УДК 004.415.2

Є. Я. Ваврук, Д. О. Кушнір
Національний університет “Львівська політехніка”,
кафедра електронних обчислювальних машин

СИСТЕМА РОЗПІЗНАВАННЯ ТА ПЕРЕКЛАДУ ТЕКСТОВОЇ ІНФОРМАЦІЇ В МОБІЛЬНИХ ДОДАТКАХ З ВИКОРИСТАННЯМ БІБЛІОТЕКИ MICROSOFT COGNITIVE OCR

Розглянуто програмні засоби обробки як рукописного, так і друкованого тексту з подальшим його перекладом на мобільних платформах Android та IOS. Наведено метод реалізації повністю кросплатформних рішень для складних мобільних систем. Запропоновано реалізацію системи на базі алгоритму обробки тексту за допомогою бібліотеки Microsoft Cognitive OCR, наведено діаграму класів взаємодії модулів системи на основі технологій машинного навчання. Забезпечено кросплатформне рішення для мобільних систем Android та IOS. Досліджено ефективність розпізнавання різних шрифтів, написаних різними мовами, та виведено вірогідність правильного розпізнавання слів відносно кількості символів у кожному тесті.

Ключові слова: біблотека обробки зображень, кросплатформність, програмні інтерфейси, машинне навчання.

E. Vavruk, D. Kushnir
Lviv Polytechnic National University,
Computer Engineering Department

MOBILE SYSTEM FOR TEXT RECOGNITION AND TRANSLATION WITH USING MICROSOFT COGNITIVE OCR

The solution of handwritten and printed text processing problem with subsequent translation in such mobile platforms like Android and IOS is proposed. It was demonstrated method of fully cross-platform solutions development for large mobile systems. It was implemented system with the base on general algorithm of text recognition and processing using Microsoft Cognitive OCR, and illustrated main system modules communication with basics on machine learning, using class diagram. Cross-platform solution for Android and IOS mobile systems was provided. It was investigated different font types, which were used in recognized text. Also set of different language texts was investigated and probability of correct recognition was calculated.

Key words: image process library, cross-platform solutions, program interfaces, REST technologies, machine learning.

Вступ

Широке використання сучасних мобільних засобів, збільшення їх функціональних можливостей забезпечує розв’язання нового класу задач. Однією з них є задача виділення текстової інформації з фотопотоку в друкованому та рукописному вигляді, її розпізнавання та переклад на інші мови. Розроблено ряд програмні засоби реалізації цієї задачі [1–3]. Проте залишається проблема забезпечення

якості обробки залежно від таких факторів, як: колір, шрифт, відстань між символами, мова та тип (друкований чи рукописний) тексту. Також недостатньо поширеною є обробка україномовного тексту.

Сьогодні для розпізнавання текстової інформації використовують сучасні технології. Більшої популярності набирає ідея використання машини навчання, оскільки ця технологія дозволяє отримати максимальний результат, використовуючи тільки веб-технології, такі як REST (архітектурний стиль для розподілених гіпертекстових систем).

Разом з тим, на зростаючому ринку мобільного програмного забезпечення переважають нативні додатки, тобто розроблені під певну платформу. Незважаючи на свою поширеність, вони мають певні недоліки, основним з яких є їх використання на пристроях тільки з певною платформмою (операційною системою).

Основною задачею системи є її використання для найпоширеніших мобільних платформ та мов обробки та перекладу тексту.

Беручи до уваги вищенаведене, можна стверджувати, що робота у цьому напрямку є своєчасною та актуальною.

Огляд літературних джерел

У загальному випадку розв'язання поставленної задачі вимагає таких кроків:

- Формування зображення тексту і передача його у мобільний додаток.
- Розпізнавання друкованого чи рукописного тексту.
- Переклад тексту на інші мови.
- Забезпечення кросплатформного рішення для мобільних систем Android та IOS.

Оскільки метою цієї роботи є реалізація і дослідження розпізнавання та перекладу тексту, шляхи виконання першого кроку розглядатися не будуть. Слід зауважити, що якість зображення суттєво впливає на кінцевий результат (відповідність текстової інформації оригіналу, точність відтворення, час опрацювання тощо).

Існує багато систем, які підтримують обробку як рукописного, так і друкованого тексту. Основними з них є:

Anyline SDK [1]. Забезпечується підтримка багатьох мов програмування: Java (Android), Objective-C & Swift (IOS), C# (Xamarin), Javascript (Cordova, React-Native). Спеціалізується на використанні в мобільних додатках. Підтримує режим "near real-time" (аналіз відеопотоків у режимі близькому до реального часу). Недоліком системи є її дорожнеча порівняно з іншими аналогічними системами. Перевагою є можливість обробки в реальному часі.

