Вінницький національний технічний університет
факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій

Магістерська кваліфікаційна робота на тему:

«РОЗРОБКА ЕКСПЕРТНОЇ СИСТЕМИ ГРАФОЛОГІЧНОГО АНАЛІЗУ З ВИКОРИСТАННЯМ СЕРВЕРНОЇ АРХІТЕКТУРИ»

Виконав: студент II курсу, групи 1АКІТ-22м
спеціальності 151 – Автоматизація та
комп'ютерно-інтегровані технології
(шифр і назва спеціальності)

Денис ТУЛЬЧІЙ
(ім'я та прізвище)

Керівник: д.т.н., проф., зав. каф. АІІТ
Олег БІСІКАЛО
(ім'я та прізвище)
«4» грудня 2023 р.

Опонент: д.т.н., проф. каф. КН
Ярослав ІВАНЧУК
(ім'я та прізвище)
«7» грудня 2023 р.

Допущено до захисту
Завідувач кафедри АІІТ

«11» грудня 2023 р.

Вінниця ВНТУ – 2023 рік

Вінницький національний технічний університет

Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій
Рівень вищої освіти другий (магістерський)
Галузь знань автоматизація та приладобудування
Спеціальність 151 – Автоматизація та комп'ютерно-інтегровані технології
Освітньо-професійна програма інтелектуальні комп'ютерні системи

затверджую
Завідувач кафедри АІІТ
д.т.н., проф. Олег БІСІКАЛО
«20» вересня 2023 р.

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ

Тульчію Денису Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: розробка експертної системи графологічного аналізу з використанням серверної архітектури
керівник роботи: Олег БІСІКАЛО, д.т.н., проф., зав. каф. АІІТ
затверджені наказом вищого навчального закладу від «18» вересня 2023р

2. Строк подання студентом роботи 12.12.2023

3. Вихідні дані до роботи:
   - мова програмування Python
   - кросплатформеність за рахунок серверної архітектури
   - графічне зображення рукописного тексту
   - точність класифікаторів більше 80%

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити). Провести огляд існуючих систем та засобів для визначення психологічного стану людини за її почерком, запропонувати методи вирішення цієї задачі, розробити експертну систему та серверну архітектуру, програмне забезпечення для графологічного аналізу

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень). Блок-схема експертної системи. структура бази знань експертної системи, алгоритм роботи серверу

6. Консультанти розділів роботи

Розділ	Ім'я ПРІЗВИЩЕ та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-3	Олег БІСІКАЛО, д.т.н., проф., зав. каф АІІТ		
4	Володимир КОЗЛОВСЬКИЙ, к.е.н., проф. каф. ЕПВМ		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської дипломної роботи	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз та обгрунтування створення програмного продукту.	01.09.2023	24.09.2023	виконав
2	Постановка задачі, та вибір оптимальної моделі	25.09.2023	30.09.2023	виконав
3	Проектування та розробка програмного забезпечення	01.10.2023	18.10.2023	виконав
4	Тестування та методика використання розробленого програмного забезпечення	19.10.2023	24.10.2023	виконав
5	Оформлення пояснювальної записки (ПЗ) та графічної частини	25.10.2023	29.11.2023	виконав
6	Перевірка ПЗ на відповідність вимогам	30.11.2023	7.12.2023	виконав
7	Попередній захист МКР, доопрацювання, рецензування МКР	08.12.2023	11.12.2023	виконав
8	Захист МКР	12.12.2023	13.12.2023	виконав

Студент _____ Денис ТУЛЬЧІЙ
(підпис) (прізвище, ініціали)

Керівник роботи _____ Олег БІСІКАЛО
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

УДК 004.891:159.925.6

Тульчій Д.С. Розробка експертної системи графологічного аналізу з використанням серверної архітектури. Магістерська кваліфікаційна робота зі спеціальності 151 – Автоматизація та комп’ютерно-інтегровані технології, освітня програма – Інтелектуальні комп’ютерні системи. Вінниця: ВНТУ, 2023. 102 с. На укр.мові. Бібліогр.: 32 назв; рис.: 45; табл.: 8.

Магістерська дипломна робота присвячена розробці експертної системи графологічного аналізу з використанням серверної архітектури. В роботі проводиться ґрунтовний огляд існуючих систем графологічного аналізу та методів графологічного аналізу для створення якісного програмного продукту. Проводиться аналіз варіантів створення серверної архітектури. Виконано тестування програмного забезпечення та його компонентів.

Ключові слова: експертна системи, графологія, база знань, графологічний аналіз серверна архітектура, магістр, кваліфікаційна робота, Python, BASH, PHP, веб-додаток, Ubuntu.

ANNOTATION

Master's qualification work with specialty 151 – Automation and computer-integrated technologies, educational program - Intelligent computer systems. Vinnytsia: VNTU, 2023. 102 c. In the Ukrainian language. Bibliography: citnum titles; Fig.: figures; tab.: 8.

The master's thesis is devoted to the development of an expert systemwe graphological analysis using server architecture. At work a thorough review of existing systems of graphological analysis is carried out and methods of graphological analysis to create a high-quality software product to whom An analysis of options for creating a server architecture is carried out. Execution but testing software and its components.

Keywords: expert systems, graphology, knowledge base, graphology dynamic analysis of server architecture, master's degree, qualification work, Python,BASH, PHP, Web App, Ubuntu.

ЗМІСТ

ВСТУП

Актуальність розробки експертної системи графологічного аналізу з використанням серверної архітектури полягає в нагальній потребі високоточної, об'єктивної та швидкої оцінки рукопису, яка має важливе значення в різних аспектах життя та діяльності:

а) «Психологія і психіатрія»: Експертний аналіз рукопису може допомогти виявити психологічні становища та розлади особистості, що є критично важливим у клінічній психології та психіатрії. Це може полегшити діагностику та надання ефективної допомоги.

б) «Рекрутинг і управління персоналом»: Роботодавці можуть використовувати графологічний аналіз для визначення психологічної сумісності кандидатів із вакансіями, покращуючи процес найму та зменшуючи ризики неправильного вибору персоналу.

в) «Кримінальна юстиція»: Експертиза рукопису грає важливу роль у розслідуванні злочинів, особливо у випадках фальсифікації документів чи підробки підписів.

г) «Психологічне та особистісне самопізнання»: Індивіди можуть використовувати графологічний аналіз для кращого розуміння власної особистості та стосунків із іншими.

д) «Навчальна система і виховання»: В освітній сфері, особливо в розвитку дітей, можливий аналіз рукопису може допомогти вчителям і батькам зрозуміти і підтримувати індивідуальні особливості дітей.

е) «Економія часу та ресурсів»: Автоматизований графологічний аналіз може розглядатися як засіб економії часу та людських ресурсів, особливо у великих організаціях.

Експертна система графологічного аналізу з серверною архітектурою є актуальною, оскільки вона вирішує проблеми, які мають суттєвий вплив на різні сфери життя та діяльності людей і організацій, прискорюючи та поліпшуючи процеси аналізу та прийняття рішень.

Мета і задача дослідження. Метою даної роботи є розробка систе-

ми, яка здатна автоматизовано аналізувати графічні ознаки рукопису для виявлення психологічних і особистісних особливостей людини та надавати корисну інформацію для різних сфер застосування.

Об'єктом дослідження є графічні ознаки почерку, серверні системи та їх особливості.

Предметом дослідження є методи та засоби розробки експертної системи графологічного аналізу з використанням серверної архітектури.

Методи дослідження, технічні та програмні засоби: аналіз та синтез, методи проєктування і розробки програмного забезпечення, використання мови програмування Python з метою побудови експертної системи з використанням бібліотеки SciKit-Learn, технології розробки програмного забезпечення для серверної архітектури, а також статистичний та психологічний аналіз даних.

Науково технічний результат роботи. На основі розробки ефективних моделей машинного навчання для графологічного аналізу та їх впровадження в серверну архітектуру отримано експертну систему, що дозволяє автоматизувати і об'єктивізувати аналіз сторінок рукописного тексту.

Практичною цінністю даної роботи є розроблене програмне забезпечення та система, яка має великий потенціал застосування в психології, психіатрії (для діагностики розладів), рекрутингу (для відбору персоналу), кримінальній юстиції (для експертизи документів). Вона може полегшити та поліпшити роботу фахівців у цих галузях, зекономити час та ресурси, і сприяти розвитку сучасних технологій.

Апробація результатів та публікації. Основні результати дослідження, викладені у магісткрській кваліфікаційній роботі доповідалися, обговорювалися та отримали позитивну оцінку на LI науково-технічній конференції працівників ВНТУ, Факультет інтелектуальних інформаційних технологій та автоматизації у 2022 році [1], а також на Міжнародній науково-практичній інтернет-конференції молодих науковців, аспірантів та студентів "Молодь в науці: дослідження, проблеми, перспективи (МН-2024)" [2].

1 АНАЛІЗ ТА ОСОБЛИВОСТІ СТВОРЕННЯ ЕКСПЕРТНИХ СИСТЕМ З ГРАФОЛОГІЇ

1.1 Загальні особливості експертних систем

Експертна система – це методологія для адаптації алгоритму рішень однієї сфери науково-практичної діяльності в іншу. З поширенням комп'ютерних технологій – це тотожна інтелектуальна комп'ютерна програма, що містить знання й аналітичні здібності одного чи кількох експертів в деякій галузі застосування і здатна робити логічні висновки на основі цих знань, тим самим забезпечуючи вирішення специфічних завдань без участі експерта. Визначається також як система, яка використовує базу знань для вирішення завдань у певній предметній галузі. Цей клас програмного забезпечення спочатку розроблявся дослідниками штучного інтелекту в 1960-ті та 1970-ті, а потім здобув комерційне застосування, починаючи з 1980-х. Часто термін система, заснована на знаннях, використовується як синонім експертної системи, однак можливості експертних систем ширші за можливості систем, заснованих на детермінованих (обмежених, реалізованих на поточний час) знаннях.

Для розробки експертних систем вимагається значний людський досвід та професіоналізм для вирішення проблем в області. Експертні системи здатні вирішувати лише обмежені проблемні питання. Проте навіть у дуже обмежених областях експертні системи потребують великих обсягів знань порівнюючи з фахівцями; виконують операції, які зазвичай вимагають наявності значного людського досвіду. Крім того, для цього потрібен експерт – людина, яка вміє знаходити вирішення проблем в конкретній предметній області. Для створення складної ЕС, потрібні знання і практичні навички декількох фахівців.

Експертна система відрізняється від інших прикладних програм наявністю таких ознак

– Моделює мислення людини під час вирішення задач в цій предме-

тній області. Це істотно відрізняє експертні системи від систем математичного моделювання. Однак, екпертні системи не повинні повністю повторювати психологічну модель фахівця в цій області, а повинні лише відтворювати за допомогою комп'ютера деякі методи розв'язання проблем, що використовує експерт.

- Експертна система, крім виконання операцій обчислення, робить висновки, опираючись на ті знання, якими вона володіє. Знання в системі, описані деякою спеціалізованою мовою і зберігаються окремо від програмного коду, що дає змогу формувати висновки. Компонент збереження знань прийнято називати базою знань.
- Під час розв'язання задач основну роль відіграють евристичні і наближені методи, що, на відміну від алгоритмічних, не завжди гарантують успіх. Евристика, в принципі, є правилом впливу, що в машинному вигляді відображає деяке знання, набуте людиною разом із накопичуванням практичного досвіду розв'язання аналогічних проблем. Такі методи є наближеними у тому сенсі, що, по-перше, вони не потребують вичерпної вихідної інформації, а, по-друге, існує певний ступінь впевненості в тому, що запропонований розв'язок є правильним.

База знань — це особливого роду база даних, розроблена для управління знаннями, тобто збором, зберіганням, пошуком і видачею знань. Розділ штучного інтелекту, що вивчає бази даних і методи роботи із знаннями, називається інженерією знань [1].

Прості бази знань використовуються для зберігання даних про організації. Головна мета створення таких баз — допомогти менш досвідченішим людям знайти існуючий опис способу для вирішення тої чи іншої проблеми предметної області.

База знань є важливим компонентом інтелектуальної системи. Найвідоміший клас таких програм — експертні системи. Вони призначені для знаходження способу вирішення специфічних проблем, базуючись на записах бази знань і на користувацькому описі ситуації. Створення і використання

систем штучного інтелекту потребує величезних баз знань [3].

Механізм логічного виведення складається з інтерпретатора, який визначає, як застосовувати правила для виведення нових знань. Машина логічного виведення (висновку) підтримує прямий логічний вивід та зворотній логічний вивід. Перший (індуктивний) – веде від даних до гіпотез, а другий (дедуктивний) – це спроба знайти дані для доведення або спрощення певної гіпотези. Найбільш досконалі системи використовують комбінацію обох типів виведення. Інтерпретатор команд визначає, як застосувати правила для виведення нових знань, встановлюючи порядок застосування цих правил.

1.2 **Графологія**

Графологія — це вчення, що стверджує про чіткий зв’язок між почерком та характером людини [4].

Графологічні методи дозволяють вирішити чимало задач, але, перш за все, слід зазначити чого не може графологія. Графологія не встановлює: стать виконавця рукопису, його вік (встановлюється лише психотип (чоловічий чи жіночий) та психологічний вік); професію (лише схильність до певного виду діяльності). Фахівець-графолог за рукописом може визначити психотип виконавця, психоемоційний стан в момент написання, вплив окремих збиваючих факторів на виконавця, як об’єктивних так і суб’єктивних, в момент написання рукопису та інші відомості, які можна обґрунтувати науково. В окремих джерелах, особливо ранніх, зустрічаються твердження про те, що графологи можуть за рукописом встановити статуру людини, окремі анатомічні ознаки. Вважаємо, що це лише припущення, які на даний час не підтверджені науково [5].

Перевагою графології перед іншими методами діагностики є те, що не потрібен безпосередній контакт з особою, яка перевіряється, достатньо мати лише зразки рукописного тексту, виконаного цією особою. Вважаємо, що безпосередній контакт може відіграти навіть негативну роль в процесі

аналізу [1]. Експертна система графологічного аналізу повинна брати до уваги:

- Розмір почерку
- Відстань між буквами
- Кут нахилу букв
- Натиск
- Міжрядкові інтервали

В результаті аналізу система повинна зробити висновки що до психологічного стану людину за її почерком.

1.3 **Основні бібліотеки для розробки експертної системи**

Scikit-learn — це безкоштовна програмна бібліотека машинного навчання для мови програмування Python, яка надає функціональність для створення та тренування різноманітних алгоритмів класифікації, регресії та кластеризації, таких як лінійна регресія, random forest, градієнтний бустинг, і працює у зв'язці з бібліотеками NumPy та SciPy. Scikit-learn є однією з найбільш популярних бібліотек машинного навчання [6].

Scikit-learn здебільшого написаний на Python та широко використовує NumPy для розв'язання задач лінійної алгебри та операцій з масивами. Крім того, деякі алгоритми написані на Cython для покращення продуктивності Scikit-learn добре інтегрується з багатьма бібліотеками Python, такими як Matplotlib та plotly для побудови графіків, NumPy для масивів, Pandas, SciPy та багатьма іншими [7].

Matplotlib — бібліотека на мові програмування Python для візуалізації даних двовимірною 2D графікою (3D графіка також підтримується). Отримувані зображення можуть бути використані як ілюстрації в публікаціях [8]. Matplotlib написана і підтримується в основному Джоном Хантером і поширюється на умовах BSD-подібної ліцензії. Зображення, які генеруються в різних форматах, можуть бути використані в інтерактивній графіці, наукових

публікаціях, графічному інтерфейсі користувача, вебдодатках, де потрібно будувати діаграми . В документації автор зізнається, що Matplotlib починався з імітування графічних команд MATLAB, але є незалежним від нього проєктом.

OpenCV (Open Source Computer Vision Library, бібліотека комп'ютерного зору з відкритим кодом) — бібліотека алгоритмів комп'ютерного зору, обробки зображень та чисельних алгоритмів загального призначення з відкритим кодом. Реалізована C/C++, також розробляється для Python, Java, Ruby, Matlab, Lua та інших мов. Може вільно використовуватися в академічних та комерційних цілях, поширюється за умов ліцензії BSD [9, 10].

1.4 **Основні поняття серверної архітектури**

Серверна архітектура є ключовим елементом сучасних інформаційних систем і мереж. Ця архітектура визначає спосіб організації серверів та їх взаємодію в мережі з метою забезпечення надійності, масштабованості, продуктивності і безпеки.

Однією з основних концепцій в серверній архітектурі є розділення обов'язків між серверами. Це означає, що різні сервери виконують різні функції, такі як веб-сервери, бази даних, кешування, обробка завдань тощо. Це дозволяє забезпечити більшу ефективність та масштабованість системи, а також полегшує розподілене управління ресурсами.

Для забезпечення надійності серверні архітектури використовують різні стратегії, такі як резервне копіювання, балансування навантаження, резервні джерела живлення і т. д. Важливо також мати відповідну систему моніторингу та управління для вчасного виявлення і вирішення проблем.

Однією з суттєвих архітектурних рішень є вибір між фізичними та віртуальними серверами. Віртуалізація дозволяє ефективніше використовувати ресурси і спрощує управління серверами, але вимагає високого рівня безпеки для захисту від атак на віртуальні машини.

Усі ці аспекти серверної архітектури мають велике значення у сучасному інформаційному середовищі, де надійність і продуктивність серверів є важливими чинниками для успішного функціонування бізнесу та інших організацій [11].

1.5 **Основні принципи серверної архітектури**

Розділення обов'язків між серверами в інформаційній системі є ключовим принципом серверної архітектури. Це дозволяє ефективно розподіляти завдання та ресурси для забезпечення надійності, масштабованості та ефективності системи.

Функціональність серверів може бути розділена на декілька типів:

- Веб-сервери: Вони відповідають за обробку HTTP-запитів і надання веб-сторінок користувачам через браузери. Веб-сервери обробляють запити на веб-сайти, веб-додатки та API. Популярні веб-сервери включають Apache, Nginx, Microsoft IIS.
- Бази даних (DB-сервери): Вони відповідають за зберігання, управління та доступ до даних. Бази даних використовуються для зберігання структурованих даних і дозволяють додаткам звертатися до цих даних. Популярні СУБД включають MySQL, PostgreSQL, Microsoft SQL Server, Oracle.
- Поштові сервери (Mail-сервери): Вони відповідають за обробку і доставку електронної пошти. Поштові сервери обробляють вхідні і вихідні листи, керують поштовими скриньками та фільтрують спам. Приклади: Sendmail, Postfix, Microsoft Exchange Server.
- Файлові сервери: Вони забезпечують доступ до файлів і папок через мережу. Файлові сервери дозволяють користувачам спільно працювати над документами та обмінюватися файлами. Приклади: Samba, Windows File Server.
- Подієві та проксі-сервери: Вони відповідають за маршрутизацію,

фільтрацію і обробку мережевого трафіку. Проксі-сервери можуть використовуватися для кешування веб-сторінок, захисту від DDoS-атак та фільтрації контенту.

- DNS-сервери: Вони відповідають за перетворення імен доменів на відповідні IP-адреси і навпаки. Вони дозволяють вам знаходити веб-сайти та інші ресурси за їхніми іменами. Приклади: BIND, Microsoft DNS Server.

Це лише кілька прикладів серверів та їхніх функцій. Залежно від потреб вашої організації та конкретного застосування, ви можете використовувати один або кілька типів серверів, щоб забезпечити необхідну функціональність і продуктивність вашої інформаційної системи [12].

1.6 **Серверні операційні системи та та їх специфікації**

Найбільш популярними операційними системи для серверів є дистрибутиви Linux:

- Ubuntu Server: Ubuntu Server є популярним вибором завдяки своїй простоті встановлення та управління, регулярним оновленням і довготерміновою підтримкою LTS-версій
- Red Hat Enterprise Linux (RHEL): RHEL є комерційною операційною системою, яка використовується в корпоративних середовищах і надає високий рівень підтримки та безпеки.
- CentOS: CentOS є безкоштовним клоном RHEL і також надає стабільну та надійну платформу для серверів.
- Debian: Debian є довготерміново підтримуваним дистрибутивом з акцентом на вільне програмне забезпечення і стабільність.
- Fedora Server: Fedora Server є швидко оновлюваним дистрибутивом з останніми версіями програмного забезпечення.

Важко визначити одну "найкращу"лінуксову систему, оскільки вибір операційної системи залежить від конкретних потреб, знань та вимог про-

екту. Проте, однією з найбільш популярних та широко використовуваних лінуксових систем для серверів є Ubuntu Server.

Ubuntu — операційна система для робочих станцій, лептопів і серверів, найпопулярніший у світі дистрибутив Linux. Серед основних цілей Ubuntu — надання сучасного й водночас стабільного програмного забезпечення для пересічного користувача із сильним акцентом на простоту встановлення та користування.

Ubuntu надає користувачу мінімальний набір програм загального призначення: багатовіконне стільничне середовище, засоби для перегляду Інтернету, організації електронної пошти, офісні програми з можливістю читати і записувати файли у форматах, що використовуються в пакеті програм Microsoft Office, редактор зображень, програвач компакт-дисків тощо. Спеціалізоване програмне забезпечення, потрібне досвідченішим користувачам, можна отримати з відповідних репозиторіїв. Серверний варіант системи включає також засоби, потрібні для організації сервера баз даних, вебсервера, сервера електронної пошти тощо [13].

