

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК, ФІЗИКИ ТА МАТЕМАТИКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

АВТОМАТИЗОВАНА СИСТЕМА ЗБОРУ, АНАЛІЗУ ТА ОБРОБКИ
ІНФОРМАЦІЇ ПРО ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
В МЕРЕЖІ ІНТЕРНЕТ

Кваліфікаційна робота

на здобуття ступеня вищої освіти «магістр»

Виконав: здобувач 2 курсу 12-241М з групи
Спеціальності 121 Інженерія програмного
забезпечення

Освітньо-професійної програми «Інженерія
програмного забезпечення»

Євгеній ІЩЕНКО

Керівник: доктор педагогічних наук, професор
Співаковський Олександр Володимирович

Рецензент: старший інженер-програміст ІТ-
компанії USoftware

Краковський О.О.

Івано-Франківськ – 2025

ЗМІСТ

ВСТУП.....	3
РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ РОЗРОБКИ АВТОМАТИЗОВАНИХ СИСТЕМ ЗБОРУ ТА АНАЛІЗУ ІНФОРМАЦІЇ В ІНТЕРНЕТ СЕРЕДОВИЩІ.....	
1.1. Інформаційний простір закладів вищої освіти як об’єкт моніторингу.....	9
1.2. Методи збору та обробки інформації з відкритих джерел.....	11
1.3. Алгоритми аналізу та класифікації текстової інформації.....	21
Висновки до розділу 1.....	26
РОЗДІЛ 2. МЕТОДИКА РОЗРОБКИ ТА АНАЛІЗУ АВТОМАТИЗОВАНОЇ СИСТЕМИ ЗБОРУ ІНФОРМАЦІЇ.....	
2.1. Обґрунтування вибору напрямку дослідження.....	28
2.2. Методи збору та обробки даних.....	29
2.3. Порівняльна характеристика методів аналізу текстів.....	30
2.4. Методика проведення дослідження.....	33
2.5. Реалізація системи збору та аналізу інформації.....	34
Висновки до розділу 2.....	46
ВИСНОВКИ	47
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ.....	49

ВСТУП

Актуальність дослідження. В умовах глобальної цифрової трансформації суспільства та стрімкого розвитку інформаційних технологій Інтернет-простір перетворився на ключову арену для формування громадської думки та репутаційного капіталу. Для закладів вищої освіти (ЗВО), які функціонують у висококонкурентному середовищі, боротьба за абітурієнта, наукові гранти та міжнародних партнерів напряду залежить від їхнього публічного іміджу. Особливо це актуально в умовах повномасштабної війни в Україні, коли інформаційна безпека та довіра до освітніх інституцій стають критичними чинниками стабільності й розвитку. У цей період репутаційний образ університетів не лише впливає на їхню впізнаваність, але й визначає місце у національних і міжнародних рейтингах, рівень підтримки з боку держави та громадськості. Саме тому питання ефективного моніторингу, аналізу та управління інформаційними потоками набуває сьогодні особливої ваги.

Для Херсонського державного університету (ХДУ) як провідного освітнього та наукового центру регіону життєво важливо своєчасно відстежувати, аналізувати та систематизувати весь спектр інформації, що поширюється про нього в мережі Інтернет. Це вже не лише питання престижу, а складова стратегічної безпеки та розвитку. Оперативний моніторинг дозволяє не лише пасивно оцінювати поточний імідж закладу, але й проактивно реагувати на виклики: виявляти потенційні та реальні інформаційні загрози, осередки дезінформації, відстежувати динаміку громадських настроїв та своєчасно коригувати власну комунікаційну політику.[3]

В умовах воєнного стану особливої актуальності набуває питання збереження інтелектуального потенціалу країни. Формування позитивного образу українських університетів, зокрема ХДУ, сприяє підвищенню довіри молоді до національної системи освіти, мотивує абітурієнтів залишатися в

Україні, здобувати якісну освіту вдома та долучатися до відновлення і розвитку держави після війни. Таким чином, інформаційна присутність університету в мережі стає не лише інструментом комунікації, а й чинником державотворчої та соціальної стабільності.

Сучасні дослідження в галузі інформаційного аналізу, інтелектуального видобутку даних (Data Mining)[43] та автоматизованих систем збору даних активно розвиваються завдяки впровадженню технологій штучного інтелекту. Зокрема, у працях та розробках таких компаній і відкритих спільнот, як OpenAI, Google DeepMind (Gemini), Hugging Face, сформовано потужну науково-практичну базу для створення інтелектуальних систем аналізу текстової інформації. Ці рішення демонструють високу ефективність у задачах обробки природної мови (NLP), семантичного аналізу, класифікації контенту та виявлення емоційно-тональної спрямованості повідомлень. [4] Саме інтеграція подібних підходів відкриває можливості для розробки автоматизованих систем моніторингу інформаційного простору закладів вищої освіти, зокрема Херсонського державного університету.

Проте, незважаючи на наявність загальних інструментів моніторингу ЗМІ та соціальних мереж, залишається недостатньо дослідженим питання розробки спеціалізованої системи, адаптованої саме до специфіки діяльності університету та українського інформаційного середовища. Популярні платформи, такі як Brandwatch, Meltwater, YouScan, Loomly, Hootsuite чи Sprout Social, орієнтовані переважно на комерційний сегмент і побудовані навколо аналізу споживацької поведінки, маркетингових кампаній та управління репутацією брендів. [5] Такі системи ефективно працюють із рекламними метриками, охопленням, конверсіями та тональністю відгуків, однак не враховують особливостей освітнього дискурсу — наукової, академічної, міжнародної та соціальної складових діяльності закладів вищої освіти.

У результаті спостерігається методологічний і практичний розрив між наявними універсальними інструментами моніторингу та реальними

потребами інформаційно-аналітичних підрозділів університетів, які потребують не просто оцінки медійної активності, а глибокого контекстного аналізу публікацій, що формують імідж і довіру до закладу.

Таким чином, актуальність даної кваліфікаційної роботи зумовлена нагальною потребою у створенні цілісної системи, здатної забезпечити автоматизований моніторинг, аналіз та обробку інформації про Херсонський державний університет у мережі Інтернет. В умовах інформаційних викликів і динамічного розвитку цифрового середовища університету необхідно мати інструмент, що дозволить оперативно відстежувати медійну активність, аналізувати тональність та контекст повідомлень, виявляти потенційні загрози репутації й формувати об'єктивну картину інформаційного поля.[6]

Наукове значення роботи полягає у розробці концептуальної моделі та алгоритмічного забезпечення автоматизованої системи аналізу інформаційних потоків, адаптованої до специфіки функціонування закладів вищої освіти. У межах дослідження передбачено використання сучасних підходів до обробки природної мови (NLP), морфологічного аналізу, класифікації текстів за тематикою й полярністю (позитивна, негативна, нейтральна). Особлива увага приділяється врахуванню лінгвістичних особливостей українського сегменту Інтернету, що забезпечує більш високу точність і релевантність отриманих результатів у порівнянні з універсальними моделями.

Практичне значення роботи полягає у створенні дієвого програмного інструменту, який може бути впроваджений у діяльність Херсонського державного університету для інформаційно-аналітичної підтримки управлінських рішень. Розроблена система дає змогу прес-службі та адміністрації університету:

- здійснювати постійний моніторинг публікацій у ЗМІ та соціальних мережах;

- автоматично визначати зміст, тематику та емоційно-тональну спрямованість повідомлень;

виявляти джерела дезінформації або негативного інформаційного впливу;

підвищувати ефективність комунікаційної політики закладу;

своєчасно нейтралізувати репутаційні ризики та зміцнювати позитивний імідж ХДУ як відкритої, сучасної та надійної освітньої установи.

Таким чином, результати дослідження мають як наукову новизну, так і практичну цінність, забезпечуючи основу для подальшого розвитку систем інтелектуального моніторингу освітнього простору в Україні.

Зв'язок роботи з науковими програмами, планами, темами.

Кваліфікаційна робота виконана в межах наукового напрямку кафедри комп'ютерних наук Херсонського державного університету, що передбачає розробку та впровадження інтелектуальних систем збору, аналізу й обробки інформації. Тема дослідження узгоджується з планом науково-дослідної роботи факультету комп'ютерних наук, фізики та математики й відповідає стратегії цифрового розвитку університету.

Мета кваліфікаційної роботи полягає у розробці автоматизованої системи збору, аналізу та обробки інформації про Херсонський державний університет у мережі Інтернет з метою підвищення ефективності моніторингу інформаційного простору та формування аналітичних висновків щодо інформаційного іміджу закладу.

Для досягнення поставленої мети передбачено виконання таких **завдань:**

- 1) здійснити теоретичний аналіз сучасних підходів до збору, аналізу та обробки інформації в інтернет-середовищі.
- 2) Дослідити технології вебпарсингу, обробки природної мови та аналітики даних, що застосовуються для моніторингу інформаційного простору.
- 3) Розробити архітектуру автоматизованої системи збору та аналізу інформації про Херсонський державний університет.

- 4) Реалізувати програмний прототип системи з використанням сучасних інструментів вебпарсингу, баз даних та аналітичних модулів.
- 5) Провести тестування, перевірку функціональності та оцінку ефективності розробленої системи.

Об'єкт дослідження: процес збору, аналізу та обробки інформації про діяльність закладів вищої освіти у мережі Інтернет.

Предмет дослідження: методи, алгоритми та програмні засоби автоматизованого збору, аналізу та обробки інформації про Херсонський державний університет у мережі Інтернет.

Методи і методики дослідження:

- Теоретичні методи: аналіз наукових джерел, присвячених питанням інформаційного моніторингу, обробки природної мови та автоматизації аналітичних процесів; порівняльний аналіз сучасних підходів до збору та структурування даних.

- Практичні методи: застосування інструментів вебпарсингу (HTML-аналітики, регулярних виразів), алгоритмів обробки текстової інформації (морфологічний аналіз, класифікація тональності), а також засобів програмування (PHP, Laravel, Python, FastAPI) для реалізації функціональної частини системи.[7]

- Методи оцінювання: тестування, аналіз продуктивності, перевірка точності збору та коректності аналітичних результатів.

Наукова новизна одержаних результатів полягає у розробці комплексної моделі автоматизованої системи збору та аналізу інформації про діяльність університету, що поєднує технології вебпарсингу, лінгвістичного аналізу та машинної обробки тексту в єдиній інформаційно-аналітичній платформі. Розроблена система дозволяє отримувати структуровані дані про інформаційний простір університету в автоматичному режимі.

Практичне значення одержаних результатів. Результати дослідження можуть бути використані для створення університетських

інформаційно-аналітичних систем моніторингу, що забезпечують автоматичний збір даних із вебресурсів та соціальних медіа. Розроблена система може бути впроваджена у відділ інформаційної політики або медіацентр Херсонського державного університету для відстеження динаміки інформаційного іміджу закладу в онлайновому середовищі.