Microsoft Cognitive API [2]. Окрім обробки тексту із зображень, підтримує: аналіз облич, емоцій, визначення ключових слів, які характеризують зображення тощо. Дозволяє швидко обробляти вхідний текст. Підтримується понад 25 словників мов та автоматичне розпізнавання мови тексту, зокрема і рукописного, розміщенного на зображенні документів, чеків, покажчиків.

Tesseract [3]. Є "вільною комп'ютерною програмою", що дозволяє модифікувати в існуючі алгоритми обробки, покращуючи їх. Є однією з найуживаніших систем для обробки тексту з зображення, підтримуючи зокрема і україномовний текст. Підтримує обробку рукописного тексту. Недоліком системи є достатньо низький рівень розпізнавання порівняно з іншими системами.

Широке впровадження нового програмного забезпечення і збільшення кількості існуючих платформ (Android, IOS, Windows Phone, BlackBerry тощо) ставить перед розробниками нові вимоги. Тому все популярнішим стає використання кросплатформної розробки додатків для сучасних пристроїв.

Поступово на зміну класичним мовам програмування (Java, C/C++ та іншим) прийшли так звані SDK – комплект розробника програмного забезпечення, який суттєво спрощує роботу програмістові завдяки використанню створених іншими розробниками бібліотек. Найпопулярнішими SDK є: Android SDK, iPhone SDK, Adobe Flex, комплект розробника Java, Windows Phone SDK, які орієнтовані на нативні платформи. Тому використовуються техногології мультиплатформної розробки мобільних додатків. Головними з них є [4]: Xamarin, Adobe Phone Gap, Appcelerator Titanium. Опишемо їх детальніше.

Appcelerator Titanium [5] – платформа для створення мобільних і десктопних кросплатформних додатків на JavaScript (HTML + CSS), підтримує три платформи: Android, IOS і BlackBerry. Основними перевагами є повна автоматизація процесу тестування, детальна інформація про помилки у вихідному коді і способи їх усунення, великий магазин плагінів. Appcelerator Titanium є одночасно і хмарною платформою для побудови та поширення додатків.

Adobe Phone Gap [6] – технологія для реалізації кросплатформної розробки мобільних додатків. Позиціонується як безкоштовний Фреймворк з відкритим кодом, дозволяє реалізовувати додатки на 8 основних мобільних платформах (серед них Android, IOS, Windows Phone). При цьому код може писатися простими мовами, такими як HTML, JavaScript, CSS, що збільшує число розробників, які не мають достатніх знань мов вищого рівня.

Xamarin [7] дозволяє створювати нативні додатки під такі мобільні платформи, як: Android, IOS, Windows, BlackBerry, використовуючи мову C#. Наявний практично повний доступ до нативних (рідних) методів кожної з платформ, з можливістю поєднання з іншими додатками програмної платформи .NET Framework.

Для втілення повноцінної інтерпретації платформи .NET з підтримкою багатьох платформ використовується проект з відкритим кодом Mono. До його складу входять: мова C# – dmcn, середовища виконання .NET – mono (з підтримкою JIT – динамічної компіляції), та mint (без підтримки JIT), власний відлагоджувач, а також набір різних біблотек (наприклад, WinForms та ASP.NET). Отже, додатки, написані мовою C#, можна виконувати на відмінних від Windows операційних системах (наприклад, UNIX). Вищесказане зумовлює використання цієї платформи для розроблення поставленого завдання.

Найбільш подібною до розроблюваної кросплатформної системи обробки тексту зі зображення та його перекладу є *Google Translate* [8], де підтримуються 103 мови перекладу та 37 мов для обробки зображення.

Основними недоліками є складність процесу визначення мови тексту та недостатня швидкість розпізнавання. Також у таких системах недостатньо уваги приділено комлпексному підходу до розпізнавання та перекладу тексту.

Ще однією проблемою є підтримка усіх нативних біблотек, оскільки не всі біблотеки чи методи можливо зробити кросплатформними через їхню специфічність.

Тому розроблювана система має мати деякі переваги, наприклад, автоматичне визначення мови тексту на зображенні, швидкість обробки, яких досягають за допомогою машини навчання Azure від Microsoft.

