УДК 004.4

О.А. ТАТАРІНОВА, В.В. ОВСЯНІКОВ

DOI: 10.20998/2078-9130.2021.2.249715

ПРОГРАМНИЙ ЗАСІБ ДЛЯ РОЗПІЗНАВАННЯ УКРАЇНОМОВНИХ НАУКОВИХ СТАТЕЙ

Розглядається задача комп'ютерного розпізнавання, як окремо друкованих символів, так і цілих текстів, що можуть містити математичні формули, та подальшого збереження результуючого документа у форматі "Латекс". В розробленому програмному забезпеченні реалізовано можливість розпізнавання друкованих символів латиниці, кирилиці, літер грецького алфавіту та спеціальних математичних знаків. Для цього застосовуються багатошарова згортальна нейронна мережа, побудована за допомогою бібліотеки машинного навчання "Керас", та додаткові валідаційні евристики. Для підвищення якості розпізнавання нейронної мережі розроблено складний механізм преобробки зображень, що допомагає видалити шуми із зображення, виключити похибки пов'язані з нахилом символів, коректувати дефекти символів, пов'язані з якістю вхідного зображення. Також реалізовано механізми збирання окремих символів в слова або ж математичні формули, відтворення положення знаків індексів та ступенів, формування звичайних дробів та виразів під знаком кореня. Відбувається збереження результатів розпізнаного тексту до файлу з одночасною побудовою структури «latex» документу. Для демонстрації можливостей розробленого програмного забезпечення додано графічний інтерфейс користувача, за допомогою якого можна ще до початку розпізнавання обрати та оглянути вхідне зображення. Під час тестування програмного засобу, було проведено розпізнавання зображень різних типів: повністю текстуальні, математичні формули без тексту, математичні формули, які знаходяться між блоками тексту.
Ключові слова: оптичне розпізнавання символів, згортальні нейронні мережі, комп'ютерний зір.

Рассматривается задача компьютерного распознавания, как отдельно печатных символов, так и целых текстов, которые могут содержать математические формулы, и дальнейшего сохранения результирующего документа в формате "Латекс". В разработанном программном обеспечении реализована возможность распознавания печатных символов латиницы, кириллицы, букв греческого алфавита и специальных математических знаков. Для этого применяются многослойная свертывающая нейронная сеть, построенная с помощью библиотеки машинного обучения Керас, и дополнительные валидационные эвристики. Для повышения качества распознавания нейронной сети разработан сложный механизм переработки изображений, помогающий удалить шумы из изображения, исключить погрешности, связанные с наклоном символов, корректировать дефекты символов, связанные с качеством входного изображения. Также реализованы механизмы сбора отдельных символов в слова или математические формулы, воспроизведение положения знаков индексов и степеней, формирование обычных дробей и выражений под знаком корня. Происходит сохранение результатов распознанного текста в файле с одновременным построением структуры «latex» документа. Для демонстрации возможностей разработанного программного обеспечения добавлен графический пользовательский интерфейс, с помощью которого можно еще до начала распознавания выбрать и осмотреть входное изображение. Во время тестирования программного средства было проведено распознавание изображений разных типов: полностью текстуальные, математические формулы без текста, математические формулы, которые находятся между блоками текста.
Ключевые слова: оптическое распознавание символов, сверточные нейронные сети, компьютерное зрение.

The problem of computer recognition, both separately printed characters and whole texts, which may contain mathematical formulas, and further saving the resulting document in the "Latex" format, is considered. The developed software implements the ability to recognize printable Latin, Cyrillic, Greek letters and special mathematical symbols. For this, a multilayer convolutional neural network built using the Keras machine learning library and additional validation heuristics are used. To improve the quality of neural network recognition, a sophisticated image processing mechanism has been developed that helps to remove noise from the image, eliminate errors associated with the inclination of characters, and correct character defects associated with the quality of the input image. Also implemented are mechanisms for collecting individual characters into words or mathematical formulas, reproducing the position of signs of indices and degrees, forming ordinary fractions and expressions under the root sign. The results of the recognized text are saved in a file with the simultaneous construction of the "latex" document structure. To demonstrate the capabilities of the developed software, a graphical user interface has been added, with which you can select and inspect the input image even before the start of recognition. During testing of the software, the recognition of images of different types was carried out: completely textual, mathematical formulas without text, mathematical formulas that are between blocks of text.

Keywords: .optical character recognition, convolutional neural network, computer vision.