Ubuntu Server має кілька переваг, що роблять його популярним вибором:

- Простота використання: Встановлення та налаштування Ubuntu Server досить прості завдяки дружньому графічному інтерфейсу.
- Велика спільнота користувачів і розробників: Ubuntu має велику та активну спільноту, яка надає підтримку, розвиває програмне забезпечення та допомагає вирішувати проблеми.
- Регулярні оновлення та довготермінова підтримка: Ubuntu Server має регулярний графік виходу нових версій, а LTS-версії надають підтримку протягом 5 років, що дозволяє стабільно працювати з системою.
- Безкоштовність і вільний код: Ubuntu Server є безкоштовним та вільним програмним забезпеченням, що означає, що ви можете використовувати його без витрат на ліцензії.

Найкращою версією Ubuntu для створення програмного продукту я

вважаю Ubuntu Server 22.04 LTS оскільки це сама сучасна версія Ubuntu Server та буде підтримуватись до квітня 2027 року та має найновіші функції серед інших версій [14].

1.7 **Основні мови та серверні додатки для написання програмного забезпечення на сервері.**

Для створення та налаштування серверу потрібно буде використовувати мову програмування Bash.

Bash, або Bourne Again SHell, є командним інтерпретатором для Unix-подібних операційних систем, таких як Linux і macOS. Це середовище командного рядка, яке дозволяє користувачам взаємодіяти з операційною системою шляхом введення текстових команд.

Основною метою Bash є виконання команд, введених користувачем, і запуск відповідних процесів. Він має свою власну синтаксичну структуру для написання скриптів, що дозволяє автоматизувати повторювані завдання, використовуючи послідовність команд.

Bash підтримує змінні, умовні вирази, цикли, функції та інші конструкції програмування, що робить його потужним інструментом для написання скриптів оболонки. Він також має багато вбудованих команд для роботи з файлами, процесами, текстовими рядками і т.д [15].

Одна з особливостей Bash - це можливість використання пайпів (|), які дозволяють направляти вивід однієї команди на вхід іншій, що спрощує складні завдання обробки даних.

Коротше кажучи, Bash є потужним інструментом для взаємодії з операційною системою через командний рядок і для написання скриптів для автоматизації різних завдань на рівні системи [16].

Для розробки програмного забезпечення потрібно використати Python. Python - це високорівнева, легко зрозуміла і популярна мова програмування. Вона має багато бібліотек для обробки зображень, текстового розпізнавання

і веб-розробки. Python також підходить для швидкого прототипування.

Python також використовується для створення експертної системи, що дає можливість без зайвих проблем об’єднувати виконання скриптів в одній мові без втрат швидкодії програмного продукту.

Для створення веб додатку потрібно налаштувати сервер для обробки HTTP/HTTPS запитів. Для цієї задачі можна використати такі сервіси як Nginx, або Apache.

Nginx (engine x) — це веб-сервер і проксі-сервер, який призначений для обробки HTTP-запитів і доставки веб-змісту користувачам. Він володіє високою продуктивністю, надійністю і широким спектром функцій, що робить його одним з найпопулярніших веб-серверів і проксі-серверів у світі [17].

Основні характеристики та переваги Nginx:

- Висока продуктивність: Nginx розроблений з огляду на високу продуктивність і вміє обробляти тисячі одночасних з’єднань і запитів. Він є популярним вибором для обробки великих навантажень і веб-сайтів з високим рівнем відвідуваності.
- Проксі-сервер і балансувальник навантаження: Nginx може виступати як проксі-сервер і балансувальник навантаження, розподіляючи трафік між декількома серверами для забезпечення надійності та масштабованості.
- Можливості кешування: Nginx підтримує кешування, що дозволяє зберігати статичні ресурси і віддавати їх швидше користувачам. Це зменшує навантаження на сервер і покращує час завантаження сторінок.
- Підтримка SSL/TLS: Nginx має потужні можливості щодо обробки SSL/TLS-з’єднань, що робить його ідеальним вибором для захисту веб-сайтів з використанням шифрування.
- Легко налаштовується: Конфігурація Nginx здійснюється через файл конфігурації з простим синтаксисом. Це робить його дружнім до адміністраторів серверів.
- Споживання ресурсів: Nginx відомий своєю низькою витратою ре-

сурсів, що дозволяє ефективно використовувати обмежені апаратні ресурси.

Apache — є одним з найпопулярніших веб-серверів у світі. Він є вільним і відкритим програмним забезпеченням і зазвичай використовується для обробки HTTP-запитів і доставки веб-змісту користувачам.

Основні характеристики та переваги Apache:

- Надійність та стабільність: Apache відомий своєю надійністю і стабільністю. Він давно існує на ринку і зазнав безлічі тестів та покращень, що робить його надійним вибором для великих та критичних серверів.
- Модульна архітектура: Apache має модульну архітектуру, яка дозволяє додавати і відключати різні функції і можливості через модулі. Це дозволяє налаштовувати сервер для конкретних потреб.
- Підтримка SSL/TLS: Apache підтримує шифрування SSL/TLS для захисту з'єднань і забезпечення конфіденційності даних користувачів.
- Багатофункціональність: Він має багато корисних функцій, таких як перенаправлення, обробка заголовків, кешування, контроль доступу і багато іншого.
- Багатофронтендність: Apache може бути використаний разом з різними мовами програмування, такими як PHP, Python, Perl і багатьма іншими, для створення динамічних веб-сторінок.
- Споживання ресурсів: Apache відомий своєю помірною витратою ресурсів, що дозволяє його ефективно використовувати на серверах з обмеженими апаратними ресурсами [18].

Інтерфейс FastCGI — клієнт-серверний протокол взаємодії вебсервера та програми, подальший розвиток технології CGI. У порівнянні з CGI є продуктивнішим і безпечнішим.

Також потрібно встановити на сервер Python та потрібні бібліотеки для виконання програми беспосередньо на сервері.

1.8 **Аналіз аналогів програмного продукту**

Перед початком створення додатку, важливо визначити основну конкуренцію у предметній сфері та на основі відповідного аналізу, обгрунтувати новизну своєї ідеї.

Проаналізувавши відомі програмні продукти було виявлено такі аналоги: Одним з найбільших конкурентів є додаток під андроїд з назвою "Сигнатурна графологія" (рис. 1.1). Цей додаток визначає психологічний стан людини за її підписом (рис. 1.2).

Даний додаток включає в себе наступні функції

- Аналіз за допомогою штучного інтелекту на графічному зображенні.
- Ручний аналіз, за допомогою якого людинна заповнює анкету з основними характеристиками підпису. На основі цього додаток робить висновок щодо психологічного стану людини.

Додаток має наступні переваги

- Простий та зрозумілий інтерфейс.
- Всі обчислення виконуються на стороні клієнта, що дає змогу працювати без доступу до інтернету.
- Швидка обробка зображення та виведення результату (рис. 1.3).
- Додаток працює швидко (протестовано на Xiaomi Redmi Note 10 Pro).

Додаток має наступні недоліки

- Не працює з текстом.
- Має малу кількість ознак.
- Досить обмежений опис психологічного стану людини.
- Не дуже точний графологічний аналіз підпису.
- Не дуже приємний інтерфейс програми. Занадто великі букви, а деякий текст перекриває інший.

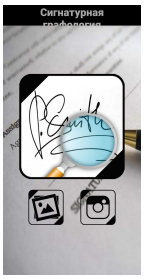


Рисунок 1.1 – Приклад роботи додатку "Сигнатурна графологія". Головне меню.

Рисунок 1.2 – Приклад роботи додатку "Сигнатурна графологія". Внесення графічного документу.

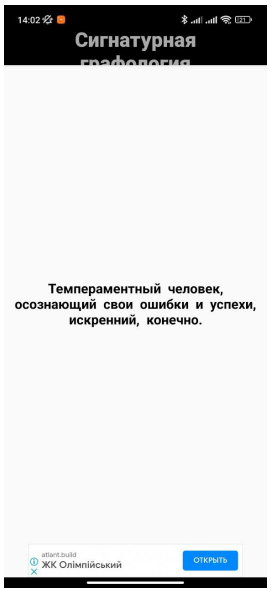


Рисунок 1.3 – Приклад роботи додатку "Сигнатурна графологія". Результат.

Наступний аналог - це також додаток під андроїд з назвою "Графологія" (рис. 1.4). Додаток дає змогу людині проаналізувати її психологічний стан та особливості по її почерку.

Даний додаток включає в себе наступні функції:

– Ручний аналіз за допомогою якого людинна може визначити основні характеристики почерку і одразу дізнатись, що означають ті чи інші характеристики (рис. 1.5).

Додаток має наступні переваги:

– Простий та зрозумілий інтерфейс.
– Велика кількість ознак почерку.
– Додаток працює швидко (протестовано на Xiaomi Redmi Note 10 Pro).

Додаток має наступні недоліки:

– Аналіз потрібно проводити вручну.
– Повний аналіз почерку займає велику кількість часу.
– Не працює з графічними документами.

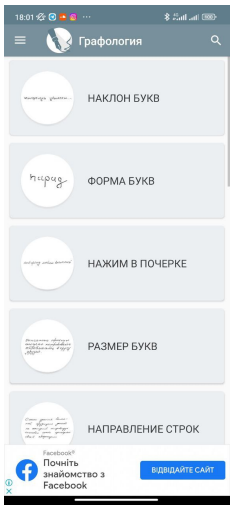


Рисунок 1.4 – Приклад роботи додатку "Графологія". Головне меню.

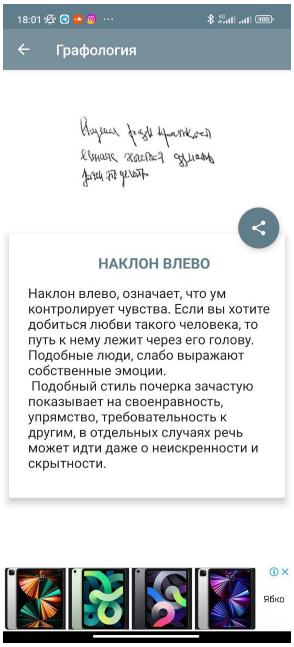


Рисунок 1.5 – Приклад роботи додатку "Графологія". Результат.

1.9 **Постановка задачі для побудови експертної системи**

Отже, актуальною задачею є розробка експертної системи графологічного аналізу, що надає можливість користувачу визначити психологічний стан певної особи за його/її почерком. Додаток буде розроблений на Python

та повинен забезпечувати користувачу висновки щодо, наприклад, власного психологічного стану. На відміну від існуючих аналогів, програмний засіб має бути більш точним та більш функціональним, що дасть змогу налаштувати додаток під необхідні умови використання.

На основі вхідного графічного файлу з рукописним текстом експертна система має визначити такі характеристики почерку автора цього тексту [19]:

– Параметри букв - розмір букв та кут їх нахилу.
– Інтервали - міжрядковий та між словами.
– Базова лінія.
– Верхня маржа.
– Натиск ручки.

У свою чергу, отримана множина характеристик почерку має дозволити дати оцінку восьми рис особистості автора. Зокрема, це:

– Емоційна стабільність.
– Психічна енергія (сила волі).
– Скромність.
– Особиста гармонія та гнучкість.
– Недисциплінованість.
– Погана потужність концентрації.
– Некомунікативність.
– Соціальна ізоляція.

Для розв’язання поставленої задачі пропонується така схема експертної системи графологічного аналізу (рис. 1.6).

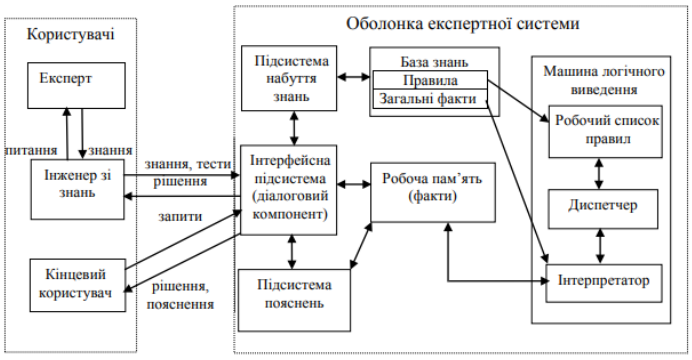


Рисунок 1.6 – Схема експертної системи

Для створення потрібний експерт в даній області та інженер який розробить дану систему. Експертна система буде мати у собі базу знань з якої буде брати інформацію для точного аналізу поставленого завдання.

Після запиту до бази знань експертна система повина виконати основні задачі в машині логічного введення. Через робочу пам'ять експертна система передасть результати до інтерфейсу даної системи і нарешті потрапить до кінцевого користувача.

1.10 **Висновки по розділу 1**

У даному розділі проведено аналіз сучасного стану, розкито поняття графології та експертної системи.

Загальний висновок полягає в тому, що веб-сервери, сервери Ubuntu, Nginx, Apache та програмування на Python є важливими компонентами для розгортання та адміністрування веб-проектів. Розуміння їх функціональності і налаштування є ключовими для успішної роботи з веб-додатками та веб-сайтами.

Мова Bash є дуже корисною для налаштування додатків на сервері та програмування логіки роботи серверу для окремих задач.

Мова Python є одною з найкращих мов для написання експертних систем та навчання їх. Також має велику кількість бібліотек для обробки зображення, що дає можливість створити доволі непоганий програмний продукт з використанням різних бібліотек для покращення бистродії програмного продукту

Інструменти для розробки експертних систем знаходятся в стадії активного розвитку. Розробка експертних систем вражає своєю різноманітністю та потенціалом застосування в різних сферах життя та науки. Розвиток технологій, зокрема штучного інтелекту та машинного навчання, створює унікальні можливості для створення і вдосконалення експертних систем.

Сучасні експертні системи відкривають безмежні можливості для ви-

користання штучного інтелекту в різних галузях і сферах діяльності. Такі системи можуть полегшити прийняття рішень, зробити їх більш об’єктивними та швидкими, і відіграти ключову роль у розвитку сучасних технологій і покращенні якості життя.

2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ЕКСПЕРТНОЇ СИСТЕМИ

2.1 Збір даних

Отримано дані з бази даних рукописного введення IAM дослідницької групи з комп'ютерного зору та штучного інтелекту INF, Бернський університет, Швейцарія (рис. 2.1). Дані легко доступні для завантаження для використання в некомерційних дослідницьких цілях. База даних містить 1538 сторінок відсканованого тексту.

Ці зображення обрізаються та зберігаються як зображення PNG зі сценарієм автоматичної дії. Тепер ширина всього зображення становить 850 пікселів, а висота відповідно до змісту почерку на зображенні (рис. 2.2).

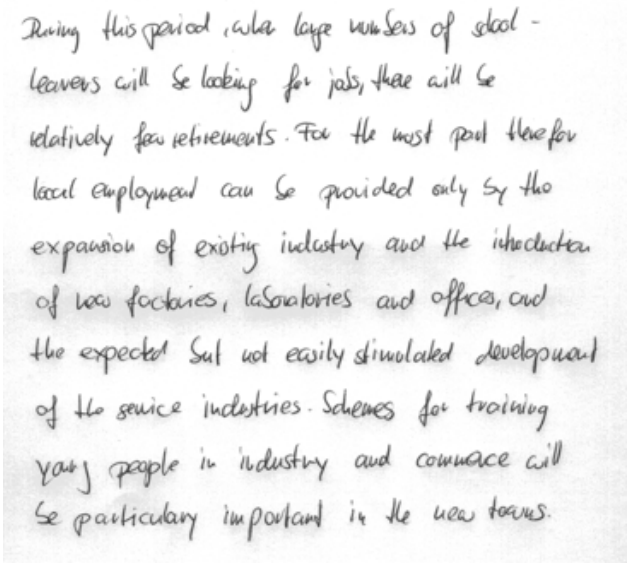
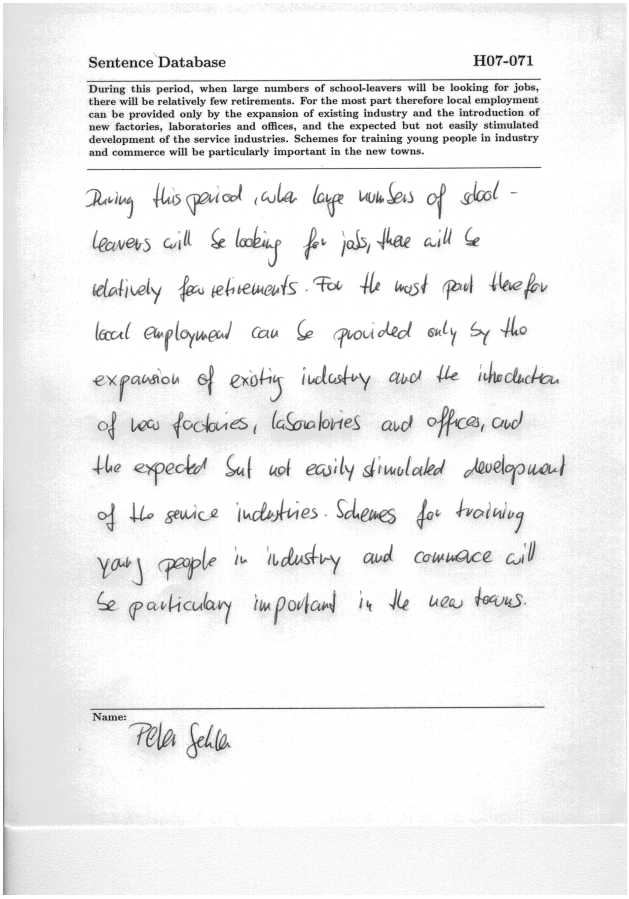


Рисунок 2.1 – Зразок даних обрізаного та нормалізованого зображення з шириною 850 пікселів.



Sentence Database H07-071

During this period, when large numbers of school-leavers will be looking for jobs, there will be relatively few retirements. For the most part therefore local employment can be provided only by the expansion of existing industry and the introduction of new factories, laboratories and offices, and the expected but not easily stimulated development of the service industries. Schemes for training young people in industry and commerce will be particularly important in the new towns.

During this period, when large numbers of school-leavers will be looking for jobs, there will be relatively few retirements. For the most part therefor local employment can be provided only by the expansion of existig industry and the introduction of new factories, laboratories and offices, and the expected but not easily stimulated development of the service industries. Schemes for training yang people in industry and commerce will be particulary important in the new towns.

Name: Peter Schlo

Рисунок 2.2 – Зразок даних оригінального зображення, отриманий з бази даних рукописного введення IAM

2.2 **Попередня обробка графічних документів**

Отримані нами зображення рукописного тексту містять небажані шуми, друковані тексти та рядки. Оригінальні зображення також мають дуже велику роздільну здатність за замовчуванням. Мета попередньої обробки це зробити дані зображення придатними для вилучення ознак шляхом фільтрації небажаних атрибутів, підвищення якості та виконання перетворень.

2.2.1 Роздільна здатність та обрізка зображень

Для обробки зображень використовуємо Adobe Photoshop, для автоматичного обрізання лівого та правого полів, зміни розміру всіх зображень із шириною та висотою 850 пікселів, а також збереження зображень у форматі PNG. Оригінальні зображення містять небажані друковані тексти, рядки та вільний простір, які не придатні для подальшої обробки.

Формат PNG використовується замість JPEG, оскільки перший формат без втрат у якості та більше підходить для зберігання текстових зображень, друкованих або рукописних.

2.2.2 Усунення шумів на зображеннях

Шум зображення визначається як випадкова зміна яскравості або інформації про колір зображень і зазвичай є аспектом електронного шуму. Він може бути створений датчиком і схемою сканера або цифрової камери.

Шум зображення є небажаним побічним продуктом захоплення зображення, який приховує потрібну інформацію. Існує кілька основних типів шуму.

- Гауссовий шум, він також відомий як електронний шум, оскільки він виникає в підсилювачах або детекторах. Гауссовий шум викликається природними джерелами, такими як теплові коливання атомів і дискретна природа випромінювання теплих об’єктів [20].
- Шум солі та перцю - поширений або «імпульсивний» шум також називається шумом солі та перцю або шумом спайків. Зображення, що містить шум солі та перцю, матиме темні пікселі в світлих областях і яскраві пікселі в темних областях. Цей тип шуму може бути викликаний помилками аналого-цифрового перетворювача, бітовими помилками передачі тощо [21].

– Гамма-шум - зазвичай спостерігається на зображеннях на основі лазера. Він підпорядковується гамма-розподілу.

Шум можна видалити, відфільтрувавши зображення. Існує кілька методів фільтрації для видалення шуму всередині зображень. Деякі з них є середнім фільтром, середнім фільтром та двостороннім фільтром тощо. З негативного боку, застосування цих функцій також може знизити рівень деталізації зображення [22].

Деякі небажані шуми присутні на оригінальних зображеннях. Ці шуми необхідно видалити із зображень для ефективного вилучення ознак. Для видалення цих шумів використовується двосторонній фільтр, оскільки він зберігає краї елементів зображення, що дуже бажано. Двосторонній фільтр — це нелінійний фільтр згладжування зображень, що зберігає краї та зменшує шум. Він замінює інтенсивність кожного пікселя на середнє зважене значення інтенсивності з сусідніх пікселів. Ця вага може бути заснована на розподілі Гаусса [20–22].

2.2.3 Афінне перетворення контуру та деформації

Після видалення шумів і перетворення зображення у відтінки сірого та зворотної бінаризації, лінії почерку вирівнюються за допомогою розширення, контуру та афінного перетворення за допомогою бібліотеки OpenCV [10]. Це дасть кращий результат при подальших операціях із використанням горизонтальної проекції зображення для вилучення цих рядків рукопису (рис. 2.3).

Контур – це замкнута крива з точок або відрізків, що представляють межі об’єкта на зображенні. Іншими словами, контури представляють формио б’єктів, знайдених на зображенні. Якщо на зображенні видно внутрішні деталі,об’єкт може створити кілька пов’язаних контурів, які повертаються в ієрархічній структурі даних. Як тільки ми знайдемо контури об’єктів на зображенні, ми можемо виконувати такі дії, як визначати кількість об’єктів на

зображенні, класифікувати форми об’єктів або вимірювати розмір об’єктів. Таким чином, правильне вилучення контуру дасть більш точні ознаки, що підвищить шанси правильно класифікувати даний візерунок (рис. 2.4).

