Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка застосунку для розпізнавання математичних формул засобами згорткових нейронних мереж

Виконала: студентка IV курсу, групи СНз-41
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

_____ Ділай О.І.
(підпис) (прізвище та ініціали)

Керівник _____ Мацюк Г.Р.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Шимчук Г.В.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль - 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«__» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студентці Ділай Ользі Іванівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка застосунку для розпізнавання математичних формул засобами згорткових нейронних мереж

Керівник роботи Мацюк Галина Ростиславівна, к.соц. ком., доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «29» 04 2024 року № 4/7-473

2. Термін подання студентом завершеної роботи 09.06.2024 р.

3. Вихідні дані до роботи наукові літературні джерела

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ
1. Аналіз предметної області.
2. Теоретична частина
3. Реалізація та тестування застосунку
4. Безпека життєдіяльності, основи охорони праці
Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Титул. 2. Актуальність. 3. Мета, задачі дослідження. 4. Огляд аналогів
5. Схема дії нейромережі та згорткового шару. 6Демонстрація роботи YOLO.
7. Діаграма компонентів застосунку. 8. Засоби програмної розробки
9. Розмітка даних. 10. Скріншот боту та код
11. Функціональне тестування. 12. Приклад роботи застосунку.
13. Результати тестування застосунку. 14. Основні результати проведеного дослідження

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи хорони праці			

7. Дата видачі завдання 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	29.04 – 30.04.24	*Виконано*
2.	Підбір джерел про розпізнавання зображень	01.05 – 05.05.24	*Виконано*
3.	Опрацювання джерел про розпізнавання зображень	01.05 – 05.05.24	*Виконано*
4.	Виконання дослідження щодо розпізнавання		
	математичних формул	11.05 – 15.05.24	*Виконано*
5	Розроблення програмного коду	16.05 – 19.05.24	*Виконано*
6.	Оформлення розділу «Аналіз предметної області»	20.05 – 23.05.24	*Виконано*
7.	Оформлення розділу «Теоретична частина»	24.05 – 27.05.24	*Виконано*
8.	Оформлення розділу «Реалізація та тестування		
	застосунку»	28.05 – 30.05.24	*Виконано*
9.	Виконання завдання до підрозділу «Безпека		
	життєдіяльності, основи хорони праці»	31.05 – 03.06.24	*Виконано*
10.	Оформлення кваліфікаційної роботи	20.05 – 03.06.24	*Виконано*
11.	Нормоконтроль	01.06 – 05.06.24	*Виконано*
12.	Перевірка на плагіат	04.06 – 07.06.24	*Виконано*
13.	Попередній захист кваліфікаційної роботи	07.06 – 09.06.24	*Виконано*
14.	Захист кваліфікаційної роботи	10.06.24	

Студент _____ Ділай О.І.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Мацюк Г.Р.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка застосунку для розпізнавання математичних формул засобами згорткових нейронних мереж // Ділай Ольга Іванівна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра комп'ютерних наук, група СНз–41 // Тернопіль, 2024 // С. – 53, рис. – 45, табл. – 3, слайдів – 14, бібліогр. – 41.

Ключові слова: YOLOv5, машинне навчання, нейронна мережа, розпізнавання формул

Кваліфікаційна робота присвячена розробці мобільного застосунку розпізнавання рукописних математичних формул із застосуванням згорткових нейромереж.

У першому розділі проведено аналіз предметної області, що включає дослідження існуючих аналогів та наукової літератури..

Другий розділ присвячено теоретичній частині, яка включає в себе опис роботи архітектури YOLOv5, теоретичні відомості про згорткові нейронні мережі і огляд набору даних, котрий використовується при проведенні дослідження. Також міститься аналіз функціональних та нефункціональних вимог та проектування архітектури застосунку за допомогою використання діаграми компонентів. Описані програмні засоби, як застосовуються.

Третій розділ присвячено реалізації застосунку. Описано процес розмітки наборів даних, кінцевий пакет файлів після обробки зображень, архітектура попередньої моделі YOLOv5та інтерфейс взаємодії з користувачем. Наведено процес функціонального тестування реалізованої розробки, тестування нейронних мереж та експерименти щодо навчання нейронних мереж.

У четвертому розділі розглянуто важливі питання безпеки життєдіяльності та основ охорони праці.

ANNOTATION

Development of an application for recognizing mathematical formulas using convolutional neural networks // Dilai Olha // Ternopil Ivan Pul'uj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science // Ternopil, 2024 // P. - 53, Fig. - 45, Table - 3, Slides - 14, References - 41.

Keywords: YOLOv5, machine learning, neural network, formula recognition

Thesis deals with the development of a mobile application for recognition of handwritten mathematical formulas using convolutional neural networks.

In the first chapter, an analysis of the subject area is carried out, which includes the study of existing analogues and scientific literature.

The second section is devoted to the theoretical part, which includes a description of the YOLOv5 architecture, theoretical information about convolutional neural networks and an overview of the data set used in the research. Also includes functional and non-functional requirements analysis and application architecture design using a component diagram. Software tools are described as they are used.

The third section is devoted to application implementation. The dataset markup process, the final file package after image processing, the architecture of the YOLOv5 pre-model, and the user interface are described. The process of functional testing of the implemented development, testing of neural networks and experiments on learning neural networks are presented.

The fourth chapter deals with important issues of life safety and the basics of labor protection.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ СКОРОЧЕНЬ І ТЕРМІНІВ

MNIST (Modified National Institute of Standards and Technology) – об'ємна база даних зразків рукописного написання цифр.

OCR (optical character recognition) – оптичне розпізнавання символів.

YOLO (You Only Look Once) – алгоритм глибинного навчання, який широко використовується для виявлення об'єктів.

ЗНМ – згорткова нейронна мережа.

Навчання – первинна обробка кількох великих наборів розмічених або нерозмічених даних, на основі яких мережі можуть більш точно обробляти невідомі вхідні дані.

Нейронна мережа – це метод у штучному інтелекті, який вчить комп'ютери обробляти дані таким же способом, як і людський мозок.

ПЗ – програмне забезпечення.

Розмітка набору даних – додавання мітки класу зображення для того, щоб навчити модель нейронної мережі.

ШНН – штучна нейронна мережа.

ЗМІСТ

ВСТУП

Актуальність теми. В даний час інформаційні технології все більше проникають у різні сфери життя людини, починаючи від автоматизації діловодства, закінчуючи постановкою діагнозів рекомендаційними системами.

У багатьох сферах життя нейронні мережі [1] вже давно стали повсякденністю, а їх застосування в системах вважається запорукою успіху для продажу тих чи інших застосунків чи послуг. Це ж торкнулося й галузі науки [2].

У більшості наукових текстів, в тому числі у кваліфікаційних та дисертаційних роботах і наукових статтях, містяться математичні вирази. Для вирішення задачі їх розпізнавання потрібно застосувати нестандартні концепції. Складність розпізнавання математичних записів перебуває у залежності від різних факторів, як то числа об’єктів, діапазону доступних символів і т.п. [3]

У цій роботі розглядається прототип системи для розпізнавання математичних формул з використанням ЗНМ. Подібні системи дозволяють вченим економити час для перенесення своїх розрахунків з паперу або дошки в текстові редактори.

Мета роботи – розробка програми для розпізнавання математичних формул з використанням ЗНМ.

Для досягнення мети виділено ряд завдань:

– провести огляд наукової літератури та аналогів для розпізнавання математичних формул з використанням ЗНМ;

– підготувати два навчальні набори даних;

– реалізувати дві ШНМ, провести їх навчання та тестування;

– спроектувати та реалізувати застосунок для розпізнавання математичних формул;

– провести тестування реалізованої програми.

Практичне значення цієї роботи полягає в тому, що в результаті буде отримано реальний застосунок для розпізнавання формул, написаних від руки, котрий вподальшому може бути використаний студентами та науковцями.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Порівняльний аналіз аналогів

1.1.1 Microsoft Math Solver

Цей засіб [4] використовує ШНМ для розпізнавання рівняння та подальшого пошуку його рішень на основі її передбачень. Тип нейронних мереж, що використовуються, не вказаний розробником. Програма існує для різних платформ: iOs, android, web -застосунок. У випадку, коли користувач використовує мобільний застосунок, зображення постає з камери смартфона і далі обробляється. Також завдання можна ввести і в рукописному режимі. Приклади роботи застосунку наведені на рис.1.1 – 1.3.

Рисунок 1.1 - Сканування рівняння

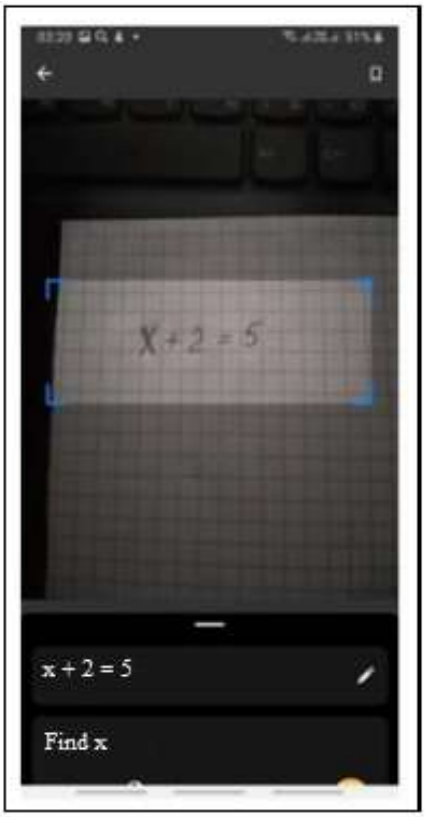

Рисунок 1.2 – Отримання результату

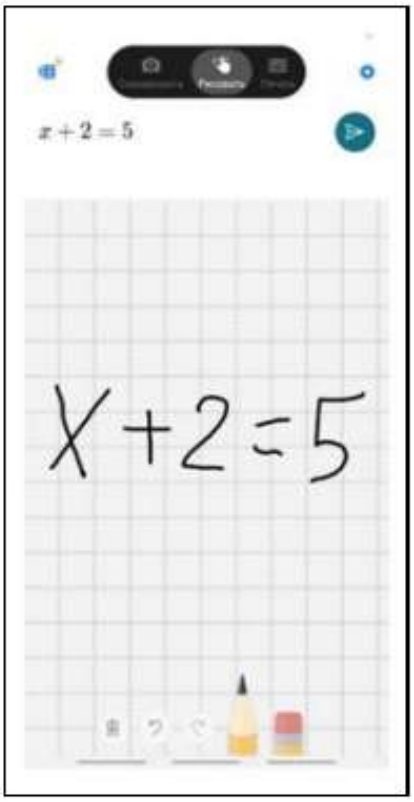

Рисунок 1.3 – Робота в рукописному режимі

1.1.2 PhotoMath

Даний мобільний застосунок [5] має такі функції:

– сканування рівняння;

– відправлення рівняння до хмарного сховища;

– видати результат.

У хмарному сховищі рівняння розпізнається високоточною ШНМ та ведеться пошук розв'язання. Тип нейронних мереж, що використовуються, не вказаний розробником. Програма не може працювати без підключення до Інтернету. Приклад роботи представлений на рис. 1.4.

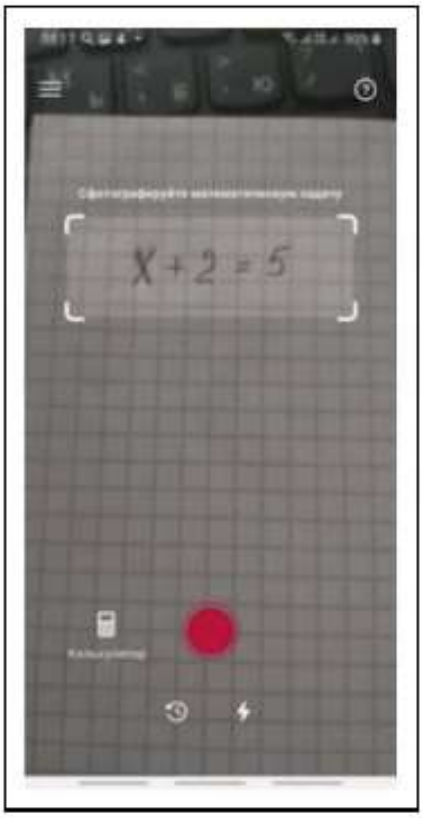

Рисунок 1.4 – Скан рукописного рівняння з аркуша паперу

1.1.3 OneNote

Є настільною програмою [6], в якій присутня функція перекладу рукописного введення математичних рівнянь у друкований текст. У перетворенні рівнянь OneNote використовує OCR [7].

Приклади роботи програмного засобу показані на рис. 1.5, 1.6.

