Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(рівень вищої освіти)

Методи аналізу коду веб-сторінок (HTML) з метою виявлення важливої інформації .
(тема)

Виконав:
студент 2 курсу, групи СШІм-18-2
Лизогуб Р.С.
(прізвище, ініціали)
Спеціальність 122 – Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми Освітньо-наукова
(освітньо-професійна або освітньо -наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва освітньої програми)

Керівник доц. Узлов Д.Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти другий (магістерський)

Спеціальність 122 – Комп'ютерні науки
(код і повна назва)

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

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри
_____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Лизогуб Руслан Сергійович
(прізвище, ім'я, по батькові)

1. Тема роботи Методи аналізу коду веб-сторінок (HTML) з метою виявлення важливої інформації
затверджена наказом по університету від 30.03.20 р. №480Ст
2. Термін подання студентом роботи до екзаменаційної комісії 21 травня 2020 р.
3. Вихідні дані до роботи аналіз існуючих підходів до виявлення важливої інформації з веб -сторінки та створення алгоритму для досягнення такої мети. В роботі вивчаються існуючі проблеми та рішення, а також запропонована модель побудови структурованої інформації на веб-сторінці для подальшого її використання.

4. Перелік питань, що потрібно опрацювати в роботі аналіз предметної галузі, аналіз підходів та постановка задачі, моделювання алгоритму, опис розробленої системи.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Рисунок 1 – загальний взаємозв'язок між категоріями Web Mining, Рисунок 2 – сторінка The New York Times, Рисунок 3 – Приклади дерева DOM, Рисунок 4 – Дерево DOM і дерево стилів, Рисунок 5 – Зовнішній вигляд і код блоку HTML документа, Рисунок 6 – Веб-сторінка сайту oracle.com, Рисунок 7 – Веб-сторіка з шаблонними елементами, Рисунок 8 – Процес вибору класу елементу для блоку, Рисунок 9 – Структкра розлядаємого в алгоритмі документу, Рисунок 10 – Результат класифікації елементів, Рисунок 11 -Результат об'єднання класів сусідніх елементів, Рисунок 12 – Результат об'єднання груп класів елементів, Рисунок 13 – Результат обробки дочірніх елементів головних блоків, Рисунок 14 – Сторінки, яка використовується для демонстрації роботи програмного додатку, Рисунок 15 – Результат роботи програми, як сторінки з важливою інформацієї, Рисунок 16 – Результат роботи програми, як сторінка з виділенною важливою інформацію

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

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	Доцент Узлов Д.Ю.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Об'єктний аналіз поставленої задачі	04.04.20 – 06.04.20	Виконано
2	Розробка моделі взаємодії даних	07.04.20 – 15.04.20	Виконано
3	Розробка структури зберігання даних	16.04.20 – 20.04.20	Виконано
4	Створення коду програми	21.04.20 – 29.04.20	Виконано
5	Тестування і налагодження програми	30.04.20 – 03.05.20	Виконано
6	Підготовка пояснювальної записки	03.05.20 – 13.05.20	Виконано
7	Нормоконтроль, рецензування	14.05.20 – 15.05.20	Виконано

Дата видачі завдання 30 березня 2020 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Записка пояснювальна: 80 с., 16 рис., 2 дод., 14 джерел.

PYTHON, HTML, CSS, LXML, DATA MINING, ДОДАТОК, АНАЛІЗ

Об’єктом дослідження є сучасні технології, методи та засоби реалізації алгоримів для виявлення важливої інформації на веб-сторінці.

Метою роботи є вивчення існуючих підходів для аналізу коду веб-сторінок, створення алгоритму для аналізу сторінки та побудови структурованної моделі для подальшого використовування для задач вилучення інформації та пошуку, а також створення демонстраційного програмного додатку. Метою створення програми є навчання програмуванню та застосування набутих за час навчання знань та навичок на практиці.

Метод розробки оснований на використанні мови програмування Python.

У результаті розроблено алгоритм та програмний додаток, що знаходить важливу інформацію на веб-сторінці.

РЕФЕРАТ

Пояснительная записка: 80 с., 16 рис., 2 прил., 14 источников.

PYTHON, HTML, CSS, LXML, DATA MINING, ПРИЛОЖЕНИЕ, АНАЛИЗ

Объектом исследования являются современные технологии, методы и средства реализации алгоримив для выявления важной информации на веб-странице.

Целью работы является изучение существующих подходов для анализа кода веб-страниц, создание алгоритма для анализа страницы и построения структурованнои модели для дальнейшего использования для задач извлечения информации и поиска, а также создание демонстрационного программного приложения. Целью создания программы является обучение программированию и применение полученных за время обучения знаний и навыков на практике.

Метод разработки основан на использовании языка программирования Python.

В результате разработан алгоритм и программное приложение, которое находит важную информацию на веб-странице.

ABSTRACT

Explanatory note: 80 pp., 16 figs., 2 ann, 14 sources.

PYTHON, HTML, CSS, LXML, DATA MINING, APPLICATION, ANALYSIS

The object of study is the advanced technology, methods of implementing algorithms to identify important information on web page.

The purpose of this paper is to study existing approaches to analyze the code of web pages, creating an algorithm to analyze the page and build structural model for use for the tasks of information extraction and search, and creating demonstration software application. The purpose of the program is to teaching of programming, application of acquired knowledge and skills in practice.

The development method is based on the use of Python programming language.

As a result, the algorithm and software application which find important information on web page were developed.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

HTML – HyperText Murkup Language – Мова гіпертекстової розмітки;

CSS - Cascading Stylesheets - каскадні таблиці стилів;

BTE - Body Text Extraction - Вилучення тексту тіла;

FE - Feature Extractor - Функція витяжки;

DOM - Document Object Model - Модель об'єкта документа;

SST - Tree Style Text - Деревом стилів тексту

HITS - Hyperlink Induced Topic Search - Пошук за темами гіперпосилання

ВСТУП

Всесвітня павутина є найрізноманітнішим джерелом інформації. Однією з головних причин збільшення кількості сайтів є відсутність будь-яких суворих правил подання інформації та відносна простота використовуваних для цього технологій. Мова HTML, що використовується для створення веб-сторінок, дає авторам достатню свободу представляти будь-які дані за мінімальних витрат. Також розробник може використовувати такі технології, як CSS, щоб отримати бажану якість презентації. Разом із гіпертекстовим характером документів ці властивості роблять Всесвітню павутину розповсюдженим та динамічним джерелом інформації.

Веб-сторінку можна вважати одним із видів джерел інформації, які можна використовувати для різних цілей. Крім простого перегляду веб-сторінки користувачем, існують також проблеми, пов’язані з отриманням інформації для будь-якої конкретної мети, яка вимагає аналізу інформації та пошуку. Такими завданнями можуть бути: отримання (та вилучення) конкретних уніфікованих даних з різних джерел (веб-сторінок), наприклад, вибір серед інтернет-магазинів, порівняння обмінних курсів тощо. Також існують системи пошуку в Інтернеті, які потребують роботи зі змістом веб-сторінка.

Одним із цікавих прикладів аналізу та вилучення необхідної інформації є сайти для перегляду інших веб-сторінок, а саме з можливістю виявлення лише певної необхідної інформації, яка може бути надалі відформатована або перетворена в інший формат, наприклад, статті можуть бути обрані з різних блогів лише тема та текст самих статей і форматування їх унікальним способом або перетворюються на документ.

Аналіз та аналіз коду веб-сторінки можна вважати одним із етапів процесу Data Mining [3]. Спочатку потрібно знайти частини документа, які мають важливу інформацію, а потім проаналізувати ці блоки. Вищезазначені

завдання в основному виглядають тривіально, але для певних завдань є проблеми, пов’язані з аналізом коду веб-сторінки.

Основна проблема - сама мова розмітки HTML. Ця мова чудово підходить для оформлення зовнішнього вигляду веб-документа, а разом із CSS - дозволяє красиво оформляти сторінку. Однак у більшості випадків ця мова не має семантики за змістом документа, що ускладнює аналіз інформації на веб-сторінці. В останніх версіях HTML з'явилися теги, які можуть надати інформацію про мету його вмісту, але цього недостатньо. Багато стандартних тегів, таких як заголовки, теоретично можуть також описувати вміст документа, але сучасні веб-сторінки мають дуже хаотичний набір різних тегів, які використовуються у всіх частинах документа. З боку коду сторінки, враховуючи синтаксис і призначення HTML, а також абсолютну свободу в написанні коду для веб-сторінки - завдання визначення основної інформації стає нетривіальним.

Метою даної роботи є аналіз існуючих підходів до виявлення важливої інформації з веб-сторінки та створення алгоритму для досягнення такої мети, а також створення демонстраційної програми. В роботі вивчаються існуючі проблеми та рішення, а також запропонована модель побудови структурованої інформації на веб-сторінці для подальшого її використання.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Дослідження та аналіз методів інтелектуального аналізу даних

У цій роботі ми поговоримо про отримання інформації з веб-сайтів, веб-сторінок. Завдання, пов’язане з цим, можна віднести до концепції Data Mining. Data Mining (укр. глибинний аналіз даних) - збірне ім'я, яке використовується для позначення набору методів виявлення раніше невідомих, нетривіальних, практично корисних та доступних інтерпретацій знань, необхідних для прийняття рішень у різних областях людської діяльності. Методи майнінгу даних поділяються на статистичні (описовий аналіз, кореляційний та регресійний аналіз, факторний аналіз, аналіз дисперсії, компонентний аналіз, дискримінантний аналіз, аналіз часових рядів) та кібернетичні (штучні нейронні мережі, еволюційне програмування, генетичні алгоритми, асоціативна пам’ять, нечітка логіка, дерева рішень, експертні системи обробки).

Аналіз кластерів (кластеризація даних) – завдання розбиття заданої вибірки об'єктів (ситуацій) на підмножини, що називаються кластерами, так що кожен кластер складається з подібних об'єктів, а об'єкти різних кластерів значно відрізняються. Завдання кластеризації стосується статистичної обробки, а також широкого класу навчальних завдань без вчителя. Кластерний аналіз – це багатовимірна статистична процедура, яка збирає дані, що містять інформацію про вибірку об'єктів, а потім організовує об'єкти у відносно однорідні групи (кластери) (Q-кластеризація, або Q-техніка, сам кластерний аналіз) [9].

Кластерний аналіз – це спосіб групування багатовимірних об'єктів, заснований на представленні результатів окремих спостережень за точками відповідного геометричного простору з подальшим виділенням груп як «скупчення» цих точок (скупчення, таксони). Кластер англійською мовою означає «згусток», «гроно винограду», «грона зірок» тощо. Цей метод дослідження був розроблений в останні роки у зв'язку з можливістю

комп'ютерної обробки великих баз даних. Кластер – це група елементів, що характеризується загальною властивістю, головна мета кластерного аналізу – знайти групи подібних об’єктів у вибірці [8].

Кластерний аналіз передбачає відбір компактних, віддалених одна від одної груп об'єктів, пошуку «природного» поділу населення на ділянки кластерних об’єктів. Він використовується, коли вихідні дані подаються у вигляді матриць близькості або відстані між об'єктами або у вигляді точок у багатовимірному просторі [9]. Найбільш поширеними є дані другого типу, для яких кластерний аналіз орієнтований на виявлення деяких геометрично віддалених груп, в межах яких об'єкти знаходяться поблизу [9].

Діапазон застосувань кластерного аналізу дуже широкий: він використовується в археології, медицині, психології, хімії, біології, державному управлінні, філології, антропології, маркетингу, соціології та інших дисциплінах. Однак універсальність програми призвела до появи великої кількості несумісних термінів, методів та підходів, що перешкоджають однозначному використанню та послідовній інтерпретації кластерного аналізу.

Завдання, вирішені методом Data Mining:

а) Класифікація - присвоєння вхідного вектора (об'єкта, події, спостереження) одному з відомих раніше класів.

б) Кластеризація - це поділ набору вхідних векторів на групи (кластери) за ступенем "схожості" один з одним.

в) Скорочення опису - для візуалізації даних, спрощення обчислення та інтерпретації, стиснення обсягів інформації, що збирається та зберігається.

г) Асоціація - пошук дублікатів шаблонів. Наприклад, пошук «стійких стосунків у кошику».

д) Прогнозування - пошук майбутніх станів об'єкта на основі попередніх станів (історичні дані)

е) Аналіз відхилення - наприклад, виявлення нетипових мережевих дій може виявити зловмисне програмне забезпечення.

є) Візуалізація даних.

Кластерний аналіз має такі вимоги до даних:

а) показники не повинні співвідносити один одного;

б) показники повинні бути безрозмірними;

в) їх розподіл має бути близьким до нормального;

г) показники повинні відповідати вимозі «стійкості», що розуміється як відсутність впливу випадкових факторів на їх значення;

д) зразок повинен бути однорідним, не містити «залишків».

