Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Модель розпізнавання тексту на основі імунного підходу

(тема)

Виконав:
студент II курсу, групи СПм-19-1
Слісаренко Р.В.
(прізвище, ініціали)

Спеціальність
123 – Комп'ютерна інженерія
(код і повна назва спеціальності)

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

Освітня програма
Системне програмування
(повна назва освітньої програми)

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

Допускається до захисту

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

2020 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

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

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

Освітня програма Системне програмування
(повна назва)

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

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові Слісаренку Роману Валерійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Модель розпізнавання тексту на основі імунного підходу

затверджена наказом по університету від " 30 " жовтня 2020 р. № 1486Ст

2. Термін подання студентом роботи до екзаменаційної комісії 14 грудня 2020 р.

3. Вхідні дані до роботи ocr, імунний підхід, методи скелетонізації, aws textract

4. Перелік питань, що потрібно опрацювати в роботі Як працює ocr, реалізація розпізнавання зображення на основі імунного підходу, розпізнавання зображення за допомгою стороннього сервісу aws textract

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 8 слайдів

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

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Огляд OCR технології	02.11 – 09.11	
2	Аналіз імунного підходу і моделей	08.11 – 15.10	
3	Аналіз методів скелетонізації	15.11 – 20.11	
4	Розробка web-додатку	20.11 – 30.11	
5	Розробка власного api	01.12 – 05.12	

Дата видачі завдання 02 листопада 2020 р.

Студент _____
(підпис)

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

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 78 с., 14 рис., 2 дод., 7 джерел.

OCR, ІМУННИЙ ПІДХІД, AWS TEXTRACT.

Актуальність проблеми полягає в тому, що а даний час більшість документів зберігається на комп'ютерах і завдання створення повністю електронного документообігу далека до повної реалізації. Існуючі системи охоплюють діяльність окремих організацій, а обмін даними між організаціями здійснюється за допомогою традиційних паперових документів. Завдання перекладу інформації з паперових або ж графічних на електронні носії актуальна не тільки в рамках потреб, що виникають в системах документообігу. Сучасні інформаційні технології дозволяють нам істотно спростити доступ до інформаційних ресурсів, накопичених людством, за умови, що вони будуть переведені в електронний вигляд.

Розпізнавання образів є дуже складним завданням в теоретичному і практичному сенсах, незважаючи на те, що з нею досить легко справляються багато живі організми і людина. Вкрай складно створити штучну систему і її технічно реалізувати для того, щоб ефективно виконувати даний процес. В даному випадку, під розпізнаванням розуміється співвіднесення зображення об'єкта, його образу, набору ознак самого об'єкта.

ABSTRACT

Master’s thesis: 78 pages, 14 figures, 2 appendices, 7 sources.

OCR, IMMUNE APPROACH, AWS TEXTRACT.

The aim of the project is to develop OCR software for online writing recognition. OCR is an Optical character recognition and is the mechanical or electronic translation of images of handwritten or typewritten text (usually captured by a scanner) into machine-editable text. OCR is a field of research in pattern recognition, artificial intelligence, and machine vision. Recognition is used most often to describe the ability of a computer totranslate human writing into text. This may take in one of the two ways, either by scanning ofwritten text or by writing directly on peripheral input devices.

We are going to implement the software which will recognize the characters from an online document (in image format) and use it as an individual user profile. Here we are developing OCR which will recognize handwritten English characters.OCR is an Optical character recognition and is the mechanical or electronic translation of images of handwritten or typewritten text (usually captured by a scanner) into machine-editable text.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ІА – імунний алгоритм.

ІМ – імунна модель.

ІО – імунний оператор.

ШІС – штучна імунна система.

OCR – механічне або електронне переведення зображень рукописного, машинописного або друкованого тексту в послідовність кодів, що використовуються для представлення в текстовому редакторі.

ВСТУП

Оптичне розпізнавання тексту (англ. optical character recognition, OCR) - це механічне або електронне переведення зображень рукописного, машинописного або друкованого тексту в послідовність кодів, що використовуються для представлення в текстовому редакторі. Розпізнавання широко використовується для конвертації книг і документів в електронний вигляд, для автоматизації систем обліку в бізнесі або для публікації тексту на веб-сторінці. Оптичне розпізнавання тексту дозволяє редагувати текст, здійснювати пошук слова або фрази, зберігати його в компактнішій формі, демонструвати або роздруковувати матеріал, не втрачаючи якості, аналізувати інформацію, а також застосовувати до тексту електронний переклад, форматування або перетворення в мовлення.

1 ОПТИЧНЕ РОЗПІЗНАВАННЯ СИМВОЛІВ (OCR)

1.1 Технологія OCR

Як перейти від паперового до цифрового документообігу, щоб заощадити час і гроші? Як перемістити тонни паперової інформації на невеликий жорсткий диск або навіть в хмару? Завдяки технології оптичного розпізнавання символів (OCR) перетворити відскановані документи в доступні для читання і редагування цифрові файли досить просто.

OCR - це використання технології для ідентифікації та перетворення відсканованих рукописних або друкованих текстових символів в електронну форму, легше розпізнавану комп'ютерами та іншими програмами. Базовий процес розпізнавання включає вивчення тексту і переклад символів в код, який можна використовувати для обробки даних. OCR іноді також називають розпізнаванням тексту.

Технологія складається з поєднання апаратного та програмного забезпечення, яке використовується з метою перетворення фізичних документів в машинозчитуваний текст. Апаратне забезпечення, таке як оптичний сканер або спеціалізована монтажна плата, використовується для копіювання або читання тексту, в той час як програмне забезпечення відповідає за розширену обробку. Програмне забезпечення може використовувати штучний інтелект для реалізації більш досконалих методів інтелектуального розпізнавання (ICR), таких як ідентифікація мов або стилів рукописного введення.

OCR найчастіше використовується для перетворення друкованих юридичних або історичних документів в PDF-файли. Після цього отримані електронні копії користувачі можуть редагувати, форматувати за допомогою звичайних редакторів тексту.

1.2 Етапи розвитку OCR

В 1929 році Густав Таущек отримав патент на метод оптичного розпізнавання тексту в Німеччині, після чого за ним пішов Гендель, отримавши патент на свій метод у США в 1933. В 1935 Таущек також отримав патент США на свій метод. Машина Таущека являла собою механічний пристрій, що використовує шаблони й фотодетектор.

В 1950 році Девід Х. Шепард, криптоаналітик з агентства безпеки збройних сил Сполучених Штатів, проаналізувавши задачу перетворення друкованих повідомлень у машинну мову для обробки комп'ютером, побудував машину, що розв'язує дане завдання. Після того як він отримав патент США, він сповістив про це в «Вашингтон Дейлі Ньюз» (27 Квітня 1951) і в «Нью-Йорк Таймс» (26 грудня 1953). Потім Шепард заснував компанію, що розробляє інтелектуальні машини, що незабаром випустила перші у світі комерційні системи оптичного розпізнавання символів.

Перша комерційна система була встановлена на «Рідерз дайджест» в 1955 році. Друга система була продана компанії «Standard Oil» для читання кредитних карт для роботи із чеками. Інші системи, що поставляються компанією Шепарда, були продані наприкінці 1950-х років, у тому числі сканер сторінок для національних повітряних сил США для читання й передачі телетайпом машинописних повідомлень. Пізніше ліцензію на використання патентів Шепарда отримала IBM.

Приблизно в 1965 «Рідерз Дайджест» і «Ар-Сі-Ей» почали співробітництво з метою створити машину для читання документів, що використовує оптичне розпізнавання тексту, призначену для оцифрування серійних номерів купонів «Рідерз Дайджест», що повернулися з рекламних оголошень. Для документів, надрукованих барабанним принтером «Ар-Си-Эй», був використаний спеціальний шрифт OCR-A. Машина для читання документів працювала безпосередньо з комп'ютером RCA 301 (одним із перших масивних комп'ютерів). Швидкість роботи машини була 1500

документів у хвилину: вона перевіряла кожний документ, крім тих, які вона не змогла обробити правильно. Поштова служба Сполучених Штатів використовує машини, що використовують оптичне розпізнавання тексту, для сортування пошти з 1965 року на основі технологій, розроблених дослідником Яковом Рабиновим. В Європі першою організацією, що використовує машини з оптичним розпізнаванням тексту, був британський поштамт. Пошта Канади застосовує системи оптичного розпізнавання символів з 1971 року. На першому етапі в центрі сортування системи оптичного розпізнавання символів зчитують ім'я й адресу одержувача й друкують на конверті штрих-код. Він наноситься спеціальним чорнилом, яке чітко видиме в ультрафіолетовому світлі. Це робиться, щоб уникнути плутанини з полем адреси, заповненим людиною, що може бути в будь-якому місці на конверті.

В 1974 році Рей Курцвейл створив компанію «Курцвейл Комп'ютер Продактс», і почав працювати над розвитком першої системи оптичного розпізнавання символів, здатної розпізнати текст, надрукований будь-яким шрифтом. Курцвейл уважав, що краще застосування цієї технології - створення машини читання для сліпих, котра дозволила б сліпим людям мати комп'ютер, що вміє читати текст уголос. Цей пристрій вимагав винаходу відразу двох технологій - планшетний ПЗЗ-сканер і синтезатор, що перетворює текст у мову. Кінцевий продукт був представлений 13 січня 1976 під час прес-конференції, очолюваної Курцвейлом і керівниками національної федерації сліпих.

В 1978 році компанія «Курцвейл Комп'ютер Продактс» розпочала продаж комерційної версії комп'ютерної програми оптичного розпізнавання символів. Через два роки Курцвейл продав свою компанію «Ксерокс», що була зацікавлена в подальшій комерціалізації систем розпізнавання тексту. «Курцвейл Комп'ютер Продактс» стала дочірньою компанією «Ксерокс», відомого як «Скансофт» [5].

1.2.1 The Apple Newton MessagePad

MessagePad - припинена серія персональних цифрових асистентських пристроїв, розроблена компанією Apple Computer Inc. для платформи Newton в 1993 році. Деякою електронною технікою та виробництвом пристроїв MessagePad від Apple займалася в Японії компанія Sharp Corporation. Пристрої базувались на процесорі ARM 610 RISC та усім розробленим програмним забезпеченням для розпізнавання рукописного вводу, і були розроблені та продані компанією Apple. На пристроях працювала ОС Newton.

Рисунок 1.1 – The Apple Newton MessagePad 100

У початкових версіях (Newton OS 1.x) розпізнавання рукописного вводу дало надзвичайно неоднозначні результати для користувачів і часом було неточним. Оригінальний механізм розпізнавання рукописного вводу називався «Каліграф» і ліцензувався російською компанією «Параграф Інтернешнл». Дизайн каліграфа був досить вишуканим; він намагався вивчити природний почерк користувача, використовуючи базу даних відомих слів, щоб здогадуватися про те, що користувач писав, і міг інтерпретувати написання в будь-якому місці екрана, будь то відбитки від руки, скоропис

або їх поєднання. На відміну від цього, графіті Palm Pilot мали менш вишуканий дизайн, ніж Каліграф, але іноді виявлялися більш точними та точними завдяки опорі на фіксований, заздалегідь визначений штриховий алфавіт. Абетка обведення використовувала форми букв, які нагадували стандартний почерк, але які були модифіковані, щоб бути простими та дуже простими для диференціації. Компанія Palm Computing також випустила дві версії Graffiti для пристроїв Newton. Версія Ньютона іноді працювала краще, і вона також могла показувати штрихи, оскільки вони писалися, оскільки введення робилося на самому дисплеї, а не на шовкографії.

Для редагування тексту Newton мав дуже інтуїтивно зрозумілу систему рукописного редагування, наприклад, викреслювання слів, які слід видалити, обведення тексту, який потрібно виділити, або використання письмових символів для позначення вставок.

Пізніші випуски операційної системи Newton зберегли оригінальний розпізнавач для сумісності, але додали розпізнавач, роздрукований лише текстом (не скоропис), під назвою «Розетта», який був розроблений Apple, включений у версію 2.0 операційної системи Newtonта вдосконалено в Newton 2.1. Розетта, як правило, вважається значним покращенням, і багато рецензентів, тестувальників та більшості користувачів вважають програмне забезпечення розпізнавання рукописного тексту Newton 2.1 кращим за будь-яку з альтернатив навіть через 10 років після його введення. Розпізнавання та обчислення рукописних горизонтальних та вертикальних формул, таких як «1 + 2 =», також розроблялося, але ніколи не випускалося. Однак користувачі писали подібні програми, які могли оцінювати математичні формули за допомогою Інтелектуального помічника ОС Ньютон, унікальної частини кожного пристрою Newton.

Розпізнавання рукописного вводу та частини користувальницького інтерфейсу для Newton найкраще розуміються в контексті широкої історії обчислень пера, яка є досить великою.

Життєво важливою системою розпізнавання рукописного тексту

Newton є немодерне виправлення помилок. Тобто корекція проводиться «на місці» без використання окремого вікна або віджета, з використанням мінімуму жестів. Якщо слово розпізнано неправильно, користувач може двічі торкнутися його, і в меню під стилусом з’явиться список варіантів. Найчастіше правильне слово буде в списку. Якщо ні, кнопка внизу списку дозволяє користувачеві редагувати окремі символи цього слова. Інші жести пером можуть робити такі речі, як транспонування літер. Спливаюче вікно коригування також дозволило користувачеві повернутися до початкової, нерозпізнаної форми букв - це було б корисно в сценаріях створення нотаток, якщо не вистачало часу для негайних виправлень. Для економії пам’яті та місця для зберігання, альтернативні гіпотези розпізнавання не будуть економити необмежено довго. Якщо користувач повернеться до примітки через тиждень, наприклад, він побачить лише найкращий збіг. Виправлення помилок у багатьох сучасних системах рукописного вводу надає таку функціональність, але додає більше кроків до процесу, значно збільшуючи переривання робочого процесу користувача, що вимагає дана корекція.

1.2.2 CAPTCHA

CAPTCHA (англ. «completely automated public turing test to tell computers and humans apart» - повністю автоматизований публічний тест Тюринга для розрізнення комп'ютерів і людей) - торгова марка Університет Карнегі - Меллона, комп'ютерний тест типу виклик-відповідь, який використовується для того, щоб визначити, хто використовує систему - людина чи комп'ютер. Термін з'явився в 2000 році. Простими словами, Капча - це спеціальний тест, який використовується для того щоб різні боти та автоматизовані програми не могли видавати себе за людей та робити подібні дії:

- автоматичне створення акаунтів;
- підписка на пропозиції;

- збір електронних адрес;
- створення електронних адрес;
- порушення конфіденційності.