Вступ. Внаслідок низького темпу технологічного розвитку України наприкінці минулого та на початку теперішнього сторіччя, велика кількість україномовних наукових робіт, монограм, дисертацій, рефератів та у загальному сенсі наукових текстів, існують лише як фізичні примірники, не маючи цифрової копії. Це робить майже неможливим вільне поширення та обмін багатьма вітчизняними науковими здобутками, що закладені в такі роботи, бо кількість надрукованих екземплярів, навіть найвідоміших вітчизняних наукових текстів не сягає більше 500000 Частина оцифрованих примірників розповсюджується у розширенні «pdf», або ж у формі набору графічних зображень, що робить неможливим їх задовільну індексацію пошуковими системами. З цього випливає, що наявність додатка, який би переформатовував скановані копії наукових статей у традиційний формат для текстів такого роду, примірників. Таке охоплення виявляється мізерним у відсотковому співвідношенні із кількістю людей на планеті, що наразі перевищує сім мільярдів, які б могли бути ознайомлені з українськими науковими працями. Ця проблема найбільше заважає швидшій інтеграції вітчизняної науки у європейський та світовий науковий простір, ще з часів здобуття незалежності, але є актуальною і дотепер.

Наявність цифрової копії у всесвітній мережі, у рази поліпшує ситуацію з доступністю даного документа для широких мас людей. Та нажаль, переважна
посприяла б скорішому розповсюдженню наукових здобутків українських вчених.

Алгоритм роботи. Надамо загальний алгоритм, що використовувався у роботі для реалізації розпізнавання україномовних наукових текстів.

1 Змінюється кольоровий формат вхідного зображення з «RGB-255» до формату «відтінки сірого».

Далі, воно замилюється з використанням фільтру Гауса, щоб покращити якість виділення рядків, а потім бінаризується, щоб видалити із зображення непотрібний шум.

2 Відбувається пошук білих ліній, які за розміром дорівнюють ширині зображення. Лінії, менші за розміром, помічаються як помилки процесу трейсхолдінгу. Потім рядки, розділені такими лініями, копіюються в окреме зображення. Рядки класифікуються за типом – "текстовий" чи "математичний".

3 Формулою, чиї коефіцієнти адаптуються відповідно до характеристик зображення, реалізується класифікація проміжків на міжбуквені та міжслівні. Таким чином в усіх рядках знаходяться області, що відповідають за репрезентацію слів та формул.

4 Виділяються контури знаків. Підвищується чіткість зображення для покращення якості розпізнавання.

5 Відбувається зведення до єдиного розміру і подальше розпізнавання згортальною нейронною мережею тих частин вхідного зображення, які містять контури символів.

6 Відбувається відтворення граматичної структури речень та формул, за координатами символів на зображені. Перетворення сукупностей зв'язних символів на специфічні конструкції мови latex для їх коректного відтворення.

7 Відбувається збереження результатів розпізнаного тексту до файлу з одночасною побудовою структури «latex» документу.

Окремим кроком перед підготовкою алгоритму обробки зображення є навчання нейронної мережі. Чисельні рекомендації пропонують обирати для роботи із зображеннями згортальні нейронні мережі [1-6]. Багатошарові згортальні нейронні мережі прямого розповсюдження, в переважній більшості, застосовуються для "навчання з вчителем" [7-8]. Виключенням є випадки, коли вони застосовуються в якості декодера. Як наслідок, ще до етапу проектування структури мережі, потрібно підготувати навчальну вибірку. Вибірка повинна містити ASCII коди символів, як мітки класів, та зображення символів у якості вектору ознак. Наразі, у вільному доступі не існує навчальної вибірки, прийнятної для тренування мережі ідентифікації знаків, що застосовують для формування latex документів. Зважаючи на це, було прийнято рішення сформувати її самостійно.

Було зібрано скановані зображення 50-ти latex документів, збережених у розширені png. Усі зображення збережені у високій якості, форматі 1100x2100 пікселів. З них було сформовано вибірку трохи меншу за 120000 зображень знаків.

Усього було отримано 329 унікальних класів символів. До навчальної вибірки було відібрано по 300 зображень для кожного класу, по тридцять для валідаційної і сто для тестової.

Базовою архітектурою методу розпізнавання було обрано згортальну нейронну мережу. Вона включала пару послідовних комбінацій згортальних шарів (по шістдесят чотири фільтри) і розріджуючих шарів, та три повно зв'язних шари у кінці (рис. 1).