Microsoft Cognitive API [9] оснований на машинному навчанні, яке реалізовано Microsoft за допомогою проекту Azure. За таким підходом використовують багато обчислювальних потужностей, які розташовані віддалено, для обробки текстової інформації. Єдиним обмеженням такої системи є пропускна здатність та якість мережі, до якої підключений пристрій користувача. Тим не менше це набагато ефективніше, ніж використання потужності лише одного пристрою.

Постановка завдання

Розробити комлексний підхід до розв'язання задачі, а саме: дослідження проблеми кросплатформності нативних функцій для кожної з платформ розробки (Android та IOS). Дослідити бібліотеку, яка не піддається кросплатформності, та реалізувати інтерфейс, який перевизначатиме її для кожної платформи. Проаналізувати алгоритм роботи модуля обробки текстової інформації та навести діаграму класів усієї системи. Дослідити використання розподілених технологій обміну інформацією (REST) та використання машини навчання для обробки порівняно з використанням локальних систем обробки текстової інформації.

Розроблення системи розпізнавання та перекладу тексту

Проектуючи систему, необхідно виконати такі етапи: написання високорівневих інтерфейсів до операційних систем для кожної з розроблюваних мобільних платформ, написання коду модулів обробки та перекладу із врахуванням специфіки мобільного середовища.

На рис. 1 наведено структурну схему розробленої системи розпізнавання та перекладу тексту. Ця схема реалізовує наведені вище кроки розроблення поставленого завдання.

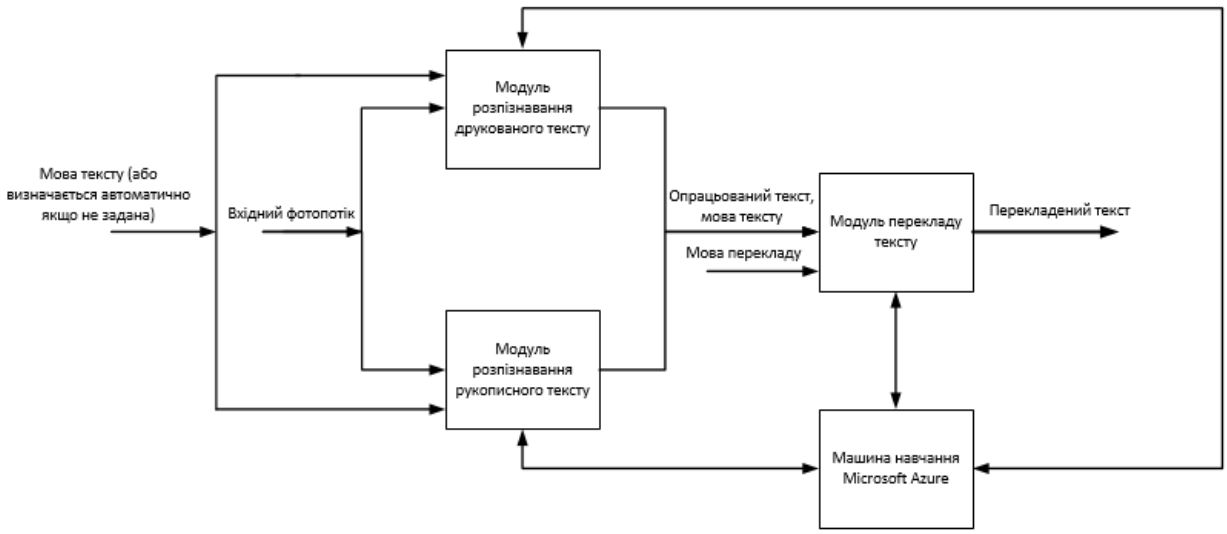

Рис. 1. Структурна схема системи розпізнавання та перекладу тексту зі зображення

Як видно зі схеми, фотопотік, отримуваний з мобільного пристрою (IOS або Android), надходить до одного з модулів, потім вхідні дані надходять на сервер машини навчання засобами REST-технологій. Проблему кросплатформності та реалізацію обробки текстової інформації описано нижче.

1. Розв'язання задачі кросплатформності для нативних бібліотек Android та IOS

Оскільки додатки є повністю нативними, то і спосіб їх компіляції так само повинен бути рідним для кожної з платформ. Для розробки використано платформу Xamarin, яка дозволяє використовувати окремий компілятор для кожної з підтримуваних платформ, використовуючи при цьому апаратні ресурси конкретної платформи. Розроблену схему компляції наведено на рис. 2.