Якщо перед кластерним аналізом передує факторний аналіз, то вибірку не потрібно «відновлювати» – заявлені вимоги дотримуються автоматично самою процедурою моделювання факторів (є ще одна перевага – z-стандартизація без негативних наслідків для вибірки; якщо він проводиться безпосередньо для кластерного аналізу, це може призвести до зниження чіткості поділу груп). В іншому випадку вибір потрібно скорегувати.

Цілі кластеризації:

а) розуміння даних шляхом ідентифікації структури кластера. Поділ вибірки на групи подібних об'єктів дає можливість спростити подальшу обробку даних та прийняття рішень шляхом застосування різного методу аналізу до кожного кластеру (стратегія поділу та підкорення);

б) стиснення даних. Якщо початковий зразок надмірно великий, то ви можете зменшити його, залишивши одного з найбільш типових представників кожного кластеру;

в) виявлення новинок. Вибираються нетипові об'єкти, які не можна приєднати до жодного кластеру;

У першому випадку вони намагаються зробити кількість кластерів меншими. У другому випадку важливіше забезпечити високу схожість об’єктів всередині кожного кластеру, і може бути якомога більше кластерів. У третьому випадку найбільший інтерес представляють окремі об'єкти, які не входять до жодного кластеру.

У всіх цих випадках може застосовуватися ієрархічна кластеризація, коли великі кластери розбиваються на менші, які в свою чергу розбиваються ще меншими і т. д. Такі завдання називаються проблемами систематики.

Результатом таксономії є деревоподібна ієрархічна структура. Більше того, кожен об’єкт характеризується переліком усіх кластерів, до яких він належить, як правило, від великих до малих.

Кластеризація (навчання без вчителя) відрізняється від класифікації (навчання з викладачем) тим, що мітки вихідних об'єктів спочатку не встановлені, і навіть сам набір може бути не відомий.

Рішення проблеми кластеризації принципово неоднозначне, і для цього є кілька причин:

а) не існує однозначно найкращого критерію якості кластеризації. Відомо низку евристичних критеріїв, а також ряд алгоритмів, які не мають чітко визначеного критерію, але реалізують досить обґрунтовану кластеризацію «за конструкцією». Усі вони можуть дати різні результати;

б) кількість кластерів, як правило, не відома заздалегідь і встановлюється відповідно до якогось суб'єктивного критерію;

в) результат кластеризації істотно залежить від метрики, вибір якої, як правило, також суб'єктивний і визначається експертом.

Завдання кластерного аналізу (або навчання без вчителя) полягає в наступному. Існує навчальний зразок $X\ell = \{x1, ..., x\ell\} \in X$ і функція відстані між об'єктами $\rho(x, x')$. Необхідно розбити вибірку на роз'єднані підмножини, що називаються кластерами, так що кожен кластер складається з об'єктів, близьких за метрикою ρ, а об'єкти різних кластерів значно відрізняються. У цьому випадку мітка (номер) кластера уі присвоюється кожному об'єкту хі є $X\ell$. Алгоритм кластеризації – це функція $a: X \to Y$, яка пов'язує мітку кластера у є Y з будь-яким об'єктом х є X. Набір міток Y відомий в деяких випадках, але частіше завдання полягає у визначенні оптимальної кількості кластерів з точки зору того чи іншого критерію якості кластеризації [2].

Рішення проблеми кластерного аналізу – це розділ, який задовольняє деяку умову оптимальності. Цей критерій може представляти деякий функціонал, що виражає рівні бажаності різних розділів і груп. Ця функціональність часто називається об'єктивною функцією. Завдання кластерного аналізу – це проблема оптимізації, тобто пошук мінімуму цільової функції для заданого набору обмежень. Прикладом цільової функції є, зокрема, сума квадратів внутрішньо групових відхилень для всіх кластерів [2].

Можна виділити наступні основні етапи кластерного аналізу.

Формування системи змінних. Часто необхідно заздалегідь вибрати з початкового набору змінних найефективнішу підсистему (в зарубіжній літературі цей процес називається «вибір функції»). Крім того, у деяких завданнях доцільно трансформувати вихідні змінні таким чином, щоб сформувати нові, більш інформативні показники («вилучення функції»). Щоб уникнути «домінування» змінних з великою шкалою вимірювання, проводиться попередня нормалізація початкових змінних.

Визначення способу обчислення відстані між об'єктами або групами об'єктів. Цей метод повинен відображати специфіку застосованої проблеми. Наприклад, у випадку безперервних змінних може бути вказана евклідова відстань. Для усунення ефекту сильних лінійних кореляцій між змінними використовується відстань Махалонобіс. Для номінальних змінних можна використовувати відстань Хеммінга. Для груп об’єктів метод знаходження відстані також визначається, наприклад, за принципом «далекий сусід», «найближчий сусід» тощо. Принцип «віддаленого сусіда» виправданий, коли є апріорна інформація що таксони мають компактну сферичну форму. Принцип «близького сусіда» має сенс застосовувати, якщо відомо, що таксони можуть мати «витягнуту» форму або розташовані концентрично.

Групування об’єктів. На цьому кроці створення груп об’єктів. Поділ на групи може бути «важким» (формується розділ початкового набору об'єктів) і

може бути «нечітким» (розраховується ступінь належності кожного об'єкта до груп). Існує велика різноманітність алгоритмів групування.

Представлення результатів. Потрібний простий та інформативний опис отриманих кластерів. Часто для такого опису вибирається «типовий об'єкт» або визначається набір показників, усереднених по групі показників. Опис також використовується у вигляді набору таксонів. Під таксоном маємо на увазі піддомен простору змінних мінімального обсягу, що мають деяку задану форму і містять точки відповідної групи.

Визначення якості отриманого групування. Після завершення кластеризації необхідно переконатися, що сформовані групи справді відображають внутрішні закономірності, характерні для вирішуваної проблеми, сприяють досягненню цілей аналізу та допомагають виявити нові властивості досліджуваних об’єктів. Існують також більш формальні методи контролю якості, пов'язані з знаходженням ймовірності випадкового утворення груп, які можна обчислити в рамках певної моделі розподілу (з верифікацією статистичних гіпотез щодо однорідності спостережень різних класів); методом завантаження; з розрахунком різних показників якості.

Незважаючи на велику кількість досліджень у галузі кластерного аналізу, у цій галузі існує низка нагальних проблем. Перерахуємо основні проблеми:

а) проблема обґрунтування якості результатів аналізу. Відомо, що процес групування багато в чому суб’єктивний. Це виражається, зокрема, у тому, що один і той же набір об'єктів можна класифікувати по-різному залежно від області застосування, ступеня повноти знань про об’єкти дослідження тощо. Тому необхідно розробити методи, щоб повністю прийняти враховувати наявні експертні знання, а також розробляти відповідні критерії якості групування;

б) для багатьох напрямків дослідження, які важко формалізувати, характерна відсутність знань про досліджувані об'єкти, що ускладнює формулювання їх математичних моделей;

в) проблема аналізу великої кількості різнорідних (кількісних чи якісних) факторів. У випадку гетерогенного простору виникає методологічна проблема визначення метрики в ньому. З іншого боку, навіть у просторі однотипних (кількісних) змінних із збільшенням їх кількості посилюється «прокляття виміру», що може призвести до майже повної нерозрізненості балів. Отже, відстань від будь-якої точки до її «найближчого сусіда» для деяких типів відстаней може практично збігатися (з урахуванням точності машини) з відстані до її «далекого сусіда». Візуальні аналогії, що мають відношення до простору малого розміру, стають абсолютно неприйнятними у просторі великого розміру;

г) нелінійність відносин; наявність упущень, помилок вимірювань змінних. Класичні методи зменшення розмірності (метод основних компонентів; метод незалежних компонентів), що використовуються в кластерному аналізі, в основному зосереджені на лінійних зв’язках між змінними. Для виявлення більш складних зв’язків такі алгоритми, як нелінійні (ядерні) методи основних компонентів тощо;

д) необхідність подання результатів аналізу у формі, зрозумілій фахівцям прикладної галузі. Окрім хорошої здатності прогнозування будь-якого алгоритму аналізу даних, важливо, наскільки зрозумілі та інтерпретовані його результати. Для поліпшення інтерпретації рішень можна використовувати логічні моделі. Такі моделі використовуються для вирішення завдань розпізнавання образів та прогнозування кількісних показників;

е) проблема пошуку глобального екстремуму в критерії групування якості. Критерій якості - це функція, яка залежить від великої кількості факторів, нелінійних, з багатьма локальними екстремумами. Для пошуку кластерів необхідно вирішити складну комбінаторну задачу пошуку оптимального варіанту класифікації;

є) проблема стабільності групування рішень. У класичних алгоритмах для вирішення задач кластерного аналізу результати групування можуть сильно

відрізнятися залежно від вибору початкових умов, порядку об'єктів, параметрів алгоритмів тощо.

Web Mining - це процес методів Data Mining для автоматичного виявлення і вилучення інформації з веб-документів і сервісів. Основною метою Web Mining є пошук корисної інформації з всесвітньої павутини і її схем використання.

Інтернет зараз містить величезну кількість інформації, знань. Користувачі на різних умовах можуть переглядати різноманітні документи, аудіо- і відеофайли. Однак це різноманіття даних приховує в собі проблеми, які можуть виникнути не тільки при аналізі, а й при пошуку необхідної інформації в Інтернет.

1. Проблема пошуку потрібної інформації пов'язана з тим, що користувач не завжди відразу може знайти необхідні йому електронні ресурси. Лише невеликий відсоток посилань серед запропонованих пошуковими системами призводить до необхідних документів. Також важкий пошук неіндексованих інформації такими засобами.

2. Проблема виявлення нових знань. Навіть якщо знайдено безліч інформації, для користувача витяг корисних знань є досить трудомісткою і непростим завданням. Сюди ж можна і віднести складності, пов'язані з осмисленням відомостей, поняттям тих ідей, які були вкладені авторами.

3. Проблема вивчення споживачів пов'язана з наданням користувачеві інформації, яка виявилася б йому цікава. Це особливо актуально для електронних торговельних порталів, які могли б "підказувати" користувачеві при виборі товару.

Етапи Web Mining:

1. Вхідний етап (input stage) - отримання "сирих" даних з джерел (логи серверів, тексти електронних документів);

2. Етап попередньої обробки (preprocessing stage) - дані подаються у формі, необхідної для успішної побудови тієї чи іншої моделі;

3. Етап моделювання (pattern discovery stage);

4. Етап аналізу моделі (pattern analysis stage) - інтерпретація отриманих результатів.

Це загальні кроки, які необхідно пройти для аналізу даних мережі Інтернет. Конкретні процедури кожного етапу залежать від поставленого завдання. У зв'язку з цим виділяють різні категорії Web Mining. Web Mining можна розділити на три різних типи категорій: Web Content Mining, Web Structure Mining і Mining Usage Mining, а також їх загальний взаємозв’язок (рисунок 1.1).

Рисунок 1.1 - загальний взаємозв’язок між категоріями WebMining

1. Web Content Mining. Аналіз веб-контенту - це процесс для вилучення корисної інформації з вмісту веб-документів. Веб-контент складається з декількох типів даних - текстових, графічних, аудіо, відео і т.д. Контентні дані - це група фактів, розроблених веб-сторінкою. Він може надати ефективні та цікаві шаблони про потреби користувачів. Текстові документи пов'язані з видобутком тексту, машинним навчанням і обробкою природної мови. Цей Майнінг також відомий як Text Mining. Цей тип інтелектуального аналізу виконує сканування і аналіз тексту, зображень та груп веб-сторінок у відповідності до змісту введення.

2. Web Structure Mining. Веб-аналіз структури являє собою поцесс для виявлення структурної інформації з Інтернету. Структура веб-графа складається з веб-сторінок як вузлів і гіперпосилань, як ребер, що з'єднують пов'язані сторінки. Аналіз структури в основному показує структуроване резюме певного веб-сайту. Він ідентифікує відносини між веб-сторінками, пов'язаними інформацією або прямим зв'язком. Для визначення зв'язку між двома комерційними веб-сайтами може бути дуже корисний аналіз веб-структури.

Залежно від поставленого завдання структура сайту моделюється за певним рівнем деталізації. У найпростішому випадку гіперпосилання представляють у вигляді спрямованого графа:

$$G = (D, L) \tag{1.1}$$

D - це набір сторінок, вузлів або документів; L - набір посилань.

Витяг веб-структур може бути використано як підготовчий етап для вилучення веб-контенту.

3. Web Content Mining. Пошук знань в мережі Інтернет є непростим і трудомістким завданням. Саме цей напрям Web Mining вирішує її. Воно засноване на поєднанні можливостей інформаційного пошуку, машинного навчання та Data Mining. Аналізується зміст документів, знаходяться схожі за змістом слова та їх кількість.

Потім вирішується завдання кластеризації та класифікації. Так документи групуються за смисловою близькості.

Text Mining - це процес отримання ідей з тексту. Ця інформація зазвичай виходить шляхом визначення шаблонів і тенденцій в тексті за допомогою таких методів, як статистичне вивчення шаблонів. Зазвичай це включає в себе процес

структурування вхідного тексту, вилучення шаблону з структурованих даних і, нарешті, оцінки та інтерпретації вихідних даних.