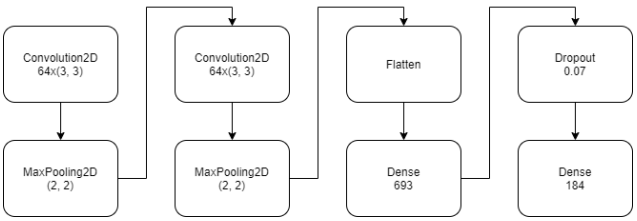


Рисунок 1 – Архітектура нейронної мережі

Розроблений програмний засіб не передбачає будь-якої обробки кольорових зображень у тексті. Що ж стосується літер та інших символів, їх забарвлення мало впливає на відмінність одного від іншого. Тож щоб підвищити ймовірність безпомилкового розпізнавання, вхідне фото позбавляється третього рівня вкладеності, що трансформується у насичення відтінків сірого формулою

Формат відтінків сірого не передає надлишкової, у випадку класифікації символів, інформації про забарвлення, а лише вказує на переважну насиченість пікселів світлими чи темними відтінками. Це дозволяє значно знизити розмірність вектору ознак без відчутної втрати основних відмінних рис об'єктів, що потрібно класифікувати.

Не можна не брати до уваги той факт, що будь-яке вхідне зображення, в залежності від розширення та якості, матиме відмінні за величиною проміжки між рядками, словами та знаками. Також, висока "зашумленість" фото взагалі може призвести до відсутності таких проміжків. Задля зменшення впливу цих факторів на роботу алгоритму виділення рядків, світлина замилюється однією з модифікацій фільтру Гауса [9]:

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}},$$

$$G(x,y) = \sum_{x_1=x-n/2}^{x+n/2} \sum_{y_1=y-m/2}^{y+m/2} \frac{1}{2\pi\sigma^2} e^{-\frac{x_1^2+y_1^2}{2\sigma^2}} D(x_1, y_1),$$

де σ – середньоквадратичне відхилення; x – позиція пікселя у вікні по осі X; y – позиція пікселя у вікні по осі Y; n –ширина вікна фільтра; m – довжина вікна фільтра; $D(x_1, y_1)$ – значення насиченості пікселя з координатами (x_1, y_1);$G(x, y)$ – отримане значення насиченості пікселя з координатами (x,y) після застосування гаусівського розмиття.

Очікуваним результатом застосування розмиття фільтром Гауса є відносне затемнення усіх пікселів, що мають більше темних пікселів по-сусідству. Тож, лінії, у яких переважають "темні" пікселі заповнюють небажані проміжки, що допомагає уникнути помилково-позитивного детектування рядка.

Алгоритм виділення рядків та їх класифікації відбувається таким чином. Спершу, пошук усіх ліній, що містять лише білі пікселі. Далі, обчислюється середня товщина таких ліній. Лінії, чия товщина не досягає 20% середньої товщини, розглядаються у якості відступів

між частинами окремих символів та не обробляються при наступних етапах.

Наступним кроком є проведення типізації виділених рядків на “алгебраїчні” та “літературні”. Така типізація є вкрай необхідною через виразну відмінність правил формування структури алгебраїчних виразів та структури рядків літературного тексту. Під час дослідження різноманітних наукових текстів було виявлено, що переважна більшість з них має чітко визначений формат. Також, був зроблений висновок, що більшість рядків, за виключенням заголовків і крайніх рядків абзаців, за координатами границь на зображені, відрізняються максимум на два відсотка. Рядки, що за шириною потрапляють у такий проміжок, після перевірки на наявність спеціальних символів та висоти, можна відразу віднести до типу “літературних”. До літературних слід, також, віднести рядки з абзацами. Їх легко ідентифікувати за лівим відступом розміром в 10% від довжини повного рядка. Абзацні рядки, права крайня координата яких може набувати будь-якого значення, із проміжками між своїми сусідами більшими за середні позначаються “підзаголовками”. Рядки, текст яких рівновіддалений від правого та лівого краю зображення, чиї символи істотно вищі за медіану висоту символів інших літературних рядків позначається як “заголовок”.

Не можна, також виключати ймовірність того, що алгебраїчні формули можуть бути рівними за шириною з “літературними” рядками, центровані або зміщені у якийсь бік, набувати різної висоти. Остаточно виявити тип рядка можна лише спираючись на значення та положення символів, з яких він складається. Але ж, під час знаходження рядків алгоритм не має інформації про значення символи. Через це для поділу зображень рядків за типом на алгебраїчні та літературні використовувалося співвідношення наповненості чорним і повної площі рядка.

Ще більш наглядною рисою, що забезпечує краще розподілення рядків на відповідні класи, є наповненість чорним кольором у межах “опорних ліній рядка”. Даний підхід є ліпшою альтернативою стандартному визначенню границь рядка за його крайніми точками. Для визначення опорних ліній у зображенні підраховуються суми насиченості для кожного рядка. Після знаходяться індекси рядків зображення, що відповідають локальним максимумам насиченості у верхній та нижній частинах. Різниця у відсотку наповненості пікселями чорного кольору для літературних рядків в межах опорних ліній значно зростає на відміну від алгебраїчних.

