https://doi.org/10.30857/2786-5371.2024.5.3

УДК 004.384

АСТІСТОВА Т. І., СЕДЛЯР А. О.
Київський національний університет технологій та дизайну, Україна

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНКИ ОРИГІНАЛЬНОСТІ ТЕКСТІВ

Мета. *Розробити програмне забезпечення, що поєднує стандартні методи та технології штучного інтелекту для оцінки оригінальності тексту при написанні наукових робіт.*

Методика. *В основу розробки програмного забезпечення було покладено теоретичні та експериментальні дослідження, які використовуються для аналізу методів розпізнавання текстів генерованих штучним інтелектом, виборі та аналізі нейронної мережі для виявлення патернів та відповідних закономірностей при написанні тексту людиною та штучним інтелектом та інтеграції цієї моделі в програмне забезпечення. для перевірки тексту на оригінальність, реалізація технологій аналізу тексту.*

Результати. *Основні результати роботи можна розділити на три взаємопов'язані аспекти: аналітичний, проектувальний і практичний, кожен з яких зробив вагомий внесок у досягнення поставленої мети. В результаті дослідження було здійснено ґрунтовний аналіз предметної області, розглянуто поняття штучного інтелекту, його архітектуру, функціонування нейронних мереж та їх застосування для аналізу текстів, технічні характеристики і можливості сучасних моделей ШІ, а також аспектам їхньої інтеграції у процес перевірки текстів, висвітлено питання академічної доброчесності, включаючи визначення плагіату, його види та культурні особливості сприйняття цього явища у процесах навчання й наукової діяльності, створено теоретичний фундамент для розроблення інструмента, який здатен враховувати як технічні, так і етичні аспекти.*

Докладно розглянуто структуру програми, її алгоритми, методи інтеграції з API для аналізу текстів та формату зберігання даних, розроблено алгоритм подібності, який став основою для виявлення схожості текстів. Це дозволило створити чітку та логічну архітектуру продукту, яка забезпечує його функціональність та ефективність.

Наукова новизна*. Запропоноване оригінальне рішення у вигляді інтегрованої системи яка має поєднувати класичні методи перевірки тексту на оригінальність з розпізнаванням текстів, створених штучним інтелектом, що є важливим кроком на шляху до доказу академічної доброчесності в контексті, коли штучний інтелект буде використовуватися все ширше в багатьох сферах нашого життя.*

Практична цінність. *Програмне забезпечення для технологій обробки тексту представляє собою комплексне дослідження, спрямоване на розроблення програмного продукту, здатного ефективно виявляти плагіат і аналізувати тексти, створені за допомогою штучного інтелекту. Розроблений інструмент відкриває нові можливості для боротьби з академічною не доброчесністю, сприяє автоматизації рутинних процесів і підтримує етичні стандарти у сфері освіти та науки.*

Ключові слова*: програмне забезпечення; API; штучний інтелект; нейронна мережа; патерни; JSON; C# ;Visual Studio; плагіат; SQLite.*

Вступ. Штучний інтелект остаточно перестав бути науковою фантастикою. З появою ШІ трансформується світова економіка та життя мільярдів людей. Генеративний штучний інтелект (ШІ) став революційною парадигмою в галузі машинного навчання, що дозволяє синтезувати складні та реальні дані за допомогою обчислювальних засобів. Здатність ШІ створювати дані, які відображають характеристики реальної інформації, знайшла застосування в таких різноманітних сферах, як комп'ютерний зір, обробка природної мови, написання музики та пошук нових ліків [1].

Машинне навчання (Machine Learning, ML) – це галузь штучного інтелекту, яка займається розробкою алгоритмів, що дозволяють комп'ютерам навчатися самостійно. Завдяки моделям ML машини здатні вирішувати широкий спектр задач, пов'язаних із аналізом даних, без потреби в конкретних алгоритмах або детальних інструкціях. Сфера застосування

машинного навчання зазвичай орієнтована на вирішення конкретних завдань, що потребують обробки інформації.

Ціль технологій штучного інтелекту полягає у відтворенні інтелектуальних здібностей і поведінки людини в різних ситуаціях. Сьогодні можливості машинного навчання доступні практично кожному. Наприклад, активуючи камеру на смартфоні, користувач користується алгоритмами, що лежать в основі мобільної зйомки, або ж може скористатися безкоштовною версією ChatGPT через веб-браузер .

Штучний інтелект високого рівня здатний моделювати шаблони мислення, переносити накопичений досвід на різноманітні нові задачі, демонструвати абстрактне мислення та творчі здібності. Крім того, він враховує соціальні й етичні аспекти під час прийняття рішень

На сьогодні за призначенням виділяють три види штучного інтелекту: ШІ вузького призначення – Artifical narrow Intelligence (ANI); ШІ загального призначення – Artifical general Intelligence (AGI), Штучний супер-інтелект – Artifical super-Intelligence (ASI) [2].