Мета інтелектуального аналізу тексту - по суті перетворити текст в дані для аналізу з застосуванням обробки природної мови (NLP) та аналітичних методів. Для цього інтелектуальний аналіз тексту включає пошук інформації і даних, лексичний аналіз для вивчення розподілу частот слів, розпізнавання образів, тегування та анотування, вилучення інформації, методи інтелектуального аналізу даних, візуалізацію і прогнозну аналітику.

Text Mining є одним з найбільш важливих способів аналізу і обробки неструктурованих даних, який становить майже 80% світових даних. Сьогодні більшість організацій і установ збирають і зберігають величезні обсяги даних в сховищах даних і хмарних платформах, і ці дані продовжують експоненціально зростати з кожною хвилиною у міру надходження нових даних з декількох джерел. В результаті для компаній і організацій стає складним завданням зберігати, обробляти і аналізувати величезні обсяги текстових даних за допомогою традиційних інструментів.

Основні завдання в text mining:

Всього за кілька кроків системи інтелектуального аналізу витягають з "корпусу" ключові смисли, визначають, чи придатний текст для вирішення поставленого завдання, і виявляють деталі його змісту. В даному випадку під "корпусом" мається на увазі набір текстів, які відповідають попередньо заданих параметрах: спочатку формуються критерії, а потім підбираються відповідні їм тексти.

а) Релевантність документа (пошук текстів по заданій темі). Тема може бути вузькою: наприклад, наукові статті по хірургії ока.

б) Іменовані суті. Якщо документ релевантний, може знадобитися відшукати в ньому деякі факти: наприклад, прізвища вчених або назви патологій.

в) Тип документу. Необхідно привласнити документу мітку в залежності від його змісту: наприклад, класифікувати огляди на товар як "позитивні" або "негативні".

г) Зв'язки між сутностями. Крім самих фактів, часто необхідно знайти ті частини документів, де йдеться про взаємозв'язок фактів: наприклад, пошук зв'язків між медичними препаратами і побічними ефектами або пошук зв'язків між ім'ям співробітника і негативними відгуками на його роботу.

До завдань Text Mining відносяться завдання класифікації документів, кластеризації, пошуку і вилучення фактів.

1. Завдання класифікації полягає у віднесенні документа до однієї з декількох заздалегідь визначених категорій на підставі аналізу його тексту. Навчання класифікатора проводиться на основі вибірки з документів, що відносяться до певних класів. Для побудови правил розміщення документів в певних категоріях використовується кореляційний аналіз. Завдання класифікації є найбільш поширеною в Text Mining. Прикладами застосування класифікації документів в бізнесі є завдання розміщення документів по певних папок, сортування повідомлень електронної пошти, поширення тематичних новин передплатникам, угруповання документів у інтранет-мережах і на веб-сайтах.

2. Кластеризація в Text Mining полягає у виявленні близьких за своїми властивостями груп об'єктів (документів). Для проведення кластеризації використовуються всі досліджувані документи. Кластеризація застосовується для виявлення класів у великих масивах документів, визначення взаємопов'язаних груп документів, знаходження унікальних документів, що не належать ні до одного кластеру, для виявлення однакових або дуже схожих за змістом документів.

3. Завдання пошуку заснована на аналізі зв'язків, які визначають появу ключових фраз в документі. Мета пошуку пов'язаних понять в окремих

документах полягає в тому, щоб знайти невідомі заздалегідь зв'язку між окремими ознаками документа.

4. Витяг фактів (fact (information) extraction). Завдання вилучення фактів полягає в отриманні з неструктурованого тексту інформації певного виду з метою поліпшення, наприклад, класифікації, пошуку і кластеризації.

П'ять основних кроків, пов'язаних з аналізом тексту:

1. Збір неструктурованих даних з декількох джерел даних, таких як простий текст, веб-сторінки, PDF-файли, електронні листи і блоги, і багато інших.

2. Виявлення і видалення аномалій з даних шляхом проведення операцій попередньої обробки і очищення. Очищення даних дозволяє витягувати і зберігати цінну інформацію, приховану в даних, і допомагає визначити коріння конкретних слів.

3. Для цього ви отримуєте ряд інструментів для інтелектуального аналізу тексту та додатків для аналізу тексту.

4. Перетворіть всю необхідну інформацію, витягнуту з неструктурованих даних, в структуровані формати.

5. Аналізувати шаблони в даних за допомогою інформаційної системи управління (MIS).

Information Extraction (витяг інформації):

а) Feature (Entity) Extraction - витяг слів або груп слів, які, з точки зору користувача, важливі для опису змісту документа. Це можуть бути згадки персон, організацій, географічних місць, термінів предметної області та інших слів або словосполучень. Видобувні суті також можуть бути найбільш значущими словосполученнями, що характеризують документ по його основної теми;

б) Feature (Entity) Association Extraction - більш складні з технологічної точки зору. Простежуються різного роду зв'язки між витягнутими сутностями. Наприклад, навіть якщо обрані суб'єкти згадані в різних документах, але мають

якусь загальну характеристику (час, місце і т. Д.), Можна з великим ступенем визначеності сказати, чи є між ними якийсь зв'язок чи ні;

в) Relationship, Event and Fact Extraction - найскладніший варіант вилучення інформації (Information Extraction), що включає в себе витяг сутностей, розпізнавання фактів і подій, а також вилучення інформації з цих фактів. Пошукова система тут безпорадна, так як звичайна людська мова має на увазі дуже багато варіантів викладу. Користуючись лише пошукачем, ми повинні були б ідентифікувати цей факт по всім ключовим словами, які його характеризують. А технологія Text Mining робить це сама, причому відповідно до заданих обмеженнями відрізняє відповідні факти від тих, що ніяк з ними не пов'язані.

Summarization (автоматичне реферування, анотування) - побудова короткого змісту документа по його повного тексту.

Categorization (категоризація, класифікація) - віднесення документа або його частини до однієї або декількох категоріях. Категорії можуть визначати "спрямованість" тексту - тематичну, жанрову, емоційну, оцінну.

Clusterization - об'єднання документів в групи за принципом їх схожості.

Проблеми такого підходу очевидні і пов'язані з многокомпонентностью рішення. Потрібно інсталювати пошуковик, інструмент отримання даних з тексту, засоби аналізу, а крім того, зробити всю супутню інтеграцію. Проте видається, що саме цим шляхом рухатимуться постачальники рішень для кінцевих користувачів. Підстав для цього декілька.

1. Інструменти аналізу, зокрема BI і Data Mining, у всьому світі стають стандартом де-факто, і все більше фахівців спирається на них як на основні засоби створення аналітичного середовища. Поряд з комерційними продуктами такого роду розвивається світ відкритих ресурсів (проекти Pentaho і Eclipse), доступних широкій аудиторії користувачів.

2. Технології Text Mining, включаючи кошти інтеграції з джерелами інформації і аналітичними інструментами, також комерціалізуються (їх пропонують такі фірми, як Clarabridge, Nstein Technologies, Attensity).

3. Розвиваються й самі наукові області - комп'ютерна лінгвістика, методи аналізу текстів. З'явилися консультанти, в основну сферу діяльності яких входить вирішення подібних завдань. Залучення цих експертів робить проекти такого роду виключно ефективними.

Відмінність технології Text Mining від Data Mining полягає в тому, що остання працює з базами даних, в той час як Text Mining дозволяє досліднику аналізувати звичайні тексти, представлені на природній мові. На практиці використання технологій глибинного аналізу текстів відкриває для такі можливості:

- моніторинг ресурсів Інтернет (контент-моніторинг), семантичний пошук інформації в Інтернет і суттєве звуження меж пошуку за рахунок включення методів Text Mining в сучасні пошукові системи;

- створення семантичної мережі великих текстів, реферування, класифікація, кластеризація текстів, пошук по тексту, інтегрування неструктурованою текстовій інформації з існуючими структурованими даними, наочна візуалізація кластерезірованной текстової інформації.

Таким чином, використання технологій глибинного аналізу текстів дозволить студентам орієнтуватися у великих інформаційних потоках, здійснювати більш плідну діяльність пов'язану з пошуком, аналізом, синтезом електронної текстовою інформацією, оцінювати її корисність з меншими тимчасовими і енергетичними затратами, формувати систему інформаційних понять.

Цей напрямок може бути використано для оптимізації пошуку індексованих документів. Інтернет містить велику кількість інформації, яка використовується різними способами. Існує ряд завдань, коли певна програма чи інший веб-ресурс потребує отримання інформації з різних веб-джерел.

Раніше багато з цих завдань вимагали роботи з кодом веб-сторінок. На даний момент зручним рішенням для цього є веб-сервіси, за допомогою яких можна зручно отримувати інформацію з потрібного ресурсу.

Однак є ще завдання, над якими потрібно працювати безпосередньо з веб-сторінками, а також з його кодом. Іноді потрібно отримати інформацію з сайту, який не має веб-служб, або вони не надають необхідної інформації. Наприклад, це можуть бути якісь інтернет-магазини тощо. Такі завдання можна розділити на 2 типи - витяг інформації з певної веб-сторінки з відомою структурою та вилучення подібної інформації з різних джерел. У першому випадку вони зазвичай використовують HTML-аналізатор або регулярні вирази. У другому випадку можуть застосовуватися різні підходи, пов'язані з машинним навчанням. Завдання такого роду тривіальні, хоча б тому, що зазвичай виконується робота над відомим змістом.

Ви також можете виділити завдання, пов’язані з пошуку інформації серед веб-ресурсів. Веб-сторінка зазвичай містить велику кількість інформації, серед якої багато корисного та непотрібного для користувача (наприклад, реклами), а також з точки зору пошуку (рисунок 1.2). Наприклад, дивлячись на сторінку блогу для статті, текст, який знаходиться безпосередньо у самій статті, важливіший, ніж решта іншого тексту на сторінці (меню, реклама та будь-яка інша інформація, яка є на сторінці). Крім того, HTML не надає інформацію про класифікацію змісту документа (на прикладі блогу - що саме таке назва, текст статті, коментарі тощо).

Рисунок 1.2 – Сторінка The New York Times

Хоча очищення веб-сторінки є важливим завданням, мало роботи було пророблено в цій області. Було запропоновано метод виявляти інформаційні блоки на новинних веб-сторінках. Концепція інформаційних блоків аналогічні нашої концепції основного змісту сторінки. Однак робота в [14] обмежена наступними двома припущення: (1) система знає, як веб-сторінка може бути розділена на узгоджені блоки контенту і (2) систему апріорі знає, які блоки є однаковими блоками в різних мережах сторінки. Як ми побачимо, розбиття веб-сторінки і виявлення відповідних блоків на різних сторінках насправді дві критичні проблеми в очищенні веб-сторінок. Наша система здатна виконати ці завдання автоматично (без допомоги користувача). Крім того, їх робочі погляди веб-сторінка у вигляді плоского набору блоків, які відповідають «TABLE» елементам на веб-сторінках, і кожен блок розглядається як збірник слів. Ці припущення часто вірні в новинах, веб-сторінки, які є доменом їх додатків. Переважно, ці припущення дуже сильні.

В [2] очищення веб-сторінки визначається як частий шаблон проблема виявлення. Вони пропонують частотний аналіз даних алгоритм для виявлення шаблонів і перегляду цих шаблонів як шуми.

Метод очищення в [9] не пов'язаний з контекстом веб-сайт, який може дати корисні поради для очищення сторінки. Крім того, в розбиття веб-сторінки попередньо фіксується, враховуючи кількість гіперпосилань, які має елемент HTML. Цей метод поділу простий і корисний для набору веб-сторінкок з різних веб-сайтів, веб-сайт як правило, має свої загальні макети або стилі, які можуть бути використані для поділу веб-сторінок і виявлення шуми. Ми порівняємо результати нашого методу з результатами метод в [9].

Інша робота включає в себе очищення даних для інтелектуального аналізу даних і складування даних [13], виявлення дублікатів записів в текстовому вигляді бази даних [16] і попередня обробка даних для Web Usage Mining [7].

Наше завдання інше, оскільки ми маємо справу з напів структурованими веб-сторінками, а також ми зосереджені на видаленні галасливих частин сторінки, а не дублікати сторінок. Отже, необхідні різні методи очищення.

Очищення веб-сторінки також пов'язана з вибором функції в традиційне машинне навчання. У виборі функції, особливості - це окремі слова або атрибути. Проте, предмети в веб-сторінкі мають деякі структури, які відображаються їх вкладені теги HTML. Отже, в контекст мережі пропонують деякі механізми навчання для розпізнавання банера реклама, надлишкові і нерелевантні посилання веб-сторінок. Проте, ці методи не є автоматичними. Вони вимагають великого набору вручну позначені дані навчання, а також предметні знання для генерації правила класифікації, покращує алгоритм HITS [12], використовуючи ентропію для оцінки важливості посилань. Основна увага приділяється поліпшення алгоритму HITS для пошуку найбільш інформативних структур в веб-сайтів. Хоча він розділяє веб-сторінки на блоки контенту уникати непотрібних авторитетів і хабів, це не виявлення або усунення галасливого вмісту на веб-сторінках.