Для використання цієї ознаки як головної для класифікації рядків потрібно підрахувати середнє значення наповненості зображення у середині опорних ліній. Сама ж типізація проходить за таким правилом — наповненість алгебраїчних рядків нижча за 20% або є нижчою за 50% від попередньо підрахованої середньої. За цим правилом розподілення рядків на типи відбувається майже з 90% точністю. Для валідації помилок перевіряються значення розпізнаних символів. Якщо спеціальні алгебраїчні символи складають хоч дванадцяту частину від усіх символів, рядку надається тип “алгебраїчний”.

Одиничні значущі елементи — це найменші частини тексту, що мають зміст. Такими елементами у складі літературного рядка є слова. Для алгебраїчного ж рядка таку роль відіграють формули. Надалі, аби спростити опис алгоритму, одиничні значущі елементи будуть іменуватися смисловими одиницями.

Смислові одиниці виділяються для рядків різного типу із застосуванням відмінних правил через кардинальну відмінність у структурі слів та алгебраїчних виразів. Початковий крок для обох ідентичний. Щоб усунути із зображення більшість міжсимвольних проміжків та залишити переважно проміжки між словами, застосовується процедура замилювання фільтром Гауса.

У межах літературного рядка пошук смислових одиниць проходить двома етапами. Спершу, вираховується товщина усіх наявних на зображені проміжків – як міжслівних, так і міжсимвольних. Для врівноваження середнього значення товщини відбувається додавання максимальних чи мінімальних значень, щоб прирівняти їх кількість. Потім обчислюється середнє, враховуючи компенсовані значення. З отриманого першим обрахунком ряду значень товщини вилучаються елементи менші за середнє, що враховує компенсовані значення. Після фільтрації підраховується нове середнє значення. Рядок поділяється на слова за проміжками, що є на 30% більшими за нове середнє значення.

Для алгебраїчного рядка смисловими одиницями є формули. Дослідження багатьох екземплярів latex документів показало, що більшість алгебраїчних виразів записуються в рядок поодинці. Але, цей формат розмітки, все ж дозволяє розташування кількох формул в рядку

На відміну від літературних рядків, розділяти алгебраїчні на окремі вирази, базуючись на середньому значенні товщини, не можна. Для такого поділу спочатку із зображення видаляються пусті відступи зліва та справа. Наступним кроком із повного зображення рядку виділяються окремі зображення виразів. Тригером для поділу слугує проміжок між символами, який більше 10% ширини рядка

Як вже описувалося раніше, зображення у форматі відтінків сірого є двовимірною матрицею яскравості пікселів. За такої умови окремим символом буде вважатися сукупність сусідніх пікселів зі значно нижчою від інших сусідів яскравістю. Щоб мати змогу визначити координати та розмір символу, необхідно мати інформацію лише про крайні точки об’єкту. Тож, можна не проводити обробку “внутрішніх” пікселів кожного символу, звертаючи увагу лише на зовнішній контур. У якості методу виділення контурів із зображення застосовувався “ланцюговий код Фрімена”. Він ініціює рекурсивний перебір чорних пікселів, перевіряючи сусідні пікселі на рівність за яскравістю, виділяючи рівно-яскраві в один символ. В результаті,

контур кожного символу зберігається у формі положення вихідного пікселя та послідовності кроків за 8-зв'язною решіткою напрямків, потрібних щоб охопити всі зовнішні пікселі

Для підвищення швидкості роботи методу вхідне зображення слід бінаризувати. Точне значення яскравості не має впливу під час збирання контуру, необхідно лише визначитись з його приналежністю до групи переважно темних чи переважно світлих. Порогові значення для розподілу на такі групи є відмінними для кожного окремого зображення. Так, яскравість переважно темного пікселя може знаходитися в межах від 0 до ста 50 для одного зображення та обмежуватися 30 для іншого. Це унеможливлює завдання універсального статичного порогу бінаризації. Зважаючи на це бінаризація проводилася методом Оцу, який адаптивно обчислює поріг бінаризації для кожного зображення. Поріг має розділяти гістограму на два класи так, щоб мінімізувати внутрікласову дисперсію. Вона дорівнює зваженій сумі дисперсій двох класів [10]:

$$\sigma_w^2 = p_1\sigma_1^2 + p_2\sigma_2^2,$$

де p_1 та p_2 – ймовірності класів; σ_1 та σ_2 – середньоквадратичне відхилення.