У найпоширенішому варіанті CAPTCHA від користувача потрібно ввести символи, зображені, як правило, в спотвореному вигляді на пропонованому рисунку, іноді з додаванням шуму або напівпрозорості. Рідше застосовуються CAPTCHA, засновані на розпізнаванні мови (в основному як альтернатива для людей з порушеннями зору), або на інших варіантах завдань штучного інтелекту [6].

CAPTCHA найчастіше використовується при необхідності запобігти використанню інтернет-сервісів ботами, зокрема, для запобігання автоматичній реєстрації, викачуванню файлів, масовим розсилкам тощо.

Рисунок 1.2 – Приклад captcha

Методи протидії:

1) Автоматизоване розпізнавання:

Існують програми, що розпізнають конкретні реалізації CAPTCHA, наприклад Pwntcha. Крім того існує можливість підключати модулі з програм розпізнавання тексту загального призначення (наприклад Fine Reader) в програми сторонніх розробників для розпізнавання картинок CAPTCHA. Окрім Fine Reader можна використовувати такі програми, як Tesseract, Ocrad, GOCR. Задля протидії автоматичному розпізнаванню, досліджується варіанти анімованої CAPTCHA або розпізнавання не тексту, а вибір певного рисунку із кількох запропонованих.

2) Ручне розпізнавання:

Є спосіб «розпізнавання», що використовує людський ресурс та ресурс сайтів з високою відвідуваністю - наприклад, соціальні мережі. Робот викачує CAPTCHA з інтернет-сервісу і демонструє його користувачу сайта, з проханням ввести код, який він бачить на картинці. В результаті, користувач дістає доступ до ресурсу, а робот дізнається код, зображений на картинці («метод лемінгів»). Варіантом цього методу є сервіс Captcha Exchange Server, запущений в березні 2007 і направлений на обхід картинок CAPTCHA, що використовуються файлообмінниками [9]. Принцип роботи сервісу ґрунтується на системі балів, які користувач може заробити, розпізнавши картинки для інших користувачів, і пізніше витратити, запустивши програму автоматичного викачування з файлообмінників, при цьому картинки будуть розпізнані іншими користувачами сервісу. Таким чином, користувач може оптимізувати витрати свого часу і грошей, набираючи бали в час, коли він знаходиться біля комп'ютера, і витрачаючи їх тоді, коли йому зручніше викачувати інформацію (наприклад, якщо вночі доступ в Інтернет дешевший).

Вразливості CAPTCHA-захисту не означають, що будь-який CAPTCHA-захист апріорі безглуздий. Тут спостерігається одвічний принцип змагання зброї і захисту (спису та щита).

1.3 Поточний стан технології OCR

Точне розпізнавання латинських символів у друкованому тексті зараз можливе тільки, якщо доступні чіткі зображення, такі як друковані документи. Точність при такій постановці задачі перевищує 99%, абсолютна точність може бути досягнута тільки шляхом наступного редагування людиною. Проблеми розпізнавання рукописного «друкованого» тексту й стандартного рукописного тексту, а також друкованих текстів інших

форматів (особливо з дуже великою кількістю символів) зараз є предметом активних досліджень.

Точність роботи методів може бути вимірювана декількома способами, і тому може сильно варіюватися. Приміром, якщо зустрічається спеціалізоване слово, відсутнє в словниках відповідного програмного забезпечення, при пошуку неіснуючих слів, помилка може збільшитися.

Розпізнавання символів он-лайн іноді плутають з оптичним розпізнаванням символів. Метод оптичного розпізнавання символів — офф-лайн-метод, що працює зі статичною формою подання тексту, у той час як он-лайн-розпізнавання символів ураховує рухи під час писання. Наприклад, в он-лайн-розпізнаванні, що використовує PenPoint OS або планшетний ПК, можна визначити, з якої сторони пишеться рядок: справа ліворуч або зліва праворуч.

Он-лайн-системи для розпізнавання рукописного тексту «на льоту» останнім часом стали широко відомі як комерційні продукти. Алгоритми таких пристроїв використовують той факт, що порядок, швидкість і напрямок окремих ділянок ліній уведення відомі. Крім того, користувач уміє використовувати тільки конкретні форми письма. Ці методи не можуть бути використані в програмному забезпеченні, що використовує скановані паперові документи, тому проблема розпізнавання рукописного «друкованого» тексту, як і раніше, залишається відкритою. На зображеннях із рукописним «друкованим» текстом без артефактів може бути досягнута точність у 80% - 90%, але з такою точністю зображення буде перетворене в текст із десятками помилок на сторінці. Така технологія корисна в дуже обмеженому числі застосувань [8].

Ще одною широко досліджуваною проблемою є розпізнавання рукописного тексту. У цей час досягнута точність навіть нижча, ніж для рукописного «друкованого» тексту. Вищі показники можуть бути досягнуті тільки з використанням контекстної й граматичної інформації. Наприклад, у процесі розпізнання шукати цілі слова в словнику легше, ніж намагатися

проаналізувати окремі символи з тексту. Знання граматики мови може також допомогти визначити, чи є слово дієсловом чи іменником. Форми окремих рукописних символів іноді можуть не містити достатньо інформації, щоб точно (більше 98%) розпізнати весь рукописний текст.

Для рішення складніших проблем у сфері розпізнавання використовуються, як правило, інтелектуальні системи розпізнавання, такі як штучні нейронні мережі.

1.3.1 Новий рівень OCR після появлення iPhone

Прихід iPhone спонукає до розробки зручних додатків для смартфонів, що дозволяють сканувати та конвертувати текст за допомогою телефонної камери. Технологія OCR ефективно починається застосовуватися компаніями, що розробляють мобільні додатки. Розробники мобільних додатків розпочинають створювати ефективні додатки на користь, а також для розваги своїх клієнтів. OCR у мобільних додатках може підвищити ефективність їх роботи. Це допомагає в управлінні інформацією за допомогою збору та управління документами. Будь-яку письмову статтю, вміст або корпоративні дані, будь то цифрові, відскановані чи написані від руки, можна перетворити в машинний друкований текст одним дотиком на мобільному телефоні, оснащеному технологією OCR. Крім того, у власника телефону є засоби для внесення необхідних змін у текст, створення незліченних копій, безпечного зберігання або надсилання якомога більшій кількості одержувачів із вашого мобільного.

Користувачі можуть відчути зручність у робочих процесах за допомогою OCR Mobile Technology. Паперова робота стає мінімальною, і було б легко аналізувати зображення або витягувати дані без клопоту про будь-які проблеми із сервером, підключення даних чи географічні обмеження. Все повинно бути доступно на екрані вашого мобільного пристрою. OCR Mobile Apps надає швидкий доступ до інформації. Крім того,

якщо вони також надають мовні версії, то вони можуть задовольнити регіональні вимоги до усічення у всьому світі. Це вигідно банкам, фінансовим установам, торговцям, роздрібним торговцям та BPO.

Ера мобільності переходить на новий етап з технологією OCR. Впровадження сучасних смартфонів, які оснащені безліччю технологічних функцій, з технологією оптичного розпізнавання тексту може поліпшити функції робочого місця, пов'язані з обробкою даних.

Технологія OCR Mobile - це розширення середовища обробки даних. Розробники мобільних додатків можуть створювати ефективні програми, які дозволяють галузі і трейдерам переносити послуги на мобільні платформи.

Деякі приклади технології OCR для мобільності можуть допомогти в кращому розумінні, наприклад:

Мобільний телефон може легко знімати зображення квитанцій, не боюючись втрати данні і таким чином, може підготувати звіт про витрати за допомогою OCR. Використання Mobile OCR логістичними компаніями для отримання відвантажувальних документів.

Рисунок 1.3 – Сканування чеку за допомогою iPhone

Таким чином, після розуміння численних переваг OCR для мобільності, пора прийняти технологію OCR з розпростертими обіймами, оскільки вона дає помітні переваги і користується як мобільним споживачам, так і корпораціям.

1.4 Як працює OCR?

Коли людина читає текст, він розпізнає символи за допомогою очей і мозку. У комп'ютера в ролі очей виступає камера сканера, яка створює графічне зображення текстової сторінки Для комп'ютера немає різниці між фотографією тексту і фотографією будинку: і те, і інше - набір пікселів. Саме OCR перетворює зображення тексту в текст. А з текстом можна робити що завгодно.

Уявіть, що в алфавіті є тільки одна буква «А». Чи зробить це завдання перетворення картинки в текст простіше? Ні. Справа в тому, що у кожної букви (і будь-який інший графеми) є алограф - різні варіанти накреслення.

Рисунок 1.4 – Варіанти накреслення букви «а»

Людина легко зрозуміє, що все це буква «А». Для комп'ютера ж є два способи вирішення проблеми: розпізнавати символи цілісно (розпізнавання патерну) або виділяти окремі риси, з яких складається символ (виявлення ознак).

У 1960-х роках був створений спеціальний шрифт OCR-A, який використовувався в документах типу банківських чеків. Кожна буква в ньому була однакової ширини (т.зв. шрифт фіксованої ширини або моно шрифт).

Рисунок 1.5 – Зразок шрифту OCR-A

Принтери для чеків працювали з цим шрифтом, і для його розпізнавання було розроблено програмне забезпечення. Оскільки шрифт був стандартизований, його розпізнавання стало відносно простим завданням. Наступним кроком стало навчання програм OCR розпізнавати символи ще в

кількох найпоширеніших шрифтах (Times, Helvetica, Courier і т.д.).

Уявіть, що ви - OCR-програма, якої дали безліч різних букв, написаних різними шрифтами. Як вам відібрати з цього перекілку всі букви «А», якщо кожна з них має заокруглення та відрізняється за висотою від іншої?

Можна використовувати таке правило: якщо бачиш дві лінії, що сходяться вгорі в центрі під кутом, а посередині між ними горизонтальна лінія, то це буква «А». Це правило допоможе розпізнати всі букви «А» незалежно від шрифту. Замість розпізнавання паттерна виділяються характерні індивідуальні риси, з яких складається символ. Більшість сучасних омнішріфтових (вміють розпізнавати будь-який шрифт) OCR-програм працюють за цим принципом. Найчастіше в них використовуються класифікатори на основі машинного навчання (тому що фактично перед нами стоїть завдання класифікації картинок по класах-буквах) останнім часом деякі OCR-движки перейшли на нейронні мережі.

Що робити з рукописним введенням? Людина здатна здогадатися про сенс пропозиції, навіть якщо воно написано самим нерозбірливим почерком (якщо мова не йде про рецепт на ліки, звичайно). Завдання для комп'ютера іноді спрощують. Наприклад, людей просять писати поштовий індекс в спеціальному місці на конверті спеціальним шрифтом. Форми, створені для подальшої обробки комп'ютером, зазвичай мають окремі поля, які просять заповнювати друкованими літерами.

Планшети і смартфони, які підтримують рукописне введення, часто використовують принцип виявлення ознак. При написанні літери «А» екран «відчуває», що спочатку користувач написав одну лінію під кутом, потім другу, і, нарешті, провів горизонтальну лінію між ними. Комп'ютера допомагає те, що всі ознаки з'являються послідовно, один за іншим, на відміну від варіанту, коли весь текст уже записаний від руки на папері.

1.4.1 Визначення підходу

Програмне забезпечення OCR зазвичай працює з великим зображенням сторінки, отриманим в результаті сканування. Зображення зі стандартною ступенем дозволу виходять при скануванні з точністю 300 пікселів на дюйм. Зображення паперового аркуша формату А4 при цьому дозволі займає близько 1 Мб пам'яті. Зображення з більшими параметрами обробляються програмою довше і вимагають більшого об'єму оперативної пам'яті. При цьому далеко не завжди збільшення дозволу призводить до поліпшення якості розпізнавання.

Більшість систем має шаблони, створені для різних накреслень тих чи інших символів. Програма визначає основний шаблон в оброблюваному документі накреслення символів і шукає відповідні символи з цим зображенням. Існують multifont- (шрифтові) і omnifont- (шріфтонезавісімие) алгоритми цього процесу.

У разі multifont растрове зображення накладається на шаблон, після чого вибирається найбільш схожий з існуючих на досліджуване зображення шаблонів.

Omnifont алгоритми ідентифікують символ по закладеним в програму правилам його написання. Обидва ці алгоритму не гарантують високу надійність розпізнавання, проте дозволяють зробити припущення про приналежність даного символу.

Тільки шаблонний опис може застосовуватися переважно для розпізнавання друкованих символів, причому якісно надрукованих. Цікавим є той факт, що рукописні шрифти теж розпізнаються із застосуванням шаблонів, але одночасно зі строгим підходом.

Алгоритми цього підходу були розроблені компанією ABBYY для розпізнавання текстів низької якості. Програма зберігає інформацію нема про правила написання символу, а про наявність в ньому структурних елементів (кілець, дуг, кутів, відрізків і точок).

Цей метод дозволяє виділяти елементи на перекручених зображеннях, краще знаходить нові шрифти і елементи форматування тексту (курсив, жирне накреслення, верхній і нижній індекси).

Інший підхід - контекстне розпізнавання. Людина здатна швидко розрізнити на папері «В» і «Е» не в останню чергу завдяки тому, що він знає контекст слова, в якому зустрічаються ці літери. У OCR-системі для допомоги алгоритмам розпізнавання і для корекції помилок служать вбудовані словники. Які, втім, мають потребу в постійному апдейте.

Сучасні OCR-програми поєднують різні підходи, застосовуючи їх в залежності від типу вихідного документа.

Таким чином, можемо зробити проміжний висновок, що алгоритм розпізнавання являє собою процес послідовного висунення і перевірки програмою цілого ряду гіпотез, заснованих на закладених в програму шаблонах і знаннях. Детальніше з процесом розпізнавання ми познайомимося пізніше.

1.4.2 Визначення принципу

Користувач поміщає документ в сканер, натискає кнопку, і через деякий час в комп'ютер надходить електронне зображення, «фотографія» сторінки. На ній присутні всі особливості оригіналу, аж до найдрібніших подробиць. Це зображення містить всю необхідну для OCR-системи інформацію про вихідний документі.