Пошук є більш ефективним, якщо документ структурований. З цієї точки зору, було б зручно отримувати структуровану інформацію з додаткової

інформації про частину документа, наприклад, для класифікації інформації або мати певні ваги (пріоритети) для елементів відповідного документа. У цій роботі ми розглянемо та застосуємо деякі підходи до отримання таких даних із коду сторінки.

1.2 Аналіз існуючих аналогів

Розглядаючи завдання вилучення основної інформації з веб-сторінки, також можна навести приклад, не пов’язаний із пошуком. Популярний приклад - популярний сайт Instapaper [4].

Instapaper - послуга призначена для читання веб-сторінок в режимі офлайн. Однак ця послуга популярна через те, що вона не просто зберігає веб-сторінку. Принцип роботи полягає в тому, що Instapaper аналізує документ і вилучає з нього саме основну інформацію, яку він класифікує, і як результат, цю інформацію можна розглядати так, як вам подобається, наприклад, використовуючи різноманітне форматування.

Raindrop - це сервіс для зберігання та організації закладок, тому він не має перегляду в автономному режимі та спрощеного форматування. Послуга доступна на всіх популярних платформах, має широкий спектр варіантів сортування та впорядкування закладок, включаючи інтелектуальне позначення тегів, а також дозволяє точно налаштувати показ вмісту.

Candy – сервіс, що дозволяє вам зберегти не лише цілі статті, але й окремі уривки з них. Підтримує форматування тексту, здатне правильно відображати медіафайли на сторінках.

Інтерфейс реалізований дуже зручно: у вигляді бічної панелі, яка відкривається натисканням плаваючої кнопки. Особливих функцій для організації вмісту в Candy немає, записи йдуть одним потоком, але ви можете закріпити їх і переглянути.

1.3 Аналіз існуючих алгоритмів

Розглянемо декілька існуючих алгоритмів, які гарантують високу точність. Такі алгоритми включають BTE, Victor, CETR, boilerpipe, VIPS, Readability, LQF, FE.

Багато цих алгоритмів використовують схожі підходи та евристики.

Метод сегментування сторінок (VIPS) на основі бачення повинен розділити веб-сторінку у дерево, де вузли візуально згруповані в блоки.

На основі методу VIPS, був представлений підхід до ранжування значення блоку для веб-сторінок через алгоритми навчання, використовуючи просторові функції (наприклад, положення або розмір) та вмісту (наприклад, кількість зображень та посилання). Значення блоку веб-сторінки обчислювалось за допомогою присвоєння ваг класам структурно подібних блоків. Однак VIPS повиннен частково відобразить сторінку для того, щоб проаналізувати її. Крім того, якщо використовуються зовнішні таблиці стилів, їх також слід отримати. Тому порівняно з іншими методи, VIPS є ресурсомістким.

Алгоритм BTE використовує прості евристики для вилучення контенту без шаблонних елементів, розглядаючи сторінку як послідовніть токенів, а потім витягується вміст шляхом ідентифікації єдиної безперервної області, яка містить найбільше слів і найменше тегів HTML. Щоб подолати обмеження BTE у виявленні лише одного суцільного блоку тексту, розширили цей метод на побудову кривих нахилів документа, в яких для пошуку регіонів документа в якому слові токенів більше, ніж токенів тегів.

FeatureExtractor (FE) та KFeatureExtractor (KFE), засновані на блоковій сегментації HTML-документа. Кожен блок аналізується на особливості, такі як кількість тексту, наявність зображень та коду. Текст вмісту витягується шляхом вибору блоків, які найкраще відповідають бажаній особливість, наприклад наявність більшості тексту.

Алгоритм Victor використовує машинне навчання для створення класифікатора, за допомогою якого він ідентифікує різні частини сторінки.

Алгоритм boilerpipe використовує різні евристики разом з особливостю тексту на сторінці.

В багатьох цих алгоритмах використовуються ефективні підходи, які можно комбінувати з іншими підходами для створення нових алгоритмів, які можуть мати деякі переваги.

Link Quota Filter (LQF) алгоритм для ідентифікації списків посилань та елементів навігації шляхом ідентифікації елементів DOM, які мають високе співвідношення тексту, що знаходиться у гіперпосиланнях. Його можна застосувати до вилучення вмісту, видаливши отримане посилання з блоків документа. Недолік цього методу є що вона опирається на елементи структури, і вона може лише ідентифікувати шум типу гіперпосилання.

Content Extraction Tag Ratios (CETR), метод вилучення вмісту текст з різних веб-сторінок за допомогою HTML-документа, при обчисленні коефіцієнів тегів. Підхід обчислює коефіцієнти тегів на основі прямої лінії, а потім кластеризує отриману гістограму в області вмісту та шуму. Це лаконічний та ефективний алгоритм, але вразливий до зміни стилю вихідного коду сторінки.

2 АНАЛІЗ ПІДХОДІВ ТА ПОСТАНОВКА ЗАДАЧІ

2.2 Аналіз існуючих підходів

Завдання аналізу структури автоматично генерованих html-сторінок в останні роки часто привертає увагу дослідників. Виділення структурної інформації використовується для різних завдань обробки документів [6], пошуку інформації [6] та для відстеження змін та кешування веб-сторінок [10, 11].

У роботі [7] було показано, що сучасна практика дизайну веб-сайтів із використанням засобів динамічного генерування веб-сторінок порушує основні припущення, на яких будуються сучасні веб-пошукові системи. А саме такі припущення є [7]:

1) припущення про відповідні посилання (Relevant Linkage

Principle) - гіпертекстові посилання, встановлені авторами сторінки, вказують на авторитетні ресурси по темі документа;

2) ресурси, що пов'язують один одного, є тематично близькими (Topical Unity Principle);

3) текст поруч із гіперпосиланням має відношення до теми ресурсу, на який посилається посилання (Lexical Affinity Principle). На відміну від [7], ми розглядаємо передусім тематичний зміст веб-сторінок та класичний пошук документів без урахування структури посилань в Інтернеті. У цьому випадку стає можливим оцінити якість пошуку інформації за допомогою вільно доступних даних.

Методи, що використовуються для аналізу структури веб-сторінок, можна поділити на:

1) методи, засновані на аналізі dom-дерева окремої сторінки html [8, 9];

2) методи, засновані на підборі повторюваних фрагментів сторінок з одного сайту [6, 7, 9, 10, 11].

Стандартний спосіб створення html-сторінок - додавання частини навігації по краях сторінки (частина вмісту знаходиться посередині). Сторінка

видається у вигляді таблиці або вкладених html-таблиць, тоді як окремі елементи сторінки розміщуються в різних клітинках.

Методи, засновані на аналізі роботи dom-дерева та на основі аналізу структури html-сторінки та вибору змісту сторінки на основі евристичних припущень про "стандартні" способи обробки документів. Такі методи зазвичай поєднуються з методами, заснованими на підборі повторюваних фрагментів сторінок одного сайту [9].

Кожна HTML-сторінка відповідає дереву DOM, в якому знаходяться теги. Внутрішні вузли і докладні тексти, зображення або гіперпосилання є листові вузли. Сегмент HTML-кодів і його відповідне дерево DOM (рисунок 2.1). У дереві DOM кожен суцільний прямокутник вузол тега. Затінене поле - це фактичний вміст вузла, наприклад, для тега IMG фактичним змістом є «src = image.gif». сповіщення, що наше вивчення веб-сторінок HTML починається з тега BODY так як всі видимі частини знаходяться в рамках BODY. Кожен вузол також пов'язаний зі своїми властивостями відображення. Для зручності аналізу, ми додаємо віртуальний кореневий вузол без будь-якого атрибуту як вузол батьківського тега BODY в дереві DOM[9].

Хоча дерева DOM досить для подання макета або стиль подання однієї HTML-сторінки, важко вивчити загальний стиль подання і зміст набору HTML-сторінок і очистити їх на основі окремих дерев DOM.

Таким чином, DOM-дерева недостатньо в нашій роботі з вилученням інформації, яка враховує як стиль презентації і реальний зміст веб-сторінок. Нам потрібен більш потужна структура для цієї мети. Ця структура має вирішальне значення тому що наш алгоритм потребує цього, щоб знайти спільні стилі сторінок з сайту, щоб усунути шуми. Вводимо нове дерево структура, звана деревом стилів (ST), яка може стискати загальні стилі уявлення набору пов'язаних веб-сторінок[10].

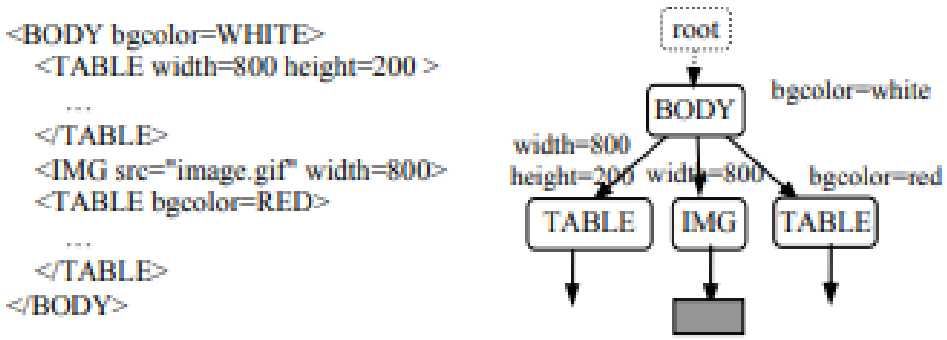

Рисунок 2.1 – Приклади дерева DOM

Приклад дерева стилів наведений на рисунку 2.2 у вигляді комбінації.

Дерева DOM d1 і d2. Ми помічаємо, що, крім чотирьох тегів (P, IMG, P і A) на нижньому рівні, усі теги в d1 мають свої відповідні теги в d2. Таким чином, d1 і d2 можна стискати. Ми використовуємо підрахунок, щоб вказати, скільки сторінок мають певний стиль певний рівень дерева стилів. На рисунку 2.2 ми можемо побачити, що обидві сторінки починаються з BODY, і таким чином BODY має кількість 2. Внизу BODY обидві сторінки також мають однаковий стиль викладу TABLE-IMG-TABLE. Ми називаємо всю цю послідовність тегів (TABLE-IMG-TABLE) вузол стилю, який укладений у прямокутник, вишитий штрихом на рисунку 2.2. Він представляє конкретну презентацію стиль у цій точці[10].

Таким чином, вузол стилю - це послідовність вузлів тегів у дерево DOM. У дереві стилів ці вузли тегів ми називаємо елемент вузлів, щоб відрізнити їх від вузлів тегів у дереві DOM. Наприклад, вузол стилю TABLE-IMG-TABLE має три вузли елементів, TABLE, IMG та TABLE[10].

Елементний вузол також містить дещо іншу інформацію від вузла тегів у DOM дерево.

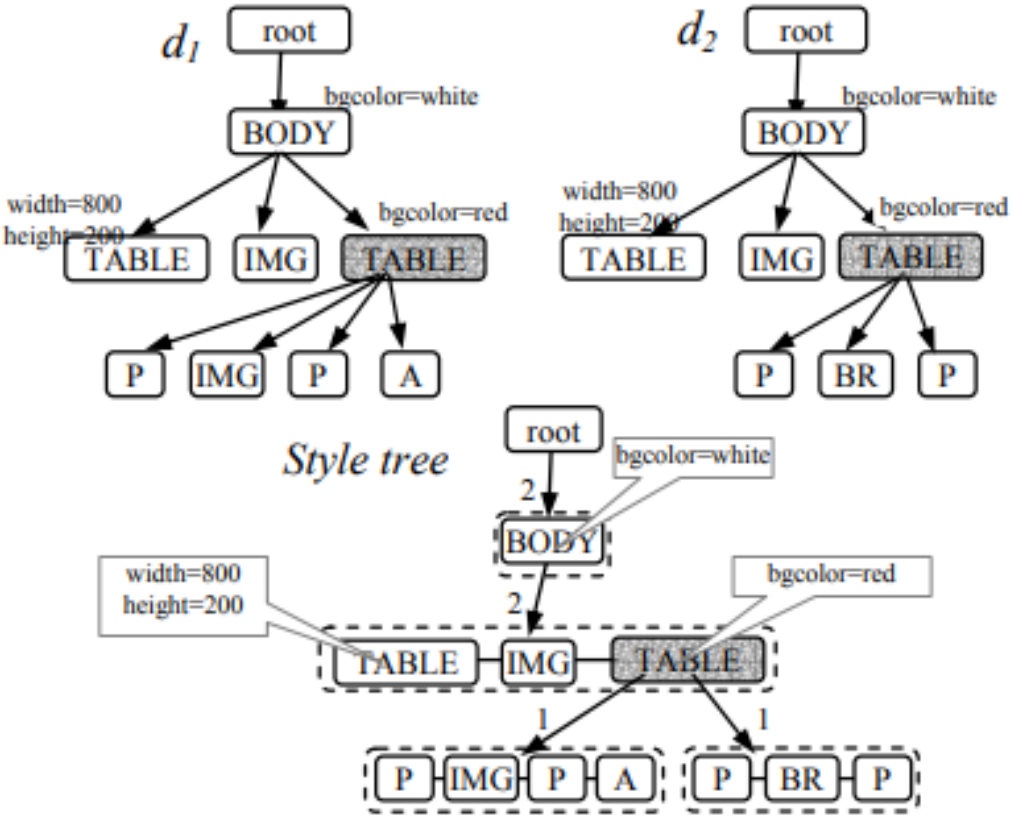

Рисунок – 2.2 Дерево DOM і дерево стилів

На малюнку 2.2 ми бачимо, що під самим правим тегом TABLE, d1 і d2 розходяться, що відбивається двома різними вузлами стилю в дерево стилів. Два стильових вузла: P-IMG-P-A і P-BR-P відповідно. Це означає, що під правим вузлом TABLE ми маємо два різні стилі презентації. Кількість сторінок цих двох вузла стилю рівні 1. Очевидно, дерево стилів є стислим уявлення двох дерев DOM. Це дозволяє нам побачити, які частини DOM-дерев є загальними, і які частини відрізняються[11].