Розглянемо ці види :

1. ANI – це алгоритм, який спеціалізується на виконанні одного конкретного завдання або працює в певній сфері знань, наприклад, у пошукових системах чи управлінні безпілотними транспортними засобами. Алгоритми машинного навчання належать до категорії ANI

2. AGI – універсальна система ШІ, здатна вирішувати будь-які інтелектуальні завдання на рівні людських можливостей. AGI оперує логічними висновками та абстрактним мисленням; прикладом таких систем є великі мовні моделі (LLM).

3. ASI – це ШІ, який у всіх аспектах перевершує людські здібності, зокрема в креативності та соціальних взаємодіях. Людство активно наближається до етапу створення таких систем.

У сфері інформаційних технологій машинне навчання вважається ключовим напрямком Data Science. У певні періоди ці терміни навіть використовували як взаємозамінні. Спеціаліст з Data Science зазвичай повинен володіти знаннями у сфері машинного навчання, включаючи розуміння алгоритмів, знання мов програмування (зокрема Python) та основ математичного аналізу й статистики.

Штучний інтелект – це комп’ютери, які можуть імітувати людський інтелект і поведінку, а глобальне навчання – це підмножина машинного навчання, яке слідує за нейронними мережами [3].

Алгоритми глибокого навчання є підмножиною машинного навчання, але у прикладних завданнях їх все частіше розглядають як окремий повноцінний напрямок розвитку ШІ [4]. Глибоке навчання базується на штучних нейронних мережах – алгоритмічних структурах, що моделюють принципи людського абстрактного мислення. Великі мовні моделі та генеративний ШІ створені саме на основі технологій глибокого навчання. Цей напрямок швидко розвивається, охоплюючи такі сфери, як передовий комп’ютерний зір та системи розпізнавання природної мови. Сучасні нейронні мережі вже демонструють результати, які в окремих галузях досягають рівня людських можливостей або навіть перевершують їх.

Звичайні комп'ютерні системи використовують алгоритмічний підхід, тобто слідують набору інструкцій, щоб вирішити проблему [5]. Нейронні мережі та звичайні алгоритмічні обчислення не конкурують, а доповнюють один одного. Є завдання, які більше підходять для алгоритмічного підходу, наприклад, арифметичні операції, і завдання, які більше підходять для підходу нейронних мереж. Нейронні мережі обробляють інформацію подібно до того, як це робить людський мозок. ANN натхненний тим, як працюють біологічні нервові системи, такі як мозок – нейронні мережі навчаються на прикладі [13]**.**

Нейронні мережі широко використовуються в неконтрольованому навчанні (рис. 1) для того, щоб навчитися краще представляти вхідні дані. Наприклад, маючи набір текстових

документів, НМ може навчитися відображення від документа до вектора дійсних значень таким чином, щоб результуючі вектори були схожими для документів зі схожим змістом, тобто зі збереженням відстані [6].

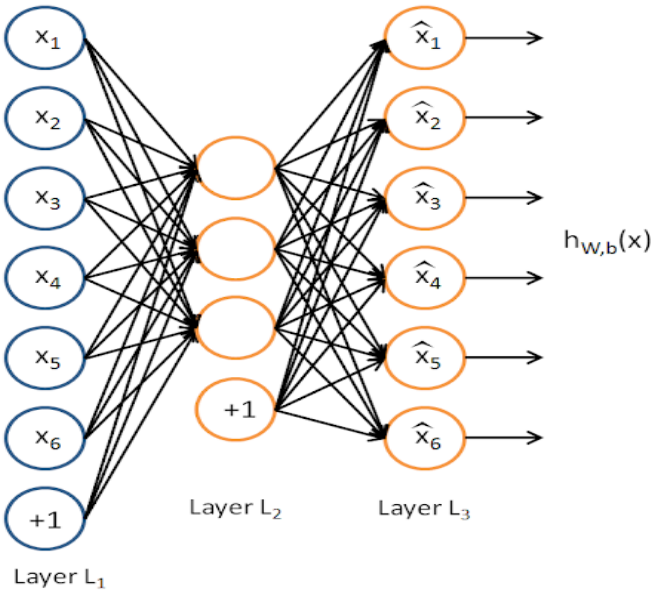

Рис. 1. **Приклад неконтрольованого навчання нейронної мережі**

Навчання під контролем, яке ще називають контрольоване машинне навчання, є однією з під категорією машинного навчання та штучного інтелекту. Його особливістю є використання маркованих наборів даних для навчання алгоритмів, які класифікують дані або точно прогнозують результати.