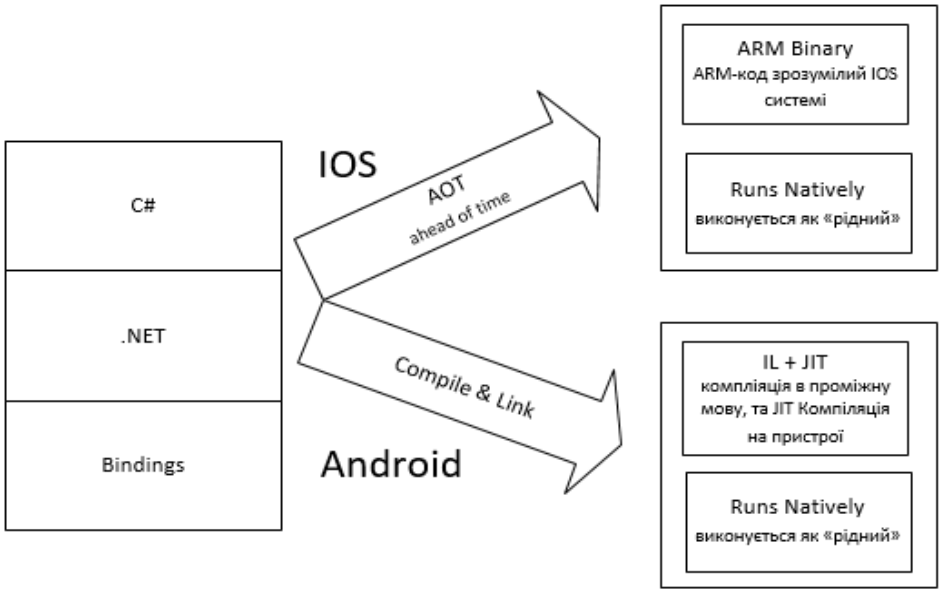

Рис. 2. Схема компіляції мобільного додатка на різних платформах (Android, IOS)

Детальніше опишемо компляцію для кожної з платформ.

- *Компіляція на Android*

Компіляція додатків на платформі Android відбувається за допомогою регістрової віртуальної машини Dalvik. Аналогічно працює середовище для байт-коду CLR (Common Language Runtime) в .NET. Така компіляція отримала назву компіляції на льоту (Just in time compilation).

Потім транслюється C# код у проміжний байт-код, який буде зрозумілий віртуальній машині Mono. Ця машина також додається в архівний файл компільованого додатка. Як Mono, так і Dalvik під час запуску програми працюють синхронно, обмінюючись даними через механізм високорівневих інтерфейсів.

- *Компіляція на IOS*

У системах IOS компіляцію використовують перед виконанням Ahead-of-Time compilation. Транслятор не потребує додаткового виділення пам'яті та виконується перед виконанням програми. У Xamarin використовують спеціально розроблений AOT компілятор Mono. Для нього програмний код повинен бути скомпільований у машинний.

Розроблено такий програмний інтерфейс та його використання для всіх нативних бібліотек:

```
public interface PCL_Translator
{
        string Translate(string sourceText, string sourceLanguage, string targetLanguage,
        string key);
}
string translatedwords = DependencyService.Get<PCL_Translator>().Translate(
words,sourceLang,destLang, Utils.GenerateRandomKey(Data.translationKeys)) + " ";
```

Кожен інтерфейс перевизначається відповідно до платформи, який запускається відповідним компілятором.

Розроблено схему системи (рис. 3). PCL_Translator та PCL_ClipBoard є програмними інтерфейсами, які перевизначають методи для кожної з платформ.

