МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
в.о. завідувача кафедри
_____ Рязанцев О.І.
«____»_____ 20__ р.

МАГІСТЕРСЬКА РОБОТА

НА ТЕМУ:

Хмарна система цифрової бібліотеки технічної літератури

Освітній рівень"Магістр"
Спеціальність 123 "Комп'ютерна інженерія"

Науковий керівник роботи: _____ (підпис) М.Є.Щербакова (ініціали, прізвище)

Консультант з охорони праці: _____ (підпис) Я.О.Критська (ініціали, прізвище)

Студент: _____ (підпис) Копьонкін В.С. (ініціали, прізвище)

Група: КІ-19дм

Сєвєродонецьк 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітній рівень магістр
Напрям підготовки _____
(шифр і назва)
Спеціальність 123 "Комп'ютерна інженерія"
(шифр і назва)

ЗАТВЕРДЖУЮ:

Т.в.о. завідувача кафедри _____
_____ В.С.Кардашук
«_____» _____20____р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Копьонкіну Владиславу Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Хмарна система цифрової бібліотеки технічної літератури

керівник проекту (роботи) Щербакова Марина Євгенівна, к.т.н., доц.
(прізвище, м.'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від «5» 10 2020 р. № 140/15.15
2. Строк подання студентом роботи 10.01.2021
3. Вихідні дані до роботи Матеріали науково-дослідної практики, теоретичні відомості про методи розпізнавання зображень, теоритичні відомості про методи автоматичного перекладу тексту, методи автоматичного розпізнавання та перекладу тексту

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Огляд методологій побудови систем розпізнавання і технічного перекладу тексту, огляд і порівняння методів машинного розпізнавання і перекладу тексу, комп'ютерна реалізація розпізнавання і перекладу тексту, охорона праці та безпека в надзвичайних ситуаціях, висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці та безпека в надзвичайних ситуаціях	Критська Я.О. ст. викл. кафедри КНІ		

7. Дата видачі завдання 14.10.2020

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Розробка технічного завдання	02.09.2020-15.09.2020	
2	Огляд та аналіз літератури	16.09.2020-22.09.2020	
3	Розробка методу автоматичного перекладу	23.09.2020-25.09.2020	
4	Реалізація системи	26.09.2020-06.10.2020	
5	Аналіз результатів дослідження	07.10.2020-25.11.2020	
6	Розробка частини проекту "Охорона праці та безпеки в надзвичайних ситуаціях"	26.11.2020-1.12.2020	
7	Оформлення пояснювальної записки, автореферату та презентації	2.12.2020-09.01.2021	

Студент _____ Копьонкін В.С.
(підпис) (прізвище та ініціали)

Науковий керівник _____ Щербакова М.Є.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Копьонкін В.С. Хмарна система цифрової бібліотеки технічної літератури.

Метою роботи є аналіз проблеми переносу іноземних текстів з друкованого варіанту в електронний вигляд одночасно з перекладом з первинної на цільову мову.

Об’єктом дослідження виступає електронне зображення що містить іноземномовний текстовий матеріал. Це можуть бути фотографії, скани, скріншоти, або будь-які інші зображення.

При аналізі проблеми було виявлено, що проблема розпізнавання тексту з його подальшим перекладом є комплексною: вона складається з 4 глобальних етапів, таких як фільтрація вхідного зображення, оптичне розпізнавання символів, пост обробки результату, та автоматичного перекладу тексту.

У результаті роботи здійснена програмна реалізація багатомодульної системи автоматичного розпізнавання символів з подальшим перекладом отриманого тексту. Отриманий додаток найбільш ефективно працює з професійними текстами, такими як технічна література для інженерів, фінансистів та програмістів.

Ключові слова: розпізнавання зображень, автоматичний переклад тексту, фільтрація зображень, постобробка.

ABSTRACT

Kopenkin V.S. Cloud system of digital library of technical literature.

The aim of the work is to analyze the problem of transferring foreign texts from the printed version into electronic form simultaneously with the translation from the primary language to the target language.

The object of the study is an electronic image that contains foreign textual material. It can be photos, scan, screenshots, or any other image.

When analyzing the problem, it was revealed that the problem of text recognition with its further translation is complex: it consists of four global stages, such as filtering the input image, optical character recognition, post processing of the result, and automatic text translation.

As a result of the work, a software implementation of a multi-module automatic character recognition system with further translation of the received text was implemented. The resulting addition works most effectively with professional texts, such as technical literature for engineers, financiers and programmers.

Keywords: image recognition, automatic text translation, image filtration, post processing.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ 6
ВСТУП 7
1 ОГЛЯД МЕТОДОЛОГІЙ ПОБУДОВИ СИСТЕМ ДІДЖІТАЛІЗАЦІЇ БІБЛІОТЕКИ ТЕХНІЧНОЇ ЛІТЕРАТУРИ 8
1.1 Фільтрація вхідного зображення 8
1.2 Знаходження символів на зображенні 10
1.3 Оптичне розпізнавання символів 12
1.4 Пост обробка отриманого тексту 14
1.5 Автоматичний переклад тексту 15
1.6 Постановка задачі дослідження 16
2 ОГЛЯД І ПОРІВНЯННЯ МЕТОДІВ МАШИННОГО РОЗПІЗНАВАННЯ І ПЕРЕКЛАДУ ТЕКСУ 17
2.1 Методи попередньої обробки та сегментації зображень 17
2.2 Математична модель локалізації тексту 18
2.2.1 Інтегральне представлення зображення 19
2.2.2 Вейвлети Хаара 20
2.3 Аналіз методів розпізнавання образів 21
2.4 Інтенсіональні методи 23
2.4.1 Методи засновані на оцінках щільності розподілу значень ознак 24
2.4.2 Методи засновані на припущеннях про клас вирішальних функцій 24
2.4.3 Логічні методи 25
2.4.4 Лінгвістичні (структурні) методи 26
2.4.5 Екстенсіональні методи 26
2.4.6 Метод порівняння з прототипом 27
2.4.7 Метод k найближчих сусідів 28
2.4.8 Алгоритми обчислення оцінок 29
2.4.9 Колективи вирішальних правил 30
2.5 Аналіз методів пост обробки результату 35
2.6 Аналіз методів машинного перекладу тексту 36
2.6.1 Машинний переклад на основі правил 37
2.6.2 Машинний переклад на основі передачі 37
2.6.3 Міжмовний машинний переклад 38
2.6.4 Машинний переклад на основі словників 39

2.6.5 Статистичний машинний переклад 39
2.6.6 Машинний переклад на основі прикладів 40
2.6.7 Гібридні системи машинного перекладу 41
2.6.8 Нейронний машинний переклад 42
3 КОМП'ЮТЕРНА РЕАЛІЗАЦІЯ СИСТЕМИ ДІДЖІТАЛІЗАЦІЇ БІБЛІОТЕКИ ТЕХНІЧНОЇ ЛІТЕРАТУРИ 43
3.1 Вибір мови програмування 43
3.2 Spring фреймворк 43
3.3 Tesseract 44
3.4 Програмна реалізація автоматичного перекладу 48
3.5 Програмна реалізація багатопотчного програмування 49
3.6 Тестування розробленої моделі 50
3.6.1 Unit тестування 51
3.6.2 Інтеграційне тестування 53
3.6.3 Протестований функціонал 53
3.7 Розміщення в хмарі 54
4 ОХОРОНА ПРАЦІ 57
4.1 Аналіз потенційно небезпечних і шкідливих виробничих факторів , що впливають на персонал 57
4.2 Заходи щодо техніки безпеки 58
4.3 Заходи, що забезпечують виробничу санітарію і гігієну праці 61
4.4 Рекомендації по пожежній профілактиці 64
ВИСНОВКИ 68
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ 69
ДОДАТОК А. Фрагмент лістингу модуля «OCRTask.java» 73
ДОДАТОК Б. Фрагмент лістингу модуля «TextTranslator.java» 77
ДОДАТОК В. Фрагмент лістингу модуля «Benchmark.java» 80
ДОДАТОК Г. Тестові зображення 82
ДОДАТОК Д. Електронні плакати 83

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

ОРС – оптичне розпізнавання символів

МП – машинний переклад

СМП – статистичний машинний переклад

МРО – методи розпізнавання образів

АПТ – автоматичний переклад тексту

IOC – inversion of control (інверсія залежностей)

DI – dependency injection (вприскування залежностей)

DAO – data access layer (пласт роботи з даними)

DPI – dot per inch (точка на дюйм)

RGB – red green blue

ВСТУП

За останні роки зріс попит на автоматизацію процесів за допомогою комп'ютерної техніки. Першим етапом автоматизації людських робочих процесів виступала автоматизація фізичної праці, коли людина замінювалася на небезпечних, складних виробництвах, або на репетативній фізичній праці. Наступним кроком людство почало навчати машини виконувати інтелектуальну працю. Для вирішення більш складних проблем почали з'являтися нетривіальні інструменти: машинне навчання, нейронні мережі, штучний інтелект та інші. Станом на сьогоднішній день, без чисельна кількість додатків використовується для автоматизації інтелектуальної праці людини. Через великий попит зростає не тільки кількість таких продуктів, але й якість роботи кожного з них.

Гарним прикладом ефективного використання машин для вирішення проблем часозатратності на інтелектуальну роботу виступає оптичне розпізнавання символів (ОРС). Розпізнавання символів почалося з громіздких сканерів, що вміли розпізнавати лише навчені шрифти, але з ходом часу з'явилися механізми які дозволяють зчитувати текст навіть зі звичайного фото, зробленого на смартфон.

Іншим прикладом автоматизації інтелектуальної роботи виступає автоматичний переклад тексту. Перші автоматичні переклади дуже відрізнялися від тих, які робили професійні перекладачі. Через велику популярність цього напрямку, точність автоматичних перекладів значно виросла. Разом з тим, виросла і кількість додатків, що використовують автоматичний, або машинний переклад тексту. Сучасні автоматичні перекладачі мають велику точність перекладу, але погано підходять для перекладу професійних текстів. Специфікою таких текстів виступає те, що вони мають свої власні словники. Через те що в перекладі дуже рідко існує ситуація коли одне слово з вхідної мови логічно відповідає одному слову з вихідної, сучасні перекладачі здебільшого виберуть найбільш часто використовуване. Саме через це точність перекладу професійних текстів в сучасних автоматичних перекладачах залишається низькою. По вищевказаним причинам, створення методу автоматичного перекладу профільних текстів є актуальною темою даної магістерської роботи.

В рамках магістерської роботи роботи був проведений аналіз методологій вищесказаних процесів та створений додаток, що виконує розпізнавання тексту з зображення і подальший автоматичний переклад отриманого результуючого тексту.

1 ОГЛЯД МЕТОДОЛОГІЙ ПОБУДОВИ СИСТЕМ ДІДЖІТАЛІЗАЦІЇ БІБЛІОТЕКИ ТЕХНІЧНОЇ ЛІТЕРАТУРИ

Задача автоматичного зчитування з подальшим перекладом тексту являється дуже складною та багатоетапною проблемою. Проблему можна розбити на 2 глобальних етапи: розпізнавання тексту з вхідного зображення та автоматичний переклад отриманого результату. При аналізі першого етапу було виявлено, що проблема зчитування символів з зображення не є монолітною, а складається з 4 під-етапів. Для початку вхідне зображення має спеціально покращене для етапу розпізнавання. Наступним кроком з профільтрованого зображення можна зчитувати символи. Якщо пропустити етап фільтрування, програма може розпізнавати шуми на зображенні як символи і навпаки, прийняти символи за шуми. Після етапу знаходження символів наступає етап оптичного розпізнавання символів. Якщо не зробити пост обробку знайденого тексту, якість обробки в умовах зчитування слів буде мінімальною, а автоматичний переклад не матиме сенсу. Нарешті після етапу пост обробки, можна приступити до фінального етапу – автоматичного перекладу тексту. Тож, після аналізу проблеми були знайдені основні етапи вирішення задачі діджіталізації бібліотеки технічної літератури:

- фільтрація зображення;
- знаходження символів на зображенні;
- оптичне розпізнавання символів;
- пост обробка зчитаного тексту;
- автоматичний переклад отриманого тексту.

Розглянемо ці етапи більш детально.

1.1 Фільтрація вхідного зображення

Якість вхідного зображення являється краєугольним каменем точності розпізнавання тексту на зображенні. Чим вище якість початкового зображення, тим вище точність розпізнавання. Сама якість зображення залежить від декількох факторів:

- чіткість меж зображення;
- контраст зображення;
- якість вирівнювання символів;

– якомога менше шуму.

Зображення високої якості можна побачити на рисунку 1.1

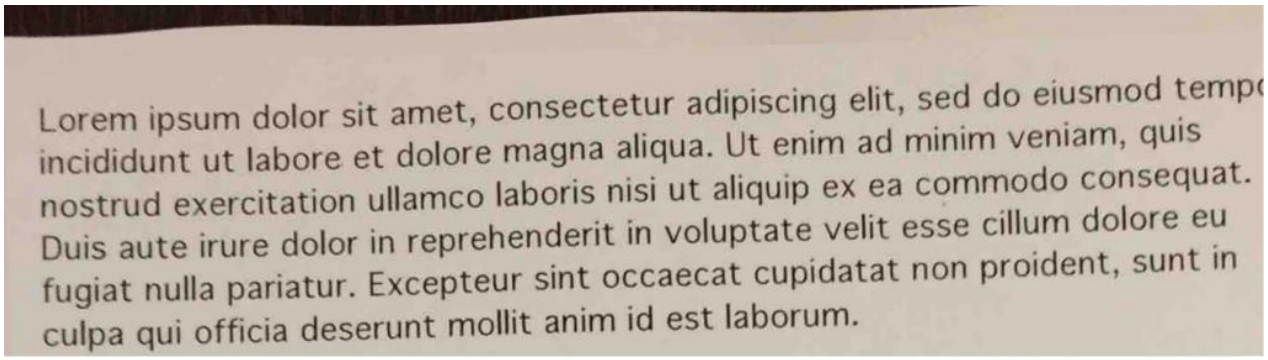


Рисунок 1.1 – Вхідне зображення високої якості до фільтрацій

Більшість движків поставляються зі вбудованими фільтрами обробки зображень OCR для автоматичного поліпшення якості текстового зображення. Проблема з цими вбудованими фільтрами полягає в тому, що ви не зможете настроїти їх відповідно до вашого варіанту використання. Розуміння кожного кроку попередньої обробки і індивідуальне налаштування параметрів попередньої обробки є ключовим чинником підвищення продуктивності розпізнавання.

Масштабування до потрібного розміру, який зазвичай складає не менше 300 точок на дюйм (точок на дюйм). Підтримка DPI нижче 200 дасть неясні і незрозумілі результати, тоді як підтримка DPI вище 600 приведе до зайвого збільшення розміру вихідного файлу без поліпшення якості файлу. Таким чином, DPI 300 краще всього підходить для цієї мети.

Збільшення контрасту. Низький контраст може привести до поганого розпізнавання. Збільшення контрасту і щільності перед виконанням процесу розпізнавання вирішує цю проблему. Це можна зробити в самому програмному забезпеченні для сканування або у будь-кому іншому програмному забезпеченні для обробки зображень. Збільшення контрасту між текстом (або зображенням) і його фоном дає більше ясності в кінцевому результаті.

Бінаризація зображення. Цей крок перетворить багатоколірне зображення (RGB) в чорно-біле зображення. Існує декілька алгоритмів для перетворення кольорового зображення в монохромне зображення, починаючи від простого визначення порогу до складнішого зонального аналізу. Більшість механізмів OCR внутрішньо працюють з монохромними зображеннями і виконують перетворення колір -> монохромний як один з перших кроків. Якщо контролювати цей етап попередньої обробки і робити все правильно, великі шанси отримати більш високу точність розпізнавання. Ще одна перевага перетворення в двійкову форму ваших зображень перед їх відправкою на

механізм розпізнавання – це зменшений розмір ваших зображень. Зображення після бінаризації (рисунок 1.2).

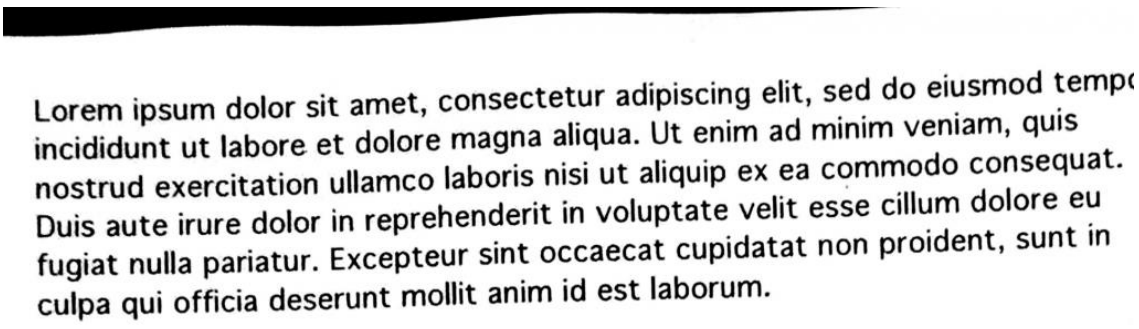


Рисунок 1.2 – Зображення після бінаризації

Видалення шумів та артефактів сканування. Шум може різко понизити загальну якість процесу розпізнавання. Він може бути присутнім на задньому плані або на передньому плані і може бути результатом поганого сканування або низької якості оригіналу даних.

Обертання зображення (рисунок 1.3). Це означає, що треба надати зображенню правильний формат і правильну форму. Текст повинен відображатися горизонтально. Якщо зображення перекошене у будь-яку сторону, нахиляється, обертається за годинниковою стрілкою або проти годинникової стрілки.

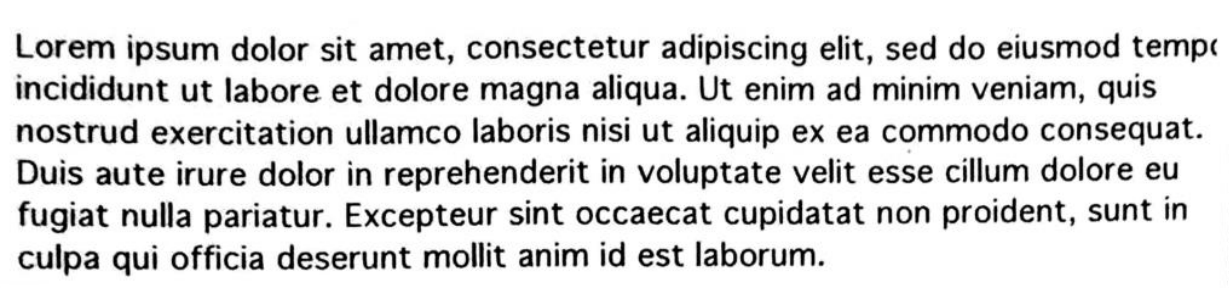


Рисунок 1.3 – Зображення після обертання

1.2 Знаходження символів на зображенні

Задача знаходження символів на зображенні – необхідний етап перед розпізнаванням. Найкращим інструментів для знаходження символів на зображенні виступає комп’ютерний зір. Фінальний етап процесу знаходження символів на зображенні зображений на рисунку 1.4.

Рисунок 1.4 – Знаходження символів на зображенні

Комп’ютерний зір – це міждисциплінарна наукова область, яка займається отриманням високого рівня розуміння цифрових зображень або відео комп’ютерами. З точки зору інженерії, він прагне автоматизувати завдання, які може виконувати зорова система людини.

Завдання комп’ютерного зору включають способи отримання, обробки, аналізу і розуміння цифрових зображень і витягання багатовимірних даних з реального світу для отримання числової або символічної інформації, наприклад, у формах рішень. Розуміння в цьому контексті означає перетворення зорових образів (введення сітківки) в описи світу, які можуть взаємодіяти з іншими розумовими процесами і викликати відповідні дії.

Таке розуміння зображення можна розглядати як відділення символічної інформації від даних зображення з використанням моделей, побудованих за допомогою геометрії, фізики, статистики і теорії навчання.

Наукова дисципліна комп’ютерного зору пов’язана з теорією штучних систем, які витягають інформацію із зображень. Ці зображення можуть приймати різні форми, такі як відеопослідовності, види з декількох камер або багатовимірні дані з медичного сканера. Технологічна дисципліна комп’ютерного зору прагне застосовувати свої теорії і моделі для побудови систем комп’ютерного зору.

Класична проблема в комп’ютерному зорі, обробці зображень і машинному зорі полягає в тому, щоб визначити, чи містять ці зображення який-небудь конкретний об’єкт, функцію або дію. Описуються різні варіанти проблеми розпізнавання:

Розпізнавання об’єктів (що також називається класифікацією об’єктів) – один або декілька заздалегідь певних або вивчених об’єктів або класів об’єктів можуть бути розпізнані, як правило, разом з їх 2d-позиціями на.

Ідентифікація – індивідуальний екземпляр об’єкту розпізнається. Приклади включають ідентифікацію особи або відбитку пальця конкретної людини, ідентифікацію рукописних цифр або ідентифікацію конкретного транспортного засобу.

Виявлення – ці зображення скануються для певної умови. Приклади включають виявлення можливих аномальних клітин або тканин на медичних зображеннях або виявлення транспортного засобу в автоматичній системі платних доріг.

Виявлення, ґрунтоване на відносно простих і швидких обчисленнях, іноді використовується для знаходження менших областей цікавих даних зображення, які можуть бути додатково проаналізовані за допомогою складніших обчислювальних методів для отримання правильної інтерпретації.

1.3 Оптичне розпізнавання символів

Уже зараз розпізнавання образів щільно увійшло в повсякденне. У медицині розпізнавання образів допомагає лікарям ставити більш точні діагнози, на заводах воно використовується для прогнозу браків в партіях товарів. Системи біометричної ідентифікації особистості як свого алгоритмічного ядра так само засновані на результатах цієї дисципліни. Подальший розвиток штучного інтелекту, зокрема проектування комп’ютерів п’ятого покоління, здатних до більш безпосереднього спілкування з людиною на природних для людей мовах і за допомогою мови, немислимі без розпізнавання. Тут рукою подати і до робототехніки, штучних систем управління, що містять в якості життєво важливих підсистем системи розпізнавання. Приклад розпізнаних цифр з фотографії зображені на рисунку 1.5.