Коли вхідні дані подаються в модель, вона коригує свої ваги до того моменту, доки модель не стане налаштована належним чином, що відбувається в межах процесу перехресної валідації. Контрольоване навчання надає покращені можливості організаціям вирішувати різноманітні реальні проблеми в масштабі, для прикладу можна назвати класифікацію спамових повідомлень десь в ізольованій папці від вашої поштової скриньки. Його можна використовувати для побудови високоточних моделей машинного навчання.

Навчання з контролем використовує навчальний набір даних для того, щоб навчити моделі генерувати бажаний результат. Цей навчальний набір даних містить вхідні значення та правильні вихідні результати, що дозволяє моделі вчитися з часом. Алгоритм оцінює свою точність за допомогою функції втрат, коригуючи параметри до тих пір, поки помилка не буде зменшена до прийнятного рівня [7].

Постановка задачі. Реалізація технологій ШІ в створенні інструмента для оцінки оригінальності текстів є розробкою системи, що буде поєднувати систему порівняння текстів з технологіями штучного інтелекту. Враховуючи вище зазначену мету та особливості системи яку необхідно розробити, можна сформувати деякі вимоги які необхідно виконати для демонстрації об’єднання системи розпізнавання плагіату з методами ШІ, а саме:

- реалізація алгоритму перевірки текстів на плагіат;
- мати можливість інтеграції з API для виявлення текстів, створених штучним інтелектом;
- здійснення локального збереження робіт для подальшого використання в аналізі;
- розробити інтуїтивно зрозумілий графічний інтерфейс користувача;
- забезпечити сумісність з операційною системою Windows.

Реалізація задачі. Це буде нейронна мережа, яка навчена розпізнавати патерни та закономірності, що використовує штучний інтелект для написання текстів, які в подальшому можуть бути використаними недобросовісними учасниками навчального процесу для написання наукових робіт.

Нажаль навіть при наявності доступних та реалізованих архітектури нейронних мереж для створення власної системи пошуку тексту, що написаний електронними помічниками, такими як chat GPT або Copilot, потрібно провести процес навчання нейронної мережі. Оскільки для реалізації настільки масштабного проекту з адекватними результатами необхідно залучити незмірні кількості обчислювальних потужностей, залучити команду спеціалістів з глибокими знаннями окремих аспектів робочого процесу які неможливо зосередити в окремому індивіді та зібрати базу робіт гігабайтних, а краще терабайтних масштабів, що остаточно та обов'язково означає залучання значних фінансових та інших ресурсів, і призводить до закономірного рішення використання готової системи використовуючи механізми API.

Середовище розробки – це важливий набір інструментів, конфігурацій і процесів, які розробники використовують для створення, тестування та розгортання програмного забезпечення [10].

Для реалізації цього проекту було обрано середовище програмування Visual Studio, а мовою написання програми *C#*. Алгоритмом аналізу тексту на плагіат обрано косінусну подібність.

Косинусна подібність – це метрика, яка використовується для визначення косинуса кута між двома ненульовими векторами в багатовимірному просторі. Це міра орієнтації, а не величини, в діапазоні від -1 до 1. В контексті схожості тексту ця метрика забезпечує надійний спосіб оцінити схожість між двома наборами текстових даних [8].

Математичне рівняння [1] косинусної подібності між двома ненульовими векторами:

$$\text{подібність} = \cos(\theta) = \frac{A * B}{\|A\|\|B\|}, \quad (1)$$

де $\cos\theta$ – це косинус кута θ між двома векторами A і B. Менший кут відповідає більшій схожості.

В контексті обробки природної мови – це міра того, наскільки схожі ідеї та концепції, представлені в двох фрагментах тексту.

Загалом, одним з основних завдань систем оцінювання оригінальності текстів є перетворення технічних результатів роботи алгоритмів у формат, зрозумілий кінцевому користувачу. У випадку аналізу на основі косинусної схожості результати, отримані в діапазоні [0,1], слід інтерпретувати як показник схожості тексту. Для цього результати нормалізуються і представляються відповідно [9].

Коефіцієнт косинусної подібності [2] коливається від 0 (немає подібності) до 1 (повністю подібний) після проведення порівняння. Для того, щоб різниця значень сприймалася більш наочно, вони перетворюються у відсоткові значення шляхом множення результату на 100.

$$\text{Схожість } (\%) = \cos\theta * 100. \quad (2)$$

Наприклад, якщо значення cos(θ) дорівнює 0,75 означає, що текст має 75% схожості з базовим зразком

Шляхи вирішення проблеми порівняння тексту, це методи аналізу мовних моделей, синтаксису та стилю для виявлення контенту, написаного ШІ.