Апробація та публікація результатів дослідження. Основні результати кваліфікаційної роботи були обговорені на засіданні кафедри комп'ютерних наук Херсонського державного університету (протокол № ____ від ____ 2025 р.) та представлені у вигляді тез доповіді на студентській науково-практичній конференції ХДУ (листопад 2025 р.).

Структура роботи. Робота складається зі вступу, двох розділів, висновків, списку використаних джерел, що налічує 40 джерел.

РОЗДІЛ 1

ТЕОРЕТИЧНІ ОСНОВИ РОЗРОБКИ АВТОМАТИЗОВАНИХ СИСТЕМ ЗБОРУ ТА АНАЛІЗУ ІНФОРМАЦІЇ В ІНТЕРНЕТ-СЕРЕДОВИЩІ

1.1. Інформаційний простір закладів вищої освіти як об'єкт моніторингу

Під інформаційним простором закладу вищої освіти розуміють сукупність електронних та цифрових джерел, каналів і комунікаційних практик, через які формуються, розповсюджуються та сприймаються відомості про діяльність університету, його структури, наукові та освітні результати, події та зацікавлені групи. Інформаційний простір включає як офіційні комунікативні майданчики (офіційний вебсайт університету, сайти факультетів, сторінки кафедр), так і неофіційні джерела — новинні ресурси, блоги, тематичні форуми, платформу соціальних мереж (Facebook, Telegram-канали, Instagram, LinkedIn), а також агрегатори новин і пошукові системи, у яких відображається інформація про заклад.

Структурно інформаційний простір ЗВО можна розглядати як багаторівневу систему, що складається з наступних компонентів:

офіційні джерела — сайти університету та підрозділів, офіційні сторінки в соцмережах, прес-релізи, електронні видання університету; ці джерела формують офіційну позицію та первинний масив контенту.

новинні та медіа-платформи — локальні і всеукраїнські онлайн-засоби масової інформації, що висвітлюють події університету; їхній тон і контекст істотно впливають на зовнішнє сприйняття інституції.

соціальні мережі та комуніті — публічні та закриті групи, канали й сторінки, де формуються громадські обговорення, оцінки й реакції (коментарі, репости, лайки).

Академічні репозиторії та публікації — наукові статті, тези конференцій, профілі дослідників у наукових базах; важливі для формування наукового іміджу.

Форуми, блоги та відгуки — ресурси, де здобувачі, випускники і працівники діляться досвідом, дають рекомендації чи критику.

Пошукові системи та агрегатори — індексація контенту, результати пошукових запитів, карти знань (knowledge panels), що визначають видимість університету в Інтернеті.

Моніторинг інформаційного простору університету має кілька завдань: своєчасне виявлення інформації (позитивної, нейтральної, негативної), оцінка тональності та тематики повідомлень, аналіз джерел і каналів поширення, а також побудова аналітичних висновків для ухвалення управлінських рішень. Для адміністрації університету та підрозділів моніторинг виступає інструментом управління репутацією, оцінки ефективності PR-кампаній, виявлення кризових сигналів і формування превентивних заходів.[8]

З погляду методології, інформаційний простір як об'єкт дослідження характеризується динамічністю, гетерогенністю та великою розподіленістю джерел. Динамічність означає швидку зміну вмісту й відгуків (стрічка новин, негайні реакції в соцмережах). Гетерогенність проявляється у різних форматах даних (текст, зображення, відео, метадані). Розподіленість — це наявність інформації в численних незалежних джерелах, які можуть дублювати або трансформувати повідомлення. Ці властивості зумовлюють необхідність застосування автоматизованих підходів збору та обробки інформації, оскільки ручний моніторинг втрачає оперативність і повноту.[9]

До ключових показників, які використовуються при моніторингу інформаційного простору, належать: обсяг згадок, тональність (полярність) згадок, частота появи ключових тем, охоплення аудиторії (reach), індекс впливу джерел (authority), швидкість розповсюдження інформації, а також індикатори взаємодії в соціальних мережах (engagement — лайки, коментарі,

репости). Комбінація цих метрик дозволяє побудувати багатовимірну картину інформаційного поля навколо університету.

Практичною складовою моніторингу є визначення набору релевантних джерел та розробка правил збору даних: формування переліку доменів і ключових фраз, налаштування парсерів і API-підключень, встановлення частоти збору (real-time, hourly, daily) та механізмів збереження даних у структурованому вигляді. Важливим є також питання нормалізації і пре-процесингу тексту (очистка HTML, токенизація, лемматизація), що забезпечує коректну роботу методів аналізу (класифікація, тональний аналіз, виявлення тем).[10]

Моніторинг інформаційного простору має й етичні та правові аспекти: дотримання авторських прав при зборі контенту, правила обробки персональних даних, а також необхідність коректної інтерпретації результатів для запобігання хибних управлінських рішень. Оскільки частина джерел є відкритою, але інша може містити чутливу або персональну інформацію, система моніторингу повинна відповідати національному законодавству та внутрішнім політикам університету.

Отже, інформаційний простір закладу вищої освіти є багатовимірним та динамічним об'єктом дослідження, що вимагає комплексних підходів для його моніторингу. Розробка автоматизованої системи, здатної збирати, структуровувати та аналітично опрацьовувати дані з різних джерел, є обґрунтованою реакцією на потреби сучасного інформаційного менеджменту університету. У наступних підрозділах розділу 1 буде детально розглянуто технологічні підходи до збору й обробки даних та алгоритмічні методи аналізу текстової інформації.

1.2. Методи збору та обробки інформації з відкритих джерел

Веб-парсинг (англ. Web Parsing) — це процес автоматизованого вилучення структурованої інформації з неструктурованих або напівструктурованих веб-ресурсів. У сучасних інформаційних системах він є

базовим елементом технологій OSINT (Open Source Intelligence) — збору, аналізу та інтерпретації відкритих даних з Інтернету.[11]

Метою парсингу є не просто копіювання даних із сайтів, а перетворення веб-контенту у формалізовану, придатну для подальшої аналітики форму: таблиці, бази даних, JSON- або XML-структури тощо.

У найзагальнішому вигляді процес веб-парсингу включає такі етапи:
отримання веб-сторінки — звернення до URL через HTTP(S)-запит;
попередня обробка HTML — очищення від службового коду, скриптів, реклами;

аналіз і розпізнавання структури документа (побудова DOM-моделі);
виділення релевантних фрагментів даних — тексту, зображень, посилань, дат.

структурування та збереження результатів у базі даних або файл.

Веб-парсинг слід відрізняти від суміжних понять:

Веб-краулінг (Web Crawling) — автоматичний обхід сайтів для виявлення посилань.

Веб-скрейпінг (Web Scraping) — процес автоматичного отримання та попереднього структурування даних із вебсторінок, призначених для перегляду людиною. (збір сирих HTML-даних).[1]

Веб-парсинг (Web Parsing) — власне синтаксичний аналіз і формалізація цих даних.

Розвиток веб-технологій призвів до появи кількох концептуально різних підходів до парсингу:

DOM-орієнтований парсинг

Базується на побудові Document Object Model — ієрархічного дерева елементів сторінки.[12]

Дозволяє ідентифікувати потрібні дані через CSS-селектори або запити XPath.[13]

Переваги: висока точність, структурність, зрозуміла логіка навігації.

Недоліки: крихкість до змін верстки, залежність від HTML-структури.

Парсинг на основі регулярних виразів (Regex)[14]

Використовує шаблони пошуку в тексті для вилучення даних без розбору структури.

Переваги: швидкість, простота реалізації.

Недоліки: не підходить для складних або вкладених структур.

Headless-парсинг (на основі рендерингу)

Застосовується до динамічних веб-додатків, де контент формується JavaScript-кодом.

Використовує «безголові» браузери (Selenium, Puppeteer, Playwright), які емулюють реальне завантаження сторінки.[15]

Переваги: здатність обробляти сучасні SPA-сайти.

Недоліки: високе навантаження на ресурси, низька швидкість.

Візуальний або семантичний парсинг

Новітній підхід, у якому сторінка розглядається не як HTML-код, а як візуальна структура.

Для аналізу застосовуються технології комп'ютерного зору (Computer Vision) і машинного навчання (ML).

Переваги: стійкість до змін у HTML-розмітці.

Недоліки: складність реалізації, потреба в навчанні моделей.

У сучасній практиці веб-парсери класифікують за кількома ознаками представлені у таблиці 1.1:

Таблиця 1.1

Класифікація веб-парсерів за рівнем автоматизації, призначенням, способом взаємодії та технологією реалізації

Ознака класифікації	Тип парсерів	Характеристика
За рівнем автоматизації	Ручні, напівавтоматичні, автоматичні	Автоматичні системи можуть самостійно обходити сайти, розпізнавати структуру та зберігати результати.
За призначенням	Загального призначення, тематичні,	Тематичні (наприклад, для новин або цін) мають вбудовані

	спеціалізовані	фільтри.
За способом взаємодії	API-парсери, HTML-парсери	API-парсери працюють через офіційні інтерфейси, HTML-парсери — напряду з кодом сторінки.
За технологією реалізації	Скриптові (Python, PHP), візуальні (GUI-парсери), гібридні	Візуальні системи (ParseHub, Octoparse) не вимагають програмування.

Таблиця 1.2

Порівняльний аналіз сучасних систем і бібліотек веб-парсингу

Назва	Тип	Основна мова	Ключові можливості	Переваги	Недоліки
BeautifulSoup	Бібліотека	Python	Простий DOM-аналіз HTML і XML	Легкість, гнучкість, інтеграція з Requests	Повільна на великих обсягах
Scrapy	Фреймворк	Python	Краулер + скрейпер + парсер, асинхронна обробка	Висока продуктивність, масштабованість	Високий поріг входу
lxml	Бібліотека	Python	XPath-пошук і швидкий XML-розбір	Висока швидкість, точність	Менш зручний синтаксис
Selenium	Headless-браузер	Багато мовна підтримка	Емуляція користувачької поведінки	Може обробляти JS-сайти	Дуже повільний, ресурсомісткий
Puppeteer	Headless-браузер	JavaScript	Повний контроль DOM, скріншоти, PDF-експорт	Ідеальний для SPA	Висока складність налаштування
Cheerio	Бібліотека	Node.js	jQuery-подібний синтаксис для HTML	Швидкий і легкий	Не підтримує JS-рендеринг
Symfony DomCrawl	Бібліотека	PHP	Навігація DOM-	Простота інтеграції у	Не підходить

er			деревом, XPath	веб-додатки	для великих обсягів
Octoparse	SaaS- сервіс	—	Візуальний редактор сценаріїв парсингу	Не потребує кодування	Закритий код, платна ліцензія
ParseHub	SaaS- сервіс	—	Парсинг із рендеринго м JS, збереження у JSON/CSV	Інтуїтивний інтерфейс	Обмеження безкоштовн ої версії
Diffbot	Комерці йна AI- платфор ма	—	Автоматичн е розпізнаван ня структур сторінок через ML	Семантичне розуміння контенту	Дуже висока вартість

Основні проблеми та виклики веб-парсингу.