Звернімо увагу, що вузол елемента відповідає вузлу тега в DOM дерево, але вказує на набір дочірніх вузлів стилю SS (рисунку 2.2). За для зручності ми зазвичай позначаємо вузол елемента його ім'ям тега, і вузол стилю по послідовності імен тегів, відповідних його послідовність вузлів елемента[11].

Створення дерева стилів (званого деревом стилів сайту або SST) для сторінок веб-сайт досить простий. Спочатку ми будуємо дерево DOM для

кожна сторінка, а потім об'єднати його в дерево стилів в низхідному стилі. У конкретному елементі вузла E в дереві стилів, який має відповідний вузол тега T в дереві DOM, ми перевіряємо чи є послідовність дочірніх вузлів тега T в дереві DOM така ж, як послідовність вузлів елементів в вузлі стилю S нижче E (в дереві стилів). Якщо відповідь так, ми просто збільшуємо кількість сторінок вузла стилю S, а потім перейти вниз по дереву стилів і дерево DOM для об'єднання інших вузлів. Якщо відповідьі немає, новий вузол стилю створюється під вузлом елемента E в дерево стилю, під дерево вузла тега T в дереві DOM копіюється в дерево стилів після перетворення в вузли стилю і елемент вузли дерева стилів[11].

2.1.1 Кроки вилучення вмісту

Крок 1 Стандартизація тегів веб-сторінки [10].

а. Символи «<» and «>» повинні містити тільки HTML-теги. При використанні в іншому місці їх слід замінити на «&lt» і «&gt» відповідно.

б. Всі теги повинні збігатися, тобто кожен початковий тег має відповідний кінцевий тег.

с. Атрибути всіх тегів повинні бути укладені в лапки мітки.

д. Всі теги повинні бути правильно вкладені. Наприклад, <a>. ,

. <b>... </ b>... - правильне, в той час як <a>... <b>... … </ b> невірно.

Крок 2 Попередня обробка тегів веб-сторінки.

Всі теги на сторінці утворюють деревоподібну структуру. Ті вузли, які не містять текст повинен бути вилучений, а також неприпустимі теги, такі як <script> <style> <form> <marquee> <meta> і т. д., які не пов'язані зі змістом. Тоді структура дерева побудована.

Крок 3 Оцінка місця розташування контенту

Метою цього процесу є вибір оптимального вузла що містить контент. Якщо вузол не задоволений цим умова, текст під цим вузлом не ідентифіковано. Як веб-сторінка новин є деревоподібну структуру, зміст повинен бути загальний вузол.

Крок 4 Витяг вмісту

Вміст витягується за допомогою таких інструментів, як аналізатор HTML. Якщо вузол не задоволений умовами, поверніть до кроку 3. Щоб знайти оптимальні вузли вузлів наступного рівня (дочірні вузли).

Крок 5 Коригування результатів вилучення з кроку 4

На кроці 3 тільки вузол, який, швидше за все, містить контент вибраний. Але якщо структура веб-сторінки децентралізована, вона дуже схильна до вилучення розділу або абзац всього змісту. Як сусідні вузли на той же рівень вільні від судді, на цьому етапі ми повинні налаштувати вищевказаний результат. Текст також повинен бути витягнутий з суміжних вузлів, що відповідають точним умов витягу контенту. Таким чином, весь текст буде вилучено з кваліфіковані вузли на одному рівні.

Всесвітня павутина та HTML принесли новий погляд на пошук інформації. На відміну від звичайних повідомлень, HTML дозволяє визначити візуальне подання вмісту. Дані HTML-документа подаються у структурованій формі. З цієї причини HTML-документи часто розглядаються як погано структуровані інформаційні ресурси. Користувач не змушений читати весь документ. Навпаки, згідно з багатьма дослідженнями, читачі дивляться на документ і шукають якісь цікаві частини, а не читають увесь зміст. Через таку поведінку користувачів Інтернету написання тексту для всесвітньої павутини стало окремим заняттям, для якого часто використовується термін веб-копірайтинг. Однією з головних вимог до тексту в Інтернеті є те, що організація сторінки повинна бути простою і зрозумілою, а користувач може легко знайти ту частину документа, яка містить необхідну інформацію. Для цього документи містять систему елементів, що мають переважно візуальний характер. За роки

Всесвітньої павутини ці методи представлення даних довели до вдосконалення, так що читання документів за допомогою веб-браузера стає зручнішим. З точки зору автоматизованої обробки документів, ситуація інша. HTML не містить інструментів для семантичного опису вмісту документа. Прийоми обробки природних мов, які використовувались для отримання інформації з простого тексту, не застосовуються, оскільки HTML-документи зазвичай не містять багатьох цілих пропозицій або блоків суцільного тексту. Також половина тексту на сторінці може взагалі не стосуватися інформації, що вважається або необхідною для аналізу. З іншого боку, теги HTML, вставлені в текст документа, надають додаткову інформацію, яку можна використовувати для ідентифікації даних. На рис. 2.1 показаний приклад простого документа та відповідної частини його HTML-коду.

Capital Cities

France – *Paris*
Japan – *Tokyo*

```
...
<i>Capital Cities</i>
<b>France</b> - <i>Paris</i>
<b>Japan</b> - <i>Tokyo</i>
...
```

Рисунок 2.1 – Зовнішній вигляд і код блоку HTML-документа

Уявімо, що ми хочемо витягнути з цього документа назви країн та їх столиць. Переглядаючи HTML-код, ми можемо побачити, що назва кожної країни оточена тегами і і, відповідно, назва кожної столиці оточена і .

Таким чином, для отримання необхідної інформації з цього документа може бути створена проста процедура, яка зчитує код документа, ідентифікує ці теги та зберігає текст між кожною парою тегів. Ця процедура називається обгорткою. Цей приклад досить тривіальний. Для складніших документів слід розробити більш складні обгортки. Крім того, на сторінці може бути багато

зайвого тексту, а також велика кількість інших елементів, які не пов’язані з необхідною інформацією. Крім того, обгортки підходять лише для документів певного типу. Наприклад, веб-сторінка з інформацією про товар. Щоб обгортка працювала в різних магазинах з різними кодами, спочатку потрібно застосувати певні алгоритми машинного навчання.

Методи машинного навчання пропонують зручний спосіб поєднання різних показників «повноти», автоматично зважування створених вручну атрибутів відповідно до їх відносної важливості.

Аналогічно, відомі підходи можуть бути використані для виведення інформації з використанням скрытих марківських моделей, принципів максимальної ентропії та т. д.

Другого підходу - евристичний аналіз коду HTML. Цей метод заснований на конкретній емпіричній еволюції, пов'язаній з мовою HTML або деякими нерозподіленими можливостями його використання. Деякі важливі слова, які вносять важливі відомості в документ (наприклад, географія, транспорт і т. д.) - так називаються токени. Маркери визначають на основі своїх текстів та оточуючих тегів HTML. Наприклад, текст між тегом <h> - заголовник і т. д. Кожний символ відмічає початок розділу документа. Далі, ієрархічна структура розділяє, будує порівняння розмірів шрифтів і відсутній текст і т. д. Друга пропозиція запропонує, що ви можете знайти один роздільник для запису даних у документі. Документ змінено у вигляді дерева та на основі різної евристики виявляється об'ємною структурою та використовується в якості розрізних частин документа. Крім того, для аналізу використовується відома інформація про знання деяких тегів HTML, включаючи теги HTML5, такі:

1. section - елемент групує тематичні блоки. Елементи розділу можуть бути вкладені;
2. header - містить заголовок розділу, таблиці тощо;
3. footer - колонтитул веб-сторінки, зазвичай цей блок містить інформацію про сайт, контакти, авторські права;

4. nav - навігаційний блок, список посилань, пов'язаних тем;
5. article - основний зміст;
6. aside - не основний блок, зазвичай розташований з боків сторінки.

Ви також можете аналізувати імена класів у тегах. Прикладом таких підходів є алгоритм MDR [4], який описує, як ідентифікує область у документах для пошуку інформації. У цій роботі буде об'єднано різні евристичні підходи разом із підходами до візуального аналізу документу.

2.2 Основні етапи аналізу

Основні етапи аналізу веб-сторінки можна розділити на два етапи:

- сегментація. На етапі сегментації HTML-сторінку поділяють на семантичні блоки. Ця процедура дозволяє згодом виключити з розгляду блоки, що містять елементи навігації, рекламний та інший інформаційний шум. На основі обчислених просторових характеристик блоку та аналізу його вмісту робиться припущення, чи блок є інформаційним чи «шумовим». Текст, що міститься в блоках, піддається графематичному та морфологічному аналізу.

- класифікація. Крок класифікації визначає для кожного блоку, чи це основний контент, чи це елемент шаблону.

Підходи, що застосовуються в існуючих алгоритмах, також можна розділити на такі:

- машинне навчання, евристика;
- моделювання візуального відображення;
- використання функцій тексту;
- N-грам, класифікатори та інше.

Таким чином, у цій роботі ми розглянемо та запропонуємо підходи до вирішення проблеми виявлення основної інформації. Будуть розглянуті існуючі проблеми та рішення, запропонована модель побудови структурної інформації

веб-сторінки для подальшого її використання. Буде також програмне забезпечення для демонстрації та вимірювання точності алгоритму.

2.3 Опис постановки задачі

Метою роботи є вивчення існуючих підходів до пошуку важливої інформації з веб-сторінок, створення алгоритму аналізу сторінок та побудови структурованої інформації на основі отриманих даних, яка буде зручно використовуватись для виконання інших завдань з вилучення важливої інформації та пошуку завдання. Також буде встановлено демонстраційне програмне забезпечення для вирівнювання, і буде визначена точність методу. Програмне забезпечення покладається на прийняття HTML-сторінки та надання важливої інформації.

Основні завдання роботи:

- аналіз проблеми вилучення інформації з веб-сторінок (HTML)
- створення алгоритму аналізу веб-сторінки з точки зору вилучення важливої інформації;
- створення структурованої моделі, яка буде результатом аналізу веб-сторінки, і яку зручно використовувати для виконання завдань з вилучення інформації та пошуку;
- створення програмного додатку для демонстрації алгоритму;
- тестування алгоритму, визначення його точності.

Вимоги до програмного забезпечення:

- розбір коду HTML-документа;
- аналіз коду веб-сторінки на важливу інформацію;
- побудова структурованої моделі;
- видача важливої інформації.

Подання важливої інформації для демонстрації буде здійснюватися у вигляді HTML-сторінки, яка містить лише важливу інформацію, або сторінки з усією інформацією, що має різний колір за важливістю.

Таким чином, у розділі було сформульовано проблему та надано перелік характеристик та можливостей програмного забезпечення.

3 МОДЕЛЮВАННЯ АЛГОРИТМУ

3.2 Моделювання алгоритму

Результатом роботи буде алгоритм та програмний застосунок, яких отримає веб-сторінку як вхід і надасть структуровану модель з додатковою інформацією, яка допомагає ідентифікувати вміст сторінки. А також як приклад - програма визначить основну інформацію на сторінці за допомогою результату аналізу.

Для цього потрібно запропонувати рішення, його етапи та підзадачі, а також визначити вимоги до самої програми.

Алгоритм роботи програми складається з наступних основних етапів:

- розбір HTML-коду

- Перетворення HTML у модель для подальшого аналізу;

- аналіз (кроки аналізу будуть запропоновані та обговорені в наступних розділах)

- визначення основної інформації за результатами аналізу.

Для розбору HTML-коду ми будемо використовувати наявну бібліотеку програм - lxml для мови програмування Python.

Як було сказано раніше, HTML надає хороші можливості для візуального оформлення документа, але він не дає достатньо інформації про зміст. Таким чином, вам слід проаналізувати документ з точки зору його візуального представлення для визначення основних блоків сторінки.