Мінімізацію внутрікласової дисперсії можна записати у вигляді максимізації міжкласової дисперсії [10]:

$$\sigma_b^2 = p_1 p_2(\mu_1 - \mu_2)^2,$$

де p_1 та p_2 – ймовірності класів; μ_1 та μ_2 – середні арифметичні значення класів.

Тож, використовуючи будь-який метод оптимізації, перераховуючи дисперсію $\sigma_b^2(t)$ на мінімальній кількості кроків, визначаємо поріг, який би мінімізував внутрікласову дисперсію.

Етапом, який слідує за виділенням контурів, є розпізнавання згортальною нейронною мережею окремих символів. Аби підготувати вхідні вектори ознак найліпшим чином, з копії вхідного не обробленого зображення виокремлюються області, що містять контури. Через те, що вектори ознак кожного екземпляру слід звести до однієї розмірності, виокремленні зображення трансформуються до формату 64x64 пікселі, а потім у форму вектору. Також, щоб завадити порушенню балансу впливу на розпізнавання різних пікселів, відбувається процедура шкалювання. Вона передбачає ділення значень яскравості пікселів на максимально можливу, яка дорівнює 255. Це робиться для того, щоб всі значення яскравості перебували в межах від 0 до 1.

Результатом роботи нейронної мережі є мітка класу, яка дорівнює числу, що є кодом символу в таблиці ASCII. Підхід окремого розпізнавання кожного знаку має недолік — алгоритм не передбачає існування символів, котрі є комбінацією кількох знаків. Частини складних символів, як "...","i", "!", "?", "j", ":", та ";" класифікуються окремо, набуваючи значень найбільш схожих знаків та окремої крапки. На допомогу машинному розпізнаванню був введений етап "збирання" таких символів. Для того, щоб помилково не детектувати шум на зображенні як складову символу, відбувається пошук можливого батьківського символу для кандидату. Батьківський символ визначається за правилом: координати центрів двох знаків по осі X не зміщені відносного одне одного на більш ніж 5% від загальної ширини зображення смислової одиниці, та 10% від висоти по осі Y.

Для вирішення ситуацій, коли літери втрачають відмінності між своїми великими та малими формами, після трансформації вхідних зображень символів до єдиного розміру існує окрема емпірична обробка. Поділ на великі і малі літери відбувається, спираючись на обчисленні під час минулих кроків координат "опорних ліній рядка". Однозначно можна зазначити, що найвищі точки великих літер розташовані істотно вище верхньої з двох ліній, а найвищі точки малих — знаходяться в її околі.

Складність також виникає при класифікації тире та дефісу, бо після зведення до єдиного розміру вони втрачають характерну відмінність в довжині. Але можна з точністю зауважити, що тире має бути єдиним символом у своїй смисловій одиниці, бо оточений міжслівними проміжками. Натомість дефіс у будь-якому разі матиме сусідні символи.

Додатковим етапом до алгоритму включено перевірку розпізнавання символів "апостроф", "лапки" та "кома". Оскільки за виглядом кома, апостроф та окремі частини лапок ідентичні, вірна їх класифікація нейронною мережею неможлива. Різниця між цими символами забезпечена лише їх координатами та сусідніми символами. Кома має бути розташована у нижній половині зображення, апостроф та лапки у верхній. Два апострофа підряд помічаються, як лапки.

Збирання ж алгебраїчних виразів викликає ще більше труднощів. Одна алгебраїчна одиниця може бути виражена кількома знаками. Також труднощів додає варіативне положення символів у рядку. Різноманітні індекси, ступені, дробові вирази правильність відображення, яких критично залежить від положення символів. Та першим чином, вже описане правило застосовується для збирання частин символів "j" і "i" та для їх запису як єдиний знак. Схоже правило застосовується для об'єднання двох послідовних знаків дефіс у символ "=".

Також треба зважати на те, що змінні в алгебраїчних виразах записуються курсивним шрифтом. Мережа розпізнає ці символи як курсивні. Але компілятор latex автоматично передбачає написання змінних курсивним шрифтом. Тому, курсивна форма для символу змінної замінюється на звичайну.

Унікальної обробки заслуговують символи, розпізнані, як "∑" чи "П", так як, крім позначення грецьких літер, ними також позначають вирази суми та множення. Для обробки таких випадків відбувається пошук другорядних для виразу символів-кандидатів. Центри кандидатів мають координати, що по осі X розміщені у межах країв головного символу. По осі Y

максимально можуть бути віддалені від країв “∑” чи “П” на 20% від висоти зображення рядка. Потім відповідно до координат підібраних символів-кандидатів, формуються конструкції, що позначають вирази множення та суми. Аналогічно формується і конструкція, яка позначає визначений інтеграл.

