ω} – експоненціальне середнє квадрата градієнта; $\nabla_{\omega} L^{(t)}$ – значення градієнта за часом t; ω – вектор параметрів градієнтного спуска [37].

Враховуючи обмеженість вибірки статистичних даних та їх різну мірність, до початку навчання проведено стандартизацію вхідних даних. Дані були перетворені таким чином, щоб їх середнє значення дорівнювало 0, а дисперсія 1.

Створена мережа проходила навчання в продовж 200 епох з розміром підвибірки 75. Для тестування в продовж навчання використовувалася частина вхідних даних (10% вибірки).

У якості функції помилки в алгоритмі adam використано функцію *MSE* (Mean Squared Error) – середньо квадратичну помилку. У якості метрики мережі використано *MAE* – середню абсолютну помилку [38].

Результат навчання і роботи створеної мережі відображено на рисунках 4.16 – 4.21.

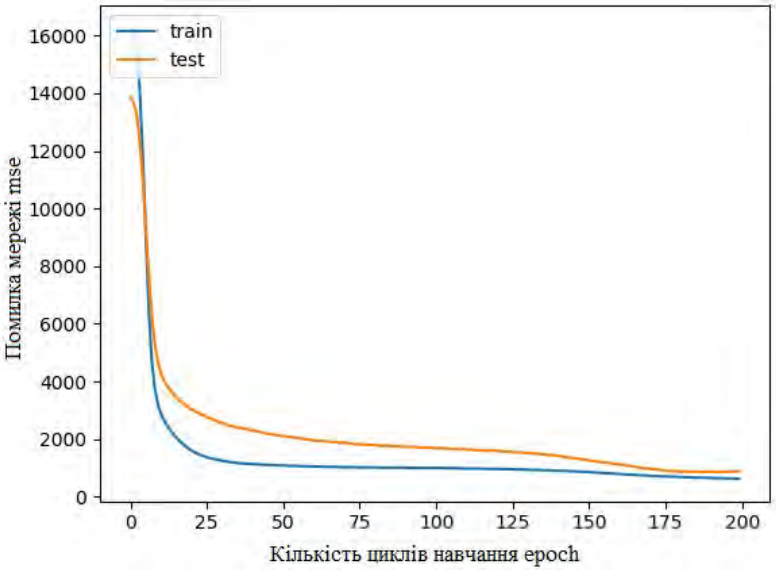


Рисунок 4.16 –Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість нових випадків туберкульозу»

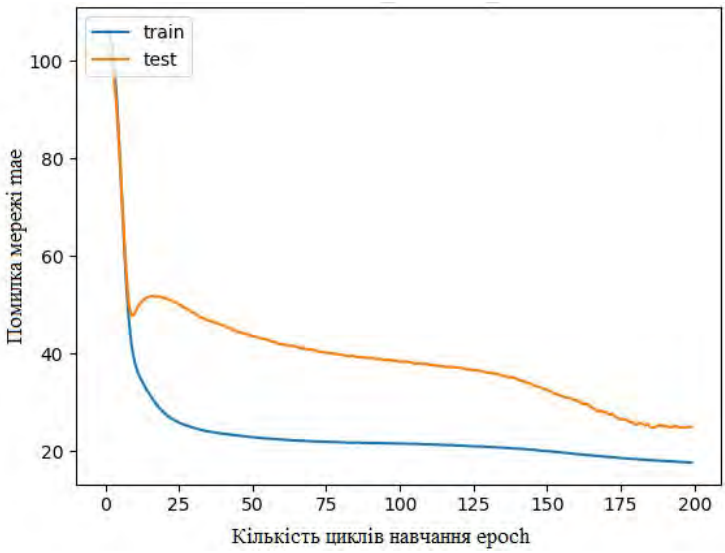


Рисунок 4.18 - Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість нових випадків туберкульозу»

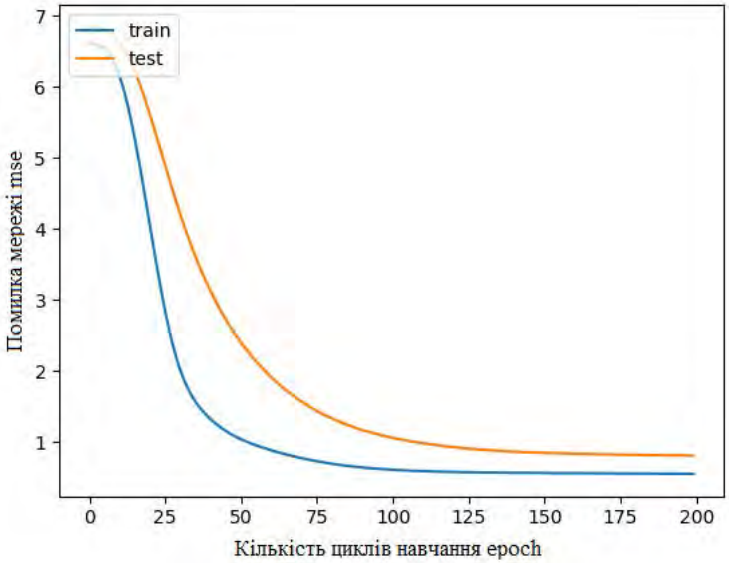


Рисунок 4.19 - Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість випадків захворювань системи кровообігу»

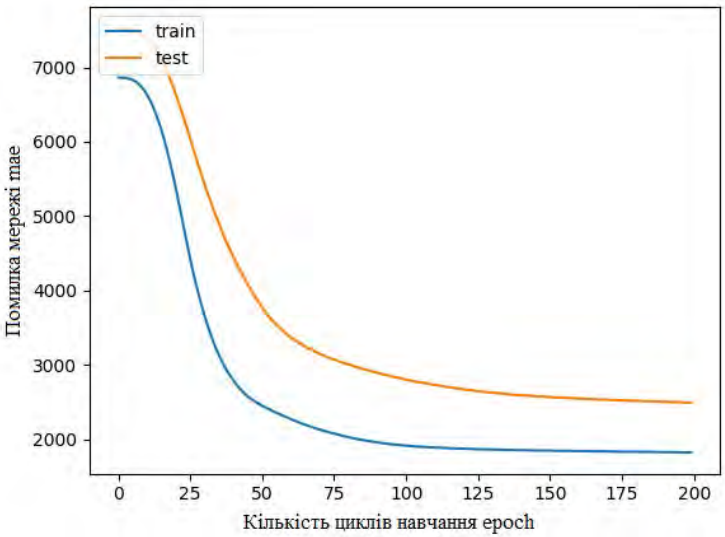


Рисунок 4.20 - Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість випадків захворювань системи кровообігу»

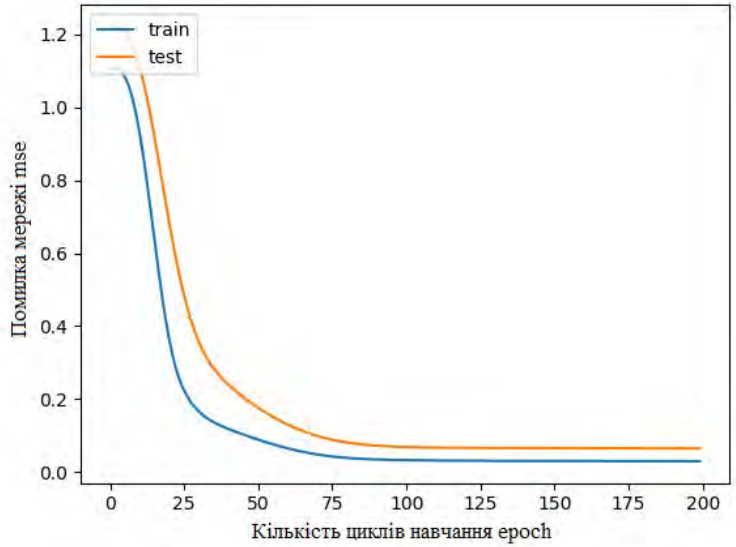


Рисунок 4.21 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показник «Кількість всіх зареєстрованих випадків раку»

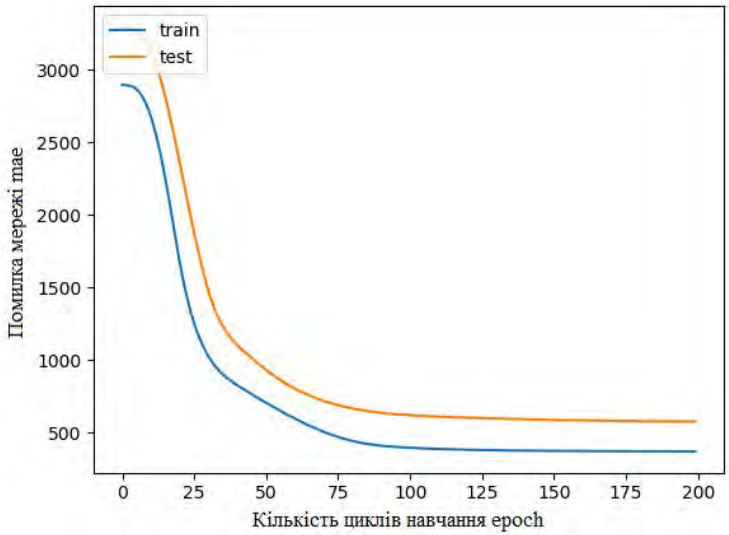


Рисунок 4.22 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість всіх зареєстрованих випадків раку»

На наведених графіках читко спостерігається поступове зниження значень помилки впродовж тренування мережі. В середньому, починаючи зі сотої епохи тренування, значення середньої помилки припиняє зменшуватися, що говорить про те, що модель стає тренованої і подальше навчання вже не відбувається. Дещо іншу картину можна побачити на рисунку 4.18. Вочевидь, в районі 10-15 епохи тренування досягає локального екстремуму. Про локальність мінімуму помилки може свідчити подальше поступове зменшення помилки мережі. Отже, доцільним в даному випадку є подальше тренування моделі.

Для покращення метрики нейронних мереж, їх збіжності, витрат на навчання, тощо існує декілька підходів пов'язаних з підбором оптимальної топології мережі та методів навчання.

В ході роботи над проєктом було створено декілька моделей на основі багатошарового перцептрону. Так у створено раніше модель було додано ще один повнозв'язний прихований шар з 12 нейронів, який у якості функції активації також використовує ReLU. Схема створеної мережі відображена на рисунку 4.23. Вхідний шар мережі містить 7 нейронів (на кожний з них подається вхідний сигнал), прихований повнозв'язний шар (12 нейронів), прихований повнозв'язний шар (12 нейронів) і вихідний шар, що складається з одного нейрону. У якості функції активації використано ReLU. Первинна ініціалізації синаптичних ваг відповідає нормальному розподіленню.

Як і попередньому випадку, навчання мережі проводилося впродовж 100 епох (розмір підвибірки 75) і сплітом валідаційної вибірки, що дорівнював 0,1. Первинна ініціалізація синаптичних ваг мережі відповідає нормальному розподіленню. Таким чином, мережа має 321 параметр (синаптичних ваг), що можуть бути треновані.

Результати навчання мережі наведені на рисунках 4.24– 4.29.

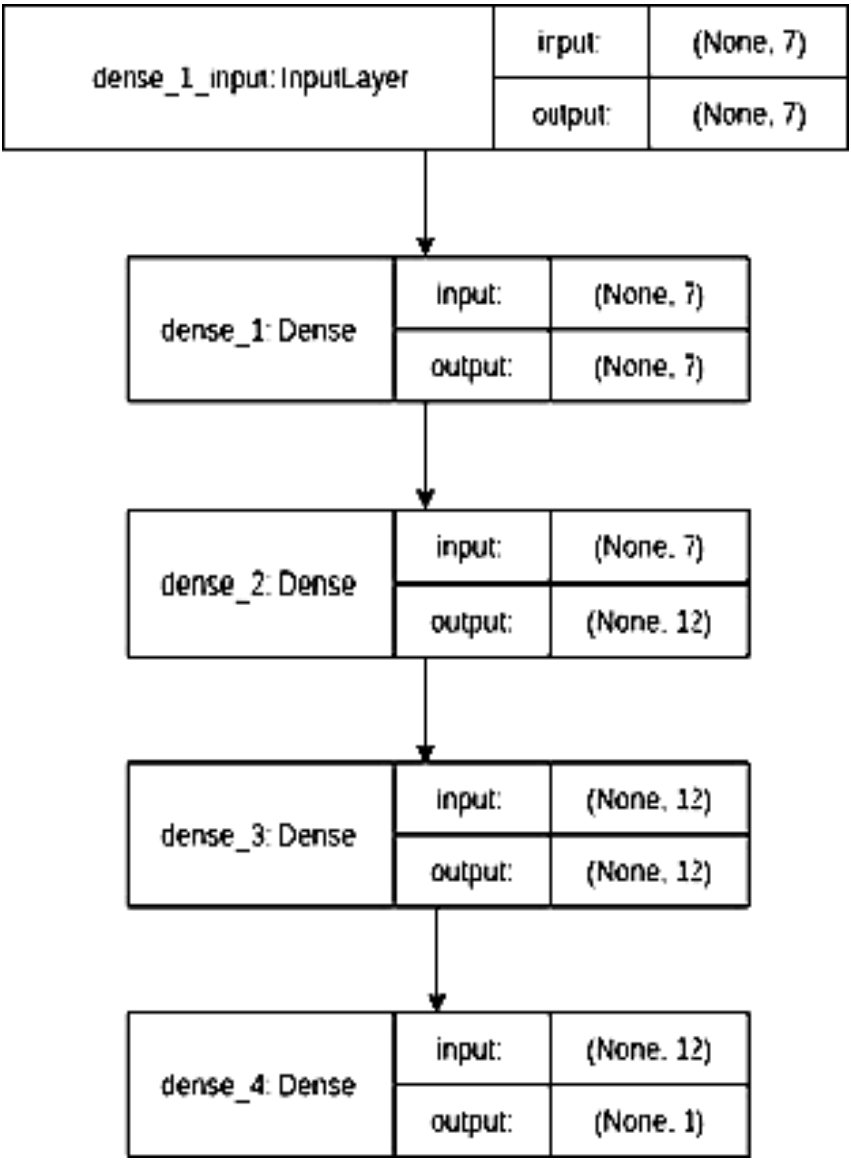


Рисунок 4.23 – Схема моделі нейронної мережі
з двома прихованими шарами (в нотації Keras)

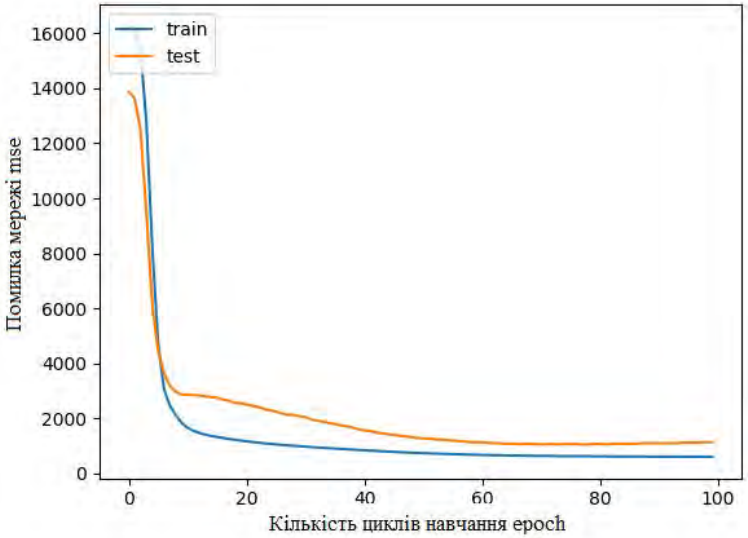


Рисунок 4.24 - Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показник «Кількість нових випадків туберкульозу»

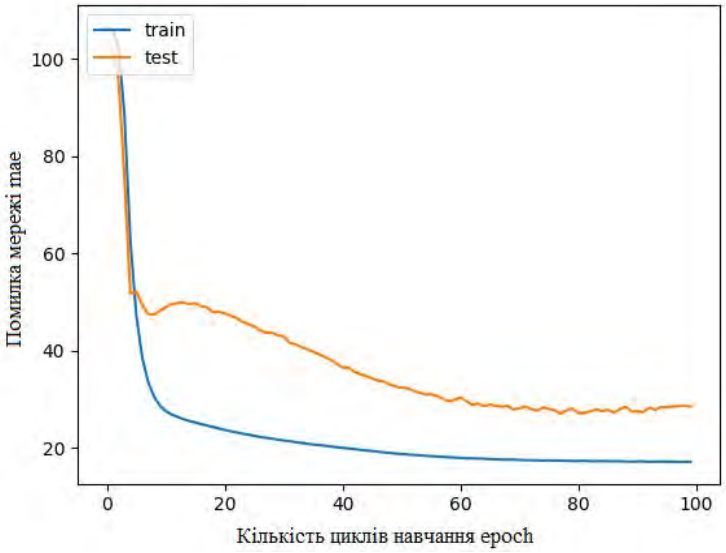


Рисунок 4.25 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість нових випадків туберкульозу»

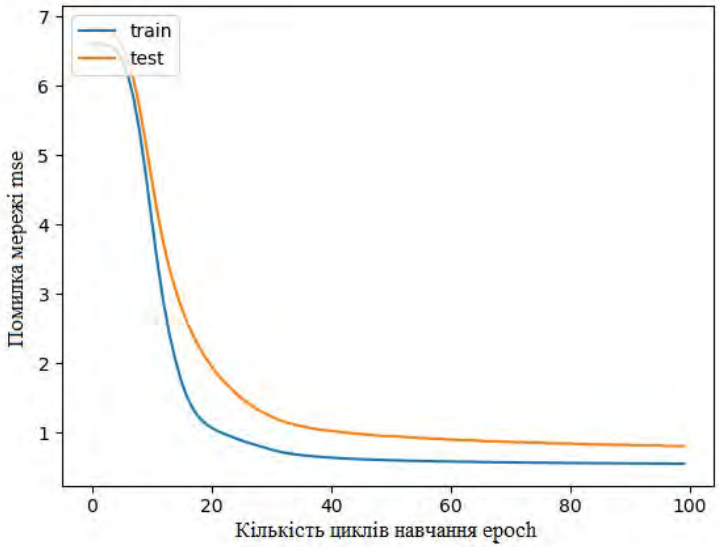


Рисунок 4.26 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість випадків захворювань системи кровообігу»