Зміни структури сайтів — навіть незначне редагування HTML-коду може призвести до некоректного вилучення даних.

Обмеження доступу — CAPTCHA, rate-limits, обмеження у robots.txt або Cloudflare-захист.[16]

Етичні та правові аспекти — необхідність дотримання політики конфіденційності та авторського права.[17]

Масштабування — при зборі даних із сотень джерел потрібна оптимізація запитів, балансування навантаження та черги завдань.

Якість даних — необхідність очищення, нормалізації та перевірки достовірності результатів.

Тенденції розвитку веб-парсингу

Сучасні дослідження у цій сфері спрямовані на:

інтелектуалізацію парсерів через застосування алгоритмів машинного навчання для автоматичного розпізнавання структури сторінок;

інтеграцію з великими мовними моделями (LLM) для семантичного аналізу контенту;[18]

розвиток безкодових рішень (no-code scraping), які роблять технологію доступною для нефахівців;[19]

створення гібридних систем, що поєднують DOM-, JS- та ML-парсинг у єдиній архітектурі.

Після етапу веб-парсингу, який забезпечує видобуток первинних фрагментів інформації з веб-ресурсів, наступним логічним кроком є агрегування даних.

Агрегування (англ. Data Aggregation) — це процес інтеграції, об'єднання та уніфікації інформації, отриманої з численних і різномірних джерел, у єдине структуроване сховище. Якщо веб-парсинг можна розглядати як "видобуток" інформаційної сировини, то агрегування — це її "збагачення", очищення та приведення до уніфікованого вигляду, придатного для подальшого аналізу.[2]

У контексті систем моніторингу освітнього простору, зокрема інформаційного поля Херсонського державного університету, агрегування відіграє ключову роль.

Одна й та сама новина може бути опублікована на десятках порталів, у блогах і соціальних мережах, тому необхідно виявити дублікати, нормалізувати формат і забезпечити єдність структури даних перед аналітичною обробкою.

Концептуальна модель ETL-процесу.

Теоретично, агрегування належить до класичних процесів ETL (Extract – Transform – Load), які є фундаментом побудови аналітичних систем і сховищ даних (Data Warehouse). [20]

Таблиця 1.3

Модель ETL-процесу

Етап	Назва	Короткий опис
Extract	Видобування	Отримання даних із відкритих джерел (реалізується парсером).
Transform	Трансформація	Очищення, нормалізація, уніфікація структури та виявлення дублювань.

Load	Завантаження	Збереження оброблених даних у централізовану базу або аналітичне сховище.
------	--------------	---

Найважливішим і найскладнішим етапом цього процесу є Transform, який забезпечує перехід від "сирих" парсингових результатів до єдиної аналітичної моделі даних.

Етап трансформації: уніфікація та нормалізація.

Уніфікація схеми (Schema Unification)

У відкритих джерелах структура інформації істотно відрізняється: новинні сайти, блоги, RSS-стрічки, API-сервіси та соціальні мережі подають дані в різних форматах.

Тому першочерговим завданням є проектування канонічної моделі даних (Canonical Data Model) — уніфікованої схеми, до якої трансформуються всі зібрані об'єкти.

Метою уніфікації є приведення різномірних структур до єдиного формату з типовими полями (заголовки, зміст, джерело, дата публікації, автор, категорія, посилання тощо).

Це дозволяє здійснювати подальший контент-аналіз незалежно від джерела походження інформації.

Очищення та нормалізація даних (Data Cleaning & Normalization)

Зібрані під час парсингу дані зазвичай містять "шум": HTML-теги, зайві символи, повтори, різні формати дат, некоректну кодування текстів.

Теоретичною основою очищення виступає концепція інформаційної нормалізації, що передбачає перехід від візуально-орієнтованого веб-контенту до чистого, уніфікованого тексту.

Основні завдання цього етапу:

Семантичне очищення — видалення HTML-розмітки, скриптів, рекламних блоків.[21]

Лінгвістична нормалізація — усунення варіацій у регістрі, пунктуації, символах.

Темпоральна уніфікація — стандартизація форматів дат і часу (наприклад, ISO-8601).

Таким чином, на виході формується єдина структурована база знань, де кожен об'єкт має уніфікований вигляд, незалежно від його початкового джерела.

Виявлення дублювань і нечіткої подібності.

Однією з ключових проблем агрегування є дедуплікація даних (Data Deduplication).

В умовах інформаційного середовища ЗВО одні й ті самі новини або пости часто копіюються з офіційних джерел і розповсюджуються на різних платформах, що призводить до штучного дублювання повідомлень.

Підходи до дедуплікації наведені у таблиці 1.4

Таблиця 1.4

Типи та методи дедуплікації з поясненням теоретичної суті

Тип	Метод	Теоретична суть
Точна дедуплікація	Хешування (MD5, SHA-256)	Визначення повних збігів на основі цифрового відбитку тексту.
Нечітка дедуплікація	Шингли, MinHash, Jaccard Similarity	Виявлення часткових збігів, коли текст відредагований, але зміст збережено.

Теоретично, нечітка дедуплікація належить до задач інформаційного співставлення (Information Matching) і спирається на концепцію схожості множин (міра Жаккара), що дозволяє визначати близькість текстів за змістом.[22]

Зберігання та індексація агрегованих даних.

Після очищення та уніфікації інформація завантажується у сховище даних.

Тип вибраної системи зберігання залежить від структури даних і потреб аналітики наведеної у таблиці 1.5.[23]

Таблиця 1.5

Класифікація сховищ даних за структурою та призначенням із прикладами

Тип сховища	Характеристика	Приклади
Реляційні БД (SQL)	Висока структурованість, підтримка складних запитів	MySQL, PostgreSQL
Документні БД (NoSQL)	Гнучкість структури, масштабованість	MongoDB, Elasticsearch
Аналітичні сховища (Data Warehouse)	Орієнтовані на агрегацію і звітність	ClickHouse, BigQuery

Для забезпечення швидкого пошуку в агрегованому корпусі текстів використовується обернений індекс (Inverted Index) — основна структура інформаційного пошуку (Information Retrieval).

Вона дозволяє зіставляти ключові терміни з документами, де вони зустрічаються, що радикально скорочує час пошуку.

Інформаційний пошук і моделі релевантності.

Теоретичний фундамент сучасних пошукових систем базується на моделях інформаційного пошуку (IR-моделях), які визначають, наскільки документ є релевантним пошуковому запиту.

Таблиця 1.6

Класифікація IR-моделей: принципи роботи, переваги та обмеження

Модель	Принцип	Переваги	Обмеження
Булева модель	Пошук точних збігів за логічними операторами AND/OR/NOT	Простота, точність у вузьких запитах	Не оцінює ступінь релевантності
TF-IDF	Залежність важливості слова від його частоти у документі та колекції	Об’єктивна оцінка значущості терміну	Ігнорує контекст і синонімію
Векторна модель (Semantic Search)	Перетворення текстів у вектори значень за допомогою мовних моделей (BERT, Word2Vec)	Семантичне розуміння, пошук за змістом	Висока обчислювальна складність

Векторний пошук дозволяє здійснювати семантичне групування документів за змістом, навіть якщо вони не містять ідентичних ключових слів.[24]

Для задач аналізу медіаполя університету це має вирішальне значення, оскільки забезпечує виявлення тематично подібних, але лексично різних повідомлень.

Теоретичне значення агрегування для систем моніторингу.

У науковому контексті агрегування виконує функцію інформаційної консолідації, створюючи основу для подальшого аналізу — класифікації, тонального оцінювання, тематичного моделювання.

Без уніфікації та нормалізації даних будь-яка аналітична система втрачає точність і репрезентативність результатів.

Теоретично, процес агрегування можна розглядати як модель багаторівневої абстракції даних, що включає:

Синтаксичний рівень — очищення та стандартизація структури.

Семантичний рівень — інтерпретація змісту, виявлення дублювань.

Аналітичний рівень — побудова зв'язків між сутностями (джерела, теми, автори, дати).

Таким чином, агрегування виступає не лише технічним, а й концептуальним мостом між процесами веб-парсингу та інтелектуального аналізу даних.

Процес агрегування даних є центральною ланкою у побудові автоматизованих систем збору та аналізу інформації з відкритих джерел.

Його теоретична сутність полягає у перетворенні неструктурованих фрагментів контенту, здобутих парсингом, на єдиний узгоджений інформаційний масив.

Використання сучасних підходів до нормалізації, дедуплікації та семантичного пошуку забезпечує високу точність і достовірність аналітичних результатів, що особливо важливо для систем моніторингу репутаційного та освітнього середовища університетів.

1.3. Алгоритми аналізу та класифікації текстової інформації

Після того як дані зібрані, агреговані й проіндексовані, настає етап аналітичної обробки. Саме він перетворює нечіткий текстовий потік на структуровані знання. У цьому підрозділі я розширю теоретичну базу, додам сучасні моделі (ChatGPT-5, Gemini 2.5 Pro, DeepSeek V3) і покажу, як їх можна інтегрувати в конвеєр NLP.

Методи обробки природної мови (NLP) — з урахуванням сучасних моделей.[25]

Обробка природної мови (Natural Language Processing, NLP) досліджує, як комп'ютери можуть «розуміти», інтерпретувати й генерувати людську мову. Однією з ключових проблем є неоднозначність (ambiguity) мовлення: омоніми, синоніми, еліпсис, складна синтаксика, сарказм тощо.

Стандартний конвеєр NLP (pipeline) складається з кількох етапів, кожен з яких уточнює результати попереднього:

Сегментація та токенизація — поділ тексту на речення та слова.

Морфологічний та синтаксичний аналіз[26]:

a) Part-of-Speech Tagging (PoS) — присвоєння частин мови (іменник, дієслово тощо)

b) Лематизація / стемінг — приведення слова до початкової форми[27]

Семантичний аналіз:

a) Розпізнавання іменованих сутностей (NER — Named Entity Recognition)[28]

b) Видобування відношень між сутностями (Relation Extraction)

Векторизація (Representation) — перетворення тексту чи слів у числові вектори (ембеддинги)

Модулі задачної обробки — наприклад, класифікація тематик, аналіз тональності, тематичне моделювання, кластеризація тощо.