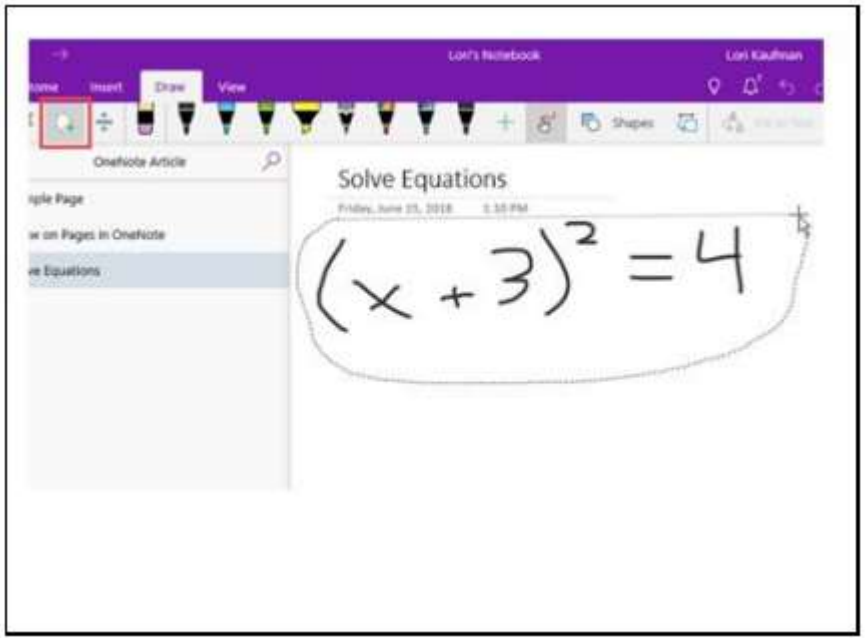

Рисунок 1.5 - Рукописне введення [8]

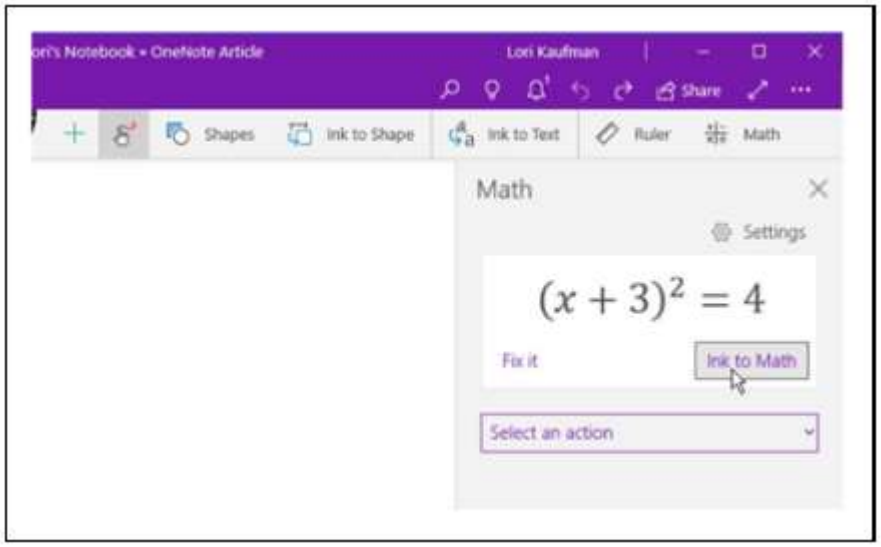

Рисунок 1.6 - Результат роботи [8]

Проведений огляд аналогів показав, що дані застосунки є дуже актуальними. Вони реалізовані у форматі мобільних та веб додатків. Це не завжди буває зручно, тому було прийнято рішення реалізувати програму у вигляді чат-бота.

1.2 Огляд літературних джерел

У [9] описується архітектура ЗНМ, експериментальна частина та результат

роботи.

Базова архітектура має 3 складові: вхідний шар; прихований шар; вихідний шар. Приховані шари представлені наступною архітектурою: шар згортки, функція активації, шар для зменшення розмірності. Схема представлена на рис. 1.7.

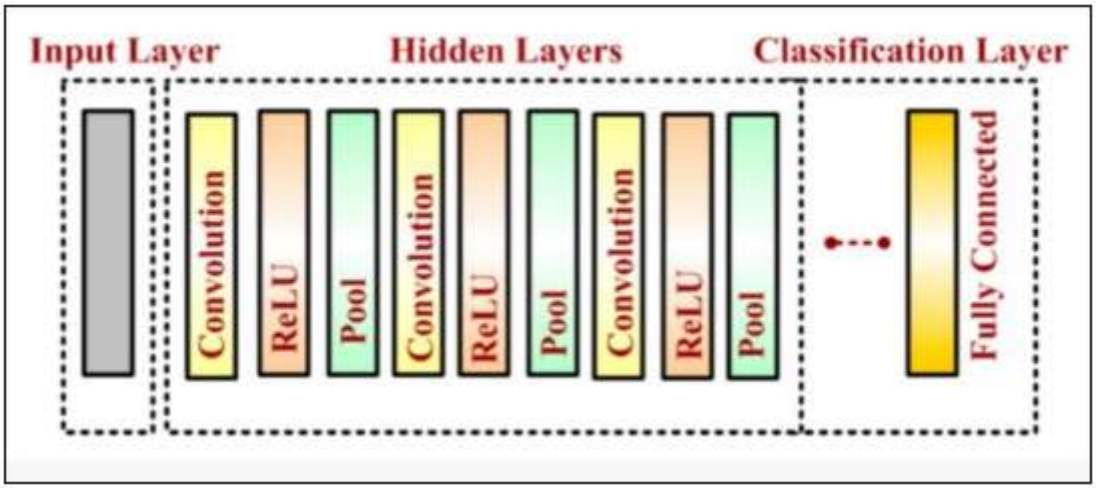

Рисунок 1.7 – Типова архітектура ЗНМ

У вхідному шарі завантажуються дані, в ньому ж вони описуються (висота, ширина, кількість RGB каналів).

Прихований шар вважається основою ЗНМ. У ньому відбувається виділення ознак рукописних цифр, у яких застосовуються функції згортки, об'єднання та активації.

Згортковий використовується для отримання ознак зображення. Зображення вхідного шару з розміром $n \times n$ і ядро розміром $m \times m$ згортаються в наступний шар розміром $L \times L$ за формулою (1.1):

$$L = n - m + 1. \quad (1.1)$$

Об'єднуючий шар додається серед двох згорткових, щоб зменшити розмірність вхідних даних для зменшення обчислювальної складності. Об'єднання дозволяє передавати вибрані значення на наступний рівень, залишаючи непотрібні значення позаду. Об'єднуючий шар також допомагає у

виборі функцій та управлінні переоснащенням. Операція об'єднання виконується незалежно. Він працює, витягуючи тільки одне вихідне значення з мозаїчних підобластей вхідних зображень, що не перекриваються. Приклад представлений на рис. 1.8.

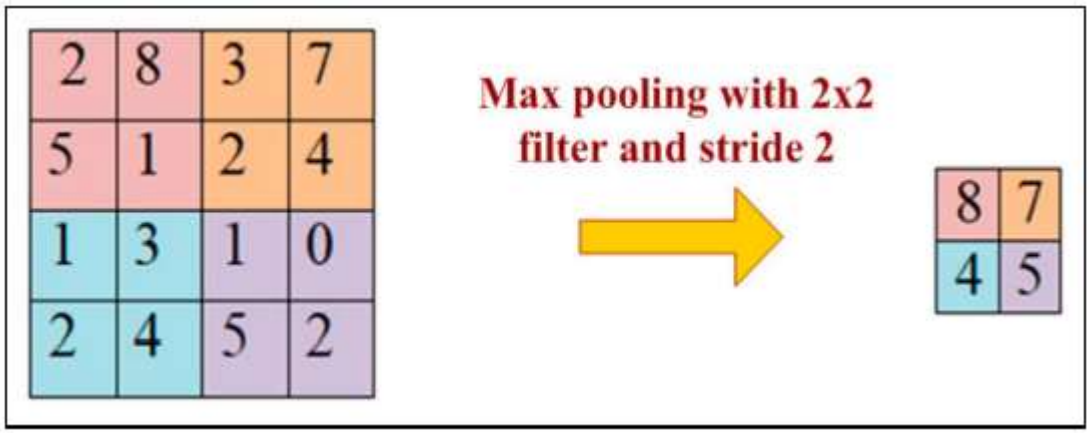

Рисунок 1.8 – Максимальне поєднання з ядром 2 x 2 та кроком 2 [9]

У [10] описуються послідовні кроки створення ШНМ для розпізнавання рукописних чисел, що використовуються TensorFlow та Keras.

Ціль роботи: досягти високої точності за рахунок використання чистої структури ЗНМ.

ЗНМ дуже сильні у сприйнятті структури рукописних цифр способами, які допомагають в автоматичному добуванні функцій і роблять їх найбільш вдалою технікою для вирішення завдань розпізнавання рукописного введення.

Нижче наведено алгоритм створення нейромережі:

– попередня обробка;

– кодування даних;

– побудова моделі. Розглянемо цей етап фрагмент, інші частини описані докладніше в інших статтях.

Після кодування даних зображення та мітки готові для встановлення в модель. Модель складається з вилучення ознак за допомогою згортки та двійкової класифікації. Згортка і максимальний пул проводиться для вилучення особливостей зображення, і 32 згорткових фільтра 3 x 3 застосовуються до зображення 28 x 28, за яким слідує шар максимального пулу розміру 2 x 2, за

яким йде ще один шар згортки з 64 фільтрами розміру 3 х 3. У результаті отримуємо зображення 7 х 7 для вирівнювання. Вирівнюючий шар згладить зображення 7 х 7 в ряд із 128 значень, які будуть зіставлені щільному шару з 128 нейронів, з'єднаних з вихідним шаром з 10 нейронів. Схема представлена на рис. 1.9;

– навчання та перевірка;

– оцінка моделі та прогнозування. Навчання моделі у 12 епохах з точністю 99,87% та втратою 0,043464561576.

Схема дії нейромережі представлена на рис. 1.9.

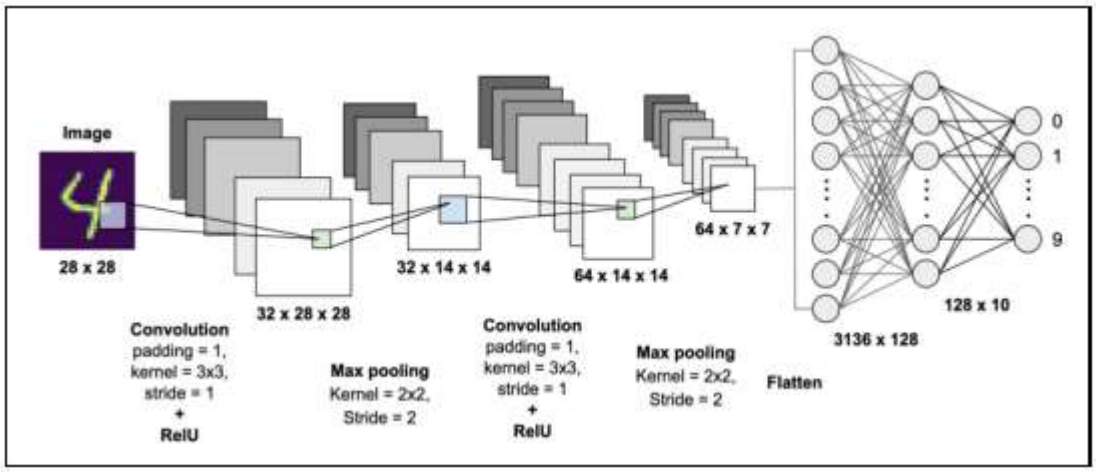

Рисунок 1.9 – Схема дії ЗНМ [10]

У [11] розповідається, як створити нейромережу, засновану на базі даних MNIST, для класифікації рукописного тексту. Продемонстрована робота з бібліотекою TensorFlow, написаною на Python 3, та надбудовою до неї - Kerns.

MNIST [12] складається з зразків написання цифр з 0 до 9. Зображення були взяті з документів з перепису населення США. Використовується для калібрування методів розпізнавання зображень для машинного навчання .

База складається з 60 000 прикладів рукописних цифр та 10 000 тестових зображень. Кожен файл має розмір 28 * 28 пікселів і наведено до сірого зображення. Приклад зображень представлений на рис. 1.10.