А далі включається штучний інтелект, який використовує машинні підходи, описані вище.

Однак найкращі в світі системи оптичного розпізнавання - найточніші, швидкі і надійні - конструює жива природа. У їх числі і та система, що вірою і правдою служить кожному з нас, наш внутрішній «распознаватель». Механізми, що дозволяють людині безпомилково впізнавати побачені предмети, поки не досліджені досконально, однак їх базові принципи вивчені

добре. Таких налічують три.

1) Принцип цілісності (integrity), згідно з яким споглядаємо об'єкт розглядається як ціле, що складається з пов'язаних частин. Зв'язок частин виражається в просторових відносинах між ними, і самі частини отримують тлумачення тільки в складі передбачуваного цілого, тобто в рамках гіпотези про об'єкт. Приклад: ми бачимо зображення деревовидної структури. Розпочато розпізнавання. Висуваються гіпотези: це або рисунок дерева, і тоді «гілки» структури відповідають гілкам, або схема автобусних маршрутів, де «гілки» позначають шляху автобусів з різними номерами, або це карта річкової заплави, а «гілки» - русла річок і струмків.

2) Принцип цілеспрямованості (purposefulness) формулюється просто: будь-яка інтерпретація даних переслідує певну мету. Згідно з цим принципом, розпізнавання являє собою процес висунення гіпотез про цілий об'єкті і цілеспрямованої їх перевірки. Приклад (продовження): якщо спостережуване нами зображення - схема маршрутів, то на «гілках» повинні бути позначені зупинки. Якщо зображення - карта заплави, повинні бути назви річок і струмків, а також масштаб. Якщо ж це рисунок дерева, на «гілках» ймовірна наявність листя, а біля основи - зображення трави або землі. Перевірка: позначень зупинок немає, листя і трави немає, у кожної «гілки» надписані назви, внизу проставлено масштаб. Підтверджено гіпотезу: це карта річкової заплави, а «гілки» відповідають руслах. Розпізнавання закінчено.

3) Принцип адаптивності (adaptability) має на увазі здатність системи до самонавчання. Отримана при розпізнаванні інформація упорядковується, зберігається і використовується згодом при вирішенні аналогічних завдань. Перевага самообучающихся систем полягає в здатності направляти шлях логічних міркувань, спираючись на раніше накопичені знання. Приклад: ми бачимо нове зображення деревовидної структури, внизу проставлено масштаб. Інформація: у минулий раз таке зображення виявилося картою, тому перш ніж висувати інші гіпотези слід перевірити наявність назв річок.

Перевірка: назви виявлені. Розпізнавання закінчено.

Замість повних назв цих принципів часто вживають абревіатуру IPA, складену з перших букв відповідних англійських слів. Переваги системи розпізнавання, що працює відповідно до принципів IPA, очевидні навіть неспеціалісту; саме вони здатні забезпечити максимально гнучке і осмислену поведінку системи. Цілком можна порівняти з тим, що демонструють живі «розпізнавачі», створені природою.

Відповідно до вищеописаними принципами на всіх етапах обробки документа діє ABBYY FineReader - безсумнівно, найбільш поширена на пострадянському просторі OCR-система.

Наприклад, на етапі розпізнавання символів зображення, відповідно до принципу цілісності, буде інтерпретовано як якийсь об'єкт тільки в тому випадку, якщо на ньому присутні всі структурні частини цього об'єкта, і ці частини знаходяться в відповідних відносинах. Інакше кажучи, ABBYY FineReader не намагається приймати рішення, перебираючи тисячі еталонів в пошуках найбільш підходящого. Замість цього висувається ряд гіпотез щодо того, на що схоже виявлене зображення, потім кожна гіпотеза цілеспрямовано перевіряється.

Причому, на відміну від конкурентів, набагато краще справляється саме з форматуванням тексту і всього документа, а не тільки «голого» тексту. Як і слід чинити, виходячи з принципу цілеспрямованості. Причому перевіряти, чи вірна висунута гіпотеза, система буде, спираючись на накопичені раніше відомості про можливі накресленнях символу в розпізнається документі. У повній відповідності з принципом адаптивності.

Отже, ми прийшли до початку процесу розпізнавання. С допомогою OCR-програми комп'ютер зможе «прочитати» на відсканованої сторінці текст, відокремивши його від ілюстрацій і інших елементів оформлення, знайти таблиці і розібратися в їх вмісті.

1.4.3 Труднощі при експлуатації OCR

Занадто широкі ліміти на розміри розпізнаються зображень. Виходячи з практики моїх колег, замовники часто встановлюють занадто широкі ліміти на розміри розпізнаються графічних файлів. Так, щоб OCR працював добре, потрібно обмежувати розміри зображень. Але замовники прагнуть контролювати все підряд, вважаючи, що навіть в зображенні розміром 100x100 pixels і 5 КБ можуть витекти цінні дані. В цілому, звичайно, 100x100 pixels і 5 Кб теж обмеження, але занадто вже низькі ці пороги.

Інша крайність - прагнення розпізнати важкі файли по кілька сотень Мб. Зрозуміло, що через корпоративну пошту такі зображення не пролізуть через обмеження на розмір пересилаються. Але ось по інших каналах перехоплення важкі файли наполегливо прагнуть розпізнавати. Якщо ж замовник хоче додати до цього ще й великий обсяг high-res зображень, то для цього потрібно мати відповідні серверні потужності. У підсумку, при настільки широких мінімальних і максимальних порогах на розмір розпізнаються файлів створюється високе навантаження на процесор на серверах, що уповільнює роботу всіх підсистем.

Що тут можна порекомендувати? Перш за все проаналізувати, в якій компанії містяться конфіденційні дані, після чого прикинути розумні мінімальні і максимальні обмеження на розміри контрольованих зображень. Зазвичай ми рекомендуємо замовникам зафіксувати нижню межу дозволу зображення від 200 pixels, в ідеалі від 400 pixels (по осях X і Y), і розміру файлів не менше 20 Кб, краще більше. Також не має сенсу відправляти в OCR великовагові зображення - вони елементарно перенавантажуватимуть ваші сервера і не факт, що будуть розпізнані.

Надмірне навантаження на сервери, що виникає по вищеописаним причинам, веде по ланцюжку до збільшення часу розпізнавання зображень і обробки запитів в цілому. В результаті в системі починає збільшуватися черга повідомлень на фільтрацію. Крім того, в OCR-модуль можуть

приходити графічні файли, які в принципі неможливо розпізнати (важкі файли, низька якість і т.д.), в результаті чого виникає таймаут обробки зображень. Якщо нерозпізнаних файлів надходить багато, а в системі встановлені високі таймаут на розпізнавання, сервіс фільтрації чекає, поки цей таймаут настане, і тільки потім приступає до обробки наступного запиту. Весь процес обробки може серйозно гальмуватися.

Що радять в таких випадках? При виникненні черги на обробку графічних зображень потрібно подивитися налаштування OCR в системі і спробувати знайти причину гальмування. Це може відбуватися, наприклад, через проблеми взаємодії між процесами на самому сервері. Крім цього важливим моментом при налаштуванні OCR є виставлення адекватних таймаутів на розпізнавання зображень. У загальному випадку досить 90 секунд, щоб зображення точно розпізналося. Якщо з зображення не отримали текст за 90 секунд, то можна припустити, що OCR не розпізнає зображення в принципі. У цьому місці також можуть виникати проблеми конфігурації OCR, коли виставляють високі таймаут на розпізнавання і тим самим робляться спроби розпізнати нераспознаваемой.

Що ще може стати причиною таймаута? Тут ми знову повернемося до питання конфігурації системи. Сервіс фільтрації, як і сервіс OCR, оперує тред, які обробляють повідомлення і зображення. Система може бути некоректно налаштована в частині кількості оброблювачів сервісу фільтрації і кількості оброблювачів OCR. У такій ситуації в якісь моменти OCR може просто не встигати обробляти всі запити на розпізнавання, і таким чином будуть з'являтися таймаути обробки зображень. Подібна поведінка системи наводить на думки про проблеми проектування і баги в архітектурі, але насправді це не так.

Якщо в DLP-систему потрапляє на аналіз зображення, яке OCR не може розпізнати, існує кілька варіантів вирішення проблеми.

З яких причин зображення можуть не розпізнаватися? Наприклад, за такими:

1) нестандартна колірна схема зображення;

2) низький дозвіл зображення;

3) неправильна орієнтація зображення і міститься в ньому тексту в просторі;

4) перекоси рядків і спотворення пропорцій тексту в зображенні;

У OCR-двигунах закладені різні функції автоматичної корекції зображення, які сильно підвищують шанси на успішне розпізнавання міститься в ньому тексту. Однак, на практиці ці чарівні інструменти не завжди спрацьовують. В даному конкретному випадку ми повинні доналаштувати для замовника OCR-модуль таким чином, щоб він розпізнавав цю нестандартну колірну схему.

5) Невідповідність одного з параметрів документа заданих розмірах розпізнаються зображень.

Наприклад, в конфігурації системи задані кордону розмірів розпізнаються зображень 200x1000 pixels, а в OCR надійшов файл розміром 500x1500 pixels (верхній ліміт перевищено). В цьому випадку необхідно виправити налаштування OCR для розпізнавання таких зображень.

Це, мабуть, один з найпопулярніших сценаріїв доналаштування системи після того, як нам кажуть, що OCR не працює.

2 ІМУННИЙ ПІДХІД (OCR)

2.1 Особливості імунних моделей і алгоритмів

Розвиток інформаційних технологій і систем штучного інтелекту на основі біологічних систем призвів до появи ряду нових напрямків, таких як штучні нейронні мережі, генетичні алгоритми, алгоритм мурашника, штучні імунні системи. Використання біологічних принципів організації обчислень не тільки розширює можливості вирішення практичних і науково-дослідних завдань, але призводить також до збагачення термінологічного апарату. У зв'язку з цим, в даний час і області інформаційних технологій використовуються такі терміни, як імунна модель, імунний алгоритм (імунний метод), імунний оператор та ін.

Алгоритм називається імунним, якщо він працює відповідно основних принципів теорії штучних імунних систем на основі однієї з існуючих імунних моделей [3, 4]. В даний час існує велика кількість імунних алгоритмів, призначених для вирішення певних практичних завдань, таких як класифікація даних, розпізнавання образів, прогнозування, забезпечення інформаційної безпеки, управління та ін. З усієї сукупності існуючих на сьогоднішній день імунних алгоритмів можна виділити кілька універсальних методів, які є базовими по відношенню до решти. Ці методи є універсальними, і при незначній модифікації можуть використовуватися при вирішенні великої кількості завдань. Моделювання саме цих ІА має підтримуватися розробляється системою моделювання ШІС.

Слід зазначити, що однією з основних властивостей ІА є можливість їх навчання при вирішенні специфічних завдань, тобто ІА можуть функціонувати в двох основних режимах :

1) Режим навчання ІА, при якому, в залежності від виду розв'язуваної задачі, алгоритму надається навчальна вибірка об'єктів, що представляє

собою деякий вирішений приклад (сукупність розпізнаний образів або класів), для знаходження правил поділу на групи або будь-яких інших залежностей між ними;

2) Режим виконання поставленого завдання, при якому навчений алгоритм вирішує завдання при використанні правил розбиття або іншої інформації, отриманої в результаті навчання.

Крім цього, ІА відрізняються від методів, що функціонують на основі інших, не імунних, а математичних принципів, можливістю роботи з динамічно змінюючими даними. Ця особливість дозволяє використовувати ІА для роботи з об'єктами, що змінюють свою поведінку в режимі реального часу і є одним з основних переваг.

Математична модель називається імунної в тому випадку, якщо вона функціонує на основі використання принципів роботи природних імунних систем хребетних [2] . Це накладає на модель ряд вимог і обмежень:

- подання даних у вигляді популяцій імунних об'єктів - антигенів, антитіл, клонів, Т-лімфоцитів, В-лімфоцитів, фагоцитів та ін. З поділом між ними різних функціональних обов'язків;

- використання специфічної системи вимірювання - для вимірювання ступеня близькості (подібності) ознак різних об'єктів і ІМ використовується поняття аффінності між об'єктами [1].

$$Aff_{ij} = (1 + d_{ij})^{-1}$$

де d_{ij} – відстань між і-тми і j-тими імунними об’єктами.

Зазвичай для визначення відстані використовується евклідова відстань, проте крім цього можуть бути використані інші види метрик (лінійна відстань, відстань Хеммінга):

- використання біологічних принципів взаємодії між об'єктами різних популяцій - в результаті взаємодії з антигенами - антитіла переходять в збуджений стан і піддаються клонуванню і мутації; поведінку В-лімфоцитів

визначається Т-лімфоцитами і ін;

- використання, в залежності від виду моделі і використовуваного способу організації взаємодії між імунними об'єктами, певній послідовності виклику елементарних імунних процесів - імунних операторів, кожен з яких є деякою функцією і має своє специфічне призначення.

Однією з основних особливостей, характерних для ІМ, є використання специфічних типів даних - антитіл і антигенів. При цьому антигени являють собою вирішуване завдання, або результат, який повинен бути досягнутий при виконанні ІА, який функціонує на основі деякої імунної моделі. Зазвичай антигени не виробляють ніяких дій, а тільки надаються популяції антитіл. Всі процеси, що проходять в ІМ, зачіпають популяцію антитіл, тобто антитіла є основними робочими елементами ІВС. В ході роботи ІА антитіла піддаються клонуванню, мутації, відбору, старіння і т.д. Таким чином, поділ даних на антигени і антитіла є одним з головних і необхідних умов для того, щоб математична модель могла називатися імунною

Як зазначалося раніше, теорія ШІС є малодослідженим напрямком в області інформаційних технологій і технологій моделювання штучного інтелекту, тому на сьогоднішній день існує невелика кількість основних імунних моделей. Найбільш поширеними ІМ є:

- Модель клонального відбору (clonal selection models) - описує основні властивості ЄІС, які проявляються при формуванні імунної відповіді на стимуляцію системи антигенами. Ця модель ґрунтується на тому, що в результаті взаємодії антитіл з антигенами в системі залишаються лише антитіла, які характеризуються найкращою аффинностью до популяції антигенів. Потім цього ці антитіла піддаються клонуванню і мутації, в результаті чого ІМ виробляє антитіла, специфічні даного антигену [12]. Після цього з антитіл з найкращого аффінністю в ІМ утворюються клітини пам'яті для забезпечення вторинної імунної відповіді у разі повторного потрапляння в організм антигену даного типу (рисунок 2.1).