Таблиця 1.7

Підходи до текстової обробки: принципи, застосування та обмеження

Підхід	Ідея	Недоліки / обмеження
Bag of Words (BoW), TF-IDF	Представити текст як вектор частот слів	Ігнорування порядку слів, семантики, контексту
Статичні ембеддинги (Word2Vec, GloVe)	Кожне слово має фіксований вектор у просторі	Всі вживання слова мають однаковий вектор (не враховується контекст)
Контекстуальні моделі (BERT, RoBERTa, XLM-R, LaBSE)	Розрахунок векторів залежно від контексту	Високі обчислювальні витрати, потреба в GPU / TPU
Сучасні великі моделі / LLM (ChatGPT-5, Gemini 2.5, DeepSeek V3 тощо)	Моделі з архітектурою трансформерів, часто агентного чи multi-модального типу, здатні “думати” (chain-of-thought), інструментально інтегрувати модулі	Вартість, ресурси, ризик галюцинацій, складне налаштування

Сьогодні аналітичні системи часто використовують великі контекстуальні моделі як ядро для багатьох задач: класифікація, резюмування, генерація пояснень, аналіз тональності та емоційності.

Морфологічний аналіз текстів для української мови.

Мова з розвиненою флексивністю (наприклад, українська) має типові форми, які змінюються згідно з відмінком, родом, числом. Це означає, що слово “університет”, “університету”, “в університеті” — це фактично одна й та сама лексема, але з різними закінченнями. Якщо цього не врахувати, система може рахувати їх як різні терміни.

Методи морфологічного аналізу:

Стемінг (Stemming) — просте відсікання закінчень (грубий, швидкий метод, але часто “урізає” корінь неправильно).

Лематизація (Lemmatization) — кращий підхід, який аналізує морфологію, граматику та словник, щоб повернути слово до стандартної форми (леми).

Для української часто використовують бібліотеки / інструменти: UDPipe, simplemma, mystem (російсько-орієнтований, може бути менш точним для української), або комбіновані підходи.

Морфологічний аналіз — ключова ланка для:
коректної агрегації згадок з різними формами слова,
поліпшення результатів тематичної класифікації,
зменшення “розпилу” даних (коли форми слова вважаються різними термінами).[29]

Класифікація текстів за тональністю (Sentiment / Emotion Analysis).

Аналіз тональності — це процес автоматичного визначення емоційного забарвлення тексту: позитивне, негативне, нейтральне. У контексті системи моніторингу ХДУ це допомагає автоматично ділять публікації за їх емоційним навантаженням.

Методи розвитку:

Лексиконні методи — використовують списки тональних слів із вагами (наприклад, “добре” → +1, “погано” → -1). Підходить для простих випадків, але не враховує контексту, іронію, заперечення тощо.

Класичні ML-методи — наприклад, SVM, Naïve Bayes, логістична регресія на основі TF-IDF-векторів або біграм/триграм.

Глибинні / контекстуальні моделі — BERT-подібні моделі, тонко підлаштовані (fine-tuned) для задачі аналізу тональності.

Aspect-based Sentiment Analysis (ABSA) — аналіз тональності не просто для всього документа, а для окремих аспектів / сутностей (наприклад, “якість освіти”, “умови проживання”, “взаємодія зі студентами” тощо).

З використанням сучасних LLM (ChatGPT-5, Gemini 2.5, DeepSeek V3) відкривається можливість:

тональність + емоційність: модель здатна не просто сказати “позитивно / негативно”, а оцінити емоційний спектр (радість, сум, злість, здивування тощо);

пояснення / інтерпретованість: модель може обґрунтувати, чому текст вважається негативним чи позитивним;

робота з довгими контекстами, з великою кількістю тексту, де тональність може змінюватися всередині тексту.

Сучасні великі мовні моделі у задачах аналізу та класифікації.

У таблиці 1.8 наведено короткий опис трьох актуальних моделей (на 2025 рік), які часто інтегрують в аналітичні NLP-системи.[30]

Таблиця 1.7

Основні характеристики та можливості актуальних LLM у задачах NLP-аналітики

Модель	Тип / архітектура	Основні можливості	Переваги / обмеження в задачах аналізу текстів
ChatGPT-5 (OpenAI)	Multimodal LLM, архітектура Transformer із “thinking” режимами (OpenAI)	Генерація текстів, розуміння контексту, аналіз тональності, резюмування, інструментальне виконання	Підтримка дуже довгих контекстів, інтеграція в API; ризик галюцинацій; дорогий обчислювально
Gemini 2.5 Pro (Google / DeepMind)	Multimodal thinking model, інтеграція із графікою, аудіо, інструментами (Google DeepMind)	Потужне міркування, робота з мультимодальними даними, великий контекст	Закрита модель; вартість; контроль над пам’яттю; суворість у fine-tuning
DeepSeek V3 (DeepSeek)	(гіпотетично) відкритий / напіввідкритий LLM із фокусом на ефективність та узгодженість	Ефективне використання ресурсів, баланс “думання” та продуктивності (DocsBot AI)	Може поступатися преміум-моделям у дуже складних завданнях; обмеження контексту; менша документація

Декілька ключових моментів:

Контекстне “мислення” (chain-of-thought / reasoning) — сучасні моделі часто активують внутрішні кроки “міркування” перед остаточним відповіддю, що покращує точність, особливо для завдань тональних оцінок, класифікації або логіки.

Інструментальна інтеграція (tool use / function calling) — моделі можуть викликати субмодулі / API (наприклад, словники, бази знань, зовнішні ентоніми) під час генерації відповіді.

Управління довгим контекстом — Gemini 2.5 Pro позиціонується як модель з великими вікнами контексту (long context)

DeepSeek V3 для деяких варіантів має контексти до 128K токенів

GPT-5, ймовірно, також підтримує великі контексти (офіційні дані — в системах OpenAI).

Вартість токенів — DeepSeek V3 є значно нижчим за ціною вхідного / вихідного токена порівняно з Gemini 2.5 Pro.

Галюцинації та безпечність — великі моделі ризикують генерувати неправдиву інформацію; крім того, потрібно валідувати, фільтрувати результати або комбінувати з класичними методами.

Ці моделі можна використовувати як ядро аналітичної системи:

Підлаштовувати (fine-tune) моделі під специфіку української мови / тематики ХДУ

Використовувати гібридні підходи: класичні моделі + LLM

Впроваджувати пояснювальні компоненти (model interpretability)

Аналізувати тональність, емоції, аспекти

Будувати семантичні графи (entities + relation + sentiment)

Побудова інформаційних моделей, тематичне моделювання та графи знань.

Кінцева мета аналітичного конвеєру — отримати інформаційні моделі, які відображають структуру знань, взаємозв'язки, динаміку тем і тональності.

Тематичне моделювання — LDA (Latent Dirichlet Allocation) був традиційним підходом, але має обмеження у контекстному розумінні. Сучасні підходи, як BERTopic, поєднують ембеддинги (з BERT чи аналогічної моделі) + кластеризацію (наприклад, HDBSCAN) для побудови тем у контексті.

Кластеризація / embedding clustering — тексти перетворюються у вектори (із великої моделі), після чого групуються (K-Means, HDBSCAN тощо).[31]

Графи знань (Knowledge Graphs) — на основі NER + Relation Extraction будується мережа сутностей, між якими встановлюються зв'язки. До графу можна приєднувати атрибути (наприклад, тональність, частота згадок, час).

Запити / аналітика над графом — система може відповідати на питання: “Які організації найбільше згадуються поруч із ХДУ у позитивному контексті?”, “Які теми в негативному тоні зростали за останній місяць?”.

Висновки до розділу 1

Інформаційний простір закладу вищої освіти є складною багаторівневою системою, у якій поєднуються офіційні комунікаційні платформи, медіа-ресурси, соціальні мережі, блоги, форуми та агрегатори новин. Його динамічність, гетерогенність і розподіленість зумовлюють потребу в автоматизованих підходах до збору, структурування та аналізу даних.

У сучасних умовах ручний моніторинг втрачає ефективність, тому центральне місце займають технології веб-парсингу та агрегування, що забезпечують вилучення, очищення, нормалізацію й уніфікацію даних із відкритих джерел. Ці процеси утворюють основу інформаційної інфраструктури, у якій кожен об'єкт набуває чітко визначеної структури та атрибутів, придатних для подальшого аналізу.

Перехід від зібраних текстів до аналітичних знань забезпечують алгоритми обробки природної мови (NLP). Вони включають морфологічний, синтаксичний і семантичний аналіз, векторизацію, класифікацію та

оцінювання тональності. Особливо важливим для української мови є морфологічний аналіз, який дозволяє об'єднувати усі форми слів у спільні лексеми, забезпечуючи точність тематичного й тонального моделювання.

Сучасна еволюція NLP пов'язана з появою великих мовних моделей (LLM), що працюють на архітектурі Transformer. Такі системи, як ChatGPT-5 (OpenAI), Gemini 2.5 Pro (Google DeepMind) та DeepSeek V3, поєднують обробку тексту, зображень і знань, здатні до контекстного “мислення” (chain-of-thought), виявлення емоцій, побудови семантичних зв'язків і класифікації повідомлень за настроєм чи тематикою. Використання подібних моделей дозволяє переходити від традиційного статистичного аналізу до глибокого семантичного розуміння текстів і формування аналітичних інсайтів у реальному часі.

Алгоритмічний конвеєр — від парсингу та агрегування до NLP-аналізу — створює підґрунтя для побудови інформаційних моделей і графів знань, які відображають структуру зв'язків між джерелами, темами, персоналіями й подіями. Це дає змогу не лише кількісно оцінювати інформаційне поле університету, а й виявляти його приховані тренди, ризики та репутаційні зміни.

Отже, теоретичні засади, розглянуті у цьому розділі, формують методологічну основу розробки автоматизованої системи моніторингу інформаційного простору Херсонського державного університету, орієнтованої на інтеграцію сучасних технологій збору, агрегування та інтелектуального аналізу текстових даних.

РОЗДІЛ 2

МЕТОДИКА РОЗРОБКИ ТА АНАЛІЗУ АВТОМАТИЗОВАНОЇ СИСТЕМИ ЗБОРУ ІНФОРМАЦІЇ

2.1. Обґрунтування вибору напрямку дослідження

В умовах цифровізації суспільства та динамічного розвитку інформаційного простору заклади вищої освіти потребують систематичного моніторингу свого інформаційного іміджу в мережі Інтернет. Потік новин, публікацій і згадок про університет формується у режимі реального часу, тому ручний аналіз є неефективним і не забезпечує повноти охоплення даних. Ці обставини зумовили необхідність створення автоматизованої системи збору, аналізу та обробки інформації про Херсонський державний університет.

Попередній аналіз показав, що більшість наявних інформаційно-аналітичних систем орієнтовані на бізнес-сегмент і не враховують специфіку освітніх установ. Водночас університети як соціально значущі інституції потребують інструментів, здатних виявляти публікації, що впливають на їхню репутацію, аналізувати тональність повідомлень і формувати аналітичні звіти для керівництва. Тому напрям дослідження, спрямований на створення інтелектуальної системи збору й аналізу текстів, є обґрунтованим як із наукового, так і з практичного погляду.