Не можна обійти увагою і вирази для позначення косинусів, синусів, тангенсів, котангенсів та лімітів. При кожному розпізнаванні символу, що позначає першу літеру вищевказаних операторів, наприклад, символ “l” для оператору “lim”, ініціюється підбір символів-кандидатів. Щоб сукупність кандидатів була прийнята за оператор, усі символи мають набувати відповідних значень у вірній послідовності. Додатковою умовою є відповідність кандидатів вимогам до розташування на обох осях.

Як відомо, символ рівності для будь-якого рівняння є знаком, від розташування якого залежить розташування усіх інших його складових. Саме цей знак задає “головну лінію”. Якщо вираз має лише один символ рівності, то “головна лінія” буде центром знаку “=” по осі Y. Якщо ж вираз міститиме декілька таких знаків, буде обраний центр того, що буде розташований ближче до центру зображення. За повної відсутності такого знаку, “головну лінію” виразу позначають знаки додавання, віднімання, множення тощо. В іншому разі буде обрано центр знаку найближчого до вертикального центру зображення. Спираючись на значення “головної лінії”, ті символи, нижні точки яких знаходяться вище за “головну лінію”, записуються, як степені. Індексом змінної позначаються ті, чия найвища точка розташована в околі “головної лінії”, а найнижча не менш ніж на 10% нижче координати низу попередньої змінної.

Останнім кроком алгоритму преобробки є формування виразів, якими записуються звичайні дроби. Кожен символ класифікований як тире або дефіс проходить перевірку на наявність підрядних символів, з яких складаються чисельник і знаменник. Такі символи набувають значення знаку мінус за відсутності кандидатів. За умови наявності кандидатів зверху і знизу символу починається формування конструкції звичайного дробу. У такому разі, чисельник і знаменник розглядаються, як окремі рядки. Рекурсивно до них застосовуються вищеописані методи обробки.

Розпізнаний результат необхідно представити latex файлом. Для коректного запису усі рядки сортуються за висотою положення на зображені. Смислові одиниці сортуються у рядку по лівій крайній координаті. Формат розмітки документу latex вимагає для забезпечення правильної обробки обертання в спеціальні блоки усіх підзаголовків, заголовків та алгебраїчних виразів, а також включення у початок та кінець документу специфічних блоків та макросів. Таким чином, текст документу знову обробляється з метою спрощення окремих виразів. Наприклад, результатом роботи алгоритму буде вираз “^n^+^x”, який означає змінну, що мала б вираз з сумою інших змінних у своєму ступені. Такий запис під час рендерінгу буде оброблено правильно. Та все ж він містить семантичну помилку. Постобробка спростить вираз до форми “^{n+1}”, що є семантично правильною.

Опис програмного засобу. На основі розглянутого алгоритму було реалізовано програмний засіб, діаграма варіантів використання якого наведена на рисунку 2 [11].

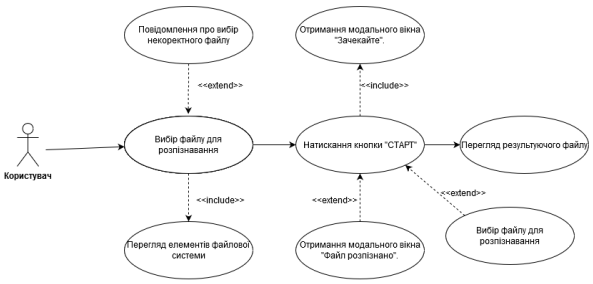


Рисунок 2 – Діаграма варіантів використання

Програмна реалізація описана у формі діаграми класів, яка приведена на рисунку 3.