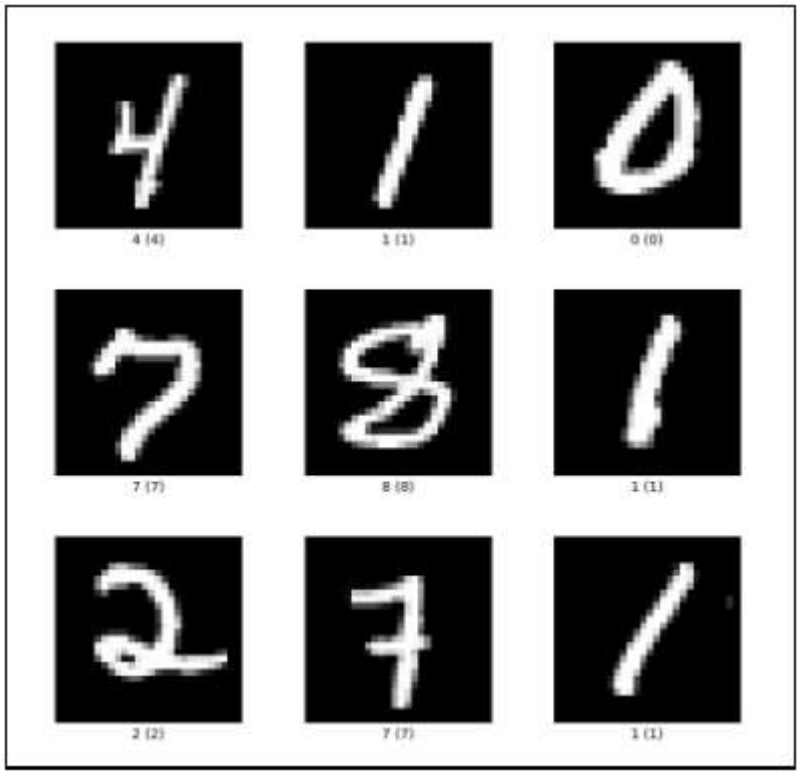

Рисунок 1.10 – Приклад зображень MNIST [12]

У [11] позначено етапи розробки, описані нижче:

– завантаження набору даних. Усі зображення завчасно вирівняні (для прикладу, кожне з них містить тільки нарисовану від руки цифру), а також мають один і той самий розмір 28x28 пікселів, окрім того всі зображення містять відтінки сірого. Отже, ми можемо завантажувати зображення та змінювати форму масивів даних, щоб мати один колірний канал;

– підготовка піксельних даних. Хорошою відправною точкою є нормалізація значень пікселів зображень у градаціях сірого, наприклад, масштабування їх до діапазону від 0 до 1. Це включає спочатку перетворення типу даних з цілих чисел без знака в числа з плаваючою комою, а потім поділ значень пікселів на максимальне значення;

– визначення моделі. Модель складається з двох частин: шари згортки та об'єднання, класифікатор, який здійснює прогноз;

– оцінка моделі. Модель оцінюється за допомогою *k*-кратної перехресної перевірки (поділ на групи, одна - контрольна група набору даних, інші - навчальні групи даних, підбір моделі до тестової групи та її оцінка, сума навичок

моделі, використовуючи вибірку оцінок).

Значення до $k = 5$ було обрано, щоб забезпечити базовий рівень як для повторної оцінки, так і для того, щоб воно не було настільки великим , щоб вимагати тривалого часу виконання.

Також описані підходи поліпшення алгоритму навчання, один з яких є нормалізація партії, як оцінка впливу пакетної нормалізації на нашу базову модель.

Так як в огляді наукової літератури були успішно використані ЗНМ, було прийнято рішення використовувати їх для вирішення поставлених завдань.

РОЗДІЛ 2. ТЕОРЕТИЧНА ЧАСТИНА

2.1 Згорткові мережі

ЗНМ - клас ШНМ, який став домінуючим у різних завданнях комп'ютерного зору і викликає інтерес у різних галузях. ЗНМ призначена для автоматичного та адаптивного вивчення просторових ієрархій об'єктів за допомогою зворотного розповсюдження з використанням кількох будівельних блоків, таких як шари згортки, що об'єднують шари та повнозв'язкові шари [13].

Згортковий шар є головною складовою частиною ЗНМ, і саме тут відбувається більша частина розрахунків. Це передбачає наявність кількох складових: вхідні дані, фільтр та карта об'єктів. Приклад роботи згорткового шару показаний на рис. 2.1.

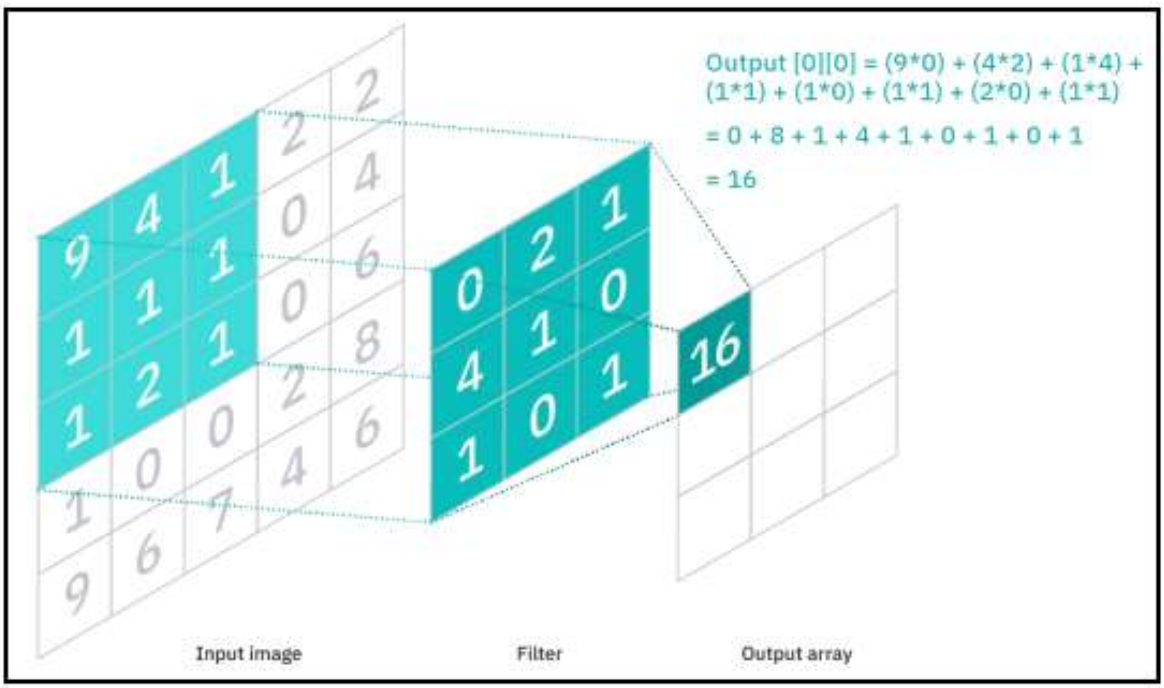

Рисунок 2.1 – Приклад роботи згорткового шару [14]

Об'єднуючий шар, або шар субдискретизації, знижує розмірність, шляхом зменшення числа параметрів на вході. Як і згортковий шар, об'єднувальна операція фільтрує дані по всьому входу, але відмінність полягає в тому, що цей фільтр не має ваги. Натомість ядро використовує функцію агрегування до

значень в приймаючому полі, заповнюючи масив на виході.

Назва повнозв'язкового шару точно описує себе. Як згадувалося раніше, значення пікселів зображення на вході не перебувають у безпосередньому зв’язку із вихідним шаром у власне частково пов'язаних шарах. Однак на повнозв'язковому рівні кожен вузол властиво вихідного шару напряму з'єднується з вузлом попереднього рівня.

Цей шар вирішує завдання класифікації на онові тих ознак, котрі вдалося видобути при допомозі попередніх шарів, а також їх різноманітних фільтрів. У той час як згорткові шари, а також шари об'єднання, як правило, використовують функції ReLu, повнозв'язкові шари, зазвичай, використовують функцію активації для відповідної класифікації вхідних даних, створюючи визначену ймовірність від 0 до 1 [14].

2.2 YOLOv5

Для реалізації системи було обрано архітектуру YOLO. Виявлення об'єктів у YOLO виконується як завдання регресії та надає ймовірності класів виявлених зображень. Цей алгоритм використовується ШНМ для виявлення об'єктів у режимі реального часу. Також він популярний через його швидкість і точність. YOLO використовується для детектування автомобільного трафіку, автомобільних знаків, тварин тощо.

Алгоритм YOLO працює, поділяючи зображення на N сіток, кожна з яких має область однакового розміру S x S. Кожна з цих N сіток відповідає за виявлення та локалізацію об'єкта, що міститься в ній.

Відповідно, ці сітки передбачають координати рамки, що обмежує, щодо координат їх осередків, а також мітку об'єкта і ймовірність присутності об'єкта в комірці. Але цей процес призводить до великої кількості прогнозів, що повторюються, через те, що кілька осередків прогнозують той самий об'єкт з різними прогнозами рамки, що обмежує. Щоб вирішити цю проблему, детектор використовує Non Maximal Suppression - YOLO пригнічує всі рамки, що обмежують, з більш низькими показниками ймовірності. YOLO досягає цього,

спочатку переглядаючи оцінки ймовірності, пов'язані з кожним рішенням, та вибираючи найбільшу з них. Після цього він пригнічує рамки, що обмежують, що мають найбільше перетин над об'єднанням, з поточною обмежувальною рамкою з високою ймовірністю.

YOLOv5 є open-source розробкою, котра містить множину моделей детектуванн об'єктів та методів детектування, які базуються на алгоритмі YOLO. Ці моделі пройшли завчасне навчання на наборі даних COCO [15].

Приклад роботи представлений на рис. 2.2.

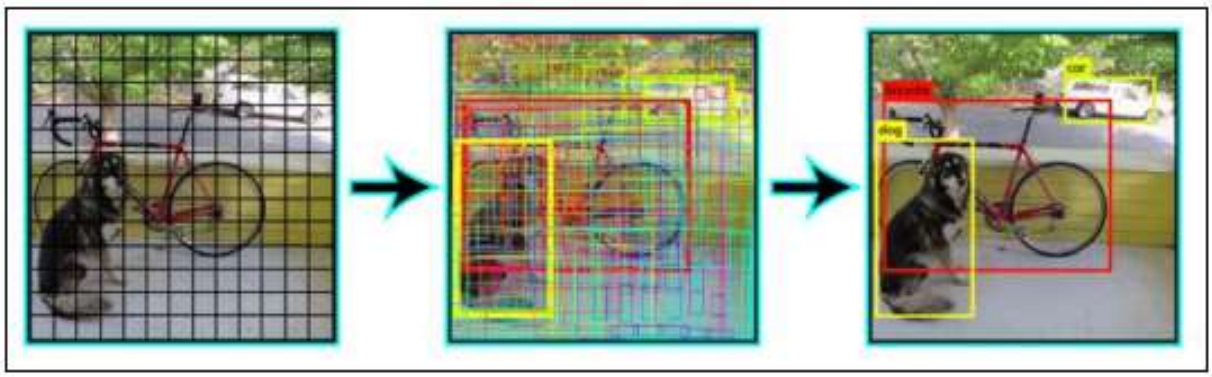

Рисунок 2.2 – Демонстрація роботи YOLO [15]

2.3 Вихідний набір даних

Для того, щоб ШНМ визначала елементи ланцюжка математичних символів потрібно розділити роботу на два етапи:

– розпізнавання повноцінної формули;

– детекція кожного символу зі знайденої області.

Для роботи алгоритму використовується два, окремо розмічені, набори даних: формули та математичні символи. У наборах є лише друковані символи. Спочатку в наборі даних з формулами було 321 зображення та 1 642 розмічені області, у наборі даних із символами відповідно 544 зображення та 6 664 області. Так само обидва набори були почищені від невірних та зайвих даних, таких як "other class" - 1781 область.

Для розмітки всіх даних використовувалась платформа Roboflow [16].

Приклад інтерфейсу платформи показаний на рис. 2.3.

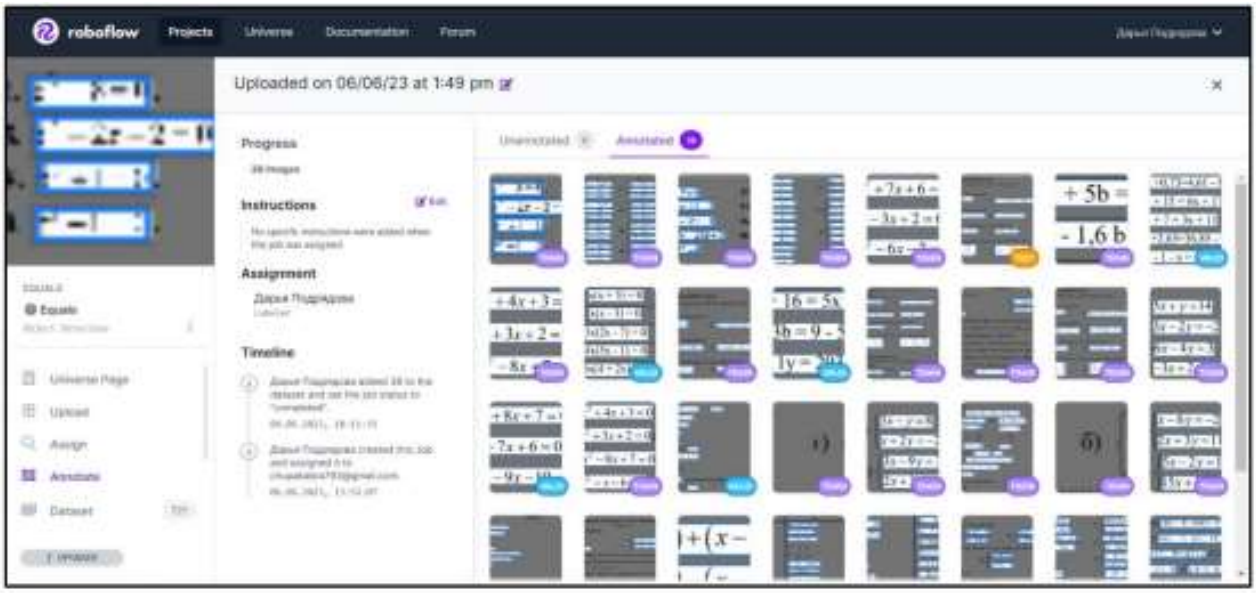

Рисунок 2.3 – Інтерфейс платформи Roboflow

2.4 Збір даних

2.4.1 Набір даних з цілими формулами

Дані для навчального набору даних були зібрані вручну та з відкритого джерела – платформи для розмітки даних Roboflow.

