МЕТОДИ ЗНАХОДЖЕННЯ КЛЮЧОВИХ СЛОВОСПОЛУЧЕНЬ

Олена Шапошнікова[1], Андрій Білик[2]
[1]*Харківський* національний автомобільно-дорожній університет, Харків, *Україна, ORCID 0000-0002-0405-8205, e-mail:* shaposhnikovaep@gmail.com
[2]*Харківський національний автомобільно-дорожній університет, Харків*.

В сучасному світі майже всі документи створюються в електронному вигляді та зберігаються на комп'ютері. Однак документів настільки багато, що виникає складність у здійсненні ефективного пошуку потрібного документу. Для спрощення цього процесу виникає необхідність в їхньому індексуванні. Проте розміри текстового об'єму можуть бути настільки великі, що ускладнюють завантаження до Azure [1] або до баз даних, для їх подальшого пошуку.

Для розв'язання даної задачі доступні різноманітні алгоритми, включаючи «Spacy» та «RAKE-NLTK» [2]. Більшість існуючих алгоритмів, які використовуються у програмах, спрямовані на зменшення обсягу тексту з метою зберігання потрібної інформації на сервері (у базі даних) для зменшення кінцевого обсягу даних. Але ці алгоритми можуть бути не ефективними з точки зору швидкодії, особливо на слабких системах.

Наразі одним із найбільш перспективних методів для вирішення задачі скорочення обсягу тексту є алгоритм «RAKE» (Rapid Automatic Keyword Extraction) [3]. Цей метод базується на принципі аналізу частотності та поєднання слів у тексті, дозволяючи виділити найбільш важливі терміни без необхідності попереднього навчання моделі.

Система «RAKE» виконує наступні основні функції:

- розбиття введеного тексту на окремі слова або фрази (цей крок є важливим для подальшого аналізу слів);
- визначення можливих ключових слів шляхом аналізу шаблонів, що включає в себе виділення слів, які часто зустрічаються тексті;
- надання балів кожному слову на основі його частоти входження складових слів;
- аналіз оцінок потенційних ключових слів, ранжування їх за популярністю та вибір найбільш важливих термінів або фраз у якості кінцевих виллучених ключових слів.

При пошуку надто великих файлів «RAKE» не завжди справляється з задачею стискання розміру тексту до розмірів, які дозволяють завантажити його до Azure. Тому було розроблено інший алгоритм, який ще ефективніше зменшує обсяг тексту – це «Frequently occurring». «Frequently occurring» працює дуже швидко і ефективно на слабких системах та дуже сильно зменшує об'єм тексту. Варто зауважити, що алгоритм «Frequently Occurring» не визначає ключові словосполучення, а замість цього запам'ятовує ключові слова та не видаляє символи з тексту. А це може негативно вплинути на подальший пошук цього файлу .

На рис. 1 зображено інтерфейс програми індексації файлів «Index File»[4], де для проведення дослідження застосовуються ці два алгоритми.

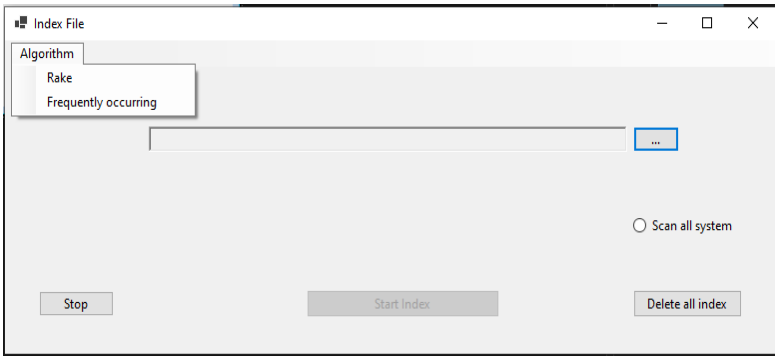

Рисунок 1 – Інтерфейс «Index File»

Для оцінки ефективності роботи алгоритмів «RAKE» та «Frequently Occurring» був здійснений порівняльний аналіз.