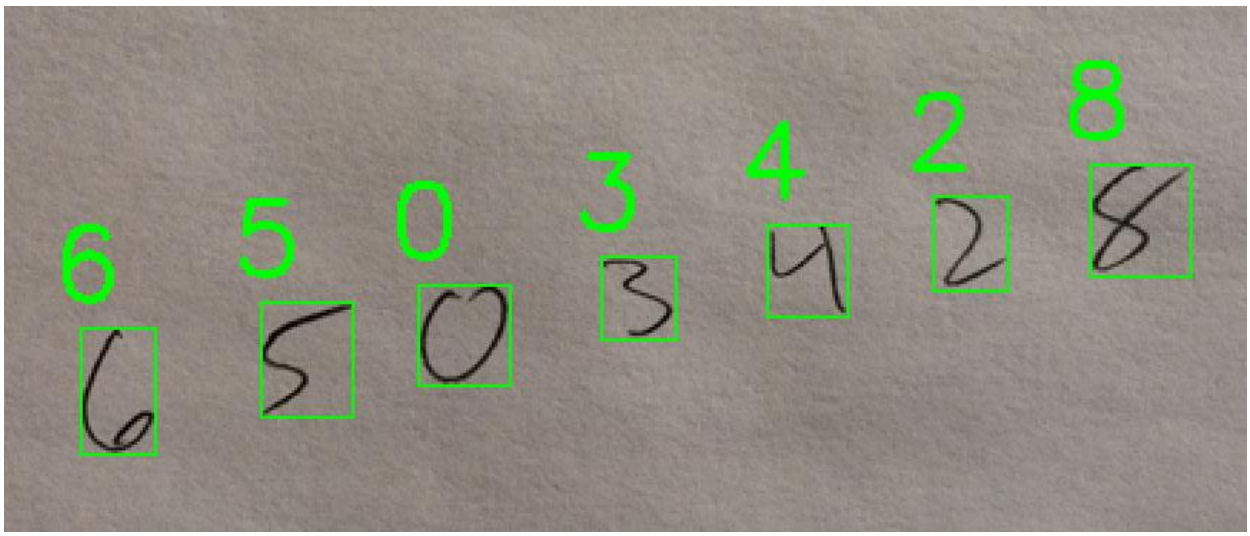


Рисунок 1.5 – Оптичне розпізнавання чисел на фотографії

Розпізнавання образів – це автоматичне розпізнавання закономірностей в даних. Розпізнавання образів тісно пов’язане з штучним інтелектом і машинним навчанням разом з такими застосуваннями, як збір даних і виявлення знань у базах даних, і часто використовується взаємозамінний з цими термінами.

Проте вони відрізняються: машинне навчання – це один з підходів до розпізнавання моделей, тоді як інші підходи включають ручні (не вивчені) правила або евристику; І розпізнавання образів є одним з підходів до штучного інтелекту, тоді як інші підходи включають символічний штучний інтелект.

Область розпізнавання образів пов’язана з автоматичним виявленням закономірностей в даних за допомогою використання комп’ютерних алгоритмів і з використанням цих закономірностей для здійснення таких дій, як класифікація даних за різними категоріями.

Є ряд істотних проблем, пов’язаних з друкованих символів і розпізнаванням рукописних. Найбільш важливі з них наступні:

– різноманітність форм накреслення символів;

– спотворення зображень символів;

– варіації розмірів і масштабу символів.

Кожен окремий символ може бути написаний різними стандартними шрифтами, наприклад (Times, Gothic, Elite, Courier, Orator), а також – безліччю нестандартних шрифтів, використовуваних в різних предметних областях. Спотворення цифрових зображень текстових символів можуть бути викликані:

– шумами друку, зокрема, непродруковка (розривами злитих символів), «злипання» сусідніх символів, плямами і помилковими точками на тлі поблизу символів і т.д.;

– зміщенням символів або частин символів щодо їх очікуваного положення в рядку;

– зміною нахилу символів;

– спотворенням форми символу за рахунок оцифрування зображення з «грубим» дискретом;

– ефектами освітлення (тіні, відблиски і т.д.).

1.4 Пост обробка отриманого тексту

Точність оптичного розпізнавання символів не буває сто відсотковою. До того ж, різні символи можуть мати подібними контурами. Наприклад, «U» і «V», «S» і «5», «Z» і «2», «G» і «6». Дуже часто при розпізнаванні символів в ситуації колізії система вибирає неправильний символ. Окрім того, навіть якщо текст розпізнаний вірно, ніхто не дає гарантії граматичної правильності вхідного тексту. Для вирішення обох цих проблем необхідно використовувати системи перевірки правопису, або як вони більш відомі – spellcheckers.

Базова перевірка орфографії виконує наступні процеси:

Система порівнює кожне слово з відомим списком правильно написаних слів (тобто словником). Це може містити тільки список слів або може містити додаткову інформацію, таку як точки перенесення або лексичні і граматичні атрибути.

Додатковим кроком є залежний від мови алгоритм для обробки морфології. Навіть для злегка зміненої мови, такого як англійський, для перевірки орфографії необхідно враховувати різні форми одного і того ж слова, такі як множину, словесні форми, скорочення і присвійні вирази. Для багатьох інших мов, таких як ті, які мають аглютинацію і складніші відміни і відмінювання, ця частина процесу є складнішою.

Альтернативний тип перевірки орфографії використовує виключно статистичну інформацію, таку як n-грами, для розпізнавання помилок замість правильно написаних слів. Цей підхід зазвичай вимагає великих зусиль для отримання достатньої статистичної інформації. Основні переваги включають необхідність меншого об’єму пам’яті під час виконання і можливість виправляти помилки в словах, які не включені в словник.

1.5 Автоматичний переклад тексту

Машинний переклад, що іноді називається абревіатурою МП – підмножина обчислювальної лінгвістики, досліджуване використання програмного забезпечення для перекладу тексту з однієї мови на іншу.

На базовому рівні МП виконує просту підстановку слів в одній мові для слів в іншому, але та одна зазвичай не може зробити хороший переклад тексту, оскільки потрібне розпізнавання цілих фраз і їх найближчих аналогів в цільовій мові. Вирішення цієї проблеми з корпусними статистичними і нейронними методами – це швидко зростаюча область, яка призводить до кращих перекладів, обробки відмінностей в лінгвістичній типології, перекладу ідіом і ізоляції аномалій.

Поточне програмне забезпечення машинного перекладу часто дозволяє настроювати по домену або професії (наприклад, звіти про погоду), покращуючи вихідні дані за рахунок обмеження області допустимих заміщень. Цей метод особливо ефективний в областях, де використовується формальна мова. Звідси витікає, що машинний переклад урядових і юридичних документів легше дає корисний вихід, чим розмова або менш стандартизований текст.

Поліпшена якість виведення також може бути досягнута втручанням людини: наприклад, деякі системи здатні точніше переводити, якщо користувач однозначно визначив, які слова в тексті є правильними іменами. За допомогою цих методів МТ довела свою корисність в якості інструменту для надання допомоги перекладачам з числа людей і в дуже обмеженому числі випадків може навіть давати результати, які можуть бути використані як є.

Про прогрес і потенціал машинного перекладу багато говорили упродовж усієї його історії. Починаючи з 1950-х років, ряд учених поставили під сумнів можливість досягнення повністю автоматичного машинного перекладу високої якості, передусім Йехошуа Бар-Хиллель. Деякі критики стверджують, що існують принципові перешкоди для автоматизації процесу перекладу.

Процес перекладу людини може бути описаний як:

– декодування значення початкового тексту;
– перекодування цього значення на цільовій мові.

За цією нібито простою процедурою стоїть складна когнітивна операція. Для декодування значення початкового тексту в усій його повноті перекладач повинен інтерпретувати і аналізувати усі особливості тексту, процес, що вимагає глибокого знання

граматики, семантики, синтаксису, ідіоми і т. д. початкової мови, а також культури його носіїв. Перекладачеві потрібні такі ж поглиблені знання, щоб перекодувати значення в цільову мову.

У своєму найзагальнішому застосуванні це виходить за рамки сучасної технології. Попри те, що він працює набагато швидше, ніяка автоматизована програма перекладу або процедура, без участі людини, не може робити продукцію, навіть близьку до якості, яка може робити людина-перекладач. Проте вона може забезпечити загальне, хоча і недосконале наближення початкового тексту, отримуючи його «gist» (процес, званий «gisting»). Цього вистачає для багатьох цілей, включаючи найкраще використання кінцевого і дорогого часу перекладача людини, зарезервованої для тих випадків, коли потрібна повна точність.

1.6 Постановка задачі дослідження

Після проведення аналізу предметної області було виявлено що задача комплексна та мусить проводитися в декілька етапів. Система, що буде реалізовувати автоматичне розпізнавання тексту з подальшим автоматичним перекладом має виковувати усі етапи розпізнавання: фільтрацію вхідного зображення, знаходження символів на зображенні, оптичне розпізнавання знайдених символів, пост-обробку отриманого результату, та автоматичний переклад тексту. По кожному з вищевказаних етапів треба провести аналіз, виявити найкращий спосіб вирішення поставленої проблеми та програмно реалізувати рішення усієї системи загалом в вигляді готового програмного додатка.

Основним джерелом вхідних зображень для розпізнавання будуть фотографії, скріншоти або скани сторінок підручників та наукових статей. Ці зображення характеризуються високою щільністю символів на сторінці, що значно полегшує процес знаходження символів на зображенні. Окрім того, фотографії або скріншоти книжних сторінок характеризуються високою якістю, що полегшує етап первинної фільтрації зображення, а кінцева якість отриманого тексту буде максимальною. Іншою особливістю підручників або наукових статей виступає низька кількість граматичних помилок. Через це пост-обробка знайденого тексту буде використовуватися в більшій мірі для вирішення проблеми оптичної колізії букв та чисел.

Через те що планована галузь використання складається з професійної літератури, автоматичний переклад тексту має бути гнучким і настроюватися на різні тематики.

2 ОГЛЯД І ПОРІВНЯННЯ МЕТОДІВ МАШИННОГО РОЗПІЗНАВАННЯ І ПЕРЕКЛАДУ ТЕКСУ

2.1 Методи попередньої обробки та сегментації зображень

Предобробка є важливим етапом в процесі розпізнавання символів і дозволяє виробляти згладжування, нормалізацію, сегментацію і апроксимацію відрізків ліній.

Під згладжуванням в даному випадку розуміється велика група процедур обробки зображень Зокрема, широко використовуються морфологічні оператори заповнення і стоншення. Заповнення усуває невеликі розриви і прогалини. Потоншення є процес зменшення товщини лінії, в якій на кожному кроці області розміром в декілька пікселів ставиться у відповідність тільки один піксель «витонченої лінії».

Геометрична нормалізація зображень документів має на увазі використання алгоритмів, що усувають нахили і перекоси окремих символів, слів чи рядків, а також включає в себе процедури, які здійснюють нормалізацію символів по висоті і ширині після відповідної їх обробки.

Процедури сегментації здійснюють розбиття зображення документа на окремі області. Як правило, перш за все необхідно відокремити друкований текст від графіки і рукописних позначок. Далі більшість алгоритмів оптичного розпізнавання поділяють текст на символи і розпізнають їх окремо. Це просте рішення дійсно найбільш ефективно, якщо тільки символи тексту не перекривають один одного. Злиття символів може бути викликано типом шрифту, яким був набраний текст, поганим дозволом друкувального пристрою або високим рівнем яскравості, обраним для відновлення розірваних символів.

Додаткове розбиття текстових областей і рядків на слова доцільно в тому випадку, якщо слово є заможним об’єктом, відповідно до якого виконується розпізнавання тексту. Подібний підхід, при якому одиницею розпізнавання є не окремий символ, а ціле слово, складно реалізовувати через великої кількості елементів, що підлягають запам’ятовуванню і розпізнаванню, але він може бути корисний і вельми ефективний в конкретних приватних випадках, коли набір слів в кодовому словнику істотно обмежений за умовою задачі.

Під апроксимацією відрізків ліній аппроксімацією відрізків ліній розуміють складання графа опису символу у вигляді набору вершин і прямих ребер, які безпосередньо апроксимують ланцюжка пікселів вихідного зображення. Дана апроксимація здійснюється для зменшення обсягу даних і може використовуватися при

розпізнаванні, заснованому на виділення ознак, що описують геометрію і топологію зображення.

2.2 Математична модель локалізації тексту

Найбільш відомим і популярним алгоритмом в області комп'ютерного зору виступає метод Віоли-Джонса. Даний метод був розроблений у 2001 році Майклом Джонсом та Полом Віолой. Дуже швидко він став широко використовуватися через свою швидкість та ефективність в питаннях локалізації тексту.

Перед тим як використовувати цей метод, необхідно навчати алгоритм.

Швидкість навчання не має жодного значення на кінцевий результат, однак дуже важливим фактором виступає швидкість розпізнавання об'єктів. До переваг методу можна віднести:

– дуже велика точність роботи алгоритму призводить до малої кількості помилкових спрацювань алгоритму;

– відсутність обмежень на кількість елементів для розпізнавання на зображенні;

– відсутність обмежень на навчання класифікатора для пошуку об'єктів. Класифікатор навчається знаходити будь який об'єкт;

– при використанні простих класифікаторів в процесі роботи алгоритму одразу показує високу швидкість. Саме ця перевага дозволяє використовувати метод в аналізі відеоматеріалів на наявність певних об'єктів;

– через те що використовуються інтегральне уявлення зображення, перевірка будь якого признаку на певні має часову складність;

– велика кількість реалізацій в різноманітних бібліотеках на різних мовах програмування. Значна кількість з них open-source.

Недоліки методу:

– навчання класифікатору займає велику кількість часу через те що потребується зробити аналіз велику кількість зображень;

– точність розпізнавання залежить від правильності вибору навчаючої вибірки;

– низька точність розпізнавання за умови знаходження необхідного елементу за умови не горизонтального розміщення елементу;

– точність залежить від масштабу зображення.

Основною ідеєю алгоритму Віоли-Джонса для розпізнавання об’єктів є виділення локальних особливостей зображення і подальшого навчання алгоритму на них. Пошук на зображенні виконується за принципом скануючого вікна-зображення сканується вікном пошуку, і на кожному положенні вікна застосовується класифікатор.

Спрощена схема алгоритму виглядає наступним чином. Перед початком розпізнавання алгоритм навчання на навчає класифікатор, що складається з значень певних ознак. Далі алгоритм розпізнавання шукає об’єкти на різних масштабах зображення, грунтуючись на класифікаторі. На виході алгоритму видається безліч знайдених об’єктів на різних масштабах.

2.2.1 Інтегральне представлення зображення

Щоб ефективно визначити наявність або відсутність безлічі вейвлетів Хаара на величезній кількості областей зображення різного розміру, Віола і Джонс застосували метод інтегрального представлення зображення. Даний метод дозволяє істотно прискорити цю операцію. При цьому час обчислення суми яскравості прямокутника не залежить від його площі.

Якщо ми маємо відображення I з множини матриць $R^{m\times n}$ в $R^{(m+1)\times(n+1)}$ таке що $\forall X \in R^{m\times n}\ \exists Y \in R^{(m+1)\times(n+1)}, Y = I(X)$:

$$Y(x,y)=\begin{cases}\sum\limits_{i=0,y=0}^{x-1,y-1} X(i,y), \textit{при } x>0 \textit{ та } y>0;\\ 0, \textit{при } x=0 \textit{ або } y=0.\end{cases}$$

Розрахунок матриці $Y = I(X)$ займає лінійний час, пропорціональний кількості пік селів в зображенні:

$$Y(x,y)=X(x-1,y-1)-Y(x-1)(y-1)+Y(x,y-1)+Y(x-1,y).$$

Час її обчислення можна трохи зменшити, якщо спочатку проінтегрувати стовпці матриці X, а потім рядка.

За інтегральною матриці всього за 4 звернення можна обчислити суму інтенсивностей пікселів всередині довільного прямокутника $ABCD$ вихідного зображення X. Сумарна інтенсивність в прямокутнику $ABCD$ обчислюється за формулою:

$$S_X(ABCD) = Y(A_x, A_y) + Y(C_x, C_y) - Y(B_x, B_y) - Y(D_x, D_y).$$

Таким чином, значення будь-якої ознаки можна обчислити за допомогою всього декількох операцій, кількість яких залежить тільки від кількості прямокутників в ознаці і не залежить від площі досліджуваної області.

2.2.2 Вейвлети Хаара

Історично склалося так, що алгоритми, що працюють тільки з інтенсивністю зображення (наприклад значення RGB в кожному пікселі), мають велику обчислювальну складність. Віола і Джонс адаптували ідею використання вейвлетів Хаару та розробили те, що було названо ознаками Хаару. Головною причиною знаходження вейвлетів Хаара в алгоритмі виступає спроба відійти від піксельного уявлення зі збереженням швидкості роботи

Ознаки Хаару– ознаки які використовуються для розпізнавання образів в цифрової графіці. Ознаки Хаара є прямокутні області, які складені з декількох сусідніх прямокутних областей, позначених світла чи темна. Під ознакою будемо розуміти тривимірний вектор виду: j = {маска, положення, розмір}. У цих прямокутників є вага котра задається кожному з прямокутників. Ця вага обчислюється сумою значень яскравості пікселей на зображенні котрі закриває цій прямокутник. Завдяки цьому можливо швидко обчислювати суму яскравостей для прямокутних областей, але не для фігур довільної орієнтації. Ознаки Хаара можна побачити на рисунку 2.1.

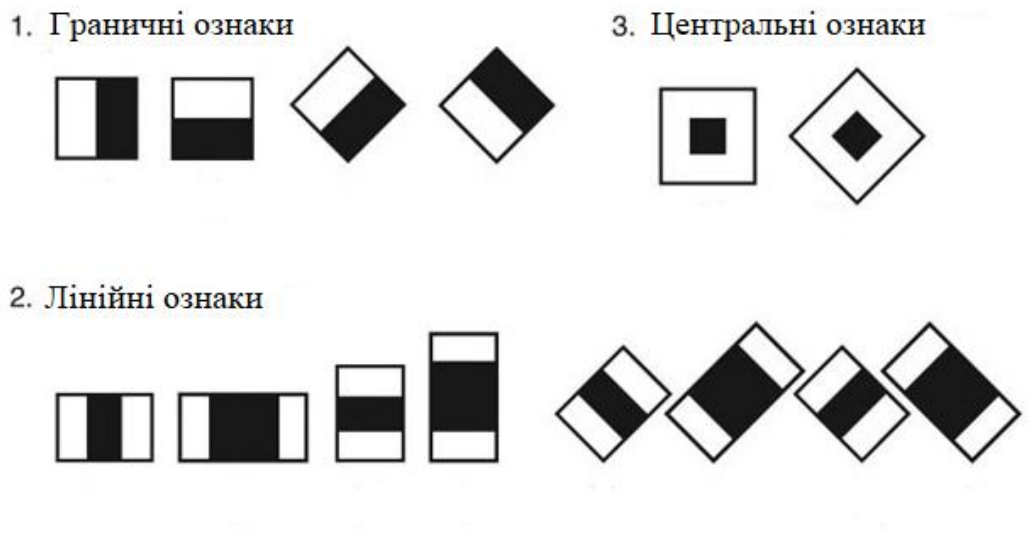


Рисунок 2.1 – Ознаки Хаара

Велика швидкість знайдення суми яскравості пікселей прямокутного регіону досягається за допомогою оптимізації а саме за допомогою інтегрального перетворення зображення.

Алгоритм використовує базу ознак для виявлення об’єктів. Базу можливо згенерувати з усіх можливих комбінацій шляхом відсіювання «слабких» класифікаторів, які дають помилку другого роду, тобто помилково не знаходять об’єкт на зображенні.

Ключовою особливістю ознак Хаара є найбільша, в порівнянні з іншими ознаками, швидкість. При використанні інтегрального представлення зображення, ознаки Хаара можуть обчислюватися за постійний час (приблизно 60 процесорних інструкцій на ознака з двох областей), що дозволяє використовувати класифікатор в режимі реального часу.

2.3 Аналіз методів розпізнавання образів

В існуючих системах OCR використовуються різноманітні алгоритми класифікації, тобто віднесення ознак до різних класів. Вони істотно розрізняються в залежності від прийнятих наборів ознак і застосовуваної по відношенню до них стратегії класифікації.

Для ознакової класифікації символів необхідно, в першу чергу, сформувати набір еталонних векторів ознак по кожному з розпізнаються символів. Для цього на стадії навчання оператор або розробник вводить в систему OCR велику кількість зразків накреслення символів, супроводжуваних зазначенням значення символу. Для кожного

зразка система виділяє ознаки і зберігає їх у вигляді відповідного вектора ознак вектора ознак. Набір векторів ознак, що описують символ, називається класом, або кластером.

В процесі експлуатації системи OCR може з'явитися необхідність розширити сформовану раніше базу знань. У зв'язку з цим деякі системи мають можливість додаткового навчання в реальному режимі часу.

При побудові алгоритмів розпізнавання класи еквівалентності можуть задаватися дослідником, який користується власними змістовними уявленнями або використовує зовнішню додаткову інформацію про подібність і відмінність об'єктів у контексті розв'язуваної задачі. Тоді говорять про «розпізнаванні з учителем». В іншому випадку, тобто коли автоматизована система вирішує завдання класифікації без залучення зовнішньої навчальної інформації, говорять про автоматичної класифікації чи «розпізнаванні без вчителя». Більшість алгоритмів розпізнавання образів вимагає залучення досить значних обчислювальних потужностей, які можуть бути забезпечені тільки високопродуктивної комп'ютерної технікою.

Основна типологія методів розпізнавання образів:

– методи, засновані на принципі поділу;
– статистичні методи;
– методи, побудовані на основі «потенційних функцій»;
– методи обчислення оцінок (голосування);
– методи, засновані на обчисленні висловлювань, зокрема на апараті алгебри логіки.

В основі даної класифікації лежить відмінність у формальних методах розпізнавання образів і тому опущено розгляд евристичного підходу до розпізнавання, отримав повне і адекватне розвиток в експертних системах. Евристичний підхід заснований на важко формалізованих знаннях та інтуїції дослідника. При цьому дослідник сам визначає, яку інформацію і яким чином система повинна використовувати для досягнення необхідного ефекту розпізнавання.

Подібна типологія методів розпізнавання з тій чи іншій ступенем деталізації зустрічається в багатьох роботах по розпізнаванню. У той же час відомі типології не враховують одну дуже суттєву характеристику, яка відображає специфіку способу подання знань про предметної області за допомогою якого-небудь формального алгоритму розпізнавання образів.

Виділяють два основних способи подання знань:

– інтенсіональне, у вигляді схеми зв'язків між атрибутами;
– екстенсіональності, за допомогою конкретних фактів (об'єкти, або приклади).