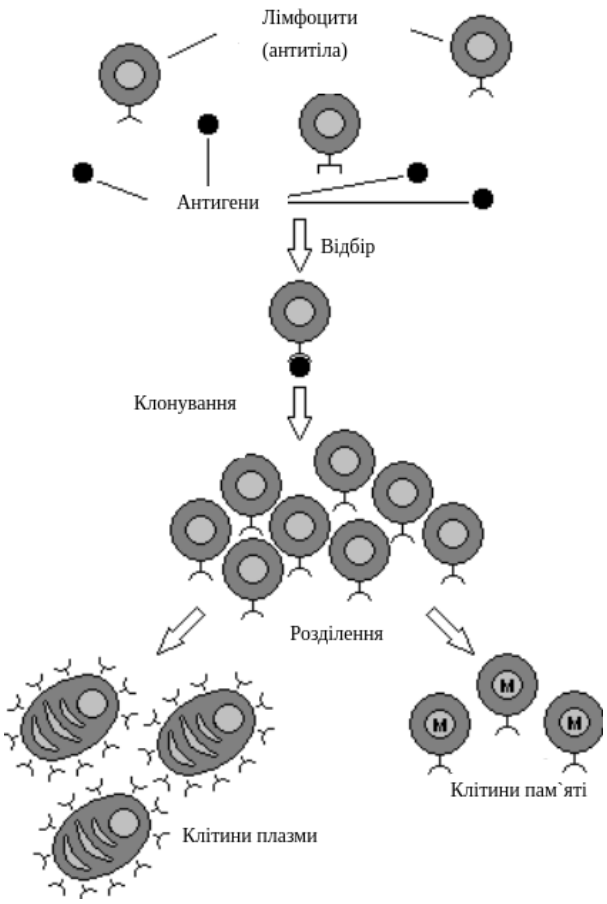

Рисунок 2.1 – Принцип роботи моделі клонального відбору

В роботі імунної моделі клонального відбору використовуються кілька основних принципів функціонування ЄІС:

- клонування антитіл в результаті взаємодії з антигеном;

- мутація антитіл;

- видалення антитіл з мінімальною афінністю до антигенів.

На сьогоднішній день існує декілька універсальних алгоритмів, що використовують принцип клонального відбору: CLONALG, BCA . Дані алгоритми використовуються в основному для вирішення завдань розпізнавання образів, і класифікації об'єктів.

- Моделі на основі негативного / позитивного відбору (negative / positive selection models). Принцип негативного / позитивного відбору є найбільш досліджуваними в теорії ШІС, при цьому найбільшого поширення набули моделі негативного відбору, проте методи позитивного відбору за

своєю організацією від них практично не відрізняються. В основі даних методів лежить властивість ЄІС розпізнавати клітини організму і класифікувати їх на «свої» і «чужі» для забезпечення відповідної імунної реакції організму (рисунок 2.2).

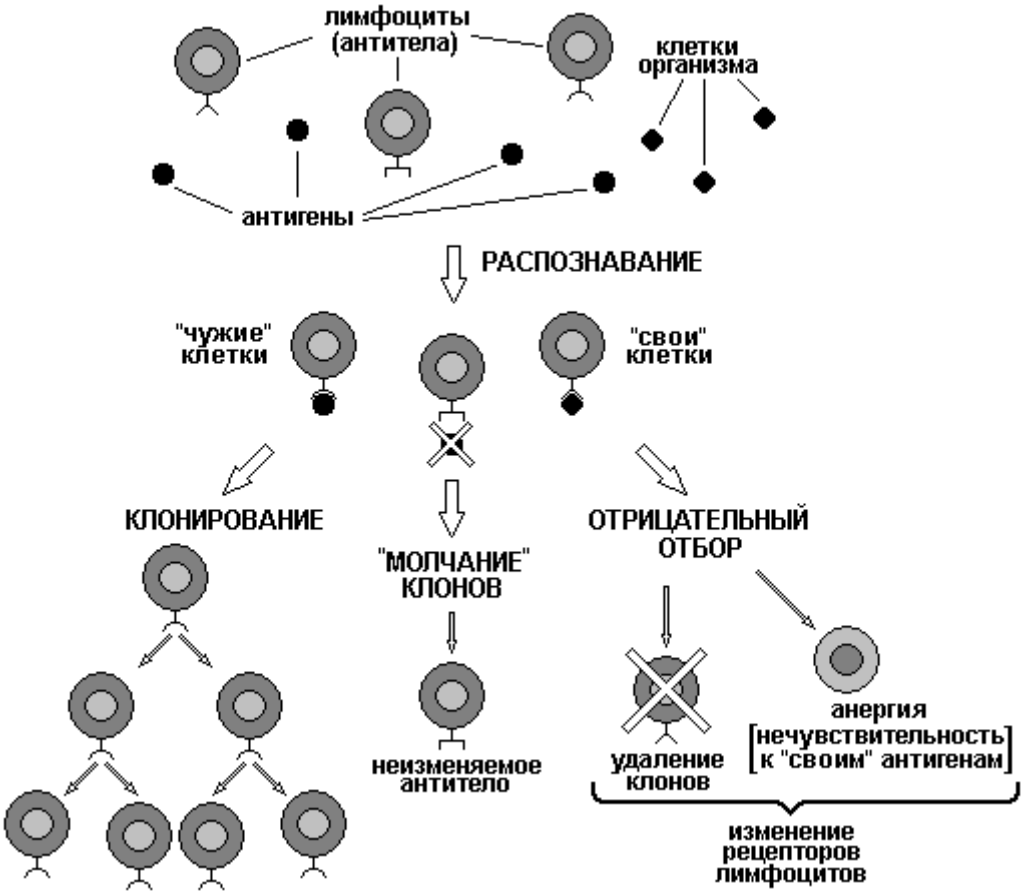

Рисунок 2.2 – Принцип негативного відбору

Для організації негативного відбору використовуються лімфоцити, що мають на своїй поверхні рецептори (антитіла з детекторами), які в результаті взаємодії з клітинами здатні виявляти чужорідні білки (антигени). Набір антитіл-рецепторів формуються випадковим чином на поверхні лімфоцитів. При цьому, антитіла, які розпізнають «свої» клітини видаляються, а антитіла, які розпізнають «чужі» антигени залишаються. Таким чином, в результаті негативного відбору формується популяція лімфоцитів, здатна виявляти і розпізнавати будь-які «чужі» антитіла.

Найбільш поширеними методами негативного відбору є методи Форрест-Еспонди, що використовуються для створення алгоритмів виявлення аномалій і організації захисту від вірусів.

Слід зазначити, що основні принципи функціонування даних ІМ за принципом роботи багато в чому схожі, але між ними є ряд істотних відмінностей. Модель клонального відбору за принципом організації дуже походу на модель імунної мережі. Основна відмінність між ними полягає в механізмі редагування популяції антитіл після клонування і мутації. У моделі клонального відбору для цього використовується принцип відбору клонів, робота якого полягає в аналізі афінності між клонами і антигенами, видаленні клонів з гіршою афінностю, а також заміні клонованих антитіл які залишилися клонами в процесі старіння популяції. На відміну від моделі клонального відбору, модель імунної мережі для редагування популяції використовує механізм супресії, головним завданням якого є придушення (стримування) зростання популяції антитіл шляхом аналізу афінних не тільки між антитілами і антигенами, а й між всередині популяції антитіл. Модель негативного / позитивного відбору клонів використовується для вирішення обмеженої кількості специфічних завдань, пов'язаних з організацією захисту від вірусів, управлінням, виявленням аномалій і т.д. Дана ІМ відрізняється від інших моделей принципом проведення відбору клонів. При цьому аналізуються, як і в моделі імунної мережі аналізуються не тільки аффинности між антитілами і антигенами, а й між антитілами. Головна відмінність даної моделі від моделі імунної мережі полягає в зіставленні таких цих величин і відборі клонів на основі встановленого принципу відбору - якщо в моделі використовується негативний відбір, то серед безлічі клонів відбираються клони, у яких аффинності з антигенами вище афінності з антитілами (перевищують певний поріг). Якщо в моделі використовується позитивний відбір - то з безлічі клонів вибираються об'єкти, у яких афінність з антитілами перевищує аффинности з антигенами. Розглянуті імунні моделі є найбільш поширеними, тому, середа моделювання ІА повинна надавати

можливість роботи з даними моделями і їх алгоритмами.

Серед існуючих імунних алгоритмів виділяють кілька універсальних методів, які можу використовуватися для вирішення великої кількості практичних завдань, такими алгоритмами є: BCA, CLONALG, aiNET, RLAIS, метод Форрест-Еспонди .

Метод BCA (B-cell algorithm) є одним з найбільш ранніх універсальних ІА, що функціонують на основі імунної моделі клонального відбору. В даному методі відсутній етап первинного відбору антитіл, і кожному антитілу представляється популяція антигенів. При цьому основною характеристикою антитіла є рівень його стимуляції, який визначається на підставі афінності з антигенами:

$$sl_i = \sum_{j=1}^{n} aff_{ij} \Big/ n, \; i \in AB, \; j \in AG$$

де sl_i - рівень стимуляції i-го антитіла;

aff_{ij} - аффіннісит між i-тим антитілом і j-тим антигеном.

На відміну від більшості ІА в BCA використовується послідовне клонування антитіл, тобто для кожного антитіла власна популяція клонів, при цьому кількість клонів, що виділяються антитіла при клонуванні, не залежить від рівня його стимуляції. Сформованим клонам даного клонованого антитіла піддаються мутації, і після визначення рівня стимуляції для кожного клону популяцією антигенів проводиться клональний відбір, в результаті якого залишається тільки один клон, який замінює свого батька (клонована антитіло) в разі, якщо його рівень стимуляції вище ніж рівень стимуляції антитіла - батька.

Слід зазначити, що в даному методі при клонуванні кожне антитіло формує максимально можливу кількість клонів, яка дорівнює кількості антитіл в популяції - за рахунок цього забезпечується велика різноманітність, що підвищує ймовірність досягнення специфічності клонів якого-небудь

антигену. Станом специфічності називається стан, при якому ознаки будь-якого клону в ході мутації повністю повторили ознаки будь-якого антигену. При цьому досягається найкраща афінність між таким клоном і антигеном, антиген вважається відновленим і більше не представляється популяції антитіл. Як критерій зупинки алгоритму ВСА використовується гранична кількість поколінь антитіл.

Основними перевагами даного методу є простота реалізації і модифікації, що забезпечує його універсальність. Завдяки послідовному клонування з максимальною кількістю клонів, що виділяються кожному антитілу, існує висока ймовірність досягнення стану специфічності популяцій до завершення роботи алгоритму.

Основним недоліком даного методу є велика кількість обчислень, вироблених при визначенні рівнів стимуляції клонів, наслідком чого є збільшення часу, що витрачається на обробку однієї популяції антитіл алгоритмом.

Метод CLONALG (clonal selection algorithm) є одним з найбільш поширених універсальних ІА. Даний метод функціонує на основі моделі клонального відбору і багато в чому схожий на ВСА, проте має ряд істотних відмінностей:

- уявлення одного антигену популяції антитіл на кожній ітерації циклу відновлення антигенів;

- популяційне клонування;

- кількість клонів, що виділяються для клонування антитіл, залежить від їх афінності з поточним антигеном.

На відміну від алгоритму ВСА, в якому кожному оброблюваному антитілу представляється вся популяція антигенів, в методі CLONALG всім антитілам представляється тільки один антиген, що призводить до скорочення кількості обчислювальних операцій і скорочення витрат часу на обробку кожної популяції антитіл. При цьому вибір антигену може відбуватися випадковим чином, або по порядку проходження в популяції.

Після представлення антигену і визначення афінності для кожного антитіла, відбувається процес первинного відбору, в результаті якого з усієї популяції для подальшого клонування відбирається певна кількість антитіл з кращою афінністю, тобто, використання первинного відбору призводить до скорочення кількості клонованих антитіл. Наслідком цього є скорочення кількості обчислювальних операцій і зниження обсягів оперативної пам'яті обчислювальної системи, яка споживається алгоритмом на етапі клонування.

Слід зазначити, що в CLONALG, на відміну від BCA, використовується популяційне клонування, при якому створення клонів проводиться не для кожного окремого антитіла (як це відбувається в BCA), а для всієї множини антитіл, які пройшли етап первинного відбору. Крім того, кількість клонів, що виділяються кожному з клонованих антитіл, залежить від його афінності з поточним антигеном, наслідком чого є скорочення кількості клонів, що створюються на популяції. Це призводить до зниження ймовірності відновлення антигенів, але призводить до скорочення витрат часу.

Основними достоїнствами CLONALG є простота реалізації і модифікації, висока швидкість обробки популяцій антитіл, яка досягається за рахунок скорочення кількості обчислювальних операцій і скорочення кількості клонів, що створюються при клонуванні антитіл, відібраних в результаті первинного відбору. Крім того, в порівнянні з BCA, алгоритм CLONALG споживає менші обсяги оперативної пам'яті обчислювальної системи.

Основними недоліком CLONALG є низька, в порівнянні з BCA, ймовірність того, що стан специфічності популяцій антитіл і антигенів може бути досягнуто за обмежену кількість популяцій антитіл. Наслідком цього є накладення вимог на вибір критерію зупину алгоритму, тобто в тому випадку, якщо критерієм зупинки є гранично допустима кількість популяцій антитіл, то воно повинно бути більше кількості антигенів для того, щоб кожен антиген хоча б один раз був представлений популяції антитіл і міг бути відновлений клонами.

Алгоритм aiNet (artificial immune network) - є базовим для всіх алгоритмів, що функціонують на основі моделі штучної імунної мережі. Даний алгоритм широко використовується при вирішенні задач класифікації, кластеризації та розпізнавання образів. Метод aiNet багато в чому схожий на CLONALG, але має ряд істотних відмінностей:

- використання механізму супресії замість відбору клонів і старіння;
- формування імунної пам'яті з антитіл і клонів з кращою афінністю до пропонованого антигену;
- використання механізму додаткового розкиду антитіл в разі,
- скорочення кількості антитіл в результаті супрессии імунної мережі.