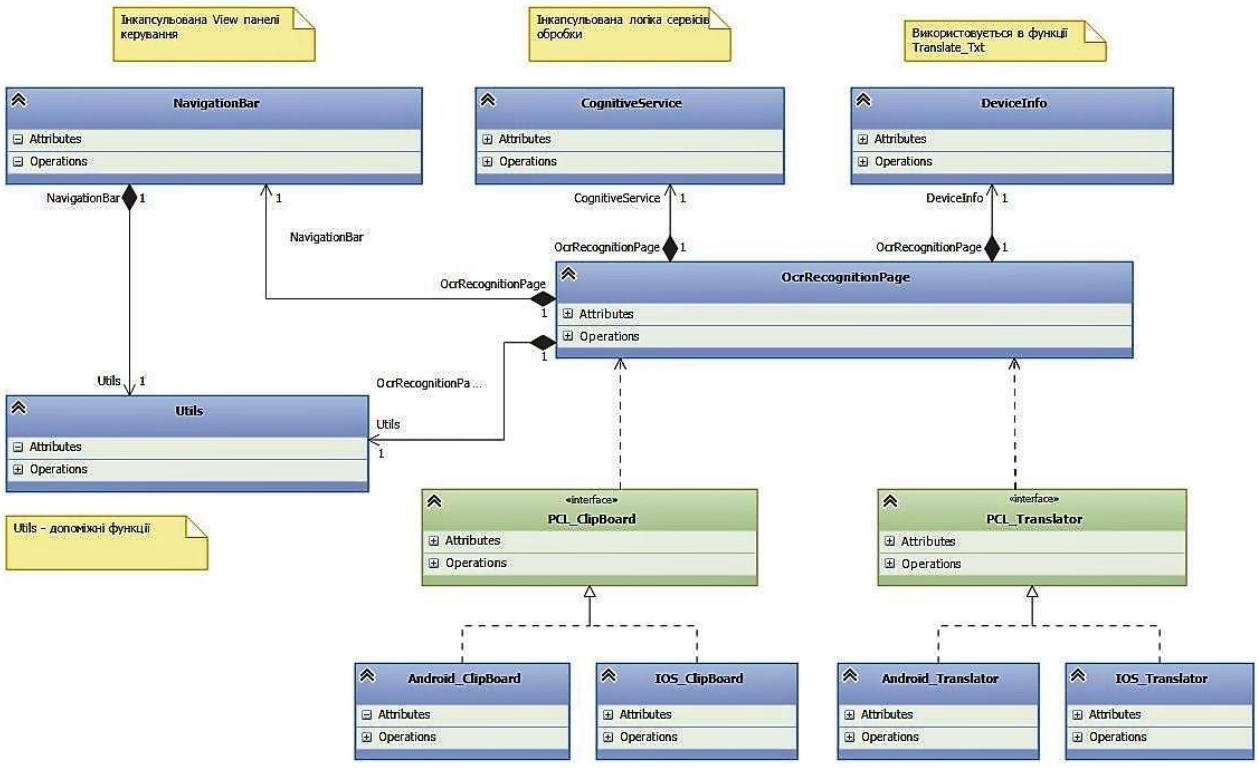

Рис. 3. Загальна схема системи та програмні інтерфейси у вигляді діаграми класів

2. Реалізація обробки текстової інформації.

Отримують дані за допомогою REST запитів до серверу машини навчання Azure з відповідними вхідними параметрами для конкретного запиту.

Результати обробки тексту наведено на рис. 4, де показано запит до машини навчання на переклад зазначеного тексту з англійської мови на українську та результат, отриманий від серверу.

POST https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=uk

```
[
    {"Text":"I would really like to drive your car around the block a few times."}
]
[
    {
        "translations": [
            {
                "text": "Я б дуже хотів, щоб керувати автомобілем по всьому блоку кілька разів.",
                "to": "uk"
            }
        ]
    }
]
```

Рис. 4. Результати запиту та відповідь від сервера машини навчання Azure

Аналогічний алгоритм задіяно для модулів обробки текстової інформації. Всі запити використовують високорівневі інтерфейси, які викликаються мобільним додатком (тобто вимагається активне інтернет-з'єднання).

Алгоритм роботи Microsoft Cognitive OCR:

Так звана процедура лінійного поділу розбиває рядки на слова, а слова – на окремі літери.

Після цього, за принципом IPA (integrity, purposefulness, adaptability) формується набір гіпотез (тобто можливих варіантів того, що це за символ, на які символи розбито слово тощо) і, забезпечивши кожну оцінкою ймовірності, результат передається на вхід механізму розпізнавання символів.

Відповідно до першого правила – принципу цілісності (integrity) – це такий об'єкт, який завжди розглядається як ціле, що складається з багатьох взаємозалежних частин.

Принцип цілеспрямованості (purposefulness): будь-яка інтерпретація даних повинна мати якусь мету. Отже, розпізнавання – це процес висунення гіпотез про весь об'єкт цілком і цілеспрямована їх перевірка. Третій принцип – адаптивності (adaptability) – має на меті здатність системи до самостійного навчання і вміння використовувати раніше накопичені знання про об'єкти. Отримана при розпізнаванні інформація упорядковується, зберігається і використовується згодом для вирішення аналогічних завдань.

Попередня обробка і структурний аналіз зображення

На цьому етапі розв'язуються дві основні задачі: підготовка зображення до процедур розпізнавання і виявлення логічної структури документа з тим, щоб надалі мати можливість відтворити її в електронному вигляді.