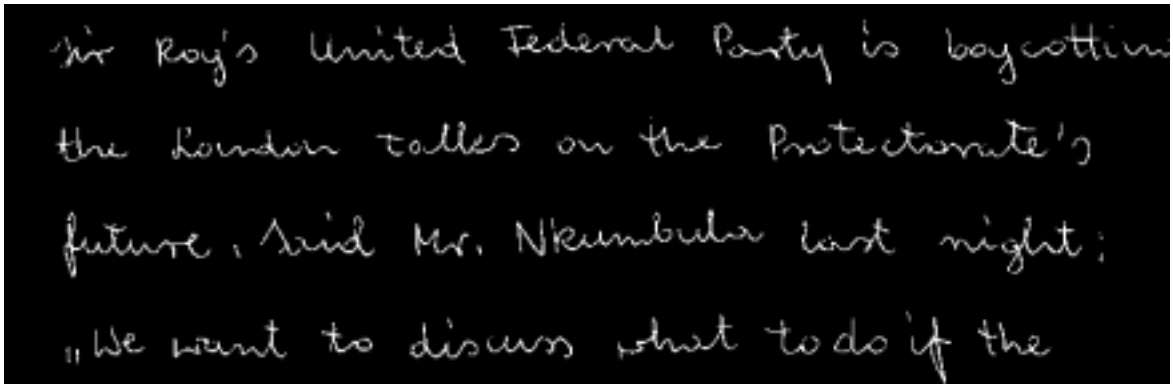


Рисунок 2.3 – Зразок зображення, що має чорні фонові пікселі та білі передні пікселі.

Рисунок 2.4 – Зразок зображення після застосування розширення з ядром 5x100.Передні пікселі розподілені по горизонталі.

Афінне перетворення — це важливий клас лінійних двовимірних геометричних перетворень, які відображають змінні (наприклад, значення інтенсивності пікселя, розташовані в позиції x на вхідному зображенні) у нові змінні (наприклад, x у вихідному зображенні) шляхом застосування лінійної комбінації операцій трансляції, обертання, масштабування та/або зсуву (тобто нерівномірного масштабування в деяких напрямках) (рис. 2.5).

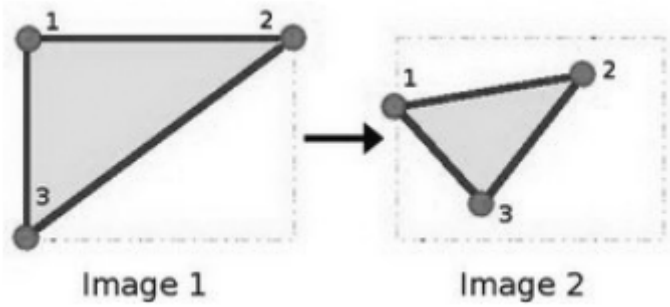


Рисунок 2.5 – Приклад афінного перетворення

Афінне перетворення деформації застосовується для повороту контурів, знайдених на зображенні, щоб базова лінія почерку була строго горизонтальною. Кут обертального перетворення, повернутий операцією, також використовується як одна із семи ознак пізніше [23].

2.2.4 Горизонтальні та вертикальні проекції

У контексті цього проекту горизонтальна проекція зображення є списком Python, суми всіх значень пікселів кожного рядка зображення, тоді як вертикальна проекція є списком Python суми всіх значень пікселів кожного стовпця зображення. Обидві ці операції виконуються на зображеннях у відтінках сірого.

2.3 **Машинне навчання**

Машинне навчання — це галузь інформаційних тезнологій, яка використовує статистичні методи, щоб надати комп’ютерним системам можливість «навчатися» (наприклад, поступово покращувати ефективність виконання певного завдання) з даними, не будучи явно запрограмованими.

У машинному навчанні машини опорних векторів (SVM) — це контрольовані моделі навчання з пов’язаними алгоритмами навчання, які ана-

лізують дані, що використовуються для класифікації та регресійного аналізу [24]. З огляду на набір навчальних прикладів, кожен із яких позначено як належний до однієї чи іншої з двох категорій, навчальний алгоритм SVM будує модель, яка призначає нові приклади до тієї чи іншої категорії, роблячи її неімовірнісним бінарним лінійним класифікатором (хоча методи такі як масштабування Платта існують для використання SVM в умовах імовірнісної класифікації). Модель SVM — це представлення прикладів у вигляді точок у просторі, нанесених на карту таким чином, щоб приклади окремих категорій були розділені чітким розривом, який був якомога ширшим (рис. 2.6). Нові приклади потім відображаються в цьому самому просторі та передбачаються належати до категорії, виходячи з того, з якого боку розриву вони потрапляють. На додаток до виконання лінійної класифікації, SVM можуть ефективно виконувати нелінійну класифікацію, використовуючи так званий трюк ядра, неявно відображаючи свої вхідні дані у просторі високовимірних ознак [25].

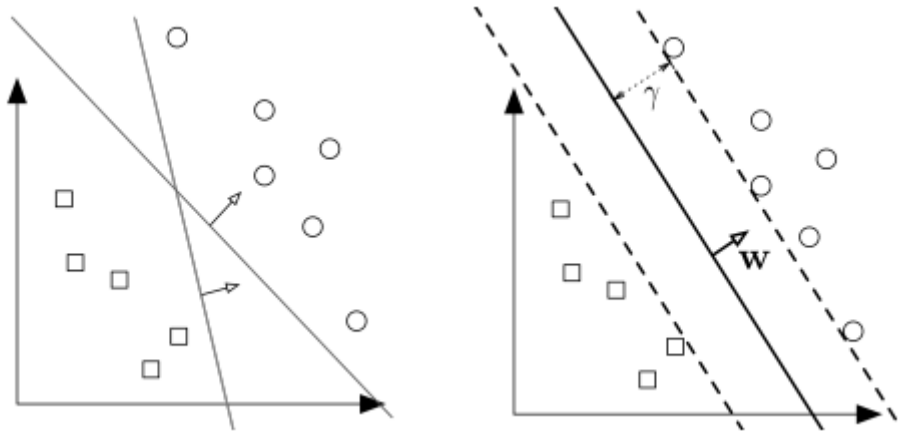


Рисунок 2.6 – Приклад роботи моделі SVM

2.4 **Особливості рукописного введення**

Існують численні характеристики почерку, прості та складні, за допомогою яких графолог може передбачити психологічний стан письменника.

З них для виділення вибираються сім ознак, які вважаються значущими. Тут вони коротко обговорюються.

2.4.1 Базова лінія

Базова лінія почерку утворює невидиму лінію між середньою і верхньою зонами вгорі і нижньою зоною внизу. Через свої характеристики рівності або нестійкості він показує, наскільки добре особистість справляється з сумішшю впливів інтелектуальних, соціальних та інстинктивних потягів. Базову лінію можна візуалізувати як лінійний графік між его і совістю вгорі та соматичними напруженнями внизу; якщо воно тримається стійко, але розслаблено, письмо ближче до здорового цілого, але якщо воно тягнеться вгору через думки та турботи про его і вниз через інстинктивні потреби, або якщо воно жорстке, як багнет, особистість у біді. Як індикатор настрою, морального та соціального контролю, темпераменту, настрою та гнучкості базовою лінією є його налаштування [19].

- Зазвичай прямі лінії (рис. 2.7): зібраність, впорядкованість, емоційна стабільність, надійність, наполегливість. Розум письменника дисциплінує його емоції.
- Зростання лінії (рис. 2.8): бадьорий дух, амбіції, оптимізм, непосидючість. Письменник хоче втекти від вимог рутини. Він збудливий і швидко спонукається до дії. Іноді він втрачає себе від зовнішніх впливів.
- Падіння лінії (рис. 2.9): втома, депресія, розчарування, нещастя, розчарування. Крім того, впертість і рішучість.

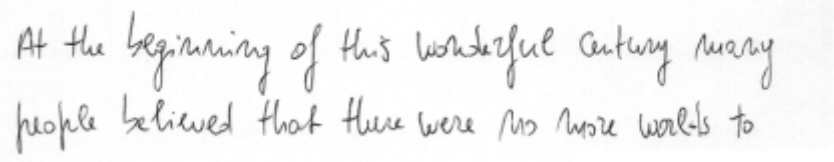


Рисунок 2.7 – Зразок почерку з прямим базовим кутом.

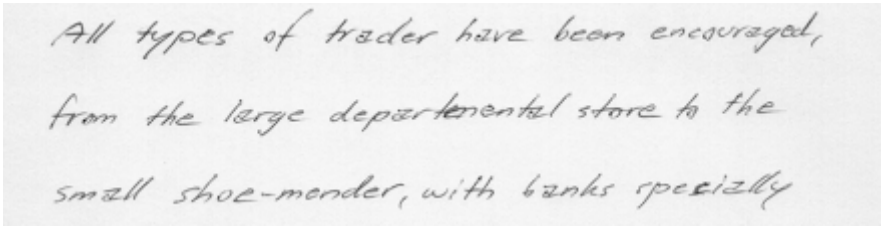


Рисунок 2.8 – Зразок почерку зі зростанням базового кута.

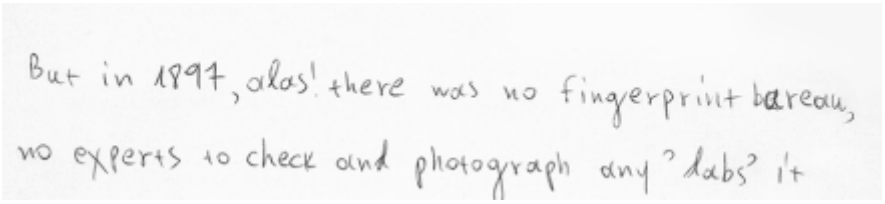


Рисунок 2.9 – Зразок почерку зі спадним кутом базової лінії.

2.4.2 Розмір букви

Щоб визначити розмір зразка почерку, в основному розглядають літери середньої зони. Вони мають бути 1/8 дюйма або 3 міліметри заввишки, щоб підпадати під категорію звичайної книги. Написання, в якому літери середньої зони постійно піднімаються вище 1/8 дюйма, вважається більшим, ніж зазвичай, а все, що менше 1/8 дюйма, вважається меншим за норму [26].

- Звичайний або середній розмір літер (рис. 2.10): люди, які пишуть шрифтом звичайного розміру, можуть вписуватися в звичайні або переважні обставини з адаптивністю та врівноваженістю. Вони практичні та реалістичні.
- Більший за середній розмір (рис. 2.11): він показує, що людині потрібно справити враження, бути поміченими, щоб отримати визнання. Ці люди потребують уваги та захоплення та користуються ними; вони не люблять бути на самоті. Вони можуть діяти з сміливістю, ентузіазмом і оптимізмом, але також здатні до хвалькості, непосидючості, недостатньої концентрації та дисципліни.

– Розмір менший за середній (рис. 2.12): позначає інтроспективну людину, яка не схильна шукати в центрі уваги і яка не дуже комунікабельна, за винятком близьких друзів. Маленькі лытери часто мають академічний менталітет і можуть тривалий час зосередитися на своїх дослідженнях і проектах. Хоча вони скромні, іноді аж до відчуття неповноцінності, талант цих письменників до деталей і організації часто дає їм хороші виконавські здібності.

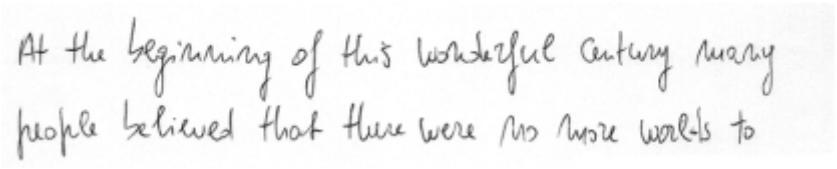


Рисунок 2.10 – Зразок почерку із середнім розміром літер.

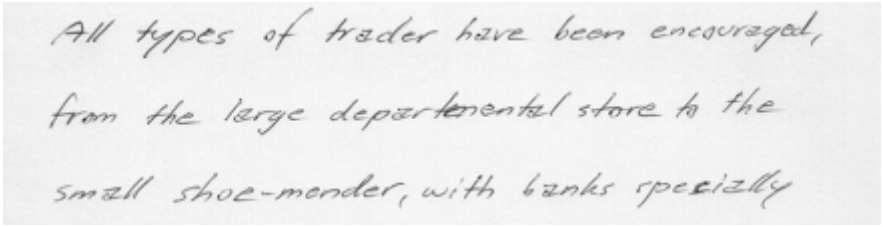


Рисунок 2.11 – Зразок почерку з великим розміром літер.

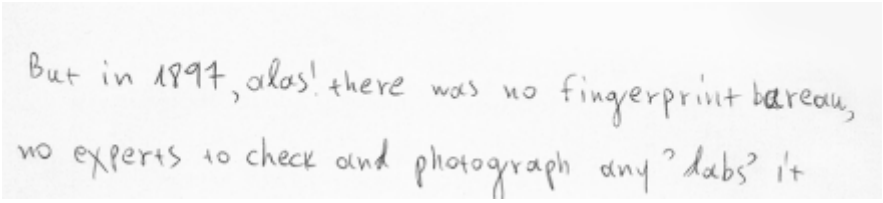


Рисунок 2.12 – Зразок почерку з малим розміром літер.

2.4.3 Міжрядковий інтервал

Обсяг простору, який письменник залишає між рядками на сторінці, дає підказки про впорядкованість і ясність його мислення, а також про обсяг

взаємодії, який він хоче мати зі своїм оточенням [27].

2.4.4 Інтервал між словами

Простір, що залишився між написаними словами, представляє дистанцію, яку письменник хотів би підтримувати між собою та суспільством в цілому. Як і з окремою літерою, письменник представляє себе, коли розміщує кожну словесну одиницю на сторінці; між словами лежить дистанція, необхідна йому для емоційного комфорту з іншими, його територіальні межі [27].

2.4.5 Тиск пера

Сильний тиск: такі письменники справляють враження. Їм доступна велика кількість енергії для своїх дій. Вони виражені важко, вольові, тверді і легко збуджуються до гарячих дій. Ті, хто має сильний тиск, можуть надихнути інших. Негативно, вони можуть бути суворими, впертими і схильними до похмурих думок або депресії. Їх наявність точно відомо (рис. 2.13).

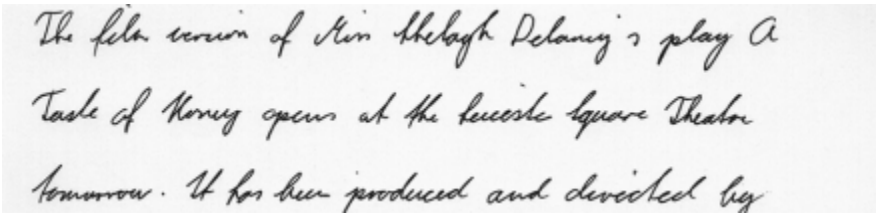


Рисунок 2.13 – Зразок почерку з сильним тиском пера.

Легкий тиск: ці люди володіють певною тонкістю почуттів. Особистість чутлива і вразлива (рис. 2.14). Часто є великі творчі здібності, але потенціал рідко реалізується, оскільки ці люди, не в змозі поглинути свій досвід. Вони набагато толерантніші та доброзичливіші, ніж інші. Коли ре-

шта письма гармонійна, досягаються найтонші вершини духовності та ідеалізму. При негармонійному письмі легкий натиск є показником ламкості та слабкості нервів.

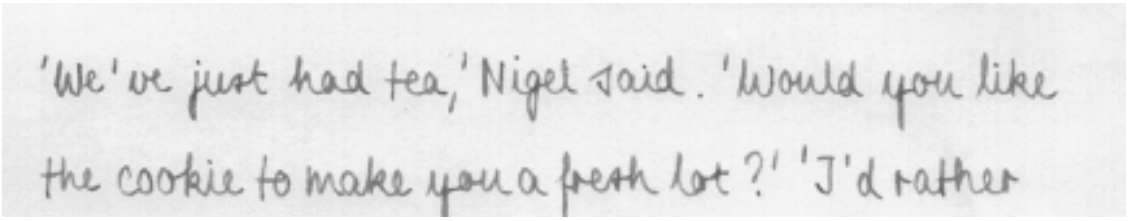


Рисунок 2.14 – Зразок почерку з легким тиском пера.

Середній тиск (рис. 2.15): це норма між крайнощами і є показником здорової життєвої сили та сили волі.

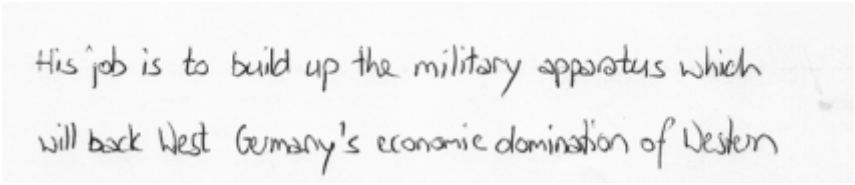


Рисунок 2.15 – Зразок почерку з середнім тиском пера.

2.4.6 Нахил букв

Вертикаль: Цей тип має емоційне ставлення до серця (рис. 2.16). Він відкритий для переживань моменту, але його відповіді обережні й обдумані. Тут емоційне вираження контролюється. Манера недемонстративна, незалежна, відсторонена і навіть байдужа. Після втрати емоційного контролю він швидко відновлюється, тому цей тип добре функціонує в надзвичайних ситуаціях і стає хорошим лідером або задоволеним самотником. Людина зацікавлена в собі і запитує: «Що може зробити для мене ця ситуація?». Аргументуючи якусь думку, цей письменник буде звертатися до судження, а не до емоцій. Часто він має великий особистий магнетизм та сухий розум, що є досить привабливим.

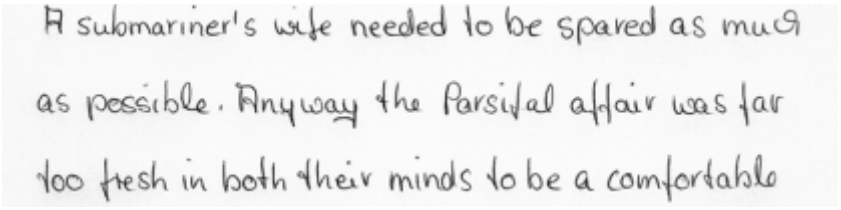


Рисунок 2.16 – Зразок почерку з вертикальним нахилом

Трохи нахилений: цей нахил вважається нормальним (рис. 2.17). Письменник зазвичай чутливий і емоційно здоровий, але скромний із відповідями. Тут панують судження і логіка, але більше співчуття .

Дуже нахилений: ці люди охоче плачуть і сміються, дають волю своїм почуттям, орієнтовані на майбутнє та цілеспрямовані, мають палкий, ласкавий, привітний і чутливий емоційний характер. Вони імпульсивно висловлюють свою емоційність. Почуття впливатимуть на рішення, і вони швидко реагують із піднесенням або знеохоченням, вони ідентифікують себе з оточенням та з точкою зору інших людей і реагують зі співчуттям.

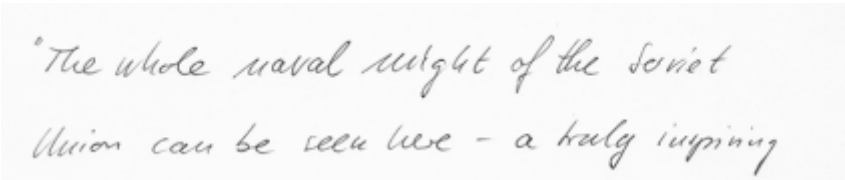


Рисунок 2.17 – Зразок нахиленого почерку

Нестабільний нахил: цей тип непостійний і непослідовний (рис. 2.18). Він підвладний настроям і думкам моменту. Емоційний характер непостійний; ніколи не знаєш, як він відреагує. Він коливається між придушенням і виразом з повною відсутністю панівного ставлення. Характер нервовий, недисциплінований, примхливий, збудливий, непостійний і позбавлений розсудливості чи здорового глузду. Усередині людина відчуває себе соціально неповноцінною і нецентричною

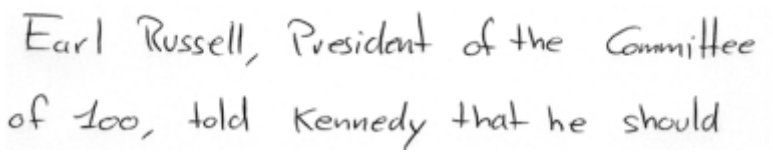


Рисунок 2.18 – Зразок почерку з нестабільним нахилом

2.5 **Риси особистості**

Комбінації семи виділених ознак почерку використовуються для прогнозування восьми рис особистості письменника, як наведено нижче [26].

- Емоційна стабільність: визначається базовою лінією та кутом нахилу.
- Психічна енергія або сила волі: визначається розміром літер і натиском пера.
- Скромність: визначається верхнім полем і розміром літер.
- Особиста гармонія та гнучкість: дається міжрядковим і міжслововим інтервалом.
- Недисциплінованість: визначається верхнім полем і кутом нахилу.
- Погана потужність концентрації: визначається розміром літер та міжрядковим інтервалом.
- Некомунікативність: Визначається розміром літер і міжслівним інтервалом.
- Соціальна ізоляція: визначається міжрядковим і інтервалом між словами.