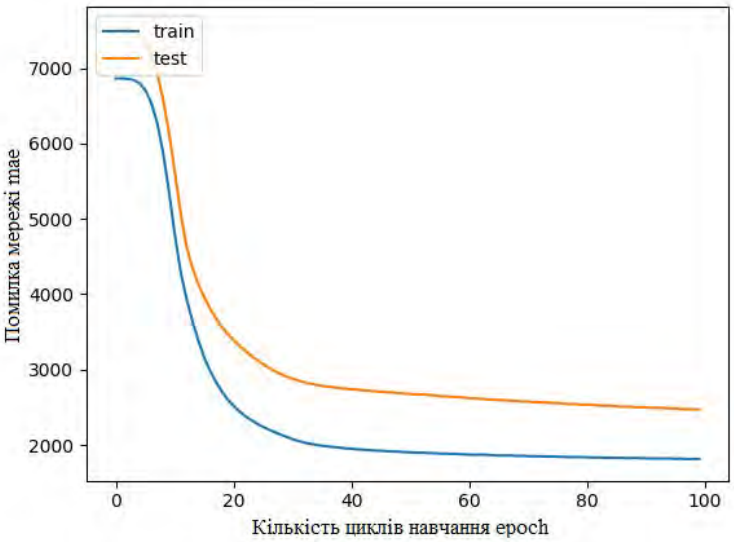


Рисунок 4.27 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість випадків захворювань системи кровообігу»

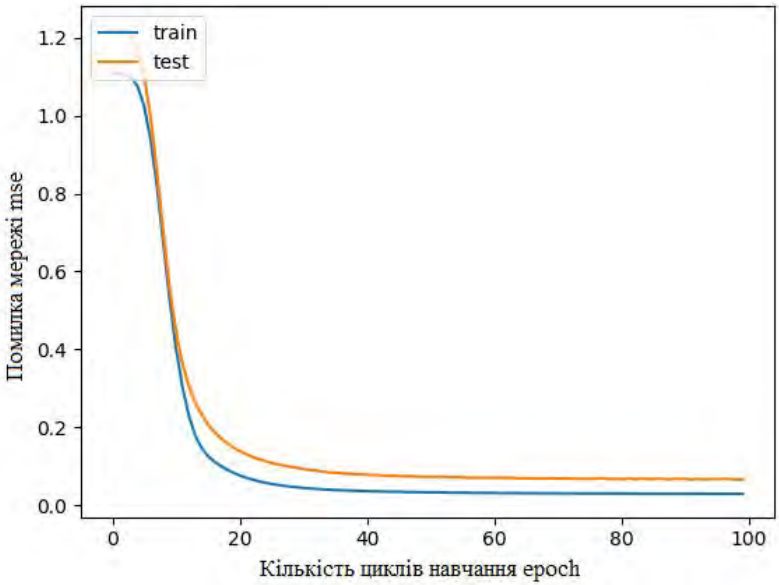


Рисунок 4.28 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість всіх зареєстрованих випадків раку»

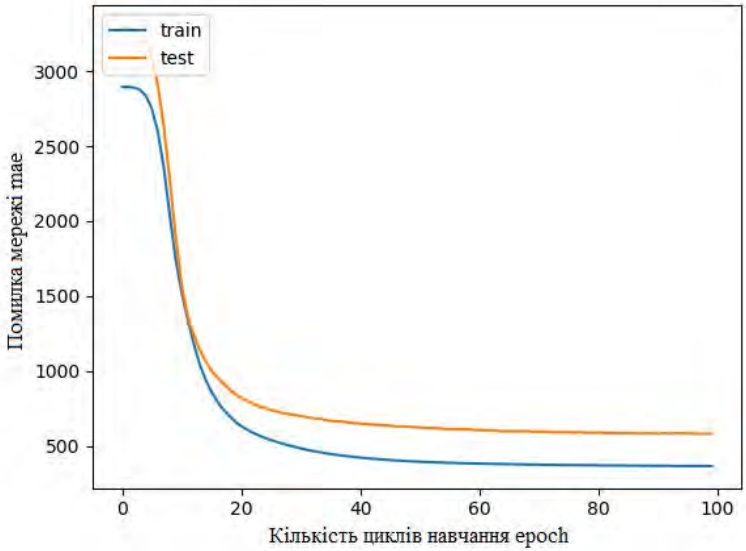


Рисунок 4.29 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість всіх зареєстрованих випадків раку»

В даному випадку, спостерігається схожа з попередньою моделлю картина – поступове зниження значень помилки впродовж тренування мережі. Враховуючи попередній досвід тренування MLP мережі, тренування відбувалося в продовж 100 епох. Наприкінці тренування спостерігається збіг мережі. Як і в попередньому випадку, для показника «Кількість нових випадків туберкульозу» спостерігається досягнення локального мінімуму помилки.

Одним з прийомів запобігання ефекту перенавчання нейронної мережі є метод виключення (Dropout), що полягає в виключенні деяких нейронів мережі підчас процесу навчання [39].

В створену раніше модель було додано виключення після першого прихованого шару (виключено 50% нейронів). Схема створеної моделі представлена на рис. 4.30. Вхідний шар мережі містить 7, повнозв'язний прихований шар (12 нейронів), дропаут (вимикає половину всіх нейронів шару), повнозв'язний прихований шар (12 нейронів і вихідний шар, що складається з одного нейрону. У якості функції активації використано ReLU. Первинна ініціалізації синаптичних ваг відповідає нормальному розподіленню.

Таким чином, мережа має 321 параметри (синаптичних ваг), що можуть бути треновані.

Результати навчання мережі відображені на рисунках 4.31 – 4.36.

Тренування моделі проводилося впродовж 100 епох, розмір підвибірки складав 75, валідаційний спліт – 0,1. Як і в попередніх випадках, під час тренування спостерігається збіг мережі, але дещо раніше – приблизно під час 60 – 70 епохи навчання. В усіх трьох випадках для показника «Кількість нових випадків туберкульозу» спостерігається знаходження локального мінімуму помилки мережі, що вочевидь є особливістю моделі на основі багатошарового перцептрону для даного показника захворюваності. Крім цього, завдяки використанню дропауту, крива тренування втрачає плавність і перетворюється на ломану.

Результати випробування моделі нейронної мережі відображені в таблицях Таблиця 4.7 – 4.9:

Представлені дані можна розглядати як часовий рядок, тобто значення розглянутих параметрів змінюються в часі. Для аналізу і прогнозування часових рядів можна використовувати моделі на основі нейронних мереж довгої короткочасної пам'яті.

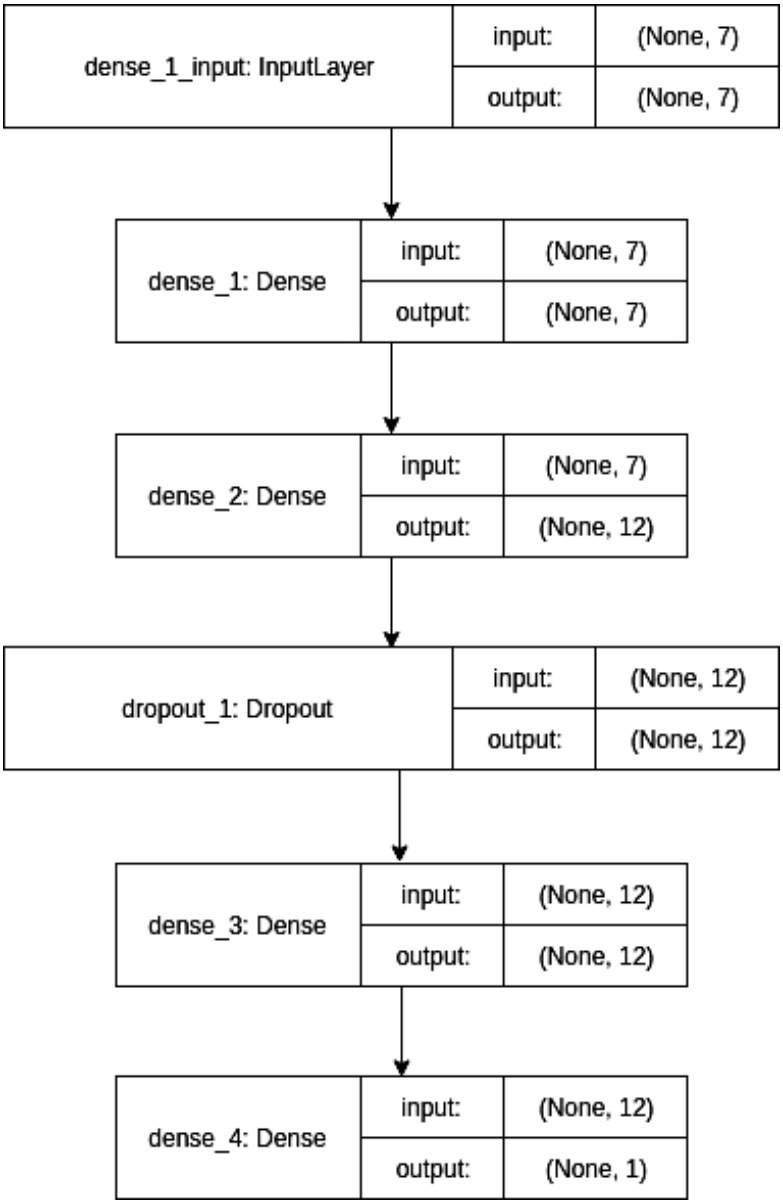


Рисунок 4.30 – Модель нейронної мережі з двома прихованими шарами і дропаутом

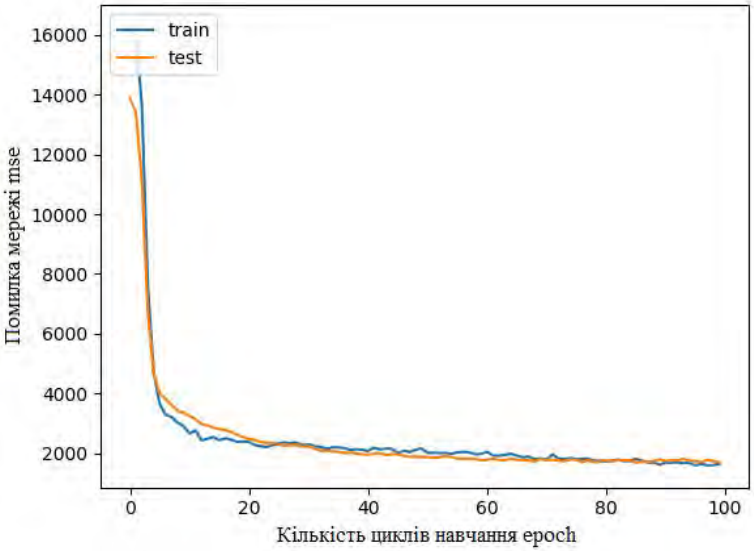


Рисунок 4.31 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість нових випадків туберкульозу»

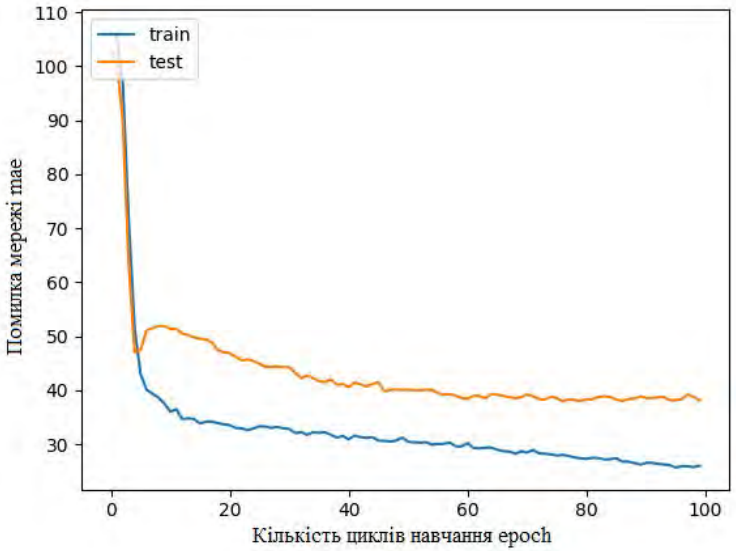


Рисунок 4.32 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість нових випадків туберкульозу»

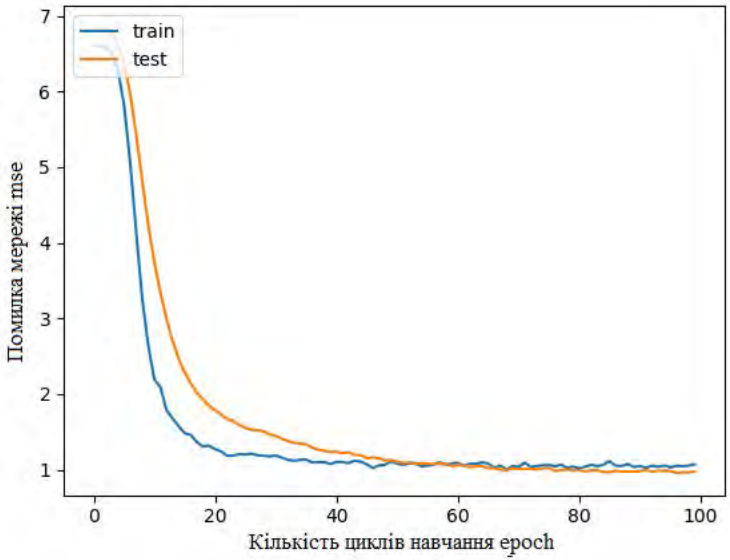


Рисунок 4.33 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість випадків захворювань системи кровообігу»

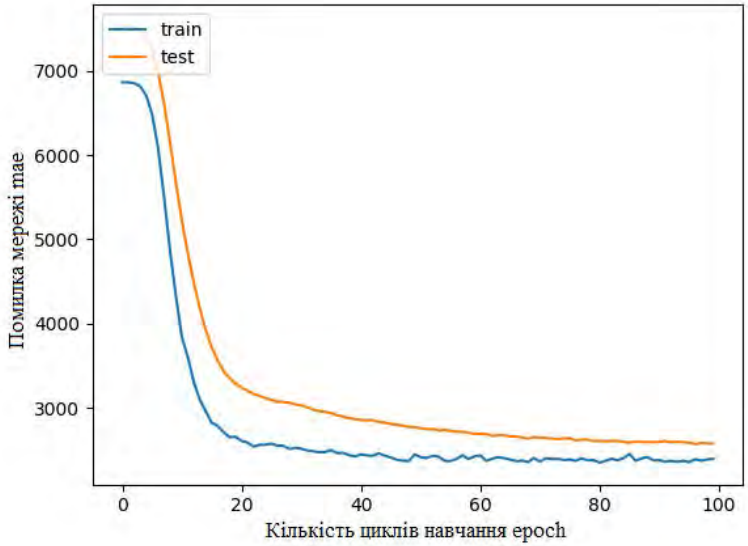


Рисунок 4.34 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість випадків захворювань системи кровообігу»

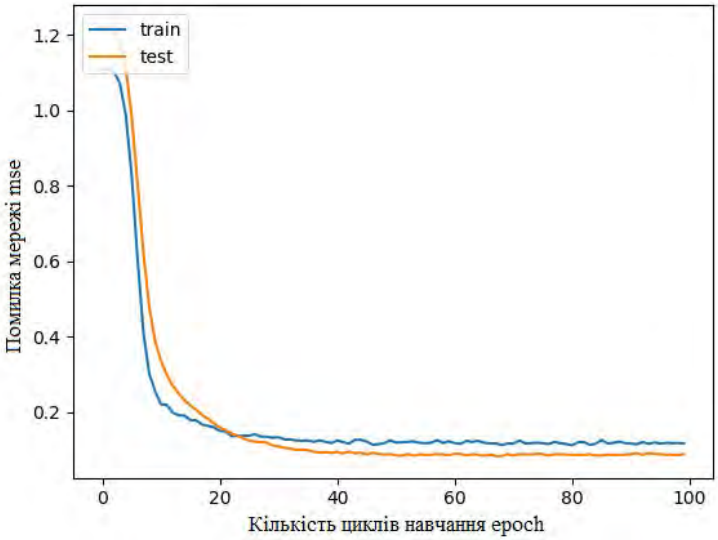


Рисунок 4.35 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість всіх зареєстрованих випадків раку»

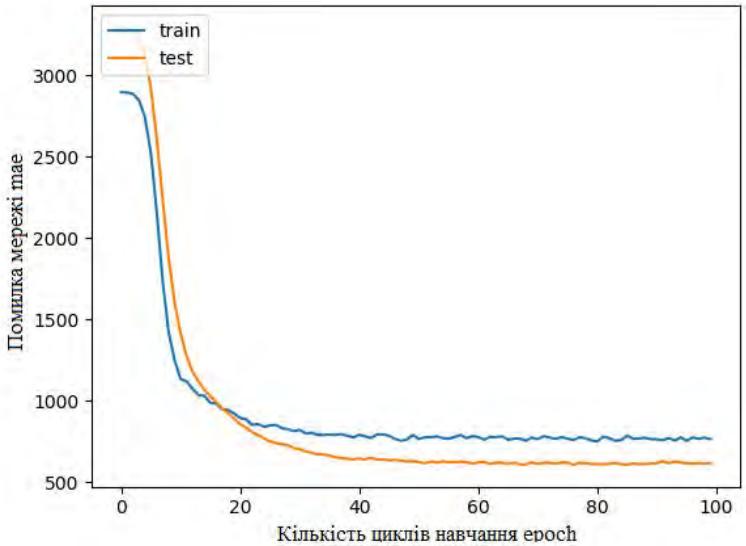


Рисунок 4.36 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість всіх зареєстрованих випадків раку»

Таблиця 4.7 – Результати випробувань моделі нейронної мережі з одним прихованим шаром

Показник захворюваності	*MAE*
Число нових випадків туберкульозу	23,4455625425
Число випадків захворювань системи кровообігу	1766,47043614
Число всіх зареєстрованих випадків раку	336,419792045

Таблиця 4.8 – Результати випробувань моделі нейронної мережі з двома прихованими шарами

Показник захворюваності	*MAE*
Число нових випадків туберкульозу	21,6752039937
Число випадків захворювань системи кровообігу	1752,4164209
Число всіх зареєстрованих випадків раку	338,953238874

Таблиця 4.9 – Результати випробувань моделі нейронної мережі з двома прихованими шарами і дропаутом

Показник захворюваності	*MAE*
Число нових випадків туберкульозу	22,3047895132
Число випадків захворювань системи кровообігу	1620,67692139
Число всіх зареєстрованих випадків раку	272,465711234

Для перевірки можливості використання мереж такого типу для вирішення задачі було створено мережу, схема якої представлена на рисунку 4.34:

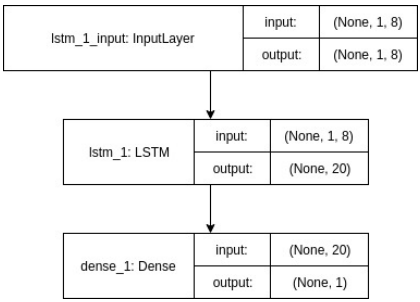


Рисунок 4.34 – Схема нейронної мережі з використанням LSTM шару (що містить 20 вузлів) і один нейрон на вихідному шарі (в нотації Keras)

Мережа отримує на вхід вісім параметрів – данні за попередній період: показник захворюваності, обсяг викидів забруднюючих речовин в атмосферне повітря від стаціонарних джерел, кількість населення, середній вік населення, середню температуру і середню кількість опадів, кількість лікарів в регіоні, кількість ліжок у стаціонарних відділеннях закладів охорони здоров'я у регіоні. Прихований LSTM шар складається з двадцяти нейронів, а вихідний шар з одного нейрону. У якості алгоритму оптимізації використано алгоритм adam, у якості метрики моделі – *MAE*.