Це пов'язано з тим, що навіть при високому ступені вільного володіння мовою, який може зробити її схожою на людську, ШІ завершує текст з певним «відбитком» у структурі та

лексиці. Більше того, інструменти розпізнавання навчаються на великих вибірках як людських, так і штучно створених текстів, щоб виявляти такі шаблони, які набагато рідше зустрічаються в людському письмі. Наприклад, штучний контент може використовувати повторювані структури речень, передбачувані фрази або неприродний вибір слів, які звучать відшліфовано, але позбавлені нюансів людської експресії.

Інший підхід пов'язаний з моделюванням ймовірностей. Моделі ШІ розпізнають ймовірні комбінації слів і структур речень, які органічно виглядають у мові. Підозри на причетність ШІ виникають, коли патерни слів у тексті відхиляються від цих органічних патернів. У зв'язку з цим OpenAI використовує цей вид імовірнісного аналізу для оцінки того, наскільки той чи інший твір у класифікаторі ШІ написаний людиною, серед інших інструментів

В даній роботі використання API зумовлено нераціональною кількістю фінансових та інших ресурсів які потрібно використати для навчання власної нейронної мережі для пошуку фрагментів тексту створених за допомогою штучного інтелекту. Пошук в тексті штучного інтелекту втілено з використанням Application Programming Interface. API спрощують і прискорюють розробку додатків і програмного забезпечення, дозволяючи розробникам інтегрувати дані, послуги та можливості з інших додатків замість того, щоб розробляти їх з нуля [12].

Процес створення проекту є обов’язковим і ключовим моментом створення системи представлено на рис. 2.

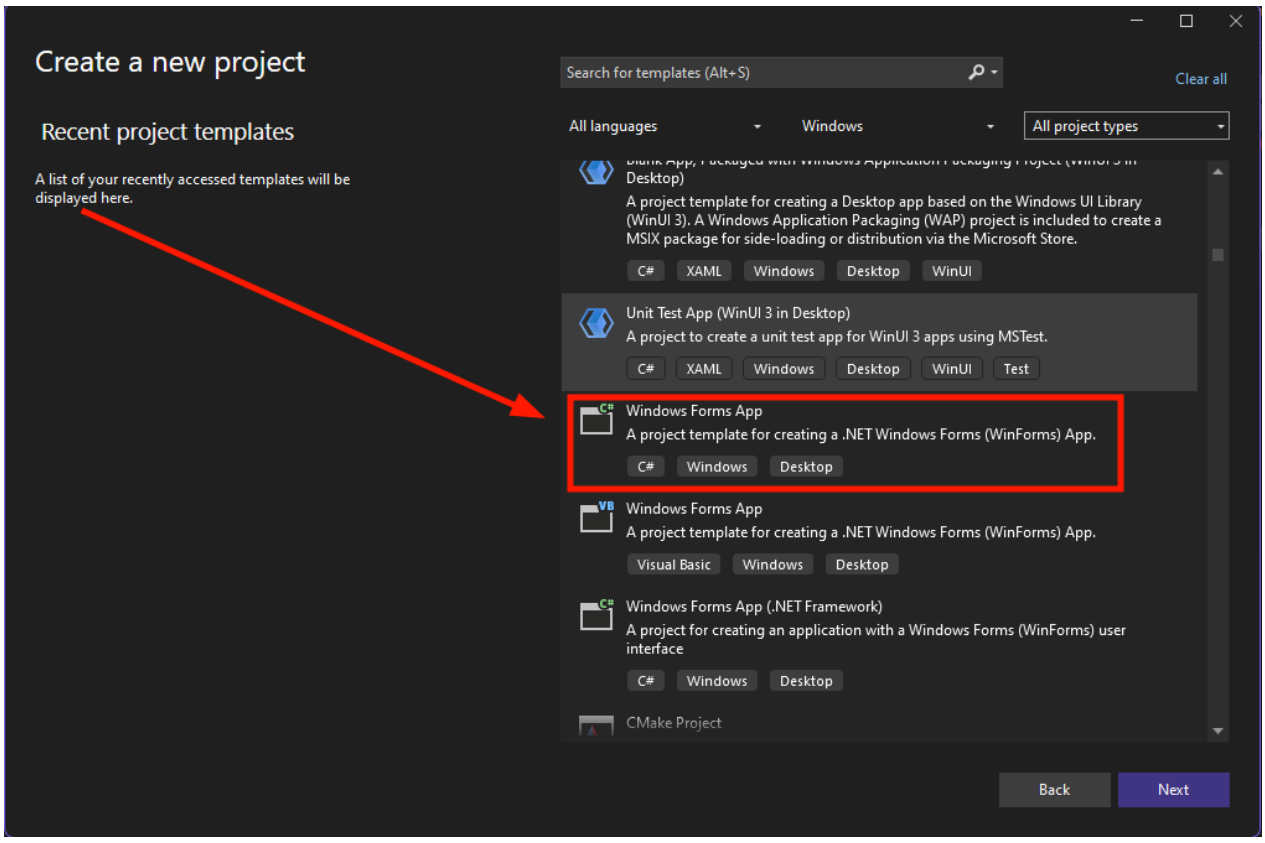

Рис. 2. **Створення проекту**

Створюємо базу даних в якій будуть зберігатись роботи для аналізу (рис. 3).