Загалом завдання виглядає як один із етапів до вилучення текстової інформації, полегшує процес пошуку необхідної інформації, завдяки структурованому виду з додатковою інформацією про фрагменти документа разом із ключовими словами можна визначити в аналізі.

Аналізуючи підходи до створення веб-сторінки та методи її візуального форматування, можна виділити різні факти. Ці знання можна розділити на знання про оформлення тексту на веб-сторінці та розташування елементів один до одного.

Для оформлення тексту на веб-сторінці використовується загальновідомий набір тегів HTML разом із CSS. Можна виділити наступні теги:

- <b>, <i>, та інші, метою яких є виділення частини тексту;
- <h1>, <h2>, <h3> та інші, призначені для позначення заголовків на сторінці;
- <p>, <pre>, , ,
 та інші, які призначені для розбиття вмісту на блоки.

Теоретично ці теги можуть допомогти класифікувати інформацію, а також вказати, яка інформація є основною, наприклад, якщо текст вказаний як заголовок.

Ви також можете вибрати теги, які немає сенсу аналізувати, оскільки вони не відображаються на сторінці. Такими тегами, наприклад, є "script", "meta", "noscript", "iframe", "head", "form", "input", "fieldset", "select", "style".

Розглядаючи структуру веб-сторінки, її можна розділити на 2 частини - головну частину з основним текстом, та різні елементи стандартного шаблону сторінки, такі як навігація, рекламні блоки, коментарі, різні набори посилань. Ці блоки документів завжди розташовані в певному порядку - навколо основного змісту сторінки. Таким чином, завдання пошуку основного контенту також може розглядатися з точки зору ідентифікації блоків шаблону сторінки.

Слід зазначити, що, як показали спостереження, використання вмісту тегів за їх назвою, призначенням, а також використання назв класів тегів є досить обмеженим. Ці обмеження засновані на тому, що не тільки основний текст може використовувати теги для виділення основного тексту на сторінці. Наприклад, текст (веб-сайт oracle.com), виділений на рис. 3.1, розміщений у

тегах <h1>, хоча це основний текст для навігації, а також має невеликий розмір. Таких запасів на сторінках сайтів дуже багато, тому ці факти потрібно враховувати, щоб використовувати евристичні поняття.

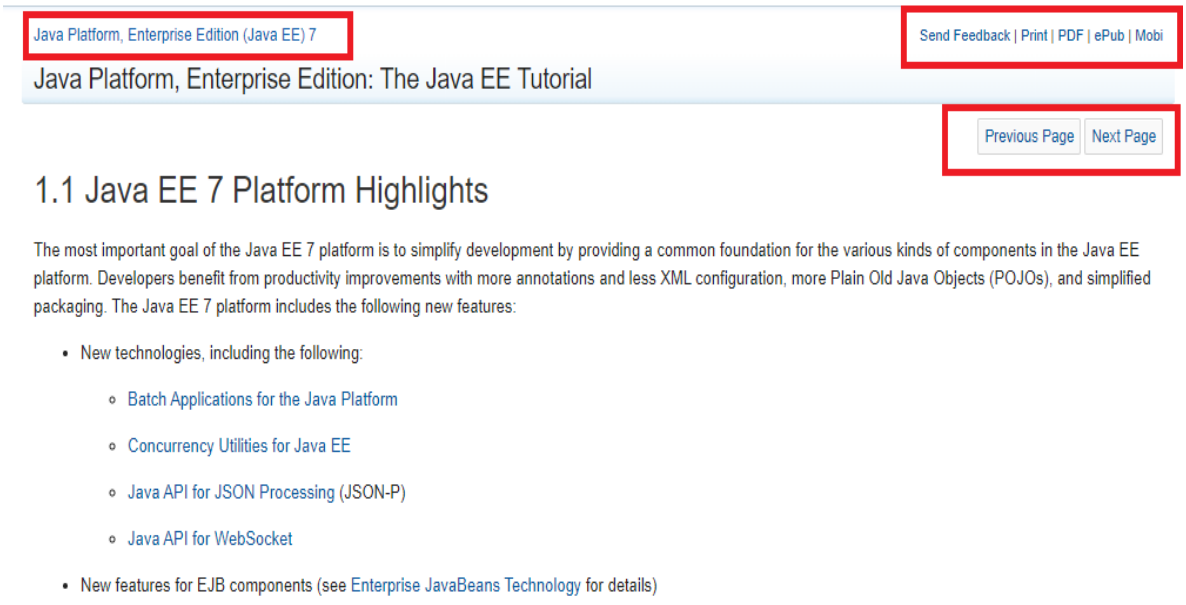

Рисунок 3.1 – Веб-сторінка сайту oracle.com

Розділимо блоки сторінки на основні та шаблони (навігація, реклама і т. д.). На рис. 3.2 показує веб-сторінку з вибраними блоками. Елементи шаблону виділені червоним.

Для класифікації елементів можна виділити такі класи: основний блок, короткий блок, кандидат на основний блок, шаблон.

Рисунок 3.2 – Веб-сторінка с шаблонними елементами

Проаналізувавши процес створення сторінки, ми можемо виділити наступні факти:

- короткі блоки, що містять посилання, майже завжди є шаблонними (за винятком того, що може бути назвою)

- будь-які блоки, що містять багато посилань, майже завжди є шаблонами. (За винятком текстових блоків)

- довгі блоки, що містять текст, майже завжди є головними, тоді як усі інші довгі блоки майже завжди є шаблонами (або коментарями);

- основні та шаблонні блоки, як правило, оточені іншими подібними (основними або шаблонами відповідно) блоками;

- Блоки з багатою кількістю слів можна вважати кандидатами в основні блоки.

Алгоритм буде заснований на цих основних евристичних концепціях. Він класифікує блоки документів на основні та шаблонні блоки за допомогою пошуку основного тексту та блоків шаблонів, а потім комбінує ці блоки на основі вишневих концепцій про розташування елементів.

Для визначення цих класів ми скористаємось спостереженням, що багато тексту є основним блоком, а більша частина тексту у посиланні - елемент шаблону (навігація тощо). Для цього ми вводимо 2 поняття для блоків:

- щільність посилання - відношення довжини вмісту блоків посилань до всього вмісту розглянутого блоку (значення від 0.0 до 1.0);

- length - довжина тексту в блоці.

Граничні значення для цих властивостей блоку будуть взяті з інших експериментів. Таким чином, якщо щільність зв’язку більше 0.3, то це елемент шаблону, інакше ми розглянемо елемент з точки зору властивості довжини. Для короткого тексту це значення повинно бути менше 70. Процес вибору класу елементів для блоку(рисунок 3.3).

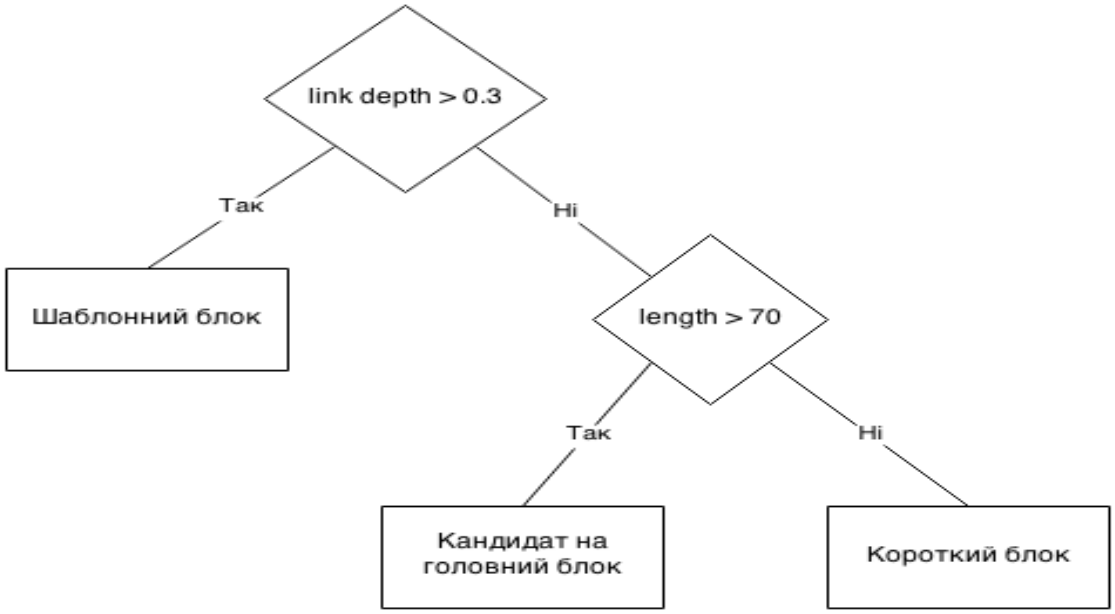

Рисунок 3.3 - Процес вибору класу елементу для блоку

Перш ніж виконати алгоритм, потрібно побудувати дерево елементів, видалити з нього непотрібні елементи, не пов’язані з візуальним відображенням сторінки.

Коли описані кроки алгоритму розглянемо приклад обробки документів, має структуру, показану на рис. 3.4.

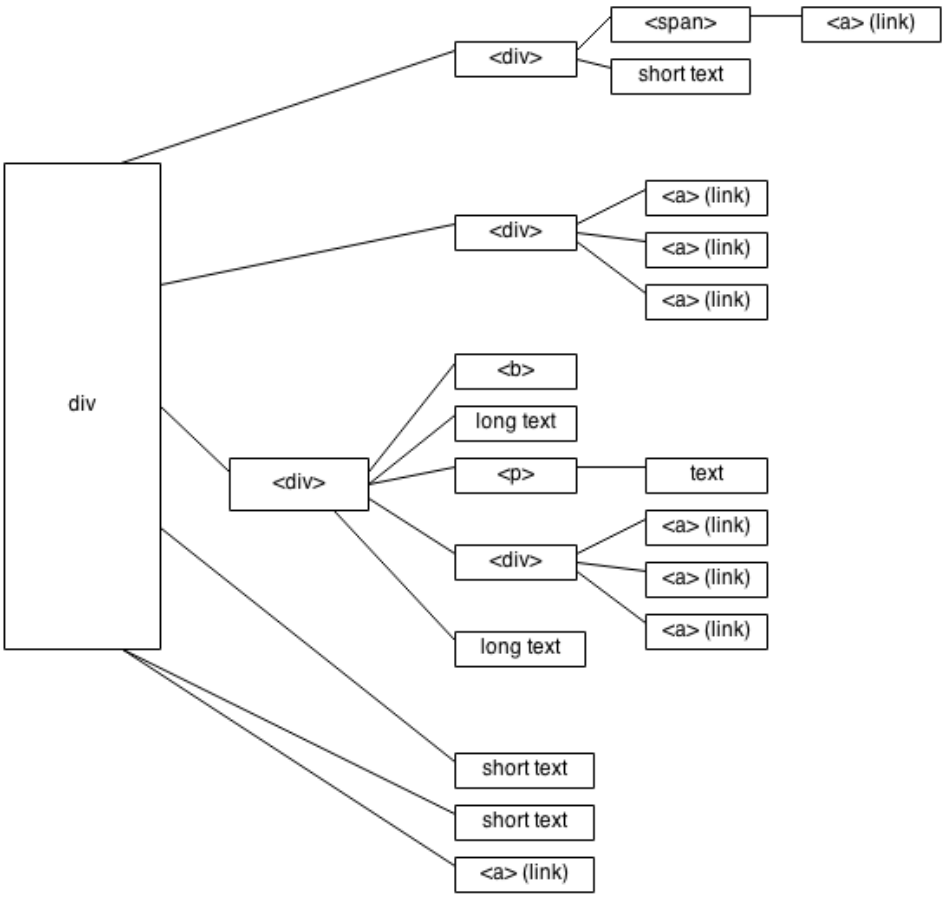

Рисунок 3.4 – Структура розглядаємого в алгоритмі документу

Етапи алгоритму:

а) визначення значень глибини та довжини ланки;

б) класифікація елементів за алгоритмом, описаним раніше. Результат цього етапу можна побачити на рис. 3.5. Співвідношення кольору і класу елемента:

1) червоний - елемент шаблону;

2) зелений - основний елемент;

3) сірий - короткий елемент;

4) жовтий - кандидат на головний елемент;

в) об’єднання сусідніх блоків в один. Під цим ми маємо на увазі визначення класу одного блоку. Для цього ми використаємо описану раніше евристику, на основі якої можна виділити наступні твердження:

1) якщо клас блоку є основним, а клас наступного - кандидатом на основний блок, то обидва блоки є основними;

2) якщо клас блоку короткий, а клас наступного основного, то цей блок є кандидатом на основний блок;

3) якщо клас блоку короткий, а клас наступного є кандидатом на основний блок, то цей блок є кандидатом на основний блок;

4) якщо клас блоку короткий, а клас наступного - шаблон, то цей блок є шаблоном;