«Frequently occurring» виявляє часто повторювані слова, букви та символи, які зустрічаються більше 1-го разу. На рисунок 2 наведено результати роботи цього алгоритму.

IEnumerable Visualizer

Filter

Export to CSV...

KeyValuePair`2	KeyValuePair`2.key	KeyValuePair`2.value	KeyValuePair`2.Key	KeyValuePair`2.Value
[міністерство, 1]	міністерство	1	міністерство	1
[освіти, 2]	освіти	2	освіти	2
[і, 17]	і	17	і	17
[науки, 1]	науки	1	науки	1
[україни харківський, 1]	україни харківський	1	україни харківський	1
[національний, 1]	національний	1	національний	1
[автомобільно, 1]	автомобільно	1	автомобільно	1
[дорожній, 1]	дорожній	1	дорожній	1
[університет кафедра, 1]	університет кафедра	1	університет кафедра	1
[комп'ютерних, 1]	комп'ютерних	1	комп'ютерних	1
[технологій, 2]	технологій	2	технологій	2
[мехатроніки курсова, 1]	мехатроніки курсова	1	мехатроніки курсова	1
[робота1 з, 1]	робота1 з	1	робота1 з	1
[дисципліни, 1]	дисципліни	1	дисципліни	1
[«програмування, 1]	«програмування	1	«програмування	1
[в, 39]	в	39	в	39
[ос, 1]	ос	1	ос	1
[android» на, 1]	android» на	1	android» на	1
[тему «розробка, 1]	тему «розробка	1	тему «розробка	1
[android, 190]	android	190	android	190
[додатку, 16]	додатку	16	додатку	16
[«читалка, 2]	«читалка	2	«читалка	2
[електронних, 7]	електронних	7	електронних	7
[книг»» студента, 1]	книг»» студента	1	книг»» студента	1

Total Rows: 1890

Рисунок 2 – Робота алгоритму «Frequently occurring»

«RAKE» [5] спочатку розбиває текст на окремі слова або фрази, а потім створює список можливих ключових слів. Він надає бали кожному кандидату в ключові слова на основі його частоти та частоти складових його слів. На рисунку 3 представлені результати роботи алгоритму «RAKE», який визначає ключові словосполучення.