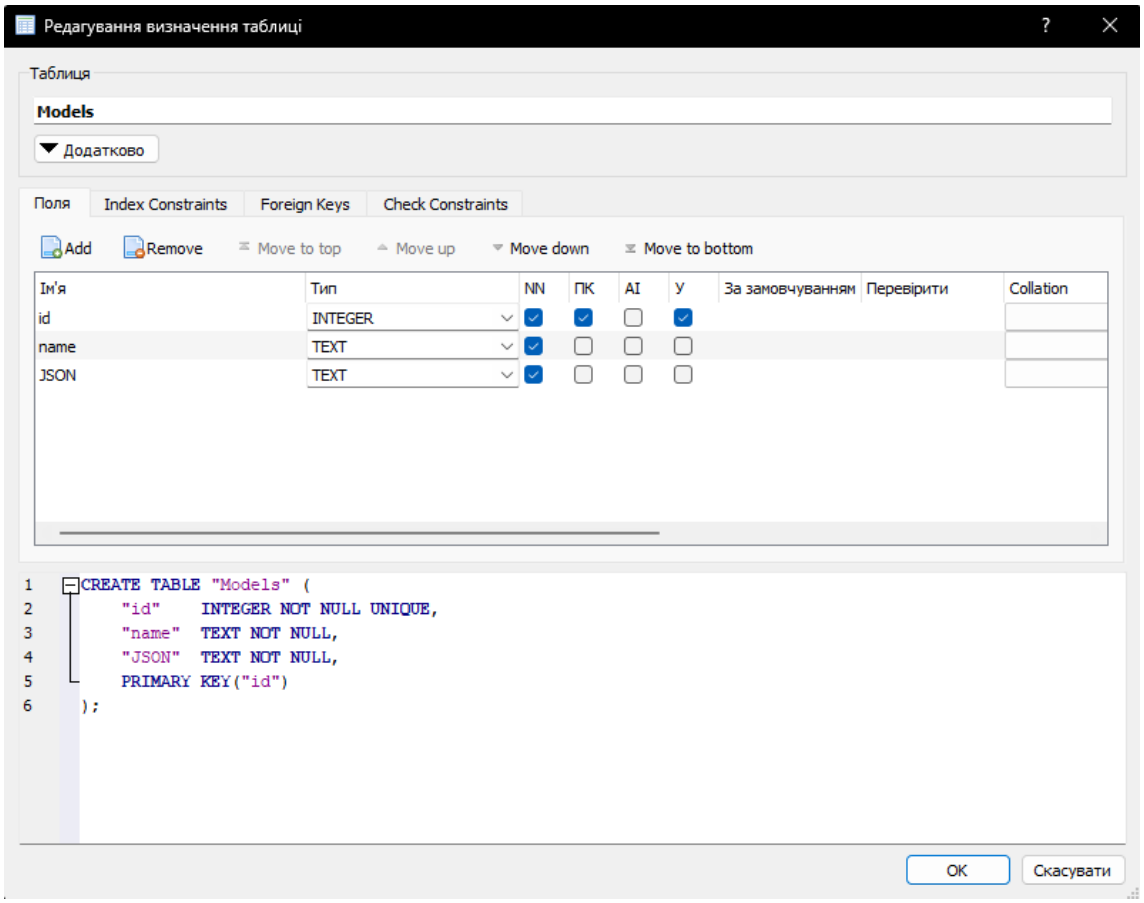

Рис. 3. **Створена база даних**

Тепер, коли усі обов'язкові етапи налаштування пройдені ми можемо перейти до підключення БД до проекту, шляхом створення відповідного програмного коду (рис. 4).

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SQLite;