Для навчання моделі використано вибірку по одному з регіонів, а для випробування моделі – по іншому. Динаміка зміни значень вхідних параметрів (середня по всій Україні) має наступний вигляд (рис. 4.35).

Для використання дані було перетворено на відповідний датсет придатний для навчання LSTM мережі з вчителем з часовим кроком, який дорівнює одиниці, а значення параметрів стандартизовано. Перші п'ять рядків вибірки, таким чином будуть мати вигляд табл. 4.10.

Таблиця 4.10 – Перші п'ять рядків вибірки з часовим шагом 1

	var1(t-1)	var2(t-1)	var3(t-1)	var4(t-1)	var5(t-1)	var6(t-1)	var7(t-1)	var8(t-1)	var1(t)
1	0,065574	1,000000	0,92861	0,086957	0,714286	0,296046	0,772079	0,000000	0,049180
2	0,049180	0,615404	0,92861	0,173913	0,591837	0,307781	1,000000	0,000000	0,065574
3	0,065574	0,692298	0,92861	0,173913	0,714286	0,288010	0,764025	0,000000	0,016393
4	0,016339	1,000000	0,92861	0,304348	0,489796	0,181888	0,623681	0,000000	0,000000
5	0,000000	0,846211	0,92861	0,673913	0,408163	0,206888	0,661081	0,000000	0,000000

Стовпчики (t-1) – значення параметрів на попередньому часовому кроці, а значення (t)–передбачення показника захворюваності на наступному часовому кроці.

Навчання проходило впродовж двохсот епох з розміром підвибірки (batch_size), що дорівнював 50.

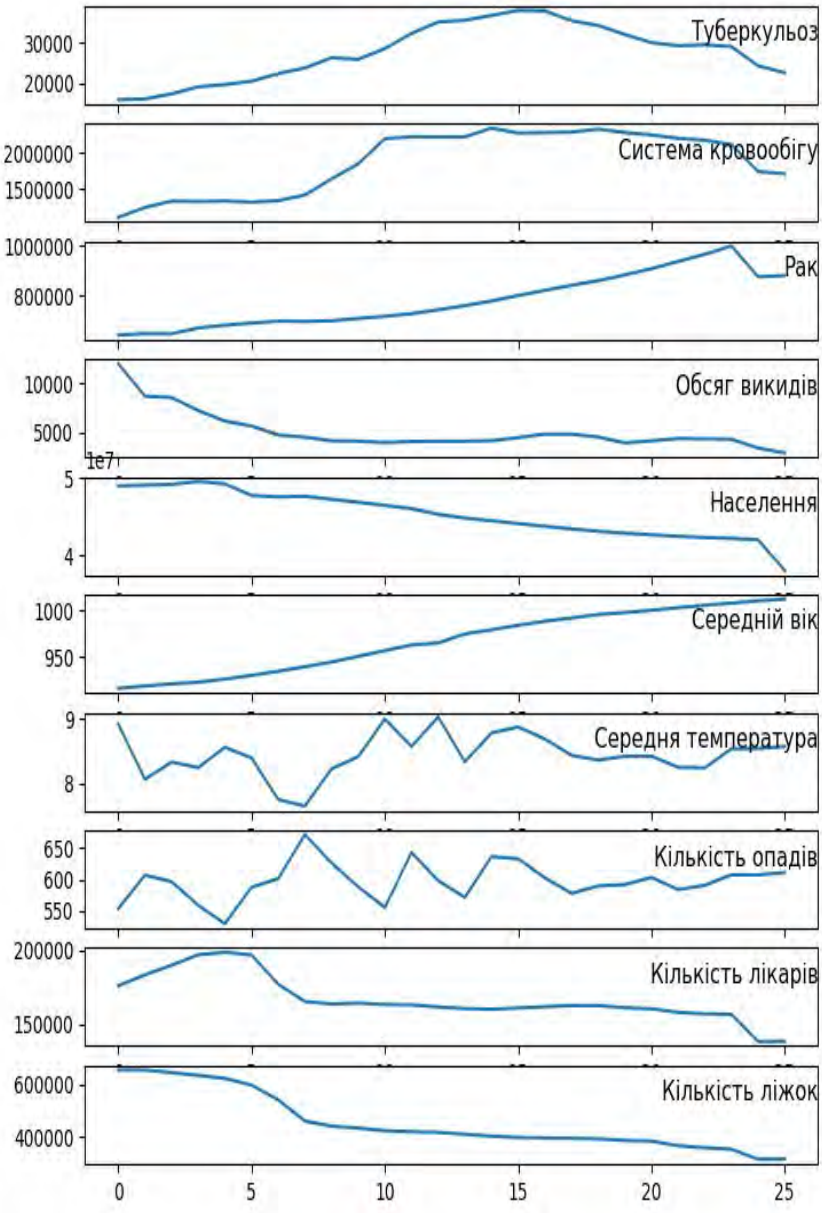


Рисунок 4.35 – Зміна параметрів у часі

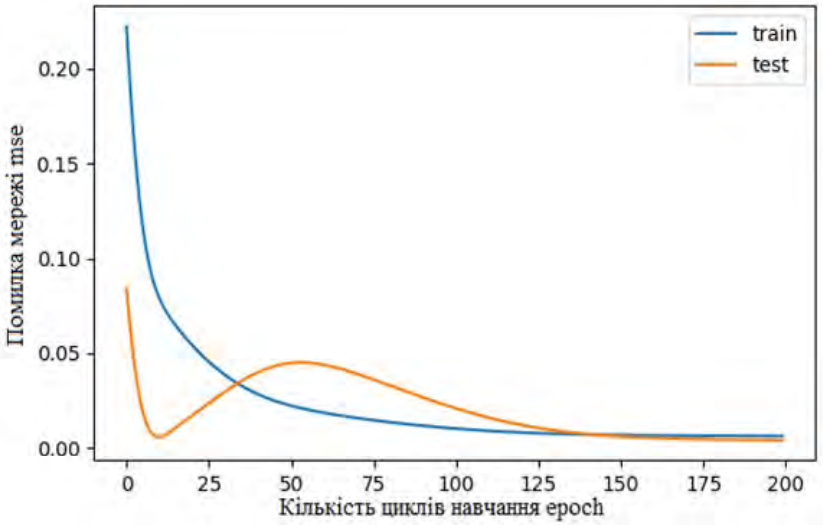


Рисунок 4.36 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість нових випадків туберкульозу»

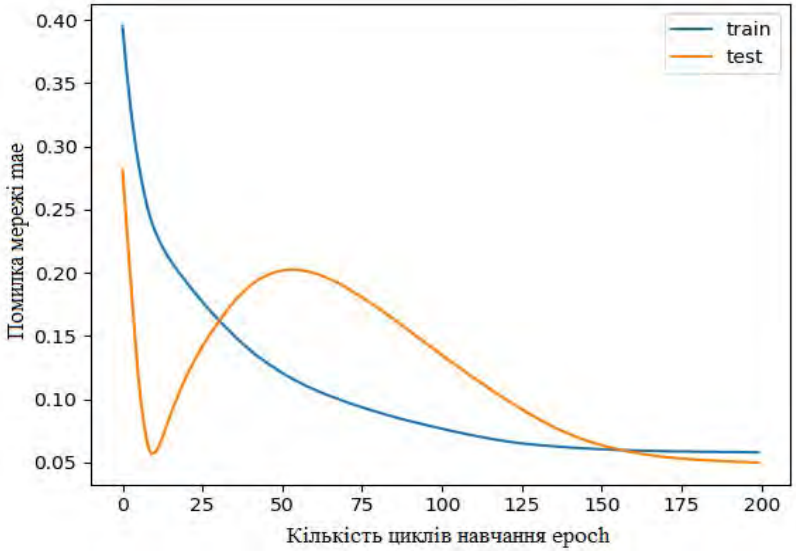


Рисунок 4.37 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість нових випадків туберкульозу»

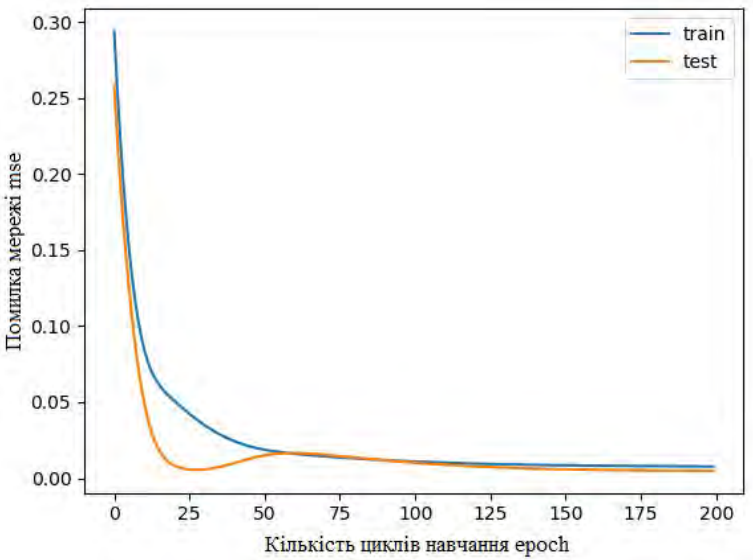


Рисунок 4.38 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість випадків захворювань системи кровообігу»

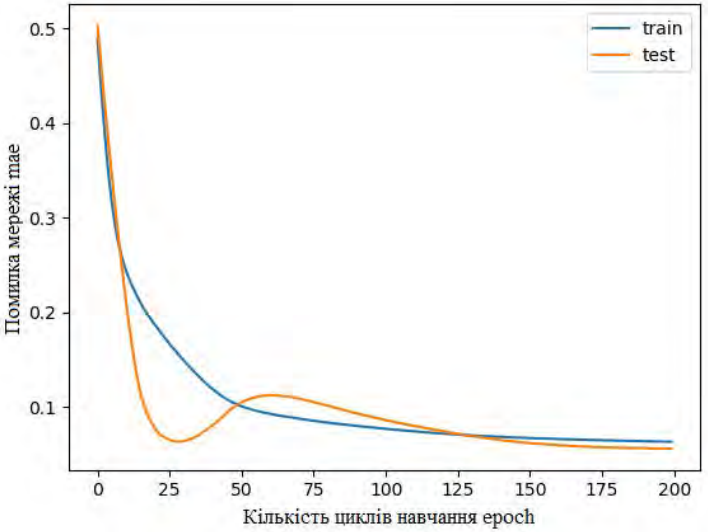


Рисунок 4.39 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість випадків захворювань системи кровообігу»

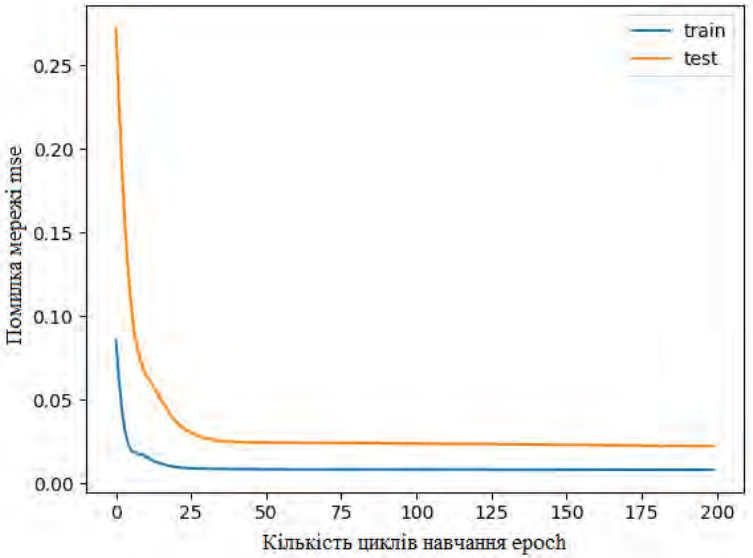


Рисунок 4.40 – Графік залежності помилки мережі mse від кількості циклів навчання epoch для моделі показника «Кількість всіх зареєстрованих випадків раку»

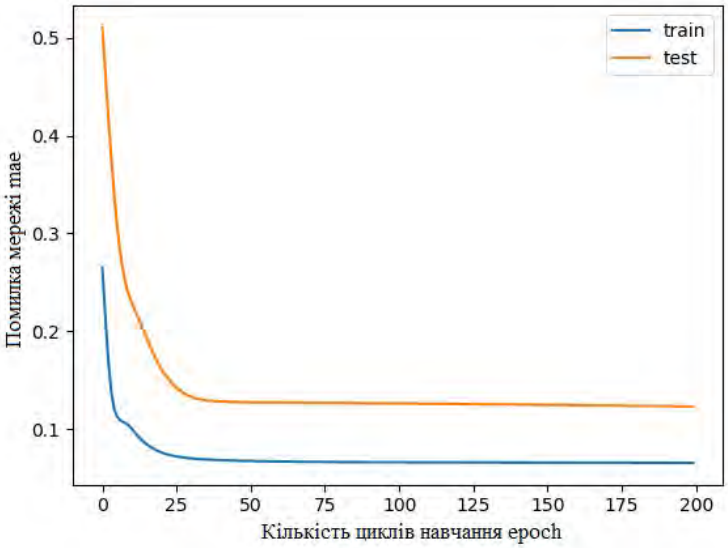


Рисунок 4.41 – Графік залежності помилки мережі mae від кількості циклів навчання epoch для моделі показника «Кількість всіх зареєстрованих випадків раку»

На рисунках 4.36 – 4.41 відображено зміну значення помилки мережі впродовж тренування. Для показників «Кількість нових випадків туберкульозу» та «Кількість випадків захворювань системи кровообігу» спостерігається досягнення локального мінімуму с подальшим виходом з нього. Під кінець тренування спостерігається відсутність подальшого зменшення значення помилки моделі, тому можна вважати, що під час тренування досягнуто глобального мінімуму помилки, а мереже вважається тренованою.

Результати випробувань моделі наведені у таблиці 4.11.

Таблиця 4.11–Результати випробування моделі на основі LSTMNN

Показник захворюваності	*RMSE*	*MAE*
Число нових випадків туберкульозу	8,168	6,139
Число випадків захворювань системи кровообігу	583,357	441,889
Число всіх зареєстрованих випадків раку	296,234	226,096

Таким чином, можна побачити, що найменшу помилку передбачення з наведених методів дає модель на основі штучної нейронної мережі короткої довгочасної пам’яті.

Значення середньої абсолютної помилки під час випробувань моделей створених в ході дослідження проблеми наведені у таблиці 4.12.

4.3 Синтез нейромережевих моделей показників захворюваності населення

Зазвичай, архітектура моделі нейронної мережі, її топологія, значення макропараметрів обирається на основі експертної оцінки або емпірично. Для мереж довгої короткочасної пам’яті такими параметрами можуть бути кількість вузлів шару довгої короткочасної пам’яті, оптимізатор, розмір підвибірки і кількість епох навчання.

Для підбору оптимальних значень цих параметрів можуть бути використані такі стохастичні методи, як метод рою часток і генетичні алгоритми.