2.6 **Визначення базової лінії**

Для того, щоб вилучити базовий кут лінії почерку виконуються такі операції:

а) Перетворити зображення в двійкове з порогом 120. Тепер фонові пікселі чорні, а пікселі переднього плану (рукописні) білі

б) На зображенні виконується розширення за допомогою ядра 5x100, для того , щоб кожна лінія була приведена у вигляді горизонтального сегмента

в) Контури (з попереднього кроку), які мають висоту менше 20 пікселів, відкидаються, оскільки вони не можуть бути лінією рукописного введення. Тепер контури, що залишилися, представляють кожну лінію або групу скупчених рядків почерку

г) Функція прямокутника мінімальної площі за допомогою OpenCV приймає об'єкт контуру і повертає кут, який контур складає з гіпотетичною вертикальною лінією, як одне з повернутих значень. Потім визначається кут, який складають ці контури з гіпотетичною горизонтальною лінією

д) Середнє значення кутів усіх контурів береться за базовий кут, наша перша ознака

е) Крім того, навколо контуру формується матриця обертання. Матриця обертання використовується для повороту контуру на його базовий кут у протилежному напрямку, щоб він був ідеально горизонтальним. Ця операція допомагає максимізувати ефективність операцій з горизонтальною проекцією

Цей алгоритм реалізовано у функції straighten() extract.py.

2.7 **Визначення окремих ліній**

Вилучення окремих ліній виконує функція extractLines() сценарію extract.py витягує окремі рядки рукописного введення, як описано в наступному алгоритмі.

а) Береться випрямлене зображення, отримане в результаті вилучення базової лінії, і виявляється його горизонтальна проекція в список

python (рис. 2.19).

б) Список сканується зверху вниз. Деякі значення горизонтальної проекції можуть бути 0, які представляють рядки пробілів. Рядок із ненульовим значенням становить рядок пікселів, що зустрічає принаймні один піксель (контур) переднього плану. Кожен контур ідентифікується від початку ненульового значення до наступного нульового значення (рис. 2.20).

в) Контур знову сканується для виділення окремих рядків, якщо це група переповнених ліній. Контур із переповненою групою ліній матиме дуже низьке значення у зоні перекриття лінії зверху та під лінією. Для ідентифікації таких рядків встановлюється поріг. Під час сканування зверху список буде поступово збільшуватися, а потім знову зменшуватися, перейшовши від верхнього рядка до нижнього. Навколо нижньої області значення буде меншим за порогове значення, тому ми знаємо, що це зона перекриття цього і наступного рядка. Індекс цього приймається як кінцевий індекс цього рядка та початковий індекс наступного рядка (рис. 2.21).

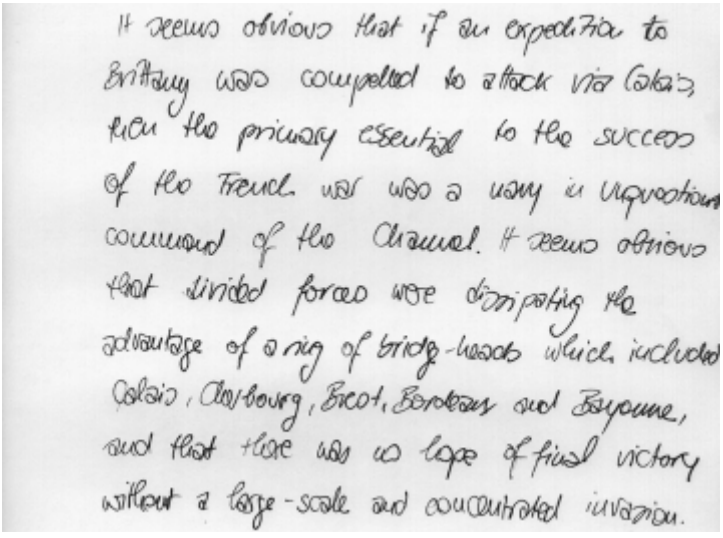


Рисунок 2.19 – Зразок зображення для початку

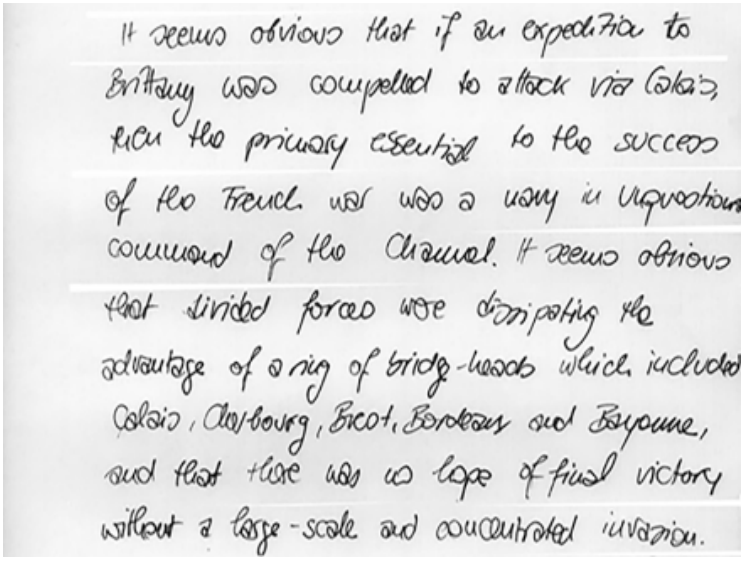


Рисунок 2.20 – Зразок зображення після випрямлення контурів за допомогою афінного перетворення.

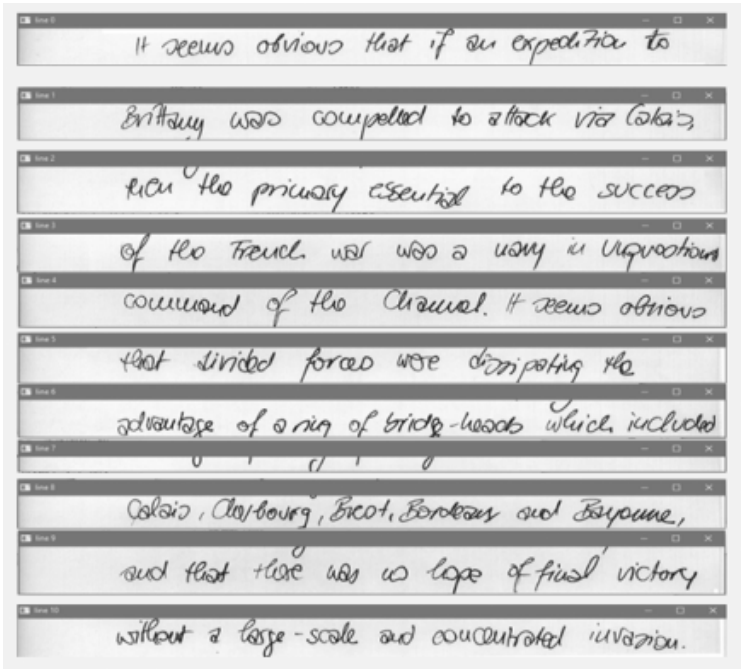


Рисунок 2.21 – Вилучення окремих рядків почерку на зразку зображення.

2.8 **Вилучення міжрядкових інтервалів**

Міжрядковий інтервал почерку визначається наступним чином.

а) Підраховується загальна кількість рядків з горизонтальною проекцією 0, крім верхнього поля. Назвемо a.

б) У витягнутих рядках підраховується загальна кількість рядків з горизонтальною проекцією, меншою за поріг. Назвемо b. Вони складаються з верхньої та нижньої зон ліній. (Ця операція також була частиною пошуку розміру літери.)

в) Нехай n кількість рядків у графічному документі тоді x = (a+b)/n . Дасть середній міжрядковий інтервал

г) Кінцевий міжрядковий інтервал приймається як x розділити на розмір літери, щоб він відповідав розміру почерку (рис. 2.22).

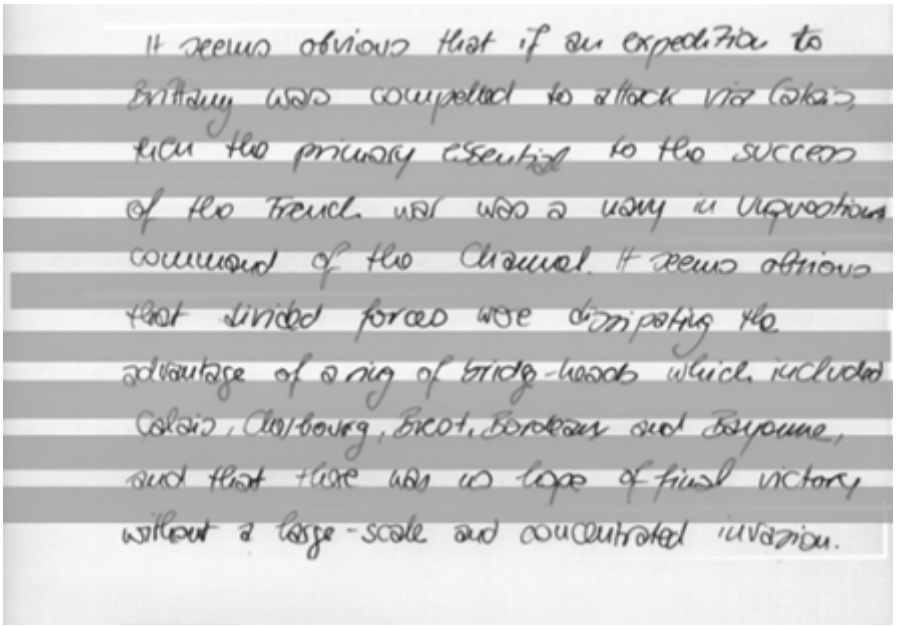


Рисунок 2.22 – Вилучення розміру літери. Темні зони містять серединні зони.Враховується середня висота середньої зони.

Цей алгоритм також реалізовано у функції extractLines()

2.9 **Вилучення інтервалів між словами**

Відстань між словами почерку визначається наступним чином

а) Вертикальна проекція кожного рядка зображення обчислюється в список (масив) Python.
б) Підраховується кількість стовпців у списку зі значенням 0 (стовпець зі значенням усіх пікселів 0, порожня область), крім лівого та правого полів. Нехай буде a.
в) Кількість запусків ненульових стовпців дасть кількість слів або роз'єднаних літер. Нехай так буде b.
г) Далі x=a/b буде середній інтервал між словами.
д) Середня усіх x з оброблених рядків нехай буде y. Інтервал між словами приймається як y розділити на розмір літери, щоб інтервал відповідав розміру почерку (рис. 2.23).

Рисунок 2.23 – Вилучення інтервалів між словами. Темні зони представляють проміжки між словами.

Цей алгоритм також реалізовано у функції extractLines()

2.10 **Вилучення верхньої маржі**

Щоб витягти верхнє поле, ми просто скануємо горизонтальну проекцію зображення зверху для першого запуску 0 с. Число 0 є висотою верхнього поля, яке знову поділено на розмір літери, щоб воно відповідало розміру рукописного введення (рис. 2.24). Алгоритм також реалізований у функції extractLines().

Рисунок 2.24 – Вилучення верхнього поля.

2.11 **Визначення натиску ручки**

Натиск ручки при письмі визначається наступним чином:

а) Зображення (рис. 2.25) перевертається за формулою 3.1:

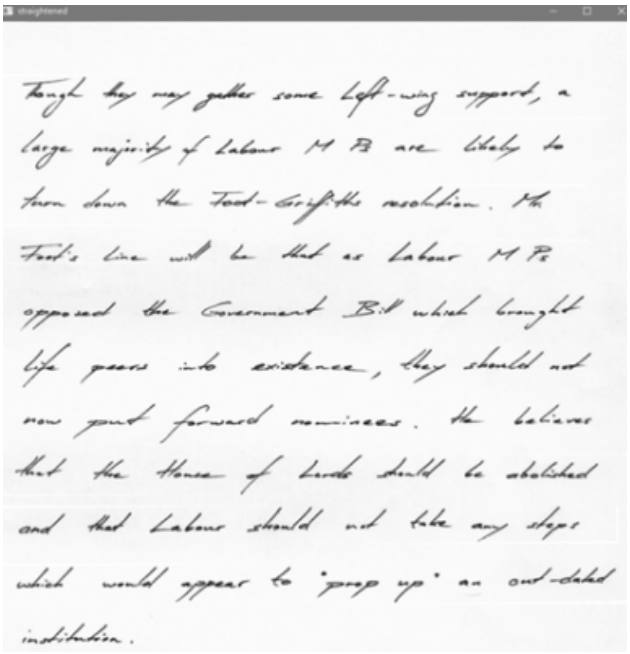
$$\text{dst[x][y]} = 255 src[x][y] \tag{2.1}$$

Цей крок є дуже витратним з точки зору обчислень (рис. 2.26).

б) Інвертований двійковий поріг (THRESH TO ZERO) виконується , якщо src(x, y) нижчий за поріг = 100, нове значення пікселя dst(x, y) буде встановлено на 0, інакше він залишиться недоторканим (рис. 2.27).

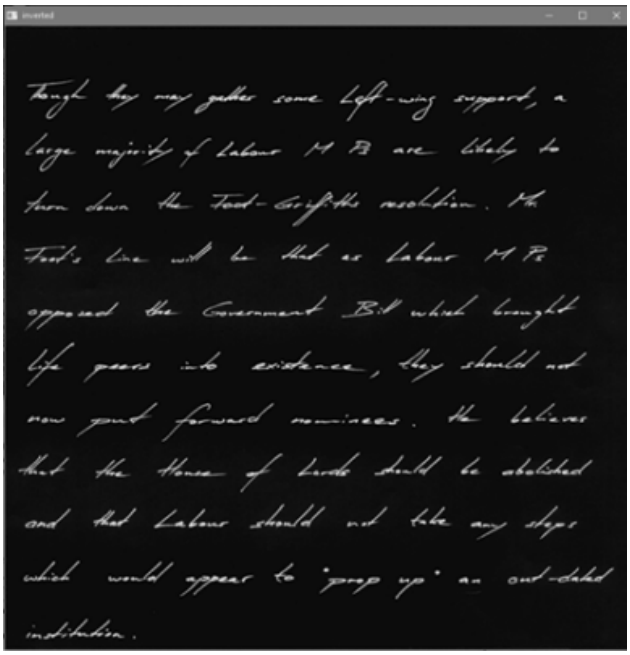
в) За тиск пера береться середнє значення всіх ненульових пікселів. Значення не інвертується знову , чим вище значення тим більший тиск пера.

Цей алгоритм реалізовано у функції barometer().



Though they may gather some Left-wing support, a large majority of Labour M Ps are likely to turn down the Foot-Griffiths resolution. Mr Foot's line will be that as Labour MPs opposed the Government Bill which brought life peers into existence, they should not now put forward nominees. He believes that the House of Lords should be abolished and that Labour should not take any steps which would appear to 'prop up' an out-dated institution.

Рисунок 2.25 – Визначення тиску пера: вхідне зображення



Though they may gather some Left-wing support, a large majority of Labour M Ps are likely to turn down the Foot-Griffiths resolution. Mr Foot's line will be that as Labour MPs opposed the Government Bill which brought life peers into existence, they should not now put forward nominees. He believes that the House of Lords should be abolished and that Labour should not take any steps which would appear to 'prop up' an out-dated institution.

Рисунок 2.26 – Визначення тиску пера: перетворене зображення у відтінках сірого

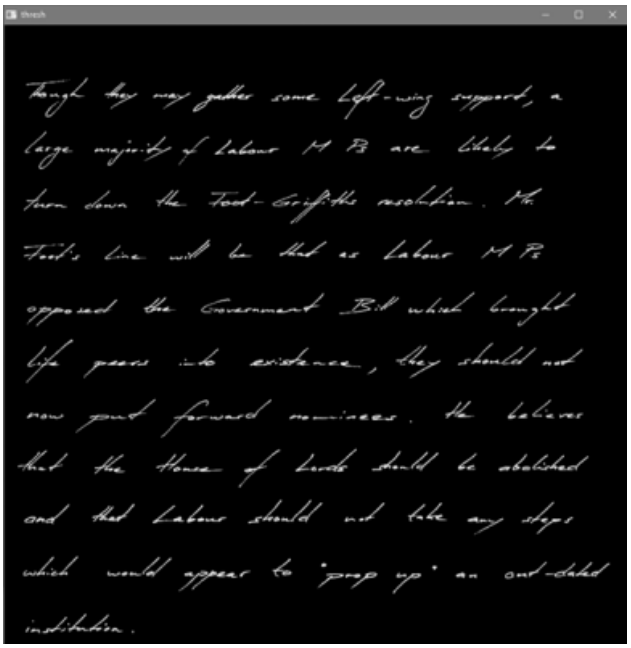


Рисунок 2.27 – Визначення тиску пера: перетворене зображення THRESH TO ZERO

2.12 **Визначення нахилу букв**

Методика визначення нахилу букв почерку заснована на гіпотезі про те, що слово відхиляється, коли кількість стовпців, що містять безперервний штрих, максимальна. Нахил визначається за наступним алгоритмом.

а) Для 9 різних кутів (-45, -30, -15, -5, 0, 5, 15, 30 і 45 градусів) застосовується перетворення зсуву та обчислюється наступна гістограма.

$$X(m) = \frac{h(m)}{\Delta y(m)} \tag{2.2}$$

де $X(m)$ – це вертикальна щільність (кількість пікселів переднього плану на стовпець) у стовпці m;

y(m) – відстань між найвищим і найнижчим пікселем у тому самому стовпці.

Якщо колонка містить безперервний штрих, $X(m) = 1$, інакше $X(m) \in [0, 1]$.

в) Для кожного зображення, перетвореного на зсув, обчислюється така функція.

$$S = \sum (h(i)^2) \tag{2.3}$$

г) За нахил почерку приймається кут, що дає найбільше значення S (рис. 2.28)

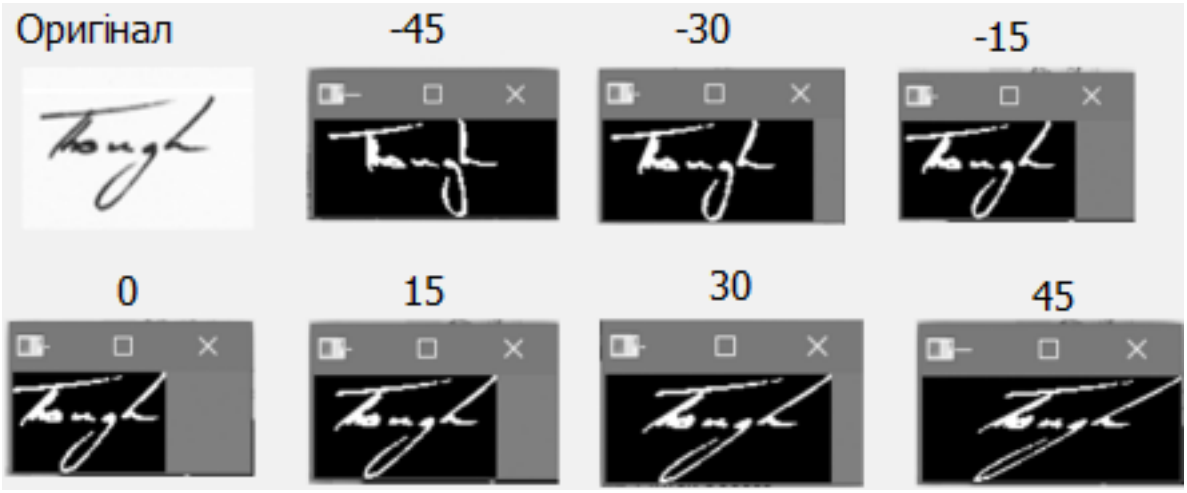


Рисунок 2.28 – Приклад визначення нахилу.

2.13 **Машина опорних векторів**

Сім необроблених ознак, отриманих із зразків рукописного тексту, нормалізуються на дискретні значення відповідно до експериментально визначених порогових значень.

Вісім рис особистості будуть передбачені комбінаціями цих семи ознак (таблиця. 2.1). Отже, буде вісім окремих міток для кожної риси особистості та вісім класифікаторів SVM. Зображення позначаються шляхом вивчення кожного зразка почерку та відповідних нормованих ознак. Використовується реалізація SVM бібліотеки Sci-kit Learn Library, а вісім класифікаторів навчаються за допомогою ядра радіальної базисної функції (RBF).

Таблиця 2.1 – Нормалізація ознак

Ознака	Нормалізоване значення
Базова лінія	0 = спадання 1 = зростаюча 2 = пряма
Верхня маржа	0 = середня або велика 1 = мала(вузька)
Розмір букви	0 = великий 1 = малий 2 = середній
Міжрядковий інтервал	0 = великий 1 = малий 2 = середній
Інтервал між словами	0 = великий 1 = малий 2 = середній
Натиск ручки	0 = сильний 1 = легкий 2 = середній
Нахил букв	0 = дуже нахилений 1 = трохи нахилений 2 = вертикаль 3 = нестабільний

2.14 **База знань**

Головною відмінністю ЕС від інших програмних засобів є наявність бази знань, в якій зберігаються знання експертів про відповідну проблемну область. Знання зберігаються у вигляді сукупності записів деякою мовою подання знань, яка дозволяє легко змінювати та поповнювати базу знань в формі, яку розуміють спеціалісти. В традиційних же програмах знання жорстко „зашиті” в алгоритм і скорегувати їх може лише сам автор програми,

тобто лише програміст.

До останнього часу саме різні мови подання знань (МПЗ) були центральною проблемою при розробці ЕС. Зараз існують десятки мов і моделей знань, серед яких найбільше поширення отримали продукції, семантичні сітки, фрейми, числення предикатів першого порядку і об'єктно-орієнтовані мови програмування.