Оскільки aiNet функціонує на основі моделі штучної імунної мережі, в даному алгоритмі для редагування популяцій антитіл і клонів використовується механізм супресії, в роботі якого використовується кілька критеріїв, що визначають граничні значення супрессії:

- коефіцієнт супрессії клонів;
- коефіцієнт супрессії мережі антитіл.

Коефіцієнт супрессії використовується для скорочення кількості клонів після з мутації та подання антигену. Слід зазначити, що перед початком супрессии, з безлічі клонів виділяються об'єкти з найкращими Аффинной до пропонованого антигену. Ці клони стають клітинами імунної пам'яті. Клони, що не стали клітинами пам'яті видаляються в результаті супрессии, якщо їх афінність до представленого антигеном менше порогового значення. Що залишилися після супрессии клони додаються в популяцію антитіл.

Коефіцієнт супрессії використовується для редагування клітин пам'яті і антитіл імунної мережі. При цьому кожне антитіло або клітина імунної пам'яті визначає аффінність з усією популяцією антитіл. Антитіло або клітина пам'яті видаляється в разі, якщо його аффінність до популяції антитіл менше встановленого значення. Що залишилися після супрессії антитіла і клітини пам'яті формують нову популяцію, якій буде представлятися наступний антиген.

Додатковий розкид антитіл виробляється тільки в тому випадку, коли в результаті супресії імунної мережі видаляється занадто велика кількість антитіл, тобто кількість антитіл після супресії мережі менше початкової кількості антитіл до подання антигену. При додатковому розкиду в мережу випадковим чином додаються нові антитіла до тих пір, поки вихідна кількість антитіл в популяції не буде відновлена.

Основними перевагами алгоритму aiNet є простота реалізації і модифікації. За своїми характеристиками алгоритм aiNET можна порівняти з методом CLONALG. Слід зазначити, що даний алгоритм є одним і найбільш швидкодіючих імунних алгоритмів, наслідком чого є його широке застосування при вирішенні практичних завдань, в яких велика увага приділяється швидкості обробки даних.

Основними недоліками алгоритму aiNet є можливість втрати даних при супресії антитіл, що робить скрутним застосування даного методу для вирішення задач класифікації та кластеризації. Додаткові труднощі при відновленні антигенів створюються в результаті додаткового розкиду, при якому антитіла формуються випадковим чином. Формування антитіл випадковим чином призводить до того, що аффінності до антигенів у таких антитіл можуть бути гірше аффінності антитіл, які були видалені з популяції в результаті супресії мережі. Така організація додаткового розкиду є найбільш спірним місцем в роботі методу aiNet. Тому при вирішенні завдань групування даних або розпізнавання від додаткового розкиду антитіл випадковим чином або відмовляються, або піддають модифікації [10].

Метод RLAIS (resource limited artificial immune system, RAIN) як і алгоритм aiNet функціонує на основі моделі імунної мережі і імітує взаємодія не тільки на рівні антигенів і антитіл, а й на рівні лімфоцитів. Найбільшого поширення даний метод отримав при вирішенні задач розпізнавання образів та ідентифікації об'єктів в багатовимірному просторі ознак. Даний метод значно відрізняється від інших імунних алгоритмів. Серед основних відмінностей можна виділити:

- використання розпізнаваючих областей (ARB), які є аналогами B-лімфоцитів;

- використання універсального порога аффінності NAT, що обмежує зростання кількості антитіл в ARB;

- використання антитіл пам'яті як центри ARB;

- використання механізму соматичної гіпермутація для зміни антитіл і їх клонів всередині ARB.

Використання принципу областей розпізнавання ARB в алгоритмі RLAIS є основною відмінною рисою даного методу і визначає організацію його роботи. Кожна область розпізнавання ARB є штучним B-лімфоцитів, що містить кілька антитіл і характеризується рівнем стимуляції антигенами. Слід зазначити, що ARB конкурують між собою за антитіла, тобто антитіла або клони можуть переходити від одного ARB до іншого. Така міграція антитіл відбувається в результаті порівняння рівнів стимуляції декількох ARB, розташованих на гранично малих відстанях один від одного. Таким чином, кожна ARB прагне отримати якомога більше антитіл при клонуванні і за рахунок міграції антитіл. Для обмеження росту ARB використовується універсальний поріг аффинности NAT, який використовується при злитті ARB і міграції антитіл. У RLAIS кілька ARB можуть зливатися в одну область розпізнавання, якщо аффинность між їх центрами перевищує поріг NAT. Антитіло може перейти з однієї ARB в іншу, якщо аффинность між антитілом і центром другий ARB, або всіма її антитілами перевершує NAT і аффинность з центром першої ARB. Слід зазначити, що граничне значення NAT не є вхідним параметром і визначається наступним чином:

$$NAT_{AG} = \sum_{i=1}^{n} \sum_{\substack{j=1, \\ j \neq i}}^{n} aff_{ij} \Big/ \left[n \cdot (n-1) : 2 \right], \quad i, j \in AG$$

де aff_{ij} - афінність між i-тими і j-тими антигенами, а n – загальна кількість антигенів в популяції.

Завдяки такій конкуренції за антитіла між ARB даний метод отримав назву ресурсно-обмеженою імунної системи (мережі). Слід зазначити, що центрами ARB є клітини пам'яті, які, як і в методі aiNet є антитілами, які досягли стану специфічності антигенів, або характеризуються найкращими афінностями до представленії антигенів.

Однією з особливостей даного методу є організація мутації антитіл - в RLAIS використовується соматична гіпермутація. Оскільки всі антитіла належать одній з ARB, коефіцієнт мутації визначається не для кожного окремого антитіла, що сформував популяцію клонів, а на підставі рівня стимуляції ARB, якому належить антитіло. При цьому коефіцієнт мутації клонів обернено пропорційний рівню стимуляції ARB, до якої вони належать. Рівень стимуляції ARB антигенами визначається наступним чином:

$$sl_{ARB} = \sum_{i=1}^{k} aff_i \Big/ k$$

де k – кількість антитіл, що входять в данну ARB;

aff_i - афінність i-го антитіла до антигенів.

Слід зазначити, що використання механізму соматичної гіпермутація клонів дозволяє скоротити кількість обчислювальних операцій, що призводить до підвищення швидкодії алгоритму без втрати точності.

Основними перевагами алгоритму RLAIS є хороша точність угруповання антитіл, що дозволяє використовувати даний метод при вирішенні задач класифікації і кластеризації, висока швидкість роботи при невеликих обсягах оперативної пам'яті обчислювальної системи, необхідної для роботи алгоритму. За своїми характеристиками цей метод є одним з кращих імунних алгоритмів і часто використовується при порівнянні з іншими імунними методами як еталон [11].

Головним недоліком RLAIS є складність реалізації. Слід зазначити, що

в даному методі, на відміну алгоритму aiNet, супрессия проводиться окремо для кожної ARB, а не для всієї популяції антитіл і клонів, що створює певні труднощі в розумінні організації роботи алгоритму дослідником, які не мають доброго розуміння організації роботи моделі імунної мережі, а також створює труднощі модифікації даного алгоритму.

Алгоритм Форрест-Еспонди (NSA - negative selection algorithm) функціонує на основі моделі негативного відбору. При цьому в даному алгоритмі антигенами є «чужі» об'єкти, а антитілами - свої. Робота алгоритму NSA зводиться до двох основних етапів: формування детекторів і визначенню простору чужих об'єктів. Безліч детекторів формується шляхом клонування і мутації «своїх» об'єктів, тобто антитіл вихідної популяції. Після проведення мутації, клони визначають аффінності з антигенами і антитілами. Після мутації і визначення аффінності клони додаються в популяцію антитіл. Редагування популяції зводиться до видалення антитіл, у яких аффинность до антитіл вище аффінності до антигенів, тобто таким імунним об'єктів, які розпізнають «своє» з більшою аффінностю, ніж «чужі». Критерієм зупинки алгоритму зазвичай є гранична кількість популяцій антитіл, або досягнення стану, коли формування нових антитіл, у яких аффінность з антигенами перевершує аффінность з антитілами, неможливо. Слід зазначити, що даний алгоритм, як і більшість алгоритмів, що функціонують на основі моделі негативного відбору, використовується для вирішення вузького кола завдань, пов'язаних головним чином з організацією інформаційної безпеки і захисту від вірусів.

Основними достоїнствами алгоритму NSA є його простота можливість модифікації. Для реалізації даного алгоритму досліднику не потрібно вивчати особливості клонування або мутації, принципи організації відбору клонів або додаткового розкиду, тому NSA широко використовується для вирішення практичних завдань з організації захисту даних.

Серед основних недоліків методу NSA виділяють швидкодію, що виявляється в тому, що тимчасові витрати при обробці популяції антитіл

перевищують витрати інших імунних алгоритмів, що функціонують на основі моделі клонального відбору або моделі імунної мережі.

2.2 Роль імунних операторів в функціонуванні ШІС

Функціонування ІА можна представити у вигляді деякої послідовності операцій, вироблених над антитілами або антигенами. Такими операціями, що використовуються більшістю ІА є клонування, мутація, відбір клонів, старіння і т.д. Для позначення таких операцій, універсальних для всіх ІА, використовується поняття імунного оператора. Імунним оператором називається операція по перетворенню деякого безлічі імунних об'єктів (антитіл, антигенів, клонів, клітин імунної пам'яті і ін.), яка проводиться в одному або декількох ІА. Таким чином, роботу будь-якого імунного методу незалежно від виду поставленого завдання формально можна описати послідовним викликом імунних операторів. При цьому серед ІО, можна виділити групу операторів, завжди використовуються при навчанні ІА, незалежно від вибору імунної моделі (наприклад, оператори клонування, мутації, старіння), і групу операторів, використання яких залежить від вибору моделі ШІС або виду розв'язуваної задачі (наприклад, оператори первинного відбору, супрессии, відбору клонів, додаткового розкиду). Опис ІА у вигляді послідовності ІС дозволяє спростити схему роботи будь-якого імунного методу. Завдяки цьому підвищується простота використання ІА при вирішенні практичних завдань, підвищуються можливості модифікації імунного алгоритму, спрощується його аналіз і моделювання. Таким чином, ІА, що розглядаються в даній роботі, можуть бути описані на рівні імунних операторів. Алгоритм формально представляється як:

$$BCA(AG, AB, p) = Ag\,\mathrm{Pr}esent(AG, AB) \rightarrow \{[Cloning(ab, CL) \rightarrow$$
$$\rightarrow Mutation(CL) \rightarrow ClonSelection(AG, CL) \rightarrow$$
$$\rightarrow Apoptosys(ab, cl)]\}TermTest(p),$$

де $\text{Ag Present}(AG, AB)$ – оператор представлення антигенів антитілам;

$\text{Cloning}(ab, CL)$ – оператор клонування антитіла;

$\text{Mutation}(CL)$ – оператор мутації клонів;

$\text{ClonSelection}(AG, CL)$ – оператор відбору клонів;

$\text{Apoptosys}(ab, cl)$ – оператор старіння;

$\text{TermTest}(p)$ – перевірка критерію зупинки алгоритма;

p – кількість популяцій антитіл.

Алгоритм CLONALG можна представити у вигляді такої послідовності:

$$\begin{aligned}\text{CLONALG}(AG, AB, p) = \text{Ag Present}(AG, AB)\{&\text{FirstSelection}(ag, AB) \rightarrow \\ &\rightarrow \text{Cloning}(AB, CL) \rightarrow \text{Mutation}(CL) \rightarrow \\ &\rightarrow \text{ClonSelection}(ag, CL) \rightarrow \text{Apoptosys}(AB, CL)\}\text{TermTest}(p),\end{aligned}$$

де $\text{FirstSelection}(ag, AB, n)$ – оператор первичного отбора антитіл.

Алгоритм aiNet на рівні імунних операторів описується наступним чином:

$$\begin{aligned}\text{aiNet}(AG, AB, p, n, k, dc, ds) = \{&\text{FirstSelection}(ag, AB, n) \rightarrow \\ &\rightarrow \text{Cloning}(AB, n, CL) \rightarrow \text{Mutation}(CL) \rightarrow \\ &\rightarrow \text{MemForm}(ag, k, CL, ABM) \rightarrow \text{ClonSupr}(CL, dc) \rightarrow \\ &\rightarrow \text{MemSupr}(ABM, ds) \rightarrow \text{NetForm}(AB, CL, ABM) \rightarrow \\ &\rightarrow \text{NetSupr}(AB, ds)\}\text{TermTest}(p),\end{aligned}$$

де $\text{MemForm}(ag, k, CL, ABM)$ – оператор формування імунної пам'яті; $\text{ClonSupr}(CL, dc)$ – оператор супрессії клонів;

$\text{MemSupr}(ABM, ds)$ – оператор супрессії пам'яті;

$\text{NetForm}(AB, CL, ABM)$ – оператор формування иммуннії мережі;

$\text{NetSupr}(AB, ds)$ – оператор супрессії мрежі;

n – кількість антитіл, що проходять первичний відбір;

k – кіількість клітин пам'яті, що вибираються із популяції клонів;

dc – коефіцііїнт супрессії клонів;

ds – коефіцієт супрессії антитіл.

На рівні імунних операторів алгоритм RLAIS описується як:

$$\begin{aligned} RLAIS(AG, AB, p, dc) = NATForm(AG) \rightarrow \{ARBForm(ag, AB, ARB) \rightarrow \\ \rightarrow ARBCloning(ARB, AB, CL) \rightarrow ARBMutatio\,n(CL) \rightarrow \\ \rightarrow ARBSupr(ARB, dc) \rightarrow ARB\,Re\,size(AB, ARB)\}TermTest(p), \end{aligned}$$

де $NATForm(AG)$ – оператор визначення критерія NAT;

$ARBForm(ag, AB, ARB)$ – оператор формування вихідних ARB з популяції антитіл;

$ARBCloning(ARB, AB, CL)$ – оператор клонування антитіл в кожній області ARB;

$ARBMutatio\,n(CL)$ – оператор соматичної гіпермутації клонів; $ARBSupr(ARB, dc)$ – оператор супрессії антитіл в ARB;

$ARB\,Re\,size(AB, ARB)$ – оператор зміни розмірів областей в результаті конкуренції за клони и антитіла.