namespace WinFormsApp1
{
    0 references
    internal class BD
    {
        string connectionString = "Data Source=database.sqlite;Version=3;";
    }
}
```

Рис. 4. **Скріншот коду**

Для зручності використання застосунку клієнтом потрібно реалізувати систему яка зможе надати можливість зручно обирати файл формату docX і зчитувати наведену в ньому інформацію в змінну типу string для подальших маніпуляцій. Для реалізації цього аспекту обрано бібліотеку під назвою DocX,

Для користувача створимо візуальний інтерфейс у вигляді клавіши Вибрати документ для аналізу (рис. 5).

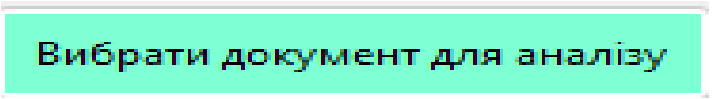

Рис. 5. **Клавіша Вибрати документ для аналізу**

Функціоналом для цієї клавіші буде окрема функція, яка при натисканні на клавішу викликає подію, що відкриває вікно обрання файлу

Також для порівняння робіт між собою неможливо обійтись без можливості функціонального вирішення питання зчитування файлів з бази даних наукових робіт призначених для перевірки. Реалізуємо цю задачу за допомогою створеної функції ReadDocFile (рис. 6, що повинна брати інформацію з створеної бази даних та почергово записує цю інформацію в окремий масив даних для подальшого зручного використання при порівнянні текстів.

```
private void ReadDocFile(string filePath)
{
    try
    {
        // Відкриваємо документ
        using (var document = DocX.Load(filePath))
        {
            // Отримуємо весь текст документа
            string content = document.Text;