Для побудови системи обрано поєднання методів вебпарсингу, класифікації текстів і структурованого збереження даних. Збір інформації здійснюється автоматизовано з використанням регулярних виразів і бібліотеки Readability[32], що дає змогу виділяти основний контент сторінок. Обробка текстів та морфологічний аналіз виконуються за допомогою Python FastAPI[33] і моделі Gemini 2.5 Flash Lite, яка визначає тематику, полярність і контекст публікацій.

Система реалізована на основі вебфреймворку Laravel (PHP)[34], який забезпечує модульну структуру, принципи MVC та інтеграцію з базою даних

MySQL. Для візуалізації результатів використано Tailwind CSS[35], що сприяє створенню зручного адміністративного інтерфейсу.

Дослідження здійснювалося поетапно:

Перший етап (січень–лютий 2025 р.) полягав у пошуково-аналітичній роботі: аналізі джерел, визначенні вимог до системи, виборі технологічного стеку, розробці архітектури бази даних і логіки взаємодії між модулями.

Другий етап (березень–квітень 2025 р.) присвячено створенню та тестуванню модулів збору даних — парсера посилань і парсера повного контенту, що взаємодіють із базою даних і системою логування.

Третій етап (травень–червень 2025 р.) включав інтеграцію модуля морфологічного аналізу, реалізацію API-взаємодії між Laravel-сервісом і Python-обробником, а також тестування функцій аналізу тональності текстів і візуалізації результатів.

Четвертий етап (липень–серпень 2025 р.) був спрямований на оптимізацію системи, тестування продуктивності та підготовку аналітичних звітів, які демонструють роботу системи на реальних даних із новинних порталів.

Таким чином, вибір напрямку дослідження базується на необхідності створення технологічно комплексного рішення, яке поєднує сучасні інструменти автоматизації збору даних, мовної обробки й інформаційної аналітики. Це дозволяє перейти від хаотичного моніторингу Інтернет-публікацій до системного аналізу інформаційного середовища університету, що є вагомим внеском у розвиток інформаційних технологій у сфері освіти.

2.2. Методи збору та обробки даних

Збір даних з відкритих джерел є ключовим етапом функціонування системи. Основним методом є вебпарсинг (web scraping) — технологія автоматизованого вилучення структурованої інформації зі сторінок вебсайтів. Вебпарсер виконує HTTP-запити, отримує HTML-код сторінок і за допомогою DOM-парсингу або регулярних виразів виділяє потрібні елементи: заголовки, дати, зображення та основний текст.

Система реалізує дворівневу архітектуру збору:

Перший рівень — парсер посилань. Він шукає сторінки, що містять згадки про Херсонський державний університет, за допомогою регулярних виразів, які враховують морфологічні форми (“Херсонський”, “ХДУ”, “університет” тощо).

Другий рівень — парсер контенту. Цей модуль отримує HTML-сторінки за знайденими посиланнями, очищує контент від технічних елементів і виділяє основний текст за допомогою бібліотеки Readability.

Первинна обробка текстів включає:

нормалізацію кодування;

видалення HTML-тегів і спецсимволів;

токенізацію — поділ тексту на слова й речення;

видалення дублікатів через обчислення ембеддингів (модель “paraphrase-multilingual-MiniLM-L12-v2”)[36];

очищення й уніфікацію текстових полів.

Комунікація між Laravel-додатком і Python-сервісом здійснюється через REST API, що забезпечує гнучкість і масштабованість системи. Після обробки тексти зберігаються у базі MySQL для подальшого аналізу.

2.3. Порівняльна характеристика методів аналізу текстів

Після збору та попередньої обробки текстових даних одним із ключових завдань автоматизованої системи є аналіз змісту отриманої інформації. Цей етап передбачає не лише класифікацію повідомлень, а й оцінювання їхнього емоційного та семантичного забарвлення. Існує низка підходів до аналізу текстів, які відрізняються рівнем складності реалізації, точністю, потребою у навчальних даних та можливостями масштабування. У сучасній практиці виділяють три основні групи методів: статистичні, лінгвістичні та інтелектуальні (на основі великих мовних моделей).

Статистичні методи аналізу текстів базуються на частотному підході до обробки лінгвістичних даних. Основна ідея полягає у визначенні найчастіше вживаних лексем, словосполучень або тематичних груп за допомогою

обчислення їхніх частот. Найпоширенішими інструментами є частотний аналіз, TF-IDF (Term Frequency – Inverse Document Frequency) та побудова тематичних словників.

Такі методи відзначаються простотою реалізації, високою швидкістю обробки та мінімальними вимогами до обсягу навчальних даних. Вони дозволяють ефективно створювати статистичні профілі текстів, виявляти домінантні теми та ключові слова. Однак статистичний підхід не враховує контекстуальних зв'язків між словами, семантики та емоційного забарвлення тексту, що суттєво обмежує його застосування для завдань класифікації чи визначення тональності.

Лінгвістичні методи аналізу текстів базуються на структурному та контекстуальному розборі змісту. Вони включають морфологічний, синтаксичний і семантичний аналіз, що дозволяє встановити не лише частини мови, а й граматичні зв'язки між словами, виявити контекст значення та емоційні відтінки лексем.

Підхід передбачає використання словників позитивних та негативних слів, а також граматичних правил для розпізнавання тональності повідомлень. Наприклад, слова “успіх”, “розвиток” чи “досягнення” інтерпретуються як позитивні, а “проблема”, “загроза” чи “криза” — як негативні.

Основна перевага лінгвістичних методів полягає у глибшому розумінні тексту без потреби у великих навчальних вибірках. Водночас такі системи складні у підтримці, оскільки вимагають регулярного оновлення словникових баз і правил для коректного аналізу сучасної мови.

Методи на основі великих мовних моделей

Найсучасніший напрям у галузі аналізу текстів — це застосування великих мовних моделей (LLM, Large Language Models), які використовують глибокі нейронні мережі на базі архітектури Transformer. Такі моделі, як BERT, RoBERTa, GPT, Gemini, ChatGPT-5 тощо, дозволяють здійснювати

семантичний пошук, резюмування, класифікацію, а також аналіз емоційного забарвлення тексту.

У межах цього проєкту для аналізу полярності та емоційності повідомлень використовується модель Gemini 2.5 Flash Lite[37], розроблена компанією Google DeepMind. Ця модель належить до класу multimodal thinking LLM і здатна виконувати глибокий семантичний аналіз українських текстів, визначати їхню емоційну спрямованість та контекстну взаємодію понять.

Gemini 2.5 Flash Lite інтегрується через API, що дозволяє передавати текстові дані для автоматичного аналізу тональності (позитивна, негативна, нейтральна) та емоційних станів (радість, гнів, страх, смуток, подив, відраза). Перевага цього підходу полягає у високій точності, здатності враховувати контекст, синонімію, іронію та латентні смисли. Модель не потребує ручного налаштування словників і забезпечує стабільну якість результатів навіть для текстів зі складною лексичною структурою.

Недоліками LLM можуть бути висока вартість обчислень і залежність від зовнішнього API, однак їхня гнучкість і масштабованість роблять цей підхід найбільш перспективним для інтеграції у системи моніторингу інформаційного простору.

Таблиця 2.1

Порівняльна оцінка методів

Критерій	Статистичні методи	Лінгвістичні методи	LLM (Gemini 2.5 Flash Lite)
Складність реалізації	Низька	Середня	Середня–висока
Точність аналізу	Низька	Середня	Висока
Вимоги до даних	Мінімальні	Помірні	Залежить від API
Швидкість обробки	Висока	Середня	Середня
Врахування контексту	Ні	Частково	Так
Придатність для аналізу тональності	Обмежена	Добра	Відмінна
Інтерпретованість результатів	Висока	Висока	Частково

У межах цього дослідження розглядається можливість використання комплексного підходу, який може поєднувати статистичні та лінгвістичні методи для базового аналізу з подальшим уточненням результатів за допомогою великих мовних моделей. Такий гібридний підхід є ефективним у багатьох сучасних NLP-системах, оскільки поєднує простоту реалізації зі здатністю до глибокого контекстного розуміння.

Водночас у практичній частині даного проєкту для аналізу текстів про Херсонський державний університет застосовується виключно модель Gemini 2.5 Flash Lite. Вона забезпечує автоматичне визначення полярності та емоційності повідомлень, аналізує контекст, ідентифікує приховані смислові зв'язки та емоційні реакції у новинному контенті.

Завдяки своїй високій точності, підтримці української мови та можливості інтеграції через API, Gemini 2.5 Flash Lite є оптимальним рішенням для створення адаптивної системи моніторингу інформаційного простору університету.

Таким чином, проведене порівняння показало, що хоча статистичні та лінгвістичні методи залишаються ефективними для базового аналізу текстів, найбільш перспективним напрямом є використання великих мовних моделей, які забезпечують семантичну глибину, контекстну точність і здатність до самонавчання. У межах даного проєкту застосування моделі Gemini 2.5 Flash Lite дозволило досягти високого рівня точності при аналізі інформації про Херсонський державний університет, що підтверджує її ефективність як основного інструменту аналітичного модуля системи.

2.4. Методика проведення дослідження

Метою методики є створення автоматизованої системи, здатної збирати, аналізувати та структурувати інформацію про Херсонський державний університет. Основні завдання:

Виявлення та збір релевантних джерел.

Отримання повного контенту новинних матеріалів.

Аналіз текстів із визначенням тематики та тональності.

Збереження структурованих даних у базі.

Формування результатів у вигляді аналітичних звітів і візуалізацій.

Робота системи реалізується у п'ять етапів:

Ініціалізація дослідження — вибір активних джерел і параметрів парсингу.

Збір посилань — пошук згадок про ХДУ, перевірка унікальності.

Збір контенту — завантаження та очищення текстів статей.

Аналіз даних — визначення тематики, полярності, ключових слів.

Візуалізація результатів — відображення даних у вигляді таблиць і графіків.

Ця структура забезпечує послідовність обробки від моменту отримання даних до аналітичного узагальнення результатів.

2.5. Реалізація системи збору та аналізу інформації

Розроблена система реалізована на основі вебфреймворку Laravel 12 (PHP) з інтеграцією допоміжного сервісу FastAPI (Python) для морфологічного та семантичного аналізу. Її архітектура побудована за принципом MVC + Service Layer, що забезпечує чітке розділення логіки, високу масштабованість і простоту підтримки.

Основні компоненти системи:

Консольні команди — автоматизують процеси збору, аналізу й генерації даних.

Контролери — реалізують маршрутизацію, логіку вебінтерфейсу та інтерактивну взаємодію з користувачем.

Моделі — відповідають за зв'язок з базою даних MySQL, відображаючи її таблиці.

Сервіси — містять бізнес-логіку: парсинг, обробку, логування, створення ембеддингів, аналіз текстів і формування тегів.