Приклад аркуша з лінійними рівняннями, створеними вручну, представлений на рис. 2.4.

а) x +3y = 1;	в) 2x – 5y = 7;
б) y – 2x + 6 = 0;	г) 5x + 3y -2 = 0.

Рисунок 2.4 – Приклад аркуша з рівняннями

З джерела були взяті зображення з чотирма різними класами рівнянь: тригонометрія, межі, інтеграли та диференціали [17]. Всі запозичені картинки були перевірені на якість і достовірність розмітки. Приклад зображення з

інтегральним виразом показаний на рис. 2.5.

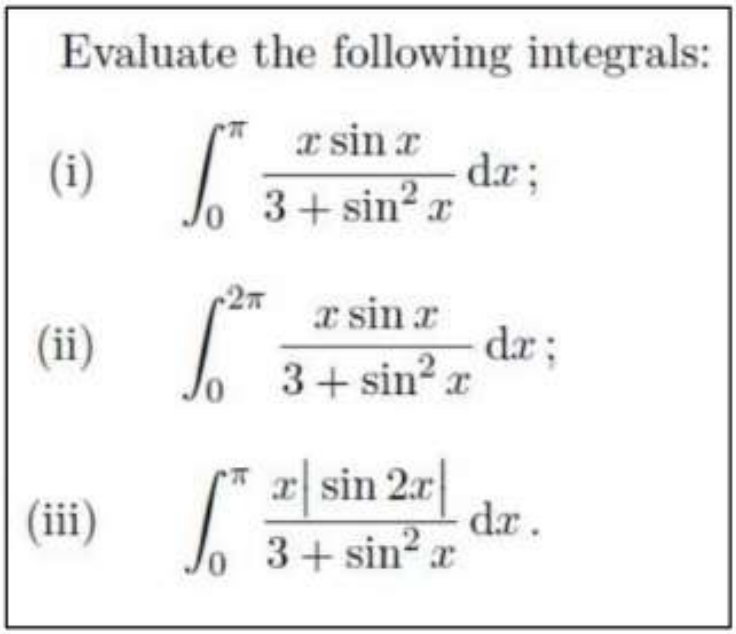

Рисунок 2.5 – Приклад зображення інтегральних виразів

Підсумковий навчальний набір даних складається з виразів з тригонометрією, диференціалами, інтегралами, межами та лінійними рівняннями. Загальна кількість зображень становила 721 штука.

2.4.2 Набір даних із окремими символами

З джерела з готовими наборами даних було взято зображення з різними класами символів [18]:

– числа 0-9;
– деякі змінні;
– синус, косинус, тангенс, котангенс;
– математичні оператори (+, -, *, /, =).

Приклад зображення з вихідного набору даних представлений на рис. 2.6.

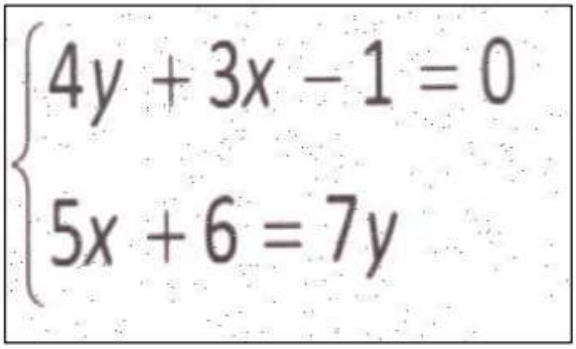

Рисунок 2.6 – Приклад зображення з набору даних із символами

Для набору даних з окремими символами також додавалися зображення. Приклад зображення з друкованими символами, яке додалося додатково представлено на рис. 2.7.

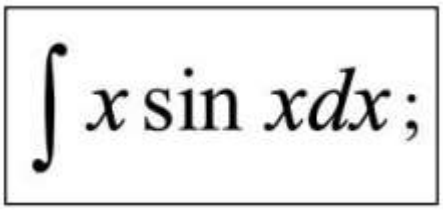

Рисунок 2.7 – Приклад розмітки друкованих символів

Загальна кількість зображень у наборі даних для ШНМ з окремими символами становила 899 штук.

2.5 Проектування застосунку

2.5.1 Функціональні вимоги

Є твердженням про фрагмент необхідної функціональності або поведінки, які система виявляє за певних умов, перелічених нижче [19]:

– користувач повинен мати можливість завантажити зображення, яке оброблятиметься;

– застосунок має вивести текстовий результат у повідомленні;

– розробка повинна вимагати від користувача лише зображення та відкидати інші формати файлів.

2.5.2 Нефункціональні вимоги

Є описом властивих властивостей або характеристик, які система повинна демонструвати, або обмеження, яких вона повинна дотримуватися, на відміну від спостерігається поведінки системи (зручність використання, безпека , надійність, розширюваність і т.д.), перерахованих нижче [20]:

– розробка має бути реалізована мовою Python 3.9;

– застосунок має використовувати ШНМ для розпізнавання математичних формул.

2.6 Діаграма структури системи

У застосунок включені основні компоненти:

– Main;

– DetectFormula;

– DetectSymb.

Діаграма компонентів системи представлена на рис. 2.8.

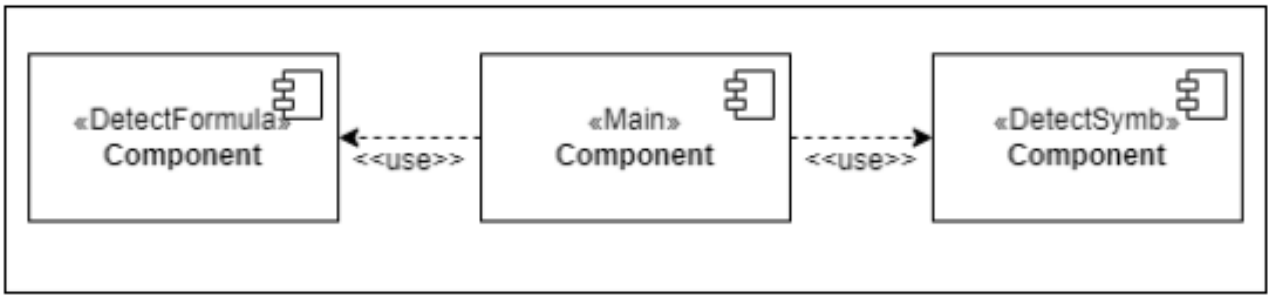

Рисунок 2.8 – Діаграма компонентів

Main - головний модуль, що ініціює взаємодію з користувачем і роботу системи в цілому. У даному модулі прописано функцію перекладу умовних позначень у загальноприйняті. Написано взаємодію з телеграм-ботом, обробка помилок та основний алгоритм обробки зображення та під’єднання нейромереж.

DetectFormula – компонент системи, який виявляє цілі формули. Також мною були додані команди, за допомогою яких про виявлені зображення «вирізуються» і поміщаються в папку з результатом для подальшого

використання іншою ШНМ.

DetectSymb - компонент системи, який детектує символи у формулі, котра була виявлена першою ШНМ, розташовує їх у потрібному порядку, за допомогою додатково написаних функцій. Також є функція, яка очищає результат від зайвої інформації, наприклад , подвійне виявлення одного і того ж символу або детекція точки з комою в кінці виразу, і функція визначення ступеня у виразі.

2.7 Програмні засоби реалізації

Використана мова програмування – Python, з версією 3.9. Для виконання роботи було підключено бібліотеки, наведені нижче.

Google Colaboratory [21] – хмарне середовище для роботи мовою Python у браузері.

PyCharm [22] – середовище на Python, що використовується для розробки та налагодження проекту. Версія середовища - community edition 2022.3.3.

Roboflow [23] - пакет для роботи з веб-сервісом Roboflow, що дозволяє виводити список інформації про робочі області, проекти, версії; завантажувати зображення у свої проекти; візуалізувати та зберігати прогнози, які були зроблені для моделі.

Matplotlib 3.2.2 [24] – бібліотека для створення статистичних, анімованих та інтерактивних візуалізацій на Python.

Numpy 1.18.5 [25] – бібліотека з підтримкою багатовимірних масивів, матриць, високорівневих математичних функцій для операцій із масивами.

Opencv-python 4.1.1 [26] – бібліотека для комп'ютерного зору, призначена для роботи із зображеннями. Використовується для завантаження та коректного відображення зображення у системі.

Pillow 7.1.2 [27] – бібліотека для роботи з обробкою зображення.

PyYAML 5.3.1 [28] – бібліотека для роботи із файлами формату yaml. Файл з розширенням .yaml використовується для налаштування набору даних.

Torch 1.7.0 [29] – бібліотека використовується для розробки та навчання

нейромережі, що базується на моделі глибинного навчання.

Torchvision 0.8.1 [30] – бібліотека з популярними наборами даних, архітектурами моделей та загальними перетвореннями зображень для комп'ютерного зору.

Tqdm 4.64.0 [31] – бібліотека для відображення рядка прогресу.

Tensorboard 2.4.1 [32] – інструмент для забезпечення візуалізації метрик, отриманих у результаті навчання ШНМ.

Pandas 1.1.4 [33] – бібліотека аналізу даних, необхідна для структурованої візуалізації даних у процесі навчання.

Seaborn 0.11.0 [34] – бібліотека для візуалізації даних. Використовується для відображення результатів навчання.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ЗАСТОСУНКУ

3.1 Розмітка даних

Розмітка набору даних з друкованими символами проводилася через виділення кожного символу у свою рамку та призначення йому відповідного класу. Приклад показаний на рис. 3.1.

Рисунок 3.1 – Приклад розмітки рукописних символів

Розмітка формул проводилася аналогічно, приклад наведений на рис. 3.2.

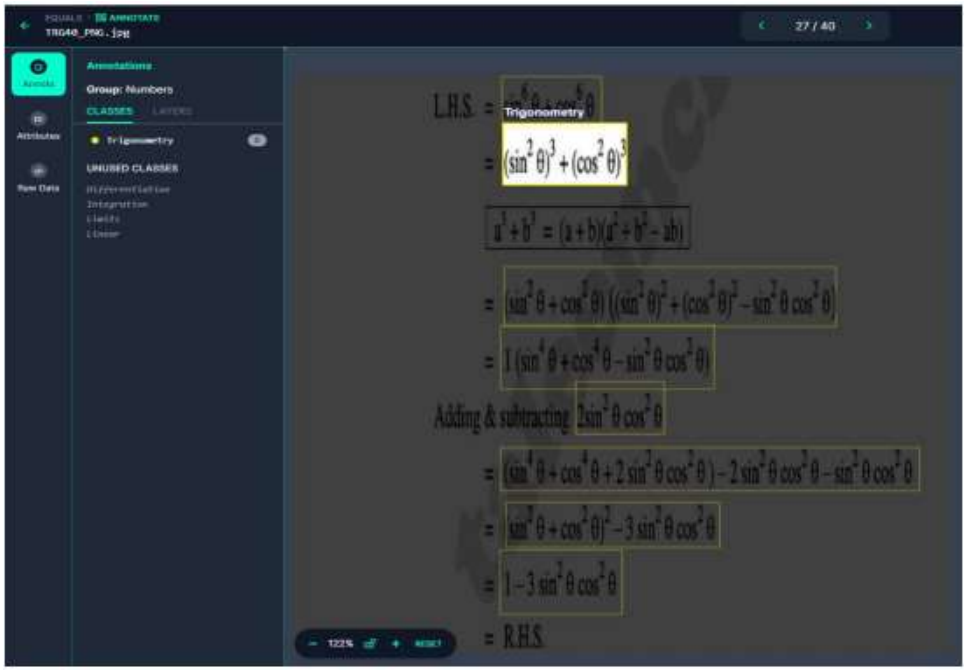

Рисунок 3.2 – Приклад розмітки рукописних формул

Детальну характеристику набору даних можна побачити на балансі класів [35]. Баланс формул представлений на рис. 3.3.

Рисунок 3.3 – Баланс кількості областей друкованих формул

Знайдений набір даних із символами був розмічений некоректно і для використання розмітка була перевірена та виправлена. Приклад неправильної розмітки подано на рис. 3.4 і правильно на рис. 3.5.