Алгоритм негативного відбору Форрест-Еспонди NSA на рівні імунних операторів представляється в такий спосіб:

$$\begin{aligned} NSA(p) = \{Cloning(ag, AB, CL) \rightarrow Mutation(CL) \rightarrow \\ \rightarrow NegativeCl\,onSelection(AG, AB, CL) \rightarrow \\ \rightarrow NetForm(AB)\}TermTest(p), \end{aligned}$$

де $NegativeCl\,onSelection(AG, AB, CL)$ – оператор негативного відбору клонів, у яких аффінності до антигенів вище аффінності до антитіл,

$NetForm(AB)$ – оператор формування иммунної мережі по результатам негативного відбору.

Використання оператора певвичного відбору $FirstSelection(ag, AB, n)$ дозволяє скоротити кількість викличних операцій та підвищити швидкість роботи алгоритму за рахунок того, що антитіла з гіршою аффінностю до антигенам не буде клонуватися. У ІА використовується кілька видів операторів первинного відбору антитіл:

- статичний первинний збір - при цьому з популяризації вихідних антитіл для клонування відбирається раніше встановлена кількість об'єктів, яка є вхідним параметром для ІА;
- відбір на основі граничної аффінності - при цьому з безлічі антитіл для подальшого клонування відбираються всі антитіла, у яких аффінність із популяцією антигенів підвищують деяке порогове значення, що визначається в ході роботи алгоритмів.

Зазначаємо, що в деяких ІА, таких як BCA, RLAIS та NSA оператор первинного відбору повідомлення не використовується, а алгоритми CLONALG та aiNet використовують статичний первинний збір для скорочення кількості клонованих антитіл.

Основним недоліком оператора статичного первинного відбору антитіл є більша чутливість ІА до кількості антител, яка буде відображатися для клонування. При невеликій кількості антитіл алгоритму для досягнення стану специфічності потребує великої кількості популяцій, у той час як при великому кількісному складі відбираються об'єктів вірогідність ризику роста вибіткових вичислень на етапах мутацій і відборів клонів, в результаті такого відбору антитела відображають недостатньо хороші аффінності і будуть підвергатися клонуванню. Після цього є збільшення часу роботи ІА. Головним достоїнством статистичного збору є простота реалізації.

Основним недоліком відбору, заснованим на використанні граничних аффінностей, є підвищення рівня складності роботи алгоритмів.

Достоїнством такого відбору є вибір оптимального обсягу об'єктів для клонування, що зменшує кількість популяційних антитіл, необхідність навчання та зменшення ризику появи вихідних вичислень.

Потім відзначаємо, що у випадку відмови від використання первинного збору відбувається зростання кількості популяцій антител, необхідних для досягнення стану специфічності, що збільшує час виконання та обсяги пам'яті, що використовуються алгоритмом.

Оператор клнування антитіл грає важливу роль у роботі ІА. Він присутній у всіх типах імунних алгоритмів незалежності від вибору моделей, на основі яких вони формуються. При клонуванні антитіла створюється безліч клонів, що приймають всі його визнання. За способом визначення кількості клонів, оператор клонування може бути:

- статичним - при цьому кількість клонів, створених антитілом, визначає збереження і не залежить від значень його аффінності з антигенами;

- пропорціональним - при цьому кількість клонів антитіла залежить від його аффінності з антигенами, в деяких випадках антитіла можуть конкурувати між собою за кількість створених клонів.

Основним недоліком статистичним клонірованням можливості формування єдиного кількості клонів антителами, що характеризуються найкращою аффінністю, та антителамами з аффінністю незначно підвищується порогове значення критерію відбору. Це може сприяти підвищенню ймовірності появи вихідних вичислень на наступних етапах роботи алгоритмів. Основним достоїнством статистичного визначення кількості клонів є його простота.

Основним недоліком статичного клонування можливість формування однакової кількості клонів антитілами, що характеризуються кращою аффінністю, і антитілами з аффінністю незначно перевищує порогове значення критерію відбору. Це може привести до підвищення ймовірності появи надлишкових обчислень на наступних етапах роботи алгоритму. Основною перевагою статичного визначення кількості клонів є його простота.

Основним недоліком пропорційного клонування є необхідність вирішення завдання визначення залежності кількості клонів, які формуються

антитілом, від значення його афінності до антигенів або будь-яких інших його ознак, що може підвищити складність реалізації алгоритму. Перевагою пропорційного визначення кількості клонів є можливість формування великої кількості клонів, що характеризуються кращими аффінностями до антигенів. Це призводить до скорочення кількості популяцій антитіл, необхідного для досягнення стану специфічності до антигенів.

2.3 Сумісність імунних операторів з різними імунними моделями і алгоритмами

Різні імунні моделі і алгоритми можуть використовувати різну кількість імунних операторів. При цьому в алгоритмах моделі клонального відбору не використовується оператор супрессии і додаткового розкиду, а в алгоритмах моделі імунної мережі не використовуються оператори відбору клонів і старіння. В алгоритмах негативного відбору, оператори добору клонів і старіння замінюються оператором негативного відбору антитіл, який за принципом роботи багато в чому нагадує оператор супрессії.

З цього випливає, що в архітектурі імунного алгоритму, який функціонує на основі теорії конального відбору, використання операторів супресії, додаткового розкиду, позитивного або негативного відбору антитіл, обмежується, тобто імунні алгоритми клонального відбору і перераховані оператори принципово несумісні. Аналогічні обмеження на використання оператора відбору клонів накладаються на алгоритми, що функціонують відповідно до принципів роботи моделі штучної імунної мережі, в якій завдання редагування популяцій антитіл вирішується оператором подвійної чи потрійної супрессії. При цьому слід зазначити, що в методі aiNet використовується потрійна супрессія, тому на різних етапах роботи алгоритму механізму супрессії піддаються клони, антитіла і клітини імунної пам'яті. З іншого боку, алгоритми, що функціонують на основі моделі негативного відбору, несумісні з операторами первинного відбору антитіл,

клонального відбору, старіння антитіл, супрессії і додаткового розкиду.

Замість цих операторів в таких методах використовується оператор негативного відбору антитіл. Винятком із загальних правил сумісності імунної моделі з операторами є метод RLAIS. Даний алгоритм функціонує на основі моделі імунних систем, але по своїй організації значно відрізняється від методу aiNET і його модифікацій. Специфіка роботи RALIS полягає в використанні ARB. При цьому робота з антитілами і клонами, що належать різним областям розпізнавання відбувається виключно в рамках кожної окремої ARB, тобто кожній області розпізнавання для визначення рівня стимуляції представляється поточний антиген. При цьому клонування, мутація і супрессия антитіл і їх клонів відбувається для антитіл кожної ARB окремо, а не для всієї популяції антитіл в цілому, як це відбувається в алгоритмах aiNET. При цьому в деяких модифікаціях RLAIS при редагуванні популяції антитіл замість оператора подвійної супрессії використовується оператор клонального відбору. Таким чином, незважаючи на те, що алгоритм RLAIS ґрунтується на принципах моделі штучної імунної мережі, завдяки використанню областей розпізнавання ARB, є модельно незалежним і може бути легко модифікований для вирішення різних завдань. З цього випливає, що на даний алгоритм для редагування популяції антитіл можуть використовуватися не тільки оператор супрессії або клонального відбору, а й оператор негативного відбору антитіл. У зв'язку з цим в деяких випадках метод RLAIS виділяють як окрему імунну модель.

3 МЕТОДИ СКЕЛЕТИЗАЦІЇ БІНАРНИХ РАСТРОВИХ ЗОБРАЖЕНЬ

3.1 Алгоритм скелетизації Зонга-Суня

T.Y. Zhang і C.Y. Suen в 1984 році запропонували власний алгоритм скелетизації бінарних растрових зображень. При асимптотично оптимальної реалізації швидкодію цього алгоритму лінійно залежить від кількості пікселів вихідного зображення.

Ідея алгоритму полягає в послідовному виконанні певного набору дій до тих пір, поки хоча б один піксель зображення змінюється після виконання цих дій.

Під час опису алгоритму найбільш зручним буде вважати, що пікселі, що відносяться до графічного поданням символу, є чорними, що залишилися пікселі - білими. Для кожного пікселя розглядаються його вісім сусідніх з ним пікселів бінарного зображення, які в свою чергу, пронумеровані за схемою, показаної на рисунку 3.1.

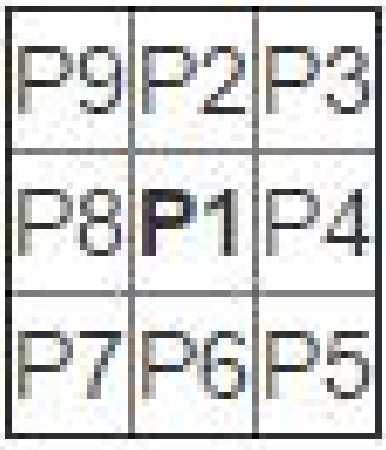

Рисунок 3.1 – Порядок нумерації сусідніх пік селів

Для пікселів, які знаходяться на кордоні, все неіснуючі сусідні пікселі покладаються білими.

Визначимо A (P1), як кількість переходів від білого пікселя до чорного при циклічному обході сусідів в порядку P2, P3, P4, P5, P6, P7, P8, P9, P2. Також визначимо B (P1) рівною кількості чорних сусідів центрального пікселя P1.

На першому кроці алгоритму потрібно додати тег всі пікселі зображення P1, які одночасно задовольняють наступним вимогам:

- піксель є чорним і має вісім сусідів;
- виконується нерівність;
- виконується рівність;
- як мінімум один з пікселів P2, P4 або P6 є білим;
- як мінімум один з пікселів P4, P6 або P8 є білим.
- наприкінці першого кроку потрібно видалити всі пікселі, які були помічені.

На другому кроці алгоритму потрібно додати тег всі пікселі зображення P1, які одночасно задовольняють дещо іншим списку вимог:

- пиксель є чорним і має вісім сусідів;
- виконується нерівність;
- виконується рівність;
- як мінімум один з пікселів P2, P4 або P8 є білим;
- як мінімум один з пікселів P2, P6 або P8 є білим.

Аналогічно першому кроці, після другого кроку алгоритму потрібно видалити всі помічені під час цього кроку пікселі.

Перший і другий кроки алгоритму потрібно виконувати до тих пір, поки хоча б один піксель позначений в ході цих дій.

Варто відзначити, що при тривіальному підході, буде потрібно багаторазовий прохід по вихідного зображення, що призведе до обчислювальної складності порядку O (P^2). Однак, якщо на кожній ітерації алгоритму, крім першої, перевіряти лише пікселі, хоча б один з сусідів яких змінився минулого ітерації, то обчислювальна складність такого алгоритму

визначається, як O (P). Реалізувати такий підхід можна з використанням абстрактного типу даних queue (черга) [13].

3.2 Алгоритм скелетизації Ву-Цая

В 1992 році Rey-Yao Wu і Wen-Hsiang Tsai запропонували власний метод скелетизації бінарного растрового зображення, що володіє високою швидкодією.

Автори запропонували підхід, при якому всі пікселі зображення потрібно розглянути лише по одному разу. Для виконання скелетизації передбачається використовувати 14 шаблонів, які наведені на рисунку 3.2.

У наведених шаблонах символи 'c', '0', '1', 'x' позначають, відповідно, розглянутий піксель, білий піксель, чорний піксель і піксель довільного кольору. Символ 'y' має особливу властивість - хоча б один з таких пікселів, при накладенні шаблона, повинен бути білим.

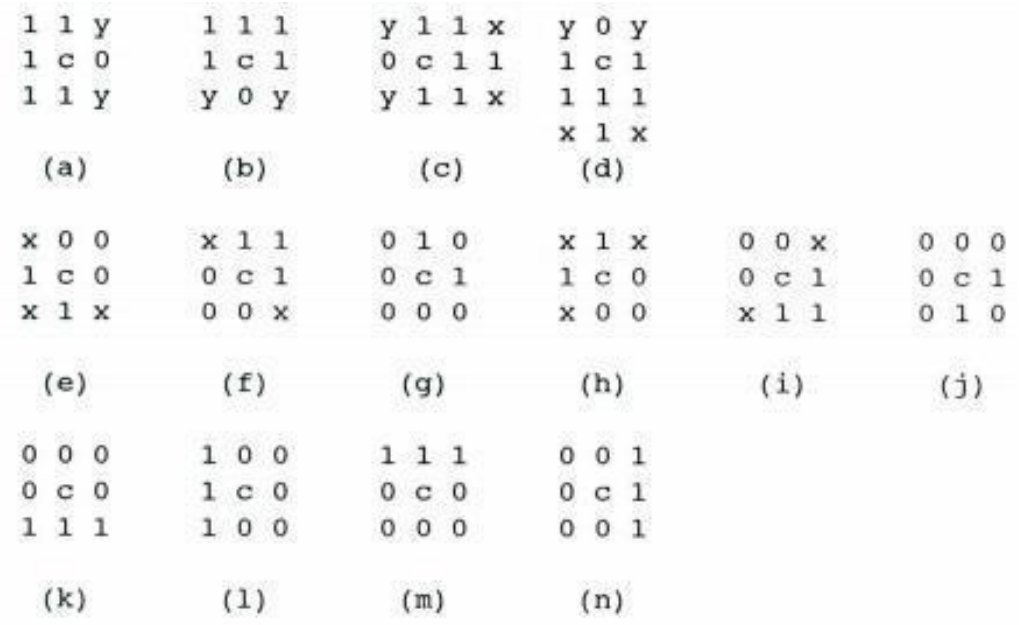

Рисунок 3.2 – Шаблони, що використовуються в алгоритмі скелетизації Ву-Цая

Кожен з наведених шаблонів послідовно застосовується для кожного з пікселів при лінійному підході по всим пікселям віхідного зображення. Піксель відаляється в разі, якщо піксель, з сусідніми йому пікселями, підходить хоча б під один з описом шаблонів. Більш того, віддалення проводитися до того, як буде розглядатися наступний піксель.

Через те, що всі пікселі зображення розглядаються рівно по одному разу, обчислювальна складність алгоритму складає O(P).