Інтенсіональне уявлення фіксують закономірності і зв’язки, якими пояснюється структура даних. Стосовно діагностичним завданням, така фіксація полягає у визначенні операцій над атрибутами (ознаками) об’єктів, що призводять до необхідного діагностичного результату. Інтенсіональні уявлення реалізуються за допомогою операцій над значеннями атрибутів і не припускають твори операцій над конкретними інформаційними фактами (об’єктами).

У свою чергу, екстенсіональні представлення знань пов’язані з описом і фіксацією конкретних об’єктів з предметної області та реалізуються в операціях, елементами яких служать об’єкти як цілісні системи.

Можна провести аналогію між інтенсіональні і екстенсіональності уявленнями знань і механізмами, що лежать в основі діяльності лівого і правого півкуль головного мозку людини. Якщо для правої півкулі характерна цілісна прототипна репрезентація навколишнього світу, то ліва півкуля оперує закономірностями, що відображають зв’язку атрибутів цього світу.

Описані вище два фундаментальних способу представлення знань дозволяють запропонувати наступну класифікацію методів розпізнавання образів:

– інтенсіональні методи, засновані на операціях з ознаками;
– екстенсіональності методи, засновані на операціях з об’єктами.

Таким чином, в основу класифікації методів розпізнавання, покладені фундаментальні закономірності, що лежать в основі людського способу пізнання взагалі, що ставить її в особливе (привілейоване) становище порівняно з іншими класифікаціями, які на цьому тлі виглядають більш легковажними і штучними.

2.4 Інтенсіональні методи

Відмінною особливістю інтенсіональних методів є те, що в якості елементів операцій при побудові і застосуванні алгоритмів розпізнавання образів вони використовують різні характеристики ознак і їх зв’язків. Такими елементами можуть бути окремі значення або інтервали значень ознак, середні величини і дисперсії, матриці зв’язків ознак і т.д., З яких виробляються дії, що виражаються в аналітичній або конструктивній формі. При цьому об’єкти в даних методах не розглядаються як цілісні інформаційні одиниці, а виступають в ролі індикаторів для оцінки взаємодії і поведінки своїх атрибутів.

Група інтенсіональних методів розпізнавання образів обширна, і її розподіл на підкласи носить в певній мірі умовний характер.

2.4.1 Методи засновані на оцінках щільності розподілу значень ознак

Ці методи розпізнавання образів запозичені з класичної теорії статистичних рішень, в якій об’єкти дослідження розглядаються як реалізації багатовимірної випадкової величини, розподіленої в просторі ознак з якого-небудь закону. Вони базуються на Байєсівської схемою прийняття рішень, що апелює до апріорним можливостям приналежності об’єктів до тієї чи іншої розпізнається класу і умовним щільності розподілу значень вектора ознак. Дані методи зводяться до визначення відношення правдоподібності в різних областях багатовимірного простору ознак.

Група методів, заснованих на оцінці щільності розподілу значень ознак, має пряме відношення до методів дискримінантного аналізу. Байєсівський підхід до прийняття рішень і відноситься до найбільш розробленим в сучасній статистиці так званим параметричних методів, для яких вважається відомим аналітичний вираз закону розподілу (в даному випадку нормальний закон) і потрібно оцінити лише невелика кількість параметрів (вектори середніх значень і коваріаційні матриці).

Основними труднощами застосування зазначених методів вважаються необхідність запам’ятовування всієї навчальної вибірки для обчислення оцінок локальних густин розподілу ймовірностей і висока чутливість до непредставницьким навчальної вибірки.

2.4.2 Методи засновані на припущеннях про клас вирішальних функцій

У даній групі методів вважається відомим загальний вигляд вирішальної функції і заданий функціонал її якості. На підставі цього функціоналу по навчальної послідовності знаходять найкраще наближення вирішальної функції. Найпоширенішими є уявлення вирішальних функцій у вигляді лінійних і узагальнених нелінійних поліномів. Функціонал якості вирішального правила зазвичай пов’язують з помилкою класифікації. Основною перевагою методів, заснованих на припущеннях про клас вирішальних функцій, є ясність математичної постановки задачі розпізнавання, як завдання пошуку екстремуму. Різноманіття методів цієї групи пояснюється широким спектром

використовуваних функціоналів якості вирішального правила і алгоритмів пошуку екстремуму. Узагальненням розглянутих алгоритмів, до яких відносяться, зокрема, алгоритм Ньютона, алгоритми перцептроном типу та ін., Є метод стохастичною апроксимації.

Можливості градієнтних алгоритмів пошуку екстремуму, особливо в групі лінійних вирішальних правил, досить добре вивчені. Збіжність цих алгоритмів доведена тільки для випадку, коли розпізнаються класи об'єктів відображаються в просторі ознак компактними геометричними структурами.

Досить висока якість вирішального правила може бути досягнуто за допомогою алгоритмів, які не мають строгого математичного докази збіжності рішення до глобального екстремуму. До таких алгоритмам відноситься велика група процедур евристичного програмування, що представляють напрям еволюційного моделювання. Еволюційне моделювання є біонічним методом, запозиченим у природи. Воно засноване на використанні відомих механізмів еволюції з метою заміни процесу змістовного моделювання складного об'єкта феноменологическим моделюванням його еволюції. Відомим представником еволюційного моделювання в розпізнаванні образів є метод групового урахування аргументів (МГУА). В основу МГУА покладено принцип самоорганізації, і алгоритми МГУА відтворюють схему масової селекції.

Однак досягнення практичних цілей в даному випадку не супроводжує витяг нових знань про природу розпізнаваних об'єктів. Можливість вилучення цих знань, зокрема знань про механізми взаємодії атрибутів (ознак), тут принципово обмежена заданою структурою такої взаємодії, зафіксованої в обраній формі вирішальних функцій.

2.4.3 Логічні методи

Логічні методи розпізнавання образів базуються на апараті алгебри логіки і дозволяють оперувати інформацією, укладеної не тільки в окремих ознаках, але і в поєднаннях значень ознак. У цих методах значення якої-небудь ознаки розглядаються як елементарні події.

У найзагальнішому вигляді логічні методи можна охарактеризувати як різновид пошуку по навчальній вибірці логічних закономірностей і формування певної системи логічних вирішальних правил (наприклад, у вигляді кон'юнкція елементарних подій), кожне з яких має власну вагу. Група логічних методів різноманітна і включає методи

різної складності і глибини аналізу. Для дихотомічних (булевих) ознак популярними є так звані деревоподібні класифікатори, метод тупикових тестів, алгоритм «Кора» і ін.

Алгоритм «Кора», як і інші логічні методи розпізнавання образів, є досить трудомістким в обчислювальному відношенні, оскільки при відборі кон'юнкція необхідний повний перебір. Тому при застосуванні логічних методів пред'являються високі вимоги до ефективної організації обчислювального процесу, і ці методи добре працюють при порівняно невеликих розмірностях простору ознак і тільки на потужних комп'ютерах.

2.4.4 Лінгвістичні (структурні) методи

Лінгвістичні методи розпізнавання образів засновані на використанні спеціальних граматик, що породжують мови, за допомогою яких може описуватися сукупність властивостей розпізнаваних об'єктів.

Для різних класів об'єктів виділяються непохідні (атомарні) елементи (під-образи, ознаки) і можливі відносини між ними. Граматикою називають правила побудови об'єктів з цих непохідних елементів.

Таким чином, кожен об'єкт являє собою сукупність непохідних елементів, «з'єднаних» між собою тими чи іншими способами або, іншими словами, «пропозицією» деякого «мови». Хотілося б особливо підкреслити дуже значну світоглядну цінність цієї думки.

Шляхом синтаксичного аналізу (граматичного розбору) «пропозиції» визначається його синтаксична «правильність» чи, що еквівалентно, чи може певна фіксована граматика, що описує клас, породити наявний опис об'єкта.

Однак завдання відновлення (визначення) граматик по деякому безлічі висловлювань (пропозицій – описів об'єктів), що породжують цю мову, є важко формалізується.

2.4.5 Екстенсіональні методи

У методах даної групи, на відміну від інтенсіональних напрямків, кожному об'єкту, що вивчається в більшій чи меншій мірі надається самостійне діагностичне значення. За

своєю суттю ці методи близькі до клінічного підходу, який розглядає людей не як проранжувати з того чи іншого показника ланцюжок об’єктів, а як цілісні системи, кожна з яких індивідуальна і має особливу діагностичну цінність. Таке дбайливе ставлення до об’єктів дослідження не дозволяє виключати або втрачати інформацію про кожну окрему об’єкті, що відбувається при застосуванні методів інтенсіональні напрямки, які використовують об’єкти тільки для виявлення і фіксації закономірностей поведінки їх атрибутів.

Основними операціями в розпізнаванні образів за допомогою обговорюваних методів є операції визначення подібності та відмінності об’єктів. Об’єкти в зазначеній групі методів грають роль діагностичних прецедентів. При цьому в залежності від умов конкретного завдання роль окремого прецеденту може змінюватися в найширших межах: від головної і визначальною і до вельми непрямої участі в процесі розпізнавання. У свою чергу умови задачі можуть вимагати для успішного вирішення участі різної кількості діагностичних прецедентів: від одного в кожному розпізнається класі до повного обсягу вибірки, а також різних способів обчислення заходів подібності та відмінності об’єктів. Цими вимогами пояснюється подальше розділення екстенсіональних методів на підкласи.

2.4.6 Метод порівняння з прототипом

Це найбільш простий екстенсіональний метод розпізнавання. Він застосовується, наприклад, в тому випадку, коли розпізнаються класи відображаються в просторі ознак компактними геометричними угрупованнями. В такому випадку зазвичай в якості точки – прототипу вибирається центр геометричній угруповання класу (або найближчий до центру об’єкт).

Для класифікації невідомого об’єкта знаходиться найближчий до нього прототип, і об’єкт відноситься до того ж класу, що і цей прототип. Очевидно, ніяких узагальнених образів класів в даному методі не формується.

В якості запобіжного близькості можуть застосовуватися різні типи відстаней. Часто для дихотомічних ознак використовується відстань Хеммінга, яке в даному випадку дорівнює квадрату евклидова відстані. При цьому вирішальне правило класифікації об’єктів еквівалентно лінійної вирішальної функції.

Зазначений факт слід особливо відзначити. Він наочно демонструє зв’язок прототипної і признаковой репрезентації інформації про структуру даних. Користуючись наведеними поданням, можна, наприклад, будь-яку традиційну вимірювальну шкалу, яка є

лінійною функцією від значень дихотомічних ознак, розглядати як гіпотетичний діагностичний прототип. У свою чергу, якщо аналіз просторової структури розпізнаються класів дозволяє зробити висновок про їх геометричній компактності, то кожен з цих класів достатньо замінити одним прототипом, який фактично еквівалентний лінійній діагностичної моделі.

На практиці, безумовно, ситуація часто буває відмінною від описаного ідеалізованого прикладу. Перед дослідником, що мають намір застосувати метод розпізнавання, заснований на порівнянні з прототипами діагностичних класів, встають непрості проблеми.

По-перше, це вибір міри близькості (метрики), від якого може істотно змінитися просторова конфігурація розподілу об'єктів. По-друге, самостійної проблемою є аналіз багатовимірних структур експериментальних даних. Обидві ці проблеми особливо гостро постають перед дослідником в умовах високої розмірності простору ознак, характерною для реальних завдань.

2.4.7 Метод k найближчих сусідів

Метод k найближчих сусідів для вирішення завдань дискримінантного аналізу був вперше запропонований ще в 1952 році. Він полягає в наступному.

При класифікації невідомого об'єкта знаходиться задане число (k) геометрично найближчих до нього в просторі ознак інших об'єктів (найближчих сусідів) з уже відомою приналежністю до розпізнаваним класах. Рішення про віднесення невідомого об'єкта до того чи іншого діагностичного класу приймається шляхом аналізу інформації про цю відому приналежності його найближчих сусідів, наприклад, за допомогою простого підрахунку голосів.

Спочатку метод k найближчих сусідів розглядався як непараметричний метод оцінювання відношення правдоподібності. Для цього методу отримані теоретичні оцінки його ефективності в порівнянні з оптимальним Байєсова класифікатором. Доведено, що асимптотичні ймовірності помилки для методу k найближчих сусідів перевищують помилки правила Байеса не більше ніж в два рази.

При використанні методу k найближчих сусідів для розпізнавання образів дослідніку доводиться вирішувати складну проблему вибору метрики для визначення близькості діагностованих об'єктів. Ця проблема в умовах високої розмірності простору ознак надзвичайно загострюється внаслідок достатньої трудомісткості даного методу, яка

стає значущою навіть для високопродуктивних комп'ютерів. Тому тут так само, як і в методі порівняння з прототипом, необхідно вирішувати творчу задачу аналізу багатовимірної структури експериментальних даних для мінімізації числа об'єктів, що становлять діагностичні класи.

Необхідність зменшення числа об'єктів в навчальній вибірці (діагностичних прецедентів) є недоліком даного методу, так як зменшує представництво навчальної вибірки.

2.4.8 Алгоритми обчислення оцінок

Принцип дії алгоритмів обчислення оцінок (АОО) складається в обчисленні пріоритетів (оцінок подібності), що характеризують «близькість» розпізнається і еталонних об'єктів по системі ансамблів ознак, що представляє собою систему підмножин заданої множини ознак.

На відміну від всіх раніше розглянутих методів алгоритми обчислення оцінок принципово по-новому оперують описами об'єктів. Для цих алгоритмів об'єкти існують одночасно в самих різних підпросторах простору ознак. Клас АВО доводить ідею використання ознак до логічного кінця: оскільки не завжди відомо, які поєднання ознак найбільш інформативні, то в АВО ступінь подібності об'єктів обчислюється при зіставленні всіх можливих або певних поєднань ознак, що входять в опису об'єктів.

Ярлики ознак (підпростору) автори називають опорними множинами або множинами часткових описів об'єктів. Вводиться поняття узагальненої близькості між розпізнаваним об'єктом і об'єктами навчальної вибірки (з відомою класифікацією), які називають еталонними об'єктами. Ця близькість представляється комбінацією близькості розпізнається об'єкта з еталонними об'єктами, обчислених на множинах часткових описів. Таким чином, АВО є розширенням методу k найближчих сусідів, в якому близькість об'єктів розглядається тільки в одному заданому просторі ознак.

Ще одним розширенням АВО є те, що в даних алгоритмах завдання визначення подібності та відмінності об'єктів формулюється як параметрична і виділено етап настройки АВО за навчальною вибіркою, на якому підбираються оптимальні значення введених параметрів. Критерієм якості служить помилка розпізнавання, а параметрізується буквально все:

- правила обчислення близькості об'єктів в підпросторах ознак;
- правила обчислення близькості об'єктів за окремими ознаками;

– ступінь важливості того чи іншого еталонного об’єкта як діагностичного прецеденту;

– значимість вкладу кожного опорного безлічі ознак в підсумкову оцінку подібності розпізнається об’єкта з будь-яким діагностичним класом.

Параметри АВО задаються у вигляді значень порогів і (або) як ваги зазначених складових. Теоретичні можливості АВО принаймні не нижче можливостей будь-якого іншого алгоритму розпізнавання образів, так як за допомогою АВО можуть бути реалізовані всі мислимі операції з досліджуваними об’єктами.

Але, як це зазвичай буває, розширення потенційних можливостей наштовхується на великі труднощі при їх практичному втіленні, особливо на етапі побудови (настройки) алгоритмів даного типу.

Окремі труднощі відзначалися раніше при обговоренні методу k найближчих сусідів, який можна було інтерпретувати як усічений варіант АВО. Його теж можна розглядати в параметричному вигляді і звести задачу до пошуку зваженої метрики обраного типу. У той же час вже тут для високоразмерних завдань виникають складні теоретичні питання і проблеми, пов’язані з організацією ефективного обчислювального процесу.

Для АВО, якщо спробувати використовувати можливості даних алгоритмів в повному обсязі, зазначені труднощі зростають багаторазово.

Зазначені проблеми пояснюють те, що на практиці застосування АВО для вирішення високоразмерних завдань супроводжується введенням будь-яких евристичних обмежень і припущень. Зокрема, відомий приклад використання АВО в психодіагностики, в якому апробована різновид АВО, фактично еквівалентна методу k найближчих сусідів.

2.4.9 Колективи вирішальних правил

Так як різні алгоритми розпізнавання проявляють себе по-різному на одній і тій же вибірці об’єктів, то закономірно постає питання про знаходження синтетичного вирішального правила, що адаптивно використовує сильні сторони цих алгоритмів. У синтетичному вирішальному правилі застосовується дворівнева схема розпізнавання. На першому рівні працюють приватні алгоритми розпізнавання, результати яких об’єднуються на другому рівні в блоці синтезу. Найбільш поширені способи такого об’єднання засновані на виділенні областей компетентності того чи іншого приватного

алгоритму. Найпростіший спосіб знаходження областей компетентності полягає в апріорно розбитті простору ознак виходячи з професійних міркувань конкретної науки (наприклад розшарування вибірки за певною ознакою). Тоді для кожної з виділених областей будується власний розпізнає алгоритм. Інший спосіб базується на застосуванні формального аналізу для визначення локальних областей простору ознак як околиць розпізнаваних об'єктів, для яких доведена успішність роботи будь-якого приватного алгоритму розпізнавання.

Самий загальний підхід до побудови блоку синтезу розглядає підсумкові показники приватних алгоритмів як вихідні ознаки для побудови нового узагальненого вирішального правила. В цьому випадку можуть використовуватися всі перераховані вище методи інтенсіональних і екстенсіональних напрямків в розпізнаванні образів. Ефективними для вирішення завдання створення колективу вирішальних правил є логічні алгоритми типу «Кора» і алгоритми обчислення оцінок (АВО), покладені в основу так званого алгебраїчного підходу, що забезпечує дослідження і конструктивне опис алгоритмів розпізнавання, в рамки якого укладаються всі існуючі типи алгоритмів.

Для вирішення реальних завдань з групи методів інтенсіональні напрямки практичну цінність представляють параметричні методи і методи, засновані на пропозиціях про вид вирішальних функцій. Параметричні методи складають основу традиційної методології конструювання показників. Застосування цих методів в реальних задачах пов'язано з накладенням сильних обмежень на структуру даних, які призводять до лінійних діагностичним моделям з дуже приблизними оцінками їх параметрів. При використанні методів, заснованих на припущеннях про вид вирішальних функцій, дослідник також змушений звертатися до лінійних моделям. Це обумовлено високою розмірністю простору ознак, характерною для реальних завдань, яка при підвищенні ступеня поліноміальної вирішальної функції дає величезне зростання числа її членів при проблематичному супутньому підвищенні якості розпізнавання. Таким чином, спроектувавши область потенційного застосування інтенсіональних методів розпізнавання на реальну проблематику, отримаємо картину, відповідну добре відпрацьованою традиційної методології лінійних діагностичних моделей.

Застосування екстенсіональних методів не пов'язане з будь-яким припущеннями про структуру експериментальної інформації, крім того, що всередині було розпізнати класів повинні існувати одна або кілька груп чимось схожих об'єктів, а об'єкти різних класів повинні чимось відрізнятися один від одного. Очевидно, що при будь-якій кінцевій розмірності навчальної вибірки (а інший вона бути і не може) ця вимога виконується завжди просто з тієї причини, що існують випадкові відмінності між об'єктами. Як заходи подібності застосовуються різні заходи близькості (відстані) об'єктів в просторі ознак.

Тому ефективне використання екстенсіональних методів розпізнавання образів залежить від того, наскільки вдало визначені зазначені заходи близькості, а також від того, які об'єкти навчальної вибірки (об'єкти з відомою класифікацією) виконують роль діагностичних прецедентів. Успішне вирішення цих завдань дає результат, що наближається до теоретично досяжним меж ефективності розпізнавання.

Перевагам екстенсіональних методів розпізнавання образів протиставлена, в першу чергу, висока технічна складність їх практичного втілення. Для високо розмірних просторів ознак зовні просте завдання знаходження пар найближчих точок перетворюється в серйозну проблему. Також багато авторів відзначають в якості проблеми необхідність запам'ятовування досить великої кількості об'єктів, що становлять розпізнаються класи.

Само по собі це не є проблемою, проте сприймається як проблема (наприклад, в методі k найближчих сусідів) з тієї причини, що при розпізнаванні кожного об'єкта відбувається повний перебір всіх об'єктів навчальної вибірки.

Тому доцільно застосувати модель системи розпізнавання, в якій проблема повного перебору об'єктів навчальної вибірки при розпізнаванні знімається, так як він здійснюється лише один раз при формуванні узагальнених образів класів розпізнавання. При самому ж розпізнаванні здійснюється порівняння ідентифікованого об'єкта лише з узагальненими образами класів розпізнавання, кількість яких фіксоване і абсолютно не залежить від розмірності навчальної вибірки. Даний підхід дозволяє збільшувати розмірність навчальної вибірки до тих пір, поки не буде досягнуто необхідне високе якість узагальнених образів, зовсім при цьому не побоюючись, що це може призвести до неприйнятного збільшення часу розпізнавання (так як час розпізнавання в даній моделі взагалі не залежить від розмірності навчальної вибірки).

Теоретичні проблеми застосування екстенсіональних методів розпізнавання пов'язані з проблемами пошуку інформативних груп ознак, знаходження оптимальних метрик для вимірювання подібності та відмінності об'єктів і аналізу структури експериментальної інформації. У той же час успішне вирішення перерахованих проблем дозволяє не тільки конструювати ефективні розпізнають алгоритми, а й здійснювати перехід від екстенсіонального знання емпіричних фактів до Інтенсіональні знання про закономірності їх структури.

Залишається недостатньо розробленим питання про практичної застосовності тих чи інших теоретичних методів розпізнавання для вирішення практичних завдань при реальних (тобто досить значних) розмірностях даних і на реальних сучасних комп'ютерах.

Це завдання:

– визначення інформаційного вкладу ознак в інформаційний портрет узагальненого образу;

– кластерно-конструктивний аналіз узагальнених образів;

– визначення семантичного навантаження ознаки;

– семантичний кластерно-конструктивний аналіз ознак;

– змістовне порівняння узагальнених образів класів один з одним і ознак між собою (когнітивні діаграми, в т.ч. діаграми Мерліна).

Метод, який дозволив досягти вирішення цих завдань, також відрізняє засновану на ньому перспективну систему від інших систем, як компілятори відрізняються від інтерпретаторів, так як завдяки формуванню узагальнених образів в цій перспективній системі досягається незалежність часу розпізнавання від обсягів навчальної вибірки. Відомо, що саме існування цієї залежності призводить до практично неприйнятним витрат машинного часу на розпізнавання в таких методах, як метод k найближчих сусідів, АВО і КРП при таких розмірностях навчальної вибірки, коли можна говорити про достатню статистикою. На закінчення короткого огляду методів розпізнавання представимо суть вищевикладеного в зведеній таблиці (таб. 2.1), що містить коротку характеристику різних методів розпізнавання за такими параметрами:

– класифікація методів розпізнавання;

– області застосування методів розпізнавання;

– класифікація обмежень методів розпізнавання.