Спостерігачі (Observers) — автоматично реагують на створення або оновлення об'єктів.

FastAPI-сервер (Python) — здійснює морфологічний аналіз текстів, обчислення ембеддингів і оцінку семантичної подібності.

Крім того, система підтримує планувальник (Scheduler), який дозволяє виконувати консольні команди у задані проміжки часу — наприклад, автоматично запускати парсер щодня або щогодини без участі користувача. Це забезпечує безперервний збір даних у фоновому режимі.

Система використовує чотири основні консольні команди Laravel, які відповідають за етапи збору, обробки та аналізу даних.

Кожна команда може виконуватися як вручну, так і автоматично через планувальник (Scheduler).

1. ConsoleParseLinks.php

Цей скрипт є відправною точкою процесу збору даних.

Під час виконання він:

обходить усі активні джерела з таблиці sources;

автоматично визначає тип парсингу (rss, html, use_browser) відповідно до параметрів джерела;

якщо джерело має RSS-стрічку[38], спочатку аналізує її структуру.

Алгоритм роботи:

Якщо RSS містить повний контент, система відразу парсить публікації з RSS і зберігає їх у таблиці nodes (без проміжного збереження у node_links).

Якщо RSS урізаний або відсутній, скрипт лише збирає посилання на новини, які зберігаються в node_links.

Для динамічних сайтів, де потрібен браузер, використовує JavaScript-скрипт server.js.

Таким чином, перший парсер поєднує дві функції — збір посилань і пряме збереження контенту з RSS, якщо це можливо.

2. ConsoleParseArticles.php

Ця команда використовується для парсингу повного контенту зі збережених у node_links посилань.

Вона застосовується лише для тих джерел, де RSS не містив повного тексту.

Контент завантажується та очищується за допомогою бібліотеки Readability, після чого статті зберігаються у таблиці nodes.

Після успішного збереження FastAPI викликається для перевірки, чи є стаття дублікатом.

Ця перевірка здійснюється через ChromaDB[39], де зберігаються векторні подання всіх статей.

Якщо нова публікація виявляється семантично подібною до існуючої, вона позначається як дублікат і не додається повторно.

3. ConsoleGenerateTags.php

Генерує тематичні мітки (теги) для кожної нової статті.

Аналізує частотність ключових слів і зберігає результати у таблицю tags.

Використовується у поєднанні з TagGeneratorService і LoggerService.

4. ConsoleAnalyzeNodes.php

Виконує семантичний і тональний аналіз контенту.

Використовує: AnalysisService, EmbeddingService, FastAPI.

FastAPI обчислює ембеддинги (вектори) текстів і визначає емоційну полярність (позитивна, нейтральна, негативна).

Результати записуються у таблиці node_sentiments і node_embeddings.

Контролери:

LinkParserController — керує запуском збору посилань і відображенням результатів.

ArticleParserController — відповідає за парсинг контенту зі сторінок.

SavedArticleController — відображає список статей із можливістю фільтрації за тегами, джерелами чи тональністю.

SentimentController — візуалізує результати аналізу емоційності у вигляді діаграм.

TagController — виводить теги, дозволяє здійснювати фільтрацію статей за категоріями.

SourceController — керує списком джерел, зокрема додає нові та перевіряє їх на наявність RSS або необхідність використання браузера.

ChromaController — взаємодіє з векторною базою Chroma DB для пошуку схожих статей.

LogController — показує журнали подій системи.

ExportController — виконує експорт даних у формат Excel (.xlsx).

TestParserController — тестовий модуль для перевірки дворівневого парсингу.

Система використовує реляційну базу даних MySQL у поєднанні з векторною базою ChromaDB, яка зберігає ембеддинги текстів для семантичного пошуку та виявлення дублікатів.

Таблиця 2.2

Таблиця моделей та структура баз даних

Модель / Таблиця	Призначення	Основні поля	Пов'язані модулі
Source / sources	Список джерел новин, автоматичне визначення типу парсингу (html, rss, use_browser)	id, name, url, has_rss, full_rss, use_browser, is_active	SourceController, SourceObserver, SourceService
NodeLink / node_links	Тимчасові посилання, зібрані під час парсингу	id, url, summary, hash, created_at	ArticleFinder, ConsoleParseLinks
Node / nodes	Повні тексти статей із метаданими	id, title, content, summary, date, hash, source_id	ArticleContentParser, ConsoleParseArticles
NodeSentiment / node_sentiments	Результати емоційного та тонального аналізу	node_id, sentiment, emotion, score	AnalysisService, FastAPI
NodeEmbedding /	Векторні подання	node_id,	EmbeddingService,

node_embeddings	статей, що зберігаються також у ChromaDB	chroma_id, vector, similarity	ChromaController
Tag / tags	Тематичні мітки статей	id, name, frequency	TagGeneratorService, TagController
Log / logs	Системні логи подій і помилок	id, module, message, created_at	LoggerService, LogController

Спостерігачі (Observers).

У системі реалізовано механізм спостерігачів (Observers), що дозволяє автоматично відстежувати зміни в базі даних і оновлювати статистику в панелі адміністратора. Основна мета цих компонентів — забезпечення точного підрахунку показників роботи системи без необхідності ручного втручання.

NodeObserver — контролює створення, оновлення та видалення записів у таблиці nodes. Під час створення нової статті автоматично збільшує лічильники загальної кількості матеріалів (total_nodes), кількості успішно розібраних статей (nodes_parsed) та відсутнього контенту (nodes_missing_content). При оновленні матеріалу перевіряє наявність поля content і коригує статистику в разі змін. Таким чином, забезпечується динамічне відстеження стану бази контенту.

NodeSentimentObserver — відповідає за ведення статистики емоційного та тонального аналізу текстів. Після створення або зміни записів у таблиці node_sentiments автоматично оновлює лічильники позитивних, негативних і нейтральних матеріалів, а також розподіл за емоційними станами (гнів, смуток, страх, відраза, радість, здивування, нейтральна). Завдяки цьому у реальному часі формується аналітична статистика щодо емоційного фону публікацій.

TagObserver — веде підрахунок загальної кількості тегів (total_tags). Під час створення нового тегу збільшує відповідний лічильник, а при

видаленні — зменшує його. Це дозволяє контролювати динаміку тематичної класифікації контенту.

SourceObserver — призначений для аналізу нових джерел у таблиці sources та автоматичного оновлення статистики активних і неактивних сайтів. Після додавання джерела перевіряє наявність RSS-потoku, потребу у headless-браузері, позначає відповідні прапорці (has_rss, use_browser, full_rss) і надалі синхронізує ці дані з лічильниками у Dashboard.

Усі спостерігачі взаємодіють із сервісом DashboardCounterService, який виконує інкремент або декремент значень у таблиці dashboard_counters. Це створює централізовану систему моніторингу, де кожна зміна у даних автоматично відображається у статистичних показниках.

Такий підхід дозволяє підтримувати актуальну аналітику в реальному часі, зменшує навантаження на сервер, підвищує точність статистики та забезпечує повну автоматизацію збору метаданих у межах системи.

Сервіси системи:

📁 /app/Services/Parser

ArticleFinder.php — здійснює пошук згадок про ХДУ за шаблонами (regex.txt), обходить сайти та збирає релевантні посилання.

ArticleContentParser.php — отримує HTML-сторінку, очищує її через Readability, витягує текст, дату, заголовок, зображення.

HashService.php — створює хеші SHA256 для перевірки унікальності статей.

📁 /app/Services/Logs

LoggerService.php — веде логування подій: помилки, успіхи, пропуски.

📁 /app/Services/Source

SourceService.php — керує списком джерел, виконує автоматичну перевірку наявності RSS, визначає тип парсингу (html, rss, use_browser), активує або деактивує джерела.

📁 /app/Services/Dashboard

DashboardCounterService.php — підраховує загальні показники для панелі адміністратора: кількість джерел, статей, тегів, аналізів.

📁 /app/Services

AnalysisService.php — надсилає тексти на API Gemini, визначає полярність і емоції.

EmbeddingService.php — створює векторні представлення текстів і зберігає їх у Chroma DB.

TagGeneratorService.php — генерує короткі тематичні теги на основі частотного аналізу.

📁 /app/Services/Scraper

server.js — Node.js-скрипт, який завантажує сторінки, що вимагають виконання JavaScript.

Взаємодія з FastAPI та regex.

FastAPI (app.py) — Python-сервер, який обробляє запити від Laravel, виконує:

формування ембеддингів (SentenceTransformers)[40],

оцінку схожості текстів.

Відповіді повертаються у форматі JSON.

regex.txt — файл, що містить регулярні вирази для пошуку згадок про ХДУ у різних морфологічних формах.

Алгоритм повного циклу роботи системи (на прикладі).

Додавання нового джерела.

Користувач створює новий запис у таблиці sources через вебінтерфейс. Після цього спрацьовує спеціальна функція, яка автоматично перевіряє:

чи має сайт RSS-потік.

чи потрібно використовувати браузер для збору даних.

Якщо RSS-потік існує і містить повний контент статей, система автоматично позначає поле `full_rss_content = true`. У такому випадку

подальший парсинг відбувається безпосередньо з RSS, що значно спрощує та пришвидшує процес збору інформації.

Якщо RSS відсутній або він містить лише анонси, система орієнтується на HTML-контент сайту, для чого може знадобитися використання браузера (наприклад, через headless-браузер або автоматизовані скрипти), щоб витягнути повні тексти статей.

Таким чином, процес адаптивний: система самостійно визначає оптимальний спосіб отримання контенту в залежності від наявності RSS і його повноти, що підвищує ефективність автоматизованого збору інформації.

Автоматичний запуск планувальника.

Для забезпечення регулярного та автономного збору інформації у системі використовується планувальник Laravel Scheduler. Він дозволяє у встановлений час автоматично викликати консольні команди через інтерфейс командного рядка `php artisan`.

Збір посилань.

`ConsoleParseLinks` обходить активні джерела, обрані як `isActive = 1`, та визначає тип парсингу для кожного сайту. Далі відбувається виклик відповідних функцій сервісів:

Якщо RSS-потік присутній і містить повний текст статей (`full_rss_content = true`), `ConsoleParseLinks` використовує функціонал `ArticleContentParser`, щоб безпосередньо отримати та зберегти повний контент у таблицю `nodes`.

Якщо RSS містить лише анонси або відсутній, викликається `ArticleFinder`, який збирає посилання на статті для подальшого парсингу повного тексту іншим етапом.

Для динамічних сторінок, де контент генерується через JavaScript і не доступний у початковому HTML або RSS, задіюється `server.js` із використанням `Puppeteer`. Це дозволяє імітувати браузер, виконати JavaScript та отримати повний DOM сторінки, після чого видобуваються потрібні посилання.

Таким чином, система автоматично адаптує метод збору посилань залежно від доступності повного контенту та типу сторінки, забезпечуючи ефективний і гнучкий процес парсингу.