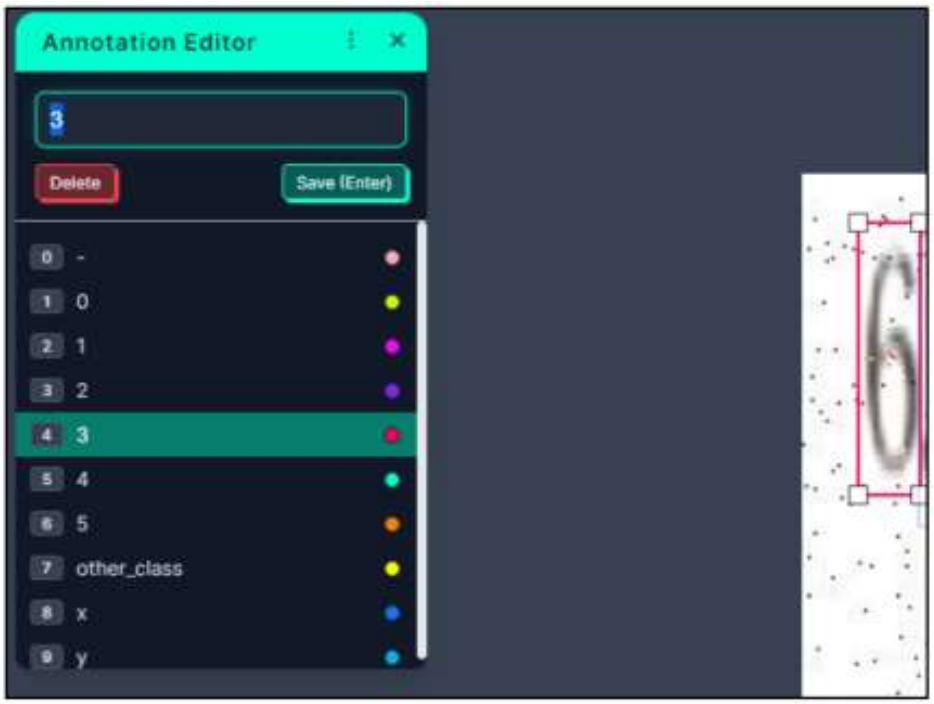

Рисунок 3.5 – Приклад неправильної розмітки. Цифра 6 позначена як 3

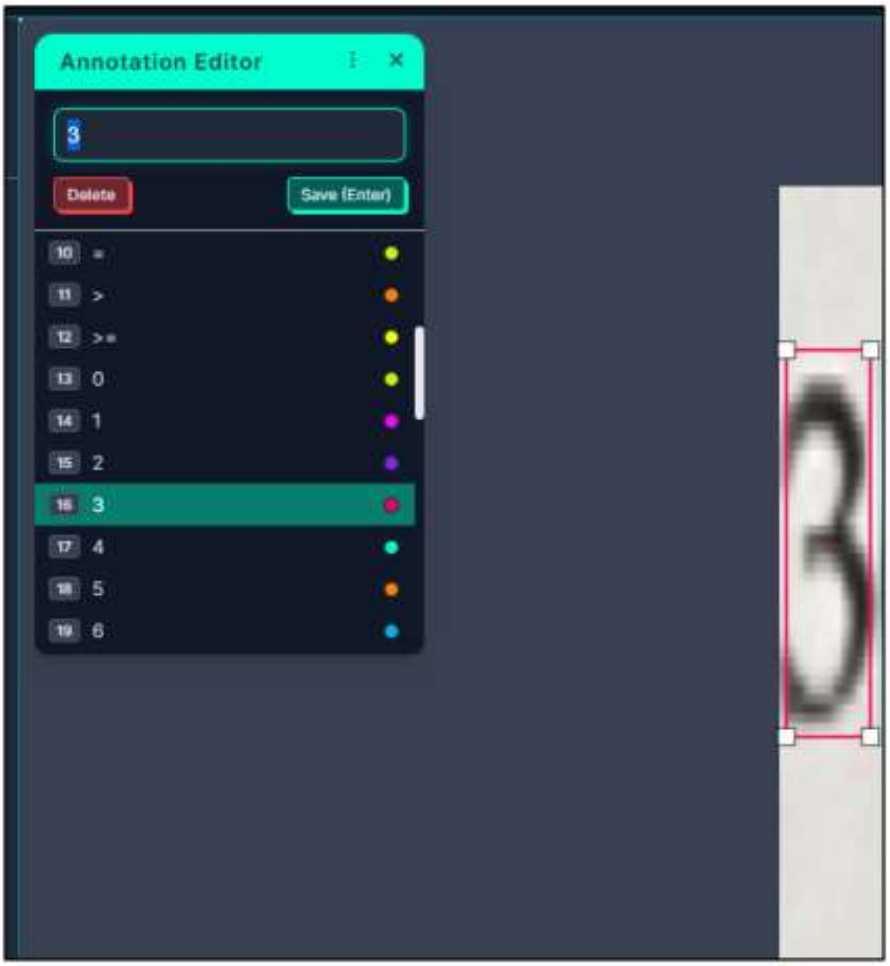

Рисунок 3.6 – Приклад виправленої розмітки

Детальну характеристику набору даних можна побачити на балансі класів [36]. Баланс друкованих символів представлений на рис. 3.7.

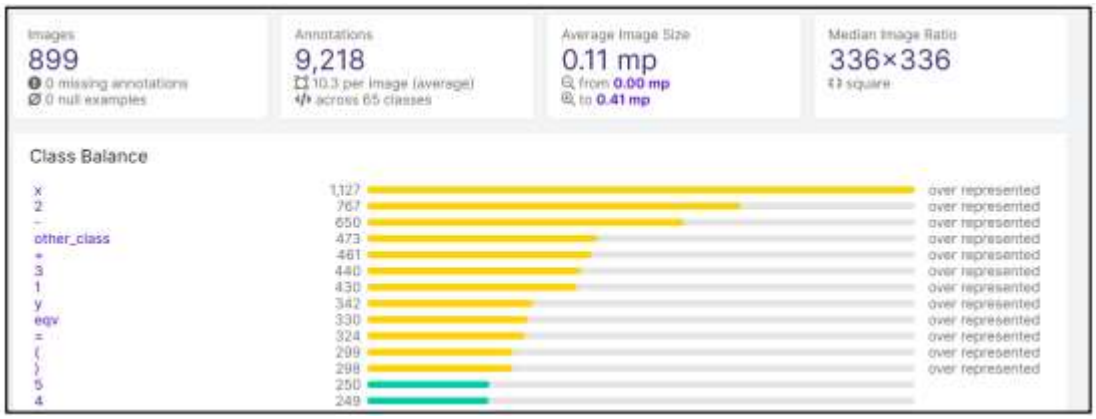

Рисунок 3.7 - Баланс кількості областей друкованих символів

На розмітку і корекцію всіх наборів даних було витрачено сумарно приблизно 250 годин, оскільки після аналізу результатів навчання та

детектування набори наново збиралися кілька разів. На кожному зображенні є кілька формул або символів. Майже всі зображення чорно-білого кольору.

3.2 Файл набору даних для навчання

Після підготовки, набір даних вивантажується з платформи у вигляді пакета файлів, що містить: папку із зображеннями, папку з файлами текстового формату з параметрами розмітки і файлом конфігурації.

Приклад файлу з параметрами навчання представлений на рис. 3.8, а саме зображення – на рис. 3.9.

Рисунок 3.8 – Файл розмітки

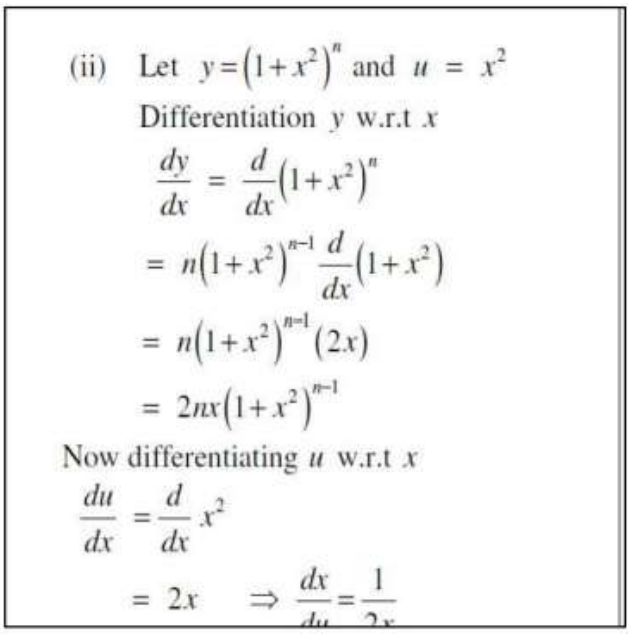

Рисунок 3.9 – Зображення до файлу розмітки

На першій позиції кожного рядка написано номер назви класу у загальному списку. Цей список позначений у файлі data.yaml. Приклад файлу показано на рис. 3.10.

data.yaml

```
names:
- Differentiation
- Integration
- Limits
- Linear
- Trigonometry
nc: 5
roboflow:
  license: CC BY 4.0
  project: equals-pzmaq
  url: https://universe.roboflow.com/equals/equals-pzmaq/dataset/2
  version: 2
  workspace: equals
test: ../test/images
train: /content/gdrive/MyDrive/yolov5/datasets/Equals-2/train/images
val: /content/gdrive/MyDrive/yolov5/datasets/Equals-2/valid/images
```

Рисунок 3.10 – Файл конфігурації

3.3 Особливості створення набору даних у Roboflow

Дослідницьким шляхом виявили, що система Roboflow працює не коректно, якщо клас у наборі даних назвати одним символом. Наприклад, замість знака "+" потрібно записати назву символу - "Plus". В іншому випадку навчання моделі буде проведено неправильно і всі назви класів із символами будуть замінені на прочерк. Результат неправильного навчання та подальшого детектування подано на рис. 3.11.

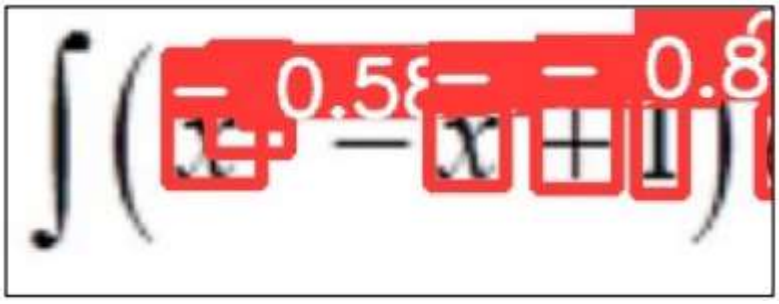

Рисунок 3.11 – Результат неправильного навчання

3.4 Архітектура моделі YOLOv5s

Для навчання на даних і реалізації детектування було обрано попередньо навчену модель YOLOv5s. Відповідно до дослідження [37], в якому наводиться порівняння швидкостей навчання, найвищою швидкістю і найменшими витратами по пам'яті має YOLOv5s. Тому було ухвалено рішення навчати цю модель. Графік представлений на рис. 3.12.

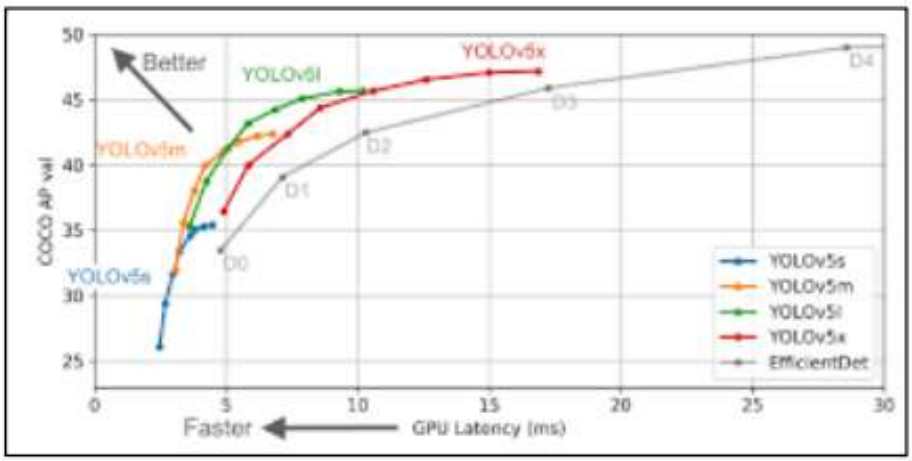

Рисунок 3.12 – Графік швидкості моделей навчання

На рис.3.13 представлений частина програмного коду YAML -файлу, що описує архітектуру моделі.

```
# Parameters
nc: 80  # number of classes
depth_multiple: 0.33  # model depth multiple
width_multiple: 0.50  # layer channel multiple
anchors:
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32

# YOLOv5 v6.0 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Conv, [64, 6, 2, 2]],  # 0-P1/2
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4
   [-1, 3, C3, [128]],
   [-1, 1, Conv, [256, 3, 2]],  # 3-P3/8
   [-1, 6, C3, [256]],
   [-1, 1, Conv, [512, 3, 2]],  # 5-P4/16
   [-1, 9, C3, [512]],
   [-1, 1, Conv, [1024, 3, 2]],  # 7-P5/32
   [-1, 3, C3, [1024]],
```

Рисунок 3.13 – Фрагмент коду

3.5 Навчання моделей YOLOv5s

Навчання проводилося окремо для двох різних завдань: детектування та класифікація формул і детектування та класифікація символів. Тому необхідно було навчити дві ШНМ з однаковою топологією та різними наборами даних.