Таблиця 2.1 – Зведена таблиця класифікації методів розпізнавання, порівняння їх областей застосування і обмежень.

Классифікація методів розпізнання		Область використання	Недоліки
1	2	3	4
Інтенсіональні методи розпізнавання	Методи, грунтовані на оцінках щільності розподілу значень ознак (чи схожість і відмінності об’єктів)	Завдання з відомим розподілом, як правило, нормальним, необхідність набору великої статистики	Необхідність перебору усієї повчальної вибірки при розпізнаванні, висока чутливість до непоказності повчальної вибірки і артефактів
	Методи, грунтовані на припущеннях про клас вирішальних функцій	Класи мають бути такими, що добре розділяються, система ознак ортонормована	Має бути заздалегідь відомий вид вирішальної функції. Неможливість обліку нових знань про кореляції між ознаками

Продовження таблиці 2.1

1	2	3	4
	Логічні методи	Завдання невеликої розмірності простору ознак	При відборі логічних вирішальних правил (коньюнкций) потрібний повний перебір. Висока обчислювальна трудомісткість
	Лінгвістичні (структурні) методи	Завдання невеликої розмірності простору ознак	Завдання відновлення (визначення) граматики по деякій безлічі висловлювань (описів об'єктів), є такою, що важко формалізується. Невирішеність теоретичних проблем
Экстенсіональні методи розпізнавання	Метод порівняння з прототипом	Завдання невеликої розмірності простору ознак	Висока залежність результатів класифікації від міри відстані (метрики). Невідомість оптимальної метрики
	Метод k найближчих сусідів	Завдання невеликої розмірності по кількості класів і ознак	Висока залежність результатів класифікації від міри відстані (метрики). Необхідність повного перебору повчальної вибірки при розпізнаванні. Обчислювальна трудомісткість
	Алгоритми обчислення оцінок (голосування)	Завдання невеликої розмірності по кількості класів і ознак	Залежність результатів класифікації від міри відстані (метрики). Необхідність повного перебору повчальної вибірки при розпізнаванні. Висока технічна складність методу
	Колективи вирішальних правил	Завдання невеликої розмірності по кількості класів і ознак	Дуже висока технічна складність методу, невирішеність ряду теоретичних проблем, як при визначенні областей компетенції приватних методів, так і в самих приватних методах

Таким чином, огляд методів розпізнавання показує, що в даний час теоретично розроблений цілий ряд різних методів розпізнавання образів. У літературі наводиться розгорнута їх класифікація. Однак для більшості цих методів їх програмна реалізація

відсутня, і це глибоко закономірно, можна навіть сказати «зумовлено» характеристиками самих методів розпізнавання. Про це можна судити по тому, що такі системи мало згадуються в спеціальній літературі і інших джерелах інформації.

2.5 Аналіз методів пост обробки результату

Перевірка орфографії працює шляхом пошуку заданого рядка у своєму словнику відомих рядків. Будь-який рядок, не знайдений в словнику, вважається передбачуваною помилкою. Помилкова орфографія затверджується, тому що це орфографічна помилка по відношенню до словника перевірки орфографії – який може бути обмеженим або неповним.

Щоб виправити орфографічну помилку, перевірка орфографії припускає, що ваша орфографічна помилка має бути мутацією одного з рядків в його словнику. Щоб запропонувати виправлення, перевірка орфографії шукає у своєму словнику рядки, схожі на друкарські помилки. Потім він упорядковує потенційні виправлення по мірі їх схожості і вірогідності. Відомий рядок, найближчий до вашої орфографічної помилки, є запропонованою поправкою.

Краща міра схожості робить більше, ніж просто підрахунок кількості загальних елементів між наборами. Враховуючи набір елементів в наборах A і B, можна вирухувати долю елементів, знайдених в обох наборах. Чим вище доля, тим більше схожості наборів. Це називається алгоритмом Жакара.

Колекція усіх унікальнії елементів в (A, B) => (A об’єднання B).

Набор загальних елементів в (A, B) => (A перетин B)

Ці два набори використовуються для обчислення коефіцієнта Жакара. Схожість між двома наборами A і B визначається як різність об’єднання і перетину.

Таблиця 2.2 Коефіцієнт Жакара для заданого слова.

A (Potatoes)	Кількість(Об’єднання)	Кількість(Перетин)	Результат
B (Potato)	**8**	6	6/8 = 0.75
C (Tomatos)	**9**	6	6/9 = 0.66

Базовий алгоритм перевірки орфографії виглядає так:

– розрахування коефіцієнт Жакара для друкарської помилки з кожним рядком в словнику;

– збір цих рядків з рейтингом Жакара вище, ніж деякий поріг;

– сортування збігів і вибір кращих збігів в якості підказок.

2.6 Аналіз методів машинного перекладу тексту

Машинний переклад може використати метод, грунтований на лінгвістичних правилах, тобто слова переводитимуться лінгвістично – найбільш відповідні (що усно говорять) слова цільової мови замінять ті, що в початковій мові.

Часто стверджується, що успіх машинного перекладу вимагає першого рішення проблеми розуміння природної мови.

Як правило, методи на основі правил аналізують текст, зазвичай створюючи проміжне, символічне представлення, з якого генерується текст на цільовій мові. Відповідно до характеру проміжного представлення, підхід описується як міжязиковий машинний переклад або машинний переклад на основі перенесення. Ці методи вимагають великих лексиконів з морфологічною, синтаксичною і семантичною інформацією і великими наборами правил.

Враховуючи достатню кількість даних, програми машинного перекладу часто працюють досить добре, щоб що рідний, що говорить однієї мови отримав приблизний сенс того, що написано іншим таким, що рідним, що говорить. Складність полягає в отриманні достатньої кількості даних потрібного типу для підтримки конкретного методу. Наприклад, великий багатомовний корпус даних, необхідний для роботи статистичних методів, не є необхідним для граматичних методів.

Але потім, граматичним методам потрібний досвідчений лінгвіст, щоб ретельно конструювати граматику, яку вони використовують.

Для перекладу між тісно пов’язаними мовами може використовуватися методика, що називається машинним перекладом на основі правил.

2.6.1 Машинний переклад на основі правил

Парадигма машинного перекладу на основі правил включає парадигми машинного перекладу на основі передачі, міжмовного машинного перекладу і машинного перекладу на основі словників. Цей тип перекладу використовується в основному при створенні словників і граматичних програм. На відміну від інших методів, RBMT включає більше інформації про лінгвістику початкової і цільової мов, використовуючи морфологічні і синтаксичні правила і семантичний аналіз обох мов.

Базовий підхід припускає зв'язок структури вхідної пропозиції із структурою вихідного речення за допомогою синтаксису і аналізатора для початкової мови, генератора для цільової мови і лексикону передачі для фактичного перекладу. Найбільший спад RBMT полягає в тому, що все повинно бути ясно : орфографічні варіації і помилкове введення мають бути частиною аналізатора початкової мови, щоб впоратися з ним, і лексичні правила вибору мають бути написані для усіх випадків неоднозначності.

Адаптація до нових доменів сама по собі не так складна, оскільки основна граматика однакова по доменах, а специфічне для домена коригування обмежується лексичним коригуванням вибору.

2.6.2 Машинний переклад на основі передачі

Машинний переклад на основі передачі аналогічний міжмовному машинному перекладу в тому сенсі, що він створює переклад з проміжного представлення, що імітує значення вхідного речення. На відміну від міжмовного МП, він частково залежить від мовної пари, що бере участь в перекладі.

Однією з основних особливостей систем машинного перекладу на основі передачі є фаза, яка «переносить» проміжне представлення тексту на мові оригіналу на проміжне представлення тексту на мові мети. Це може працювати на одному з двох рівнів лінгвістичного аналізу, або десь між ними.

Поділяють 2 рівні перенесення:

– поверхневе перенесення (синтаксис)
– глибока передача (семантика).

Поверхневе перенесення. Цей рівень характеризується передачею «синтаксичних структур» між початковою і цільовою мовами. Він підходить для мов в одній сім'ї або

одного типу, наприклад в романських мовах між іспанським, каталонським, французьким, італійським і т. д.

Глибока передача. Цей рівень створює семантичне представлення, залежне від початкової мови. Це представлення може складатися з ряду структур, які представляють значення. У цих системах перенесення зазвичай утворюються предикати. Переклад також зазвичай вимагає структурного перенесення. Цей рівень використовується для перекладу між віддаленіше спорідненими мовами (наприклад, іспано-англійським або іспано-баським і так далі).

2.6.3 Міжмовний машинний переклад

Міжмовний машинний переклад – один з прикладів грунтованих на правилах машинно-перекладацьких підходів. При такому підході початкова мова, тобто текст, що перекладається, перетвориться в між'язикову мову, тобто «мовне нейтральне» представлення, незалежне від будь-якої мови. Цільова мова потім генерується з интерлингва. Однією з основних переваг цієї системи є те, що міжмовна стає ціннішою у міру збільшення числа цільових мов, в які вона може бути перетворена.

У міжмовних системах машинного перекладу існують два одномовні компоненти: аналіз початкової мови і міжмовної мови і генерація интерлингуа і цільової мови. Проте необхідно розрізняти міжмовні системи, що використовують тільки синтаксичні методи (наприклад, системи, розроблені в 1970-х роках в університетах Гренобля і Техасу) і системи, грунтовані на штучному інтелекті (з 1987 року в Японії і дослідження в університетах Південної Каліфорнії і Карнеги-Меллона).

Для міжмовної системи машинного перекладу потрібні наступні ресурси:

– словники (чи лексикони) для аналізу і генерації (специфічні для домена і задіяних мов);

– концептуальний лексикон (специфічний для домена), що є базою знань про події і сутності, відомі в домені;

– набір правил проекції специфічних для домена і мов;

– граматики для аналізу і генерації відповідних мов.

Одна з проблем систем машинного перекладу, ґрунтованих на знаннях, полягає в неможливості створення баз даних для доменів, більших, ніж дуже специфічні області. Інша справа, що обробка цих баз даних є дуже дорогою з точки зору обчислень.

2.6.4 Машинний переклад на основі словників

Машинний переклад може використати метод, ґрунтований на словникових одиницях, тобто слова переводитимуться як словникові – слово за словом, як правило, без великої кореляції сенсу між ними. Пошук в словнику може виконуватися з морфологічним аналізом або лематизацією або без нього.

Хоча такий підхід до машинного перекладу, ймовірно, є найменш складним, машинний переклад на основі словників ідеально підходить для перекладу довгих списків фраз на під-центральний (тобто не повній пропозиції) рівень, наприклад, інвентарних запасів або простих каталогів продуктів і послуг.

Його також можна використати для прискорення ручного перекладу, якщо людина, що його, що виконує, вільно володіє обома мовами і, отже, здатний коригувати синтаксис і граматику.

2.6.5 Статистичний машинний переклад

Статистичний машинний переклад намагається генерувати переклади з використанням статистичних методів, ґрунтованих на двомовних текстових корпораціях, таких як канадський корпус Hansard, англо-французький запис канадського парламенту і EUROPARL, запис Європейського парламенту. Там, де такі корпорації доступні, добрі результати можуть бути досягнуті при перекладі подібних текстів, але такі корпорації як і раніше рідкісні для багатьох мовних пар. Першим статистичним програмним забезпеченням машинного перекладу була CANDIDE від IBM.

Компанія Google використала SYSTRAN впродовж декількох років, але в жовтні 2007 року перейшла на метод статистичного перекладу. У 2005 році компанія Google поліпшила свої можливості внутрішнього перекладу, використовуючи приблизно 200 мільярдів слів з матеріалів Організації Об’єднаних Націй для навчання їх системі;

Google Translate і аналогічні програми статистичного перекладу працюють шляхом виявлення шаблонів в сотнях мільйонів документів, які раніше переводилися людьми, і створення розумних припущень на основі отриманих результатів. В цілому, чим більше документів, що перекладаються людиною на цій мові, тим більше вірогідності того, що переклад матиме хорошу якість.

Новіші підходи до статистичного машинного перекладу, такі як METIS II і PRESSEMT, використовують мінімальний розмір корпусу і замість цього фокусуються на виведенні синтаксичної структури за допомогою розпізнавання шаблонів. З подальшим розвитком, це може дозволити статистичному машинному перекладу працювати з одномовним текстовим корпусом.

Найбільший спад СМП включає його залежність від величезної кількості паралельних текстів, його проблеми з морфологічними мовами (особливо з перекладом такими мовами) і його нездатність виправити одномовні помилки.

2.6.6 Машинний переклад на основі прикладів

У основі машинного перекладу на прикладах лежить ідея перекладу аналогічно. Стосовно процесу перекладу людиною, думка про те, що переклад виконується аналогічно, є відмовою від ідеї, що люди переводять пропозиції, роблячи глибокий лінгвістичний аналіз. Замість цього, ця думка грунтована на переконанні, що люди переводять, спочатку розбираючи пропозиції на певні фрази, потім переводять ці фрази, і, нарешті, правильно складають ці фрагменти в одно довгу пропозицію.

Переклади по фразах виконуються по аналогії з попередніми перекладами. Принцип перекладу аналогічно кодується в машинному перекладі на основі прикладів за допомогою прикладів перекладів, які використовуються для навчання такої системи. Інші підходи до машинного перекладу, включаючи статистичний машинний переклад, також використовують двомовні корпуси для вивчення процесу перекладу. Загалом, система складається з трьох компонентів: пошуку відповідностей, рекомбінації і вирівнювання.

Пошук відповідностей : У відповідному компоненті виконується пошук безлічі прикладів перекладу для визначення схожих фрагментів текстів у вхідному реченні.

Рекомбінація: На цьому етапі фрагменти тексту, витягнуті на етапі відповідностей, об’єднуються для створення цілої пропозиції. Грунтуючись на структурі зберігання прикладів (наприклад, дерева, таблиці і т. д.), процес об’єднання може зажадати конкретні процедури для об’єднання текстових одиниць. Наприклад, якщо приклади зберігаються в деревовидній структурі, для утворення вихідних даних слід використати метод уніфікації деревовидних структур.

Вирівнювання: Щоб повністю відповідати граматиці цільової мови і зменшити кількість невідповідностей у вихідних даних, необхідно виконати деяку подальшу обробку, наприклад, узгодження підмета з дієсловом.

Машинний переклад на основі прикладів краще всього підходить для таких явищ мови, як фразові дієслова. Фразові дієслова мають дуже контекстно-залежні значення. Вони поширені в англійській мові і складаються з дієслова, за яким йде прислівник і/або привід, який називається часткою у складі дієслова. Фразові дієслова утворюють спеціалізовані контекстно-специфічні значення, які не можуть бути витягнуті з сенсу складових.

2.6.7 Гібридні системи машинного перекладу

Гібридний машинний переклад (ГМП) використовує сильні сторони статистичних і грунтованих на правилах методологій перекладу.

Декілька організацій МП затверджують гібридний підхід, що використовує як правила, так і статистику. Підходи розрізняються по ряду напрямів:

Правила після обробки статистикою : переклади виконуються з використанням механізму на основі правил. Потім статистика використовується при спробі коригування/виправлення вихідних даних з механізму правил.

Правила використовуються для попередньої обробки даних з метою ефективнішого управління статистичним механізмом. Правила також використовуються для постобробки статистичних вихідних даних для виконання таких функцій, як нормалізація. Такий підхід має набагато більшу потужність, гнучкість і керованість при перекладі.

Він також забезпечує широкий контроль над способом обробки змісту як під час попереднього перекладу (наприклад, розмітка змісту і неперекладні терміни), так і після перекладу (наприклад, коригування і коригування після перекладу).

Зовсім нещодавно, з появою Neurean MT, з’являється нова версія гібридного машинного перекладу, яка поєднує в собі переваги правил, статистичного і нейронного машинного перекладу. Цей підхід дозволяє отримати переваги від попередньої і подальшої обробки в потоці операцій, керованому правилами, а також від НМП і СМП. Мінусом є природжена складність, яка робить підхід придатним тільки для конкретних випадків використання.

2.6.8 Нейронний машинний переклад

Нейронний машинний переклад (НМП) не є кардинальним кроком далі за те, що традиційно робиться в статистичному машинному перекладі (СМП). Його основним відходом є використання векторних представлень («вбудовувань», «безперервних просторових представлень») для слів і внутрішніх станів. Структура моделей простіша за фрази. Немає ніякої окремої мовної моделі, моделі перекладу, і моделі переупорядковування, але просто єдиній моделі послідовності, яка передбачає одно слово за один раз.

Проте це пророцтво послідовності залежить від усього вхідного речення і усієї вже отриманої цільової послідовності. Моделі НМП використовують глибоке навчання і навчання показності.

Спочатку моделювали послідовність слів, зазвичай використовуючи нейронну мережу що повторюється. Двонаправлена нейронна мережа, що повторюється, відома як кодер, використовується нейронною мережею для кодування вхідного речення для другого, відомого як декодер, який використовується для пророцтва слів на цільовій мові.

Згортальні нейронні мережі (Конвекції) в принципі дещо краще для тривалих безперервних послідовностей, але спочатку не були використані із-за декількох слабких місць, які були успішно компенсовані до 2017 року з використанням так званих підходів на основі уваги. Існують інші моделі покриття, що порушують питання в традиційному механізмі уваги, такі як ігнорування минулої інформації про вирівнювання, що призводить до надмірного перекладу і недостатнього перекладу.

До 2016 року більшість кращих МП – систем використали нейронні мережі. Google, Microsoft, IBM, Yandex і PROMT перекладацькі сервіси тепер використовують НМП. Google використовує Google Neuronean Machine Translation (GNMT) як перевага своїм попереднім статистичним методам. Microsoft використовує аналогічну технологію для своїх мовних перекладів (включаючи Microsoft Translator Live і Skype Translator). Група Harvard NLP випустила систему нейронного машинного перекладу з відкритим початковим кодом OpenNMT.

Технологія НМП може використовуватися поза рамками природної мови. Наприклад, було показано, що НМП може також працювати над початковим кодом комп’ютерних програм. При ретельному кодуванні початкового коду система автоматичного виправлення помилок SequencR навчається минулим фіксаціям, виробленим в сховищах Git, і здатна генерувати правильні однолінійні виправлення.

3 КОМП'ЮТЕРНА РЕАЛІЗАЦІЯ СИСТЕМИ ДІДЖІТАЛІЗАЦІЇ БІБЛІОТЕКИ ТЕХНІЧНОЇ ЛІТЕРАТУРИ

3.1 Вибір мови програмування

Для розробки застосунку була мова Java. Це сильно типізована, об'єктно-орієнтована, проста для навчання мова. Програми котрі написані на цій мові транслюються у байт-код та виконуються на віртуальній Java-машині. Це дозволяє виконуватися цим програмам на будь-якому пристрої для котрого є відповідна віртуальна машина.

Мова Java потрібна для створення інтерактивних продуктів для мережі Internet. Два ключові елементи об'єдналися в технології мови Java:

– Java вивільняє міць об'єктно-орієнтованої розробки додатків, поєднуючи простий і знайомий синтаксис з надійним і зручним в роботі середовищем розробки. Це дозволяє широкому колу програмістів швидко створювати нові програми і нові аплети;

– Java надає програмісту багатий набір класів об'єктів для ясного абстрагування багатьох системних функцій, використовуваних при роботі з вікнами, мережею і для введення-виведення. Ключова риса цих класів полягає в тому, що вони забезпечують створення незалежних від використовуваної платформи абстракцій для широкого спектра системних інтерфейсів.

3.2 Spring фреймворк

Майже кожен написаний на мові програмування Java веб-додаток, має під собою якийсь фреймворк, що дозволяє будувати на Java серверні додатки. Найбільш популярним з цих фреймворків є Spring.

Spring Framework включає декілька модулів, які надають цілий ряд послуг, серед найбільш цікавих з них:

– Spring Core Container: це базовий модуль Spring, в якому передбачені пружинні контейнери (BeanFactory і потоковий контекст);

– орієнтоване на аспекти програмування: дозволяє реалізувати наскрізні виклики методів;

– аутентифікація і авторизація : процеси безпеки, що настроюються, підтримують ряд стандартів, протоколів, інструментів і практик за допомогою підпроекту Spring Security;

– доступ до даних: робота з реляційними системами управління базами даних на платформі Java з використанням Java Database Connectivity (JDBC) і об'єктно-реляційних засобів відображення, а також з базами даних NoSQL;

– інверсія контейнера управління : налаштування компонентів додатка і управління життєвим циклом Java- об'єктів, виконувана головним чином за допомогою уприскування залежностей;

– model-view-controller (контроллер представлення моделі): інфраструктура на основі файлів і сервлетів, що забезпечує можливості розширення і налаштування для веб-застосувань і веб-сервісів RESTful (передача представницького стану);

– управління транзакціями: уніфікує декілька API управління транзакціями і координує транзакції для об'єктів Java;

– тестування: класи підтримки для запису одиничних тестів і інтеграційних тестів.

3.3 Tesseract

Tesseract це відкритий джерело оптичного механізму розпізнавання символів. Tesseract розпочав свою роботу в дослідницькому проекті PhD в HP Labs у Брістолі. Він був розроблений у HP в період між 1984 і 1994 роками. Вона була модифікована та вдосконалена у 1995 році з більшою точністю. Наприкінці 2005 року HP випустила Tesseract для відкритого коду і зараз доступна.

Переваги Tesseract:

– програма має відкритий початковий код, тож може використовуватися розробниками без будь-якої ліцензії;

– Tesseract має підтримку більш ніж 100 мов та постачається з вбудованими натренованими моделями високої якості для англійської мови та чисел;

– користувач може натренувати свою модель для підвищення якості розпізнавання;

– має велику кількість режимів оптичного розпізнавання, серед яких з'явився режим нейронної мереже.