            // Виводимо в консоль або записуємо в змінну
            MessageBox.Show(content);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Помилка: {ex.Message}");
    }
}
```

Рис. 6. **Програмний код функції**

Останнім, але не менш важливим аспектом роботи з JSON, є можливість запису створених об’єктів в розроблену базу даних, цей процес має автоматично проводитись при закінченні аналізу й порівняння обраної роботи. JSON – це загальний формат для представлення значень і об'єктів [11].

Такий підхід забезпечить те, що при перевірці наступної роботи, які частіше за все перевіряються одразу великим набором, робота що перевіряється в даний момент буде використовуватись для порівняння, при чому не змусить користувача напружено контролювати цей процес і покращить досвід роботи з створеним проектом.

Побудова алгоритму методу косинусної подібності була виконана мовою C#, як і весь проект, та бібліотекою RegularExpressions

Виконавши програмну частину проведемо тестування відповідності на плагіат і подивимось на косинус кута двох тестових фраз, що в майбутньому буде перетворений у відсоткове значення.

Розглянемо тестову фразу номер 1 – This is a sample document і тестову фразу 2 – This document is a sample. Плагіат цих тестових фраз дорівнює 1 (рис. 7).

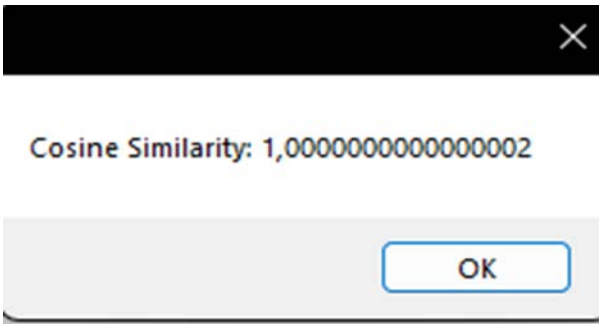

Рис. 7. **Результат плагіату текстових фраз**

Перевірка текстових фраз на наявність ШІ була проведена з використанням API. Створено програмний код, який розділяє текст на частини залежно від заданої мінімальної та максимальної кількості слів і перевіряє ці частини через API сервісу Grammarly. Це здійснюється методом SplitTextIntoParts, який працює з масивом слів і створює рядки заданої довжини, об'єднуючи їх назад у частини тексту.

Потім програма створює HTTP-запит для доступу до API Grammarly. Вона починає з відправки GET-запиту для отримання початкової сторінки, де зберігаються необхідні cookie-файли. Ці файли потрібні для авторизації подальших запитів.

Далі кожна частина тексту, отримана після розділення, відправляється через POST-запити до API Grammarly, використовуючи збережені cookie-файли. Результати обробки кожної частини тексту виводяться на екран.

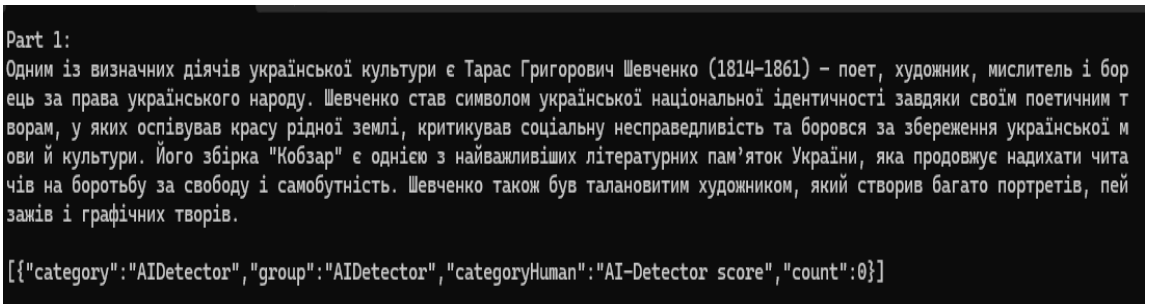

Рис. 8. **Результат відповіді на запит для тестової фрази**

Результати аналізу тесту на використання ШІ повертаються значенням у JSON форматі. Це останній рядок у фрагменті (рис. 8), де значення count = 0 говорить про те, що штучний інтелект не був використаний.

Висновки. Розробка програмного забезпечення для технологій обробки тексту представляє собою комплексне дослідження, спрямоване на розроблення програмного продукту, здатного ефективно виявляти плагіат і аналізувати тексти, створені за допомогою штучного інтелекту.

В роботі було реалізовано такі аспекти програми:

- Підключили базу даних, що дозволить нам зберігати роботи студентів призначені для пошуку плагіату. Кожна робота в цій базі покращує результати аналізу для наступних перевірок.
- Зчитування docX файлів, що надасть можливість спростити використання цього продукту користувачами та зекономити час завдяки зручному та інтуїтивно зрозумілому способу додавання робіт в програму.

- Робота з JSON – конвертація, запис та зчитування JSON файлів дасть можливість краще працювати з базою даних та полегшить впровадження технологій API які використовують даний формат для надання результатів роботи.
- Реалізація методу косинусної подібності створить надійний інструмент виявлення не доброчесного копіювання змісту чужих робіт і є важливим аспектом цієї роботи.
- Використання API замінить дороге та ресурсне навчання власної нейронної мережі для пошуку тексту написаним штучним інтелектом.

Результатом стало створення потужного інструмента, який допомагає не лише виявляти плагіат, але й сприяє збереженню академічної доброчесності.

Таким чином, проведене дослідження та практична реалізація довели, що інтеграція технологій штучного інтелекту у процес перевірки текстів на оригінальність є перспективним напрямом. Розроблений інструмент відкриває нові можливості для боротьби з академічною не доброчесністю, сприяє автоматизації рутинних процесів і підтримує етичні стандарти у сфері освіти та науки.

References

1. Deckler, G. (2022). Learn Power BI – Second Edition: A comprehensive, step-by-step guide for beginners to learn real-world business intelligence: Model. 2nd ed. Publisher Packt Publishing. 8 p.

2. Tkachenko, R. O., Kustra, N. O., Pavliuk, O. M., Polishchuk, U. V. (2014). Zasoby shtuchnoho intelektu: navch. posibnyk [Means of artificial intelligence: a textbook]. Lviv: Lviv Polytechnic University Press. 204 p. [in Ukrainian].

3. Nikolsky, Y. V. (2013). Systemy shtuchnoho intelektu: navch. posibnyk [Systems of artificial intelligence: a textbook]. Lviv: Magnolia-2006. 279 p. [in Ukrainian].

4. Hlybovets, M. M., Oletsky, O. V. (2002). Shtuchnyi intelekt [Artificial Intelligence]. Kyiv: Kyiv-Mohyla Academy. 364 p. [in Ukrainian].

5. Nikolsky, Y. V., Pasichnyk, V. V., Shcherbyna, Y. M. (2013). Systemy shtuchnoho intelektu: navch. posib. [Artificial Intelligence Systems: a textbook]. Lviv: Magnolia-2006. 279 p. [in Ukrainian].

6. What is a Neural Network? *AWS*. URL: https://aws.amazon.com/what-is/neural-network/

7. Simon, J. D. Prince, Understanding Deep Learning: Reinforcement learning. 11 p.

8. Varun (2020). Cosine similarity: How does it measure the similarity, Maths behind and usage in Python. *Medium*. 27.09.2020. URL: https:// towardsdatascience.com/cosine-similarity-how-does-it-measure-the-similarity-maths-behind-and-usage-in-python-50ad30aad7db.

9. Prakash, A. (2023). Understanding Cosine Similarity: A key concept in data science. *Medium*. 21.09.2023. URL: https://medium.com/@arjunprakash027/understanding-cosine-similarity-a-key-concept-in-data-science-72a0fcc57599.

Література