Таблиця 4.12 – Значення середньої абсолютної помилки під час випробувань моделей створених в ході дослідження проблеми

Методи	Прогнозування кількості нових випадків туберкульозу	Прогнозування кількості випадків захворювань системи кровообігу	Прогнозування кількості всіх зареєстрованих випадків раку
Мережа довгої короткочасної пам'яті з 50 вузлами ДКЧП шару	6,139	441,889	226,0963791
Метод найближчого сусіда	8,236485	573,004327	210,157262
Випадковий ліс	7,671577	571,018151	156,387862
Метод найменших квадратів	22,957569	1789,727716	367,381240
Багатошаровий перцептрон з трьома прихованими шарами і дропаутами (128, дропаут (0,5), 1024, дропаут (0,5), 128 нейронів)	22,3047895132	1620,67692139	272,465711234
Метод опорних векторів	40,270765	2794,43333	1050,690702
Багатошаровий перцептрон з трьома прихованими шарами (128, 1024 і 128 нейронів)	21,6752039937	1752,4164209	338,953238874
Багатошаровий перцептрон з одним прихованим шаром (128 нейронів)	23,4455625425	1766,47043614	336,419792045
Логістична регресія	29,764103	3814,174359	1400,357372

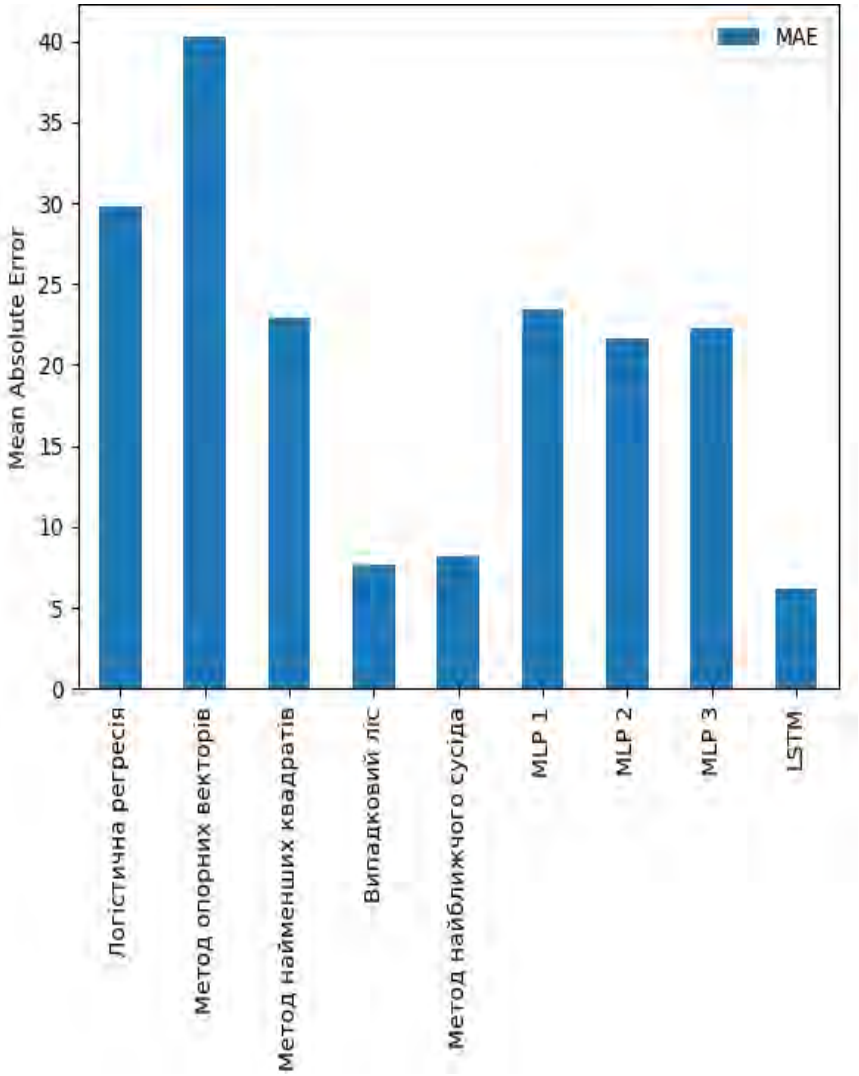


Рисунок 4.42 – Порівняння значень середньої абсолютної помилки досліджених моделей для показника «Кількість нових випадків туберкульозу»:
MLP 1 - багатошаровий перцептрон з одним прихованим шаром (12 нейронів),
MLP 2 - багатошаровий перцептрон з двома прихованими шарами (12, 12 нейронів),
MLP 3 - багатошаровий перцептрон з двома прихованими шарами і дропаутом (12, дропаут (0,5), 12 нейронів), LSTM - мережа довгої короткочасної пам'яті з 20 вузлами ДКЧП шару

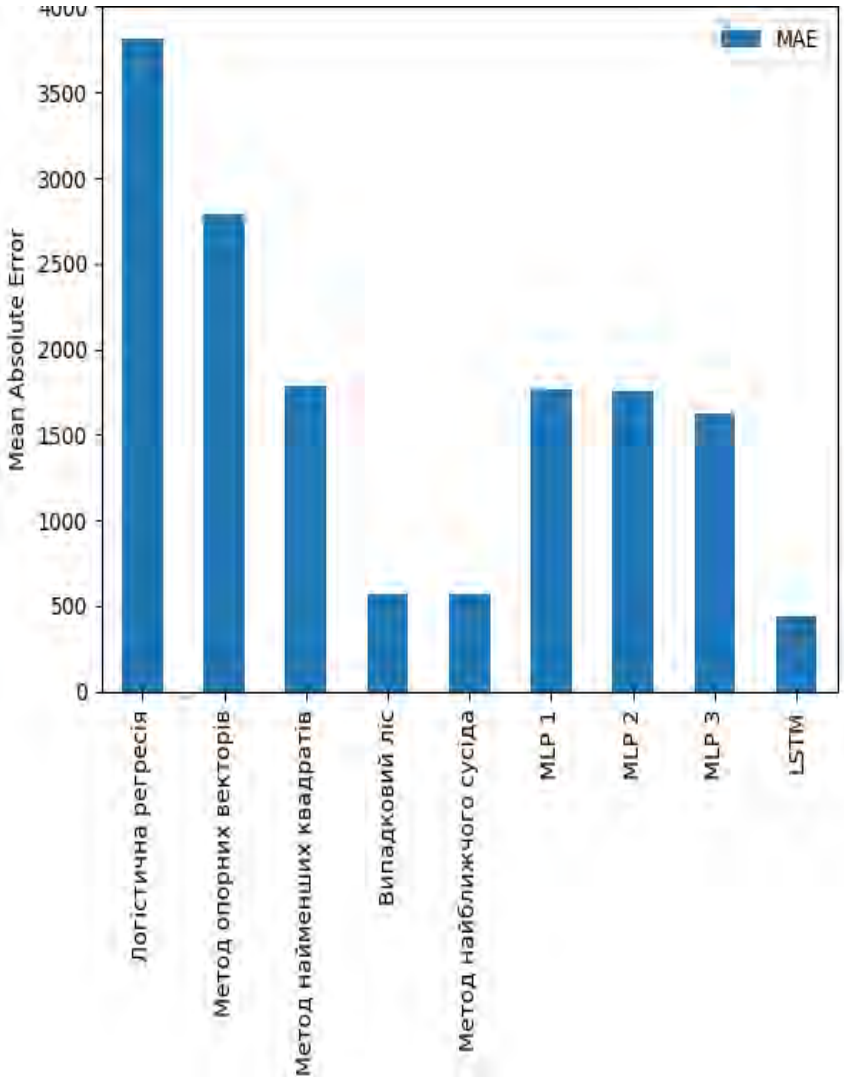


Рисунок 4.43 – Порівняння значень середньої абсолютної помилки досліджених моделей для показника «Кількість випадків захворювань системи кровообігу»: MLP 1 - багатошаровий перцептрон з одним прихованим шаром (12 нейронів), MLP 2 - багатошаровий перцептрон з двома прихованими шарами (12, 12 нейронів), MLP 3 - багатошаровий перцептрон з двома прихованими шарами і дропаутом (12, дропаут (0,5), 12 нейронів), LSTM - мережа довгої короткочасної пам'яті з 20 вузлами ДКЧП шару

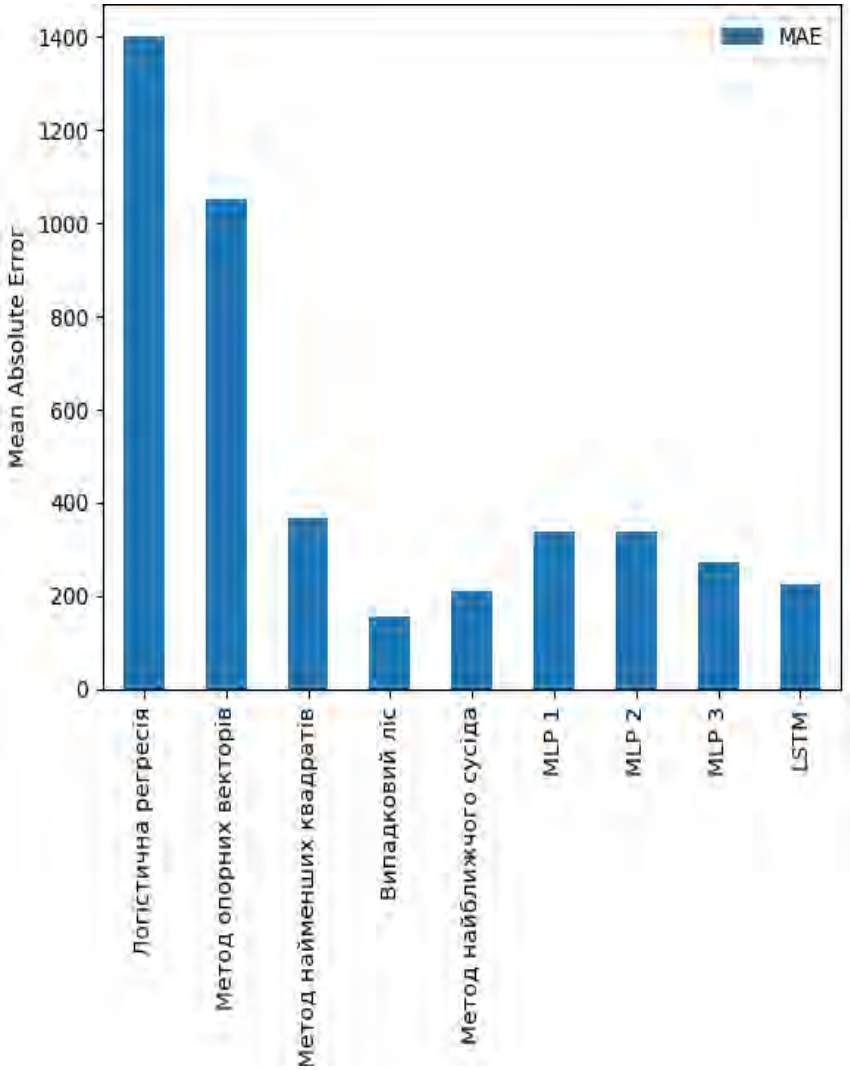


Рисунок 4.44 – Порівняння значень середньої абсолютної помилки досліджених моделей для показника «Кількість всіх зареєстрованих випадків раку»: MLP 1 - багатошаровий перцептрон з одним прихованим шаром (12 нейронів), MLP 2 - багатошаровий перцептрон з двома прихованими шарами (12, 12 нейронів), MLP 3 - багатошаровий перцептрон з двома прихованими шарами і дропаутом (12, дропаут (0,5), 12 нейронів), LSTM - мережа довгої короткочасної пам'яті з 20 вузлами ДКЧП шару

У ході роботи для оптимізації мережі довгої короткочасної пам’яті було використано метод рою часток. Було створено рій з 20 часток, що рухався у чотирехмерному просторі: кількість вузлів мережі довгої короткочасної пам’яті (область визначення від 50 до 1000), оптимізатор, розмір підвибірки (область визначення від 1 до 26) та кількість епох тренування (область визначення від 100 до 1000). Фітнес функцією частки було значення *RMSE* для тренованої мережі довгої короткочасної пам’яті з визначеними параметрами на тестовій вибірці. Виконання алгоритму було припинене на 52 ітерації у зв’язку з тим, що рій зупинив свій рух. Результати виконання алгоритму (визначення найменшого значення помилки мережі на кожній ітерації алгоритму) відображено на рисунку 4.45.

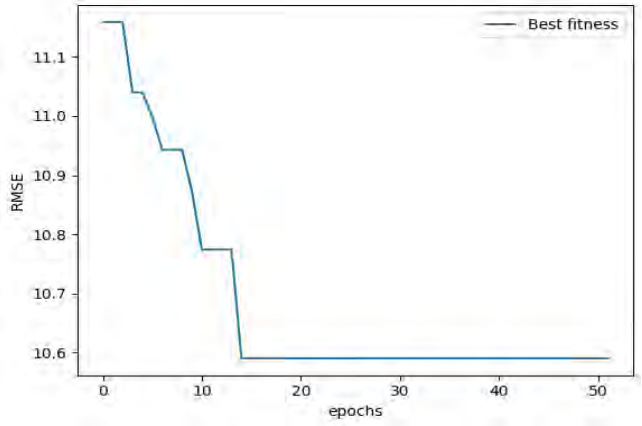


Рисунок 4.45 – Результат роботи алгоритму рою часток для оптимізації мережі довгої короткочасної пам’яті

В результаті оптимізації були отримані такі оптимальні параметри мережі:

- кількість вузлів мережі – 1000;
- оптимізатор – Adadelta;
- розмір підвибірки – 1;
- кількість епох тренування – 100.

Мережа довгої короткочасної пам’яті з такими параметрами на тестовій вибірці має значення помилки (*RMSE*) – 10,59061275654188.

Для підбору оптимальних значень цих параметрів можуть бути використані генетичний алгоритм (GA) [40-44]. Розроблено модифікацію генетичного алгоритму.

Однією з проблем, які виникають під час використання генетичних методів, є знаходження локальних екстремумів замість глобального, що в деяких випадках призводить до неприйнятних результатів. Для вирішення цієї проблеми існує багато модифікацій генетичного методу (острівна модель, елетизм, тощо).

Слід зауважити, що, хоча це і є спробою змоделювати діючи у природі процеси еволюції, метод побудовані на дещо спрощеному уявлені про основні елементи теорії спадковості і сучасної генетики.

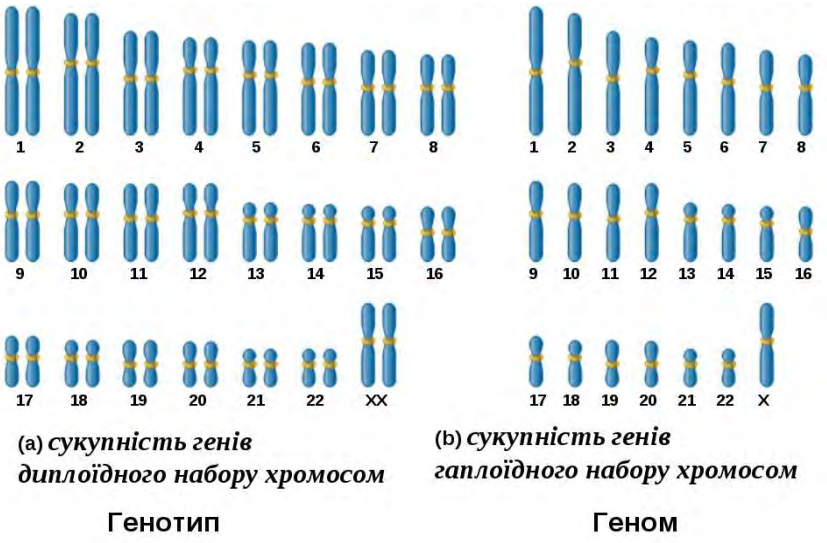


Рисунок 4.46 - Диплоїдний та гаплоїдний набір хромосом [33 - 39]

Класичний метод передбачає наявність в особини однієї єдиної хромосоми, в якій у вигляді генів закодовані параметри, що оптимізуються. В біології такий набір хромосом називають гаплоїдним (рис. 4.46). Зазвичай, гаплоїдний набір хромосом мають лише статеві клітини (тобто ті, що приймають участь у розмноженні) – гамети. Всі інші клітини мають диплоїдний набір хромосом і містять пари хромосом. Хромосоми, що утворюють пару називаються

гомологічними. Гени в хромосомах однієї пари називаються алельними і можуть відрізнятися, однак зовнішню прояву буде мати лише один з них. У випадку, коли гени розрізняються, той ген, що має зовнішню прояву, має назву домінантного, інший рецесивного. Тобто, за проявлення тієї чи іншої ознаки у фенотипі особини відповідає пара генів.

Таким чином, особина може утворювати гамети, яки будуть нести в собі одну з хромосом пари і, відповідно, генний склад гамет може розрізнятися. До того ж, згідно закону чистоти гамет [45 - 48], гамети є «чистими»: вони несуть лише один з двох алелей, певного гену.

При використанні однієї хромосоми частина генів особин популяції (які були у батьків, але відсутні у їх нащадків) не приймають участь у подальшому процесі, що врешті приводить до зменшенню варіабельності генофонду популяції і зменшенню дисперсії ознак фенотипів особин кожного покоління на рівні популяції. Однак, в даному випадку існує імовірність заняття популяцією екологічної ніші, що відповідає локальному екстремуму функції, що оптимізується. При застосуванні генетичного методу для популяцій малої кількості (наприклад, мікро - генетичні алгоритми) такий стан системи може настати досить швидко. Створюючи аналогію з біологічними процесами, можна сказати, що в такому випадку популяція буде знаходитися під дією стабілізуючого природного відбору (рис. 4.47). В даному випадку, збереження якомога більшої кількості генів у генофонді популяції може створювати потенціал для подолання локального екстремуму функції, що оптимізується.

Сутність запропонованої оптимізації класичного методу полягає в додаванні до каріотипу кожної особини ще однієї хромосоми з таким самим генним складом, тобто використовувати диплоїдний набір, що складається з двох гомологічних хромосом. Обидві хромосоми піддаються дії ти ж самих операторів з однаковими параметрами. Таким чином, при схрещуванні каріотип нащадка буде також складатися з двох гомологічних хромосом, як і у його батьків. Домінуючий ген в запропонованій модифікації обирається випадковим чином з двох алельних генів і використовується для обчислення значення функції пристосованості – фітнес функції, тобто, говорячи в термінах біології, визначає фенотип особини.

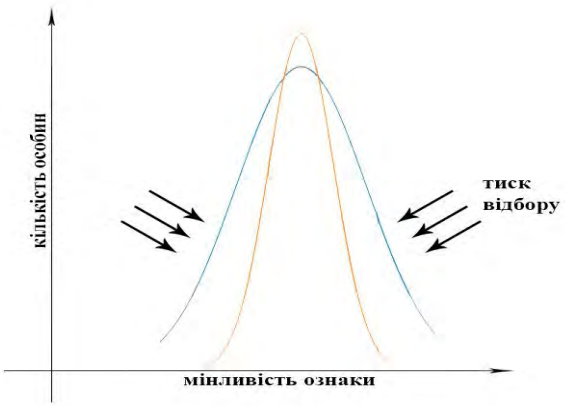


Рисунок 4.47 – Стабілізуючий відбір проявляється в постійних умовах довкілля. Він підтримує сталість певного фенотипу, якій найбільш відповідає середовищу, і відкидає будь-які менш пристосувальні зміни. Цим він звужує межі модифікаційної мінливості, тобто норму реакції.

На першому етапі відбувається ініціалізація популяції. Генний склад кожної з двох гомологічних (H,H') хромосом особин обирається випадковим чином. Для визначення фенотипу особини з кожної алелі обирається ген, що буде визначатися як домінантний і буде визначати фенотип особини, тобто приймає участь в обчисленні функції пристосованості особини. Визначення фенотипу особини можна представити у вигляді формули (4.14) [49]:

$$F_j = \sum_{i=1}^{m} rand\left[H_j g_i ; H_j^{'} g_i\right], \qquad (4.14)$$

де F_j - фенотип j-ї особини; m – кількість алелей в парі хромосом; $H_j g_i$ та $H_j^{'} g_i$ – i-й ген в парі гомологічних хромосом j-ї особини.