Недоліки Tesseract:

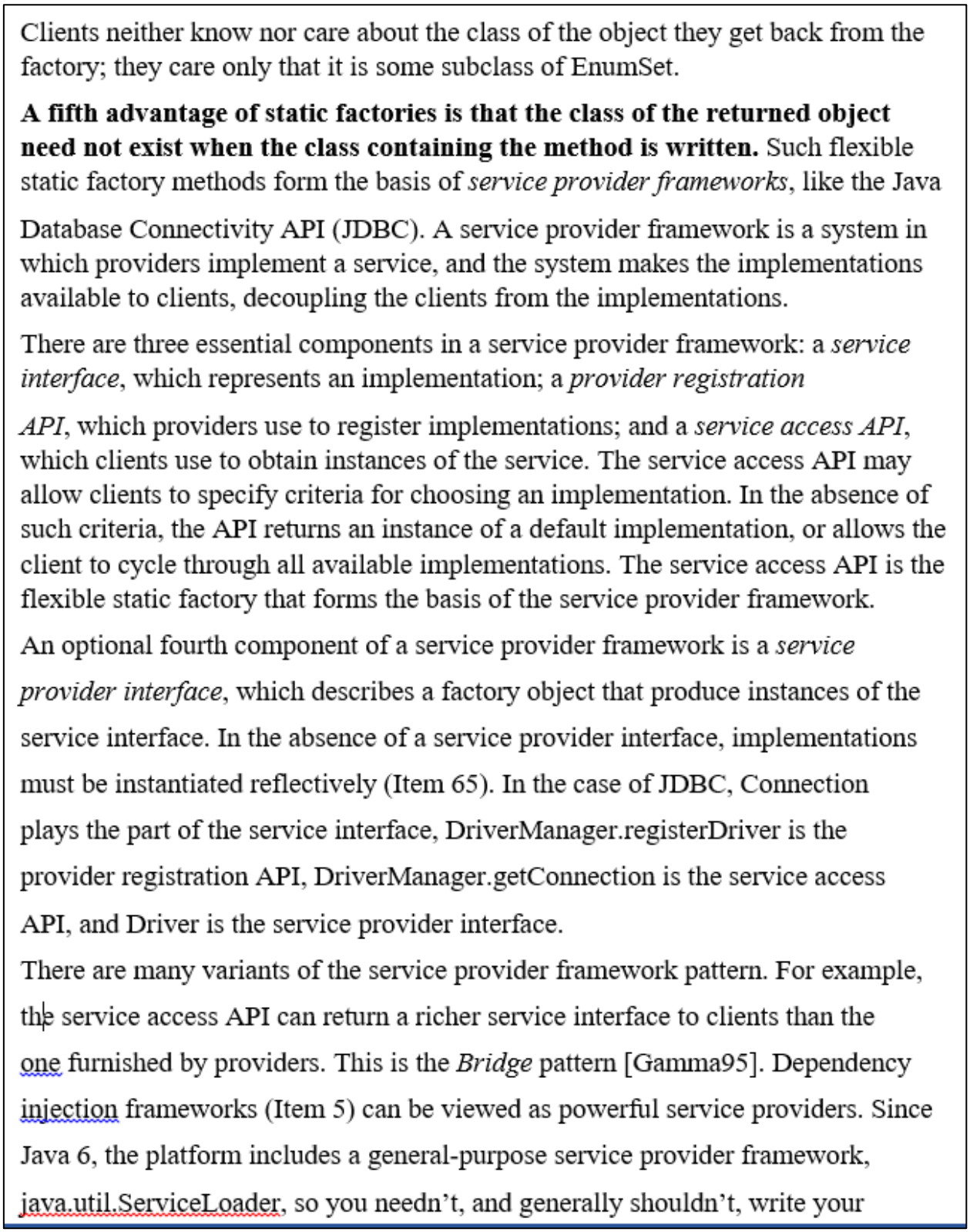
– Tesseract обмежений розпізнанням мови;

– потрібно багато зусиль, щоб зібрати дані тренера на різних мовах і реалізувати їх;

– також необхідно виконати додаткову роботу по обробці зображень, оскільки це найважливіша частина, яка дійсно має значення, коли йдеться про продуктивність оптичного розпізнавання тексту;

– виконавши такий великий об’єм роботи, ніяке розпізнавання тексту не може забезпечити точність 100 відсотків.

Незважаючи на свої недоліки, Tesseract залишається кращою OCR системою. В якості реалізації на основі Tesseract, в рамках магістерської роботи використовувалася бібліотека Tess4j, що дає додаткову підтримку документів формату TIFF, JPEG, GIF, PNG, BNP. В ДОДАТКУ А наведено фрагмент лістингу модуля «OCRTask.java», котрий і працює з бібліотекою Tess4j.



Clients neither know nor care about the class of the object they get back from the factory; they care only that it is some subclass of EnumSet.

A fifth advantage of static factories is that the class of the returned object need not exist when the class containing the method is written. Such flexible static factory methods form the basis of *service provider frameworks*, like the Java Database Connectivity API (JDBC). A service provider framework is a system in which providers implement a service, and the system makes the implementations available to clients, decoupling the clients from the implementations.

There are three essential components in a service provider framework: a *service interface*, which represents an implementation; a *provider registration API*, which providers use to register implementations; and a *service access API*, which clients use to obtain instances of the service. The service access API may allow clients to specify criteria for choosing an implementation. In the absence of such criteria, the API returns an instance of a default implementation, or allows the client to cycle through all available implementations. The service access API is the flexible static factory that forms the basis of the service provider framework.

An optional fourth component of a service provider framework is a *service provider interface*, which describes a factory object that produce instances of the service interface. In the absence of a service provider interface, implementations must be instantiated reflectively (Item 65). In the case of JDBC, Connection plays the part of the service interface, DriverManager.registerDriver is the provider registration API, DriverManager.getConnection is the service access API, and Driver is the service provider interface.

There are many variants of the service provider framework pattern. For example, the service access API can return a richer service interface to clients than the one furnished by providers. This is the *Bridge* pattern [Gamma95]. Dependency injection frameworks (Item 5) can be viewed as powerful service providers. Since Java 6, the platform includes a general-purpose service provider framework, java.util.ServiceLoader, so you needn't, and generally shouldn't, write your

Рисунок 3.1 – Скріншот сторінки професійної літератури

На виході программа повертає такий текст:

Clients neither know nor care about the class of the object they get back from the factory; they care only that it is some subclass of EnumSet.
A fifth advantage of static factories is that the class of the returned object need not exist when the class containing the method is written. Such flexible static factory methods form the basis of service provider frameworks, like the Java Database Connectivity API (JDBC). A service provider framework is a system in which providers implement a service, and the system makes the implementations available to clients, decoupling the clients from the implementations.
There are three essential components in a service provider framework: a service interface, which represents an implementation; a provider registration API, which providers use to register implementations; and a service access API, which clients use to obtain instances of the service. The service access API may allow clients to specify criteria for choosing an implementation. In the absence of such criteria, the API returns an instance of a default implementation, or allows the client to cycle through all available implementations. The service access API is the flexible static factory that forms the basis of the service provider framework.
An optional fourth component of a service provider framework is a service provider interface, which describes a factory object that produce instances of the service interface. In the absence of a service provider interface, implementations ‘must be instantiated reflectively (Item 65). In the case of JDBC, Connection plays the part of the service interface, DriverManager.registerDriver is the provider registration API DriverManager.getConnection is the service access API, and Driver is the service provider interface.
There are many variants of the service provider framework pattern. For example, th service access API can return a richer service interface to clients than the one furnished by providers. This is the Bridge pattern [Gamma95]. Dependency injection frameworks (Item 5) can be viewed as powerful service providers. Since Java 6, the platform includes a general-purpose service provider framework, java.util.ServiceLoader, so you needn’t, and generally shouldn’t.

Рисунок 3.2 – Результат розпізнавання сторінки професійної літератури

3.4 Програмна реалізація автоматичного перекладу

При вході в систему користувач має обрати до трьох цікавих йому словників. Пізніше вибір можна бути змінити безліч разів. Словники що доступні на даний момент: «Accounting», «American English», «Beer», «Biology», «Building», «Economicus», «Banking», «Economicus» «Government», «Economicus Insurance», «Economicus International», «Electronics», «Engineering», «Financial Management», «Financial Markets», «Forum Dictionary», «Geography», «Law», «Learning», «Computer», «Grammar», «Management», «Marketing», «Mechanical Engineering», «Menu Translate», «Patents», «Physics», «Psychology», «Sport», «Telecoms».

Додаток використовує комбінований спосіб перекладу даних. Спочатку текст перекладається методом на основі прикладів. Зчитаний та оброблений тест, порціями по одному абзацу, (найчастіше саме абзац виступає мінімальною змістовою одиницею) потрапляє до сервісу MyMemory. Цей сервіс зберігає величезну кількість перекладів професійних перекладачів з будь-якої тематики, що дозволяє йому легко підібрати велику кількість схожих текстів. Після того як тексти знайдені в базі даних, додаток складає матрицю відповідності вхідного тексту до знайдених аналогів щоб отримати відсоток співпадіння. Після того як матриці збудовані та отримані відсоткові значення збігів, додаток намагається з’єднати отримані уривки в один текст, та віддає кінцевий результат. В ДОДАТКУ Б наведено фрагмент лістингу модулю «TextTranslator.java», котрий працює з MyMemory API для виконання автоматизованого перекладу.

Другим етапом виступає переклад на основі словників. Після того як додаток отримав переклад разом із матрицею відповідностей, можна з легкістю знайти ті слова, в яких перший перекладач сумніваються. Слова з найменшим співпадінням відправляються до профільного словника. Після того як словник повернув результати перекладу, додаток починає фільтрувати на наявність цікавих користувачу словників. Після успішного виконання цієї операціє, додаток збирає всі отримані переклади, та починає їх консолідувати.

Після того як дані з обох систем перекладу отримані, додаток формує з них кінцеву відповідь користувачу, де зберігається зчитаний текст на англійській мові, разом з його перекладенним аналогом. На рисунку 3.3 зображений машинний переклад, зроблений на основі рисунку 3.1.

Клиенты не знают и не заботятся о классе объекта, который они возвращают с фабрики; их волнует только то, что это некоторый подкласс EnumSet. Пятое преимущество статических фабрик состоит в том, что класс возвращаемого объекта не должен существовать, когда класс, содержащий метод, написан. Такие гибкие статические фабричные методы образуют основу каркасов поставщиков услуг, таких как API подключения к базе данных Java (JDBC). Инфраструктура поставщика услуг - это система, в которой поставщики реализуют услугу, и система делает реализации доступными для клиентов, отделяя клиентов от реализаций. В структуре поставщика услуг есть три основных компонента: интерфейс службы, представляющий реализацию; API регистрации провайдера, который провайдеры используют для регистрации реализаций; и API доступа к сервису, который клиенты используют для получения экземпляров сервиса. API доступа к сервису может позволять клиентам указывать критерии выбора реализации. При отсутствии таких критериев API возвращает экземпляр реализации по умолчанию или позволяет клиенту циклически перебирать все доступные реализации. API доступа к сервису - это гибкая статическая фабрика, которая лежит в основе структуры поставщика услуг. Необязательным четвертым компонентом структуры поставщика услуг является интерфейс поставщика услуг, который описывает фабричный объект, который создает экземпляры интерфейса сервиса. В отсутствие интерфейса поставщика услуг реализации должны быть отражены (Ref. 65). В случае JDBC Connection играет роль интерфейса службы, DriverManager.registerDriver - это API регистрации поставщика. DriverManager.getConnection - это API доступа к службе, а Driver - интерфейс поставщика службы. Существует много вариантов шаблона структуры

Рисунок 3.3 – Результат машинного перекладу тексту

3.5 Програмна реалізація багатопотчного програмування

В угоду точності перекладу, перекладач на основі прикладів розбиває вхідний текст на абзаци, а переклад на основі словників – на окремі слова. Через це швидкість роботи

додатку суттєво знижуються. Для того щоб виправити цей недолік, необхідно використовувати разпараллелювання задач.

Для реалізації паралельного програмування використовувалася технологія паралельних потоків (parallel stream). Ця технологія з’явилася при виході Java8, та дозволяла розпаралелювати задачі декларативно. Після того як програміст вказував що хоче працювати з паралельним потоком, віртуальна машина Java (JVM) розбиває один цільний поток даних на декілька окремих потоків, а потім знов збирає воєдино. Кількість цих потоків визначається виключно самою JVM. Цей підхід дозволяє перенести навантаження з одного ядра процесора, на усі, що прискорює роботу додатку приблизно в 3 рази.

3.6 Тестування розробленої моделі

У відповідь на зміни в програмному забезпеченні будь-якої складності можуть виникати непередбачувані помилки. Відповідно, після внесення змін для будь-яких додатків, за винятком найпростіших або найменш важливих, необхідно проводити тестування. Тестування вручну є найповільнішим, найменш надійним і найбільш дорогим способом перевірити програмне забезпечення. На жаль, якщо можливість тестування не закладена в додаток на етапі проектування, це може бути єдиний доступний спосіб. Додатки, при написанні яких дотримувалися викладені в розділі 2 архітектурні принципи, підтримують автоматичні інтеграційні і функціональні тести.

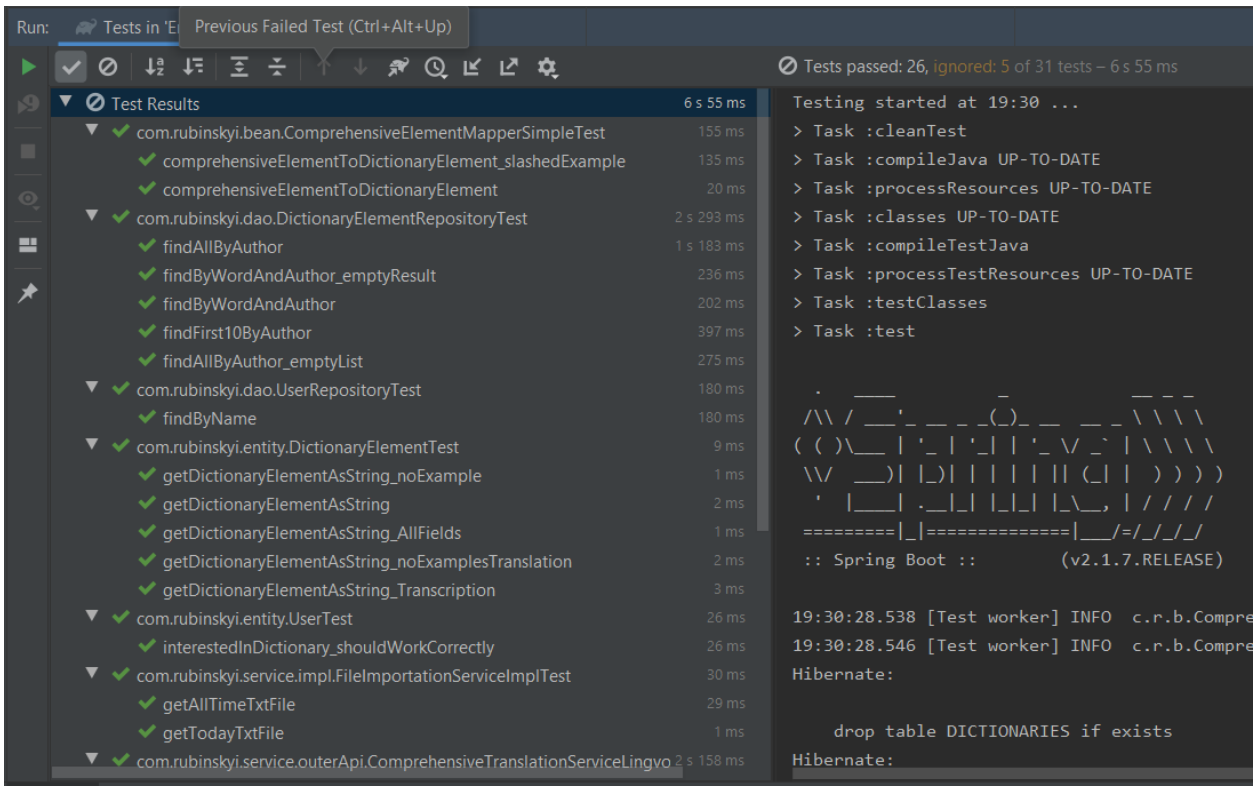


Рисунок. 3.4 – Приклад виконання тестів у середовищі розробки Intelij Idea

Тестування системи включає в себе різні рівні, такі як тестування коду, графічного інтерфейсу, тестування API. Автоматизоване тестування більш чітко виконує свої обов'язки чітко відстежуючи виконання заданих умов, але може забирати на себе забагато часу на розробку та підтримку при зміні вимог. Тому потрібно проводити аналіз областей, що підлягають тестуванню і на його основі обирати підхід до тестування.

3.6.1 Unit тестування

Unit тестування – це рівень тестування програмного забезпечення, на якому тестуються окремі блоки/компоненти програмного забезпечення. Метою є перевірка того, що кожен блок програмного забезпечення працює так, як він розроблений. Одиниця виміру є найменшою тестованою частиною будь-якого програмного забезпечення. Зазвичай він має один або декілька входів і зазвичай один вихід. У процедурному програмуванні одиницею може бути індивідуальна програма, функція, процедура і так далі

У об'єктно орієнтованому програмуванні найменшою одиницею є метод, який може належати базовому/суперкласу, абстрактному класу або похідному класу. Для

полегшення модульного тестування використовуються системи модульного тестування, драйвери, заглушки і підробні об’єкти.

Модульне тестування підвищує упевненість в зміні/підтримці коду. Якщо будуть написані хороші юніт тести і якщо вони запускатимуться кожного разу при зміні будь-якого коду, ми зможемо оперативно уловити будь-які дефекти, введені через зміну. Крім того, якщо коди вже зроблені менш взаємозалежними, щоб зробити можливим одиничне тестування, то неумисна дія змін у будь-якому коді менша. До головних переваг слід віднести:

– код придатніший для багатократного використання. Для забезпечення можливості тестування блоків код має бути модульними. Це означає, що код легше використати повторно;

– розвиток додатку відбувається швидше. При відсутності модульного тестування, після кожного корегування потаткового коду, треба ініціювати процес тестуваня. За допомоги модульного тестування після кожної модифікації початкового коду можна зловити баг, а не виправляти його після кожного змінення початкового коду;

– написання тестів займає час, але час компенсується меншою кількістю часу, який потрібно для виконання тестів. Знакає необхідність запускати графічний інтерфейс і надавати усі ці вхідні дані. І, звичайно, питомі тести надійніші. Розвиток в довгостроковій перспективі пришвидшується;

– зусилля, необхідні для пошуку і усунення дефектів, виявлених під час модульних випробувань, значно менше в порівнянні із зусиллями, необхідними для усунення дефектів, виявлених під час системних випробувань або приймальних випробувань;

– вартість фіксації дефекту, виявленого під час одиничного тестування, менше в порівнянні з дефектами, виявленими на більш високих рівнях;

– зменьшується складність відладки. Якщо тест завершується негативно, необхідно відлагоджувати тільки останні зміни. При тестуванні на більш високих рівнях необхідно сканувати зміни, зроблені впродовж декількох часових одиниць.

Враховуючи усі переваги модульного тестування, в атестаційній роботі було вирішено мати відсоток покриття останнім не меньше 70.

3.6.2 Інтеграційне тестування

Інтеграційне тестування – це рівень тестування програмного забезпечення, при якому окремі блоки об’єднуються і тестуються як група. Метою цього рівня тестування є виявлення несправностей у взаємодії між інтегрованими блоками. Для сприяння інтеграційному тестуванню використовуються тестові драйвери і тестові заглушки.

Розрізняють декілька видів інтеграційного тестування:

Big Bang – це підхід до інтеграційного тестування, при якому все або більшість блоків об’єднуються і тестуються за один хід. Цей підхід застосовується при отриманні групою тестування усього програмного забезпечення в пакеті. У чому різниця між тестуванням інтеграції Big Bang і системним тестуванням? Перший тестує тільки взаємодії між блоками, тоді як другою тестує усю систему.

Зверху «вниз» – це підхід до інтеграційного тестування, при якому спочатку тестуються блоки верхнього рівня, а потім послідовно тестуються блоки нижнього рівня. Цей підхід застосовується при застосуванні підходу, грунтованого на принципі зверху вниз. Тестові заглушки потрібні для моделювання одиниць виміри нижчого рівня, які можуть бути відсутніми на початкових етапах.

«Від низу до верху» – це підхід до інтеграційного тестування, при якому блоки нижнього рівня тестуються поетапно, а блоки верхнього рівня – поетапно. Такий підхід застосовується при застосуванні підходу від низу до верху. Драйвери тестування потрібні для моделювання одиниць більш високого рівня, які можуть бути відсутніми на початкових етапах.

Sandwide/Hybrid – це підхід до інтеграційного тестування, який є поєднанням підходів зверху «вниз» і «від низу до верху».

Для виконання автоматичного тестування розробленої системи до складу проекту розроблено модуль «Benchmark.java», лістинг котрого наведено у додатку В.

3.6.3 Протестований функціонал

Під час тестування системи були протестовані такі аспекти як:

- розпізнавання символів в одному абзаці;
- розпізнавання символів в сторінці книги;
- розпізнавання символів в умовах великої кількістю шуму на зображенні;
- розпізнавання усіх символів англійської мови та чисел;

– автоматичний переклад кожного з розпізнаних текстів;
– точність надання користувачу перекладу, що стосується його інтересів;
– безпечність веб додатку.

Програмний код був покритий на 70 відсотків, що є необхідного кількістю для великих промислових програм.

3.7 Розміщення в хмарі

Впровадження програмної системи включає в себе розгортання серверу на хмарних системах Amazon та установки обладнання для зчитування QR-кодів для кінцевих користувачів. Виготовлення та установка обладнання користувачу виконуються безпосередньо після замовлення.

Amazon надає своїм користувачам багатий набір хмарних сервісів з дуже гнучкою конфігурацією для користувача, що дозволяє обрати найоптимальніший варіант за показниками ціни та якості. Також Amazon має розвинуту та надійну інфраструктуру, що забезпечує надійність роботи навіть при дуже великих навантаженнях. У тому разі, коли навантаження перевищує максимально передбачений рівень, користувач може, у пару кліків в адміністративній панелі, додати обчислювальної потужності, або розгорнути додатковий інстанс системи для розподілу навантаження.

Amazon Elastic Compute Cloud (Amazon EC2) – це веб-сервіс, що надає безпечні масштабовані обчислювальні ресурси в хмарі. Він допомагає розробникам, полегшуючи проведення великомасштабних обчислень в хмарі. Простий веб-інтерфейс сервісу Amazon EC2 дозволяє отримати доступ до обчислювальних ресурсів і налаштувати їх з мінімальними затратами. Він надає користувачам повний контроль над ресурсами, які вони можуть запускати в зарекомендувала себе обчислювальному середовищі Amazon. Скорочуючи до декількох хвилин процес налаштування і запуску нових інстанси серверів, сервіс Amazon EC2 дозволяє швидко масштабувати обчислювальні ресурси з урахуванням мінливих вимог. Amazon EC2 змінює економічну складову процесу обчислень, надаючи можливість платити тільки за використовувані ресурси. Amazon EC2 дозволяє розробникам уникати поширених хибних сценаріїв і створювати відмовостійкі додатки.

