МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій
Кафедра «Комп'ютерні системи та мережі»

ВЕРТМАН МИКОЛА АНДРІЙОВИЧ
Група КНТ-514м

КОМП'ЮТЕРНА СИСТЕМА ДЛЯ ПЕРЕВІРКИ ТА ВИПРАВЛЕННЯ ПОМИЛОК В ТЕКСТІ З ВИКОРИСТАННЯМ ШТУЧНОГО ІНТЕЛЕКТУ

АВТОРЕФЕРАТ

дипломної роботи на здобуття освітньо-кваліфікаційного рівня
«магістр» 123 Комп'ютерна інженерія
освітньої програми Комп'ютерні системи та мережі

2025 р.

Дипломна робота є рукопис.

Робота виконана в національному університеті «Запорізька політехніка», на кафедрі комп’ютерних систем та мереж

Керівник	кандидат технічних наук, доцент **Скрупський Степан Юрійович,** Національний університет «Запорізька політехніка», доцент кафедри комп'ютерних систем та мереж
Офіційний рецензент:	**ЩЕРБАКОВ Володимир Іванович,** директор НВФ "Бізнес Комп'ютер Сервис"

Захист відбудеться "26" грудня 2025 р.

Секретар екзаменаційної комісії, доцент кафедри комп’ютерних систем та мереж Т.В. Голуб

ЗАГАЛЬНА ХАРАКТЕРИСТИКА РОБОТИ

Актуальність теми. Стрімке зростання обсягів цифрового контенту висуває високі вимоги до якості письмової комунікації. Традиційні методи ручної перевірки є трудомісткими, а існуючі автоматизовані системи не завжди враховують морфологічні особливості української мови. Перспективним шляхом вирішення цієї проблеми вбачається використання великих мовних моделей (LLM), адаптованих для глибокого аналізу тексту.

Метою роботи є підвищення ефективності автоматизованої перевірки та виправлення текстів українською мовою шляхом розробки комп'ютерної системи на основі LLM із використанням методу гранулярного виявлення помилок.

Наукова новизна очікуваних результатів полягатиме в розробці підходу, що інтегрує LLM з оптимізованим промпт-інженірингом та багаторівневою обробкою даних. Планується застосувати метод гранулярного виявлення помилок (Granular Error Detection), який, на відміну від існуючих рішень, дозволить ідентифікувати кожну помилку окремо зі збереженням контексту, що має суттєво підвищити точність аналізу.

Практичне значення роботи полягатиме у створенні клієнт-серверного вебзастосунку для корекції помилок у режимі реального часу. Проєктована архітектура має забезпечити стабільність при високих навантаженнях, що зробить систему придатною для впровадження в освітній та бізнес-сферах. Очікується, що розроблена система перевершить існуючі аналоги за показниками повноти пошуку та швидкодії.

Мета і завдання дослідження. Метою магістерської роботи є підвищення ефективності та швидкості автоматизованої перевірки текстів шляхом розробки та дослідження комп'ютерної системи на основі великих мовних моделей із використанням методу гранулярного виявлення помилок (Granular Error Detection) та структурованого виводу даних.

Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Провести комплексний аналіз предметної області, існуючих методів та алгоритмів автоматизованої перевірки текстів (від класичних алгоритмічних підходів до сучасних нейромережевих

архітектур), виділити їх переваги та недоліки, обґрунтувати доцільність використання LLM для вирішення поставленої задачі;

2. Розробити багаторівневу архітектуру обробки помилок та механізм забезпечення надійності системи, який включає валідацію вхідних даних, обробку помилок API, стратегії повторних запитів та структурований вивід результатів;

3. Розробити методику гранулярного виявлення помилок та визначити стратегії промпт-інженірингу (Few-Shot Learning, Chain-of-Thought) для забезпечення високої якості аналізу та нівелювання "галюцинацій" моделей;

4. Спроєктувати та програмно реалізувати клієнт-серверну вебсистему з використанням сучасного технологічного стеку (Next.js, React, Python Flask), що забезпечує ефективну взаємодію з великими мовними моделями через API;

5. Провести експериментальне тестування розробленої системи на спеціалізованому датасеті, порівняти її ефективність з існуючими аналогами (LanguageTool, isgen.ai) за критеріями точності, повноти пошуку та швидкодії.