Таким чином, фактично визначаються аргументи фітнес-функції особини. Після обчислення функцій пристосованості і відбору особин популяції проводиться схрещування. Генотип особини-нащадка має таку ж структуру, як і генотип батьків, тобто складається з двох гомологічних хромосом. До нащадків застосовується оператор

мутацій. При цьому, мутувати може будь-яка алель пари гомологічної хромосом, але в кожній алелі мутує лише один ген.

Далі, як і в класичному методі, цикл повторюється до настання умов закінчення виконання оптимізації.

Підсумовуючи, можна сказати, що запропонований метод відрізняється від класичного генетичного методу використанням не однієї хромосоми, а пари гомологічних хромосом, і додаванням етапу визначення тих генів алелі, яки будуть приймати участь у визначенні значення функції пристосованості особини. Результатом такої модифікації є підтримання досить високої варіабельності ознак (генів) в популяції (генофонду популяції) під час еволюції, яка, в той же час, може мати невеликий вплив на фенотип особин.

Зазначену модифікацію методу було використано для оптимізації LSTM нейронної мережі: кількості вузлів мережі, функції оптимізації при навчанні, розміру підвибірки та кількості епох навчання.

Початкова популяція складалася з 20 осіб, еволюція популяції відбувалася впродовж 100 епох. Під час виконання генетичного алгоритму використовувалися наступні оператори:

- мутації (з імовірністю 0,3);
- турнірний відбір (розмір групи 3).

У якості значення фітнес-функції використано показник помилки мережі RMSE на тестовій виборці. Результат виконання модифікованого генетичного алгоритму для показника «Кількість нових випадків туберкульозу» алгоритму наведено на рисунку 4.48:

На діаграмі – сині кулі відповідають кращим значенням фітнес функції (найменшому значенню RMSE) на кожній ітрації, чорна лінія – апроксимація цього розподілення поліномом другого ступеня.

На рисунку 4.49 наведено діаграму варіабельності значень фітнес функції популяції на кожній ітерації методу.

На рисунках 4.50 та 4.51 наведені дані по кращим значенням фітнес функції і варіабельності значень фітнес функції при виконанні оптимізації класичним методом за тих умов.

Таким чином, виходячи з отриманих результатів, можна стверджувати, що при використанні модифікованого методу популяція зберігає високу варіабельність генофонду впродовж еволюції, що створює потенціал для подолання локальних екстремумів функції, що оптимізується.

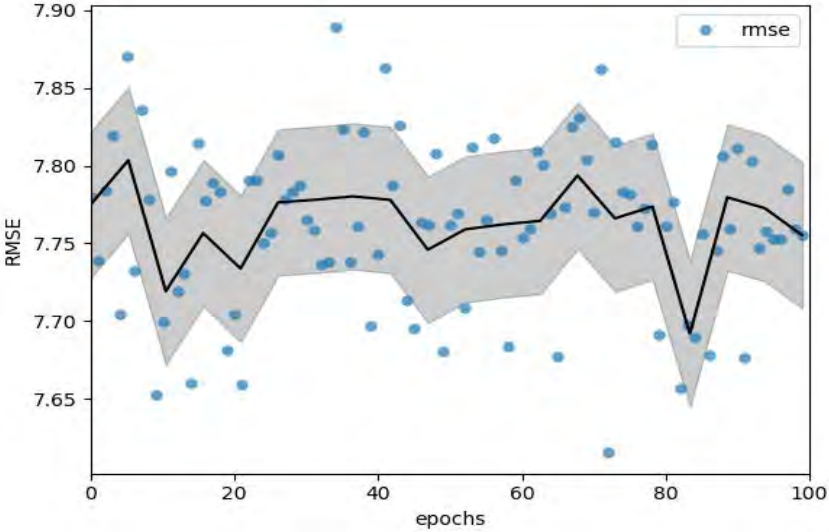


Рисунок 4.48 – Еволюція параметрів LSTM мережі (для показника «Кількість нових випадків туберкульозу») з застосуванням модифікованого методу

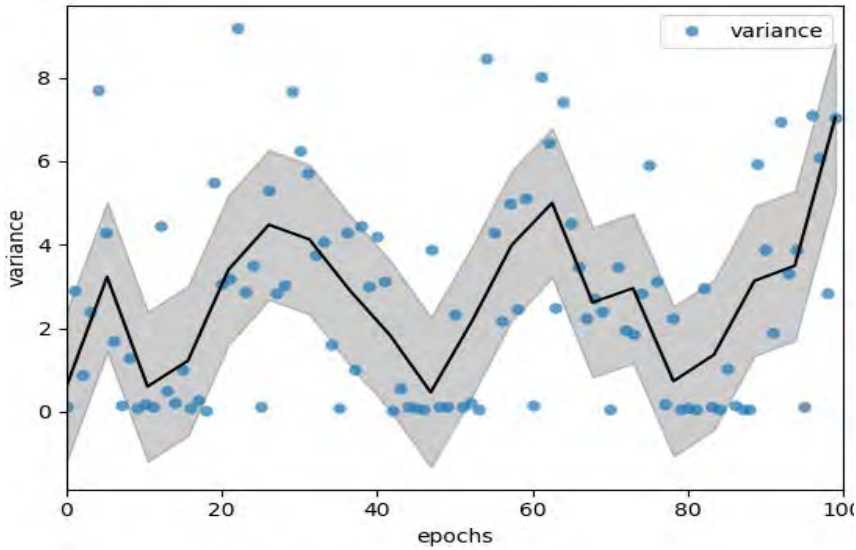


Рисунок 4.49 – Варіабельність значень фітнес функції впродовж еволюції популяції (для показника «Кількість нових випадків туберкульозу») при використанні модифікованого методу

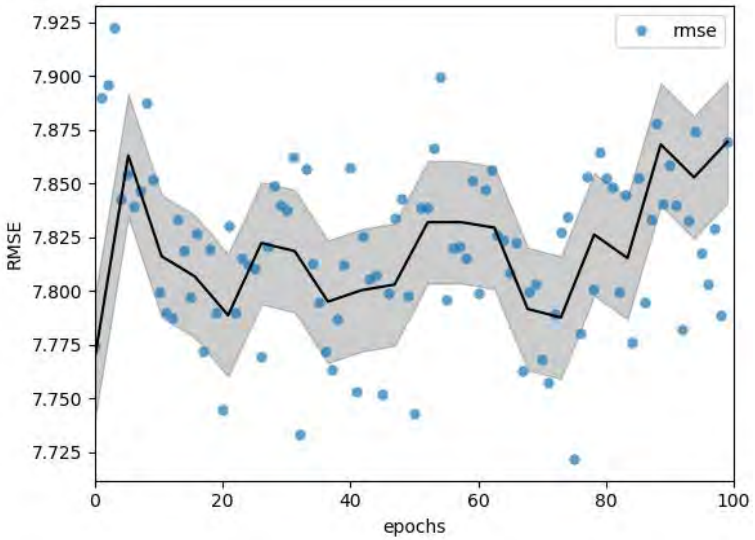


Рисунок 4.50 – Еволюція параметрів LSTM мережі (для показника «Кількість нових випадків туберкульозу») з застосуванням класичного методу

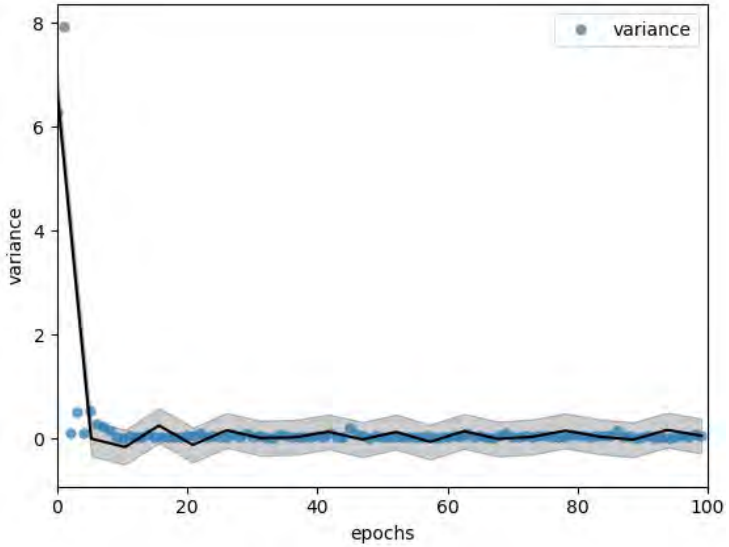


Рисунок 4.51 – Варіабельність значень фітнес функції впродовж еволюції популяції (для показника «Кількість нових випадків туберкульозу») при використанні модифікованого методу

Фактично, популяція зберігає в своєму генофонді більшу кількість генів (і їх комбінацій), ніж при використанні класичного методу. Крім цього, в результаті оптимізації з використанням модифікованого метода була отримана мережа з меншим значенням помилки, ніж при використанні класичного методу.

В таблиці 4.13 відображені результати оптимізації LSTM мережі модифікованим і класичним методом. Так, кращі значення були отримані на 72 і 75 ітераціях, відповідно, і склали: 7,615534676449391 – при використанні модифікованого методу і 7,721700621750585 – при використанні класичного методу. Крім цього, у випадку модифікованого методу варіабельність на цій ітерації значно перевіщує цей показник, що було отримано при використанні класичного методу.

Таблиця 4.13 – Кращі значення фітнес функції (найменше значення RMSE) і умови її отримання при виконанні оптимізації модифікованим і класичним методом

	Модифікований метод	Класичний метод
Епоха	72	75
Краще значення фітнес функції (RMSE)	7,615534676449391	7,721700621750585
Варіабельность значень фітнес функції	1,9381020866015326	0,053462212203755674
Кількість вузлів LSTM	20	20
Оптимізатор	SGD	SGD
Розмір підвибірки	31	13
Кількість епох тренування	550	786

В обох випадках отримані однакові значення кількості вузлів LSTM мережі (20 вузлів) і оптимізатор, що застосовується при тренуванні отриманої мережі (простий градієнтний спуск). Різниця полягає в параметрах иренування отриманої мережі, а саме, розмірі підвибірки (31 і 13 для модифікованого і класичного методів

відповідно) і у кількості епох тренування – кращі результати були отримані при тренуванні мережі впродовж 550 епох.

Слід зауважити, шо запропонована модифікація алгоритму має більшу оцінку складності алгоритму. Це пов'язано з додаванням ще однієї стадії, а саме, стадії визначення домінантних генів особин. Часова складність цієї стадії дорівнює O(2n) і залежить від кількості особин, які приймають участь у схрещуванні. Також вдвічі збільшується оцінка по пам'яті, що також пов'язано зі збільшенням кількості хромосом популяції вдвічі.

Іншою запропонованою модифікацією генетичного методу є модифікація оператора мутацій. На відміну від класичного застосування цього оператору, коли мутації піддаються усі особини генерації з певною імовірністю, пропонується ввести поняття мутаційної стійкості особини і застосовувати оператор тільки для особин з найнижчим значенням цього показника.

Обчислене значення функції пристосованості особини (фітнес функції) може бути інтерпретоване як значення мутаційної стійкості особини. Отже, пропонується на кожній ітерації методу після обчислення функції пристосованості проводити ранжування особин отриманої генерації за значенням мутаційної стійкості. На відміну від класичного оператора, на початку вказується не вірогідність мутації, а частка особин, які піддаються дії оператора. Так, наприклад, якщо отримано 20 особин нової генерації і означений параметр дорівнює 0,2, то оператор буде застосовано до 4 особин, що мають найменшу стійкість до мутацій (фактично, найменше значення фітнес функції):

$$K_{mut} = H_{hen} R_{mut}, \tag{4.15}$$

де K_{mut} – кількість особин, що піддаються дії мутації; H_{hen} – кількість особин отриманої генерації; R_{mut} – частка особин генерації, що піддаються дії мутації.

Фактично, пропонується застосовувати оператор тільки до особин з найнижчим значенням функції пристосованості. У цьому випадку, у разі потрапляння популяції в локальний екстремум функції, застосований оператор мутацій має дозволить проводити пошук виходу з нього не змінюючи отриманих найкращих на момент

застосування значень, а тільки за рахунок більш слабких особин. Визначена частка особин, що піддаються дії оператора, повинна бути достатньою для забезпечення створення потенціалу для подальшої еволюції усієї популяції.

Такі мутації повинні бути більш «м'якими» у сенсі збереження знайдених на попередніх ітераціях алгоритму найкращих значень і мають нівелювати небезпеку втрати екстремуму функції при їх застосуванні не зупиняючи пошук нових кращих значень.

Графік еволюції параметрів мережі і параметрів тренування відображений на рисунку 4.52.

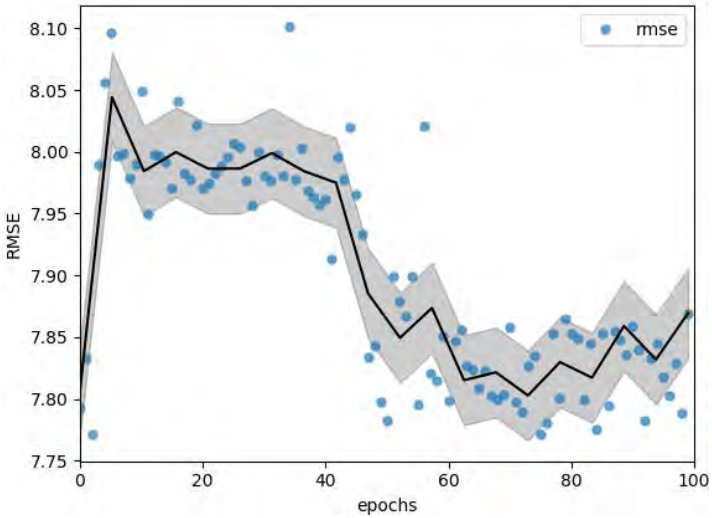


Рисунок 4.52 - Еволюція параметрів LSTM мережі (для показника «Кількість нових випадків туберкульозу») з застосуванням модифікованого оператору мутацій

На графіку визначено перехід популяції до нового мінімуму функції впродовж 40 – 50 епох еволюції. Отримане таким чином краще значення фітнес функції дорівнює 7,771700621750585 (75 епоха).

Отримане значення помилки мережі в даному випадку буде нижче ніж при використанні класичного методу, але вишче ніж при використанні методу з використанням диплоїдного набіру хромосом.

Порівнюючи алгоритмічну складність використання модифіцированого оператору мутацій з класичним треба зауважити,

що використання модифікованого опературу підвіщує складність алгоритму через додавання стадії сортування популяції по значенню мутаційної стійкості. Так, наприклад, у разі використання алгоритму сортування вставками складність цієї стадії складатиме $O(n^2)$.

Для нейронної мережі LSTM, що має 20 вузлів і тренована у відповідності до отриманих в результаті оптимізації параметрів (таб. 3.13) обчислено значення *MAE*, що склало 5,931 і є кращим з усіх досліджених під час виконання роботи моделей.

Таким чином у якості моделі, що найбільш точно відповідає наявній залежності між рівнем захворюваності населення і обсягом викидів забруднюючих речовин, можна використовувати модель на основі штучної нейронної мережі довгої короткочасної пам'яті. Для підбору таких параметрів, як кількість вузлів LSTM шару, методу оптимізації мережі, розміру підвибірки і кількості епох навчання мережі бажаним є використання модифікованого генетичного методу.

Отрмана таким чином модель є мережею довгої короткочасної пам'яті, що має вхідний шар з кількістю нейронів, що відповідає кількості вхідних параметрів, шар LSTM, що має 20 вузлів і вихідний шар з одним нейроном (таким чином, мережа має 2341 параметри, що можуть бути треновані). У якості функції активації нейронів вхідного шару і шару LSTM використано ReLU. Тренування мережі проходило впродовж 550 епох; у якості оптимізатора використано простий градієнтний спуск; у якості помилки використано *MAE*.

Тренована таким чином мережа має значення помилки *MAE* на тестовій вибірці – 5,931, що є кращим результатом з усіх досліджених моделей.

4.4 Програмне забезпечення для прогнозування залежності захворюваності від обсягів викидів

Для розробки і тестування моделі залежності захворюваності від обсягів викидів використано статистичну інформацію про обсяги викидів забруднюючих речовин та діоксиду вуглецю в атмосферне повітря від стаціонарних джерел забруднення і інформацію про рівень захворюваності по таким показникам, як кількість випадків захворювань системи кровообігу (зареєстровані в амбулаторних установах), кількість нових випадків туберкульозу і кількість

зареєстрованих випадків раку. Період спостережень – з 1990 року по 2015 рік з розбивкою по роках і регіонах України.

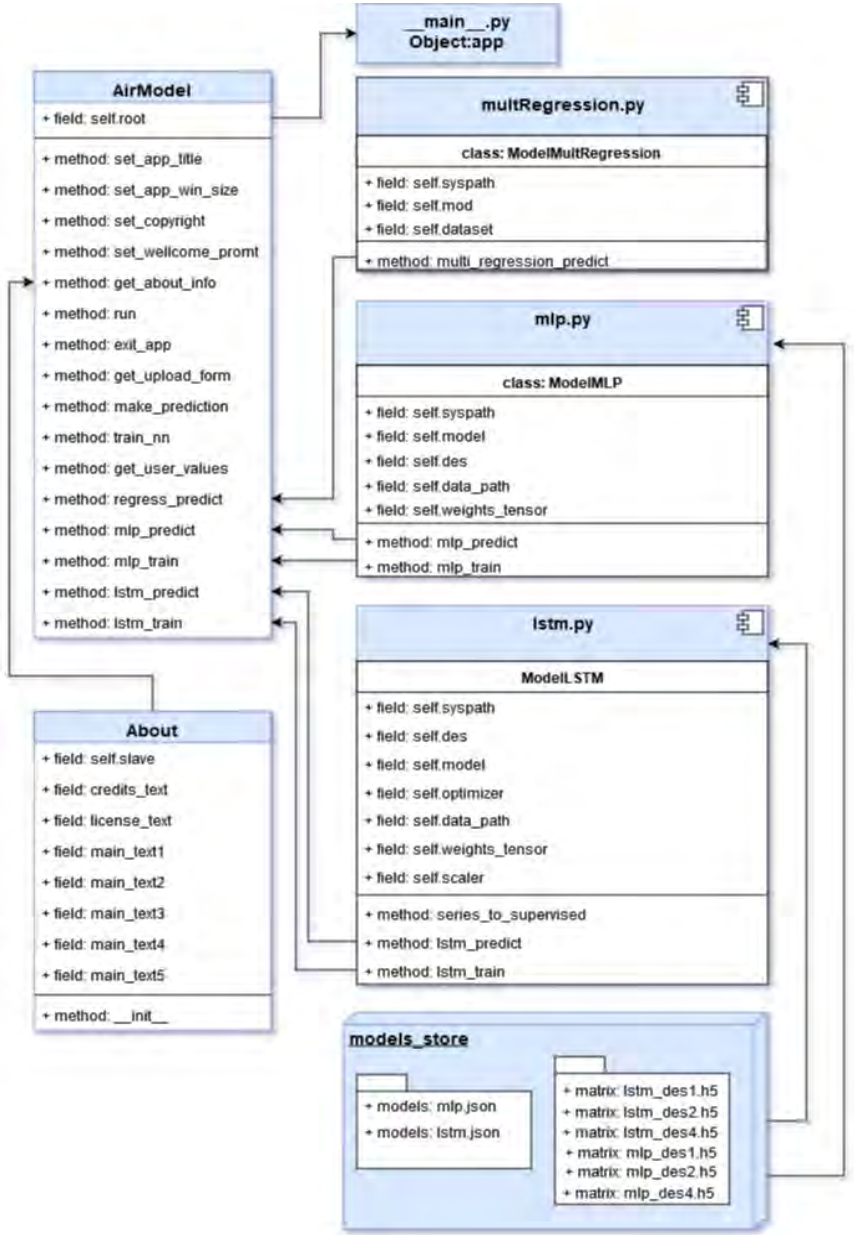
Враховуючи той факт, що наведені дані виражені в абсолютних значеннях, доцільно зробити поправку на кількість населення у регіоні. Тому використано також дані про кількість населення у регіонах по роках.