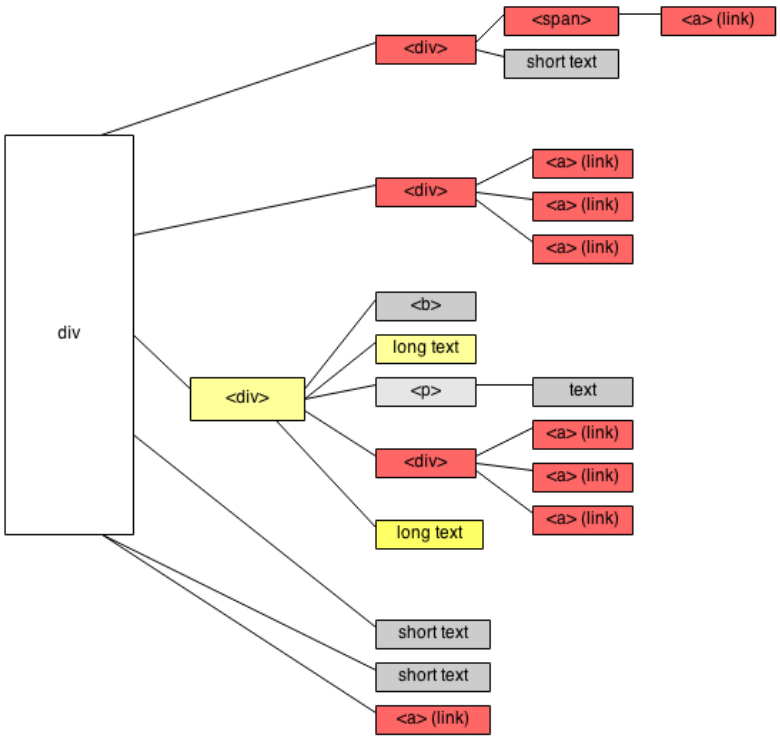

Рисунок 3.5 – Результат класифікації елементів

Таким чином, ми об’єднаємо елементи дерева в послідовні групи, манекени будуть більшими за розмірами. Також на цьому етапі будуть уточнені класи для деяких коротких блоків, які можуть бути як основними, так і

шаблоновими, а також уточнені класи елементів, які є кандидатами для основних. Результат цього етапу (рисунок 3.6).

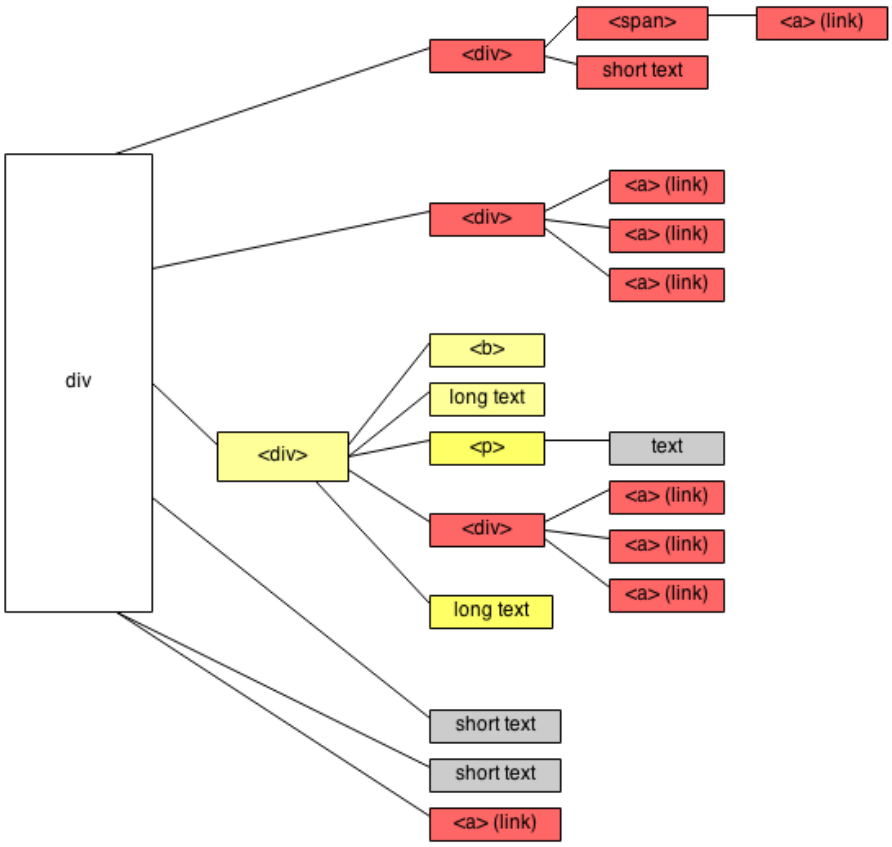

Рисунок 3.6 – Результат об’єднання класів сусідніх елементів

г) об’єднання груп (за класом) елементів. Після цього були створені групи елементів, їх можна об’єднати у великі логічні блоки за тими ж правилами. Це буде остаточне уточнення класу для елементів. Результат цього етапу (рисунок 3.7). На цьому етапі будуть створені великі групи кандидатів, згруповані в основні блоки, а великі основні групи та кандидати також заповнять невеликі групи елементів серед основних. Такі групи можуть з'являтися, оскільки текст може містити набори посилань тощо;

д) пошук назв на сторінці. Після класифікації блоків ви можете знайти заголовки, використовуючи визначення класів та інформацію про теги елементів (<h1>, <h2>). Зауважте, що заголовки можуть бути також у тезі посилання (<a>). Тому є сенс шукати заголовки на цьому етапі після того, як посилання разом з попередніми елементами вже класифіковані. Таким чином, ідентифікаційний заголовок не змінить класифікацію сусідніх елементів. Ви також можете використовувати інформацію про розташування елементів для пошуку заголовка - зазвичай заголовок розташований перед основним текстом;

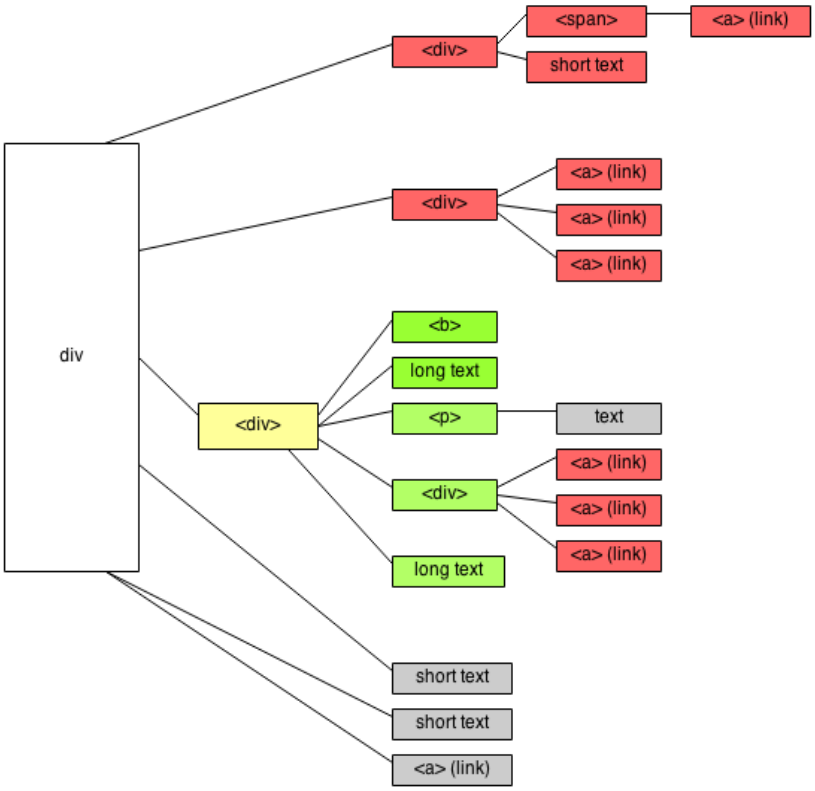

Рисунок 3.7 – Результат об’єднання груп классів елементів

ж) на цьому етапі ви можете побачити елементи дерева, які мають основний клас блоку, і, таким чином, виставити один і той же клас для всіх дочірніх елементів, які можуть мати клас - короткий блок або шаблон (наприклад, у випадку списку посилань між текстом). Результат (рисунок 3.8).

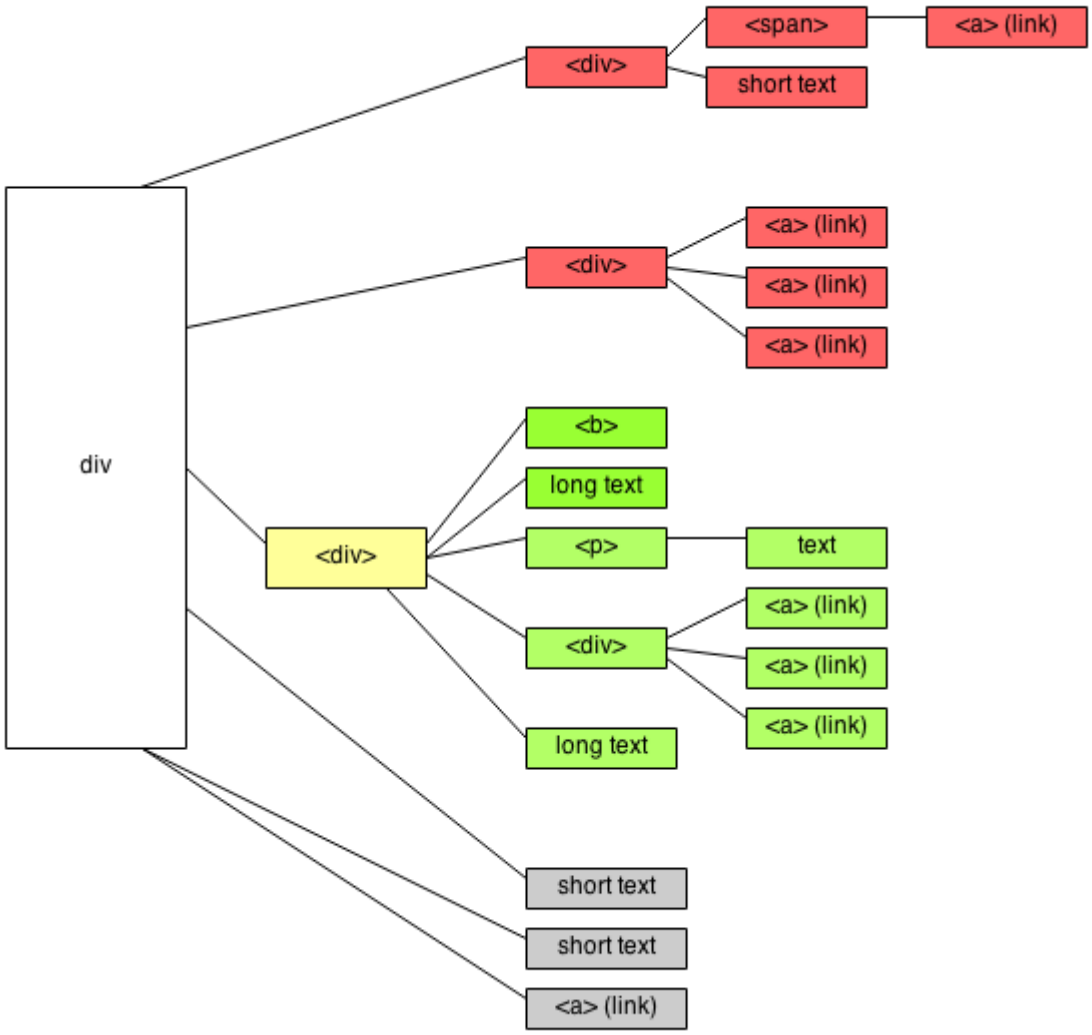

Рисунок 3.8 – Результат обробки дочірніх елементів головних блоків

Таким чином, алгоритм знаходить основні блоки, що містять текст з основною інформацією, а також блоки шаблонів, якими можуть бути навігація, реклама тощо. В результаті алгоритм створює дерево класифікованих елементів, у яких елементи з класом головного блоку або кандидат на основний блок мають важливу інформацію, яка може бути використана для різних цілей, а також для проведення більш детального аналізу.

Для перевірки точності алгоритму його необхідно перевірити на великій кількості даних різного роду. Такі алгоритми, засновані переважно на

евристиці, не мають чіткої математичної бази, тому їх потрібно багато перевірити на практиці. Для тестових даних буде використовуватися існуючий великий набір веб-сторінок з різних новинних порталів, блогів тощо. Цей комплект доступний в Інтернеті і називається L3S-GN1 [11]. Колекція сторінок складається з 621 статті новин з 408 різних веб-сайтів. Статті з новин були вибрані випадковим чином з 254 000 статей з 7854 сайтів, які були отримані після моніторингу новин Google у першій половині 2008 року.

Сторінки мають спеціальні позначення в атрибутах класів тегів, які вказують на тип елемента та важливість інформації. Ці позначки будуть порівнянні з даними, які будуть отримані після тестування алгоритму. На основі цих даних можна обчислити приблизну точність створеного алгоритму.