3.3 Алгоритм скелетизації Го-Холла

Співробітники університету Піттсбурга Zicheng Guo і Richard W. Hall в 1989 році запропонували підхід, чимось схожий з алгоритмом Зонга-Суня. Основна ідея алгоритму, як і в раніше розглянутих алгоритмах, ґрунтується на оцінці восьми сусідів кожного з пікселів. Автори використовують позначення сусідів по схемі, наведеній на рисунку 3.3.

$$\begin{matrix} p_1 & p_2 & p_3 \\ p_8 & p & p_4 \\ p_7 & p_6 & p_5 \end{matrix}$$

Рисунок 3.3 – Схема нумерації сусідів пікселя p в алгоритмі Го-Холла

Позначимо B (p), як кількість сусідніх пікселів рі чорного кольору. Аналогічно, позначимо C (p) дорівнює кількості різних компонент восьми-зв'язності, що є сусідами з піксельом p.

Символами «∧» і «∨» позначимо, відповідно операції кон'юнкції і диз'юнкції, в той час, як звичайні арифметичні операції додавання множення стандартно позначаються знаками «+» і «·» відповідно. Для зручності опису алгоритму введемо нове позначення N (p).

$$N(p) = \min(N_1(p),\ N_2(p)),$$

где $N_1(p) = (p_1 \vee p_2) + (p_3 \vee p_4) + (p_5 \vee p_6) + (p_7 \vee p_8)$,

и $N_2(p) = (p_2 \vee p_3) + (p_4 \vee p_5) + (p_6 \vee p_7) + (p_8 \vee p_1)$.

Фактично, кожне з значень N1 (p) і N2 (p) розбиває всіх сусідів пікселя p на чотири пари сусідніх між собою пікселів, вважаючи кількість таких пар з одним або двома чорними пікселями.

Використовуючи дані позначення можна сформулювати алгоритм Го-Холла. Будемо виконувати ітерації алгоритму до тих пір, поки чергова з них не залишить зображення без змін.

На кожній з ітерацій видаляються тільки пікселі, які відповідають кожному з наступних умов:

- C(p) = 1;
- $2 \leq N(p) \leq 3$;
- для непарного номера ітерації, умова зображена на рисунку 3.4;
- для непарного номера ітерації, умова зображена на рисунку 3.5.

Умова непарного номера ітерації алгоритму Го-Холла:

$$(p_2 \vee p_3 \vee \bar{p}_5) \wedge p_4 = 0$$

Умова парного номера ітерації алгоритму Го-Холла:

$$(p_6 \vee p_7 \vee \bar{p}_1) \wedge p_8 = 0$$

Аналогічно алгоритму Зонга-Суня, алгоритм володіє обчислюваною складністю O(P^2), але при реалізації з використанням черги обчислювальна складність оцінюється як O(P).

4 РЕАЛІЗАЦІЯ WEB ДОДАТКУ

4.1 Реалізація OCR використовуючи нативний PHP

Для тестування алгоритмів розпізнавання символів необхідно було підготувати базу зображень з графічними накресленнями символів.

Щоб обмежити безліч способів графічного представлення одних і тих же символів необхідно позначити еталонні форми накреслення. В якості таких зразків можна використовувати наступні форми, які наведено на рисунку 4.1.

Рисунок 4.1 – Еталонні форми зображення символів

Алгоритм розпочинається з головної функції parseImage – де, власне, ми отримуємо вихідні символи. Дана функція приймає назву файла, яку ми повинні передати і масив з еталонними наборами символів. Вхідні параметри зоброжені на рисунку 4.2.

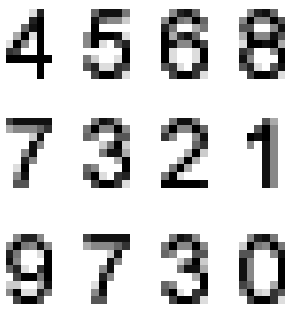

Рисунок 4.2 – Вхідні параметри

Перше, що нам необхідно зробити - це отримати данні про зображення. Для цього була створена функція extractArray, де ми за допомогою php функціїї GetImageSize, яка визначить розмір будь-якого заданого, підтримуваного зображення і поверне цей розмір разом з типом файлу і текстовим рядком height / width, яку можна буде використовувати всередині тега HTML IMG, а також поверне відповідний тип вмісту HTTP.

На даному етапі ми вже знайшли висоту, ширину і тип зображення. Наступне, що ми зробимо - це створимо саме зображення за допомогою php функції imagecreatefrompng, в даному випадку в нас було зображення типу png. Далі, за рахунок, перебору висоти і ширини - ми кожному пікселю присвоюємо колір і в кінцевому результаті ми вертаємо данні, а саме: максимальний колір, який присутній на зображенню, висота, ширина, колір заднього фону (background) та зображення, яке представленно в виді двохвиміного масиву - де кожний елемент цього масиву - це колір зображення, по-піксельно.

Наступним кроком в моїй реалізації було розбиття вхідного зображення на лінії зі збереженням змісту в цих лініях. Потрібно більш детально розглянути функцію parseWordLines, в цьому методі ми перебираємо отриману висоту зображення і на кожному кроці, ми отримуємо рядок з двохвимірного масиву і дисперсію. Якщо виконується умова, що дисперсія є додатньою, ми в масив з лініями додаємо відповідний рядок – де спрацювала ця умова.

Після того, як ми отримали лінії з відповідним змістом, ми переходимо до методу parseLetters, де ми перебираємо отриману ширину зображення і на кожному кроці, ми отримуємо рядок з двохвимірного масиву і дисперсію. Якщо виконується умова, що дисперсія є додатньою, ми в масив з символами додаємо відповідний рядок - де спрацювала ця умова. І останнім кроком цього методу є отримання транспонованої матриці на основі отриманих літер.

В заключній частині алгоритму ми проводимо наступні маніпуляції: Перебираємо наш глобальний масив з еталонними символами і літери які були отримані з методу parseLetters, далі, за допомгою функції flattenMatrix ми отримуємо літеру в вигляді одновимірного масиву, яка знаходиться в нашому глобальному масиві еталонних літер. Потім ми провіряємо отриману літеру за допомоги функції testChars, яка приймає наступні параметри: літера, глобальний масив літер. Ми перебираємо глобальний масив літер, де на кожній ітерації викликаємо функцію testChar, за допомгою якої, ми отримуємо дисперсію шлязом перебору висоти та ширини літери і саму дисперсію вираховуємо за допомоги функції diffChar, яка виглядає наступним чином:

```
function diffChar($letter, $char, $xDiff, $yDiff, $x, $y)
{
    global $color_diff;

    //translate $x, $y to relative coords
    $xRelativeCoordinate = ceil($x * $xDiff);
    $yRelativeCoordinate = ceil($y * $yDiff);

    $valM[]             =             abs(($letter[$y][$x])            -
($char[$yRelativeCoordinate][$xRelativeCoordinate]));

    $valM[]  =  abs(($letter[$y][$x])  -  ($char[$yRelativeCoordinate  +
1][$xRelativeCoordinate]));
    $valM[]  =  abs(($letter[$y][$x])  -  ($char[$yRelativeCoordinate  -
1][$xRelativeCoordinate]));
    $valM[]             =             abs(($letter[$y][$x])            -
($char[$yRelativeCoordinate][$xRelativeCoordinate + 1]));
    $valM[]             =             abs(($letter[$y][$x])            -
($char[$yRelativeCoordinate][$xRelativeCoordinate - 1]));

    $valM[]  =  abs(($letter[$y][$x])  -  ($char[$yRelativeCoordinate  +
1][$xRelativeCoordinate + 1]));
    $valM[]  =  abs(($letter[$y][$x])  -  ($char[$yRelativeCoordinate  -
1][$xRelativeCoordinate - 1]));
    $valM[]  =  abs(($letter[$y][$x])  -  ($char[$yRelativeCoordinate  +
1][$xRelativeCoordinate - 1]));
    $valM[]  =  abs(($letter[$y][$x])  -  ($char[$yRelativeCoordinate  -
1][$xRelativeCoordinate + 1]));

    return min($valM);
}
```

Приклад 4.1 – функція diffChar

Фінальним кром є виведенння результатів на екран, за допомогою функції: printOutputHtml, де ми отримуємо результат зображений на рисунку 4.3, в блоці Parsed content: данні виведені наступним чином:

X(YY)

де X – число;

Y – відсоток довіри.

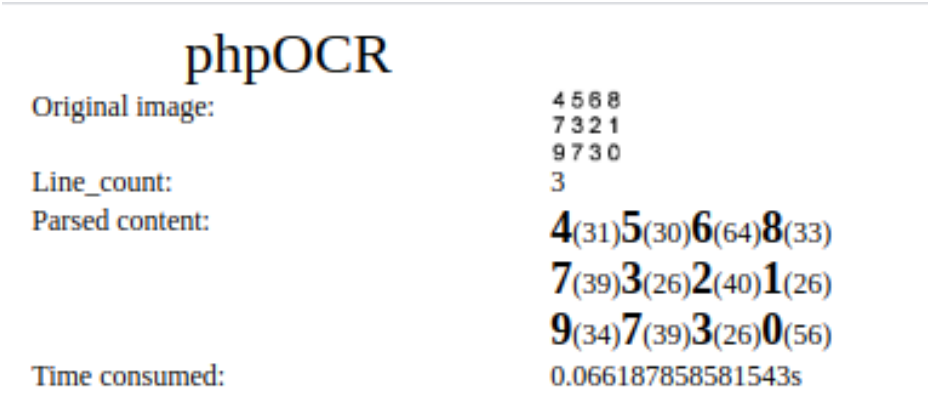

Рисунок 4.3 – Результат функції printOutputHtml

5 РЕАЛІЗАЦІЯ API

5.1 Опис документації

При проектуванні сучасних програмних систем часто входить завдання узгодження та розробки інтерфейсів для взаємодії їх компонентів один з іншим. В останнє десятиліття величезної популярності та розвитку отримали SPA та товсті мобільні додатки взаємодії з сервером через інтерфейси API.

Якщо раніше розроблена інтерактивна веб-сторінка, що виникає через поетапний правовий код кодової серверної сторони для генерації HTML-розміток з її подальшою передачею браузерного клієнта, то тепер розробляються динамічні веб-додатки, розміщені в стороні створення єдиного API-сервісу та паралельних розробок багатьох додатків (у тому числі та SPA) з цим API як з головним джерелом даних.

Такий підхід дозволяє більш зручно розподіляти завдання, організовувати команди, спеціалізуватись лише на конкретних технологіях (привертати більшу кількість узагальнених спеціалістів), організовувати паралельну розробку на самих перших етапах, а також дозволяє створювати єдину точку комунікацій - інтерфейс API.

Така єдина точка комунікації вимагає формального та однозначного визначення, цим документом є специфікація API.

Один з найпопулярніших протоколів для обміну повідомленнями між (мікро) сервісами - це REST.

Проблема в тому, що REST не є лише описаним протоколом. Це означає, що клієнт повинен знати конкретну комбінацію URL-адреси, методу HTTP та формату відвета.

У деяких випадках також необхідно знати також формат тексту запрошення. Зазвичай реалізація інтерфейсу REST базується на інших принципах та традиціях, прийнятих у вашій організації.

У будь-якому випадку кінцеві точки REST завжди повинні бути описані в одному конкретному документі, доступному для всіх інших розробників.

Доменні моделі прикладних областей не повинні обов'язково відповідати моделям, описаним у специфікаціях API. Їх можливі цільові співпадіння з структурою класів у коді клієнтських додатків або із структурою схем БД вводиться скорее для розподілу процесів розробки.

Документацію слід легко читати, писати, розподіляти, генерувати, виправляти, оновлювати та читати.

Swagger – це фреймворк та специфікація для визначення REST API у форматі, дружньому для користувача та комп’ютера (у нашому випадку JSON або YAML).

Тому описав відповідний метод для передачі зображення наступним чином:

```
swagger: "2.0"
info:
  description: ""
  version: "1.0.0"
  title: "Nure"
basePath: "/api/v1"
schemes:
  - "http"
  - "https"
securityDefinitions:
  bearerAuth:
    type: apiKey
    name: Authorization
    in: header
security:
  - bearerAuth: []
paths:
  /textract:
    post:
      tags:
        - "Captcha"
      summary: "Get text from image"
      operationId: "aws"
      produces:
        - "application/json"
      parameters:
        - name: "photo"
          in: "formData"
          description: "Photo to upload"
```

```
required: false
type: "file"
```

Приклад 5.1 – Опис документації

В самому браузері це буде виглядатив більш зручному вигляді для розробників, яке зображене на рисунку 5.1.

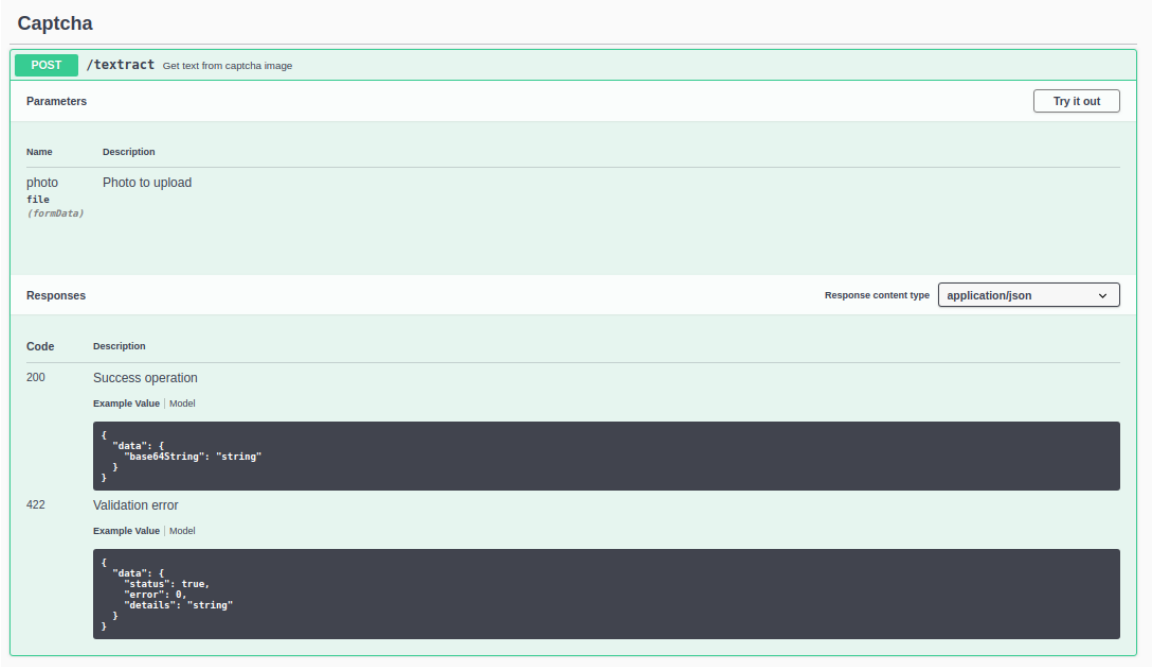

Рисунок 5.1 – Опис документації

5.2 AWS Textract

Amazon Textract дозволяє легко додавати виявлення та аналіз тексту документа до ваших програм. API розпізнавання тексту Amazon Textract може виявляти текст у різноманітних документах, включаючи фінансові звіти, медичні записи та податкові форми. Для документів зі структурованими даними можна використовувати API аналізу документів Amazon Textract для вилучення тексту, форм та таблиць.