Крім цього, доведено, що сила і характер впливу забруднення повітря на рівень здоров'я населення може корелювати в залежності від метеорологічних (кліматичних) і медично-соціальних умов регіону. Зазвичай, гостра дія викидів в атмосферне повітря провокується різкою зміною погодних умов на даній території, наприклад, температурна інверсія, штиль, туман, сильний стійкий вітер з боку промислової зони. Максимальна концентрація летких токсичних речовин у повітрі залежить від температури повітря і, як правило, зменшується при зниженні останньої. При високій вологості повітря краплі вологи зважені у повітрі можуть абсорбувати токсичні речовини и розчинювати їх у собі. Тому для побудови більш точної моделі також використано статистичні дані про середню температури в регіоні і рівень опадів, кількість лікарів у регіоні, кількість ліжок у стаціонарних відділеннях закладів охорони здоров'я регіону і середній вік населення у регіоні.

Розроблене програмне забезпечення являє собою пакет програм для моделювання залежності показників захворюваності від обсягів викидів забруднюючих речовин в атмосферне повітря, до складу якого входять набір скриптів, що можуть бути використані для обчислення прогнозів або створення власних моделей для вирішення споріднених задач, та окремий (віконний) додаток, за допомогою якого оператор може обчислювати прогноз захворюваності за допомогою різних моделей та тренувати моделі на основі багатошарового перцептрону і мережі довгої короткочасної пам'яті з використанням завантажених вибірок.

Діаграма класів розробленого додатку відображена на рисунку 4.52, на рисуноку 4.53 зображена діаграма співвідношень

Файл __main__.py - головний файл проєкту, в якому відбувається ініціалізація додатку, створюється об'єкт основного класу додатку class: AirModel і, який запускає головний цикл додатку. Клас AirModel має такі методи:



.

Рисунок 4.52 – Структурна схема прграми

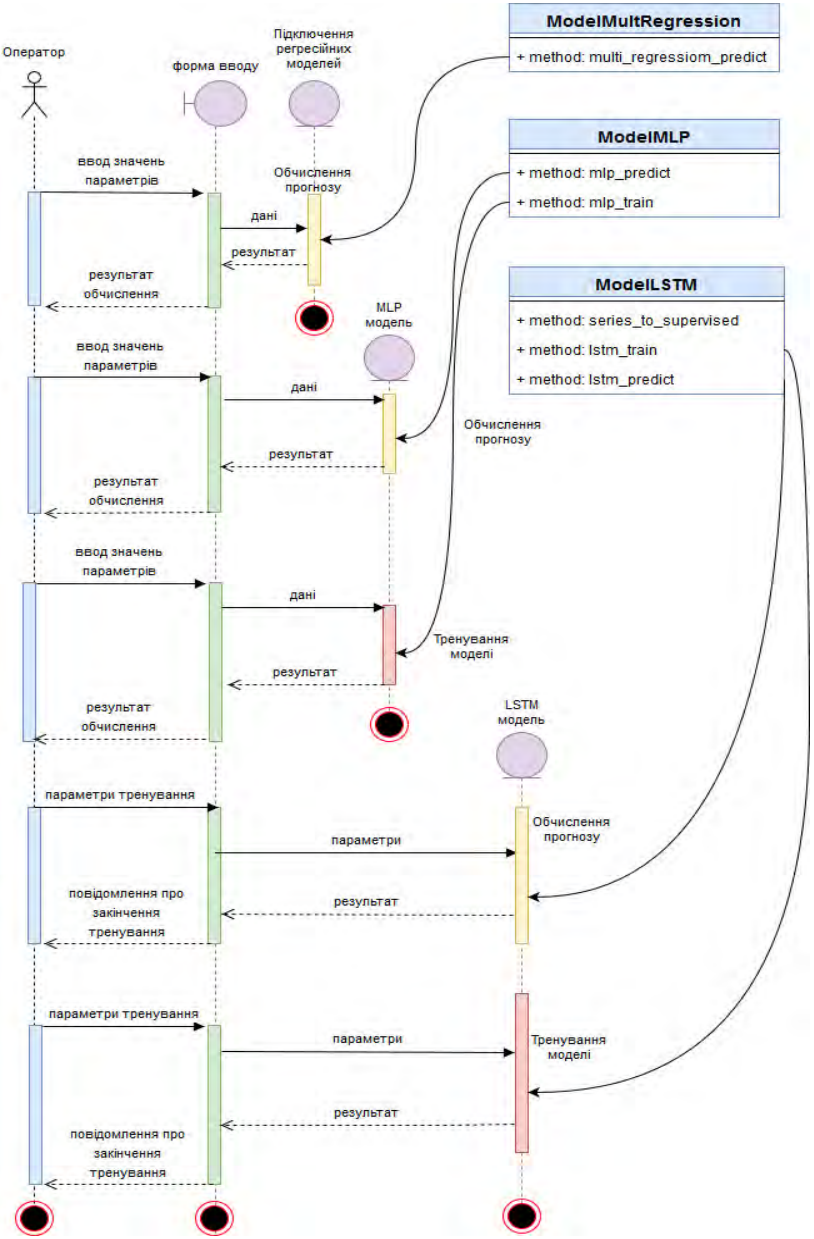


Рисунок 4.53 – Діаграма співвідношень

– __init__ - конструктор класу. В конструкторі відбувається ініціалізація змінних класу і елементів інтерфейсу користувача;

– set_app_title - допоміжний метод, що встановлює заголовок головного вікна додатку;

– set_app_win_size - допоміжний метод, що встановлює розмір головного вікна додатку;

– set_copyright - допоміжний метод, що виконує оформлення елементу "Copyright" головного вікна додатку;

– set_wellcome_promt - метод, що виводить відповідне повідомлення для користувача у разі вдалого запуску додатку.

– get_about_info - метод, що будує допоміжне модальне вікно з інформацією про додаток.

– run - метод, що запускає основний цикл додатку.

– exit_app - метод що завершує головний цикл додатку.

– get_upload_form - метод, що забезпечує завантаження тренувальних вибірок у додаток.

– make_prediction - контролер виконання прогнозів.

– train_nn - контролер тренування моделей (на основі MLP і LSTM).

– get_user_values - валідатор введених користувачем даних.

– regres_predict - створює екземпляр класу ModelMultRegression (модулю multRegression.py) і виконує обчислення прогнозу використовуючи моделі на основі методів логістичної регресії, опорних векторів, найменших квадратів, випадкового лісу, найближчого сусіда.

– mlp_predict - створює екземпляр класу ModelMLP (модулю mlp.py) і виконує обчислення прогнозу використовуючи модель на основі багатошарового перцептрону.

– lstm_predict - створює екземпляр класу ModelLSTM (модулю lstm.py) і виконує обчислення прогнозу використовуючи модель на основі мережі довгої короткочасної пам’яті.

– mlp_train - створює екземпляр класу ModelMLP (модулю mlp.py) і виконує тренування моделі на основі багатошарового перцептрону.

– lstm_train - створює екземпляр класу ModelLSTM (модулю lstm.py) і виконує тренування моделі на основі мережі довгої короткочасної пам’яті

Модуль multRegression.py призначений для обчислення прогнозу з використанням моделей на основі методів логістичної регресії, опорних векторів, найменших квадратів, випадкового лісу, найближчого сусіда і містить єдиний клас ModelMultRegression. Метод класу multi_regression_predict виконує обчислення прогнозу показників захворюваності.

Модуль mlp.py містить єдиний клас ModelMLP, що має два методи: mlp_predict і mlp_train, які виконують обчислення прогнозу використовуючи модель на основі багатошарового перцептрону і проводять тренування цієї моделі відповідно.

Модуль lstm.py містить єдиний клас ModelLSTM. Клас має три методи: series_to_supervised, lstm_predict і lstm_train. Метод series_to_supervised створює вибірку в такому вигляді, в якому він може бути застосований для тренування нейронної мережі довгої короткочасної пам'яті.

Методи lstm_predict і lstm_train виконують обчислення прогнозу використовуючи модель на основі мережі довгої короткочасної пам'яті і проводять тренування цієї моделі відповідно.

Сховище моделей (model_store) містить нотації моделей на основі багатошарового перцептрону і мережі довгої короткочасної пам'яті у форматі json (mlp.json і lstm.json), а також тензори синаптичних ваг цих мереж, що застосовуються при обчисленні прогнозу для різних показників захворюваності.

Усі використані в додатку вибірки являють собою файли у форматі csv, але для зручного зберігання і обробки первинної статистичної інформації і подальшого створення таких вибірок, до складу пакету додано базу даних (MySQL), яка має наступну структуру (рис. 4.54):

– Таблиця «Країни» (tab_countries). Наразі, має єдиний запис.

– Таблиця «Області» (tab_regions). Містить перелік областей по регіонах. Зв'язана з таблицею «Області» у співвідношенні «багато до одного».

– Таблиця «Населення» (tab_population_month). Містить кількість населення по регіонах і місяцях. Зв'язана з таблицею «Області» у співвідношенні «багато до одного».

– Таблиця «Показники забруднення» (tab_polution_params). Містить один запис (обсяг викидів забруднюючих речовин і діоксиду вуглецю від стаціонарних джерел).

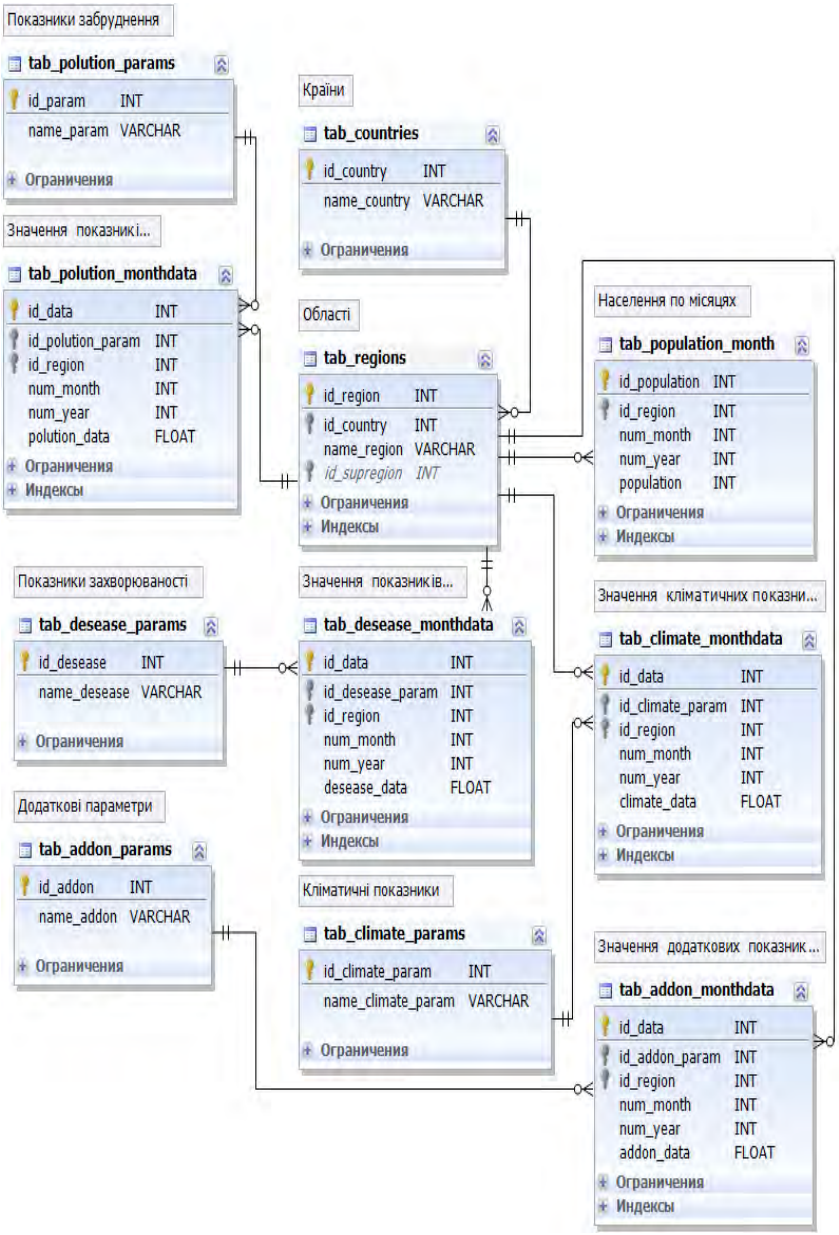


Рисунок 4.54 – ER-діаграма бази даних

– Таблиця «Значення показників забруднення» (tab_polution_monthdata). Містить значення параметру з таблиці «Показники забруднення» по регіонах і місяцях. Зв’язана з таблицями «Області» і «Показники забруднення» у співвідношенні «багато до одного».

– Таблиця «Показники захворюваності» (tab_desease_params). Містить показники захворюваності.

– Таблиця «Значення показників захворюваності» (tab_desease_monthdata). Містить значення показників захворюваності (таблиця «Показники захворюваності») по регіонах і роках. Зв’язана з таблицями «Області» і «Показники захворюваності» у співвідношеннях «багато до одного».

– Таблиця "Кліматичні показники" (tab_climate_params). Містить кліматичні показники.

– Таблиця "Значення кліматичних показників" (tab_climate_monthdata). Містить значення кліматичних показників по регіонах і роках Зв’язана з таблицями «Області» і «Кліматичні показники» у співвідношеннях «Багато до одного».

– Таблиця "Додаткові параметри" (tab_addon_params). Містить додаткові параметри.

– Таблиця "Значення додаткових параметрів" (tab_addon_monthdata). Містить значення додаткових параметрів по регіонах і роках. Зв’язана з таблицями «Області» і «Додаткові параметри» у співвідношенні «багато до одного».

Як зазначено вище, пакет містить набір рішень для моделювання залежності показників захворюваності від обсягів викидів забруднюючих речовин в атмосферне повітря від стаціонарних джерел.

Файлова структура проєкту наведена на рисунку 4.55.

Папка datasets містить набір тренувальних вибірок, що використовувались під час тренування моделей. LSTM_optim - набір скриптів для оптимізації топології і макропараметрів LSTM мережі за допомогою генетичного алгоритму та методу рою часток.

Папка model_store містить моделі MLP та LSTM мереж (у вигляді json - скриптів в нотації keras), а також тензори синаптичних ваг для цих моделей, які можуть змінюватися при виконанні тренування моделей користувачем додатку.

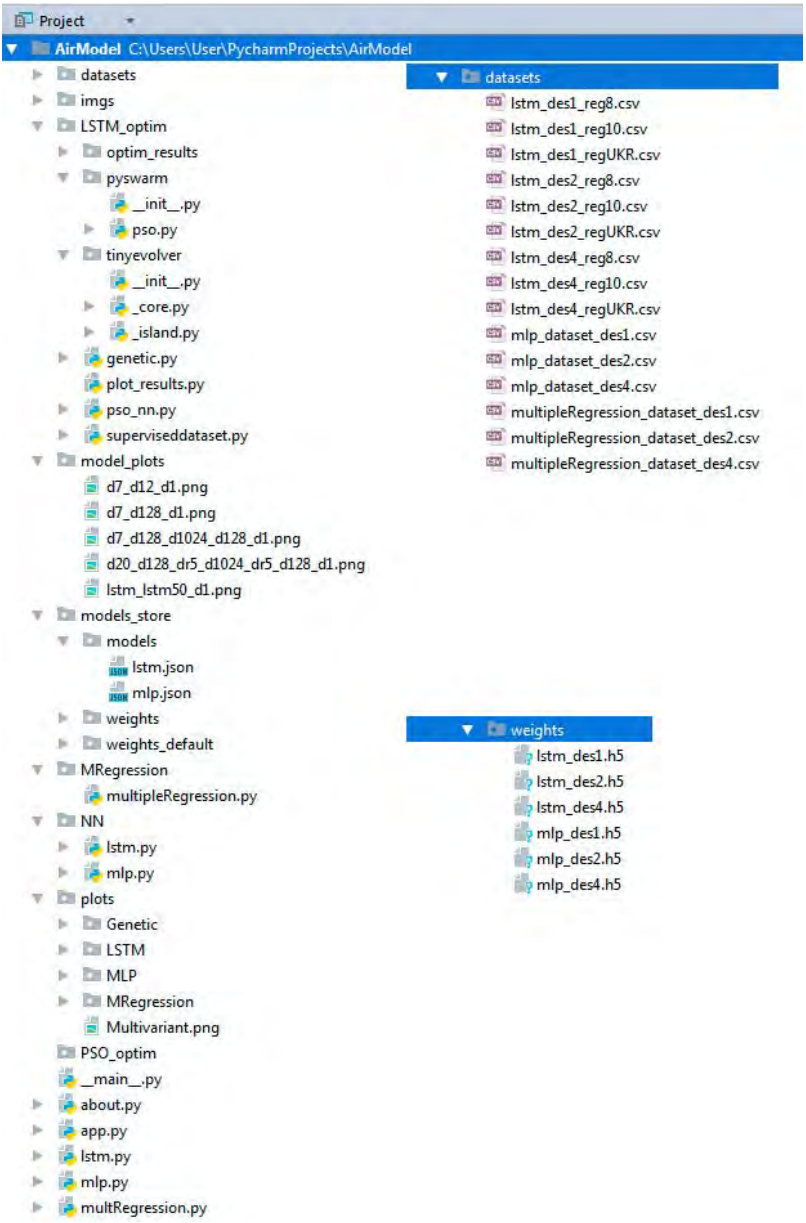


Рисунок 4.55 - Файлова структура пакету Air Health Model

Папки NN і MRegression містять скрипти для тренування і перевірки традиційних моделей та моделей на основі MLP і LSTM.