Розгортання проекту на Amazon EC2 майже не відрізняється від розгортання системи на локальній машині розробника. При запуску інстансу Amazon EC2 користувач обирає операційну систему, в якій буде працювати. Він може обрати серед Amazon Linux,

Ubuntu, CentOS та Red Hat Enterprise Linux. Після цього користувач під'єднується до серверу через ssh та конфігурує все самостійно. У разі використання контейнерів, таких як Docker, розгортання системи може складатись лише з декількох команд.

Amazon Aurora – MySQL і PostgreSQL сумісна реляційна база даних, побудована для хмари, яка поєднує в собі продуктивність і доступність традиційних корпоративних баз даних з простотою і економічністю баз даних з відкритим початковим кодом.

Amazon Aurora до п'яти разів швидше за стандартні бази даних MySQL і в три рази швидше за стандартні бази даних PostgreSQL. Він забезпечує безпеку, доступність і надійність комерційних баз даних на 1/10 вартостей. Amazon Aurora повністю управляється Amazon Relative Database Service (RDS), яка автоматизує трудомісткі завдання адміністрування, такі як виділення устаткування, налаштування баз даних, виправлення і резервне копіювання.

Amazon Aurora має розподілену відмовостійку самовосстановляющуюся систему зберігання даних, яка автоматично масштабується до 64tb на екземпляр бази даних. Вона забезпечує високу продуктивність і доступність завдяки 15 реплікам для читання з низькою затримкою, відновленню на певний момент часу, безперервному резервному копіюванню на Amazon S3 і реплікації в трьох зонах доступності (AZS).

Відвідайте консоль управління Amazon RDS, щоб створити перший екземпляр бази даних Aurora і почати міграцію баз даних MySQL і PostgreSQL.

Серед переваг Aurora необхідно відмітити такі:

– висока продуктивність і масштабованість;

В п'ять разів вища пропускна можлівість стандартного MySQL і 3х пропускну спроможність стандартного PostgreSQL. Можна легко масштабувати розгортання бази даних вгору і вниз від невеликих до великих типів екземплярів у міру зміни потреб. Для масштабування місткості і продуктивності читання можна додати до 15 реплік читання з низькою затримкою в трьох зонах доступності;

– висока доступність і довговічність;

Amazon Aurora розроблений, щоб запропонувати більше, ніж доступність 99,99% на 6 копій ваших даних через 3 Зони Доступності і підтримуючи дані безперервно в Amazon S3. Amazon Aurora автоматично нарощує сховище в міру необхідності, до 64tb на екземпляр бази даних;

– високий рівень безпеку;

Amazon Aurora забезпечує декілька рівнів безпеки для вашої бази даних. До них відносяться мережева ізоляція за допомогою Amazon VPC, шифрування на відпочинку за допомогою ключів, які ви створюєте і контролюєте за допомогою AWS Key Management Service (KMS) і шифрування даних в дорозі за допомогою SSL;

– база повністю керована.

Amazon Aurora повністю керується службою реляційних баз даних Amazon (RDS). Ви більше не повинні хвилюватися про завдання управління базою даних, таких як апаратні засоби, що забезпечують, внесення виправлень програмного забезпечення, установка, конфігурація або резервні копії. Aurora автоматично і безперервно відстежує і резервує вашу базу даних на Amazon S3, забезпечуючи детальне відновлення на певний момент часу.

Використання цих двох сервісів забезпечить надійну і стабільну роботу системи, що задовольняє нефункціональним вимогам доступності та надійності системи. Стадія впровадження та супроводу йде наступною і заключною у процесі поетапного впровадження. Після того, як усі помилки системи було виявлено, виправлено, систему протестовано знов і засвідчено, що помилок не знайдено, її можна впроваджувати для користування кінцевому споживачеві. Супровід же здійснюється шляхом впровадження нових версій програмного забезпечення для існуючої системи.

4 ОХОРОНА ПРАЦІ

4.1 Аналіз потенційно небезпечних і шкідливих виробничих факторів , що впливають на персонал

Персональні ЕОМ типу IBM PC AT має наступні характеристики:

– споживана потужність 350 Вт;

– робоча напруга 220 В;

– напруга джерел живлення +12 В, -12 В, 5 В;

– робоча частота 50 Гц.

Виходячи з приведених характеристик, очевидно, що для користувача існує небезпека поразки електричним струмом у разі недбалого поводження з комп'ютером і порушення правил експлуатації (невиконання огляду відкритих частин ПЕВМ, що знаходяться під напругою або знятих для ремонту вузлів і т. д.).

Джерелами підвищеної небезпеки можуть служити наступні елементи:

– розподільний щит;

– джерела живлення;

У відповідності з ДСН 3.3.6.042-99 [41] до легкої фізичної роботи відносяться всі види діяльності, вироблювані сидячи і не вимагаючи фізичної напруги. Робота користувача розробленого пакету програм відноситься до категорії Іа.

Згідно з ДСТУ Б А.3.2-13:2011 [42] приміщення для ПЕОМ по ступеню небезпеки поразки людини електричним струмом відноситься до приміщень без підвищеної небезпеки (немає струмопровідної полови, вогкості, підвищеної температури, можливості одночасного дотику до корпусів устаткування з “землею” і до струмонесучих частин).

У відповідності з ДСанПіН 3.3.2-007-98 [43] при обслуговуванні ПЕВМ мають місце фізичні і психофізичні небезпечні, а також шкідливі виробничі чинники:

– підвищене значення напруги в електричному ланцюзі, замикання якого може відбутися через тіло людини;

– підвищений рівень статичної електрики;

– підвищений рівень електромагнітних випромінювань;

– підвищена або знижена температура повітря робочої зони;

– підвищена або знижена рухливість повітря;

– підвищена або знижена вогкість повітря;

– відсутність або недолік природного світла;

– підвищена пульсація світлового потоку;

– недостатня освітленість робочого місця;

– підвищений рівень шуму на робочому місці;

– розумове перенапруження;

– емоційні навантаження;

– монотонність праці.

В промислових приміщеннях повинні бути забезпечені санітарно-гігієнічними етичні вимоги до повітря робочої зони згідно з ДСН 3.3.6.042-99 [41].

4.2 Заходи щодо техніки безпеки

Основним небезпечним чинником при роботі з ЕОМ є небезпека поразки людини електричним струмом, яка усугубляє тим, що органи чуття людини не можуть на відстані знайти наявності електричної напруги на устаткуванні.

Проходячи через тіло людини, електричний струм надає на нього складну дію, що є сукупністю термічної (нагрів тканин і біологічних середовищ), електролітичної (розкладання крові і плазми) і біологічної (роздратування і збудження нервових волокон і інших органів тканин організму) дій.

Ступінь ураження людини електричним струмом залежить від наступних факторів:

– значення сили струму;

– електричного опору тіла людини і тривалості протікання через нього струму;

– роду і частоти струму;

– індивідуальних властивостей людини і навколишнього середовища.

Даним проектом передбачаються наступні технічні способи і засоби, застережливі поразки людини електричним струмом:

– заземлення електроустановок;

– занулення;

– захисне відключення;

– електричне розділення сітей;

– використовування малої напруги;

– ізоляція струмоведучих частин;

– огорожа електроустановок.

Проведемо розрахунок заземлюючого пристрою.

Початкові дані для розрахунку заземлюючого пристрою:

– напруга установки, що заземляється, - 220В;

– режим нейтралу мережі - з ізольованою нейтралою;

– питомий опір ґрунту – 100 Ом·м(суглинок);

– гранично допустимий опір заземлюючого пристрою - 4 Ом;

– характеристика кліматичної зони (III):

а) середня багаторічна низька температура, °С - від –14 до -10;

б) тривалість замерзання вод, дні - 150;

в) коефіцієнт сезонності для вертикального електроду завдовжки 3м -1,5.

Визначимо розрахунковий опір ґрунту (Ом·м) по формулі (4.1).

$$\rho_{расч} = \psi \cdot \rho = 1{,}5 \cdot 100 = 150 Ом \cdot м \tag{4.1}$$

де ρ - питомий опір ґрунту;

ψ_i – кліматичний коефіцієнт, що враховує стан ґрунту під час вимірювань (таблиця 4 [42]).

Розрахуємо опір розтіканню одиночного трубчастого заземлювача по формулі (4.2).

$$R_{з.1} = (\frac{\rho_{расч}}{2 \cdot \pi \cdot l}) \cdot \ln(4 \cdot \frac{l}{d}) \tag{4.2}$$

де l – довжина заземлювача (l=5м);

d – діаметр труби і стрижня (d=0,05м);

$$R_{з.1} = (\frac{\rho_{расч}}{2 \cdot \pi \cdot l}) \cdot \ln(4 \cdot \frac{l}{d}) = (\frac{150}{2 \cdot 3{,}14 \cdot 5}) \cdot \ln(4 \cdot \frac{5}{0{,}05}) = 28{,}6 \text{ Ом}$$

Розрахуємо кількість паралельно сполучених одиночних заземлювачей по формулі (4.3).

$$n = \frac{R_{з.1}}{R_{доп} \cdot \eta} = \frac{28{,}6}{4 \cdot 0.47} = 15{,}2 \tag{4.3}$$

де Rдоп=4. – самий допустимий опір заземлюючого пристрою;

η - коефіцієнт використовування ґрунтового заземлення (для шістки заземлювачей η=0,47).

Округлятимемо отримане значення у більшу сторону n=[15,2]=16.

Розрахуємо довжину горизонтальної сполучної смуги по формулі (4.4).

$$L = a \cdot (n-1) = 3 \cdot (16-1) = 45 м \tag{4.4}$$

де a – відстань між вертикальними заземлювачами (a=3м);

n – кількість вертикальних заземлювачей (n=16).

Розрахуємо опір сполучної смуги по формулі (4.5).

$$R_n = \frac{\rho_{расч}}{2 \cdot \pi \cdot l} \cdot \ln(\frac{L^2}{d \cdot h}) \tag{4.5}$$

де d – еквівалентний діаметр смуги шириною l=5 (d=0,05м);

h – глибина заставляння смуги (h=0,8м).

$$R_n = \frac{\rho_{расч}}{2 \cdot \pi \cdot l} \cdot \ln(\frac{L^2}{d \cdot h}) = \frac{150}{2 \cdot 3{,}14 \cdot 5} \cdot \ln(\frac{45^2}{0{,}05 \cdot 0{,}8}) = 51{,}7 \text{ Ом}$$

Розрахуємо результуючий опір заземлюючого електроду з урахуванням сполучної смуги по формулі (4.6).

$$R_{гр} = \frac{R_{з.1} \cdot R_n}{R_{з.1} \cdot \eta_n + R_n \cdot n \cdot \eta_з} \le R_{доп} \tag{4.6}$$

де η_n – коефіцієнт використовування сполучної смуги (для 6-і заземлювачей $\eta_{n=0,27}$).

$$R_{гр} = \frac{R_{з.1} \cdot R_n}{R_{з.1} \cdot \eta_n + R_n \cdot n \cdot \eta_з} = \frac{26{,}6 \cdot 51{,}7}{26{,}6 \cdot 0{,}27 + 51{,}7 \cdot 16 \cdot 0{,}47} = 3{,}47 \text{ Ом}$$

3,47<4 $\Rightarrow$ умова забезпечення електробезпеки персоналу виконується.

Таким чином, остаточна кількість заземлювачей 15 шт.

4.3 Заходи, що забезпечують виробничу санітарію і гігієну праці

Підвищення працездатності людини і збереження його здоров'я забезпечується стабільними метеорологічними умовами.

Мікроклімат виробничих приміщень – це поєднання температури, вогкості і швидкості руху повітря, а також температури навколишніх поверхонь. Значне коливання параметрів мікроклімату приводить до порушення систем кровообігу, нервової і пітовидільної, що може викликати підвищення або пониження температури тіла, слабкість, запаморочення і навіть непритомність.

В приміщенні для виконання робіт операторського типу, пов'язаних з нервово-емоційною напругою, проектом передбачається дотримання наступних нормованих величин параметрів мікроклімату (див. таблицю 5.1).

Таблиця 4.1 - Оптимальні параметри мікроклімату в робочій зоні виробничого приміщення для категорії робіт 1

Період року	Температура, оС	Відносна вогкість %	Швидкість руху повітря, м/с
Холодний	22.24	40.60	0,1
Теплий	23.25	40.60	0,1

Оскільки в приміщенні немає джерел виділення шкідливих речовин, можна використовувати природну вентиляцію. Площа приміщення складає 32 $м^2$. Для забезпечення прийнятних параметрів мікроклімату в приміщенні з такою площею можна використовувати 1 кондиціонер типу БК-2000.

Спектр випромінювання монітора комп'ютера включає рентгенівську, ультрафіолетову, інфрачервону області, а також широкий діапазон хвиль інших частот. Небезпека рентгенівського проміння нехтує мала, оскільки цей вид випромінювання поглинається речовиною екрану.

Для зниження дії електромагнітного випромінювання пропонується захист часом і відстанню. Захист часом передбачає обмеження часу перебування людини в зоні дії полів. Тривалість роботи на ПЕОМ повинна складати не більше 3.5–4.5 години.

Також необхідно забезпечити раціональне освітлення в робочому приміщенні. В проекті, що розробляється, передбачається використовувати суміщене освітлення. В

світлий час доби приміщення освітлюватиметься через віконні отвори, в решту часу використовуватиметься штучне освітлення.

Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла в світильниках загального освітлення, оскільки люмінесцентні лампи володіють високою світловою віддачею до 75 Лам/Вт і більш, тривалим терміном служби до 10000 годин, спектральним складом випромінюваного світла, близьким до сонячного.

Зорова робота оператора ПЕВМ відповідно до ДБН В.2.5-28-2018 [45] відноситься до розряду Va з світловим потоком Фл=3120 кожна. Нормована освітленість на робочому місці (Ен) при загальному освітленні складає 200 лк.

Проведемо розрахунок кількості світильників в робочому приміщенні завдовжки a=6 м, шириною b=3 м, заввишки c=4 м. Формула розрахунку штучного освітлення при горизонтальній робочій поверхні методом світлового потоку (4.7):

$$Фл = \frac{Ен \cdot S \cdot Z \cdot K}{N \cdot U \cdot M} \tag{4.7}$$

де *Фл* – світловий потік, Лм;

Ен – нормована освітленість;

S – площа підлоги, кв.м;

Z=1.1–1.3 - поправочний коефіцієнт світильника (для стандартних світильників);

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації світильників;

N – число світильників;

U=0.55–0.6 – коефіцієнт використовування, залежний від типу світильника, показника індексу приміщення і др.;

M – число ламп в світильнику.

З формули (4.7) виразимо N і визначимо кількість світильників для даного приміщення:

$$N = \frac{Ен \cdot S \cdot Z \cdot K}{Фл \cdot U \cdot M} \tag{4.7}$$

$$N = \frac{200 \cdot 18 \cdot 1{,}2 \cdot 1{,}5}{3120 \cdot 0{,}6 \cdot 2} = 1{,}7$$

Виходячи з цього, рекомендується використовувати 2 світильники. Світильники слід розміщувати рядами, бажано паралельно стіні з вікнами. Схема розташування світильників зображена на рис. (4.1).

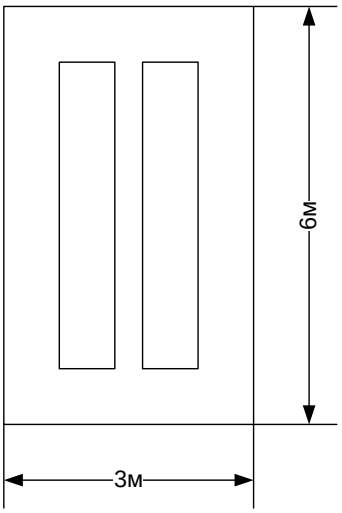


Рисунок 4.1 – Схема розташування світильників

Зниження шуму можна добитися раціонально розпланувавши приміщення, установкою устаткування на спеціальні амортизуючи прокладки. Згідно вимогам ДСН 3.3.6.037-99 [46] рівні звуку не повинні перевищувати 50 дБ.

Для зниження стомлюваності обслуговуючого персоналу в приміщеннях, де розташовані обчислювальні засоби передбачаються використовувати спокійні колірні поєднання і покриття, що не дають відблисків. Від електромагнітного випромінювання, витікаючого від ПЕОМ, використовуються захисні екрани.

Для забезпечення чистоти повітря і відповідних мікрокліматичних умов пропонується застосувати приточування-витяжну вентиляцію. Для зменшення дії шкідливих речовин і загазованості для роботи з розплавленими матеріалами робоче місце забезпечується примусовою витяжною вентиляцією. Цей метод забезпечує притоку потрібної кількості свіжого повітря (30 мм^3 /ч на одного працюючого).

Кількість повітря, яка необхідна подавати в приміщення для забезпечення необхідних параметрів повітряного середовища, визначається на підставі кількість тепла, вологи і шкідливих речовин, що поступають в приміщення, а також враховуючи видалення повітря місцевими відсмоктуваннями від устаткування, загальнообмінною вентиляцією.

4.4 Рекомендації по пожежній профілактиці

Пожежі представляють небезпеку для життя людини і зв'язані як з матеріальними втратами, так і з відмовою засобів обчислювальної техніки, що спричиняє за собою порушення ходу технологічного процесу.

Горючими матеріалами в приміщенні, де розташовані ПЕОМ, є:

– поліамід - матеріал корпусу мікросхеми. Горюча речовина. Температура самозапалення 420 ^{o}С, енергія запалення 2мДж;

– полівінілхлорид - ізоляційний матеріал. Горюча речовина. Температура самозаймання 480 єС, енергія запалення 50мДж;

– склостоліт ДЦ - матеріал друкарської платні. Складногорючий матеріал;

– пластикат кабельний No.489 - матеріал ізоляції кабелю. Складногорючий матеріал. Температура самозаймання 1500 єС;

– плита деревостружкова - будівельний і обробний матеріал, матеріал з якого виготовлені меблі. Складнозапалений матеріал. Показник горючості 1.8;

– папір – довідкова і робоча документація, література. Горючий матеріал. Показник горючості більше 2.1.

Відповідно до ДСТУ Б В.1.1-36:2016 [47] приміщення відноситься до категорії В (пожежовибухонебезпечної).

Джерелами запалення можуть бути:

– іскри при замиканні і розмиканні ланцюгів;

– іскри і дуги коротких замикань;

– перегріви від тривалого перевантаження і наявності перехідного опору;

– розряди статичної електрики.

Для того, щоб зупинити реакцію горіння, порушують умови її виникнення і підтримки. Звичайно для гасіння використовуються порушення двох основних умов сталого стану – пониження температури і режим руху газів. Пониження температури може бути досягнутий шляхом введення речовин, які поглинають багато тепла в результаті випаровування і дисоціації (наприклад, вода, порошки).

При повному тому, що згоряє органічних сполук утворюються С, SO, Н Про, N, а при тому, що згоряє неорганічних з'єднань – оксиди. Залежно від температури плавлення і тривалості реакції можуть знаходиться або у вигляді розплавів (Al O, Ti O), або підійматися в повітря у вигляді диму (P O, Na Про, MgO).

Склад продуктів неповного згоряє горючих речовин складений і різноманітний. Це можуть бути горючі речовини:

– Н, З, СН;

– атомарний водень і кисень;

– різні радикали – ВІН, СН .

Продуктами неповного згоряє можуть бути також оксиди азоту, спирти, альдегіди, кетони і високотоксичні з'єднання, наприклад, синильна кислота.

Для захисту персоналу від дій небезпечних і шкідливих чинників пожежі проектом передбачено застосування промислового фільтруючого протигаза з коробкою марки В (жовтий).

До системи запобігання пожежі відносяться: запобігання утворення горючого середовища і освіти в горючому середовищі джерел запалення, забезпечення пожежебеспеки устаткування.

Щоб запобігти пожежі в обчислювальних центрах, проектом пропонується виконання наступних вимог:

– електроживлення ЕОМ має автоматичне блокування відключення електроенергії на випадок перегріву системи, що може бути результатом зупинки системи охолоджування і кондиціонування;

– система вентиляції обчислювальних центрів обладнується блокуючими пристроями, що забезпечують її відключення на випадок пожежі. Система обладнується вогнеперегороджуючими клапанами;

– застосування устаткування, що задовольняє вимогам електростатичної іскробезпеки [42];

– після закінчення роботи, перед закриттям приміщення, всі електроустановки і персональні комп'ютери відключаються від сіті електроживлення;

– в приміщеннях обчислювальних центрів забороняється:

1) влаштовувати електророзетки на основах, що згоряють;
2) використовувати синтетичні доріжки і килими;
3) користуватися побутовими електронагрівальними приладами;
4) захаращувати евакуаційні виходи і проходи;
5) влаштовувати на вікнах глухі грати;
6) залишати без нагляду включену в електромережу апаратуру, що використовується для вимірювань і нагляду.

Для протипожежного захисту проектом пропонується обладнати приміщення площею 18 м^2, яке відноситься до категорії В, автоматичною протипожежною

сигналізацією із застосуванням датчиків сповіщення РІД-1 (оповіщувач димовий іонізаційний) в кількості 1 штуки і застосовується в первинних засобах пожежегасінні. Площа контрольована оповіщувачем 150 м^2.

Крім того, необхідно проводити навчання робочого персоналу правилам пожежної безпеки.

Розрахуємо вірогідність виникнення пожежі у виробничому приміщенні у разі запалювання транзистора:

$$Q = л \cdot T \cdot Ркз/отк \cdot Qвоспл \cdot Рзащ \quad (4.8)$$

де л – інтенсивність відмов пожежеопасних ЕРІ;

Т – час роботи пожежеопасного ЕРІ за оцінюваний інтервал часу;

Ркз/отк - умовна вірогідність виходу ЕРІ в стан короткого замикання при його відмові;

Qвоспл - вірогідність запалювання ЕРІ, що знаходиться в стані короткого замикання;

Рзащ – вірогідність відмови захисту пожежеопасного ЕРІ. Якщо захист відсутній, Рзащ приймається рівній 1.

Вірогідність виникнення пожежі у разі запалювання транзистора:

$$Q = 1 \cdot 10^{-6} \cdot 1 \cdot 10^{-4} \cdot 0.1 \cdot 1 \cdot 10^{-4} = 1 \cdot 10^{-15}$$

Розрахована вірогідність виникнення пожежі значно менше допустимої, яка складає $1 \cdot 10^{-6}$.