Amazon Textract базується на тій самій перевіреній, масштабованій

технології глибокого навчання, яку розробляли вчені з питань комп'ютерного зору Amazon для щоденного аналізу мільярдів зображень та відео. Для його використання вам не потрібні знання машинного навчання. Amazon Textract включає прості, прості у використанні API, які можуть аналізувати файли зображень та файли PDF. Amazon Textract завжди вчиться на нових даних, і ми постійно додаємо нові функції до служби.

Нижче наведено найбільш поширені випадки використання Amazon Textract:

Створення інтелектуального індексу пошуку - Amazon Textract дозволяє створювати бібліотеки тексту, які виявляються у файлах зображень та PDF.

Використання інтелектуального вилучення тексту для обробки природної мови (NLP) - Ви можете використовувати Amazon Textract для виділення тексту у слова та рядки. Він також групує текст за клітинками таблиці, якщо ввімкнено аналіз таблиці документів Amazon Textract. Amazon Textract надає вам контроль над тим, як текст групується як вхід для NLP.

Прискорюючи збір та нормалізацію даних з різних джерел - Amazon Textract дозволяє отримувати текстові та табличні дані з найрізноманітніших документів, таких як фінансові документи, звіти про дослідження та медичні записки. За допомогою API API Textract Analyze Document ви можете легко та швидко витягувати неструктуровані та структуровані дані з ваших документів.

Автоматизація збору даних із форм - Amazon Textract дозволяє отримувати структуровані дані з форм. За допомогою API аналізу тексту Textract ви можете вбудувати можливості вилучення в існуючі робочі процеси, щоб дані користувача, які надсилаються через форми, могли бути витягнуті у придатний для використання формат.

Деякі переваги використання Amazon Textract включають:

Інтеграція розпізнавання тексту документа у ваші програми - Amazon Textract усуває складність вбудови можливостей розпізнавання тексту у ваші

програми, роблячи потужний і точний аналіз за допомогою простого API. Вам не потрібен комп’ютерний зір чи досвід глибокого навчання, щоб використовувати текст текстового документа Amazon Textract. За допомогою текстових API Amazon Textract ви можете легко вбудувати виявлення тексту в будь-яку веб-програму, мобільний пристрій або підключений пристрій.

Масштабований аналіз документів - Amazon Textract дозволяє швидко аналізувати та витягувати дані з мільйонів документів, що може прискорити прийняття рішень.

Невисока вартість - за допомогою Amazon Textract ви платите лише за аналізовані документи. Немає мінімальних зборів або попередніх зобов'язань. Ви можете розпочати роботу безкоштовно та заощадити більше, коли ростете, за допомогою багаторівневої моделі ціноутворення Amazon Textract.

За допомогою синхронної обробки Amazon Textract може аналізувати односторінкові документи для програм, де затримка є критичною. Amazon Textract також забезпечує асинхронні операції для розширення підтримки багатосторінкових документів [7].

Наприклад, якщо ви створюєте мобільний додаток для збору даних з односторінкового документа, що може вимагати подальшого залучення користувачів. Якщо завантажене зображення занадто темне, програма може запропонувати знову сфотографувати його із увімкненим спалахом камери. Синхронний виклик був би доречним, оскільки користувач чекає відповіді. Однак якщо ви створюєте додаток для індексування документів, що мають різну кількість сторінок, ви можете надіслати одну і ту ж односторінкову форму на асинхронний виклик, щоб усі ваші документи могли оброблятися в одному робочому процесі.

5.3 Реалізація методу

Перш за все нам потрібно провалідувати данні які ми отримуєм, відповідно до пунктів:

- зображення не повинно бути більшим ніж 5мб;
- параметр photo повинен обов’язково існувати;
- зоброження повинно бути типів: jpeg, jpg, png.

Резалізація зображена в прикладі 5.2.

```
return [
            'photo' => [
                'required',
                'mimes:jpeg,jpg,png',
                'max:5120'
            ]
        ];
```

Приклад 5.2 – Реалізація валідації

Переходимо до методу get(PhotoRequest $request), де саме ми отримує данні з зображення. В цьому випадку ми передаєм наступний файл (рисунок 5.2).

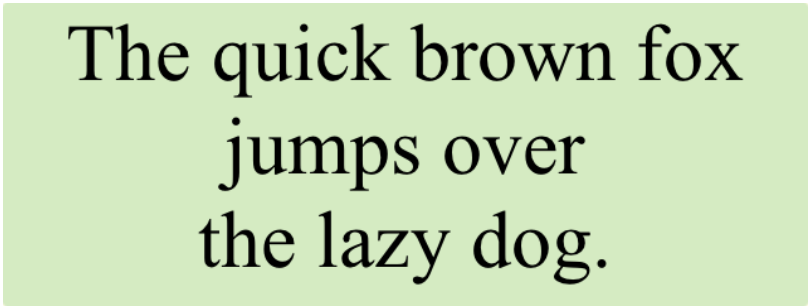

Рисунок 5.2– Вхідний параметр

Перш за все нам потрібно отримати з’єднання з сервісом Textract – це продемонстровано в прикладі 5.3:

```
return new TextractClient([
    'version' => config('aws.version'),
    'region' => config('aws.region'),
    'credentials' => [
        'key' => config('aws.credentials.key'),
        'secret' => config('aws.credentials.secret'),
    ]]);
```

Приклад 5.3 – Ініціадізація підключення

Потім потрібно підготувати данні для розпізнавання зображення, відповідно до прикладу 5.4:

```
$contents = file_get_contents($request->photo);

return [
    'Document' => [
        'Bytes' => $contents
    ],
    'FeatureTypes' => ['FORMS', 'TABLES'],
];
```

Приклад 5.4 – Данні для розпізнавання зображення

І в заключному етапі потрібно визвати функцію analyzeDocument, яка має request, що продемонстрований в прикладі 5.5:

```
{
   "Document": {
      "Bytes": blob,
      "S3Object": {
         "Bucket": "string",
         "Name": "string",
         "Version": "string"
      }
   },
   "FeatureTypes": [ "string" ],
   "HumanLoopConfig": {
      "DataAttributes": {
         "ContentClassifiers": [ "string" ]
      },
      "FlowDefinitionArn": "string",
      "HumanLoopName": "string"
   }}
```

Приклад 5.5 – Приклад реквесту

Дані які ми отримали передаємо в функцію make TextractResource. Результатом ми отримаєм об`єкт, що продемонстрований в прикладі 5.6:

```
{
    "AnalyzeDocumentModelVersion": "1.0",
    "Blocks": [
        {
            "BlockType": "WORD",
            "Confidence": 99.99360656738281,
            "Geometry": {
                "BoundingBox": {
```

```
                    "Height": 0.20916803181171417,
                    "Left": 0.07770224660634995,
                    "Top": 0.05924789234995842,
                    "Width": 0.15940335392951965
                },
                "Polygon": [
                    {
                        "X": 0.07770224660634995,
                        "Y": 0.05924789234995842
                    }
                ]
            },
            "Id": "65040915-c354-4a18-a9e0-a21f3f7ba7ab",
            "Text": "The",
            "TextType": "PRINTED"
        },
...
```


Приклад 5.6 – Результат виконання методу analyzeDocument

Отриманий response матиме типи параметрів, які можна побачити в прикладі 5.7:

```
{
   "AnalyzeDocumentModelVersion": "string",
   "Blocks": [
      {
         "BlockType": "string",
         "ColumnIndex": number,
         "ColumnSpan": number,
         "Confidence": number,
         "EntityTypes": [ "string" ],
         "Geometry": {
            "BoundingBox": {
               "Height": number,
               "Left": number,
               "Top": number,
               "Width": number
            },
            "Polygon": [
               {
                  "X": number,
                  "Y": number
               }
            ]
         },
         "Id": "string",
         "Page": number,
         "Relationships": [
            {
               "Ids": [ "string" ],
               "Type": "string"
            }
         ]
      } ]
}
```

Приклад 5.7 – Типи параметрів

ВИСНОВКИ

У процесі виконання роботи її мета була досягнута, а завдання вирішені. У роботі було реалізовано алгоритм розпізнавання тексту на основі імунного підходу, також був застосований сервіс aws textract.

В першому розділі роботи було розглянуто етапи розвитку технології OCR, проаналізовано теоретичні питання стосовно роботи ocr технології.

В другому розділі був розглянутий імунний підхід, а саме особливості імунних моделей та алгоритмів. Також була розглянута роль імунних операторів в функціонуванні штучної імунної системи і важливе питання, яке також було розглянуто - це сумісність імунних операторів з імунними моделями.

В третьому розділі було проаналізовано теоретичні відомості про методи скелетинізації бінарних растових зображень, а саме наступні алгоритми: Зонга-Суня, Ву-Цая, Го-Холла.

В четвертому розділі була виконана реалізація власного OCR використовуючи php, опираючись на імунний підхід.

В п’ятому розділі був виконаний опис API документації і розробка власног методу, використовуючи сервіс амазона aws textract в поєднанні з фреймворком Laravel.

Отже, було реалізовано власну OCR систему та розроблено API. Можна зробити висновки, що для більш вузької спецілазації, доречно сформувати власний еталонний набір символів і реалізовувати власну OCR, якщо ж задача стоїть більш глобальна, на мою думку, краще застосувати сторонні сервіси або ж бібліотеки, які використовують більш складні алгоритми для розпізнавання тексту з зображення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Кораблев Н.М. Использование аффинности на основе Манхэттенского расстояния в методах клонального отбора / Н.М. Кораблев, А.А. Фомичёв, С.Ю. Ломаненко. Тези доповідей. – Київ–Полтава–Катовице–Париж–Білгород– Черкаси–Харків, 2014. – С. 72. 23.

2. Кораблев Н.М. Повышение быстродействия алгоритмов, функционирующих на основе модели иммунной сети / Н.М. Кораблев, А.А. Фомичёв, А.Ю. Арутюнов // Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: Матеріали четвертої міжнар. наук.-техн. конференції, 4-5 грудня 2014 р. – Полтава: ПНТУ; Баку: ВА ЗС АР; Бєлгород: НДУ «БєлДУ»; Кіровоград: КЛА НАУ; Харків: ДП «ХНДІ ТМ», 2014. – С. 38. 24.

3. Кораблев Н.М. Повышение скорости иммунного обучения при использовании минимального ограничительного порога при клонировании / Н.М. Кораблев, А.А. Фомичёв, О.Ю. Крук // Сучасні напрями розвитку інформаційнокомунікаційних технологій та засобів управління: Матеріали п’ятої міжнар. наук.- техн.конференції, 23-24 квітня 2015 р. – Полтава: ПНТУ; Баку: ВА ЗС АР; Бєлгород: НДУ «БєлДУ»; Кіровоград: КЛА НАУ; Харків: ДП «ХНДІ ТМ», 2015. – С. 28. 25.

4. Кораблев Н.М. Адаптивная модель автоматической классификации объектов на основе иммунного подхода / Н.М. Кораблев, А.А. Фомичёв // Інформаційні та моделюючі технології: Матеріали всеукраїнської науково-практичної 18 конференції – ІМТ-2015. – Черкаси, 2015. – С. 73

5. ABBYY FineReader [Електронний ресурс] – 2018. – Режим доступу до ресурсу: https://www.abbyy.com/en-ee/finereader/

6. Toward Predictive [Електронний ресурс] – 2018. – Режим доступу до ресурсу: https://openreview.net/pdf?id=H1u8fMW0b

7. Amazon Textract Developer Guide [Електронний ресурс] – 2020. – Режим доступу до ресурсу: https://docs.aws.amazon.com/textract/latest /dg/textract-dg.pdf

8. Timeline of OCR [Електронний ресурс] – 2018. – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Timeline_of_optical_character_recognition

9. HPE Haven OnDemand [Електронний ресурс] – 2015. – Режим доступу до ресурсу: http://www.havenondemand.com

10. Ершов К.С. Анализ и классификация алгоритмов кластеризации / К.С. Ершов, Т.Н. Романова // Новые информационные технологии в информационных системах, 2016. – № 1. – С. 274-279.

11. Жетимекова Г.Ж. Нечеткая классификация с использованием нечеткого анализа кластеризации / Г.Ж. Жетимекова // Вестник КГУСТА. – 2014. – № 4. – С. 117-120.

12. Кораблев Н.М. Использование иммунной модели клонального отбора для кластеризации объектов / Н.М. Кораблев, А.А. Фомичев, Д.Н. Соловьев // Інтелектуальні системи прийняття рішень і проблеми обчислювального інтелекту. Матеріали міжнар. наук. конференції. – Херсон: Вид-во ФОП Вишемирський В.С., 2018. – С. 234-236.

13. Липкина, А. Распознавание текста по структуре скелета букв / А. Липкина. – М.: МГУ им. Ломоносова, 2018. – С. 31.

14. Захожай О.І. Основні аспекти структурної організації комбінованих систем розпізнавання образів / О.І. Захожай, Ю.Е. Паеранд // Вестник ХНТУ №1 (44). – Херсон: «Олди-Плюс». – 2012 – С. 221-225.