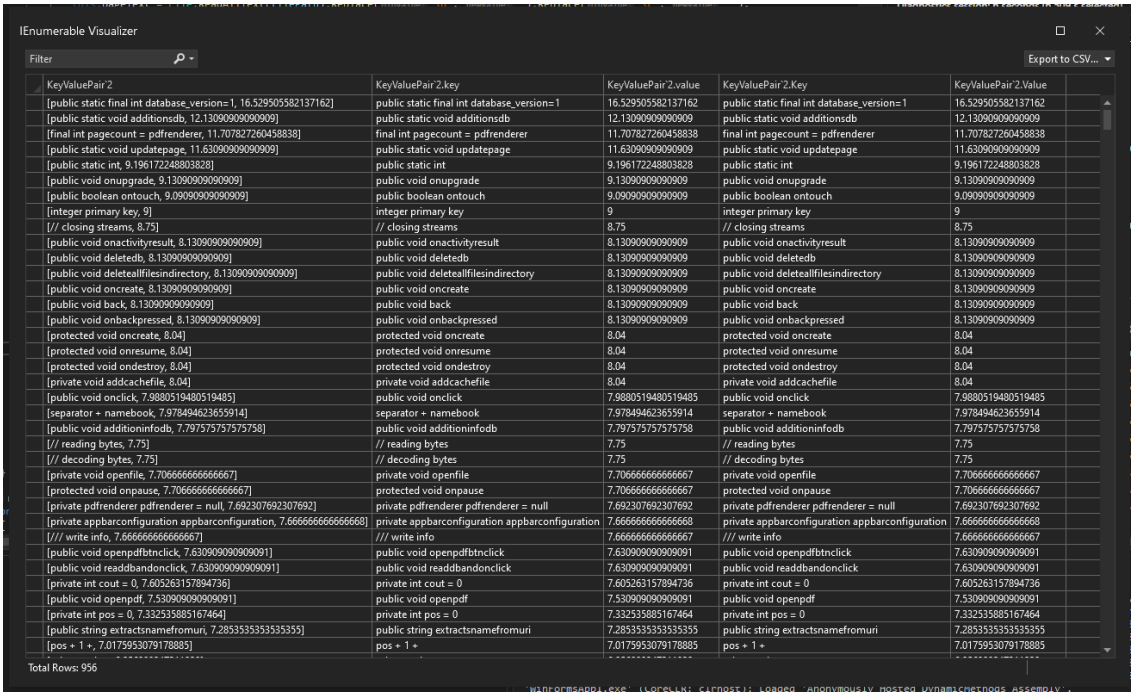

Рисунок 3 – Робота алгоритму «RAKE»

Таким чином було виконано порівняльний аналіз алгоритмів, «Frequently occurring» і «RAKE», результати якого наведені у табл. 1.

Таблиця 1 – Результати порівняльного аналізу алгоритмів «Frequently «occurring» і «RAKE».

Критерій	RAKE	Frequently occurring
Швидкість	-	+
Виділення ключових слів	-	+
Виділення ключових словосполучень	+	-
Розмір тексту	-	+

В результаті проведеного дослідження було визначено, який з цих алгоритмів більше підходить для збереження сенсу тексту, а який для зменшення його розміру.

По результатам дослідження можна зробити висновки, що алгоритм «Frequently occurring» більше підходить для ефективного зменшення об’єму тексту та знаходження слів, які часто зустрічаються, при чому у рейтинг включаються не лише слова, а й інші знаки, які часто зустрічаються, але при пошуку файлу по словосполученням, може бути не ефективним.

«RAKE» особливо корисний у випадках, коли необхідно зберегти ключові словосполучення для подальшого пошуку файлів, де ці словосполучення зустрічаються. Його застосування включає можливість зменшити текст, для подальшої індексації, що полегшує і прискорює процес пошуку файлу.

Список використаних джерел

1. Що таке Azure index [Електронний ресурс] – Режим доступу: https://learn.microsoft.com/ru-ru/azure/search/search-what-is-an-index
2. Інші алгоритми [Електронний ресурс] – Режим доступу: https://medium.com/international-school-of-ai-data-science/the-keyword-quest-exploring-automatic-keyword-extractors-db553c6ac229
3. Що таке RAKE [Електронний ресурс] – Режим доступу: https://ovchinnikov.cc/writing/rake/
4. Реалізація програми [Електронний ресурс] – Режим доступу: https://github.com/Xaker53/azure-search-dotnet.git

УДК 004.42.4

СТВОРЕННЯ ДОДАТКІВ НА WINDOWS FORMS: АРХІТЕКТУРА ТА ФУНКЦІОНАЛЬНІ МОЖЛИВОСТІ

Клячко М.М., Лебединський А.В.

Харківський національний автомобільно-дорожній університет, Харків

У сучасному світі, де Windows залишається однією з найпоширеніших операційних систем, розробка додатків для неї не втрачає своєї актуальності. Існує кілька різних платформ, які розробники можуть використовувати для створення програм для Windows. C# є відмінним вибором для створення програм для Windows завдяки тісній інтеграції з платформою .NET та інструментами розробки Microsoft. Ось деякі ключові аспекти використання C# для розробки програм під Windows:

- інтеграція з .NET Framework/.NET Core: C# щільно інтегрований з платформою .NET, що забезпечує широкі можливості розробки різноманітних додатків під Windows. Залежно від вимог проекту, ви можете вибрати .NET Framework [1] для традиційних настільних додатків або .NET Core/.NET 5+ для більш сучасних і кросплатформових додатків;
- Windows Presentation Foundation (WPF): WPF - це потужний фреймворк для створення настільних програм під Windows із використанням C#. WPF надає широкі можливості для створення інтерактивних інтерфейсів користувача за допомогою XAML (eXtensible Application Markup Language) і підтримує різні функції, такі як стилі, шаблони, анімації та багато іншого;
- Windows Forms (WinForms): Для створення класичних настільних програм під Windows також можна використовувати Windows Forms. Це більш