1. Deckler G. Learn Power BI – Second Edition: A comprehensive, step-by-step guide for beginners to learn real-world business intelligence: Model. 2nd ed. Publisher Packt Publishing, 2022. 8 p.

2. Ткаченко Р. О., Кустра Н. О., Павлюк О. М., Поліщук У. В. Засоби штучного інтелекту: навч. посібник. Львів: Вид-во Львів. політехніки, 2014. 204 с.

3. Нікольський Ю. В. Системи штучного інтелекту: навч. посібник. Львів: Магнолія-2006, 2013. 279 с.

4. Глибовець М. М., Олецький О.В. Штучний інтелект. Київ: Києво-Могилянська академія, 2002. 364 с.

5. Нікольський Ю. В., Пасічник В. В., Щербина Ю. М. Системи штучного інтелекту: навч. посіб. Львів: Магнолія-2006, 2013. 279 с.

6. What is a Neural Network? *AWS*. URL: https://aws.amazon.com/what-is/neural-network/

7. Simon J. D. Prince, Understanding Deep Learning: Reinforcement learning. 11 p.

8. Varun. Cosine similarity: How does it measure the similarity, Maths behind and usage in Python. *Medium*. 27.09.2020. URL: https://towardsdatascience.com/cosine-similarity-how-does-it-measure-the-similarity-maths-behind-and-usage-in-python-50ad30aad7db.

9. Prakash A. Understanding Cosine Similarity: A key concept in data science. *Medium*. 21.09.2023. URL: https://medium.com/@arjunprakash027/understanding-cosine-similarity-a-key-concept-in-data-science-72a0fcc57599.

10. Getting to Know Visual Studio. *Visual Studio*. URL: https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community&channel=Release&version=VS2022&source=VSLandingPage&cid=2030&passive=false.

11. Marrs, T. JSON at Work: Practical Data Integration for the Web: Core JSON. 8 p.
12. Gough, J., Bryant, D., Auburn, M. Mastering API Architecture: Design, Operate, and Evolve API-Based Systems.
13. Astistova, T. I., Sedlyar, A. O. (2024). AItechnology in the creation of a tool for assessing the originality of texts. *Proceedings of the VIII International Scientific MSIE-2024*. Kyiv: KNUTD. P. 280–281.

10. Getting to Know Visual Studio. *Visual Studio*. URL: https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community&channel=Release&version=VS2022&source=VSLandingPage&cid=2030&passive=false.
11. Marrs T. JSON at Work: Practical Data Integration for the Web: Core JSON. 8 p.
12. Gough J., Bryant D., Auburn M. Mastering API Architecture: Design, Operate, and Evolve API-Based Systems.
13. Astistova T. I., Sedlyar A. O. AItechnology in the creation of a tool for assessing the originality of texts. *VIII Міжнародна науково-практична конференція MSIE-2024*. К.: КНУТД, 2024. С. 280–281.

ASTISTOVA TETYANA
Candidate of Technical Science, Associate Professor,
Department of Computer Sciences,
Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0000-0002-8452-4797
Scopus Author ID: 6506601603
E-mail: astistova@ukr.net

SEDLIAR ANDRII
Student
Department of Computer Sciences,
Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0009-0006-6815-2950
E-mail: sedlyar.andrey.oleks@gmail.com

ASTISTOVA T. I., SEDLIAR A. O.
Kyiv National University of Technologies and Design, Ukraine

DEVELOPMENT OF SOFTWARE FOR EVALUATION OF OF TEXT ORIGINALITY

Objective. *Develop software that combines standard methods and the use of artificial intelligence technologies to evaluate the originality of the text when writing scientific papers.*

Methods. *The development of the software was based on theoretical and experimental studies used to analyze the methods of text recognition generated by artificial intelligence, the selection and analysis of a neural network to identify patterns and corresponding patterns in human and artificial intelligence text writing and the integration of this model into the software. to check the text for originality, the implementation of text analysis technologies.*

Results. *The main results of the work can be divided into three interrelated aspects: analytical, design, and practical, each of which made a significant contribution to the achievement of the goal. The research resulted in a thorough analysis of the subject area, the concept of artificial intelligence, its architecture, the functioning of neural networks and their application to text analysis, the technical characteristics and capabilities of modern AI models, as well as aspects of their integration into the text review process, the issues of academic integrity, including the definition of plagiarism, its types and cultural peculiarities of perception of this phenomenon in the processes of education and research, the theoretical basis for the development of a tool that is able to take into account both technical and ethical aspects.*

The structure of the program, its algorithms, methods of integration with the API for text analysis and data storage format are considered in detail, and a similarity algorithm is developed, which became the basis for identifying the similarity of texts. This allowed us to create a clear and logical product architecture that ensures its functionality and efficiency.

Scientific novelty. *An original solution is proposed in the form of an integrated system that combines classical methods of checking text for originality with the recognition of texts created by artificial intelligence,*

which is an important step towards proving academic integrity in a context where artificial intelligence will be used more and more widely in many areas of our lives.

***Practical value**. Thesoftware for text processing technologies is a comprehensive study aimed at developing a software product capable of effectively detecting plagiarism and analyzing texts created with the help of artificial intelligence. The developed tool opens up new opportunities for combating academic dishonesty, facilitates the automation of routine processes, and supports ethical standards in the field of education and science.*

***Keywords:** software; API; artificial intelligence; neural network; patterns; JSON; C#; Visual Studio; plagiarism; SQLite.*