База знань є ядром експертної системи, сукупністю знань предметної області, що записана на машинному носії. У процесі роботи з експертною системою користувач має можливість поповнювати базу даних і базу знань, таким чином «навчати» систему. Експертна система під час роботи використовує ту базу знань, яка закладена в неї при розробці і поповнюється в процесі експлуатації. Це знання та досвід людей, які фахівцями у конкретній галузі.

Експертна система зберігає дані після обробки зображення у файл (базу знань), що дає змогу в наступних використаннях системи покращити результати аналізу.

База знань має вигляд таблиці і зберігає оброблені данні (рис. 2.29) по графічним зображенням та висновки (рис. 2.30) , що до цих зображень. База знань даної експертної системи налічує 1500 зразків оброблених графічних документів. Це дає змогу робити точні висновки при роботі з новими графічними документами.

Навчання бази знань є важливою частиною розвитку експертної системи. Це визначає, наскільки ефективно система розуміє світ та приймає рішення. Чим більше є обробленої інформації, тим краще вона може вирішувати завдання та вирішувати проблеми. Постійне навчання також дозволяє системі адаптовуватися до змін у середовищі і оновлювати інформацію на основі нових даних.

Оновлена база знань є важливим аспектом для прийняття рішень в реальному часі. Крім того, актуальні знання є ключовим фактором у сферах, де технології швидко розвиваються. Від цього залежить якість обслуговування та здатність системи приймати інформовані рішення в різноманітних

ситуаціях.

Для створення бази знань було взято дані з бази даних рукописного введення IAM дослідницької групи з комп'ютерного зору та штучного інтелекту INF, Бернський університет, Швейцарії

Навчання реалізовано таким чином, що при кожному запуску коли система робить якісь висновки по новому файлу, вона їх записує до бази знань. Якщо ж назва файлу повторюється та є в базі зань система одразу видає збережені дані та не витрачає зусиль на обробку зображення повторно.

Такий метод збереження даних дозволяє уникниту перевантаження однаковими даними самої системи та давати більш точні дані на основі усіх зображень які збережені у базі знань.

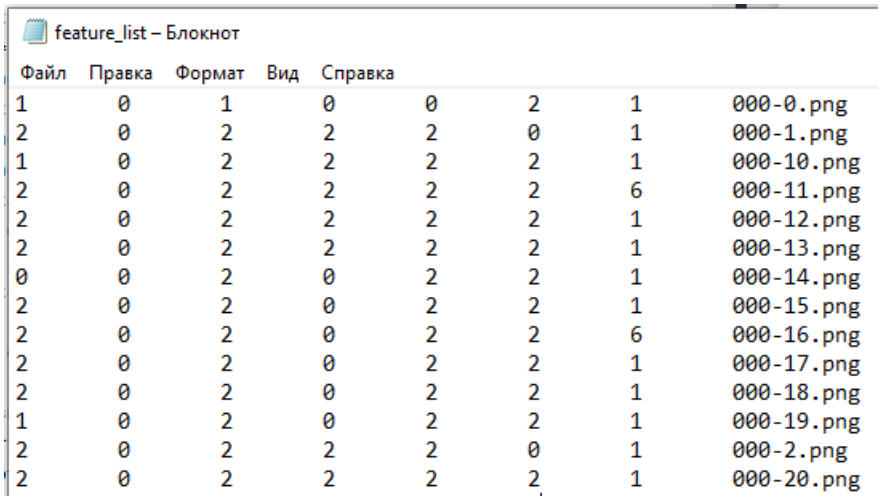


Рисунок 2.29 – База знань. Оброблені значення

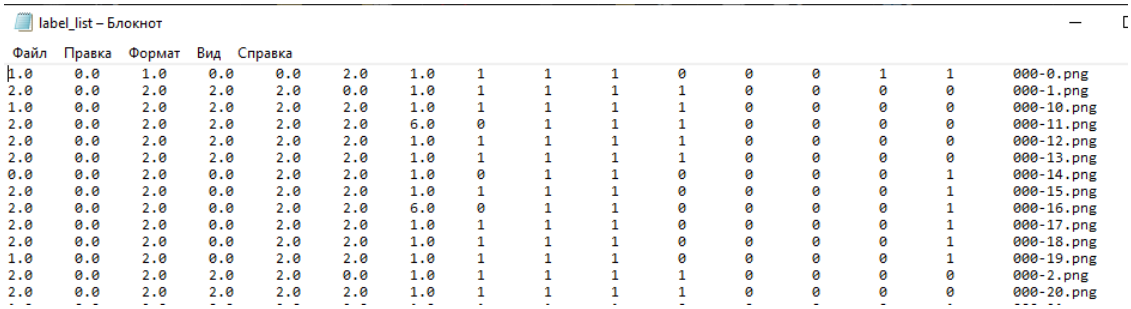


Рисунок 2.30 – База знань. Висновки по графічним документам

2.15 **Висновки по розділу 2**

У даному розділі описано процес розробки експертної системи для аналізу рукописних документів. Описується методика обробки зображення, включаючи вилучення верхнього поля та визначення натиску ручки. Зокрема, важливим етапом є методика визначення нахилу букв, базована на гіпотезі про відхилення слова при максимальній кількості стовпців із безперервним штрихом. Також надається опис визначення нахилу почерку за допомогою алгоритму з перетвореннями зсуву та обчисленням гістограми.

Також використання машини опорних векторів (SVM) для класифікації рис особистості є важливим аспектом для коректного функціонування системи. Зазначається нормалізація семи необроблених ознак та використання восьми класифікаторів SVM для прогнозування рис особистості. Основу експертної системи становить база знань, яка регулярно оновлюється під час роботи з новими файлами. Збереження та використання бази знань сприяє ефективнішому аналізу графічних документів.

3 РОЗРОБКА СЕРВЕРНОЇ АРХІТЕКТУРИ

3.1 Встановлення та оновлення операційної системи

В сучасному світі використання операційної системи Linux, зокрема Ubuntu, може бути ключовим для успішної реалізації проектів, досліджень тощо. Для встановлення та налаштування Ubuntu, потрібно виконати декілька кроків щоб забезпечити оптимальне середовище для коректної роботи нашого сервісу.

Першим етапом встановлення є завантаження образу з офіційного сайту та встановлення з флешки диску тощо.

Наступним кроком буде налаштування та встановлення всіх оновлень. Оскільки сервер працює лише як термінал то всі команди вводяться в командний рядок.

Для встановлення оновлень потрібно прописати данну команду.

```
sudo apt update –y && sudo apt upgrade –y
```

3.2 Встановлення основних програм та бібліотек

Для коректного виконання скриптів Python потрібно встановити інтерпретатор Python

Інтерпретатор Python - це програма, яка виконує програми, написані мовою програмування Python. Вона відповідає за виконання інструкцій, що містяться у вихідному коді Python, рядок за рядком, перетворюючи їх у машинний код, який комп’ютер може розуміти.

Основні функції інтерпретатора Python включають:

- Виконання Коду: Інтерпретатор Python виконує вихідний код Python без необхідності компіляції в машинний код перед виконанням. Це

дозволяє розробникам швидко та легко тестувати та виконувати свій код.

- Динамічна Типізація: Python є мовою з динамічною типізацією, інтерпретатор динамічно визначає типи змінних під час виконання програми.
- Збиральник Сміття: Інтерпретатор Python використовує автоматичне управління пам’ятю, що дозволяє відразу звільнювати пам’ять, використану об’єктами, які вже не потрібні.
- Інтерактивний Режим: Інтерпретатор Python може працювати у відомому як "інтерактивний режим де користувач може введенням команд взаємодіяти безпосередньо з інтерпретатором.
- Підтримка Різних Платформ: Python є переносною мовою програмування, і інтерпретатор Python може працювати на різних операційних системах, таких як Windows, macOS та різні варіанти Unix/Linux.

Для встановлення та перевірки версії інтерпритатора Python потрібно виконати наступні команди

```
sudo apt install python3 –y
python3 ––version
```

Наступний крок це встановлення веб серверу Nginx

Nginx — це веб-сервер та проксі-сервер з відкритим вихідним кодом. Він призначений для обробки статичних ресурсів, таких як HTML-сторінки та зображення, а також виконання завдань проксі-сервера, наприклад, переадресація запитів до веб-додатків (наприклад, додатків, написаних на PHP чи Python).

Для його встановлення та запуску потрібно ввести наступні команду

```
sudo apt install nginx –y
sudo systemctl start nginx
```

Наступний крок встановлення інтерпритатору PHP та модуля FPM веб серверу.

Інтерпретатор PHP 8.1 на сервері Ubuntu - це програмне забезпечення, яке виконує PHP-скрипти на сервері. PHP є серверною мовою програмування, і його інтерпретатор відповідає за виконання PHP-коду. Версія 8.1 означає, що встановлено конкретну версію PHP [28].

PHP і Nginx взаємодіють через FPM (FastCGI Process Manager) або модуль PHP-FPM. FPM є відокремленим процесом, який виконує PHP-скрипти та обслуговує їхні запити

Для встановлення інтерпретатора та модулю веб сервера потрібно ввести наступні команди:

```
sudo apt install php8.1
sudo systemctl start php8.1-fpm
```

Також потрібно встановити всі бібліотеки Python які використані в проекті

3.3 **Створення веб додатку**

3.3.1 Створення сайту

Для створення сайту та відображення інформації потрібно змінити налаштування файл index.html який знаходиться в папці /var/www/html/

В цей файл потрібно ввести код нашого сайту. У нашому випадку потрібно замінити код сайту у файлі index.html. Також потрібно додати додаткові файли для обробки заптів php [29]. Важливо щоб в коді було вказано що кожен файл переназивається як кількість наносекунд від початку комп’ютерного часу(тобто 01.01.1970)

Ось яку структуру буде мати папка /var/www/html/ (рис. 3.3).

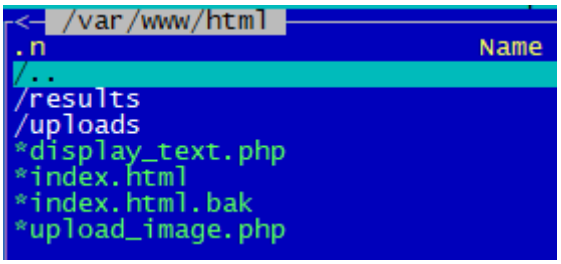


Рисунок 3.1 – Структура папки

Код представлений у файлі index.html(Додаток В. Лістинг коду)

Наступне що потрібно зробити це створити php скрипт який буде отримувати данні зображення [30].

Сам скрипт буде виконуватись після натискання на конпку відправити та завантажувати за допомогою POST запиту зображення на сервер. Потім даний скрипт повинен перенаправляти на сторінку з результатом виконання обробки екпертною системою. Це буде викноувати скрипт upload_image.php

```php
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Перевірка, чи була отримана файлова інформація
    if (isset($_FILES["image"])) {
        $uploadDir = "uploads/"; // Директорія для зберігання завантажених файлів
        $fileName = time() . "_" . basename($_FILES["image"]["name"]);
        $uploadFile = $uploadDir . $fileName;

        // Переміщення зображення з тимчасового місця до папки для завантаження
        if (move_uploaded_file($_FILES["image"]["tmp_name"], $uploadFile)) {
            echo "Файл успішно завантажено.";

            // Створення текстового файлу у папці result з іменем зображення
            $resultDir = "result/";
            $textFileName = $resultDir . pathinfo($fileName, PATHINFO_FILENAME) . ".txt";
            // Перенаправлення на скрипт виводу тексту
            header("Location: display_text.php?filename=" . urlencode($textFileName));
            exit();
        } else {
            echo "Помилка під час завантаження файлу.";
        }
    } else {
        echo "Помилка: файл не був отриманий.";
    }
}
?>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Завантаження зображення</title>
</head>
<body>
    <form action="<?php echo $_SERVER["PHP_SELF"]; ?>" method="post" enctype="multipart/form-data
        <label for="image">Виберіть зображення для завантаження:</label>
        <input type="file" name="image" id="image" accept="image/*">
        <br>
        <button type="submit">Завантажити</button>
    </form>
</body>
</html>
```

Також потрібно розробити скрипт який буде виводити інформацію з текстового файлу в якому є інформація про зображення. Це буде виконувати скрипт dispay_text.php

```
<?php
if (isset($_GET['filename'])) {
    $textFileName = $_GET['filename'];

    // Перевірка наявності текстового файлу
    if (file_exists($textFileName)) {
        // Зчитування текстового вмісту з файлу
        $textFileContent = file_get_contents($textFileName);

        // Відображення тексту
        echo "<h1>Текстовий вміст:</h1>";
        echo "<pre>{$textFileContent}</pre>";
    } else {
        echo "Помилка: текстовий файл не знайдено.";
    }
} else {
    echo "Помилка: не вказано ім'я текстового файлу.";
}
?>


```

3.3.2 Створення обробки завантажених зображень

Наступною задачею буде створити скрипт який буде перевіряти чи не з'явились нові файли в папкі /uploads які знаходиться в /var/www/html/

Після чого цей скрипт повинен запустити скрипт python для встановлення характеристик почерку. І записати ці дані у .txt файл з назвою цієї картинки в папку /results

Для цього потрібно створити простий скрипт мовою BASH та помістити цей файл у /bin/ [31, 32].

```
#!/bin/bash


directory="/var/www/html/uploads"


lock_file="$directory/.script_lock"

if [ -f "$lock_file" ]; then
    echo "Script is busy."
    exit 1
fi

inotifywait -m -e create "$directory" |
while read path action file; do
    if [[ "$action" == "CREATE" ]]; then
        echo "Find new file: $file"
        python3 /home/denis/mag/train_predict.py "$file" e >> "$file".txt
    fi
done

touch "$lock_file"

```

Також потрібно змінити налаштування crontab який буде запускати цей скрипт кожної секунди і перевіряти чи не з'явились нові зображення (рис.3.2)

Щоб скрипт працював йому потрібно дати можливість виконуватись. Для цього в Unix подібних системах є команда chmod. У нашому випадку вона буде виглядати так.

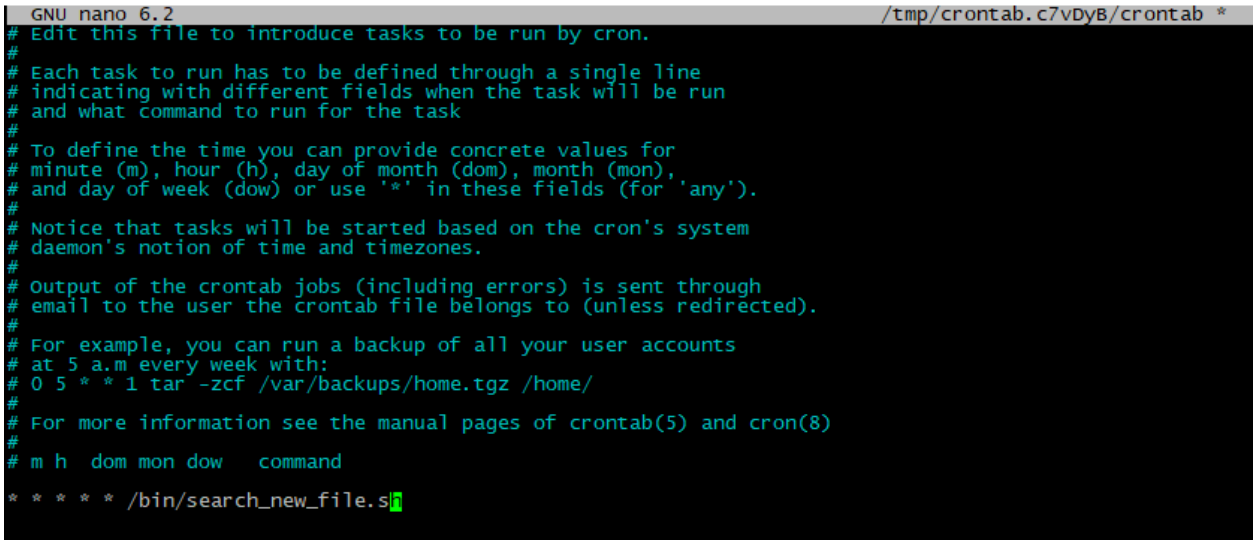


Рисунок 3.2 – Конфігурація Crontab

```
sudo chmod 777 search_new_file.sh
```

3.4 **Тестування роботи створеного веб додатку**

У даному підрозділі показано тестування роботи отриманого веб додатку.

Тестування програмного забезпечення за допомогою графічного документу проводилось для оцінки якості роботи програмного забезпечення при повному проходженні через систему. Тестування розробленого програмного забезпення за допомогою графічного документу необхідне для повного розуміння правильності роботи усіх елементів складної будови програмного забезпечення.

Під час проведення тестування розробленого програмного забезпечення було виявлено достатньо стабільну роботу додатку

Тестування проводилось на двох різних гаджетах з одним і тим самим зображенням.

Для тестування було обрано два гаджета. Перший на ОС Windows 10

(рис. 3.3) та телефон на Android 13 (рис. 3.4).

Рисунок 3.3 – Технічні характеристики ПК з ОС Windows 10

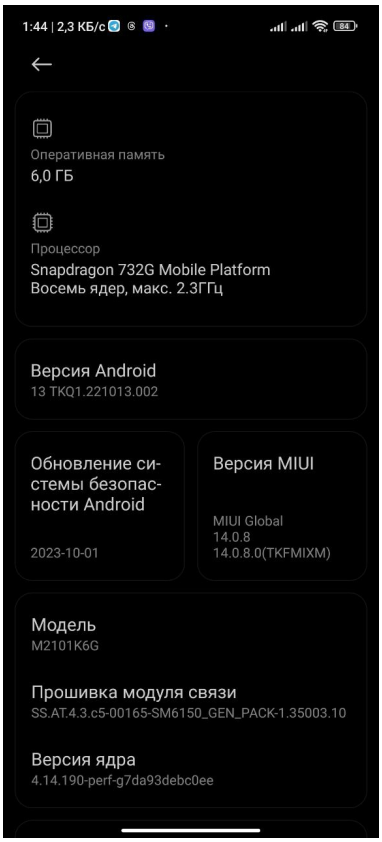


Рисунок 3.4 – Технічні характеристики телефону

Таблиця 3.1 – Показники при роботі програмного забезпечення та швидкості відкриття сторінок на різних гаджетах

ПК	Спроба 1	Спроба 2
ПК з ОС Window 10	2 секунди	3 секунди
Телефон	2 секунди	4 секунди

При перевірці класифікаторів було визначенно що точність кожного класифікатору є максимальною (таблиця 3.2). З отриманих даних можна зробити висновок що система робить точні та правильні висновки щодо психологічного стану людини за її почерком.

Таблиця 3.2 – Показники точності класифікаторів

Класифікатор	Риса особистості	Точність(у відсотках)
1	Емоційна стабільність	100
2	Психічна енергія або сила волі	100
3	Скромність	100
4	Особиста гармонія та гнучкість	100
5	Дисципліна	100
6	Концентрація	100
7	Комунікабельність	100
8	Соціальна ізоляція	100

Результати роботи та інструкція по використанню представлені в додатках

3.5 **Висновки по розділу 3**

Даний розділ присвячено розробці та реалізації веб-додатку на мові програмування Python з використанням технологій Nginx та PHP. Основні особливості мови Python, такі як динамічна типізація та збиральник сміття, дозволяють швидко та легко тестувати код.

Текст доповнений інструкціями щодо встановлення інтерпретатора Python, налаштування Nginx та PHP. Окремо висвітлено аспекти створення веб-додатку та обробки зображень, включаючи скрипти на PHP та BASH.

Тестування підтвердило стабільну роботу системи, зокрема швидкість відкриття сторінок на різних гаджетах не перевищувала 4 секунд, а класи-

фікатори показали максимальну точність. Отже, розроблений веб-додаток ефективно виконує свої функції та може бути використаний для обробки зображень та аналізу рис особистості людини у реальному часі.

4 ЕКОНОМІЧНИЙ РОЗДІЛ

4.1 Технологічний аудит розробленої експертної системи графологічного аналізу з використанням серверної архітектури

Як було відзначено раніше, все більш широке застосування в різних сферах життя та діяльності людей, підприємств та організацій експертних систем графологічного аналізу з використанням серверної архітектури дозволяє об'єктивніше вирішувати проблеми, які постійно виникають в процесів аналізування і прийняття відповідних управлінських рішень. Цей процес об'єктивно пояснюється потребою забезпечення високоточного та швидкого оцінювання написаного рукописного тексту, що має важливе значення в різних сферах життя та діяльності людини.

Тому метою виконаної нами магістерської роботи була розробка заходів з покращення роботи експертних систем, які здатні автоматизовано аналізувати графічні ознаки рукопису для виявлення психологічних і особистісних характеристик людини, та надавати відповідним органам потрібну інформацію для її використання згідно з потребами та запитами (користувачів).

Для досягнення поставленої мети нами було: розроблено бази даних з графічними зразками рукопису і психологічними характеристиками; створено моделі машинного навчання для аналізу рукопису; розроблено серверну архітектуру для запитів користувачів; проведено інтегрування роботи експертної системи з іншими системами, які можуть використовувати результати проведеного аналізу; забезпечено безпеку та конфіденційність отриманої інформації; проведено тестування та оптимізацію експертної системи.

В результаті було розроблене програмне забезпечення та експертна система, які можуть суттєво покращити роботу фахівців, зекономити їх час та ресурси, а також сприяти розвитку сучасних технологій.

Для встановлення комерційного потенціалу розробленої нами експертної системи було запрошено 3-х відомих експертів – д.т.н., професора Паламарчука Є.А., к.т.н., професора Папінова В.М. та к.т.н, доцента Маслія Р.В.