В даному розділі були проаналізовані небезпечні і шкідливі виробничі чинники, що роблять вплив на персонал, розроблені заходи щодо техніки безпеки, заходу, забезпечуючи виробничу санітарію і гігієну праці, а також заходи щодо пожежної профілактики.

4.5 Вплив на навколишньє природнє середовище

Діяльність за темою магістерської роботи в процесі її виконання впливає на навколишнє природне середовище і регламентується нормами діючого законодавства [48 - 50].

Основним екологічним аспектом в процесі діяльності за даними спеціальностями є процеси впливу на атмосферне повітря та процеси поводження з відходами, які утворюються, збираються, розміщуються, передаються на видалення (знешкодження), утилізацію, тощо в ІТ галузі.

В приміщенні немає впливу на атмосферне повітря при нормальних умовах праці, бо не використовуються сканери, принтери та інші джерела викиду забруднюючих речовин в повітря робочої зони.

В процесі діяльності користувача виникають процеси поводження з відходами ІТ галузі. Види відходів, утворення, яких можливо:

- відпрацьовані люмінесцентні лампи - I клас небезпеки;
- батарейки та акумулятори (малі) -III клас небезпеки;
- змінні носії інформації - IV клас небезпеки;
- відпрацьований ізолюючий матеріал, дроти та кабелі - IV клас небезпеки;
- макулатура - IV клас небезпеки;
- побутові відходи - IV клас небезпеки.

ВИСНОВКИ

У рамках магістерської роботи були проаналізовані усі етапи розпізнавання символі з зображення і автоматичного перекладу текстів. На основі аналізу був реалізований додаток, що поєднує в собі оптичне розпізнавання символів і подальший машинний переклад тексту.

Зокрема, був проведений огляд існуючих методологій нейронних мереж і їх застосовності для вирішення завдань оптичного розпізнавання символів.

Після повного обзору підходів до машинного перекладу текстів, був обраний комбінований спосіб на основі прикладів перекладеного тексту, та на основі тематичних словників. Комбінування цих 2 методів дозволяє досягати високою точності перекладу та високої швидкості роботи методу.

Була запропонована архітектура модульної системи розпізнавання тексту, в якій кожен етап розпізнавання реалізується окремим компонентом, а зв'язок між компонентами (у тому числі – зворотний зв'язок) здійснюється програмою.

Як приклад роботи створений трьох модульний додаток, що має модуль покращення якості зображення, модуль оптичного розпізнавання символі з зображення, та модуль автоматичного перекладу за допомогою машинного навчання. Додаток показує високу якість оптичного розпізнавання тексту на зображеннях високої якості. Автоматичний переклад тексту показує якість на рівні професійних перекладачів.

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливи та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом написаному в кваліфікаційній роботі, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника. Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки, безпеки впливу на навколишнє природнє середовище. Була наведена схема, розміри приміщення та наведено значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп'ютері.

Результати атестаційної роботи апробовано у вигляді тез доповіді на міжнародній конференції «Майбутній науковець 2020».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1) Цифрова обробка сигналів в телекомунікаційних системах: підручник/ Г. Г. Бортник, В. М. Кичак. – Вінниця : ВНТУ, 2014. – 232с.

2) Цифрова обробка аудіо- та відеоінформації у мультимедійних системах: Навчальний посібник / О.В. Дробик, В.В. Кідалов, В.В. Коваль, Б.Я. Костік, В.С. Лазебний, Г.М. Розорінов, Г.О. Сукач. – К.: Наукова думка, 2008. – 144 с

3) Обробка сигналів: Підручник / Бабак В.П., Хандецький В.С., Шрюфер Е. – К.: Либідь, 1996. – 392 с.

4) Slusar, K. (2018). The boundaries of the concepts of international journalism. Век Информации (Сетевое Издание), 2(4(5).

5) Liu, Z., Wang, L., & Hua, G. (2018). Joint Video Object Discovery and Segmentation by Coupled Dynamic Markov Networks. IEEE Transactions on Image Processing, 27(12), 5840–5853.

6) Duan, X., Wang, L., & Zhai, C. (2018). Joint Spatio-Temporal Action Localization in Untrimmed Videos with Per-Frame Segmentation. 2018 25th IEEE International Conference on Image Processing (ICIP).

7) Barghout, L., & Sheynin, J. (2013). Real-world scene perception and perceptual organization: Lessons from Computer Vision. Journal of Vision, 13(9), 709–709.

8) Aguilar, C., & Martínez, G. S. (2017). Preface: Recent Advances in Natural Language Processing. CLEI Electronic Journal.

9) Calixto, I., & Liu, Q. (2017). Sentence-Level Multilingual Multi-modal Embedding for Natural Language Processing. RANLP 2017 - Recent Advances in Natural Language Processing Meet Deep Learning.

10) Guillén, A., Gutiérrez, Y., & Muñoz, R. (2017). Natural Language Processing Technologies for Document Profiling. RANLP 2017 - Recent Advances in Natural Language Processing Meet Deep Learning.

11) Klette, R. (2014). Concise Computer Vision. Undergraduate Topics in Computer Science.

12) Zampieri, M., & Amorim, R. C. D. (2014). Between Sound and Spelling: Combining Phonetics and Clustering Algorithms to Improve Target Word Recovery. Advances in Natural Language Processing Lecture Notes in Computer Science, 438–449.

13) Almiman, A., & Ramsay, A. (2017). Using English Dictionaries to generate Commonsense Knowledge in Natural Language. RANLP 2017 - Recent Advances in Natural Language Processing Meet Deep Learning.

14) Boros, T., & Dumitrescu, S. D. (2017). Fast and Accurate Decision Trees for Natural Language Processing Tasks. RANLP 2017 - Recent Advances in Natural Language Processing Meet Deep Learning.

15) Вовк С.М., Гнатушенко В.В., Бондаренко М.В. Методи обробки зображень та компютерний зір : навч. посіб. / С.М. Вовк, В.В. Гнатушенко, М.В. Бондаренко. – Д. : ЛІРА, 2016. – 148 с.

16) Путятін Є. П., Гороховатській В.О., Матат О.О. Методи та алгоритми комп'ютерного зору: Навч. посібник. Х: СМІТ, 2006. 236 с..

17) Pratt W.K. 2016. Digital Images Processing. Third edition. Wiley.

18) Bhootra, A. (2014). Laws of Computer Vision. Basics of Computer Vision Syndrome, 19–19.

19) Reid, I. (2016). 12th Asian conference on computer vision. Computer Vision and Image Understanding, 146, 51.

20) Prince, S. J. D. (2015). Machine learning for machine vision. Computer Vision, 53–54.

21) Weinmann, M. (2013). Visual Features—From Early Concepts to Modern Computer Vision. Advanced Topics in Computer Vision, 1–34.

22) Parker J.R. 2010. Algorithms for Image Processing and Computer Vision. Second Edition. Wiley Publishing, Inc.

23) Woods J.W. Multidimensional signal, image, and video processing and coding. – New York: Academic Press, 2006. – 494 p.

24) Klette, R. (2014). Image Processing. Undergraduate Topics in Computer Science Concise Computer Vision, 43–87.

25) Delmerico, J. A., David, P., & Corso, J. J. (2011). Building facade detection, segmentation, and parameter estimation for mobile robot localization and guidance. 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems.

26) Morrissey, S. (2009). Yorick Wilks, Machine Translation: Its Scope and Limits. Machine Translation, 23(4), 265–267.

27) Specia, L., Raj, D., & Turchi, M. (2010). Machine translation evaluation versus quality estimation. Machine Translation, 24(1), 39–50.

28) Doherty, S. (2019). Translation technology evaluation research. The Routledge Handbook of Translation and Technology, 339–353.

29) Chan, S.-W. (2014). Routledge Encyclopedia of Translation Technology.

30) Junker, M., & Hoch, R. (2014). Evaluating OCR and non-OCR text representations for learning document classifiers. Proceedings of the Fourth International Conference on Document Analysis and Recognition.

31) Image Spam Filters Based on Optical Character Recognition (OCR) Techniques. (2018). Advanced Image-Based Spam Detection and Filtering Techniques Advances in Information Security, Privacy, and Ethics, 90–108.

32) Review OCR findings involving web accessibility. (2018). Campus Legal Advisor, 18(8), 16–16.

33) Rosenfeld, A. (1984). Picture processing: 1983. Computer Vision, Graphics, and Image Processing, 25(3), 400.

34) Deptuch, G. (2001). Monolithic active pixel sensors. AIP Conference Proceedings.

35) Творошенко, І. С. (2015). Конспект лекцій з дисципліни «Цифрова обробка зображень» (для студентів 5 курсу денної та заочної форм навчання спеціальності 7.08010105–Геоінформаційні системи та технології).

36) William K. Pratt Digital image processing/ Third Edition/ John Wiley & Sons, Inc. – 2001. – 723 p.

37) Федотов Н.Г. Методы стохастической геометрии в распознавании образов- М.:Радио и связь, 1990.- 144 с.

38) Крылов В.Н., Максимов М.В. Вторичные преобразователи сигналов и изображений. - Одесса, Астропринт, 1997. - 176 с

39) Ту Дж., Гонсалес Р. Принципы распознавания образов.- М.:Мир, 1998.-411 с.

40) Гонсалес Р., Вудс Р. Цифровая обработка изображений/ Пер. с англ. – М.: Техносфера, 2005. – 1072 с.

41) ДСН 3.3.6.042-99 Державні санітарні норми мікроклімату виробничих. Режим доступу: https://dnaop.com/html/34094/doc-ДСН_3.3.6.042-99

42) ДСТУ Б А.3.2-13:2011Система стандартів безпеки праці. Будівництво. Електро безпечність. Загальні вимоги. Режим доступу- : http://online.budstandart.com/ru/catalog/doc-page?id_doc=27973

43) ДСанПіН 3.3.2-007-98 Державні санітарні правила і норми. Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин Режим Доступу- : https://zakon.rada.gov.ua/rada/show/v0007282-98

44) 4. НПАОП 0.00-7.15-18 Вимоги щодо безпеки та захисту здоров`я працівників під час роботи з екранними пристроями Міністерство доходів і зборів України Наказ від 05.09.2013 р. № 443 "Про затвердження Примірної інструкції з охорони праці під час експлуатації електронно-обчислювальних машин") Режим доступу- : https://zakon.rada.gov.ua/laws/show/%D0%BD%D0%BF%D0%B0%D0%BE%D0%BF_0.00-7.15-18

45) ДБН В.2.5-28-2018. Природне і штучнеосвітлення. Режим доступу - : https://okna.ua/img_all/oknaua/dbn-V-2-5-28-2018-ed.pdf

46) ДСН 3.3.6.037-99 Державні санітарні норми виробничого шуму, ультразвуку та інфразвуку. Режим Доступу-: https://zakon.rada.gov.ua/rada/show/va037282-99

47) ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою». Наказ від 15.06.2016 №158. Режим доступу: www. URL: https://dbn.co.ua/load/normativy/dstu/dstu_b_v_1_1_36/5-1-0-1759

48) Закон України «Про охорону навколишнього природного середовища». Вводиться в дію Постановою ВР № 1268-XII від 26.06.91, ВВР, 1991, № 41 Режим доступу: www.URL: https://zakon.rada.gov.ua/laws/show/1264-12

49) Закон України «Про забезпечення санітарного та епідемічного благополуччя населення». Вводиться в дію Постановою ВР № 4005-XII від 24.02.94, ВВР, 1994, № 27 Режим доступу: https://zakon.rada.gov.ua/laws/show/4004-12

50) Закон України «Про відходи». Відомості Верховної Ради України (ВВР), 1998, № 36-37, ст.242. Режим доступу: www.URL: https://zakon.rada.gov.ua/laws/show/187/98-%D0%B2%D1%80

ДОДАТОК А.

Фрагмент лістингу модуля «OCRTask.java»

```
package de.vorb.tesseract.gui.work;

import de.vorb.tesseract.gui.io.PlainTextWriter;
import de.vorb.tesseract.gui.model.BatchExportModel;
import de.vorb.tesseract.gui.model.ProjectModel;
import de.vorb.tesseract.gui.util.DocumentWriter;
import de.vorb.tesseract.tools.preprocessing.Preprocessor;
import de.vorb.tesseract.tools.recognition.PageRecognitionConsumer;
import de.vorb.tesseract.util.Block;
import de.vorb.tesseract.util.Page;
import de.vorb.util.FileNames;

import eu.digitisation.input.Batch;
import eu.digitisation.input.Parameters;
import eu.digitisation.output.Report;

import javax.imageio.ImageIO;
import javax.swing.ProgressMonitor;
import java.awt.image.BufferedImage;
import java.io.BufferedOutputStream;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.Writer;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.StandardCopyOption;
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.Callable;
import java.util.concurrent.LinkedBlockingQueue;
import java.util.concurrent.atomic.AtomicInteger;

public class OCRTask implements Callable<Void> {
    private final Path sourceFile;
    private final ProjectModel project;
    private final BatchExportModel export;
    private final Preprocessor preprocessor;
    private final LinkedBlockingQueue<PageRecognitionProducer> recognizers;
    private final boolean hasPreprocessorChanged;
    private final Path equivalencesFile;

    private final ProgressMonitor progressMonitor;
    private final AtomicInteger progress;

    private final Writer errorLog;
    private final AtomicInteger errors;

    public OCRTask(Path sourceFile, ProjectModel project,
            BatchExportModel export, Preprocessor preprocessor,
            LinkedBlockingQueue<PageRecognitionProducer> recognizers,
            boolean hasPreprocessorChanged, Path equivalencesFile,
            ProgressMonitor progressMonitor, AtomicInteger progress,
            Writer errorLog, AtomicInteger errors) {
        this.sourceFile = sourceFile;
        this.project = project;
        this.export = export;
        this.preprocessor = preprocessor;
        this.recognizers = recognizers;
```

```
    this.hasPreprocessorChanged = hasPreprocessorChanged;
    this.equivalencesFile = equivalencesFile;

    this.progressMonitor = progressMonitor;
    this.progress = progress;

    this.errorLog = errorLog;
    this.errors = errors;
}

@Override
public Void call() throws IOException {
    if (progressMonitor.isCanceled()) {
        return null;
    }

    progressMonitor.setNote(sourceFile.getFileName().toString());

    final Path imgDestFile = project.getPreprocessedDir().resolve(
            FileNames.replaceExtension(sourceFile, "png").getFileName());

    try {

        final int width;
        final int height;
        {
            final BufferedImage sourceImg =
                    ImageIO.read(sourceFile.toFile());

            width = sourceImg.getWidth();
            height = sourceImg.getHeight();

            final BufferedImage binaryImg;
            if (!hasPreprocessorChanged && Files.exists(imgDestFile)) {
                // read existing preprocessed image
                binaryImg = ImageIO.read(imgDestFile.toFile());
            } else {
                // pre-process source image
                binaryImg = preprocessor.process(sourceImg);

                ImageIO.write(binaryImg, "PNG", imgDestFile.toFile());
            }

            if (export.exportImages()) {
                Files.copy(
                        imgDestFile,
                        export.getDestinationDir().resolve(
                                imgDestFile.getFileName()),
                        StandardCopyOption.REPLACE_EXISTING);
            }
        }

        if (Thread.currentThread().isInterrupted()) {
            return null;
        }

        final PageRecognitionProducer recognizer = recognizers.take();
        if (recognizer == null) {
            throw new IllegalStateException(
                    "No more PageRecognitionProducers");
        }

        try {
            recognizer.reset();
            recognizer.loadImage(imgDestFile);
```

```
                final List<Block> blocks = new ArrayList<>();

                recognizer.recognize(new PageRecognitionConsumer(blocks) {
                    @Override
                    public boolean isCancelled() {
                        return Thread.currentThread().isInterrupted();
                    }
                });

                final Page page = new Page(sourceFile, width, height, 300,
                        blocks);

                if (export.exportXML()) {
                    final Path xmlFile = export.getDestinationDir().resolve(
                            FileNames.replaceExtension(sourceFile,
"xml").getFileName());

                    try (BufferedOutputStream out = new
BufferedOutputStream(Files.newOutputStream(xmlFile))) {
                        page.writeTo(out);
                    }
                }

                if (export.exportHTML()) {
                }

                // write the ocr result as text
                final Path txtFile = project.getOCRDir().resolve(
                        FileNames.replaceExtension(sourceFile,
"txt").getFileName());
                {
                    final Writer out = Files.newBufferedWriter(txtFile,
                            StandardCharsets.UTF_8);

                    new PlainTextWriter(true).write(page, out);

                    out.close();

                    // if text files are exported, copy it over
                    if (export.exportTXT()) {
                        Files.copy(
                                txtFile,
                                export.getDestinationDir().resolve(
                                        txtFile.getFileName()),
                                StandardCopyOption.REPLACE_EXISTING);
                    }
                }

                if (export.exportReports()) {
                    final Path transcriptionFile =
                            project.getTranscriptionDir().resolve(
                                    txtFile.getFileName());

                    if (Files.isRegularFile(transcriptionFile)) {
                        // generate report
                        final Batch reportBatch = new Batch(
                                transcriptionFile.toFile(),
txtFile.toFile());
                        final Parameters pars = new Parameters();
                        pars.eqfile.setValue(equivalencesFile.toFile());
                        Report rep;
                        rep = new Report(reportBatch, pars);

                        // write to file
```

```
                        DocumentWriter.writeToFile(
                                rep.document(),
                                export.getDestinationDir().resolve(
                                        FileNames.replaceExtension(
                                                sourceFile,
"report.html").getFileName()));

                        // write csv file
                        final BufferedWriter csv = Files.newBufferedWriter(
                                export.getDestinationDir().resolve(
                                        FileNames.replaceExtension(
                                                sourceFile,
"report.csv").getFileName()),
                                StandardCharsets.UTF_8);

                        csv.write(rep.getStats().asCSV("\n",
",").toString());
                        csv.close();
                    }
                }
            } finally {
                recognizers.put(recognizer);

                // update progress
                progressMonitor.setProgress(progress.incrementAndGet());
            }
        } catch (Throwable e) {
            errorLog.write(e.getMessage());
            errorLog.write(System.lineSeparator());

            errors.incrementAndGet();
        }

        return null;
    }
}
```

ДОДАТОК Б.

Фрагмент лістингу модуля «TextTranslator.java»

```
package com.texttranslator.core;

import java.net.URLEncoder;
import java.util.ArrayList;
import java.util.Iterator;

import org.json.JSONObject;

import com.texttranslator.helper.ServerCom;

import android.util.Log;


public class TextTranslator {
     private static final String TAG = "TextCorrector";

     private final String URL_TRANSLATION =
               "http://api.mymemory.translated.net/get?q=%s&langpair=%s|%s";

     public static final int MAX_TRANS_LEN = 30; // Maximum string length
submitted to translation
                                                      // (to prevent big text
translations)

     private String langOri,
                    langDes,
                    textOriginal,
                    textTranslated;

     private ServerCom serverCom;

     public TextTranslator() {
          serverCom = new ServerCom();
          setTranslationOption(1);
     }

     /**
      * Set language option
      *
      * @param option Option of translation
      */
     public void setTranslationOption(int option) {
          switch(option) {
               case 1:
                    langOri = ExtractText.LANG_SUPORTED[1];
                    langDes = ExtractText.LANG_SUPORTED[0];
                    break;
               case 2:
                    langOri = ExtractText.LANG_SUPORTED[0];
                    langDes = ExtractText.LANG_SUPORTED[1];
                    break;
               default:
                    langOri = "auto";
                    langDes = "auto";
          }
          textTranslated = "";

          Log.i(TAG, "setTranslationOption: Translation redefined: " + langOri
+ " to " + langDes);
```

```
    }

    /**
     * Call remote API to translate text and store result in controller
     *
     * @param detectionController Controller to store result
     * @return True, if success, or False, if not
     */
    public boolean processTransText(ProcessResult detectionController) {
        ArrayList<ProcessStorage> detectionStorage =
                detectionController.getDetectionStorage();
        Iterator<ProcessStorage> it = detectionStorage.iterator();

        Log.i(TAG, "processTransText: Translation started");

        for (; it.hasNext();) {
            ProcessStorage detected = it.next();
            if (!processTransText(detected.getTextOriginal())) {
                Log.e(TAG, "processTransText: Translation fail. Ignoring
text region.");
            }
            detected.setTextTranslated(this.textTranslated);
        }

        Log.i(TAG, "processTransText: Translation complete");

        return true;
    }

    /**
     * Translate text based in language settings
     *
     * @param text Text to translation
     * @return True, if translation success, or False, if not
     */
    public boolean processTransText(String text) {
        if (translateText(text, langOri, langDes)) {
            return true;
        } else {
            return false;
        }
    }

    /**
     * Translate text
     *
     * @param text Text to translation
     * @param langOri Language of origin in translation
     * @param langDes Language of destine in translation
     * @return True, if translation success, or False, if not
     */
    public boolean translateText(String text, String langOri, String langDes)
{
        if (text == null || text == null || text == null ||
                text.length() == 0 || text.length() > MAX_TRANS_LEN) {
            textOriginal = "";
            textTranslated = "";
            return true;
        }

        textOriginal = text;

        try {
            String url = String.format(URL_TRANSLATION,
                    URLEncoder.encode(textOriginal, "UTF-8"),
```

```
                    langOri,
                    langDes);
            String response = serverCom.sendGETRequest(url);
            JSONObject jObjectResp = new JSONObject(response);
            int respStatus = jObjectResp.getInt("responseStatus");
            if (respStatus == 200) {
                JSONObject jObjectData =
jObjectResp.getJSONObject("responseData");
                textOriginal = text;
                textTranslated = jObjectData.getString("translatedText");
            }

        } catch (Exception e) {
            Log.e(TAG, "translateText: Fail in translation API");
            e.printStackTrace();
            return false;
        }

        return true;
    }

    public String getDictOri() {
        return langOri;
    }

    public String getDictDes() {
        return langDes;
    }
}
```

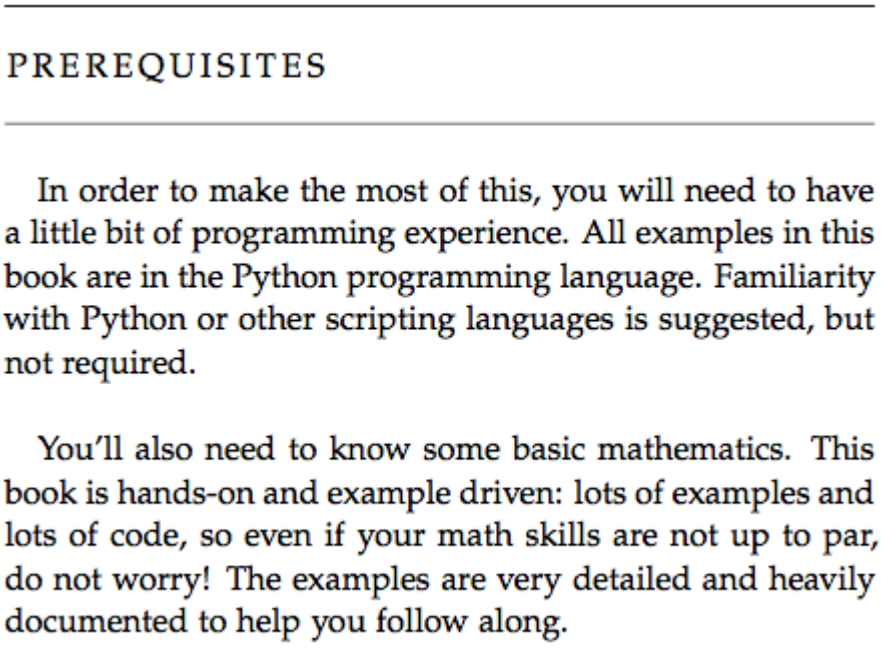
ДОДАТОК В. Фрагмент лістингу модуля «Benchmark.java»

```
package com.texttranslator.core;

import java.io.File;

import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.util.Log;

import com.texttranslator.MainActivity;

/**
 * Benchmark to save execution data
 *
 * @author luiz
 */
public class Benchmark {
     private static final String TAG = "Benchmark";

     public static final String FILE_PATH_SCAN = MainActivity.APP_PATH +
"benchmark/";

     private File folderFile;

     private ExecutionLog logger;// Logger to collect statistic data of
execution

     public Benchmark() {
          folderFile = new File(FILE_PATH_SCAN);
          logger = new ExecutionLog();
     }

     public void scanFolder(DetectText detectText,
               ExtractText extractText,
               TextTranslator textTranslator,
               ProcessResult detectionResult) {
          Bitmap imageToDetection;

          Log.i(TAG, "scanFolder: Scaning...");

          for(File file : folderFile.listFiles()){
               logger.clearLog();
               detectionResult.prepare();
               imageToDetection =
BitmapFactory.decodeFile(file.getAbsolutePath());

               if (imageToDetection != null) {
                    logger.appendLog("Detecting text...", true);
                    detectText.processDetectTextImage(imageToDetection,
detectionResult);
                    logger.appendLog("Done.", false);
               logger.appendLog("Extracting text...", true);
               extractText.processOcr(textTranslator.getDictOri(),
detectionResult);
               logger.appendLog("Done.", false);
               logger.appendLog("Translating text...", true);
               textTranslator.processTransText(detectionResult);
               logger.appendLog("Done.", false);
               logger.appendLog("Original Text: " +
detectionResult.getRecognizedText(), true);
               logger.appendLog("Translated Text: " +
detectionResult.getTranslatedText(), true);
```

```
                // Create images

                detectionResult.createTextRegionsImageBitmap();
                detectionResult.createOcrImageBitmap();
                detectionResult.createTranslatedImageBitmap();

                logger.appendLog("Done.", false);
                logger.saveLogFile(detectionResult.getCurrentDirFolder());
                }
            }

             Log.i(TAG, "scanFolder: Done.");
        }
    }
```

ДОДАТОК Г.