Для вирішення першого завдання в Microsoft Cognitive OCR задіяно механізм бінаризації, тобто перетворення кольорового або напівтонового образу на монохромний (глибина кольору 1 біт). Бінаризація істотно прискорює процес аналізу графічних елементів. Виходячи із вищесказаного, розроблена схема бінаризації виглядає так (рис. 5).

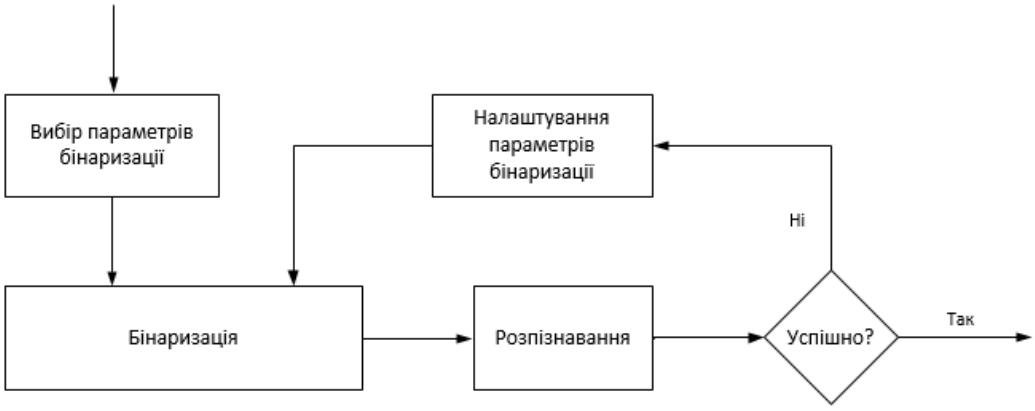

Рис. 5. Структурна схема алгоритму процедури адаптивної бінаризації

Розпізнавання символів

Для розпізнавання символів у Microsoft Cognitive OCR використовуються спеціальні механізми – так звані класифікатори, які породжують список гіпотез, які будуть перевірятися. Вхідними даними для класифікаторів може слугувати не тільки графічна інформація, але і сформований під час розпізнавання список гіпотез. В останньому випадку класифікатор не висуває нових гіпотез, а лише змінює ваги наявних, підтверджуючи або спростовуючи їх. Такий підхід, за яким також чітко простежуються принципи IPA, забезпечує більш інтелектуальний аналіз зображення і найточніше розпізнавання тексту зі зображення. Структурну схему класифікатора наведено на рис. 6.

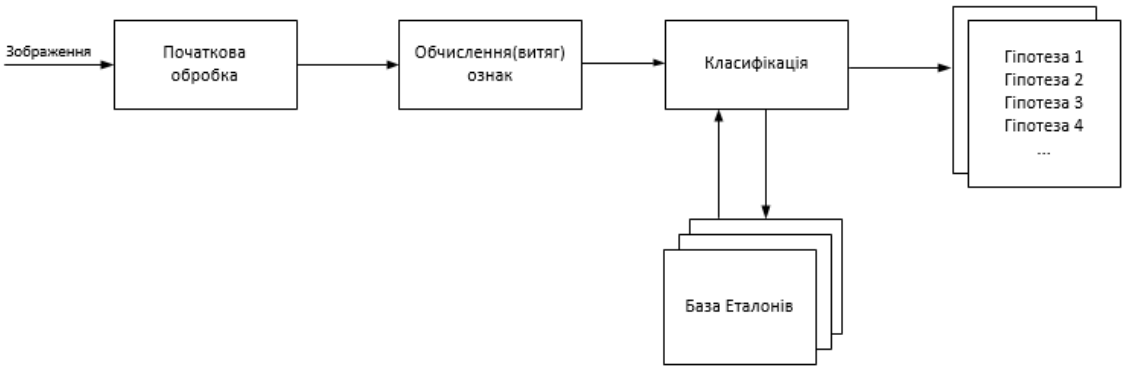

Рис. 6. Структурна схема роботи класифікаторів

Результат роботи програми розпізнавання наведено на рис. 7. У цьому випадку вхідними параметрами є український та англійский тексти, написані різними шрифтами з різними відстанями між буквами. Результатом обробки є український текст, який згодом перекладається англійською мовою. На якість обробки також впливають якість зображення, поворот тексту на зображенні, шрифт, відстань між символами, мова тексту тощо. Причому україномовний та англомовний текст перемежаються.