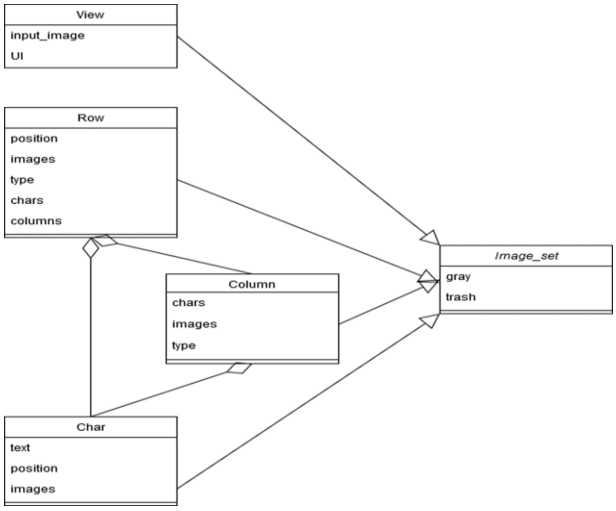
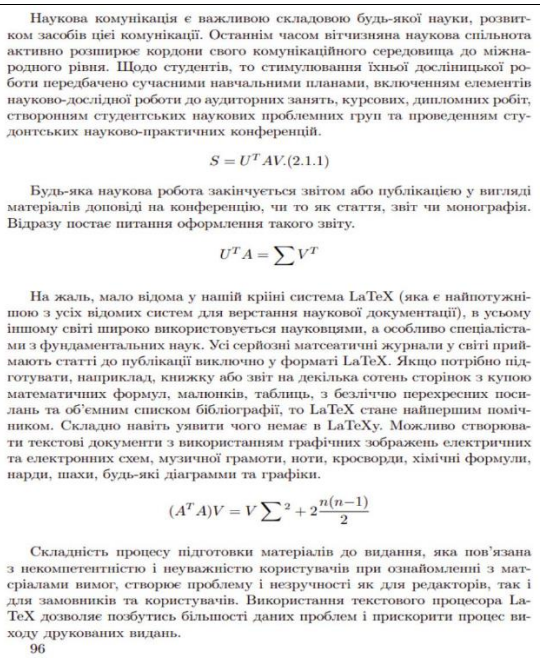


Рисунок 3 – UML-діаграма класів

Як того вимагають правила розробки програмного забезпечення, до основного коду програми також включені unit тести. Тестування проводилося засобами бібліотеки тестування python-коду pytest [12]. Також було проведене мануальне тестування з використанням придатних для цього зображень. Зміст вхідних зображень було порівняно з результатом рендерінгу вихідних файлів. Приклад розпізнаної статті наведено на рис. 4.

Під час тестування програмного засобу, було проведено розпізнавання зображень різних типів: повністю текстуальні, математичні формули без тексту, математичні формули, які знаходяться між блоками тексту. Визначено, що програмний засіб розпізнає наукові тексти з точністю вище 80%.



Наукова комунікація є важливою складовою будь-якої науки, розвитком засобів цієї комунікації. Останнім часом вітчизняна наукова спільнота активно розширює кордони свого комунікаційного середовища до міжнародного рівня. Щодо студентів, то стимулювання їхньої дослідницької роботи передбачено сучасними навчальними планами, включенням елементів науково-дослідної роботи до аудиторних занять, курсових, дипломних робіт, створенням студентських наукових проблемних груп та проведенням студентських науково-практичних конференцій.

$$S = U^T AV. (2.1.1)$$

Будь-яка наукова робота закінчується звітом або публікацією у вигляді матеріалів доповіді на конференцію, чи то як стаття, звіт чи монографія. Відразу постає питання оформлення такого звіту.

$$U^T A = \sum V^T$$

На жаль, мало відома у нашій країні система LaTeX (яка є найпотужнішою з усіх відомих систем для верстання наукової документації), в усьому іншому світі широко використовується науковцями, а особливо спеціалістами з фундаментальних наук. Усі серйозні матсеатичні журнали у світі приймають статті до публікації виключно у форматі LaTeX. Якщо потрібно підготувати, наприклад, книжку або звіт на декілька сотень сторінок з купою математичних формул, малюнків, таблиць, з безліччю перехресних посилань та об'ємним списком бібліографії, то LaTeX стане найпершим помічником. Складно навіть уявити чого немає в LaTeXy. Можливо створювати текстові документи з використанням графічних зображень електричних та електронних схем, музичної грамоти, ноти, кросворди, хімічні формули, нарди, шахи, будь-які діаграмми та графіки.

$$(A^T A)V = V\sum{}^2 + 2\frac{n(n-1)}{2}$$

Складність процесу підготовки матеріалів до видання, яка пов'язана з некомпетентністю і неуважністю користувачів при ознайомленні з матеріалами вимог, створює проблему і незручності як для редакторів, так і для замовників та користувачів. Використання текстового процесора LaTeX дозволяє позбутись більшості даних проблем і прискорити процес виходу друкованих видань.

96

Рисунок 4 – Приклад розпізнаної статті

Висновок. У процесі розробки було реалізовано програмний засіб для розпізнавання окремих знаків, що включає згортальну нейронну мережу та додаткові перевірки розпізнаного документу мережею; алгоритм відтворення структури тексту та алгебраїчних формул; механізм збереження результатів роботи алгоритмів із застосуванням правил мови розмітки latex.

Принципова новизна роботи полягає у створенні комплексного алгоритму для розпізнавання україномовних наукових статей. Рішення-аналоги не надають можливість розпізнавання наукових текстів із подальшим зберіганням у форматі latex. Також не існує програм, які б розпізнавали саме україномовні наукові статті.