Навчання проходило у хмарному сервісі Google Colaboratory. У YOLOv5s використано: оптимізатор - SGD, функції активації - SiLU і Sigmond, функція втрат - FocalLoss.

Над обома наборами даних була проведена аугментація - штучне збільшення набору даних за рахунок зміни кута нахилу або якості зображення.

Навчання ШНМ для детектування формул. Для навчання моделі розпізнавання формул було використано приблизно 1300 зображень з формулами. У відсотковому співвідношенні - 84% від основного набору даних.

За результатами експериментів навчання було обрано модель, навчену на 50 епохах з наступними метриками.

Значення: mAP – 0,983, precision – 0,961, recall – 0,950.

Графічне подання зміни метрик зображено на рис. 3.14.

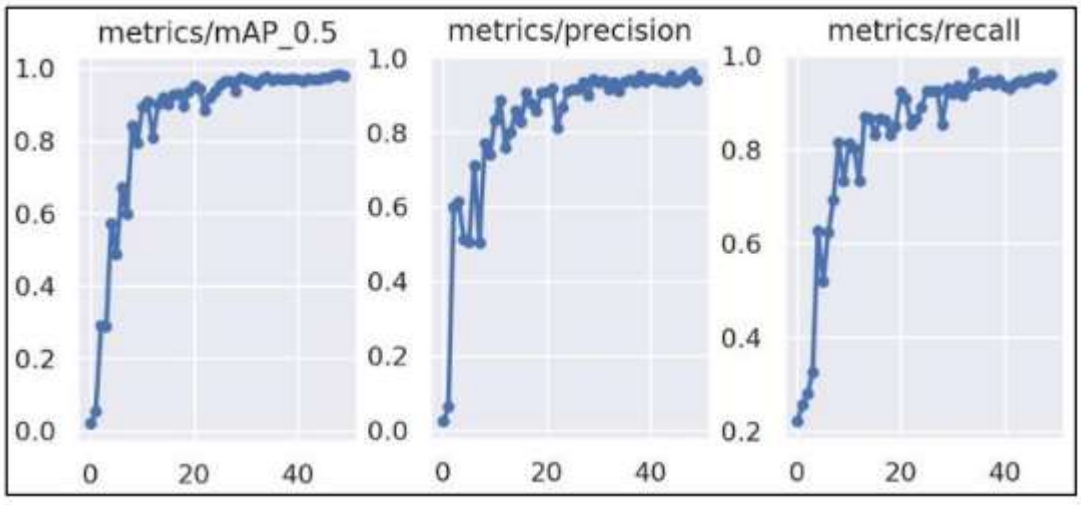

Рисунок 3.14 – Графічне представлення метрик для детектування формул

Навчання ШНМ для детектування символів. Для навчання моделі розпізнавання символів було використано приблизно 2200 зображень із

символами або групою символів, наприклад: sin, cos, tg, ln і т.д. У відсотковому співвідношенні - 93% від основного набору даних.

За результатами експериментів навчання було обрано модель, навчену на 100 епохах з наступними метриками.

Значення: mAP – 0,898, precision – 0,905, recall – 0,843.

Графічне подання зміни метрик зображено на рис. 3.15.

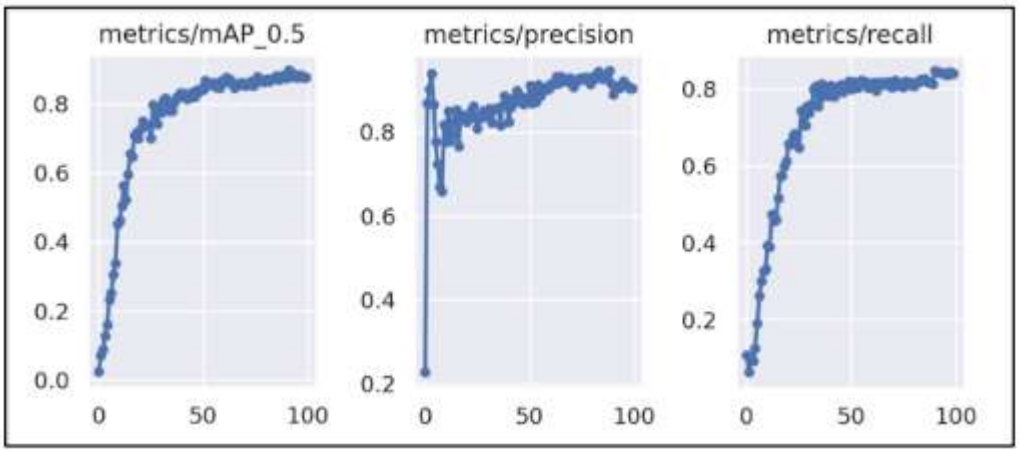

Рисунок 3.15 – Графічне представлення метрик для детектування символів

3.6 Інтерфейс взаємодії з користувачем

Для взаємодії користувача з системою був використаний телеграм-бот. Програмна реалізація робота представлена в додатку А.

Для запуску бота потрібно ввести в пошуковий рядок його назву - @NumDetectorVKR_bot.

Почати роботу системи можна з команди \start, будь-якого текстового повідомлення або відправленої фотографії. Відповідь системи на всі дії користувача веде або до запуску або відразу запускає системи. Робота продемонстрована на рис. 3.16.

Рисунок 3.16 – Відповідь на текстове повідомлення

Код реалізації наведений на рис. 3.17.

```
@bot.message_handler(commands=['start'])
def start_message(message):
        bot.send_message(message.chat.id, 'Привіт! Вишліть ваше фото!\nДокументи не приймаються!')

@bot.message_handler(content_types=['text','sticker', 'video',
'video_note', 'emoji'])
def send_welcome(message):
        bot.send_message(message.chat.id, 'Привіт! Вишліть ваше фото!\nДокументи не приймаються!');
```

Рисунок 3.17 – Код реалізації стартової команди

Застосунок не приймає фотографії у вигляді документа, у відповідь файл відправляється повідомлення з короткою інструкцією правильної відправки фото. Роботу застосунку представлено на рис. 3.18. Код реалізації показано на рис. 3.19.

Рисунок 3.18 – Відповідь застосунку на документ

```
@bot.message_handler(content_types=["document"])
def no_photo(message):
        bot.send_message(message.chat.id,'Я приймаю тільки фотографії, а не
документи\nЯкщо ви хотіли відправити фото, а вийшов документ,
будь ласка поставте відмітку у вікні "Стиснути зображення" ');
```

Рисунок 3.19 – Реалізація коду для відмови від зображення у форматі документа

Завершення роботи застосунку сигналізується повідомленням «Готово!» Результат роботи подано на рис. 3.20.

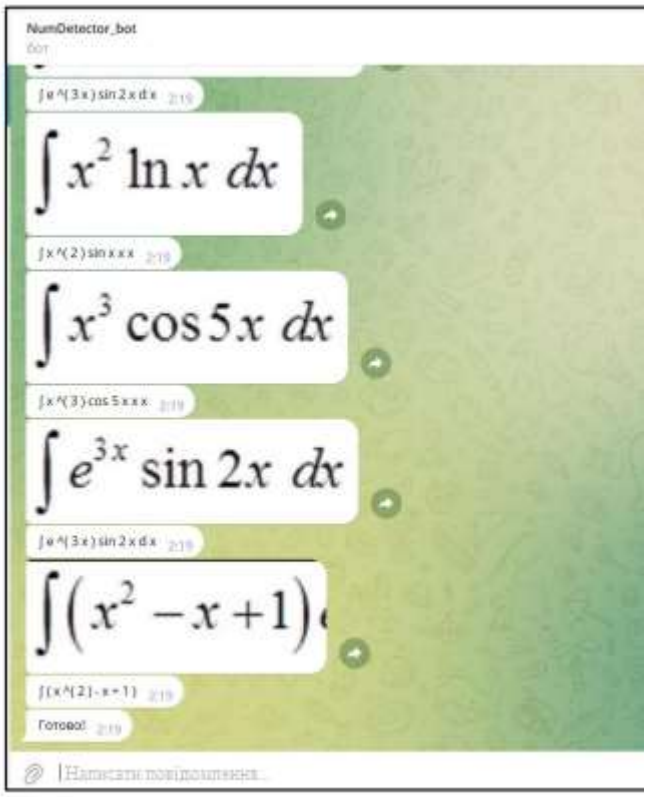

Рисунок 3.20 – Кінець роботи застосунку

Основний блок роботи з фотографією з обробкою помилки представлений у додатку А.

Також були реалізовані функції для обробки результатів детектування нейромереж. Функції основного модуля представлені у додатку А.

У модулі детектування символів, крім під’єднаної ШНМ, додатково були реалізовані функція видалення зайвих рамок детектування і функція виявлення степеня в результаті. Функції представлені на рис. 3.21 та 3.22 відповідно.

```
def delete_double_frame(list):
    b = []
    s = list[0].get('mid_y')

    for i in range(1, len(list)):
        if   list[i].get('x1s')    -    list[i-1].get('x1s')    <    3:#    and
list[i].get('x1s') not in b:
            b.append(list[i].get('x1s'))
        if len(b) == 1:
        for i in range(len(list)):
            if list[i].get('x1s') == b[0]:
                list.pop(i)
                break
    else:
        pp = len(list)
        for i in range(pp-len(b)):
            for j in range(len(b)):
                if list[i].get('x1s') == b[j]:
                   # print(list[i].get('x1s'))
                    list.pop(i)
                   # print(list)
```

Рисунок 3.21 – Функція видалення зайвих рамок

```
def get_power(list):
    s = list[0].get('mid_y')

    for i in range(1,len(list)):
        s += list[i].get('mid_y')

    mid_y = s/len(list)
    c = []

    for i in range(len(list)):
        if list[i].get('mid_y') < mid_y*0.9:
            c.append(i)

    for i in range(len(c)):
        if i == 0 or (i != 0 and c[i] - c[i-1] > 1):
            list.insert(c[i], {'x1s': list[c[i]].get('x1s') - 1, 'mid_y':
list[c[i]].get('mid_y'), 'class': '^('})
```

Рисунок 3.22 – Фрагмент коду функції визначення степеня

3.7 Результати тестування

3.7.1 Функціональне тестування застосунку

Функціональне тестування перевіряє відповідність вимогам системи, що проектується. У табл. 3.1 представлено тестування на функціональність

Таблиця 3.1 – Функціональне тестування

Назва тесту	**Кроки**	**Очікуваний результат**	**Тест пройдено?**
Надсилання непідтримуваного формату даних (документ, аудіо, відео, стікери, текст)	1. Перейти до Telegram бота. 2. Додати файл невідповідного формату. 3. Надіслати повідомлення.	Бот повідомляє, що не може обробляти подані дані.	Так
Надсилання зображення з формулами	1. Перейти до Telegram бота. 2. Додати зображення з кількома формулами. 3. Надіслати повідомлення.	Бот повертає попарно: ряд зображень і текстове повідомлення з виявленими символами.	Так
Надсилання зображення без формул	1. Перейти до Telegram бота. 2. Додати зображення без формул. 3. Надіслати повідомлення.	Бот повідомляє, що формули не виявлені.	Так
Надсилання зображення низької якості символів	1. Перейти до Telegram бота. 2. Додати зображення низької якості символів 3. Надіслати повідомлення.	Робот виділяє формулу , далі повідомляє, що символи не знайдені.	Так

Виходячи з результатів функціонального тестування розроблений застосунок відповідає функціональним вимогам.

На рис. 3.23 наведено приклад роботи програми з нейромережею.

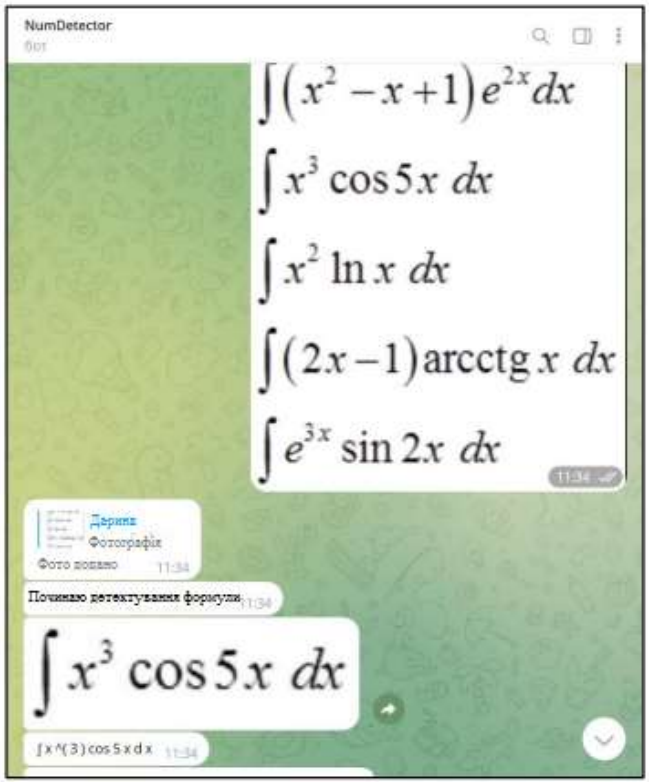

Рисунок 3.23 – Приклад роботи застосунку

3.7.2 Тестування нейронної мережі

У цьому підрозділі продемонстровано роботу нейронмереж, навчених на наборах даних для детектування формул і символів окремо. Модель навчена з результатами mAP: 0,983 - для формул і 0,898 - для символів. Метрика mAP означає середню точність для всіх класів набору даних. Для формул було виділено 5 класів (інтеграли, лінійні рівняння, диференціальні рівняння, тригонометричні вирази, границі) та для символів - 23 (числа 0-9, знаки: /, *, +, -, =, знак інтеграла, дужки, змінні: x, y, z, t, d).