Збереження результатів.

Після збору посилань вони автоматично зберігаються у таблицю `node_links`. Для кожного запису фіксуються такі дані:

URL статті – повна адреса ресурсу.

Тип парсингу – визначається залежно від джерела та доступності контенту: `rss` або `html`.

Використання браузер (use_browser) – позначка, чи був застосований headless-браузер для отримання контенту.

Статус обробки (parsed) – показує, чи була стаття вже парсена повністю (0 – ще не оброблено, 1 – оброблено).

Кількість спроб (attempts) – відстежує кількість спроб парсингу даного посилання.

Остання помилка (last_error) – зберігає текст останньої помилки у разі невдалої обробки.

Такий підхід забезпечує контроль над процесом збору інформації, дозволяє відновлювати обробку неуспішних записів та підтримує ефективну роботу системи автоматизованого моніторингу.

Збір контенту.

Збір повного тексту статей здійснюється за допомогою `ConsoleParseArticles`, яка завантажує сторінки за посиланнями, збереженими у таблиці `node_links`. Для виділення основного змісту використовується `Readability`, що дозволяє очистити HTML-сторінку від реклами, меню та інших сторонніх елементів, залишаючи лише релевантний текст статті.

Щоб забезпечити унікальність контенту та запобігти дублюванню, застосовується headless-браузер (Chrome) у поєднанні з векторною базою даних Chrome DB. Кожен текст перетворюється на векторне представлення (ембедінг) і зберігається у таблиці `node_embeddings`, після чого виконується

перевірка схожості з уже існуючими ембедингами за допомогою косинусної схожості[41]. Такий підхід дозволяє виявляти дублікати або тексти з високим ступенем подібності до існуючих записів перед остаточним збереженням у базі.

Для інтеграції з системою використовується FastAPI на Python, який приймає тексти, очищає їх від HTML, розбиває на чанки та порівнює з існуючими ембедингами у Chrome DB. У випадку відсутності дубліката, новий ембединг додається у базу, а Laravel-сервіс EmbeddingService забезпечує синхронізацію записів між node_embeddings та Chrome DB по унікальному ідентифікатору. Це гарантує цілісність даних, дозволяє зв'язати контент із відповідним посиланням і результатами перевірки на дублювання, а також забезпечує ефективне керування процесом збору, збереження та аналізу інформації.

Для оцінки тональності та емоційного забарвлення зібраних статей у системі використовується ConsoleAnalyzeNodes, яка обробляє записи з таблиці nodes. Основна мета аналізу — визначення сентименту (позитивна, негативна, нейтральна) та емоційного стану тексту (гнів, смуток, відраза, страх, радість, здивування або нейтральна).

У процесі аналізу система використовує API моделі Gemini 2.5 Flash Lite через спеціальний сервіс Laravel (AnalysisService). Текст для аналізу береться із поля summary статті, або, якщо воно відсутнє або недостатньо інформативне, використовується частина основного контенту (content).

Виклик моделі Gemini здійснюється з попередньо визначеними системними інструкціями, які задають правила визначення тональності та емоцій, і обов'язкове повернення результату у вигляді JSON[42] у форматі українською мовою.

Після отримання відповіді від моделі результати зберігаються у таблиці node_sentiments. Усі операції виконуються у рамках транзакції бази даних, що забезпечує цілісність даних.

Такий підхід дозволяє автоматизовано оцінювати емоційне та тональне забарвлення текстів, зберігати структуровані результати для подальшого аналізу та забезпечує надійний контроль за процесом обробки інформації.

Генерація тегів.

Для автоматичного створення тематичних позначок у системі використовується `ConsoleGenerateTags`, яка обробляє записи з таблиці `nodes`. Теги генеруються на основі тексту статей, беручи за основу поле `summary` або, у разі його відсутності, частину основного контенту (`content`).

Процес генерації реалізовано через API моделі Gemini 2.5 Flash Lite, яка аналізує текст та пропонує до 5 релевантних тегів у форматі JSON. Кожен тег зберігається у таблиці `tags`, а зв'язок між статтею та тегами підтримується через проміжну таблицю. При цьому враховуються правила:

Теги складаються з одного або максимум двох слів.

Використовуються іменники або короткі описові фрази, без пунктуації та загальних слів типу «новини» чи «Україна».

Запобігається дублювання тегів для однієї статті.

Роль тегів у проєкті

Теги виконують кілька важливих функцій:

Категоризація контенту – дозволяють групувати статті за темами, що спрощує їх пошук і фільтрацію.

Покращення навігації – користувачі системи можуть швидко знаходити матеріали за ключовими темами через тегові хмари або фільтри.

Аналіз інформаційного простору – теги допомагають автоматично визначати основні теми, тенденції та спрямованість контенту, що важливо для моніторингу інформаційного поля Херсонського державного університету.

Підготовка до подальшої аналітики – тематичні мітки полегшують класифікацію, побудову статистики та трендів по категоріях.

Таким чином, генерація тегів є ключовим елементом автоматизованої системи, що дозволяє структурувати зібраний контент та підвищує ефективність подальшого аналізу і обробки інформації.

Відображення у вебінтерфейсі.

У системі передбачено зручний вебінтерфейс для перегляду та управління зібраним контентом. Користувачі можуть:

Переглядати статті – відображається заголовок, дата публікації, повний текст, а також додаткові метадані (посилання на джерело, теги, тональність та емоційне забарвлення).

Редагувати записи – доступна можливість змінювати текст статті, її короткий опис (summary), а також редагувати або додавати теги для уточнення тематичних категорій.

Видаляти статті – реалізовано функціонал безпечного видалення записів із бази даних із відновленням цілісності зв'язків (теги, результати аналізу, ембединги).

Експорт і візуалізація.

Система забезпечує можливість експорту зібраної та обробленої інформації у зручному форматі для подальшого використання. Через ExportController користувач може формувати звіт у форматі Excel, який включає:

Основні дані статей (заголовок, дата, посилання, summary, повний текст).

Результати аналізу тональності та емоційного забарвлення (sentiment та emotion).

Згенеровані тематичні теги та інші метадані.

Крім табличного звіту, вебінтерфейс підтримує візуалізацію даних у вигляді графіків, що базуються на результатах аналізу текстів. Наприклад:

Розподіл статей за тональністю (позитивні, негативні, нейтральні).

Частота появи основних емоцій (гнів, смуток, радість тощо).

Тематичні тренди за тегами або категоріями.

Візуалізація здійснюється динамічно у вебінтерфейсі, що дозволяє користувачам швидко оцінювати інформаційний простір, визначати ключові теми та тенденції, а також аналізувати результати роботи системи у наочній формі.

Висновки до розділу 2

Розроблена методика створення автоматизованої системи збору, обробки та аналізу інформації про Херсонський державний університет забезпечує комплексний підхід до моніторингу інформаційного простору. Використання дворівневої архітектури збору даних, інтеграція вебпарсингу, бібліотеки Readability та моделі Gemini 2.5 Flash Lite дозволяє ефективно збирати релевантний контент, проводити його морфологічний і семантичний аналіз, визначати тональність і емоційний стан текстів.

Система демонструє високу гнучкість завдяки адаптивному підходу до джерел інформації (RSS, HTML, динамічні сторінки), автоматизації перевірки унікальності контенту та генерації тематичних тегів. Для забезпечення перевірки дублювання та семантичної подібності текстів використовується FastAPI на Python у поєднанні з векторною базою ChromaDB і моделлю SentenceTransformers/paraphrase-multilingual-MiniLM-L12-v2, що дозволяє порівнювати ембедінги текстів і автоматично ідентифікувати повторювані матеріали.

Взаємодія між компонентами через API та планувальник забезпечує безперервність збору даних і надійність обробки. Наявність вебінтерфейсу та можливостей візуалізації результатів дозволяє оперативно оцінювати інформаційний простір університету, формувати аналітичні звіти та відстежувати тенденції в публікаціях.

Таким чином, запропонована методика забезпечує ефективне й системне управління інформацією, підвищує точність аналізу текстів і створює основу для подальшого розвитку інтелектуальних систем моніторингу в освітній сфері, використовуючи сучасні інструменти машинного навчання та обробки природної мови.

ВИСНОВКИ

У результаті виконання дипломного проєкту було створено автоматизовану систему збору, обробки та аналізу інформації про Херсонський державний університет, яка забезпечує системний моніторинг інформаційного простору та сприяє підвищенню ефективності управління репутаційними процесами в онлайн-середовищі.

Система побудована на основі вебфреймворку Laravel (PHP) із інтеграцією FastAPI (Python), що дозволило реалізувати поєднання сучасних інструментів веброзробки та машинного аналізу текстів. Дворівнева архітектура збору даних (парсер посилань і парсер контенту) забезпечує гнучкість і масштабованість процесу збору, дозволяючи ефективно обробляти як RSS-потoki, так і динамічні вебсторінки.

Для очищення та структурування контенту застосовано бібліотеку Readability, а для перевірки унікальності матеріалів — векторну базу ChromaDB у поєднанні з моделлю SentenceTransformers/paraphrase-multilingual-MiniLM-L12-v2, що забезпечує визначення семантичної подібності текстів. Аналіз емоційного та тонального забарвлення здійснюється за допомогою моделі Gemini 2.5 Flash Lite, яка дозволяє точно визначати полярність повідомлень і виявляти приховані емоційні стани українськомовних текстів.

Реалізований модуль автоматичної генерації тематичних тегів підвищує рівень структурованості зібраних даних, полегшує їх фільтрацію та сприяє виявленню інформаційних тенденцій. Завдяки інтеграції вебінтерфейсу з елементами візуалізації користувач має можливість переглядати, редагувати та аналізувати матеріали в зручній аналітичній формі.

Використання планувальника та консольних команд забезпечує повну автоматизацію процесів збору, обробки й аналізу даних у режимі реального часу. Такий підхід гарантує безперервність функціонування системи, стабільність результатів і високу якість отриманих даних.