Для спрощення користування додатком за допомогою мови програмування Python та пакету kivy імплементовано графічний інтерфейс користувача. За його допомогою значно легше проводити вибір файлу для розпізнавання. Також, можна переглянути контент файлу до початку обробки. Рендерінг результуючого .tex документу забезпечується будь-яким latex-компілятором.

Список літератури

1. *Травин А.* Технологии оптического распознавания текста [Электронный ресурс] / *А. Травин*. – Режим доступа: http://travin.msk.ru/arc/OCR.html.
2. *Котович Н. В.* Распознавание скелетных образов [Электронный ресурс] / Н. В. Котович, О. А. Славин. – Режим доступа: http://ocrai.narod.ru/skeletrecognize.html.
3. *Миловская Ю.В.* Обработка сканированного текста/ *Ю.В. Миловская* // Сборник научных трудов НГТУ. – 2018. – № 3–4 (93). – С. 91–100.
4. *Parker J. R.* Algorithms for image processing and computer vision (second edition)./ J. R. Parker– Wiley, 2010.
5. *Forsyth D. A.* Computer Vision: A Modern Approach [Text] / *D. A. Forsyth, J. Ponce*. – Pearson, 2011. – 792 p.
6. *Joachims T.* Learning to classify text using support vector machines: Methods, theory, and algorithms. / *T. Joachims* – Kluwer, 2002.
7. *Хайкин С.* Нейронные сети. Полный курс [Текст] / *С. Хайкин*. – М. : Вильямс, 2006. – 1104 с.
8. *Melin P.*,Voice Recognition with Neural Networks/ *P. Melin, J. Urias, D. Solano, M. Soto, M. Lopez, O. Castillo* // Type-2 Fuzzy Logic and Genetic Algorithms. Engineering Letters. – 2006. – Vol. 13:2.
9. *Szeliski R.* Computer Vision: Algorithms and Applications (Texts in Computer Science) / *R. Szeliski* – 2021
10. *Otsu N.* A threshold selection method from gray-level histograms, / *N. Otsu* // I Systems, Man and Cybernetics EEETransactions. – 1979. – Vol. 9, № 1.
11. *Бреславський Д.В.* Проектування та розробка скінченноелементного програмного забезпечення. / *Д.В. Бреславський, Ю.М. Коритко, О.А Татарінова* - Харків: «Підручник НТУ «ХПІ», 2017. - 232 с.
12. PyInline: Mix Other Languages directly Inline with your Python. — Режим доступу: http://pyinline.sourceforge.net/

References (transliterated)

1. Travyn A. *Tekhnolohyy optycheskoho raspoznavanyya teksta* [Optical Character Recognition Technology]. Available at: http://travin.msk.ru/arc/OCR.html
2. Kotovych, N. V., Slavin, O. A. *Raspoznavanye skeletnыkh obrazov* [Recognition of skeletal images]. Available at: http://ocrai.narod.ru/skeletrecognize.html
3. Milovskaya Yu.V. *Obrabotka skanirovannogo teksta* [Processing scanned text]. Sbornik nauchnykh trudov Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Transaction of scientific papers of the Novosibirsk state technical university, 2018, no. 3–4 (93), pp. 91–100.
4. *Parker J. R.* Algorithms for image processing and computer vision (second edition).Wiley, 2010.
5. *Forsyth D. A., Ponce J.* Computer Vision: A Modern Approach [Text].Pearson, 2011. 792 p.
6. *Joachims T.* Learning to classify text using support vector machines: Methods, theory, and algorithms. Kluwer, 2002.
7. *Khaykyn S.* Neyronnie sety. Polniy kurs [Neural networks. Full course]. Moskow, Vyl'yams Publ., 2006. 1104 p.
8. Melin P., Urias J., Solano D., Soto M., Lopez M., Castillo O. *Voice Recognition with Neural Networks*. Type-2 Fuzzy Logic and Genetic Algorithms. Engineering Letters, 2006. 13:2
9. *Szeliski R.* Computer Vision: Algorithms and Applications (Texts in Computer Science) / *R. Szeliski* – 2021
10. Otsu N. *A threshold selection method from gray-level histograms.* I Systems, Man and Cybernetics EEETransactions. 1979. Vol. 9, № 1.
11. *Breslavskyi D.V., Korytko Yu.M., Tatarinova O.A.* Proektuvannia ta rozrobka skinchennoelementnoho prohramnoho zabezpechennia. [Design and development of finite element software]. Kharkiv «Pidruchnyk NTU «KhPI» Publ., 2017, 232 p.
12. PyInline: Mix Other Languages directly Inline with your Python. Available at: http://pyinline.sourceforge.net/