На рамці, що визначає об’єкт, є два значення: назва класу і значення метрики mAP. Приклад рамки показано на рис. 3.24.

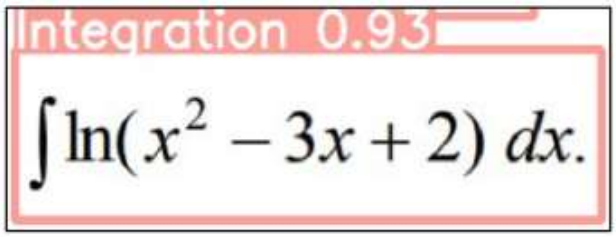

Рисунок 3.24 – Приклад рамки, котра визначає

Приклад тестування нейромережі на виявлення формул наведено на рис. 3.25.

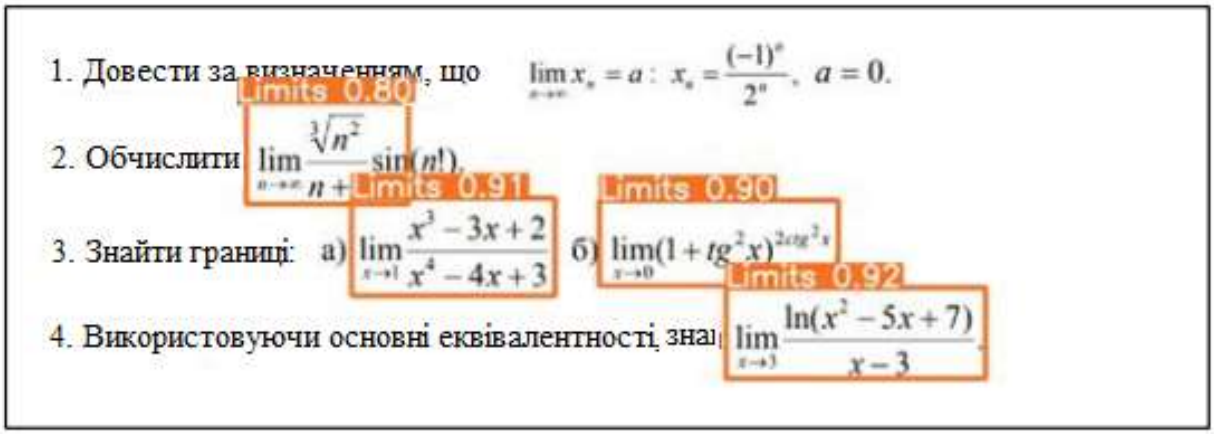

Рисунок 3.25 – Приклад детектування формул

На рис. 3.26 та 3.27 наведено приклади роботи нейронної мережі для виявлення символів.

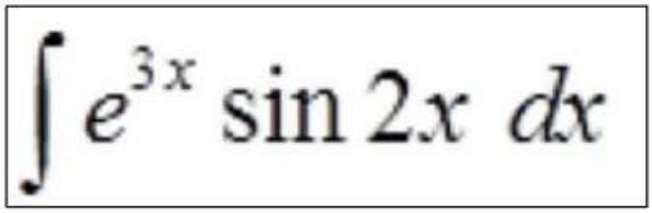

Рисунок 3.26 – Фрагмент початкового зображення з формулою

Рисунок 3.27 – Результат роботи нейронної мережі

3.7.3 Експеримент щодо навчання нейронних мереж

Для того, щоб отримати значення оптимальної кількості епох для навчання нейронних мереж, було проведено експеримент. Навчання кожної ШНМ

проводилося кілька разів з однаковим значенням параметрів, крім кількості епох, воно змінювалося з 50 до 150. Результати експериментів представлені на табл. 3.2 та 3.3.

Таблиця 3.2 – Результати експериментів навчання ШНМ для формул

Кількість епох	**Час навчання /хв.**	**mAP**	**PRECISION**	**Recall**
150	31,8	0,985	0,962	0,955
100	20,9	0,982	0,958	0,954
75	16,3	0,983	0,968	0,934
50	**10,8**	**0,983**	**0,961**	**0,950**

Таблиця 3.3 – Результати експериментів навчання ШНМ для символів

Кількість епох	**Час навчання /хв.**	**mAP**	**PRECISION**	**Recall**
150	82,3	0,881	0,936	0,833
100	**57,3**	**0,898**	**0,905**	**0,843**
75	42,8	0,863	0,903	0,820
50	28,5	0,850	0,876	0,817

В результаті експериментів, виявлено, що найефективнішим було навчання з такими параметрами: ШНМ для детектування формул - 50 епох, ШНМ для визначення символів - 100 епох.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Долікарська допомога при ураженні електричним струмом

Удар електричним струмом є поширеною травмою і часто може закінчитися летальним випадком. Ураження електричним струмом відбувається підчас контакту тіла людини з джерелом електричної енергії під напругою. Це реакція організму людини на проходження електричного струму через тіло. Вона проявляється по різному, від легких потрясінь до небезпечних здоров'ю травм, які можуть вплинути на тканини в організмі.

Шкода завдана електричним струмом залежать від кількох факторів: наскільки висока була напруга, яка область тіла була вражена і від виду струму. Фізичні наслідки для людини можуть коливатися від опіків частин тіла до серйозних вражень внутрішніх органів [38].

Часто люди стикаються з підвищеним ризиком ураження високою напругою. Низька напруги не наносить серйозних травм для люди, а з іншої сторони висока напруги, яка має більше 500 В, може призвести до серйозного пошкодження тканин. У дітей або підлітків при враженні електричним струмом в діапазоні від 110 до 200 В можуть вникнути значні травми. Зазвичай це трапляється підчас порушення техніки безпеки при роботі з електричними приладами, які є в побуті. Це можуть бити електричні шнури, подовжувачі, несправні розетки та багато чого іншого.

Виділяють чотири основні фактори від яких залежить вплив електричного струму на організм: величина струму, що протікає через тіло; органи, через які проходить струм; час, протягом якого струм вражає тіло; частота струму.

Фізичні наслідки на пряму залежать від величини струму, яка вражає тіло людини. Струм менший за 1 мА не завдає жодної шкоди людина фізичного ефекту. При 1 мА можуть відчуватися слабкі поколювання, а при 5 мА – легкий поштовх. Однак при струмі ведичною від 6 до 25 мА людина може відчувати больовий шок і деяку втрату контролю над м'язами.

При ураженні електричним струмом від 50 до 150 мА може спричинити у людини сильний біль, м’язове скорочення і навіть призвести до зупинку дихання. У певних випадках можливий летальний результат для людини. Струм від 1000 до 4300 мА призводить до ймовірної смерть, тому що дана напруга спричиняє пошкодження нервових зав’язків, м’язових скорочення та порушення ритму серця. До 10 000 мА електричний струм спричиняє важкі опіки шкіри людини та зупинку серця. Висока ймовірність смерті.

Найпоширеніші ознаки та симптоми враження електричним струмом включають:

– втрата свідомості;
– ускладнення або зупинка дихання;
– опіки, які виникають там, де струм входить і виходить з тіла;
– зупинка серця;
– слабкий і непостійний пульс або його припинення.

Перш за все, що потрібно зробити при першій допомозі, це відключити джерело живлення. Вимкнути електропостачання, від’єднати електроприлад від джерела електричного струму або вимкнути блок запобіжників, якщо він знаходиться неподалік. Не потрібно намагатися підходити близько до жертви, якщо не переконані, що це безпечно і живлення вимкнено.

Потрібно бути обережним у вологих місцях, тому що вода є електричним провідником і рятівник може стати теж жертвою. Якщо людина не впевнена щодо вологості поверхні, потрібно відключити основне електропостачання будинку. У випадку коли це неможливо, використати підручний предмет, який не є провідником і відокремити людину від джерела струму. Це може бути дерев’яна або пластмасова річ.

Після того як постраждалого відокремили від джерела електрики, потрібно викликати швидку і надати першу медичну допомогу. Далі потрібно визначити стан жертви. Перевірити, чи людина у свідомості і дихає. У складних випадках у жертви може бути слабкий пульс або його відсутність. Можливо, що дихання зупинилося. Якщо людина втратила свідомість і перестала дихати, потрібно почати серцево-легеневу реанімацію. Руки розташовуємо в центрі грудної, одна

на іншу. Сильно і швидко натиснути 30 раз приблизно до третини діаметра грудної клітки. Після кожного натискання на грудну клітку робиться два рятувальні вдихи. Потрібно відкинути голову потерпілого назад і підняти підборіддя. Затиснути ніс і створити повне ущільнення. Далі подути потерпілому в рот і подивитися, чи підніметься грудна клітка. Потрібно продовжувати робити натискань на грудну клітку та вдих, поки не прибуде медична допомога або людина не почне сама дихати. Якщо потерпілий живий, перемістити його у зручне йому положення подальше від небезпеки. Можна запобігти шоку, поклавши людину рівно на землю, з головою трохи нижче тіла [39].

Якщо людина при свідомості, нормально дихає і на тілі є опіки, потрібно накрити їх звичайною харчовою плівкою або іншою неклейкою пов’язкою, але без мазі чи лосьйону. Якщо кровотеча у потерпілого, може знадобитися компресія та джгут.

При роботі з соціальною мережею користувач повинен знати і вміти як правильно поводитися з ПК, тому що людина перебуває у непосередньому контакті з джерелом напруги. Удар електричним струмом є потенційно смертельною травмою. Негайна медична допомога важлива, щоб запобігти серйозним травмам і смерті. Для запобігання уражень електричним струмом при роботі за ПК слід встановити додаткові захисні пристрої, що забезпечують недоступність токопровідних частин для дотику. Для зменшення небезпеки використовувати розділовий трансформатор для розв'язки з основною мережею.

Удар електричним струмом є потенційно смертельною травмою. Негайна медична допомога важлива, щоб запобігти серйозним травмам і смерті.

4.2 Вимоги ергономіки до організації робочого місця оператора ПК

Робоче місце – це ділянка простору, яка облаштована необхідним обладнанням, відповідно до трудової діяльності, для виконання поставлених завдань.

Правильно побудоване робоче місце повинне забезпечувати:

– найкраще розміщення обладнання і предметів праці;

– не допускати дискомфорту;

– підвищувати продуктивність праці;

– зменшувати втому працівника.

Розмір робочого місця повинен бути таким, щоб людина не виконувала лишніх рухів і не відчувала дискомфорту під час виконання роботи. Також для працівника важливо мати змогу змінити робочу позу, наприклад, положення тулуба, рук або ніг. Потрібно мінімізувати або звести до нуля всі незручності положення тіла [40].

Різні дослідження заявляють, що при правильному проектуванні робочого місця продуктивність людини може зрости від 15-25%. Такі фактори як рівень освітлення, вологість повітря, температура, шум, вібрація, токсичність, мають значний вплив на умови життєдіяльності і працездатності людини.

Антропометричні вимоги визначають відповідність робочого місця до фізіологічних параметрів тіла людини як зріст і розміри тіла. Індикатором цього є правильна робоча поза, відсутність дискомфорту, оптимальні зони досягнення, раціональні рухи. Психофізіологічні та фізіологічні вимоги формулюють відповідність обладнання і робочого місця можливостям співробітника щодо розуміння, обробки даних, пошук і реалізації рішень.

Організація робочого місця передбачає наступні пункти:

– раціональне положення робочого місця у приміщенні;

– вибір робочих меблів відповідно до фізіологічних характеристик працівника;

– правильне компонування і розміщення обладнання на робочих місцях;

– урахування особливостей та характеру професійної діяльності.

До загальних принципів організації робочого місця відносять:

– робоче місце повинне містити тільки ті предмети, які беруть учать у робочому процесі, але не заважати йому;

– предмети, які часто використовуються у роботі, розміщуються ближче, ніж ті речі, якими користуються рідше;

– предмети, які беруться лівою рукою, повинні розміщуватися зліва, а предмети, які використовуються правою рукою — справа;

– якщо при роботі з предметом працівник використовує дві руки, то він розміщується з урахуванням зручності захоплювання його двома руками;

– робоче місце не повинно бути засмічене;

– необхідна оглядовість повинна бути забезпечена при правильній організації робочого місця.

Робоча поза – це найбільш тривале положення тіла працівника протягом робочого дня. При зручній робочій позі забезпечується стійкість положення тулуба, ніг, рук, голови і витрачається мінімальний запас енергії та максимальну продуктивність [41].