Об’єктом дослідження – процес автоматизованої перевірки та корекції текстових даних українською мовою з використанням технологій штучного інтелекту.

Предмет дослідження – методи та засоби підвищення ефективності виявлення граматичних, стилістичних та пунктуаційних помилок з використанням великих мовних моделей, зокрема архітектурні рішення для забезпечення надійності та швидкодії системи.

Методи дослідження. Для вирішення поставлених завдань використано: методи теоретичного аналізу архітектур трансформерних нейронних мереж та механізмів уваги (attention mechanisms); методи промпт-інженірингу (Few-Shot Learning, Chain-of-Thought Prompting); методи об'єктно-орієнтованого проєктування та розробки сучасних односторінкових вебзастосунків (Single Page Applications); порівняльний експеримент для оцінки ефективності різних конфігурацій системи; статистичні методи обробки результатів тестування (розрахунок метрик Precision, Recall, F-measure).

Наукова новизна отриманих результатів:

Розробка багаторівневої архітектури обробки помилок, яка, на відміну від існуючих підходів, поєднує механізми валідації на рівні клієнта, сервера та API, що забезпечує стабільну роботу системи навіть при нестабільних з'єднаннях або збоях зовнішніх сервісів. Запровадження методу гранулярного виявлення помилок (Granular Error Detection), який дозволяє аналізувати текст на різних рівнях деталізації -- від окремих слів та словосполучень до речень та абзаців -- що підвищує точність класифікації помилок та зменшує кількість хибних спрацювань.

Застосування комбінованих стратегій промпт-інженірингу (Few-Shot Learning для надання контекстних прикладів, Chain-of-Thought для поетапного міркування моделі) та структурованого виводу даних через JSON Schema, що дозволило вперше систематизувати ефективність різних конфігурацій великих мовних моделей (GPT-4o, GPT-3.5 Turbo) для задач перевірки текстів українською мовою. Експериментально встановлено, що використання Chain-of-Thought підвищує якість аналізу на 2% (за метрикою F0.5) порівняно з базовим промптингом, при цьому час обробки зростає лише на 100-150%.

Запропонована методика дозволяє досягти балансу між якістю аналізу та швидкодією: для текстів обсягом до 1600 символів час обробки не перевищує 5 секунд при точності виявлення помилок понад 80%, що є суттєвим покращенням порівняно з існуючими рішеннями..

Практичне значення отриманих результатів:

Розроблене програмне забезпечення може бути використане в різних галузях: автоматизація процесів редагування текстів у медіа та видавництвах, освітня сфера (як допоміжний інструмент для студентів та викладачів), ділове листування (перевірка документів, звітів, презентацій), створення контенту для вебсайтів та соціальних мереж, державні установи (підготовка офіційних документів).

Апробація результатів дипломної роботи. Основні положення дипломної роботи та результати досліджень подано до участі на конференції:

– XIII Міжнародна науково-практична конференція "Глобальні та регіональні проблеми інформатизації в суспільстві і природокористуванні 2025" (м. Київ, 13–14 листопада 2025 р.).

Структура та обсяг роботи. Дипломна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел, додатку. Основна частина містить 104 сторінок, 37 рисунків, 11 лістингів і 10 таблиць, список використаних джерел зі 40 найменувань.

ОСНОВНИЙ ЗМІСТ РОБОТИ

У першому розділі проведено комплексний аналіз предметної області та історії розвитку систем перевірки помилок у текстах від доцифрової епохи до сучасних нейромережевих архітектур.