Встановлення комерційного потенціалу розробленої нами експертної системи було здійснено за критеріями, наведеними в таблиці 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання технічного рівня та комерційного потенціалу будь-якої розробки і їх бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
Ринкові перспективи					
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження таблиці 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Висококваліфіковані експерти оцінили розроблену нами експертну систему за допомогою узгодженої і рекомендованої системи балів (див. таблицю 5.2):

Таблиця 4.2 – Результати технологічного аудиту розробленої експертної системи (з використанням балів: 0 – 1 – 2 – 3 - 4)

Критерії	Прізвище, ініціали експертів		
	Паламарчук Є.А.	Папінов В.М.	Маслій Р.В
	Бали, що їх виставили експерти:		
1	4	3	3
2	3	3	4
3	4	4	4
4	3	3	4
5	4	3	3
6	4	4	4
7	3	4	3
8	4	3	3
9	4	3	3
10	4	3	4
11	4	4	4
12	3	3	3
Сума балів	$СБ_1 = 44$	$СБ_2 = 40$	$СБ_3 = 42$
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{44+40+42}{3} = \frac{126}{3} = 42{,}0$		

Встановлення комерційного потенціалу розробленої нами експертної системи будемо здійснювати на основі рекомендацій, наведених в таблиці 4.3.

Таблиця 4.3 – Рівні комерційного потенціалу будь-якої наукової розробки

Середньоарифметична сума балів $\overline{СБ}$, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0 – 10	Низький
11 – 20	Нижче середнього
21 – 30	Середній
31 – 40	Вище середнього
41 – 48	Високий

Оскільки середньоарифметична сума балів, що їх виставили експерти, складає 42,0 балів, то це свідчить, що розроблена нами експертна система графо-

логічного аналізу з використанням серверної архітектури має рівень комерційного потенціалу, який вважається «високим».

Це пояснюється тим, що при розробленні експертної системи графологічного аналізу використовувалися ефективні моделі машинного навчання з їх впровадженням в серверну архітектуру, що дозволило автоматизувати і об'єктивізувати аналіз рукопису.

4.2 Розрахунок витрат на розроблення експертної системи графологічного аналізу

При розробленні експертної системи графологічного аналізу були зроблені певні витрати.

Зокрема:

А). Основна заробітна плата $З_о$ розробників, яка визначається за формулою:

$$З_о = \frac{М}{Т_р} \times t \text{ грн}, \quad (4.1)$$

де М – місячний посадовий оклад розробника, грн; приймемо, що

М = (6700…23000) грн/місяць;

$Т_р$ – число робочих днів в місяці; приймемо $Т_р$ = 23 дні;

t – число днів роботи розробників.

Зроблені розрахунки зведемо до таблиці 5.4:

Таблиця 4.4 – Основна заробітна плата розробників

Найменування посади виконавця	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на оплату праці, грн
1. Науковий керівник магістерської роботи	19400	843,48	20 годин	≈ 2812
2. Магістрант-студент-виконавець	2000 (беремо 6700)	291,30	76	≈ 22139
3. Консультант з економічної частини	18900	821,74	1,5 години	≈ 206 (при 6-годинному робочому дні)
Загалом				$З_о$ = 25 157 грн

Б). Додаткова заробітна плата $З_д$ розробників розраховується як (10…12)% від величини їх основної заробітної плати, тобто:

$$З_д = \alpha \times З_о = (0{,}1 \ldots 0{,}12) \times З_о. \qquad (4.2)$$

Приймемо, що α = 0,117. Тоді для нашого випадку отримаємо:

$$З_д = 0{,}117 \times 25157 = 2943{,}37 \approx 2944 \text{ грн.}$$

В). Нарахування на заробітну плату $НЗП_{зп}$ розробників (дослідників) розраховуються за формулою:

$$НЗП_{зп} = (З_о + З_д) \cdot \frac{\beta}{100}, \qquad (4.3)$$

де β – ставка обов'язкового єдиного внеску на державне соціальне страхування, %. β = 22%. Тоді:

$$НЗН_{зп} = (25157 + 2944) \times 0{,}22 = 6182{,}22 \approx 6183 \text{ грн.}$$

Г). Амортизація основних засобів А, які використовувались під час виконання цієї роботи:

$$A = \frac{Ц \cdot Н_а}{100} \cdot \frac{Т}{12} \text{ грн}, \qquad (4.4)$$

де Ц – загальна балансова вартість основних засобів, грн;

$Н_а$ – річна норма амортизаційних відрахувань. Для нашого випадку можна прийняти, що $Н_а$ = (2,5...25)%;

Т – термін використання основних засобів, місяці.

Зроблені розрахунки зведено в таблицю 4.5.

Таблиця 4.5 – Розрахунок амортизаційних відрахувань

Найменування обладнання, приміщень тощо	Балансова вартість, грн.	Норма амортизації, %	Термін використання, міс.	Величина амортизаційних відрахувань, грн
1. Комп'ютерна техніка, обладнання тощо	73500	25	3,2 (при 75% використанні	3675
2. Приміщення університету, кафедри	25400	3,5	3,2 при 60% використанні	142,24 ≈ 143
Всього				А = 3818 грн

Д). Витрати на матеріали М розраховуються за формулою:

$$M = \sum_1^n H_i \cdot Ц_i \cdot K_i - \sum_1^n B_i \cdot Ц_в \text{ грн.,} \qquad (4.5)$$

де H_i – витрати матеріалу *i*-го найменування, кг; $Ц_i$ – вартість матеріалу *i*-го найменування; K_i – коефіцієнт транспортних витрат, K_i = (1,1…1,15); B_i – маса відходів матеріалу *i*-го найменування; $Ц_в$ – ціна відходів матеріалу *i*-го найменування; n – кількість видів матеріалів.

Е). Витрати на комплектуючі К розраховуються за формулою:

$$K = \sum_1^n H_i \cdot Ц_i \cdot K_i \text{ грн,} \qquad (4.6)$$

де H_i – кількість комплектуючих *i*-го виду, шт.; $Ц_i$ – ціна комплектуючих *i*-го виду; K_i – коефіцієнт транспортних витрат, K_i = (1,1…1,15); n – кількість видів комплектуючих.

Під час виконання роботи загальні витрати на матеріали та комплектуючі склали приблизно 2300 грн.

Ж). Витрати на силову електроенергію $B_е$ розраховуються за формулою:

$$B_е = \frac{В \cdot П \cdot Ф \cdot K_п}{K_д}, \qquad (4.7)$$

де В – вартість 1 кВт-год. електроенергії, в 2023 р. В ≈ 4,5 грн/кВт;

П – установлена потужність обладнання, кВт; П = 1,4 кВт;

Ф – фактична кількість годин роботи обладнання, годин.

Приймемо, що Ф = 245 годин;

$K_п$ – коефіцієнт використання потужності; $K_п$ < 1 = 0,8.

$K_д$ – коефіцієнт корисної дії, $K_д$ = 0,73.

Тоді витрати на силову електроенергію будуть дорівнювати:

$$B_е = \frac{В \cdot П \cdot Ф \cdot K_п}{K_д} = \frac{4{,}5 \cdot 1{,}4 \cdot 245 \cdot 0{,}8}{0{,}73} = 1712{,}22 \approx 1713 \text{ грн.}$$

И). Інші витрати $B_{інш}$ можна прийняти як (50…300)% від основної заробітної плати розробників, тобто:

$$B_{інш} = (0{,}5....3) \times З_о. \qquad (4.8)$$

Для нашого випадку отримаємо:

$$B_{інш} = 1{,}0 \times 25157 = 25157 \text{ грн.}$$

К). Сума всіх попередніх статей витрат складає витрати на виконання роботи безпосередньо розробником-магістрантом – В.

В = 25157+ 2944 + 6183+ 3818 + 2300+1713 + 25157 = 67272 грн.

Л). Загальні витрати на розроблення експертної системи графологічного аналізу $В_{заг}$ становлять:

$$B_{заг} = \frac{B}{\beta}, \quad (4.9)$$

де β – коефіцієнт, який характеризує етап виконання цієї роботи. Можна прийняти, що, $\beta \approx 0{,}80$, оскільки робота потребує незначного доопрацювання.

Тоді: $B_{заг} = \frac{67272}{0{,}80} = 84090{,}00$ грн або приблизно 85 тисяч грн.

Тобто прогнозовані загальні витрати на розроблення експертної системи графологічного аналізу становлять приблизно 85 тисяч грн.

4.3 Розрахунок економічного ефекту від можливої комерціалізації розробленої експертної системи графологічного аналізу

Економічний ефект від впровадження та можливої комерціалізації розробленої нами експертної системи графологічного аналізу пояснюється її значно кращими функціональними характеристиками та можливостями суттєво покращити роботу фахівців в таких галузях, як психологія, психіатрія, рекрутинг, кримінальна юстиція тощо.

Тому нашу розробку можна реалізовувати на ринку дещо дорожче, ніж аналогічні за функціями розробки.

У 2022 році подібні за своїми функціями експертні системи зі значно гіршими характеристиками коштували на ринку приблизно 40 тисяч грн. Тоді розроблену нами експертну систему графологічного аналізу можна буде реалізовувати на ринку в середньому приблизно за 60 тисяч грн або на 20 тисяч грн дорожче.

Аналіз ринку також показав, що потенційна кількість замовників такої системи становить 10-12 замовників на рік. Для розрахунків приймемо 10 клієнтів. Разом з тим, проведений аналіз показав, що кількість таких замовників буде зроста-

ти, тобто можна очікувати зростання попиту на нашу розробку принаймні протягом 3-х років після її впровадження.

Тобто, якщо наша розробка буде впроваджена з 1 січня 2024 року, то її результати будуть виявлятися протягом 2024-го, 2025-го та 2026-го років.

Прогноз зростання попиту на нашу розробку складає по роках:

а) 2024 р. – приблизно +10 шт. до базового року;

б) 2025 р. – +15 шт. до базового року;

в) 2026 р. – +20 шт. до базового року.

Можливе збільшення чистого прибутку $\Delta П_i$, що його може отримати потенційний інвестор від виведення нашої розробки на ринок, становитиме:

$$\Delta П_i = \sum_1^n (\Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\upsilon}{100}), \qquad (4.10)$$

де $\Delta Ц_o$ – покращення основного якісного показника від впровадження результатів нашої розробки у цьому році. Для нашого випадку це є збільшення ціни реалізації нашої розробки $\Delta Ц_o$ = 60 - 40 = + 20 тисяч грн;

N – основний кількісний показник, який визначає обсяг діяльності у році до впровадження результатів розробки; N = 10 шт.;

ΔN – покращення основного кількісного показника від впровадження результатів розробки.

Таке покращення становитиме по роках, відповідно: у 2024 році – + 10 шт., у 2025 році + 15 шт., та у 2026 році + 20 шт.;

$Ц_o$ – основний якісний показник (тобто ціна), який визначає обсяг діяльності у році після впровадження результатів розробки, грн; $Ц_o$= 60 тисяч грн;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки; для нашого випадку n = 3;

λ – коефіцієнт, який враховує сплату податку на додану вартість; $\lambda = 0{,}8333$;

ρ – коефіцієнт, який враховує рентабельність продукту. Рекомендується приймати ρ = (0,2...0,5); візьмемо ρ = 0,5;

υ – ставка податку на прибуток. У 2023-му та наступних роках υ = 18%.

Тоді можливе зростання чистого прибутку $\Delta П_1$ для потенційного інвестора протягом першого року від можливого впровадження нашої розробки (2024 р.) становитиме:

$$\Delta П_1 = [20 \cdot 10 + 60 \cdot 10] \cdot 0{,}8333 \cdot 0{,}5 \cdot (1 - \frac{18}{100}) \approx 274 \text{тис. грн.}$$

Можливе зростання чистого прибутку $\Delta П_2$ для потенційного інвестора від можливого впровадження нашої розробки протягом другого (2025) року складе:

$$\Delta П_2 = [20 \cdot 10 + 60 \cdot 15] \cdot 0{,}8333 \cdot 0{,}5 \cdot (1 - \frac{18}{100}) \approx 376 \text{ тис. грн.}$$

Можливе зростання чистого прибутку $\Delta П_3$ для потенційного інвестора від можливого впровадження нашої розробки протягом третього (2026) року складе:

$$\Delta П_3 = [20 \cdot 10 + 60 \cdot 20] \cdot 0{,}8333 \cdot 0{,}5 \cdot (1 - \frac{18}{100}) \approx 479 \text{ тис. грн.}$$

Приведена вартість зростання всіх чистих прибутків від можливого впровадження нашої розробки становитиме:

$$ПП = \sum_1^m \frac{\Delta П_i}{(1+\tau)^t}, \qquad (4.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої роботи, грн;

т – період часу, протягом якого виявляються результати впровадженої роботи, роки. Для нашого випадку т = 3 роки;

τ – ставка дисконтування. Приймемо τ = 0,10 (10%);

t – період часу від моменту початку розроблення експертної системи до моменту отримання можливих чистих прибутків потенційним інвестором.

Тоді приведена вартість зростання всіх можливих чистих прибутків ПП, що їх може отримати потенційний інвестор від комерціалізації нашої розробки, складе:

$$ПП = \frac{274}{(1+0{,}1)^2} + \frac{376}{(1+0{,}1)^3} + \frac{479}{(1+0{,}1)^4} \approx 227 + 283 + 328 = 838 \text{ тисяч грн.}$$

Теперішня вартість інвестицій PV (вартість стартапу), що повинні бути вкладені для реалізації нашої розробки: $PV = (1{,}0\ldots5) \times В_{заг}$.

Для нашого випадку $PV = (1{,}0\ldots5) \times 85 = 2 \times 85 = 170$ тисяч грн.

Абсолютний ефект від можливих вкладених інвестицій $Е_{абс}$.

$$Е_{абс} = ПП – PV, \quad (4.12)$$

де ПП – приведена вартість збільшення всіх чистих прибутків для інвестора від можливого впровадження нашої розробки, грн;

PV – теперішня вартість інвестицій PV = 170 тисяч грн.

Абсолютний ефект від можливого впровадження нашої розробки складе:

$$Е_{абс} = 838 – 170 = 668 \text{ тисяч грн.}$$

Далі розрахуємо внутрішню дохідність $Е_в$ вкладених інвестицій:

$$E_в = \sqrt[Т_ж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.13)$$

де $Е_{абс}$ – абсолютний ефект вкладених інвестицій; $Е_{абс}$ = 668 тис. грн;

PV –теперішня вартість початкових інвестицій PV = 170 тис. грн;

$Т_ж$ – життєвий цикл розробки, роки.

$Т_ж$ = 4 років (2023-й, 2024-й, 2025-й, 2026-й роки)

Для нашого випадку отримаємо:

$$E_в = \sqrt[4]{1 + \frac{668}{170}} - 1 = \sqrt[4]{1 + 3,9294} - 1 = \sqrt[4]{4,9294} - 1 = 1,49 - 1 = 0,49 = 49\%.$$

Далі визначимо ту мінімальну дохідність, нижче за яку потенційному інвестору не вигідно буде займатися комерціалізацією нашої розробки.

Мінімальна дохідність або мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою:

$$\tau_{мін} = d + f, \quad (4.14)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2022-2023 роках в Україні d = (0,10...0,12);

f – показник, що характеризує ризикованість;

f = (0,1...0,50). Приймемо, що f = 0,25.

Для нашого випадку отримаємо:

$$\tau_{мін} = 0,12 + 0,25 = 0,37 \text{ або } \tau_{мін} = 37\%.$$

Оскільки величина $Е_в$ = 49% > $\tau_{мін}$ = 37%, то потенційний інвестор у принципі може бути зацікавлений у фінансуванні та комерціалізації нашої розробки.

Далі розраховуємо термін окупності коштів, вкладених у можливу комерціалізацію розробленої нами експертної системи графологічного аналізу.

Термін окупності $Т_{ок}$ розраховується за формулою:

$$T_{ок} = \frac{1}{E_в}. \quad (4.15)$$

Для нашого випадку термін окупності $Т_{ок}$ коштів становитиме:

$$T_{ок} = \frac{1}{0,49} = 2,05 \text{ років} < 3 \text{ років},$$

що свідчить про потенційну доцільність комерціалізації розробленої нами експертної системи графологічного аналізу.

Далі проведено моделювання залежності величини внутрішньої дохідності вкладених потенційних інвестицій від рівня інфляції в країні.

Якщо рівень інфляції в країні зросте до 20%, то:

$$ПП = \frac{274}{(1+0,2)^2} + \frac{376}{(1+0,2)^3} + \frac{479}{(1+0,2)^4} \approx 191 + 218 + 231 = 640 \text{ тисяч грн.}$$

Тоді абсолютний ефект від можливого впровадження нашої розробки за три роки складе:

$$Е_{абс} = 640 - 170 = 470 \text{ тисяч грн.}$$

Внутрішня дохідність $Е_в$ вкладених інвестицій становитиме:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1,$$

де $Е_{абс}$ – абсолютний ефект вкладених інвестицій; $Е_{абс}$ = 470 тисяч грн;

PV –теперішня вартість початкових інвестицій PV = 170 тисяч грн.

Для нашого випадку отримаємо:

$$E_в = \sqrt[4]{1 + \frac{470}{170}} - 1 = \sqrt[4]{1 + 2,7647} - 1 = \sqrt[4]{3,7647} - 1 = 1,393 - 1 = 0,393 = 39,3\%.$$

Оскільки величина $Е_в$ = 39,3% > $\tau_{мін}$ = 37%, то потенційний інвестор також може бути зацікавлений у фінансуванні та комерціалізації нашої розробки.

Якщо рівень інфляції в країні зросте до 30%, то:

$$ПП = \frac{274}{(1+0,3)^2} + \frac{376}{(1+0,3)^3} + \frac{479}{(1+0,3)^4} \approx 162 + 171 + 168 = 501 \text{ тисяч грн.}$$

Тоді абсолютний ефект від можливого впровадження нашої розробки за три роки складе:

$$E_{абс} = 501 - 170 = 331 \text{ тисяч грн.}$$

Внутрішня дохідність $E_в$ вкладених інвестицій становитиме:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1,$$

де $E_{абс}$ – абсолютний ефект вкладених інвестицій; $E_{абс} = 331$ тисяч грн;

PV –теперішня вартість початкових інвестицій PV = 170 тисяч грн.

Для нашого випадку отримаємо:

$$E_в = \sqrt[4]{1 + \frac{331}{170}} - 1 = \sqrt[4]{1 + 1{,}94717} - 1 = \sqrt[4]{2{,}9471} - 1 = 1{,}31 - 1 = 0{,}31 = 31\%.$$

Оскільки величина $E_в = 31\% < \tau_{мін} = 37\%$, то потенційний інвестор може бути не зацікавлений у фінансуванні та комерціалізації нашої розробки.

Зроблені розрахунки у вигляді графіків наведено на рис. 5.1.

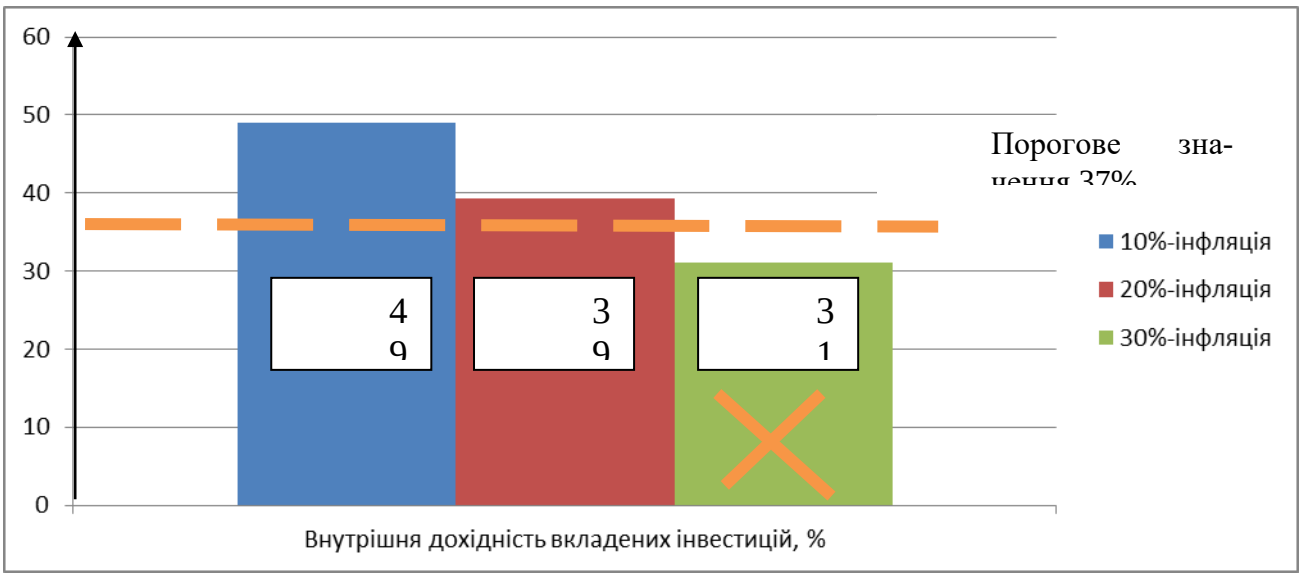


Рисунок 4.1 – Моделювання залежності величини внутрішньої дохідності потенційних інвестицій від рівня інфляції в країні

Аналіз діаграм на рис 5.1 показує, що при рівні інфляції в 10% величина внутрішньої дохідності інвестицій становить $E_в$ = 49%, що більше порогового значення $\tau_{мін} = 37\%$ і тому комерціалізація нашої розробки може бути доцільною.