Папка plots містить результати досліджень проєкту в графічному вигляді.

about.py, app.py, lstm.py, main.py, mlp.py і multRegression.py - файли додатку і інтерфейсу користувача для використання пакету оператором, що не має навичок програмування, зокрема мовою Python.

Для довготривалого зберігання статистичних даних, або даних генерованих під час роботи з пакетом Air Health Model, разом з пакетом публікується також база даних (MySQL), що початково містить статистичні дані, що були використані в ході дослідження і моделювання.

Для розробки інтерфейсу користувача використано бібліотеку Python tkinter, яка є стандартною бібліотекою Python.

Головний клас додатку – class AirModel – має такі методи:

– def __init__ - конструктор класу. У якості аргументів отримує назву додатку, версію додатку і розмір вікна додатку. В конструкторі відбувається ініціалізація змінних класу та запуск головного циклу додатку;

– def set_app_title – встановлює заголовок додатку;

– def set_app_win_size – встановлює розмір вікна додатку;

– def set_copyright – відображає інформацію про авторські права в нижній частині вікна додатку;

– def set_wellcome_promt – відображає строку привітання користувача у вікні виводу додатка;

– def get_about – створює вікно з інформацію «про додаток»;

– def run – запуск головного циклу додатку;

– def exit_app – створює вікно для підтвердження наміру користувача закрити додаток;

– def get_upload_form – створює форму завантаження тренувальних вибірок для навчання моделей використаних у додатку;

– def make_prediction – викликається при натисканні на кнопку «Обчислити прогноз», перевіряє чи обрав користувач усі необхідні параметри для обчислення прогнозу і чи введено всі необхідні дані;

– def train_nn – викликається при натисканні на кнопку «Тренувати модель», перевіряє чи обрав користувач усі необхідні

параметри для тренування обраної моделі і чи введено всі необхідні дані;

– def get_user_values – перевіряє коректність значень введених користувачем, повертає значення введені користувачем (виправлені у разі необхідності);

– def regress_predict – прогнозування з використанням традиційних моделей. Створює екземпляр класу ModelMultRegression модулю MRegression.py і використовує його методи для обчислення прогнозу на основі обраної користувачем моделі і введених значень параметрів;

– def mlp_predict – прогнозування з використанням нейронної мережі (MLP). Створює екземпляр класу ModelMLP модулю mlp.py і використовує його методи для обчислення прогнозу;

– def lstm_predict – прогнозування з використанням нейронної мережі (LSTM). Створює екземпляр класу ModelLSTM модулю lstm.py і використовує його методи для обчислення прогнозу;

– def mlp_train – тренування моделі MLP нейронної мережі. Створює екземпляр класу ModelMLP модулю mlp.py і використовує його методи для тренування моделі з використанням обраної користувачем тренувальної вибірки;

– def lstm_train – тренування моделі LSTM нейронної мережі. Створює екземпляр класу ModelLSTM модулю lstm.py і використовує його методи для тренування моделі з використанням обраної користувачем тренувальної вибірки.

Модуль multRegression.py призначений для обчислення прогнозу за допомогою моделей на основі:

1) логістичної регресії;
2) методу опорних векторів;
3) методу найменших квадратів;
4) випадкового лісу;
5) методу найближчого сусіда.

Модуль містить єдиний клас – ModelMultRegression, який, в свою чергу, містить два методи: конструктор класу __init__ і метод multi_regression_predict, який повертає значення прогнозу.

Модуль mlp.py містить клас ModelMLP і призначений для роботи з моделлю на основі багатошарового перцептрону. Методи класу mlp_predict і mlp_train призначені для обчислення прогнозу і

тренування моделі відповідно. Під час роботи модулю використовується модель зі сховища моделей mlp.json.

Модуль lstm.py містить клас ModelLSTM і призначений для роботи з моделлю на основі мережі довгої короткочасної пам'яті. Методи класу lstm_predict і lstm_train призначені для обчислення прогнозу і тренування моделі відповідно. Метод series_to_supervised якості аргументу отримує вибірку, що містить значення параметрів з метками часу (по роках) , а повертає вибірку виду, що може бути використана для тренування LSTM мережі (таблиця Таблиця 4.10 – Перші п'ять рядків вибірки з часовим шагом 1). Під час роботи модулю використовується модель зі сховища моделей lstm.json.

Допоміжний модуль about.py містить єдиний клас About і призначений для створення вікна, що містить інформацію про додаток.

Вхідні і вихідні дані. У якості вхідних даних при обчисленні прогнозу програма приймає обрану користувачем модель обчислення, показник захворюваності, який прогнозується і значення параметрів обчислення введених користувачем (табл. 4.14).

Таблиця 4.14 – Вхідні дані для обчислення прогнозу введені користувачем

Параметр	Тип даних
Обсяг викидів, тон	Число (роздільник десяткової частини - крапка)
Кількість населення	Ціле число
Середня температура повітря	Число (роздільник десяткової частини - крапка)
Середня кількість опадів	Число (роздільник десяткової частини - крапка)
Кількість лікарів	Ціле число
Кількість лікарняних ліжок	Ціле число
Середній вік населення	Число (роздільник десяткової частини - крапка)

Результат обчислення буде відображено у вікні виводу застосунку (рис. 4.56).

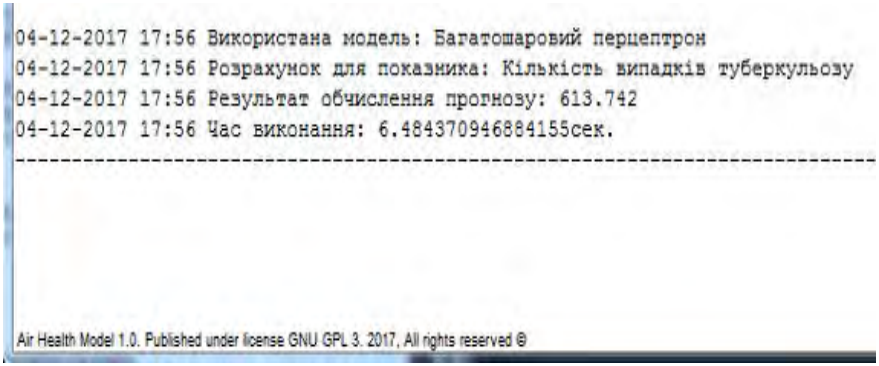


Рисунок 4.56 – Вивід результатів обчислення прогнозу для показника «Кількість випадків туберкульозу» з використанням нейронної мережі (MLP)

При тренуванні мережі вхідними даними є обрана користувачем модель та обраний із переліку тренувальний дата сет. Результат тренування також виводиться у вікно виводу додатку. Крім цього, в окремому вікні буде відображено графіки тренування (графік помилки мережі для тренувальної і валідаційної вибірки та графік зміни значення метрики під час тренування).

Під час роботи додатку усі результати виконання окремих операцій, а також інші повідомлення відображаються у вікні виводу додатку. Для більш детальної інформації, а також під час налаштування додатку рекомендовано отримувати інформацію про роботу додатку через консоль Python. Отримана таким чином інформація є більш повною і може виявитись корисною для спеціаліста.

Перелік можливих повідомлень, що генерує додаток під час роботи, наведено у таблиці 4.15.

Програмний пакет Air Health Model призначений для моделювання і відтворення залежності показників захворюваності населення від обсягів викидів забруднюючих речовин в атмосферне повітря від стаціонарних джерел забруднення.

До складу пакету входить додаток, який може бути використаний оператором для прогнозування таких показників захворюваності, як число нових випадків туберкульозу, число випадків захворювань системи кровообігу та число всіх зареєстрованих випадків раку.

Таблиця 4.15 – Можливі повідомлення під час роботи додатку Air Health Model.

Вітаємо! Програма Air Health Model завантажена і готова до роботи.	Повідомлення про те, що додаток завантажений і готовий до роботи
Будь ласка, оберіть показник захворюваності	З'являється, коли користувач намагається зробити прогноз або тренування моделі, але не обрав показник захворюваності
Будь ласка, оберіть модель	З'являється, коли користувач намагається зробити прогноз або тренування моделі, але не обрав показник модель
Використана модель: Багатошаровий перцептрон Розрахунок для показника: Кількість випадків туберкульозу Результат обчислення прогнозу: 10,742 Час виконання: 6,484370946884155сек.	Повідомлення про закінчення обчислення прогнозу – зазначено модель, що була використана, показник захворюваності, для якого обчислено прогноз, результат обчислення прогнозу та час виконання обчислення
Тренування моделі: Багатошаровий перцептрон Тренування для показника: Кількість випадків туберкульозу Використано тренувальну вибірку: mlp_dataset_des1.csv Тренування завершено за: 94,00936889648438 сек.	Повідомлення про закінчення тренування обраної моделі – зазначено обрану модель, показник захворюваності, для якого виконано тренування моделі, використаний тренувальну вибірку та час виконання операції

Для виконання таких обчислень, оператору необхідно у відповідні поля форми додатку ввести значення обсягу викидів забруднюючих речовин (в тонах), кількість населення у регіоні, для якого відтворюється прогнозування, середню річну температуру повітря у регіоні, середню кількість опадів, кількість лікарів у регіоні, кількість лікарняних ліжок в стаціонарних відділеннях закладів охорони здоров'я у регіоні та середній вік населення у регіоні.

При використанні моделі «Мережа довгої короткочасної пам'яті» також треба ввести значення обраного показника захворюваності за минулий період.

Наразі, поточна версія додатку дозволяє виконувати прогнозування по вказаним показникам на період в один місяць.

Окрім цього, додаток має відповідний функціонал для завантаження власних дата сетів для тренування моделей

«Багатошаровий перцептрон» і «Мережа довгої короткочасної пам’яті».

Виконання програми. Після вдалого запуску додатку відкриється головне вікно додатку (рис. 4.57):

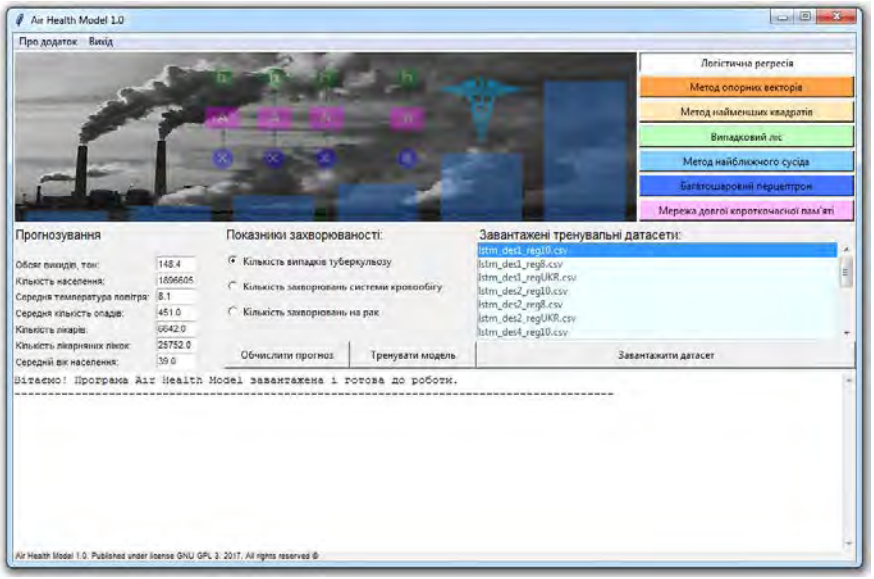


Рисунок 4.57 – Головне вікно додатку Air Health Model

У правому верхньому кутку міститься перелік доступних моделей, що можуть бути використані для обчислення прогнозу. Блок «Прогнозування» містить поля для вводу значень параметрів, що необхідні для обчислення прогнозу. Блок «Показники захворюваності» призначений для вибору показника захворюваності, по якому буде обчислений прогноз, чи для якого буде проведено тренування обраної моделі.

Блок «Завантажені тренувальні датасети» містить перелік доступних вибірок, що можуть бути використані для навчання обраної моделі.

У нижній частині вікна розташоване вікно виводу додатку (консоль), в якому під час роботи додатку будуть з’являтися інформаційні повідомлення. Після вдалого запуску програми в цьому

вікні з’явиться повідомлення «Вітаємо! Програма Air Health Model завантажена і готова до роботи.»

Для обчислення прогнозу необхідно ввести значення параметрів блоку «Прогнозування», обрати показник захворюваності в блоці «Показники захворюваності», обрати модель для обчислення з переліку доступних моделей обчислення і натиснути кнопку «Обчислити прогноз».

Після вдалого обчислення в консолі додатку повинне з’явитися відповідне повідомлення з результатами виконання обчислення.

Обчислення прогнозу з використанням мережі довгої короткочасної пам’яті дещо відрізняється від обчислень за допомогою інших моделей. Так після натискання кнопки «Обчислити прогноз», відкриється додаткове діалогове вікно, в якому треба ввести значення обраного показника захворюваності за минулий період. Тільки після цього почнеться обчислення прогнозу (рис. 4.58).

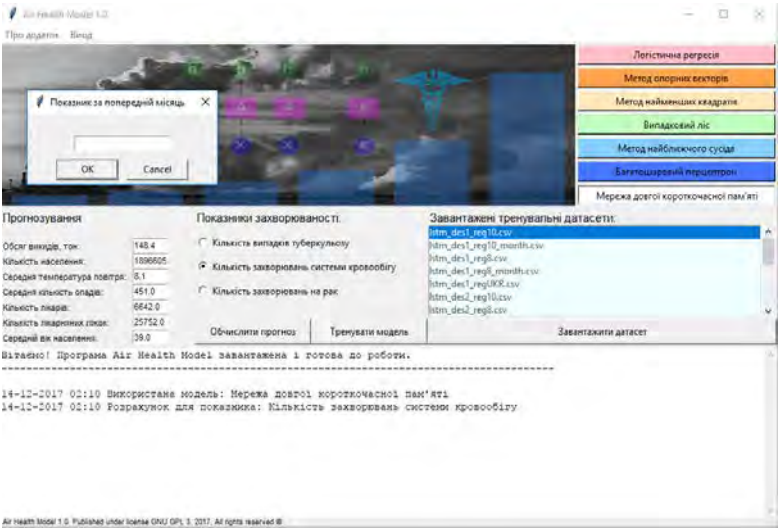


Рисунок 4.58 – Вікно вводу значення показника захворюваності за минулий період (в даному конкретному випадку – кількості випадків туберкульозу)

Додаток за замовченням вже містить треновані моделі багатошарового перцептрону і мережі довгої короткочасної пам’яті, але у разі необхідності оператор може провести перенавчання цих

моделей. Для цього необхідно завантажити тренувальну вибірку. Для завантаження тренувального датасету необхідно натиснути кнопку «Звантажити датасет», обрати файл дата сету на комп'ютері оператора і завантажити його у додаток. Після вдалого завантаження він з'явиться у переліку завантажених тренувальних вибірок.

Після обрання необхідної моделі і тренувальної вибірки необхідно натиснути кнопку «Тренувати модель».

Слід враховувати, що процес тренування моделей є досить ресурсномістким і може займати деякий час. Під час тренування інший функціонал додатку буде недоступним до закінчення обчислень.

Після закінчення тренування моделі з'явиться додаткове вікно, в якому в графічному вигляді буде представлений результат тренування моделі (графік динаміки помилки і метрики навчання мережі) (рис. 4.59).

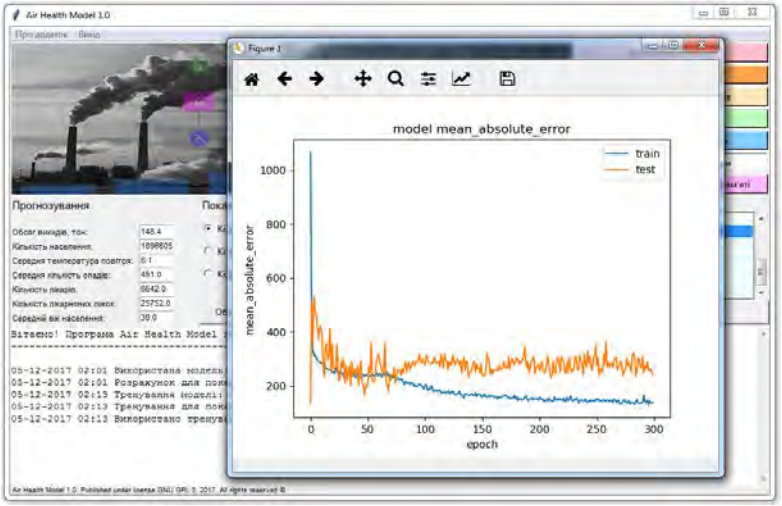


Рисунок 4.59 – Графічне представлення результату тренування моделі

Алгоритм тренування, чи топологія використаних нейронних не можуть бути змінені оператором через додаток.

Повідомлення оператору. Під час виконання обчислення прогнозу, чи тренування мережі в консолі додатку можуть з'являтися

інформаційні повідомлення. Перелік можливих повідомлень зазначений в розділі «Керівництво програміста» в таблиці 4.15.

4.5 Висновки за розділом 4

Проаналізовано наявну офіційну статистичну інформацію щодо обсягів викидів забруднюючих речовин в атмосферне повітря від стаціонарних джерел забруднення і динаміки таких показників захворюваності населення, як кількість випадків туберкульозу, кількість виявлених в амбулаторних умовах захворювань системи кровообігу і кількість захворювань на рак в різних регіонах України.