Історичний огляд охоплює ключові етапи: фонетичні алгоритми (Soundex, 1918), концепцію відстані редагування (Левенштейн, 1965), першу інтерактивну програму SPELL (Горін, 1971) та систему Writer's Workbench (1970-ті) -- першу комплексну систему перевірки граматики для Unix. Розглянуто статистичний поворот (2000-2015) з N-грамами, Word2Vec та GloVe, а також нейромережеву революцію з архітектурою Transformer (2017), BERT (2018) та сімейством GPT.

Проаналізовано специфіку української мови: вільний порядок слів, складну систему відмінювання (7 відмінків), багату морфологію дієслів, апостроф та м'який знак, багатозначність лексем. Ці особливості створюють виклики для автоматизованої перевірки.

Виконано детальний огляд існуючих систем. LanguageTool (гібридна система, правила + N-грами) показує 50% точності на складних текстах за 13.9 с. Grammarly (глибоке навчання) має високу якість для англійської, але обмежену підтримку української. Система isgen.ai (генеративний AI) демонструє 76% точності за 12.7 с, але працює як "чорна скринька" без пояснень.

Виявлено основні обмеження: недостатнє врахування контексту, низька швидкодія (понад 10 секунд), високий рівень хибних спрацювань та пропусків, відсутність пояснень. Обґрунтовано доцільність використання LLM, які мають потенціал для глибокого розуміння контексту, але створюють виклики: "галюцинації", непередбачуваність форматів та високі обчислювальні витрати.

У другому розділі розроблено багаторівневу архітектуру обробки помилок з трьома рівнями захисту. Клієнтський рівень (Next.js) виконує валідацію даних, rate limiting та кешування результатів. Серверний рівень (Flask) забезпечує додаткову валідацію, асинхронну обробку через Gunicorn workers та логування. Рівень API включає обробку помилок OpenAI, експоненційне відкладання при повторних запитах та fallback-механізми.

Запропоновано методику гранулярного виявлення помилок з чотирма рівнями аналізу. Лексичний рівень перевіряє орфографію та друкарські помилки. Морфологічний аналізує словоформи та узгодження. Синтаксичний перевіряє структуру речень та керування дієслів. Семантичний виявляє русизми, канцеляризми та тавтологію.

Розроблено стратегію промпт-інженірингу з трьома підходами. Few-Shot Learning (3-5 прикладів) підвищує точність на 4-6%. Chain-of-Thought (поетапне міркування) дає +7% для синтаксичних помилок. Structured Output через JSON Schema повністю виключає помилки парсингу (рис.1).

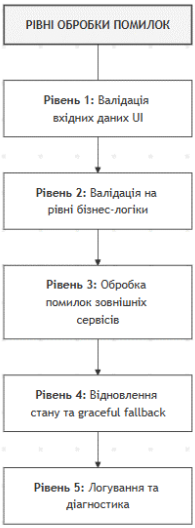

Рисунок 1 - Багаторівнева архітектура обробки помилок

Визначено дві оптимальні конфігурації: базова (GPT-3.5 Turbo, 1.2 с, точність 76.6%) для повсякденних завдань та поглиблена (GPT-4o, 2.5-3.5 с, точність 78.6%) для професійного редагування.

У третьому розділі описано реалізацію клієнт-серверної вебсистеми. Клієнтська частина використовує Next.js 14 з React 19 для оптимізованого рендерингу, автоматичного code splitting та типобезпеки через TypeScript. Серверна частина на Python Flask з Gunicorn забезпечує паралельну обробку запитів (workers = 2 * CPU_cores + 1).

Розроблено RESTful API: POST /api/analyze для аналізу тексту, GET /api/health для моніторингу та POST /api/feedback для збору відгуків. Реалізовано дворівневе кешування: клієнтське через React Query та серверне через Redis для популярних фраз.

Користувацький інтерфейс включає текстове поле з drag-and-drop, панель налаштувань (вибір моделі, рівня чутливості, типу промпту) та результати з кольоровим виділенням помилок (червоний -- граматика, жовтий -- стилістика, синій -- пунктуація). Реалізовано експорт у PDF/DOCX.

Впроваджено механізми надійності: обробку помилок мережі через Axios (до 3 retry), валідацію JSON, timeout-контроль (30 с) та graceful degradation (fallback на GPT-3.5 Turbo). Код структурований за принципами Clean Architecture.