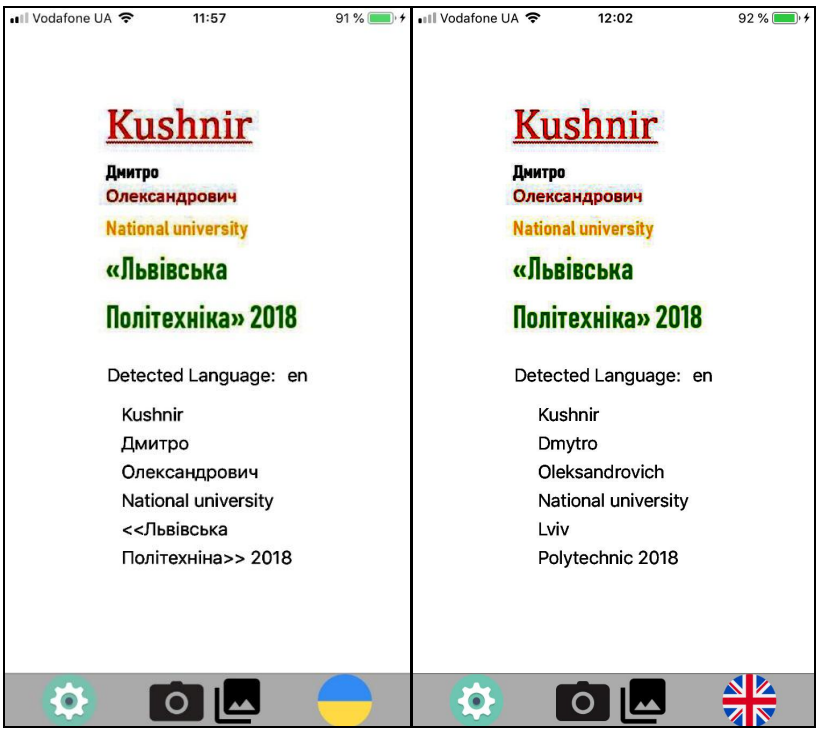

Рис. 7. Результат обробки англо- та україномовних текстів (ліворуч) та переклад обидвох англійською (праворуч) на мобільному пристрої IOS

Як видно з рис. 7, слово “Політехніка” було розпізнане з помилкою («н» замість «к»), проте при виконанні процесу перекладу цю помилку було усунуто.

Результат обробки рукописного тексту наведено на рис. 8. У цьому випадку на складність обробки додатково впливають кут нахилу тексту, почерк (символи можуть бути як довгими, так і вузькими), відстань між символами може коливатися від дуже маленької до дуже великої, якість написаного тексту та характер символу (наприклад, розпізнавати букву ‘j’ достатньо складно, оскільки вона складається з декількох частин).