При рівні інфляції в 20% величина внутрішньої дохідності інвестицій, вкладених в комерціалізацію нашої розробки, становить $Е_в$ = 39,3%, що також більше порогового значення $\tau_{мін}$ = 37%, і тому комерціалізація нашої розробки потенційним інвестором також може бути доцільною. При рівні інфляції в 30% величина внутрішньої дохідності інвестицій, вкладених в можливу комерціалізацію нашої розробки, становить $Е_в$ = 31%, що нижче порогового значення $\tau_{мін}$ = 37%, і тому комерціалізація нашої розробки потенційним інвестором може бути проблематичною. Остаточне рішення з цього питання потребує проведення додаткових розрахунків (можливо – зниження рівня ризикованості вкладень тощо).

Результати виконаної економічної частини магістерської кваліфікаційної роботи зведено у таблицю:

Показники	Задані у ТЗ	Досягнуті у магістерській кваліфікаційній роботі	Висновок
1. Витрати на розробку	Не більше 90 тис. грн	85 тис. грн.	Досягнуто
2. Абсолютний ефект від впровадження розробки, тисяч грн	Не менше 450 тисяч грн	668 тисяч грн (при 10%-інфляції) 470 тисяч грн (при 20%-інфляції	Виконано
3. Внутрішня дохідність інвестицій, %	не менше 37%	49% (при 10%-інфляції) 39,3% (при 20%-інфляції	Досягнуто
4. Термін окупності інвестицій, роки	до 3-ти років	2,05 років	Виконано

Таким чином, основні техніко-економічні показники розробленої нами експертної системи графологічного аналізу, визначені у технічному завданні, виконані.

ВИСНОВКИ

Внаслідок виконання МДР отримано повноціну систему, яка дозволяє дати оцінку деяких рис особистості людини-автора рукописного тексту шляхом аналізу почерку цієї особи за допомогою моделей машинного навчання. Також система є легкою в використанні та доступною для широкого кола користувачів, оскільки працює на будь-якому пристрої, з якого є доступ у мережу інтернет. Обгрунтовано вибір семи ознак почерку, які необхідно визначити з вихідного графічного файлу, а також вісім рис особистості, які можна оцінити за допомогою різних їх комбінацій. Для кожної з рис особистості навчається окремий класифікатор SVM. Також показано метод створення серверної архітектури для функціонування даного програмного забезпечення. Після достатнього навчання моделей експертна система дозволяє дати оцінку кожної з восьми риси особистості людини (на час написання тексту) для нових зразків почерку з високою точністю яка показал 100% при тестуванні.

У системі використовуються різні методи обробки зображень, зокрема самостійно розроблені алгоритми виділення ознак за винятком похилого вилучення. Широко використовувалась бібліотека OpenCV для обробки зображень, а також бібліотека Sci-kit Learn Library для використання стандартної реалізації машин опорних векторів з ядром RBF. Без використання цих бібліотек було б дуже важко досягти результату, який отримано зараз.

Оцінка рис особистості розробленою експертною системою дає задовільні результати за короткий проміжок часу. База знань налічує більше ніж 1500 зразків рукописного тексту та буде збільшуватись далі при використанні програми користувачами. Тим не менш, методи виділення ознак можуть не впоратися з усіма екстремальними випадками стилів рукописного введення, внаслідок цього такі випадки можуть дати неточні результати. Тому рекомендується, щоб зображення рукописного введення було спеціально підготовлено для використання в системі.

Сервер є стресостійким та не потребує ніяких втручань кінцевого ко-

ристувача для коректної роботи експертної системи. Також вся архітектура не є ресурсно затратною, оскільки може працювати на досить слабких віддалених або фізичних серверах.

Економічна частина підтверджує, що данна розробка є вигідною. Та досить швидко буде працювати в прибуток визначений термін окупності 2.05 років.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Бісікало Олег Володимирович, Тульчій Денис Сергійович*. Побудова експертної системи графологічного аналізу. 2022. URL: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2022/paper/view/15647 (дата звернення: 15.10.2023).
2. *Бісікало Олег Володимирович, Тульчій Денис Сергійович*. Розробка експертної системи графологічного аналізу з використанням серверної архітектури. 2023. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2024 (дата звернення: 15.10.2023).
3. *S. Bharti, V. Harave, R Latha*. An overview on structure, characteristics and development of an expert system graphology // *International Journal of Computer Science Trends and Technology*. 2020.
4. *S. Russell, P. Norvig*. Artificial Intelligence: A Modern Approach. NY , USA : Simon Schuster, 1995. 265 с. ISBN: 978-0-13-103805-9.
5. *О. Хлівняк*. ІСТОРІЯ ВИНИКНЕННЯ, РОЗВИТКУ ТА СТАНОВЛЕННЯ ГРАФОЛОГІЧНОЇ ЕКСПЕРТИЗИ ПИСЬМА (СУДОВОЇ ПОЧЕРКОЗНАВЧОЇ ЕКСПЕРТИЗИ) // *Молодий вчений*. 2020.
6. *Elliott Thomas*. The state of the octoverse: machine learning. 2019. URL: https://github.blog/2019-01-24-the-state-of-the-octoverse-machine-learning/ (дата звернення: 15.10.2023).
7. *scikit learn.org*. About us — scikit-learn 0.23.1 documentation. 2021. URL: https://scikit-learn.org/stable/about.html (дата звернення: 17.10.2023).
8. *Tosi Sandro*. Matplotlib for Python Developers. Birmingham , UK : PACKT, 2009. 308 с.
9. *Kaehler Adrian, Bradski Gary*. Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library. Newton , USA : O'REILLY, 2016. 1384 с. ISBN: 978-1491937990.

10. *Davis Doug*. Intel acquires computer vision for iot, automotive. 2016. URL: https://newsroom.intel.com/editorials/intel-acquires-computer-vision-for-iot-automotive/#gs.3s2wfg (дата звернення: 20.10.2023).
11. *Tanenbaum Andrew S., van Steen Maarten*. Сучасні операційні системи. Київ : Видавництво «Видавництво Національного університету «Львівська політехніка», 2007.
12. *Love Robert*. Linux System Programming: Talking Directly to the Kernel and C Library. O'Reilly Media, 2018.
13. *Menasce Daniel A., Almeida Virgilio A. F., Dowdy Lawrence W.* Performance by Design: Computer Capacity Planning By Example. Prentice Hall, 2009.
14. *Documentation Ubuntu*. Official ubuntu documentation. 2023. URL: https://help.ubuntu.com/ (дата звернення: 15.11.2023). Accessed: 11 грудня 2023 р..
15. *Garrels Machtelt*. Bash guide for beginners. 2019. URL: http://www.tldp.org/LDP/Bash-Beginners-Guide/html/index.html (дата звернення: 7.11.2023).
16. *Authors Various*. Bash academy. 2020. URL: https://guide.bash.academy (дата звернення: 11.11.2023).
17. *Nginx*. Nginx Documentation / Nginx, Inc. 2023. Accessed: 11 грудня 2023 р.. URL: https://nginx.org/en/docs/ (дата звернення: 8.11.2023).
18. *Foundation The Apache Software*. Apache HTTP Server Documentation / The Apache Software Foundation. 2023. Accessed: 11 грудня 2023 р.. URL: https://httpd.apache.org/docs/ (дата звернення: 15.11.2023).
19. *Antony D John, Cap OFM*. Personality Profile Through Handwriting Analysis. Anugraha Publications, 2008.
20. A comprehensive survey on impulse and gaussian denoising filters for digital images / Mehdi Mafi, Harold Martin, Jean Andrian, Mercedes Cabrerizo // *Signal Processing*. 2018. С. 157.

21. *Lagendijk R. L., Biemond J.* Basic methods for image restoration and identification // *Handbook of image video processing, 2nd ed.* 2005. С. 167–181.
22. *Cattin Philippe*. Image Restoration. Introduction to Signal and Image Processing. Basel , Switzerland : MIAC, 2016. 70 с.
23. *W. Eric*. Affine transformation. 2015. URL: https://mathworld.wolfram.com/AffineTransformation.html (дата звернення: 21.10.2023).
24. *Nefedov Alexey*. Support Vector Machines: A Simple Tutorial. М., 2016. 35 с.
25. *Cristianini Nello, Shawe-Taylor John*. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. UK : Cambridge University Press, 2000. ISBN: 978-1-139-64363-4.
26. Development of an automated handwriting analysis system / Vikram Kamath, Nikhil Ramaswamy, Navin Karanth та ін. // *Journal of Engineering and Applied Sciences*. 2011.
27. A Review of Human Graphology Analysis and Brainwaves / NF Akila, EMNEM Nasir, N Fuad та ін. ; IOP Publishing. 2020. Т. 917. С. 012048.
28. *Lockhart Josh*. Modern PHP: New Features and Good Practices. Sebastopol, CA, USA : O'Reilly Media, 2015.
29. Php.net, The PHP Group. 2023. URL: https://www.php.net/ (дата звернення: 11.10.2023). Accessed: 11 грудня 2023 р..
30. *Lerdorf Rasmus, Tatroe Kevin, MacIntyre Peter*. Programming PHP. Sebastopol, CA, USA : O'Reilly Media, 2013.
31. *O'Herren Terrence*. Learning the Linux Command Line: A Newbies Guide. Independently published, 2014.
32. *Bresnahan Christine, Blum Richard*. Linux Command Line and Shell Scripting Bible. Hoboken, NJ, USA : Wiley, 2015.

Додатки

Додаток А

Технічне завдання на магістерську кваліфікаційну роботу

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ЗАТВЕРДЖУЮ
Завідувач кафедри АІІТ
____ д.т.н., проф. Олег БІСІКАЛО
«__» _____2023 року

ТЕХНІЧНЕ ЗАВДАННЯ
на магістерську кваліфікаційну роботу

Розробка експертної системи графологічного аналізу з використанням серверної архітектури

08-31.МКР.018.02.000 ТЗ

Керівник: д.т.н., проф., зав. каф. АІІТ
_____ Олег БІСІКАЛО
«05» жовтня 2023 р.
Розробив студент гр. 1АКІТ-22м
_____Денис ТУЛЬЧІЙ
«05» жовтня 2023 р.

Вінниця 2023

1. Назва та галузь застосування.
Розробка експертної системи графологічного аналізу з використанням серверної архітектури. Автоматизація та комп'ютерно-інтегровані технології.
2. Підстава для проведення робіт.
Підставою для виконання роботи є наказ №__ по ВНТУ від «__» _____2023р., та індивідуальне завдання на МКР, затверджене протоколом №__ засідання кафедри АІІТ від «__» _____ 2023р.
3. Мета та призначення роботи.
Метою магістерської дипломної роботи є розробка системи графологічного аналізу з використанням серверної архітектури.
4. Джерела розробки:
4.1 Методичні вказівки до виконання магістерських дипломних робіт (проектів) для студентів спеціальностей 126 – «Інформаційні системи та технології», 151 – «Автоматизація та комп’ютерно-інтегровані технології» / Уклад. Р. Н. Квєтний, О. М. Бевз, О. В. Бісікало. Вінниця : ВНТУ, 2019. 26 с.
4.2 ДСТУ 2481-94 Системи оброблення інформації. Інтелектуальні інформаційні технології. Терміни та визначення.
4.3 ДСТУ ISO/IEC/IEEE 16326:2015 Розроблення систем та програмного забезпечення. Процеси життєвого циклу. Керування проектами (ISO/IEC/IEEE 16326:2009, IDT).
4.4 ДСТУ ISO/IEC 12207:2016 Інженерія систем і програмного забезпечення. Процеси життєвого циклу програмного забезпечення (ISO/IEC 12207:2008, IDT).
4.5. ДСТУ 3330-96 (ГОСТ 34.321-96) Інформаційні технології. Система стандартів з баз даних. Еталонна модель керування даними
5. Показники призначення
Основні технічні вимоги та мінімальні системні вимоги до програми:
5.1 Мови програмування – Python, PHP, BASH.
5.2 Оперативна пам’ять – 2 GB
5.3 Вільного місця – 1 GB
5.4 Бібліотека для створення експертної системи –SciPy, NumPy, Matplotlib
5.5 Галузь використання – комерційна
5.6 Браузер – Opera, Firefox.
5.7 Середовище розробки – PyCharm, Kate.
Вихідні дані для проведення робіт:
Вихідними даними для проведення робіт є індивідуальне завдання на магістерську дипломну роботу від 01.02.2022 р.
Результати роботи програми: Виконано розробку експертної системи з використанням серверної архітектури. Описано архітектуру серверної частини, використовуючи BASH, PHP, HTML, CSS також встановлено зв’язок з експертною системою для обробки отриманих графічних документів. Розроблена система реалізована та успішно протестована.
6. Економічні показники
До економічних показників входять:
- витрати на розробку 85 тисяч гривень.
- приведена вартість прибутку за 3 роки 331 тисяча гривень
- мінімальна дохідність 668 тисяч грн при 10% інфляції

- термін окупності 2,05 років
7. Стадії розробки:
а) Аналіз предметної області та визначення аналогів програмних продуктів ____ – ____01.09 – 26.09
б) Вибір технологічних засобів та розробка архітектури 01.09 – 19.09
в) Обґрунтування та проєктування програмного забезпечення 22.09 – 09.10
г) Розробка технічного завдання 12.09 – 28.09
д) Економічна частина 10.10 – 25.10
е) Аналіз результатів роботи та формування завдань подальших досліджень 31.09 – 07.10
ж) Оформлення матеріалів до захисту МКР 15.09 – 15.11

8. Порядок контролю та приймання

Рубіжний контроль провести до «___» _____ 2023 р.
Попередній захист МКР провести «___» _____ 2023 р.
Захист МКР провести до «___» _____ 2023 р.

Розробив студент групи 1АКІТ-22м _____ Денис ТУЛЬЧІЙ

Додаток Б

Ілюстрована частина

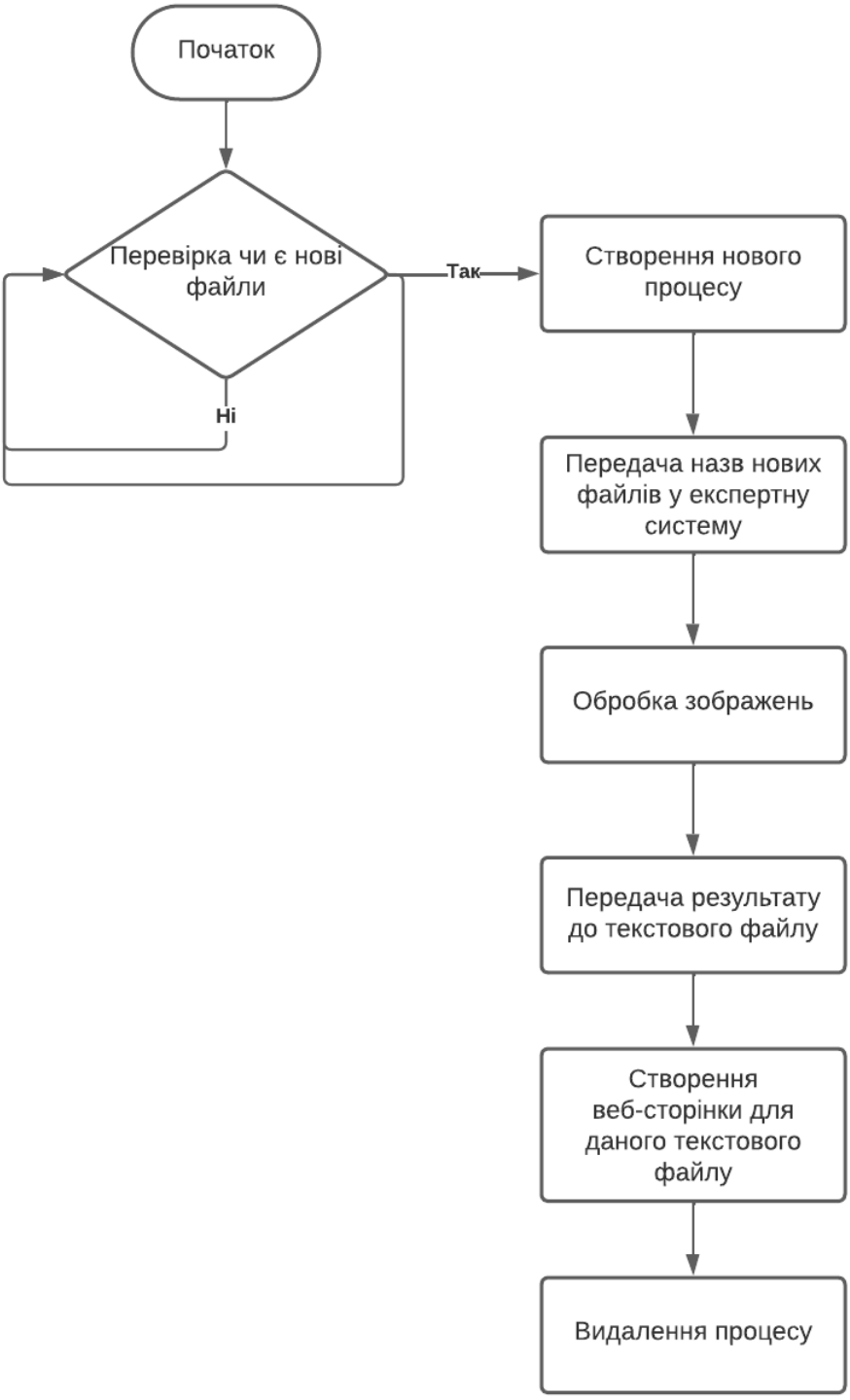


Рисунок Б.1 – Алгоритм логіки роботи серверу

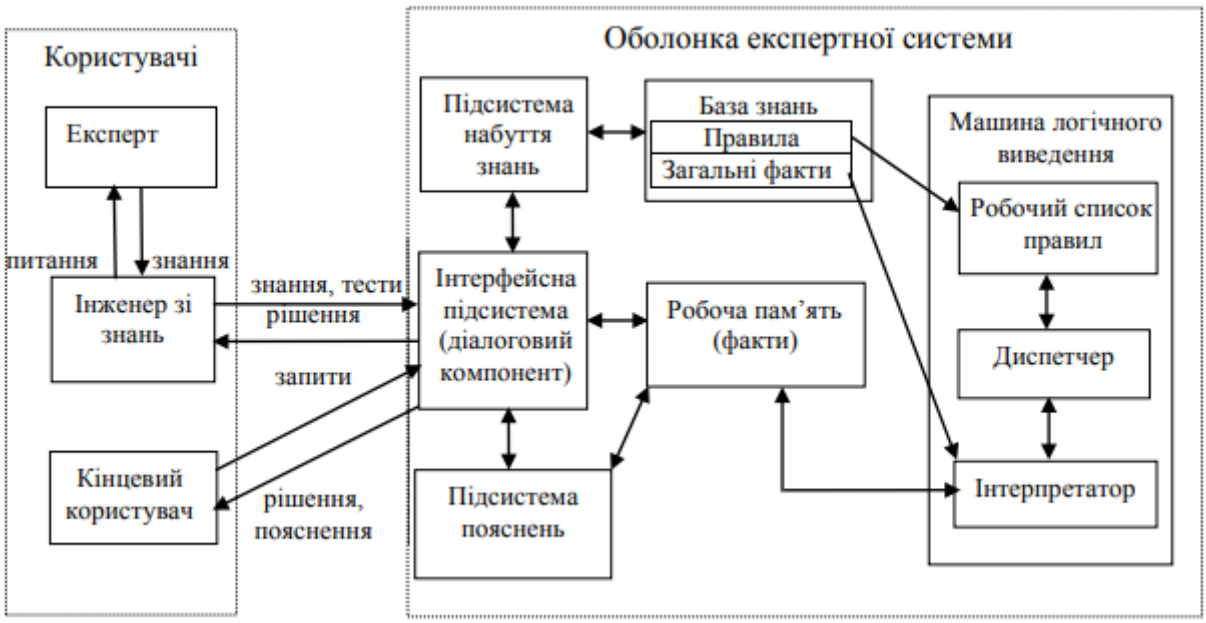


Рисунок Б.2 – Схема експертної частини

```
Info: label_list found.
Classifier 1 accuracy:  1.0
Classifier 2 accuracy:  1.0
Classifier 3 accuracy:  1.0
Classifier 4 accuracy:  1.0
Classifier 5 accuracy:  1.0
Classifier 6 accuracy:  1.0
Classifier 7 accuracy:  1.0
Classifier 8 accuracy:  1.0

**************************************************
Slant determined to be straight.
Baseline Angle: DESCENDING
Top Margin: MEDIUM OR BIGGER
Letter Size: SMALL
Line Spacing: SMALL
Word Spacing: SMALL
Pen Pressure: LIGHT
Slant: STRAIGHT
Emotional Stability:  [0.]
Mental Energy or Will Power:  [1.]
Modesty:  [1.]
Personal Harmony and Flexibility:  [0.]
Lack of Discipline:  [0.]
Poor Concentration:  [1.]
Non Communicativeness:  [0.]
Social Isolation:  [0.]
--------------------------------------------------
```