Тестові зображення



PREREQUISITES

In order to make the most of this, you will need to have a little bit of programming experience. All examples in this book are in the Python programming language. Familiarity with Python or other scripting languages is suggested, but not required.

You'll also need to know some basic mathematics. This book is hands-on and example driven: lots of examples and lots of code, so even if your math skills are not up to par, do not worry! The examples are very detailed and heavily documented to help you follow along.

PREREQUISITES

Рисунок А.1 – Приклад скріншоту сторінки

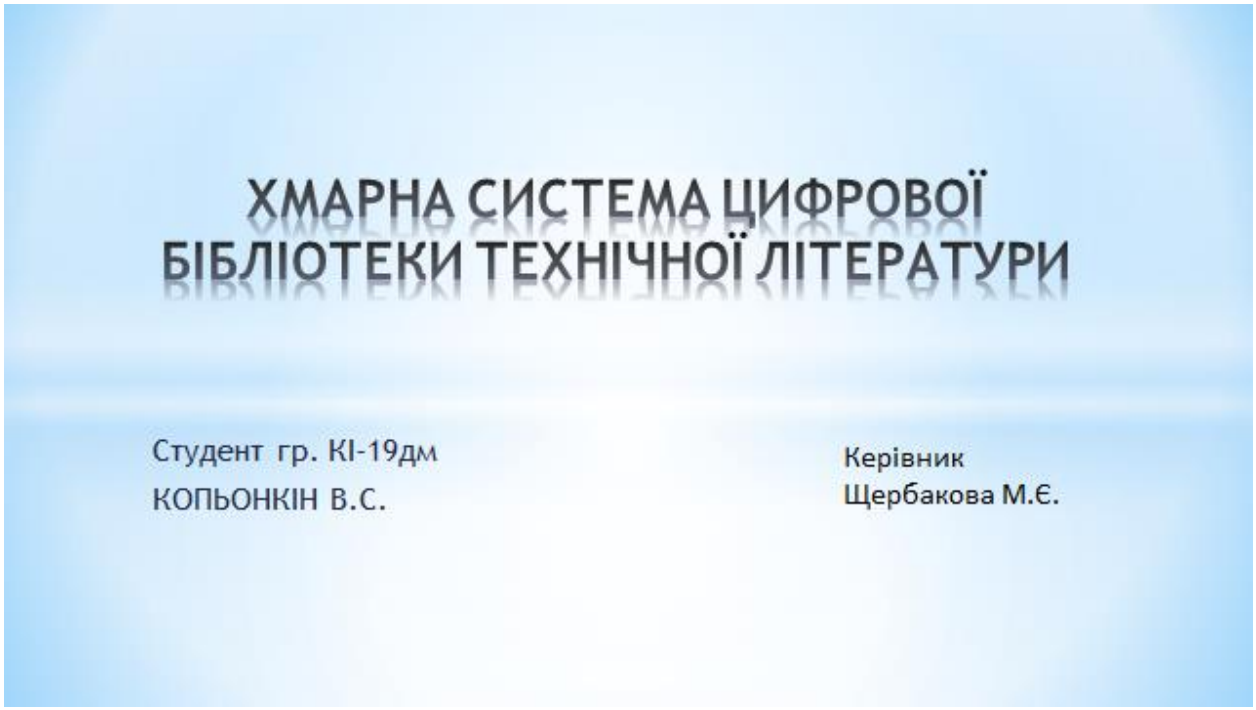
In order to make the most of this, you will need to have a little bit of programming experience. All examples in this book are in the Python programming language. Familiarity with Python or other scripting languages is suggested, but not required.

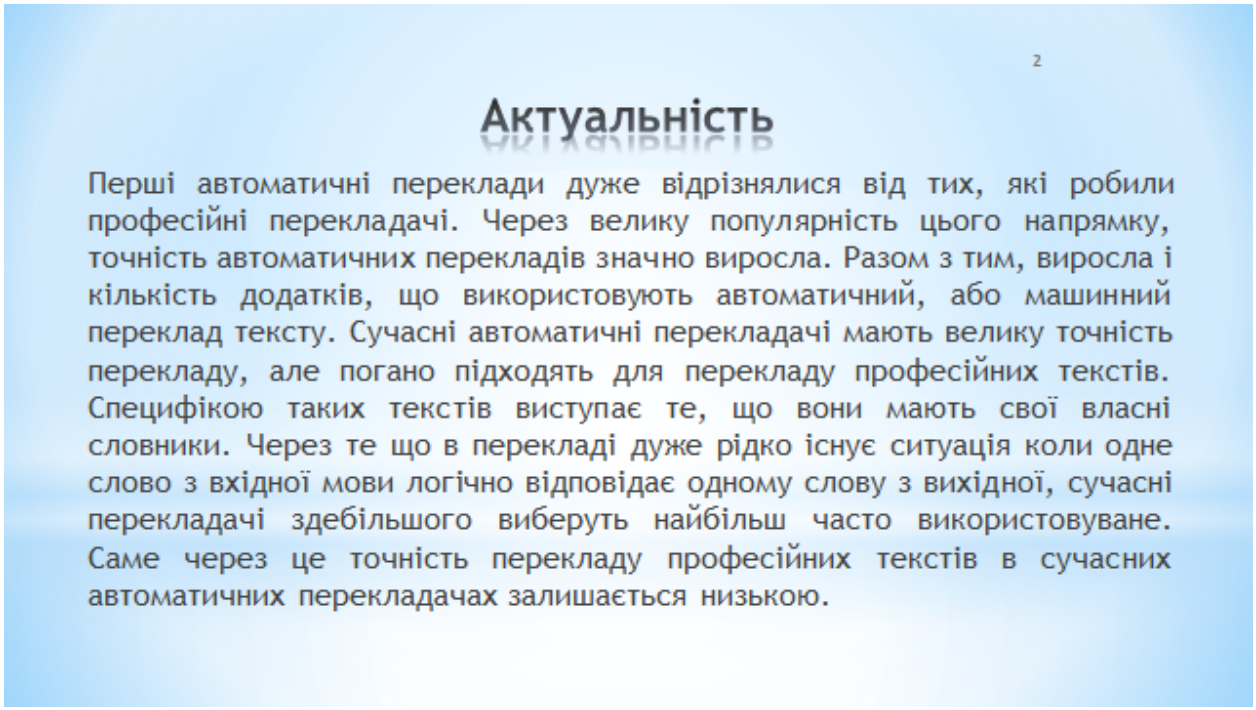
You'll also need to know some basic mathematics. This book is hands-on and example driven: lots of examples and lots of code, so even if your math skills are not up to par, do not worry! The examples are very detailed and heavily documented to help you follow along.

Рисунок А.2 – Приклад розпізнанного тексту

ДОДАТОК Д.

Електронні плакати





Постановка задачі

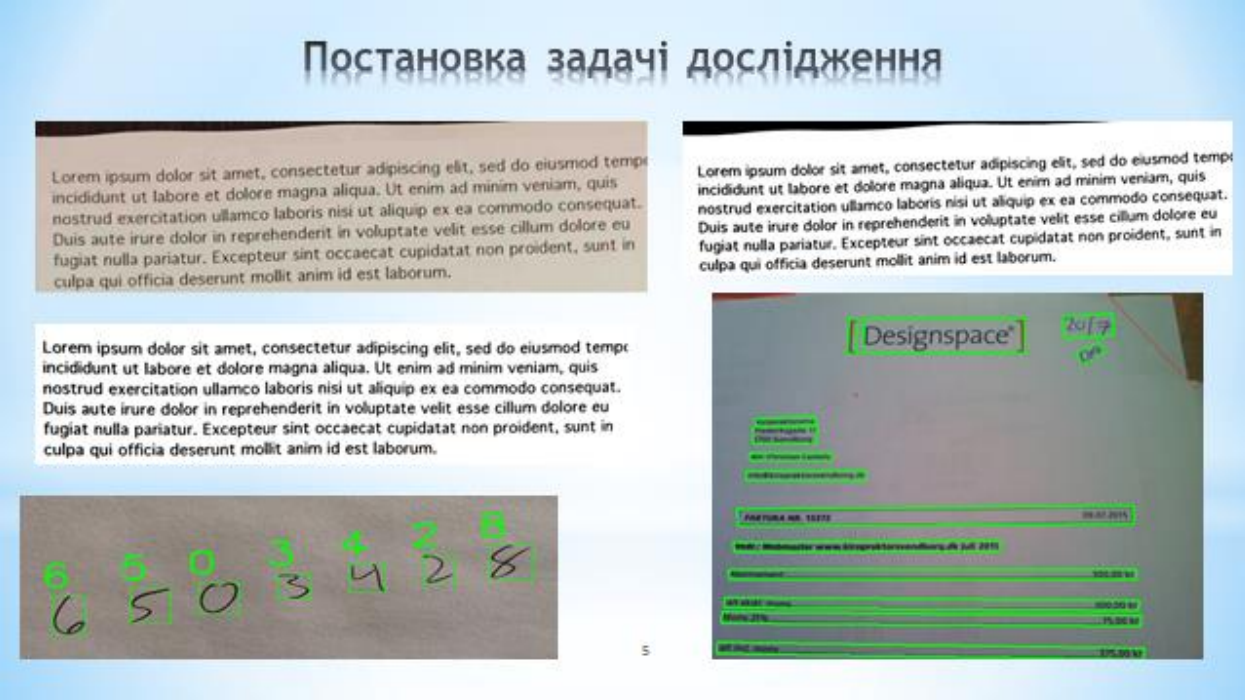
- **Метою роботи** є аналіз проблеми переносу іноземних текстів з друкованого варіанту в електронний вигляд одночасно з перекладом з первинної на цільову мову.
- **Об'єктом дослідження** виступає електронне зображення що містить іноземномовний текстовий матеріал. Це можуть бути фотографії, скани, скріншоти, або будь-які інші зображення.

3

Основні етапи вирішення задачі діджіталізації бібліотеки технічної літератури

- фільтрація зображення;
- знаходження символів на зображенні;
- оптичне розпізнавання символів;
- пост обробка зчитаного тексту;
- автоматичний переклад отриманого тексту.

4



ОГЛЯД І ПОРІВНЯННЯ МЕТОДІВ МАШИННОГО РОЗПІЗНАВАННЯ І ПЕРЕКЛАДУ ТЕКСУ

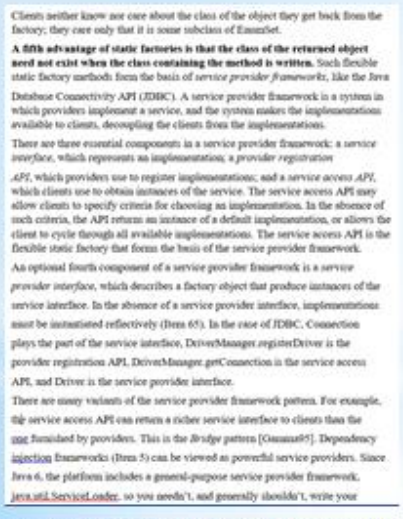
1 Методи попередньої обробки та сегментації зображень
2 Математична модель локалізації тексту
3 Аналіз методів розпізнавання образів
4 Інтенсіональні методи
5 Аналіз методів пост-обробки результату
6 Аналіз методів машинного перекладу тексту

6

КОМП'ЮТЕРНА РЕАЛІЗАЦІЯ СИСТЕМИ ДІДЖІТАЛІЗАЦІЇ БІБЛІОТЕКИ ТЕХНІЧНОЇ ЛІТЕРАТУРИ

Java™

Результат розпізнавання тексту (Застосування бібліотеки Tess4j)



Clients neither know nor care about the class of the object they get back from the factory; they care only that it is some subclass of EnumSet.
A fifth advantage of static factories is that the class of the returned object need not exist when the class containing the method is written. Such flexible static factory methods form the basis of *service provider frameworks*, like the Java Database Connectivity API (JDBC). A service provider framework is a system in which providers implement a service, and the system makes the implementations available to clients, decoupling the clients from the implementations.
There are three essential components in a service provider framework: a *service interface*, which represents an implementation; a *provider registration API*, which providers use to register implementations; and a *service access API*, which clients use to obtain instances of the service. The service access API may allow clients to specify criteria for choosing an implementation. In the absence of such criteria, the API returns an instance of a default implementation, or allows the client to cycle through all available implementations. The service access API is the flexible static factory that forms the basis of the service provider framework.
An optional fourth component of a service provider framework is a *service provider interface*, which describes a factory object that produce instances of the service interface. In the absence of a service provider interface, implementations must be instantiated reflectively (Item 65). In the case of JDBC, Connection plays the part of the service interface, DriverManager.registerDriver is the provider registration API, DriverManager.getConnection is the service access API, and Driver is the service provider interface.
There are many variants of the service provider framework pattern. For example, the service access API can return a richer service interface to clients than the one furnished by providers. This is the *Bridge* pattern [Gamma95]. Dependency injection frameworks (Item 5) can be viewed as powerful service providers. Since Java 6, the platform includes a general-purpose service provider framework, java.util.ServiceLoader, so you needn't, and generally shouldn't, write your

Скріншот сторінки професійної літератури

Clients neither know nor care about the class of the object they get back from the factory; they care only that it is some subclass of EnumSet.
A fifth advantage of static factories is that the class of the returned object need not exist when the class containing the method is written. Such flexible static factory methods form the basis of service provider frameworks, like the Java Database Connectivity API (JDBC). A service provider framework is a system in which providers implement a service, and the system makes the implementations available to clients, decoupling the clients from the implementations.
There are three essential components in a service provider framework: a service interface, which represents an implementation; a provider registration API, which providers use to register implementations; and a service access API, which clients use to obtain instances of the service. The service access API may allow clients to specify criteria for choosing an implementation. In the absence of such criteria, the API returns an instance of a default implementation, or allows the client to cycle through all available implementations. The service access API is the flexible static factory that forms the basis of the service provider framework.
An optional fourth component of a service provider framework is a service provider interface, which describes a factory object that produce instances of the service interface. In the absence of a service provider interface, implementations must be instantiated reflectively (Item 65). In the case of JDBC, Connection plays the part of the service interface, DriverManager.registerDriver is the provider registration API DriverManager.getConnection is the service access API, and Driver is the service provider interface.
There are many variants of the service provider framework pattern. For example, th service access API can return a richer service interface to clients than the one furnished by providers. This is the Bridge pattern [Gamma95]. Dependency injection frameworks (Item 5) can be viewed as powerful service providers. Since Java 6, the platform includes a general-purpose service provider framework, java.util.ServiceLoader, so you needn't, and generally shouldn't.

8 Результат розпізнавання сторінки професійної літератури

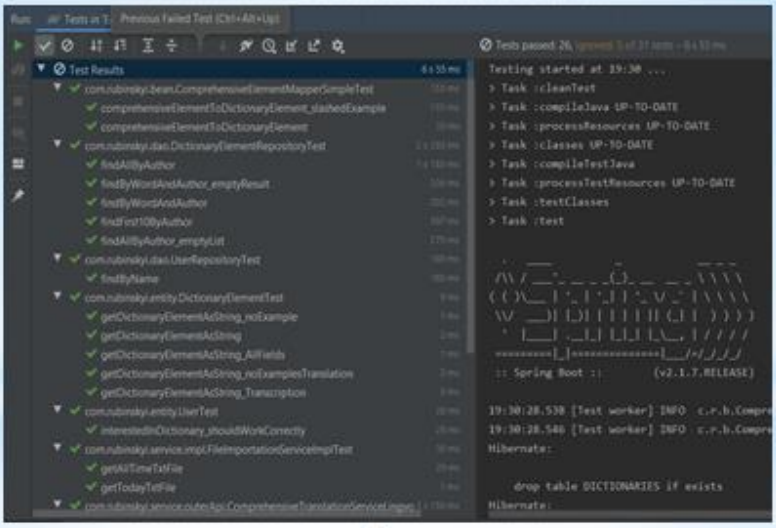
Результат машинного перекладу тексту (застосування MyMemory API)

«Accounting»,
«American English»,
«Beer»,
«Biology»,
«Building»,
«Economicus»,
«Banking»,
«Economicus Government»,
«Economicus Insurance»,
«Economicus International»,
«Electronics»,
«Engineering»,
«Financial Management»,
«Financial Markets»,
«Forum Dictionary»,
«Geography»,
«Law»,
«Learning»,
«Computer»,
«Grammar»,
«Management»,
«Marketing»,
«Mechanical Engineering»,
«Menu Translate»,
«Patents»,
«Physics»,
«Psychology»,
«Sport»,
«Telecoms»

Клиенты не знают и не заботятся о классе объекта, который они возвращают с фабрики, их волнует только то, что это некоторый подкласс EnumSet. Пятое преимущество статических фабрик состоит в том, что класс возвращаемого объекта не должен существовать, когда класс, содержащий метод, написан. Такие гибкие статические фабричные методы образуют основу каркасов поставщиков услуг, таких как API подключения к базе данных Java (JDBC). Инфраструктура поставщика услуг – это система, в которой поставщики реализуют услугу, и система делает реализации доступными для клиентов, отделяя клиентов от реализаций. В структуре поставщика услуг есть три основных компонента: интерфейс службы, представляющий реализацию; API регистрации провайдера, который провайдеры используют для регистрации реализаций; и API доступа к сервису, который клиенты используют для получения экземпляров сервиса. API доступа к сервису может позволять клиентам указывать критерии выбора реализации. При отсутствии таких критериев API возвращает экземпляр реализации по умолчанию или позволяет клиенту циклически перебирать все доступные реализации. API доступа к сервису – это гибкая статическая фабрика, которая лежит в основе структуры поставщика услуг. Необязательным четвертым компонентом структуры поставщика услуг является интерфейс поставщика услуг, который описывает фабричный объект, который создает экземпляры интерфейса сервиса. В отсутствие интерфейса поставщика услуг реализации должны быть отражены (Ref. 65). В случае JDBC Connection играет роль интерфейса службы, DriverManager registerDriver – это API регистрации поставщика, DriverManager getConnection – это API доступа к службе, а Driver – интерфейс поставщика службы. Существует много вариантов шаблона структуры

9

Тестування розробленої моделі



10

Висновки

* У рамках магістерської роботи були проаналізовані усі етапи розпізнавання символі з зображення і автоматичного перекладу текстів. На основі аналізу був реалізований додаток, що поєднує в собі оптичне розпізнавання символів і подальший машинний переклад тексту.
* Зокрема, був проведений огляд існуючих методологій нейронних мереж і їх застосовності для вирішення завдань оптичного розпізнавання символів.
* Після повного обзору підходів до машинного перекладу текстів, був обраний комбінований спосіб на основі прикладів перекладеного тексту, та на основі тематичних словників. Комбінування цих 2 методів дозволяє досягати високою точності перекладу та високої швидкості роботи методу.
* Була запропонована архітектура модульної системи розпізнавання тексту, в якій кожен етап розпізнавання реалізується окремим компонентом, а зв'язок між компонентами (у тому числі - зворотний зв'язок) здійснюється програмою.
* Як приклад роботи створений трьох модульний додаток, що має модуль покращення якості зображення, модуль оптичного розпізнавання символі з зображення, та модуль автоматичного перекладу за допомогою машинного навчання. Додаток показує високу якість оптичного розпізнавання тексту на зображеннях високої якості. Автоматичний переклад тексту показує якість на рівні професійних перекладачів.

11

* ДЯКУЮ ЗА УВАГУ

12