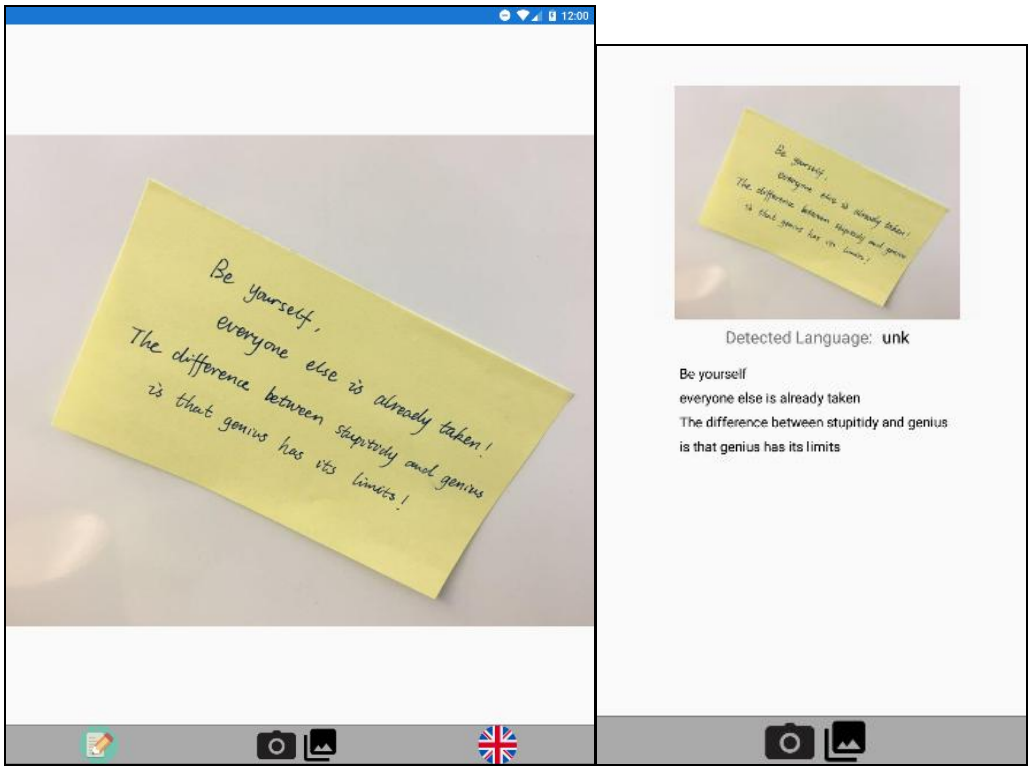

Рис. 8. Обробка рукописного англомовного тексту на мобільному пристрої Android

Використовуючи розроблену методику розпізнавання та перекладу текстової інформації, було досліджено різні шрифти, які було використано для тексту на зображенні. Було також досліджено розпізнавання текстів, написаних різними мовами, та виведено вірогідність правильного розпізнавання слів на кількість символів у кожному з текстів. Результати наведено в таблиці.

Шрифт	Розмір шрифту	Кількість символів	Тип шрифта	Мова тексту/ Колір шрифту	Відсоток коректного розпізнавання
Times New Roman	30	80	Regular	Українська/ червоний	97%
Calibri	12	11	Italic	Англійська/ чорний	97%
Bahnschrift SemiBold	14	54	Bold	Німецька/ зелений	95%
Blackadder ITC	18	21	Regular	Іспаньска/ зелений	81%
Gadugi	10	30	Regular	Українська/ чорний	93%
Matura MT Script Capitals	16	12	Regular	Англійська/ чорний	91%

Висновки

У роботі проаналізовано реалізовані кросплатформні рішення. Показано, що деякі бібліотеки та методи зробити кросплатформними неможливо, що підтверджується реалізацією програмних інтерфейсів.

Проаналізовано роботу модуля обробки інофрмації, а також на основі роботи машини навчання Azure від Microsoft. Проілюстровано переваги та недоліки використання машини навчання та REST запитів порівнянно з аналогічними локальними машинами.

Проаналізовано алгоритм роботи текстової інформації бібліотеки Microsoft Cognitive OCR API. Наведено схему класифікатора та принцип його роботи.

Продемонстровано приклад роботи системи з використанням діаграми класів. Досліджено якість обробки тексту з використаням різних шрифтів, мов, коліру, розміру та кількості тексту.

Отримані результати свідчать про високу якість обробки системи (в середньому приблизно 94% успішного розпізнавання на вхідний набір символів).

1. Anyline. Anyline SDK documentation[Elektronnyj resurs] / New York 2018 – Rezhym dostupu: https://documentation.anyline.com/ 2. Microsoft. Microsoft Cognitive Services documentation [Elektronnyj resurs] / Redmont 2018 Rezhym dostupu: https://azure.microsoft.com/enus/services/cognitive-services 3. Google. Tesseract documentation[Elektronnyj resurs] / Mountain View2018– Rezhym dostupu: https://github.com/tesseract-ocr/tesseract/wiki/Documentation 4. *Olekseev. O What Xamarin developers should know at the beginning of 2017 [Elektronnyj resurs] / Kyiv 2017 – Rezhym dostupu:http://it-ua.info/news/2017/02/03/scho-rozrobniki-xamarin-povinn-znati-na-pochatok2017-roku 5. Appcelerator. Appcelerator Titanium documentation [Elektronnyj resurs] / San Jose 2018 – Rezhym dostupu: https://www.appcelerator.com/Titanium 6. Adobe. PhoneGap documentation [Elektronnyj resurs] / San Jose 2018 – Rezhym dostupu: https://build.phonegap.com/ 7.Microsoft. Xamarin documentation [Elektronnyj resurs] / Redmont 2018 – Rezhym dostupu:https://docs.microsoft.com/en-us/xamarin/ 8. Google. Google translate OCR API documentation[Elektronnyj resurs] / Mountain View 2018 Rezhym dostupu:https://cloud.google.com/functions/docs/tutorials/ocr 9.Microsoft. Machine Learning documentation[Elektronnyj resurs]/Redmont 2018 – Rezhym dostupu:https://docs.microsoft.com/ru-ru/azure/machine-learning/machine-learning-what-is-machine-learning*