Досліджено методи побудови математичної моделі залежності вказаних показників захворюваності від обсягів викидів забруднюючих речовин. Отримані під час дослідження результати було вивчено і проаналізовано, зроблені висновки щодо можливості використання побудованих моделей для обчислення прогнозу показників захворюваності.

В результаті роботи було зроблено висновок, що найбільш оптимальною, з точки зору точності прогнозу є модель на основі нейронної мережі довгої короткочасної пам’яті (LSTM).

Під час створення і тренування моделі на основі мережі довгої короткочасної пам’яті було досліджено можливість використання методу рою часток і генетичного алгоритму для оптимізації параметрів мережі, розроблено модифікацію класичного методу і модифікацію оператору матацій, які раніше не використовувалась.

Розроблено дві модифікації генетичного методу, які раніше не застосовувалися – використання діплоїдного набору хромосом та модифікація оператору мутацій. Перша модифікація полягає в використанні не однієї хромосоми в каріотипі особин популяції, а пари гомологічних хромосом, тобто, діплоїдного набору хромосом. Під каріотипом особини мається на увазі набір хромосом, специфічний для даного виду особин, що характеризується певною кількістю хромосом та особливістю їхньої будови. Фенотип особини визначається одним з алельних генів обраним випадковим чином. Використання такої модифікації дозволяє підтримувати доволі велику варіабельність ознак популяції впродовж еволюції, створюючи потенціал для подолання імовірних локальних екстремумів.

Також було розроблено модифікацію оператору мутацій, яка також раніше не використовувалась. На відміну від класичного методу, особини, які піддаються дії оператору мутації обираються не випадковим чином, а у відповідності до їх мутаційної стійкості, що відповідає значенню функції пристосованості особини. Таким чином, мутують «слабкіші» особини, а геном «сильних» особин залишається незмінним. У цьому випадку зменшується вирогідність втрати досягнутого впродовж еволюції екстремуму функції внаслідок дії оператора мутацій, а перехід до нового екстремуму здійснюється у випадку нокопичення достатньої питомої ваги «кращих» ознак в популяції.

Отримані в результаті оптимізації моделі нейронної мережі довгої короткочасної пам'яті було використано при створені програмного пакету Air Health Model для моделювання залежності показників захворюваності від обсягів викидів забруднюючих речовин, до складу якого входить додаток, який дозволяє швидко отримувати прогноз показників захворюваності, має дружній інтерфейс користувача і використовує підготовлені і треновані моделі.

4.6 Література до розділу 4

1. Викиди забруднюючих речовин в атмосферне повітря [Електронний ресурс]. – Держстат України, Статистична інформація, 2017. – Режим доступу: http://www.ukrstat.gov.ua/druk/publicat/kat_u/publnav_ser_u.htm
2. Google maps platforv [Electronic resource]. – Access mode: https://developers.google.com/maps/
3. Стан забруднення природного середовища на території України [Електронний ресурс]. – Центральна геофізична обсерваторія ім. Б. Срезневського, 2017. – Режим доступу: http://cgo-sreznevskyi.kiev.ua/index.php?fn=u_zabrud&f=ukraine
4. Про затвердження нормативів граничнодопустимих викидів забруднюючих речовин із стаціонарних джерел: наказ Міністерства охорони навколишнього природного середовища. Наказ від 27.06.2006 № 309 [Електронний ресурс]. – Режим доступу: http://zakon2.rada.gov.ua/laws/show/z0912-06

5. Про охорону атмосферного повітря: закон України. Відомості Верховної Ради України (ВВР), 1992, № 50, ст.678 [Електронний ресурс]:– Режим доступу: http://zakon2.rada.gov.ua/laws/show/2707-12

6. Про затвердження порядку проведення та оплати робіт, пов'язаних з видачею дозволів на викиди забруднюючих речовин в атмосферне повітря стаціонарними джерелами, обліку підприємств, установ, організацій та громадян – підприємців, які отримали такі дозволи: постанова Кабінету Міністрів України Постанова від 13 березня 2002 р. N 302 [Електронний ресурс] – Режим доступу: http://zakon2.rada.gov.ua/laws/show/ 302-2002-п

7. Кодекс цивільного захисту україни (Відомості Верховної Ради (ВВР), 2013, № 34-35, ст.458) [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/laws/show/5403-17

8. United Nations Framework Convention on Climate Change [Electronic resource]. – Access mode: https://cop23.unfccc.int

9. Про забезпечення санітарного та епідемічного благополуччя населення: закон України. Відомості Верховної Ради України (ВВР), 1994, № 27, ст.218 [Електронний ресурс] – Режим доступу: http://zakon3.rada.gov.ua/laws/show/4004-12

10. Джерела і види забруднення атмосфери [Електронний ресурс]. - Экология и здоровье, 2017. – Режим доступу: http://www.childflora.org.ua/?page_id=138

11. Джерела та екологічні наслідки забруднення атмосфери [Електронний ресурс]. – Бібліотека BukLib.net, 2017. – Режим доступу: http://buklib.net/books/23920

12. Вплив забруднення навколишнього середовища на життя і здоров’я людей [Електронний ресурс]. – Екологія – Навчальні матеріали, 2017. – Режим доступу: http://pidruchniki.com/13930518/ekologiya/vpliv_zabrudnennya_navkolishnogo_seredovischa_zhittya_zdorovya_lyudey

13. Українська База Медико-Статистичної Інформації [Електронний ресурс]. – Центр Медичної Статистики МОЗ України, 2017. – Режим доступу: http://medstat.gov.ua/ukr/news.html?id=182

14. AirQ+: software tool for health risk assessment of air pollution [Electronic resource]. – WHO/Europe | Air quality – [AirQ+: software tool for health risk assessment of air pollution], 2017 – Access mode:

http://www.euro.who.int/ en/health-topics/environment-and-health/air-quality/activities/airq-software-tool-for-health-risk-assessment-of-air-pollution

15. ADMS-Forecast [Electronic resource]. – CERC – Environmental Software – ADMS-Forecast, 2017. – Access mode: http://www.cerc.co.uk/environmental-software/ADMS-Forecast.html

16. MathWorks [Electronic resource]. – Access mode: https://www.mathworks.com

17. Maple [Electronic resource]. – Access mode: https://www.maplesoft.com/products/Maple

18. RStudio – Open source and enterprise-ready professional software for R [Electronic resource]. – Access mode: https://www.rstudio.com

19. Ghazvini, K. Predictors of tuberculosis: Application of a logistic regression model / K. Ghazvini, M. Yousefi, F. Firoozeh, S. Mansouri // Gene Reports. – 2019. – Vol. 17. – P. 93-101. doi: 10.1016/j.genrep.2019.100527

20. Mei, B. Multi-task least squares twin support vector machine for classification / B. Mei, Y. Xu // Neurocomputing. – 2019. –Vol. 338. – P. 26-33. doi: 10.1016/j.neucom.2018.12.079

21. Dembinski, H. Application of the iterated weighted least-squares fit to counting experiments / H. Dembinski, M. Schmelling, R. Waldi // Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. – 2019. –Vol. 940. – P. 135-141. doi: 10.1016/j.nima.2019.05.086

22. Speiser, J. A comparison of random forest variable selection methods for classification prediction modeling / J. Speiser, M. Miller, J. Tooze, E. Ip // Expert Systems with Applications. – 2019. –Vol. 134. – P. 93-101. doi: 10.1016/j.eswa.2019.05.028

23. Alam, S. Research on particle swarm optimization based clustering: A systematic review of literature and techniques / S. Alam, G. Dobbie, Y. Koh, P. Riddle, S. Ur Rehman // Swarm and Evolutionary Computation. – 2014. – P. 1-13. Doi: 10.1016/j.swevo.2014.02.001

24. Метод найближчого сусіда: приклад роботи [Електронний ресурс]. – Режим доступу: https://uk.ruarrijoseph.com/obrazovanie/85409-metod-blizhayshego-soseda-primer-raboty.html

25. K-Nearest Neighbor Algorithm Optimization in Text Categorization [Electronic resource]. – Access mode: https://iopscience.iop.org/article/10.1088/1755-1315/108/5/052074

26. Хайкин, С. Нейронные сети. Полный курс / С. Хайкин – «Вильямс», 2016. – 1104 с.

27. Sak H. Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition [Electronic resource]/ H. Sak, A. Senior, F. Beaufays. – Access mode: https://arxiv.org/pdf/1402.1128

28. Yi, H. ACP-DL: A Deep Learning Long Short-Term Memory Model to Predict Anticancer Peptides Using High-Efficiency Feature Representation / H. Yi, Z. You, X. Zhou, L. Cheng, X. Li, T. Jiang, Z. Chen // Molecular Therapy. Nucleic Acids. – 2019. – Vol. 17. – P. 1-9. doi: 10.1016/j.omtn.2019.04.025

29. Глібовець, М.М. Гібридний генетичний алгоритм вирішення задачі оптимізації структури інтегральної схеми / М. М. Глибовець, С. С. Гороховський, О. В. Краткова // Інженерія програмного забезпечення. – 2011. – № 1. – С. 70–76.

30. PSO Tutorial [Electronic resource]. – Particle Swarm Optimization Tutorial, 2006. – Access mode: http://www.swarmintelligence.org/ tutorials.php

31. Pirani M. Analysing the health effects of simultaneous exposure to physical and chemical properties of airborne particles / M. Pirani, N. Best, M. Blangiardo, S. Liverani, R. Atkinson, G. Fuller // Environment International. – 2015. –Vol. 79. – P. 56-64. doi: 10.1016/j.envint.2015.02.010

32. Кононюк, А. Ю. Нейронні мережі і генетичні алгоритми / А. Ю. Кононюк. – К. : «Корнійчук», 2008. – 446 с.

33. Keras [Electronic resource]. – Keras: The Python Deep Learning library, 2017. – Access mode: https://keras.io

34. Theano [Electronic resource]. – Wellcome – Theano 1.0.0 documentation, 2017. – Access mode: http://deeplearning.net/software/theano

35. CUDA Toolkit [Electronic resource]. – CUDA Toolkit | NVIDIA Developer, 2017. – Access mode: https://developer.nvidia.com/cuda-toolkit

36. PyCharm [Electronic resource]. – PyCharm: Python IDE for Professional Developers by JetBrains, 2017. – Access mode: https://www.jetbrains.com/pycharm

37. Swingler, Kevin Applying Neural Networks: A Practical Guide [Text] / Keven Swingler. – Published by Morgan Kaufman, 2001. – 301 p.

38. Broomhead, D.S. Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Nctworks / D.S. Broomhead, D. Lowe // RSRE Memorandum. – 1988. – No. 4148. – 35 p.

39. Hopfield, J.J. Neural networks and physical systems with emergent collective computational abilities [Electronic resource]. – Access mode: https://bi.snu.ac.kr/Courses/g-ai09-2/hopfield82.pdf.

40. Закон чистоти гамет [Електронний ресурс] – Моя освіта – Закон чистоти гамет, 2017. – Режим доступу: http://moyaosvita.com.ua/biologija/zakon-chistoti-gamet

41. Conda [Electronic resource] – Conda – Conda documentation, 2017. – Access mode: https://conda.io

42. Rosenblatt, F. The perceptron: a probabilistic model for information storage and organization in the brain [Electronic resource] – Access mode: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.335.3398&rep=rep1&type=pdf.

43. Bourlard, H. Auto-Association by Multilayer Perceptrons and Singular Value Decomposition [Electronic resource] – Access mode: https://publications.idiap.ch/downloads/reports/2000/rr00-16.pdf.

44. Vlasic, I. Improving genetic algorithm performance by population initialisation with dispatching rules / I. Vlasic, M. Durasevic, D. Jakobovic // Computers & Industrial Engineering. – 2019. – Vol. 137. –106030. doi: 10.1016/j.cie.2019.106030

45. Sharifzadeh, M. Machine-learning methods for integrated renewable power generation: A comparative study of artificial neural networks, support vector regression, and Gaussian Process Regression / M. Sharifzadeh, A. Sikinioti-Lock, N. Shah // Renewable and Sustainable Energy Reviews. – 2019. – Vol. 108. – P. 513-538. doi: 10.1016/j.rser.2019.03.040

46. Zhang, Q. Convergence of decomposition methods for support vector machines / Q. Zhang, D. Wang, Y. Wang // Neurocomputing. – 2018. – Vol. 317. – P. 179-187. doi: 10.1016/j.neucom.2018.08.030

47. Soares, F. Support vector regression coupled with wavelength selection as a robust analytical method / F. Soares, M. Anzanello //

Chemometrics and Intelligent Laboratory Systems. – 2018. – Vol. 172. – P. 167-173. doi: 10.1016/j.chemolab.2017.12.007

48. Kumar, R. Evolving Differential evolution method with random forest for prediction of Air Pollution / R. Kumar, D. Kumar // Procedia Computer Science. – 2018. – Vol. 132. – P. 824-833. doi: 10.1016/j.procs.2018.05.094

ВИСНОВКИ

У монографії наведено опис моделей, методів та програмних засобів обчислювального інтелекту, які ефективно можуть використовуватися при вирішенні завдань прийняття рішень та оптимізації. Запропоновано авторські методи та програмні заособи на основі штучних нейронних мереж та стохастичної оптимізації, що у порівнянні з іншими здатні суттєво підвищити якість та швидкість побудови моделей

У першому розділі наведено три модифікації простого генетичного алгоритму для підбору параметрів навчання нейронних мереж – альфа-бета, альфа-бета фіксована, фіксована. У розроблених модифікаціях, на відміну від простого генетичного алгоритму, використовуються запропоновані евристичні процедури, зокрема мутація з використанням методу Монте-Карло, модифіковані оператори відбору та схрещування. Це дозволило підвищити показник точності та зменшити час оптимізації за допомогою розроблених модифікацій простого генетичного методу у порівнянні з його базовою версією. Результати експериментів показали, що модифікація Альфа-Бета є кращим підходом вирішення задачі розпізнавання учасників дорожнього руху. Оскільки дана модифікація дозволила отримати кращий показник точності за менший час підбору.

У другому розділі розглянуто методи розв’язання транспортної задачі та задачі складської логістики на основі квазіінтелектуальних методів оптимізації. Запропоновано методи розв’язання задачі комівояжера на основі еволюційного підходу, виконано їх апробацію у сфері аптечного бізнесу шляхом оптимізації процесу роботи засобу подачі ліків. Розроблено модифікацію еволюційного алгоритму – а саме 3 методи ініціалізації початкової популяції простого ГА: стандартний метод (Default), змішаний метод (Random), змішаний оптимальний (Random optimal). Запропонована модифікація дозволяє суттєво підвищити швидкість за рахунок того що при використанні стандартного методу ініціалізації початкової популяції, початкове значення розрахованої функції пристосованості та відповідно дистанції буде меншим ніж при використанні інших методів ініціалізації, що дозволить більш точніше визначити оптимум задачі.

У третьому розділі розглянуто та запропоновано математичні моделі прогнозування й оптимізації показників прибутковості та оптимальності формування асортименту. Розроблено нові генетичні оператори мутації. Перша модифікація полягає в виборі значень замін для мутації не випадковим чином, а з ряду, що підкоряється закону нормального розподілу. Друга модифікація служить для визначення доцільності мутації хромосоми, спираючись на знання ретроспективних та прогнозних даних з використанням у якості прогнозної моделі ШНМ. Третя модифікація полягає у комбінації двох зазначених модифікацій. Аналіз результатів теоретичних і експериментальних досліджень показав, що розроблений генетичний є доцільним та ефективним для вирішення проблеми оптимізації асортименту аптеки. Застосування розроблених методів призведе до більш ефективного використання площі аптек, зменшення незадоволеного попиту та, в кінцевому результаті, до зменшення роздрібної вартості ліків за рахунок зменшення видатків на зберігання та обслуговування неоптимальне завантажених площин аптеки.

У четвертому розділі досліджено та розроблено методи побудови математичної моделі залежності вказаних показників захворюваності від обсягів викидів забруднюючих речовин. Розроблено модифікований генетичний метод для оптимізації параметрів моделі на основі нейронної мережі довгої короткочасної пам’яті. Запропоновано модифікацію одного з операторів генетичного методу, а саме оператору мутацій, яка дозволяє проводити пошук оптимальних значень, виключаючи втрату надбаних під час пошуку кращих рішень. Отже, отримані результати дозволяють зробити висновок про те, що запропонована модель на основі нейронної мережі довгої короткочасної пам’яті та модифікація класичного методу і оператора мутації є доцільними та ефективним рішенням для встановлення математичної залежності показників здоров’я від обсягів викидів забруднюючих речовин. Практичне використання розроблених методів дасть можливість своєчасно коригувати плановані лікувально-діагностичні, профілактичні заходи, завчасно визначати необхідні ресурси для локалізації та ліквідації захворювань з метою збереження здоров'я населення.

Наведений у книзі матеріал пройшов практичну апробацію авторами у закладах вищої освіти м. Запоріжжя. Зокрема, матеріал

книги використано при читанні курсів «Нейроінформатика», «Технології та використання штучних нейронних мереж», «Системи штучного інтелекту», «Еволюційне програмування» в Національному університеті «Запорізька політехніка». Книга також може використовуватися при вивченні окремих розділів інших дисциплін, у курсовому проєктуванні та виконанні атестаційних робіт бакалавра та магістра, при підготовці аспірантів.

Наукове видання

СУББОТІН Сергій Олександрович,
ОЛІЙНИК Андрій Олександрович,
ФЕДОРЧЕНКО Євген Миколайович
РУДЬ Микита Сергійович
ФЕРДМАН Павел Михайлович
ФІЛАТОВ Сергій Олександрович
ФЕДОРЧЕНКО Юлія Володиміровна
ХАРЧЕНКО Анастасія Сергіївна

МАТЕМАТИЧНІ ТА ПРОГРАМНІ ЗАСОБИ ДЛЯ ПРИЙНЯТТЯ РІШЕНЬ, РОЗПІЗНАВАННЯ ОБРАЗІВ Й ІНТЕЛЕКТУАЛЬНОГО ДІАГНОСТУВАННЯ

Монографія

Верстання *Дяченко О.О.*

Підписано до друку 12.02.2020 Формат 60×84 / 16. Ум. друк. арк. 15,75.
Тираж 100 прим. Зам. № 224.

Національний університет "Запорізька політехніка"
Україна, 69063, м. Запоріжжя, вул. Жуковського, 64
Тел.: (061) 769–82–96, 220–12–14

Свідоцтво суб'єкта видавничої справи ДК № 6952 від 22.10.2019.