Рисунок Б.3 – Результат роботи програми на сервері

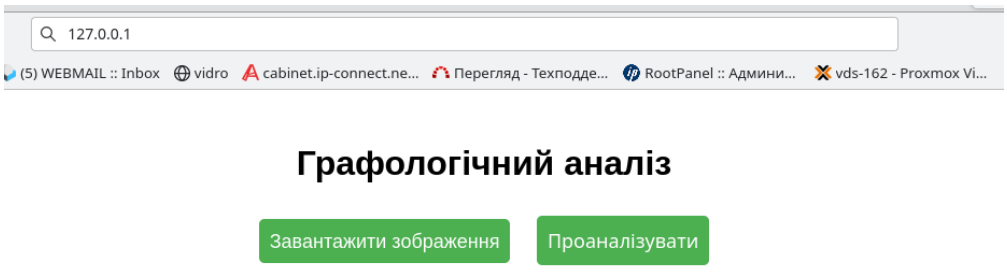


Рисунок Б.4 – Початкова сторінка веб-сайту

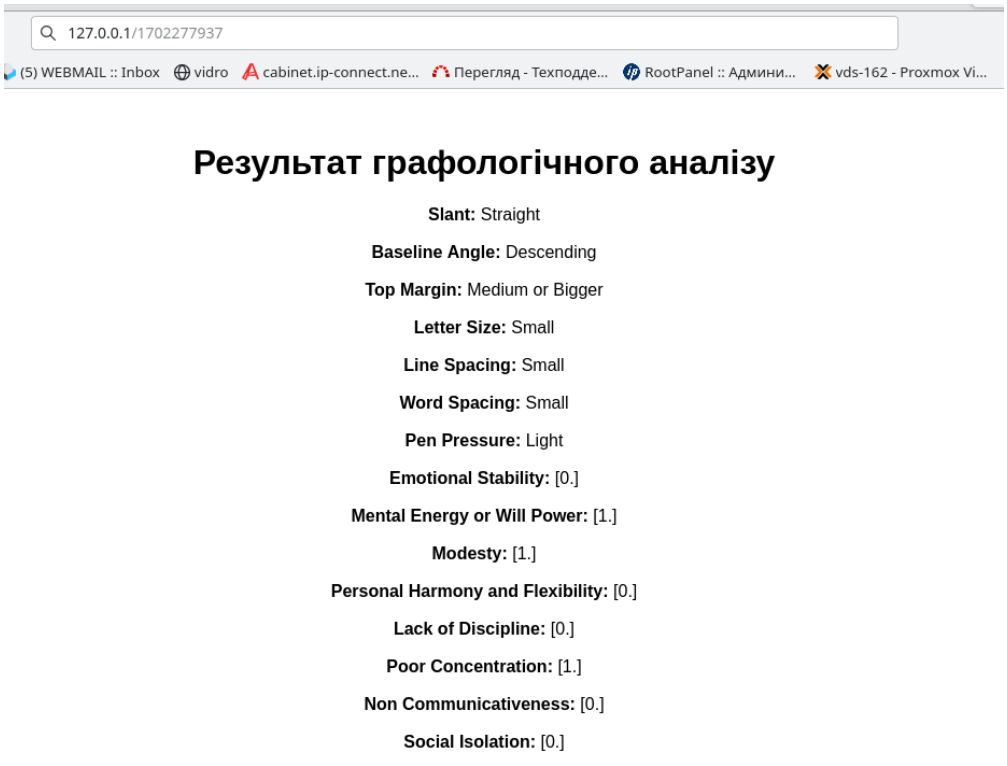


Рисунок Б.5 – Сторінка з виведенням результату

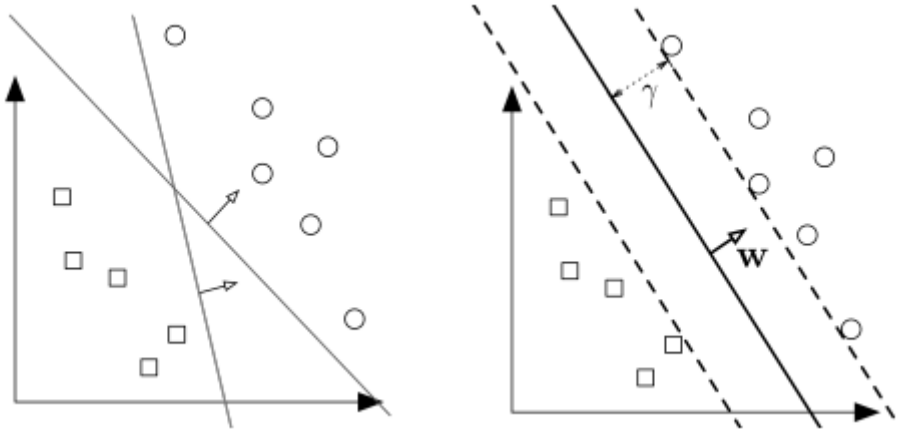


Рисунок Б.6 – Приклад роботи моделі SVM

Додаток В

Лістинг програми

```
import numpy as np
import cv2
import math
from matplotlib import pyplot as plt


ANCHOR_POINT = 6000
MIDZONE_THRESHOLD = 15000
MIN_HANDWRITING_HEIGHT_PIXEL = 20


BASELINE_ANGLE = 0.0
TOP_MARGIN = 0.0
LETTER_SIZE = 0.0
LINE_SPACING = 0.0
WORD_SPACING = 0.0
PEN_PRESSURE = 0.0
SLANT_ANGLE = 0.0

def bilateralFilter(image, d):
        image = cv2.bilateralFilter(image,d,50,50)
        return image


def medianFilter(image, d):
        image = cv2.medianBlur(image,d)
        return image


def threshold(image, t):
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        ret,image = cv2.threshold(image,t,255,cv2.THRESH_BINARY_INV)
        return image




def dilate(image, kernalSize):
        kernel = np.ones(kernalSize, np.uint8)
        image = cv2.dilate(image, kernel, iterations=1)
        return image


```

```
45  def erode(image, kernalSize):
46          kernel = np.ones(kernalSize, np.uint8)
47          image = cv2.erode(image, kernel, iterations=1)
48          return image
49
50
51  def straighten(image):
52
53          global BASELINE_ANGLE
54
55          angle = 0.0
56          angle_sum = 0.0
57          contour_count = 0
58
59
60          positive_angle_sum = 0.0
61          negative_angle_sum = 0.0
62          positive_count = 0
63          negative_count = 0
64
65
66          filtered = bilateralFilter(image, 3)
67
68          thresh = threshold(filtered, 120)
69
70          dilated = dilate(thresh, (5 ,100))
71
72          ctrs,hier = cv2.findContours(dilated.copy(),
73          cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
74
75          for i, ctr in enumerate(ctrs):
76                  x, y, w, h = cv2.boundingRect(ctr)
77
78                  if h>w or h<MIN_HANDWRITING_HEIGHT_PIXEL:
79                          continue
80
81                  roi = image[y:y+h, x:x+w]
82
83                  if w < image.shape[1]/2 :
84                          roi = 255
85                          image[y:y+h, x:x+w] = roi
86                          continue
87
88                  rect = cv2.minAreaRect(ctr)
89                  center = rect[0]
90                  angle = rect[2]
91
92                  if angle < -45.0:
93                          angle += 90.0;
```

```
94
95                  rot = cv2.getRotationMatrix2D(((x+w)/2,(y+h)/2), angle, 1)
96                  extract = cv2.warpAffine(roi, rot, (w,h),
97                  borderMode=cv2.BORDER_CONSTANT, borderValue=(255,255,255))
98
99
100                 image[y:y+h, x:x+w] = extract
101
102                 angle_sum += angle
103                 contour_count += 1
104
105         if contour_count == 0.0:
106                 mean_angle = angle_sum
107         else:
108                 mean_angle = angle_sum / contour_count
109
110         BASELINE_ANGLE = mean_angle
111         return image
112
113 def horizontalProjection(img):
114
115     (h, w) = img.shape[:2]
116     sumRows = []
117     for j in range(h):
118         row = img[j:j+1, 0:w]
119         sumRows.append(np.sum(row))
120     return sumRows
121
122
123 def verticalProjection(img):
124     column
125     (h, w) = img.shape[:2]
126     sumCols = []
127     for j in range(w):
128         col = img[0:h, j:j+1]
129         sumCols.append(np.sum(col))
130     return sumCols
131
132
133 def extractLines(img):
134
135         global LETTER_SIZE
136         global LINE_SPACING
137         global TOP_MARGIN
138
139
140         filtered = bilateralFilter(img, 5)
141
142         thresh = threshold(filtered, 160)
```

```
143
144         hpList = horizontalProjection(thresh)
145
146
147         topMarginCount = 0
148         for sum in hpList:
149
150                 if(sum<=255):
151                         topMarginCount += 1
152                 else:
153                         break
154
155
156         lineTop = 0
157         lineBottom = 0
158         spaceTop = 0
159         spaceBottom = 0
160         indexCount = 0
161         setLineTop = True
162         setSpaceTop = True
163         includeNextSpace = True
164         space_zero = []
165         lines = []
166
167         for i, sum in enumerate(hpList):
168                 if(sum==0):
169                         if(setSpaceTop):
170                                 spaceTop = indexCount
171                                 setSpaceTop = False
172                         indexCount += 1
173                         spaceBottom = indexCount
174                         if(i<len(hpList)-1):
175                                 if(hpList[i+1]==0):
176                                         continue
177
178                         if(includeNextSpace):
179                                 space_zero.append(spaceBottom-spaceTop)
180                         else:
181                                 if (len(space_zero)==0):
182                                         previous = 0
183                                 else:
184                                         previous = space_zero.pop()
185                                 space_zero.append(previous + spaceBottom-lineTop)
186                         setSpaceTop = True
187
188
189                 if(sum>0):
190                         if(setLineTop):
191                                 lineTop = indexCount
```

```
192                             setLineTop = False
193                     indexCount += 1
194                     lineBottom = indexCount
195                     if(i<len(hpList)-1):
196                             if(hpList[i+1]>0):
197                                     continue
198
199                             if(lineBottom-lineTop<20):
200                                     includeNextSpace = False
201                                     setLineTop = True
202                                     continue
203                     includeNextSpace = True
204
205                     lines.append([lineTop, lineBottom])
206                     setLineTop = True
207
208
209     fineLines = []
210
211             anchor = line[0]
212             anchorPoints = []
213             upHill = True
214             downHill = False
215             segment = hpList[line[0]:line[1]]
216
217             for j, sum in enumerate(segment):
218                     if(upHill):
219                             if(sum<ANCHOR_POINT):
220                                     anchor += 1
221                                     continue
222                             anchorPoints.append(anchor)
223                             upHill = False
224                             downHill = True
225                     if(downHill):
226                             if(sum>ANCHOR_POINT):
227                                     anchor += 1
228                                     continue
229                             anchorPoints.append(anchor)
230                             downHill = False
231                             upHill = True
232
233
234
235             if(len(anchorPoints)<2):
236                     continue
237
238             lineTop = line[0]
239             for x in range(1, len(anchorPoints)-1, 2):
240
```

```
241                     lineMid = (anchorPoints[x]+anchorPoints[x+1])/2
242                     lineBottom = lineMid
243
244                     if(lineBottom-lineTop < 20):
245                             continue
246                     fineLines.append([lineTop, lineBottom])
247                     lineTop = lineBottom
248             if(line[1]-lineTop < 20):
249                     continue
250             fineLines.append([lineTop, line[1]])
251
252
253     space_nonzero_row_count = 0
254     midzone_row_count = 0
255     lines_having_midzone_count = 0
256     flag = False
257     for i, line in enumerate(fineLines):
258             segment = hpList[line[0]:line[1]]
259             for j, sum in enumerate(segment):
260                     if(sum<MIDZONE_THRESHOLD):
261                             space_nonzero_row_count += 1
262                     else:
263                             midzone_row_count += 1
264                             flag = True
265
266
267             if(flag):
268                     lines_having_midzone_count += 1
269                     flag = False
270
271
272     if(lines_having_midzone_count == 0): lines_having_midzone_count = 1
273
274
275     total_space_row_count = space_nonzero_row_count + np.sum(space_zero[1:-1])
276     average_line_spacing = float(total_space_row_count) / lines_having_midzone_count
277     average_letter_size = float(midzone_row_count) / lines_having_midzone_count
278
279     LETTER_SIZE = average_letter_size
280
281     if(average_letter_size == 0): average_letter_size = 1
282
283     relative_line_spacing = average_line_spacing / average_letter_size
284     LINE_SPACING = relative_line_spacing
285
286     relative_top_margin = float(topMarginCount) / average_letter_size
287     TOP_MARGIN = relative_top_margin
288
289
```

```
290
291
292         return fineLines
293
294
295 def extractWords(image, lines):
296
297         global LETTER_SIZE
298         global WORD_SPACING
299
300
301         filtered = bilateralFilter(image, 5)
302
303     binary thresholding
304         thresh = threshold(filtered, 180)
305
306
307         width = thresh.shape[1]
308         space_zero = []
309         words = []
310
311
312         for i, line in enumerate(lines):
313                 extract = thresh[line[0]:line[1], 0:width]
314                 vp = verticalProjection(extract)
315
316
317                 wordStart = 0
318                 wordEnd = 0
319                 spaceStart = 0
320                 spaceEnd = 0
321                 indexCount = 0
322                 setWordStart = True
323                 setSpaceStart = True
324                 includeNextSpace = True
325                 spaces = []
326
327
328                 for j, sum in enumerate(vp):
329
330                         if(sum==0):
331                                 if(setSpaceStart):
332                                         spaceStart = indexCount
333                                         setSpaceStart = False
334                                 indexCount += 1
335                                 spaceEnd = indexCount
336                                 if(j<len(vp)-1):
337                                         if(vp[j+1]==0):
338                                                 continue
```

```
339
340
341                                 if((spaceEnd-spaceStart) > int(LETTER_SIZE/2)):
342                                         spaces.append(spaceEnd-spaceStart)
343
344                                 setSpaceStart = True
345
346
347                         if(sum>0):
348                                 if(setWordStart):
349                                         wordStart = indexCount
350                                         setWordStart = False
351                                 indexCount += 1
352                                 wordEnd = indexCount
353                                 if(j<len(vp)-1):
354                                         if(vp[j+1]>0):
355
356                                 count = 0
357                                 for k in range(line[1]-line[0]):
358                                         row = thresh[line[0]+k:line[0]+k+1, wordStart:wordEnd]
359                                         if(np.sum(row)):
360                                                 count += 1
361                                 if(count > int(LETTER_SIZE/2)):
362                                         words.append([line[0], line[1], wordStart, wordEnd])
363
364                                 setWordStart = True
365
366                 space_zero.extend(spaces[1:-1])
367
368
369         space_columns = np.sum(space_zero)
370         space_count = len(space_zero)
371         if(space_count == 0):
372                 space_count = 1
373         average_word_spacing = float(space_columns) / space_count
374         if LETTER_SIZE == 0.0:
375                 relative_word_spacing = average_word_spacing
376         else:
377                 relative_word_spacing = average_word_spacing / LETTER_SIZE
378
379         WORD_SPACING = relative_word_spacing
380
381
382         return words
383
384
385 def extractSlant(img, words):
386
387         global SLANT_ANGLE
```

```
388
389
390         theta = [-0.785398, -0.523599, -0.261799, -0.0872665, 0.01,
391         0.0872665, 0.261799, 0.523599, 0.785398]
392
393
394
395         s_function = [0.0] * 9
396         count_ = [0]*9
397
398
399         filtered = bilateralFilter(img, 5)
400
401
402         thresh = threshold(filtered, 180)
403
404
405
406         for i, angle in enumerate(theta):
407                 s_temp = 0.0
408                 count = 0
409
410
411                 for j, word in enumerate(words):
412                         original = thresh[word[0]:word[1], word[2]:word[3]]
413
414                         height = word[1]-word[0]
415                         width = word[3]-word[2]
416
417
418                         shift = (math.tan(angle) * height) / 2
419
420
421                         pad_length = abs(int(shift))
422
423
424                         blank_image = np.zeros((height,width+pad_length*2,3), np.uint8)
425                         new_image = cv2.cvtColor(blank_image, cv2.COLOR_BGR2GRAY)
426                         new_image[:, pad_length:width+pad_length] = original
427
428
429                         (height, width) = new_image.shape[:2]
430                         x1 = width/2
431                         y1 = 0
432                         x2 = width/4
433                         y2 = height
434                         x3 = 3*width/4
435                         y3 = height
436
```

```
437                     pts1 = np.float32([[x1,y1],[x2,y2],[x3,y3]])
438                     pts2 = np.float32([[x1+shift,y1],[x2-shift,y2],[x3-shift,y3]])
439                     M = cv2.getAffineTransform(pts1,pts2)
440                     deslanted = cv2.warpAffine(new_image,M,(width,height))
441
442                     vp = verticalProjection(deslanted)
443
444                     for k, sum in enumerate(vp):
445
446                             if(sum == 0):
447                                     continue
448
449
450                             num_fgpixel = sum / 255
451
452
453                             if(num_fgpixel < int(height/3)):
454                                     continue
455
456
457                             column = deslanted[0:height, k:k+1]
458                             column = column.flatten()
459
460
461
462                             for l, pixel in enumerate(column):
463                                     if(pixel==0):
464                                             continue
465                                     break
466
467                             for m, pixel in enumerate(column[::-1]):
468                                     if(pixel==0):
469                                             continue
470                                     break
471
472
473                             delta_y = height - (l+m)
474
475                             h_sq = (float(num_fgpixel)/delta_y)**2
476
477                             h_wted = (h_sq * num_fgpixel) / height
478
479                             s_temp += h_wted
480
481                             count += 1
482
483
484             s_function[i] = s_temp
485             count_[i] = count
```

```
486
487         max_value = 0.0
488         max_index = 4
489         for index, value in enumerate(s_function):
490
491                 if(value > max_value):
492                         max_value = value
493                         max_index = index
494
495         if(max_index == 0):
496                 angle = 45
497                 result =  " : Extremely right slanted"
498         elif(max_index == 1):
499                 angle = 30
500                 result = " : Above average right slanted"
501         elif(max_index == 2):
502                 angle = 15
503                 result = " : Average right slanted"
504         elif(max_index == 3):
505                 angle = 5
506                 result = " : A little right slanted"
507         elif(max_index == 5):
508                 angle = -5
509                 result = " : A little left slanted"
510         elif(max_index == 6):
511                 angle = -15
512                 result = " : Average left slanted"
513         elif(max_index == 7):
514                 angle = -30
515                 result = " : Above average left slanted"
516         elif(max_index == 8):
517                 angle = -45
518                 result = " : Extremely left slanted"
519         elif(max_index == 4):
520                 if s_function[3] == 0.0:
521                         p = s_function[4]
522                         q = s_function[4]
523                 else:
524                         p = s_function[4] / s_function[3]
525                         q = s_function[4] / s_function[5]
526
527                 if((p <= 1.2 and q <= 1.2) or (p > 1.4 and q > 1.4)):
528                         angle = 0
529                         result = " : No slant"
530                 elif((p <= 1.2 and q-p > 0.4) or (q <= 1.2 and p-q > 0.4)):
531                         angle = 0
532                         result = " : No slant"
533                 else:
534                         max_index = 9
```

```
535                         angle = 180
536                         result =  " : Irregular slant behaviour"
537
538
539                 if angle == 0:
540                         print ("\n************************************************")
541                         print ("Slant determined to be straight.")
542                 else:
543                         print ("\n************************************************")
544                         print ("Slant determined to be irregular.")
545                 cv2.imshow("Check Image", img)
546                 cv2.waitKey(0)
547                 cv2.destroyAllWindows()
548                 type = input("Press enter if okay, else enter c to change: ")
549                 if type=='c':
550                         if angle == 0:
551                                 angle = 180
552                                 result =  " : Irregular Slant"
553                                 print ("Set as"+result)
554                                 print ("************************************************\n")
555                         else:
556                                 angle = 0
557                                 result = " : Straight/No Slant"
558                                 print ("Set as"+result)
559                                 print ("************************************************\n")
560                 else:
561                         print ("No Change!")
562                         print ("************************************************\n")
563
564         SLANT_ANGLE = angle
565         return
566
567
568 def barometer(image):
569
570         global PEN_PRESSURE
571
572
573         image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
574
575         h, w = image.shape[:]
576         inverted = image
577         for x in range(h):
578                 for y in range(w):
579                         inverted[x][y] = 255 - image[x][y]
580
581         filtered = bilateralFilter(inverted, 3)
582
583         ret, thresh = cv2.threshold(filtered, 100, 255, cv2.THRESH_TOZERO)
```

```
584
585         total_intensity = 0
586         pixel_count = 0
587         for x in range(h):
588                 for y in range(w):
589                         if(thresh[x][y] > 0):
590                                 total_intensity += thresh[x][y]
591                                 pixel_count += 1
592
593         average_intensity = float(total_intensity) / pixel_count
594         PEN_PRESSURE = average_intensity
595
596         return
597
598 def start(file_name):
599
600         global BASELINE_ANGLE
601         global TOP_MARGIN
602         global LETTER_SIZE
603         global LINE_SPACING
604         global WORD_SPACING
605         global PEN_PRESSURE
606         global SLANT_ANGLE
607
608
609         image = cv2.imread('images/'+file_name)
610
611
612         barometer(image)
613
614         straightened = straighten(image)
615
616         lineIndices = extractLines(straightened)
617
618         wordCoordinates = extractWords(straightened, lineIndices)
619
620         extractSlant(straightened, wordCoordinates)
621
622         BASELINE_ANGLE = round(BASELINE_ANGLE, 2)
623         TOP_MARGIN = round(TOP_MARGIN, 2)
624         LETTER_SIZE = round(LETTER_SIZE, 2)
625         LINE_SPACING = round(LINE_SPACING, 2)
626         WORD_SPACING = round(WORD_SPACING, 2)
627         PEN_PRESSURE = round(PEN_PRESSURE, 2)
628         SLANT_ANGLE = round(SLANT_ANGLE, 2)
629
630         return [BASELINE_ANGLE, TOP_MARGIN, LETTER_SIZE, LINE_SPACING,
631         WORD_SPACING, PEN_PRESSURE, SLANT_ANGLE]
632
```

Додаток Г

(Обов’язковий) Протокол перевірки

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Розробка експертної системи графологічного аналізу з використанням серверної архітектури

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ: кафедра Автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 89.6% Схожість 10.4%

Аналіз звіту подібності (відмітити потрібне):

■ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Роман МАСЛІЙ
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Денис ТУЛЬЧІЙ
(підпис) (прізвище, ініціали)

Керівник роботи _____ Олег БІСІКАЛО
(підпис) (прізвище, ініціали)