Таким чином, було створено алгоритми та програмне забезпечення, яке було протестовано на 621 підготовлених документах. Точність алгоритму становить 84,14% і є добрим результатом. Для підвищення точності можна використовувати додаткову евристику або комбінувати алгоритм з іншими існуючими підходами.

4. ОПИС РОЗРОБЛЕНОЇ СИСТЕМИ

4.1 Опис користувацького інтерфейсу

Алгоритм розробки аналізує сторінку HTML і знаходить важливу інформацію серед всього змісту документа. Програмне додаток, що демонструє роботу алгоритму, може отримувати веб-сторінку та надавати той самий документ, але лише з основною інформацією.

Програмі необхідно пройти шлях до місця розташування коду сторінки. Результат алгоритму буде збережено у файлі "result.html". Формати запуску:

-"python application.py <шлях до html-файла> - extract ";

- "python application.py <шлях до html-файлу> - highlight ";

Програма може бути запущена в двох режимах, зі спеціальними прапорами при запуску. Для тестування програмного забезпечення ми використаємо код сторінки зі статтями на oracle.com (рис. 4.1).

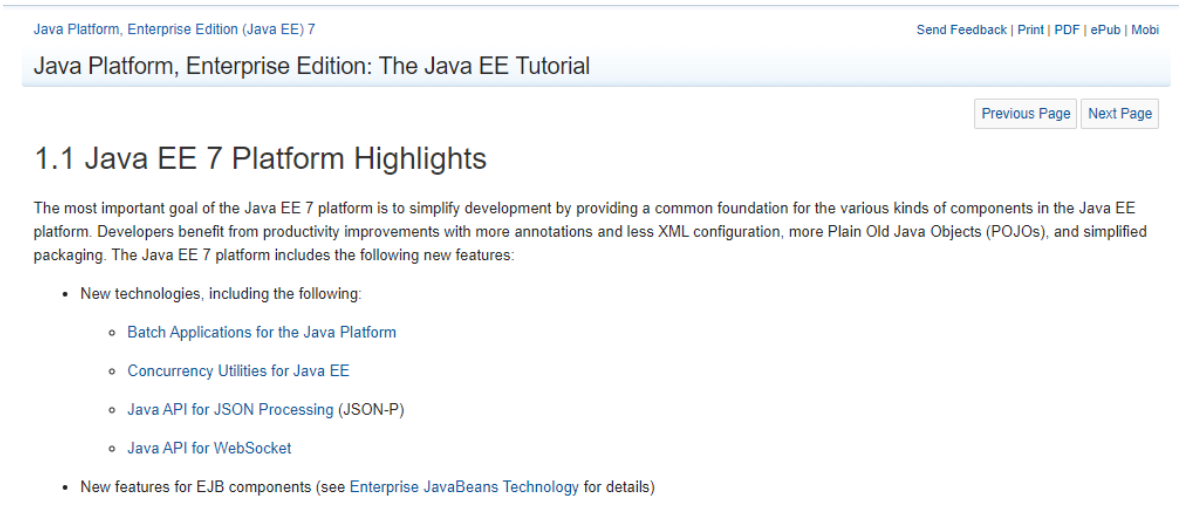

Рисунок 4.1 – Вигляд сторінки, яка використовується для демонстрації роботи програмного застусунку

Перший флаг “-extract” – створити документ тільки з важливою інформацією. Приклад вихідного документу наведен на рис. 4.2.

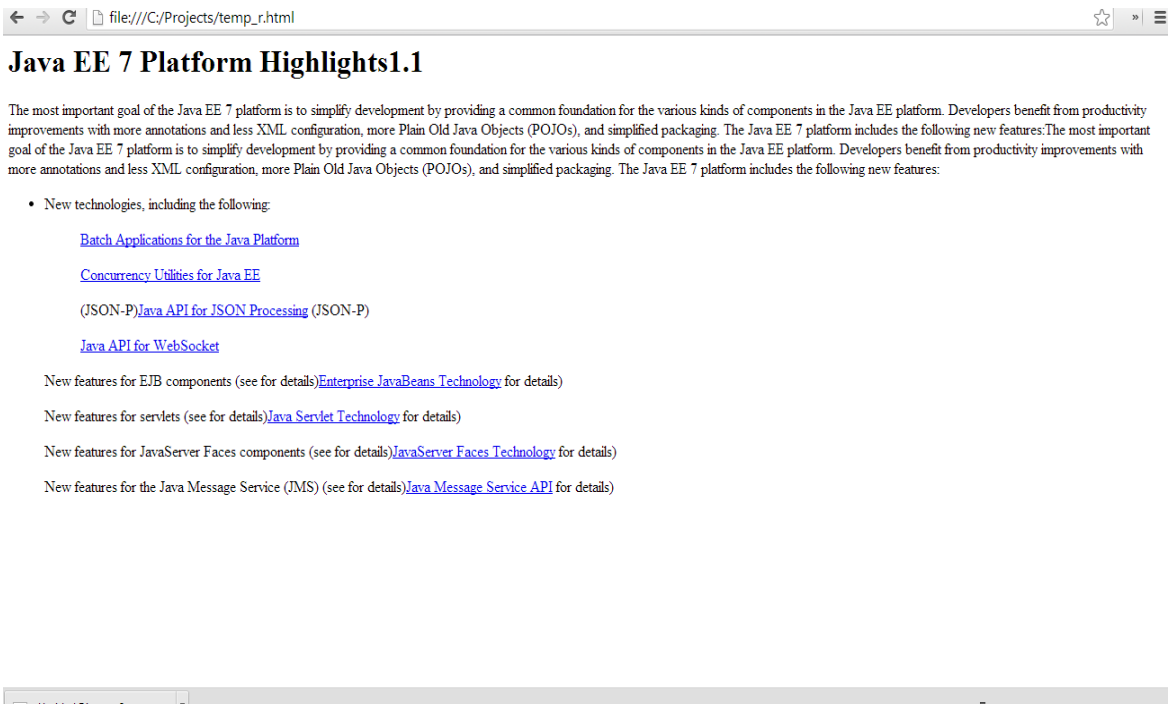

Рисунок 4.2 – Результат роботи програми, як сторінка с важливою інформацією

Інший флаг “-highlight” – створити документ, де елементи сторінки будуть мати різний фон, який відповідає класу документа. Приклад вихідного документу наведен на рис. 4.4.

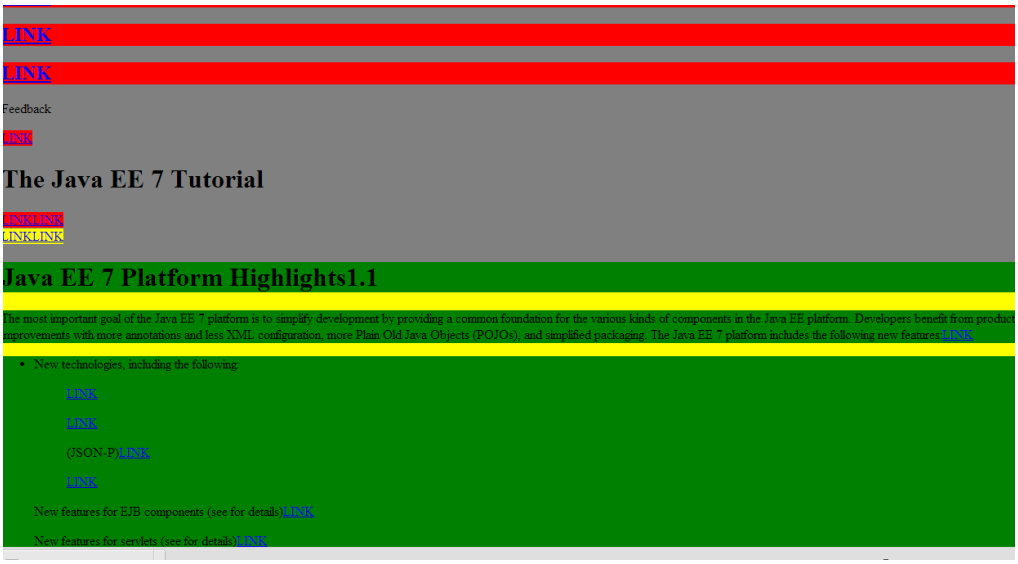

Рисунок 4.4 – Результат роботи програми, як сторінка с виділенною важливою

Таким чином, у цьому розділі було представлено реалізацію розробленої функціональності програмного забезпечення. Були вказівки на те як і в яких режимах можна запустити програмне додаток, що демонструє роботу алгоритму.

ВИСНОВКИ

Отже результатом підсумкової кваліфікаційної роботи стала розробка алгоритму та програмного забезпечення, яке шукає важливу інформацію на веб-сторінці. Інтернет зараз містить величезну кількість інформації, знань. Користувачі на різних умовах можуть переглядати різноманітні документи, аудіо- і відеофайли. Однак це різноманіття даних приховує в собі проблеми, які можуть виникнути не тільки при аналізі, а й при пошуку необхідної інформації в Інтернет.

Вирішення цієї проблеми може бути корисним для інших завдань щодо виявлення важливої інформації. Ідентифікація таких блоків сторінок дозволяє не витрачати час на аналіз частин документа, які зазвичай не мають важливої чи основної інформації.

Була також розрахована точність створеного алгоритму, який вимірювався за підготовленими даними за допомогою колекції документів L3S-GN1. Точність розробленого алгоритму становить приблизно 84%, що є доброю точністю. Але цей алгоритм все ще потребує додаткової обробки, наприклад, шляхом додавання нової евристики.

Під час роботи я закріпив навички розробки програмного продукту, починаючи від аналізу та проектування та закінчуючи тестуванням програмного продукту. Я вдосконалив навички розробки додатків та зафіксував вдосконалені методи роботи з мовою Python. Вдосконалено знання про використання моделей дизайну та написання коду, який легко підтримувати.

Подальший розвиток алгоритму можливий у кількох напрямках:

- впровадження нових евристичних концепцій та підходів до пошуку інформації;
- прискорення швидкості алгоритму / програми;
- вдосконалення вихідної моделі даних;

- створення окремої бібліотеки програмного забезпечення для повторного використання алгоритму в інших програмах.

ПЕРЕЛІК ПОСИЛАНЬ

1. Лутц М. Программирование на Python // М. Лутц: пер. с англ. – СПб.: Символ-Плюс. 2011. – 992 с.

2. Bar-Yossef, Z. and Rajagopalan, S. Template Detection via Data Mining and its Applications, WWW 2002.

3. Сервіс для читання веб-сторінок в режимі офлайн Instapaper URL: https://www.instapaper.com(дата звернення : 09.03.2020).

4. Інформаційний портал про алгоритми. Алгоритм MDR. URL : http://curtis.ml.cmu.edu/ (дата звернення : 09.05.2020).

5. Колекція веб-сторінок L3S-GN1 URL: https://github.com/geodrome/page-signal (дата звернення : 22.04.2020).

6. И. Некрестьянов, Е. Павлова. Обнаружение структурного подобия HTML-документов. // Труды четвертой всероссийской конференции RCDL'2002, 38-54, Дубна, Россия, 2002.

7. Cooley, R., Mobasher, B. and Srivastava, J. Data preparation for mining World Wide Web browsing patterns. Journal of Knowledge and Information Systems, (1) 1, 1999.

8. Ziv Bar-Yossef, Sridhar Rajagopalan Template Detection via Data Mining and its Applications // In Proceedings of WWW2002, May 7-11, 2002, Honolulu, Hawaii, USA. URL: http://www2002.org/CDROM/refereed/579/ (дата звернення : 10.04.2020).

9. Сайт факультету інформаціонних технологій Брненского технічного університету URL: http://www.fit.vutbr.cz/ (дата звернення : 14.04.2020).

10. Lakshmish Ramaswamy, Arun Iyengar, Ling Liu, and Fred Douglis. Automatic Detection of Fragments in Dynamically Generated Web Pages // In Proceedings of the 13th International World Wide Web Conference New York City, May 2004. URL: http://www.research.ibm.com/people/i/iyengar/www2004.pdf (дата звернення : 15.04.2020).

11. Lakshmish Ramaswamy, Arun Iyengar, Ling Liu, and Fred Douglis. Automatic Fragment Detection in Dynamic Web Pages and its Impact on Caching // To appear in IEEE Transactions on Knowledge and Data Engineering (TKDE'05). URL: http://www.research.ibm.com/people/i/iyengar/TKDEFragmentDetection.pdf (дата звернення 09.04.2020)

12. Kleinberg, J. Authoritative Sources in a Hyperlinked Environment. ACM-SIAM Symposium on Discrete Algorithms, 1998.

13. Lee, M.L., Ling, W. and Low, W.L. Intelliclean: A knowledge-based intelligent data cleaner. KDD-2000, 2000.

14. Shian-Hua Lin and Jan-Ming Ho. Discovering Informative Content Blocks from Web Documents, KDD-02, 2002.