Сидячи і стоячи – дві найпопулярніших пози у робочому процесі. При проектуванні робочого місця потрібно враховувати, що з фізичним навантаженням бажана поза стоячи, а при малих зусиллях – сидячи. При роботі стоячи, людина стомлюється більше ніж сидячи. У відсотковому еквіваленті це на 10% більше енергії. При додатковому навантаженні підвищується артеріальний і венозний тиск крові, розширення вен, пошкоджуються ступені та викривляється хребет. У свою чергу при сидячій роботі нижня частина тіла розслаблена, а основне навантаження спрямоване на м'язи шиї, спини, таза, стегон. При неправильній сидячій позі розвивається застій крові у ногах, а якщо пальці виконують багато роботи можливе запалення суглобів.

Організація робочого місця при використанні ПК повинна відповідати усім ергономічним вимогам. Ключові ергономічні вимоги до проектування робочого місця оператора ПК зображені на рисунку 4.1.

При роботі з персональним комп’ютером потрібно:

– зменшувати кількість статичних напружень;

– розподіляти кількість і час статичних напружень;

– змінювати робочі пози під професійної діяльності.

Саме вибір правильної робочої пози визначається від впливу багатьох факторів. Одні з найважливіших це – кількість зусиль яка прикладається, велична робочої зони, відношення висоти робочої поверхні і ростом працівника.

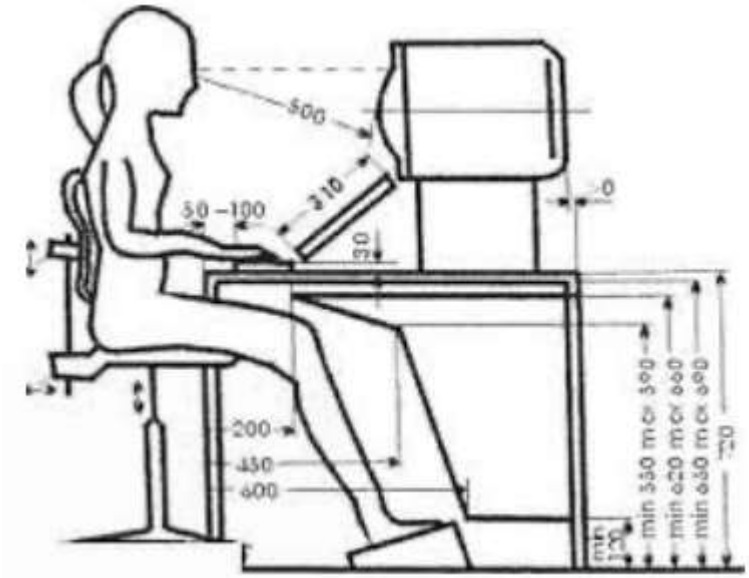

Рисунок 4.1 – Робочий стіл і розміщення користувача ПК

При використанні соціальної мережі користувач повинен дотримуватися вище зазначених правил, які будуть сприяти комфортній і продуктивній роботі. Важливо регулярно робити короткі перерви. Часта зміна заняття – кращий спосіб уникнути можливих неприємностей.

ВИСНОВКИ

В результаті виконання цієї роботи було розроблено програмний засіб у вигляді мобільного застосунку для розпізнавання рукописних математичних формул.

Основні результати, отримані в роботі:

– проведено огляд наукової літератури та аналогів для розпізнавання математичних формул з використанням ЗНМ;

– підготовлено два навчальні набори даних;

– реалізовано дві ШНМ, проведено їх навчання та тестування;

– спроектовано та реалізовано застосунок для розпізнавання математичних формул;

– проведено тестування реалізованого ПЗ.

Подальше розширення функціоналу розробки полягає у додаванні можливості обробки:

– дробів;

– диференціальних рівнянь (лапки, посимвольно);

– тригонометричних виразів (градуси, радіани);

– визначених інтегралів.

визначених

ПЕРЕЛІК ДЖЕРЕЛ

1. Що таке нейронна мережа і як вона працює? [Електронний ресурс] – Режим доступу: https://mc.today/uk/shho-take-nejronna-merezha/ (дата звернення: 20.03.2024).

2. Класифікаційні метрики [Електронний ресурс] – Режим доступу: http://www.andriystav.cc.ua/Downloads/MITER/Lecture_04.pdf (дата звернення: 20.03.2024).

3. Нейроподібні методи, алгоритми та структури обробки зображень у реальному часі: монографія / Ю. М. Рашкевич, Р. О. Ткаченко, І. Г. Цмоць, Д. Д. Пелешко ; НУ Львівська політехніка. – Львів: Вид-во Львів. політехніки, 2014. – 256 с. : іл.

4. Microsoft Math Solver. [Електронний ресурс] – Режим доступу: https://math.microsoft.com (дата звернення: 07.04.2024).

5. PhotoMath. [Електронний ресурс] – Режим доступу: https://photomath.com (дата звернення: 07.04.2024).

6. OneNote. [Електронний ресурс] – Режим доступу: https://www.microsoft.com/microsoft-365/onenote/digital-notetaking-app (дата звернення: 07.04.2024).

7. Що таке OCR та навіщо його використовують? [Електронний ресурс] – Режим доступу: https://sparsim.org/uk/shcho-take-ocr-ta-navishcho-yoho-vykorystovuiut/ (дата звернення: 07.04.2024).

8. Gadgetshelp.com – Чому потрібно перейти з OneNote 2016 на OneNote для Windows 10. [Електронний ресурс] – Режим доступу: Https://Gadgetshelp.Com/Windows/Pochemu-vy-Dolzhny-Pereiti-s-Onenote-2016-Na-Onenote-Dlia-Windows-10/, n.d. (дата звернення: 07.04.2024).

9. Ahlawat S., Choudhary A., Nayyar A., Singh S., Yoon B. Improved handwritten digit recognition using convolutional neural networks (Cnn) // Sensors (Switzerland), 2020. – vol. 20, no. 12. – 14 с.

10. Bhagyashree P.M., Likhitha L.K., Rajesh D.S. Handwritten Digit Recognition Using Deep Learning. // International Journal of Scientific Research in

Science and Technology, Jul. 2021. – 153–158 pp.

11. How to Develop a CNN for MNIST Handwritten Digit Classification. [Електронний ресурс] – Режим доступу: https://machinelearningmastery.com/how-todevelop-a-convolutional-neural-network-from-scratch-for-mnist-handwrittendigit-classification/. (дата звернення: 12.04.2024).

12. TensorFlow – MNIST. [Електронний ресурс] – Режим доступу: https://www.tensorflow.org/datasets/catalog/mnist (дата звернення: 12.04.2024).

13. SpringerOpen – Convolutional neural networks: an overview and application in radiology. [Електронний ресурс] – Режим доступу: https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9 (дата звернення: 13.04.2024).

14. IBM – Convolutional Neural Networks. [Електронний ресурс] – Режим доступу: https://www.ibm.com/cloud/learn/convolutional-neural-networks#tocwhat-are-c-MWGVhUiG9 (дата звернення: 13.04.2024).

15. V7 – YOLO: Real-Time Object Detection Explained. [Електронний ресурс] – Режим доступу: https://www.v7labs.com/blog/yolo-object-detection (дата звернення: 07.02.2024).

16. Roboflow. [Електронний ресурс] – Режим доступу: https://app.roboflow.com (дата звернення: 19.04.2024).

17. Mathleaks_filtered Computer Vision Project. [Електронний ресурс] – Режим доступу: https://universe.roboflow.com/mike-robot/mathleaks_filtered (дата звернення: 19.04.2024).

18. Athematical Expression Detection Computer Vision Project. [Електронний ресурс] – Режим доступу: https://universe.roboflow.com/equation-detectioncadby/mathematical-expression-detection (дата звернення: 07.02.2024).

19. Визначення вимог: що це таке і як це застосовувати? Електронний ресурс] – Режим доступу: https://visuresolutions.com/uk/blog/requirements-definition/ (дата звернення: 07.02.2024).

20. Тестування та верифікація програмного забезпечення. [Електронний ресурс] – Режим доступу: https://financial.lnu.edu.ua/wp-content/uploads/2019/09/konspekt-testuvannia.pdf (дата звернення: 07.04.2024).

21. Google Colaboratory. [Електронний ресурс]. – Режим доступу: https://colab.research.google.com/ (дата звернення: 05.04.2022 г.).

22. PyCharm. [Електронний ресурс]. – Режим доступу: https://www.jetbrains.com/ru-ru/pycharm/ (дата звернення: 07.04.2024).

23. Roboflow. [Електронний ресурс]. – Режим доступу: https://app.roboflow.com (дата звернення: 05.05.2022 г.).

24. Matplotlib. [Електронний ресурс]. – Режим доступу: https://matplotlib.org/ (дата звернення: 07.05.2024).

25. Numpy. [Електронний ресурс]. – Режим доступу: https://numpy.org/ (дата звернення: 07.02.2024).

26. Opencv. [Електронний ресурс]. – Режим доступу: https://pypi.org/project/opencv-python/ (дата звернення: 10.05.2024).

27. Pillow. [Електронний ресурс]. – Режим доступу: https://pypi.org/project/Pillow/ (дата звернення: 10.05.2024).

28. PyYAML. [Електронний ресурс]. – Режим доступу: https://pyyaml.org/wiki/PyYAMLDocumentation (дата звернення: 11.05.2024).

29. Torch. [Електронний ресурс]. – Режим доступу: https://pytorch.org/ (дата звернення: 11.05.2024).

30. Torchvision. [Електронний ресурс]. – Режим доступу: https://pytorch.org/ (дата звернення: 12.05.2024).

31. Tqdm. [Електронний ресурс]. – Режим доступу: https://tqdm.github.io/ (дата звернення: 12.05.2024).

32. Tensorboard. [Електронний ресурс]. – Режим доступу: https://www.tensorflow.org/tensorboard/get_started (дата звернення: 13.05.2024).

33. Pandas. [Електронний ресурс]. – Режим доступу: https://pandas.pydata.org/ (дата звернення: 13.05.2024).

34. Seaborn. [Електронний ресурс]. – Режим доступу: https://seaborn.pydata.org/ (дата звернення: 13.05.2024).

35. Roboflow – Dataset Health Check. [Електронний ресурс] – Режим доступу: https://app.roboflow.com/equals/equals-pzmaq/health (дата звернення: 14.05.2024).

36. Roboflow – Dataset Health Check. [Електронний ресурс] – Режим доступу: https://app.roboflow.com/equals/gg-s7vld/health (дата звернення: 14.05.2024).

37. YOLOv5 compared to Faster RCNN. Who wins? [Електронний ресурс]. – Режим доступу: https://towardsdatascience.com/yolov5-compared-to-faster-rcnnwho-wins-a771cd6c9fb4/ (дата звернення: 15.05.2024).

38. Удар струмом: перша допомога [Електронний ресурс] – Режим доступу: http://tomrda.gov.ua/news/5786468637438574 35/ (дата звертання: 20.05.2026).

39. Перша допомога при ураженні електричним струмом [Електронний ресурс]. – Режим доступу: https://bozhedarivskaselrada.gov.ua/ news/1576497483/ (дата звертання: 20.05.2024).

40. Ергономічні вимоги до організації робочих місць [Електронний ресурс] – Режим доступу: https://pidru4niki.com/14821111/bzhd /ergonomichni_vimogi_ organizatsiyi (дата звертання: 21.05.2024).

41. Охорона праці в офісі. Вимоги до робочого місця офісного працівника [Електронний ресурс] – Режим доступу: https://gc.ua/uk/oxorona-praci-v-ofisi-vimogi-do-robochogomiscya-ofisnogo-pracivnika/ (дата звертання: 21.05.2024).

ДОДАТКИ

Додаток А

Фрагмент коду реалізації основного модуля

```
import os
import shutil
import telebot
import detectFormula
import detectSymb

bot = telebot.TeleBot('TOKEN');

def translate(word):
   u = word.split()
   x = {'mult': '*', 'div': '/', 'minus': 'LBracket': '(',
'RBracket': 'other class':
values = x.keys()
for i in range(len(u)): if u[i] in values: w = x[u[i]]
else:
w = u[i] a += w + " "
return a

@bot.message handler(commands=['start']) def start
message(message):
bot.send message(message.chat.id, 'Привіт! Надішліть ваше фото!
\nДокументи не приймаються!')

@bot.message handler(content types=['text']) def send
welcome(message):
bot.send message(message.chat.id,'Привіт! Надішліть ваше фото!\
документи не приймаються!');

@bot.message handler(content types=['photo']) def handle docs
photo(message): try:

file info = bot.get file(message.photo[len(message.photo) - 1].
downloaded file = bot.download file(file info.file path) direl =
```

```
"photos"  -                          -              -
for f in os.listdir(direl):
os.remove(os.path.join(dire1, f))
src = файл info.file path; with open(src, 'wb') as new file:
new  file.write(downloaded  file)  bot.reply  to(message,  "Фото
додано")

bot.send message(message.chat.id, 'Починаю детекцування формули');
path  =  os.path.join(os.path.abspath(os.path.dirname(  file  )),
'detectFormula\\exp')

. . . . . .
```