Розроблена система поєднує сучасні методи вебпарсингу, обробки природної мови та машинного навчання, створюючи технологічно комплексне рішення для моніторингу інформаційного простору освітніх установ. Її впровадження підтверджує практичну ефективність підходу й відкриває перспективи для подальшого розвитку інтелектуальних аналітичних систем у сфері освіти.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Moaiad Ahmad Khder Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application *International Center for Scientific Research and Studies* C.145-165 URL: <http://www.icsrs.org/Volumes/ijasca/2021.3.11.pdf>
2. Бородій І. І., Парамуд Я. С., Сав'як В. В. Принципи побудови програмної системи формування агрегованих даних. *Національний університет "Львівська політехніка", кафедра електронних обчислювальних машин*. 2018. С. 25-32. URL: <https://science.lpnu.ua/sites/default/files/journal-paper/2019/sep/18526/182073vis905ksmzvichniy-25-32.pdf>
3. Шевчук М. О. Сучасні виклики і загрози в сфері інформаційної безпеки держави *Науковий збірник «Актуальні проблеми вітчизняної юриспруденції»* С.160-165 URL:http://apnl.dnu.in.ua/6_2024/27.pdf
4. Рябишев, О. В. Дослідження методів аналізу емоційного забарвлення тексту українською мовою *Репозиторій Харківського національного університету радіоелектроніки*. 2021. С 7-48 URL: <https://openarchive.nure.ua/bitstreams/5d8b1654-43f8-448d-b9cf-1762ddb716c3/download>
5. Diana Honcharenko Інструменти для моніторингу згадок про бренд URL:<https://hostpro.ua/blog/ua/brand-mentions-tools/>
6. Самчук В. Академічна доброчесність і репутація університету *Open Journal Systems*. 2023. С.138-151 URL: <https://journals.knute.edu.ua/scientia-fructuosa/article/view/1894>
7. Américo Rio, Fernando Brito e Abreu PHP code smells in web apps: survival and anomalies URL:<https://arxiv.org/pdf/2101.00090>
8. Закревська Л. М., Данкеєва О. М. Ефективність рекламних та PR-кампаній під час реалізації антикризових заходів *Modern Economics*. 2024. № 48. С. 31–37

9. Загородній Д. О., Войцеховська О. В. Автоматизована система збору та обробки цифрової інформації *репозитарій Вінницького Національного Технічного Університету*. URL:<https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/38176/134835.pdf>
10. Dan Jurafsky, James H. Martin *Speech and Language Processing Stanford University*. 2025. С.3-566 URL: https://web.stanford.edu/~jurafsky/slp3/ed3book_aug25.pdf
11. Cyber Huntress Web Scraping for OSINT: Methods and Tools. 2024. URL: <https://medium.com/@thecyberhuntress/web-scraping-for-osint-methods-and-tools-eacaff81ca40>
12. Диханов С. В., Гук Н. А. Аналіз структури веб-ресурсів з використанням об'єктної моделі *Eastern-European Journal of Enterprise Technologies*. 2022. С. 6-12 URL: <https://journals.uran.ua/eejet/article/view/265961/262361>
13. Irfan Darmawan, Muhamad Maulana, Rohmat Gunawan, Nur Widiyasono Evaluating Web Scraping Performance Using XPath, CSS Selector, Regular Expression, and HTML DOM With Multiprocessing Technical Applications *International journal on informatics visualization*. 2022. С.904-909 URL:<https://joiv.org/index.php/joiv/article/download/1525/576>
14. Науковий журнал «Вчені записки ТНУ імені В.І. Вернадського. Серія: Технічні науки» зареєстровано відповідно до Рішення Національної ради України з питань телебачення і радіомовлення № 1136 від 11.04.2024 року. Том 36 (75) № 2 2025 Частина 2 С.31
15. Khushi Pandey, Chinmayee Tale, Tanmayi Yere, Rajnandini Rajeshirke, Rupali Jadhav Web Scraping: Leveraging the Power of Python, APIs, and Automation *International Journal of Novel Research and Development* С.383-388
16. Cloudflare Docs Cloudflare Turnstile URL: <https://developers.cloudflare.com/turnstile/>
17. Megan A. Brown, Andrew Gruen, Gabe Maldoff, Solomon Messing, ZeveSanderson, Michael Zimme Web Scraping for Research: Legal, Ethical,

Institutional, and Scientific Considerations *ResearchGate*. 2024. C.3-42 URL: https://www.researchgate.net/publication/385442945_Web_Scraping_for_Research_Legal_Ethical_Institutional_and_Scientific_Considerations

18. Erika Petersen and Christopher Potts. 2023. Lexical Semantics with Large Language Models: A Case Study of English “break”. In *Findings of the Association for Computational Linguistics: EACL 2023*, C. 490–511, Dubrovnik, Croatia. Association for Computational Linguistics.

19. Felix Vemmer Power Automate Web Scraping: A Comprehensive Guide. 2024. URL: <https://nocodecraper.com/blog/power-automate-web-scraping>

20. Копійка В.Г. Розробка фреймворку для масштабованої etl-системи *електронний архів Національного університету "Києво - Могиллянська академія"*. 2024. С.8-87 URL: <https://ekmair.ukma.edu.ua/handle/123456789/32295>

21. Стрельченко Г. Г. Веб-додаток класифікації текстових новин з інтернет-ресурсів на основі методів машинного навчання *Електронний архів наукових та освітніх матеріалів КНУ ім. Ігоря Сікорського*. 2023. С.8-71 URL: <https://ela.kpi.ua/server/api/core/bitstreams/132f80a5-2e6d-443a-9227-ff62e81c2217/content>

22. Русин Б. П., Погрелюк Л. В., Висоцька В. А., Осипов М. М. Метод дедублікації та розподілу даних у хмарних сховищах під час резервного копіювання даних *Academic Journals and Conferences*. 2019. С.2-10 URL <https://science.lpnu.ua/sites/default/files/journal-paper/2019/nov/19752/191675maket2-3-14.pdf>

23. Dheeraj Kumar Bansal Enterprise data warehouses: types, benefits, and considerations *International Research Journal of Modernization in Engineering Technology and Science*. 2025. C.1558-1598

24. Siddharth Pratap Singh Vector search in the era of semantic understanding: a comprehensive review of applications and implementations *International Journal of Computer Engineering and Technology* C.1794-1804 URL:

https://iaeme.com/MasterAdmin/Journal_uploads/IJCET/VOLUME_15_ISSUE_6/IJCET_15_06_153.pdf

25. Yanhan Chen, Hanxuan Wang, Kaiwen Yu, Ruoshui Zhou Artificial Intelligence Methods in Natural Language Processing: A Comprehensive Review *Darcy & Roy Press*. 2024. C.545-549 URL: <https://drpress.org/ojs/index.php/HSET/article/view/18422/17958>

26. Вінтонів М. О. Динаміка методів і напрямів синтаксичного аналізу речення *Інституційний репозиторій Київського столичного університету імені Бориса Грінченка*. 2020. C.145-149 URL: https://elibrary.kubg.edu.ua/id/eprint/33823/1/M_Vintoniv_DMNSAR_V_33_1_IF_KUBG.pdf

27. Rabab Hafeez Muhammad Waqas Anwar, Muhammad Hasan Jamal, Tayyaba Fatima, Julio César Martínez Espinosa, Luis Alonso Dzul López, Ernesto Bautista Thompson, Imran Ashraf Contextual Urdu Lemmatization using Recurrent Neural Network Models *Open Access Journals*. 2023. C.1-20 URL: <https://www.mdpi.com/2227-7390/11/2/435>

28. Марченко О.О. Машинно-навчальні методи розпізнавання іменованих сутностей тексту *Науковий журнал «Проблеми програмування»*. 2023. C.150-156 URL: <https://pp.isoftware.kiev.ua/index.php/ojs1/article/view/190/185>

29. Bakhtiyor Mengliev, Shohida Shahabitdinova, Shahlo Khamroeva, Shakhnoza Gulyamova, Adiba Botirova The morphological analysis and synthesis of word forms in the linguistic analyzer *Journal of Language and Linguistic Studies*. 2021. C.559-563 URL: <https://www.jlls.org/index.php/jlls/article/view/2615/816>

30. Tyler A. Chang, Benjamin K. Bergen Language Model Behavior: A Comprehensive Survey *ACL Anthology*. 2024. C.294-323 URL: <https://aclanthology.org/2024.cl-1.9.pdf>

31. Zainab Abdul Wahid Salman Text Summarizing and Clustering Using Data Mining Technique *Al-Mustansiriyah Journal of Science*. 2024. C.58-63 URL: <https://mjs.uomustansiriyah.edu.iq/index.php/MJS/article/view/1195/649>
32. Chun Fei Lung Comparing algorithms for extracting content from web pages *Chuniversityit*. 2024. URL: <https://chuniversityit.nl/papers/comparison-of-web-content-extraction-algorithms>
33. Hasan Rustamovich Rasulov Fastapi: a modern web framework for python *International journal of scientific researchers (IJSR)*. 2024. C.727-729 URL: <https://worldlyjournals.com/index.php/IJSR/article/view/6597/10474>
34. Yuri Ariyanto Laravel framework and native PHP: Comparison in the creation of rest API *Matrix : Jurnal Manajemen Teknologi dan Informatika*. 2024. C.56-71 URL: <https://ojs2.pnb.ac.id/index.php/MATRIX/article/view/1413/890>
35. Fhadiel Putra Perdana¹, Edi Supratman, Dhimas Rosanto Saputra Designing a Modern Web Interface with Vue.js and Tailwind for University Information System *Brilliance: Research of Artificial Intelligence*. 2024. C.956-962 URL: <https://itscience.org/jurnal/index.php/brilliance/article/view/5409/4177>
36. Moneerh Aleedy, Fatma Alshihri, Souham Meshoul, Maha Al-Harhi, Salwa Alramlawi, Badr Aldaihani, Hadil Shaiba, Eric Atwell Designing AI-powered translation education tools: A framework for parallel sentence generation using SauLTC and LLMs *PeerJ Journals*. 2024. URL: <https://peerj.com/articles/cs-2788>
37. Gemini 2.5 Flash-Lite (Non-reasoning): Intelligence, Performance & Price Analysis *Artificial Analysis*. URL: <https://artificialanalysis.ai/models/gemini-2-5-flash-lite>
38. Rajiv Nariani RSS Feeds from STM Databases: Innovative Possibilities *Issues in Science and Technology Librarianship*. 2010. URL: <https://journals.library.ualberta.ca/istl/index.php/istl/article/view/2533/2444>
39. Aaron Ricadela What Is Chroma? An Open Source Embedded Database. 2024. URL: <https://www.oracle.com/bz/database/vector-database/chromadb/>

40. Bachriah Fatwa Dhini Development of an Automated Scoring Model Using SentenceTransformers for Discussion Forums in Online Learning Environments Portal of Croatian scientific and professional journals. 2022. C.85-97 URL: <https://hrcak.srce.hr/file/446378>

41. Permata Sari, L. Cosine Similarity-based Plagiarism Detection on Electronic Documents *Journal of Computer Science Application and Engineering (JOSAPEN)* . 2023. C.45-48 URL: <https://journal.lenterailmu.com/index.php/josapen/article/view/14/20>

42. Кучеренко, Є. А., Кулаковська, І. В. Аналіз методів та алгоритмів обробки неструктурованих текстових даних на основі технології JSON *Наукова періодика України*. 2024. C.10-18 URL: <https://journals.uran.ua/tarp/article/view/306435/298344>

43. Lian J. Research on Data Quality Analysis Based on Data Mining *Zeus Press*. 2025. C.2-6 URL: <https://journals.zeuspress.org/index.php/CAI/article/view/218/193>