У четвертому розділі проведено тестування на датасеті з 50 зразків: синтаксичні (30%), орфографічні (20%), пунктуаційні (20%), стилістичні (20%) помилки та контрольна група (10%). Зразки класифіковано за критичністю: критичні (50%), помірні (20%), низькі (20%).

Базова конфігурація (GPT-3.5 Turbo) показала: 100% оброблених запитів, 1.2 с середній час, 98% успішного парсингу JSON. Метрики: Precision 76.0%, Recall 79.2%, F0.5 Score 76.6%. Поглиблена конфігурація (GPT-4o) досягла 100% валідного JSON, 2.5-3.5 с обробки. Метрики: Precision 78.0% (+2.0%), Recall 81.2% (+2.1%), F0.5 Score 78.6% (+2.0%) (рис. 2).

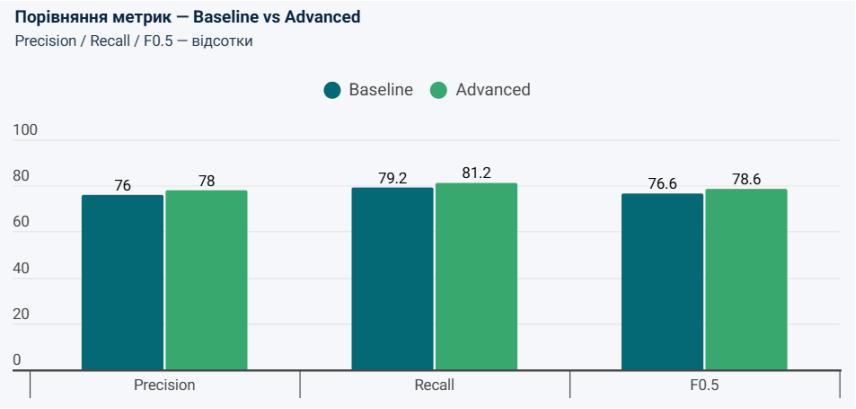

Рисунок 3 - Порівняння метрик за результатами дослідження

Аналіз за категоріями: орфографічні -- 100% проти 90%, синтаксичні -- 80.0% проти 77.5%, пунктуаційні -- 91.7% (однаково), стилістичні -- 54.5% проти 59.3% (регресія через надмірне редагування). Для критичних помилок: F0.5 88.0% проти 82.6% (+5.4%).

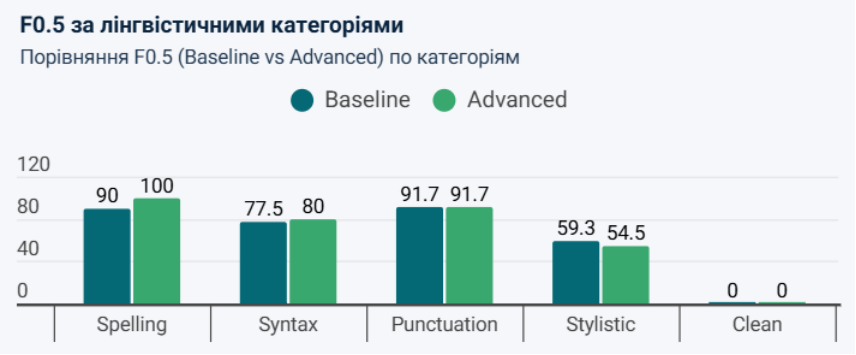

Рисунок 4 - Ефективність F0.5 за лінгвістичними категоріями

Порівняння з аналогами. Короткі тексти (22 символи): 100% за 1.34 с проти 50% за 3.64 с (isgen.ai) та 25% за 4.00 с (LanguageTool). Середні (583 символи): 100% за 3.55 с проти 80% за 6.70 с та 70% за 6.70 с. Великі (1617 символів): 84% за 5.08 с проти 76% за 12.70 с та 50% за 13.90 с (рис 5).

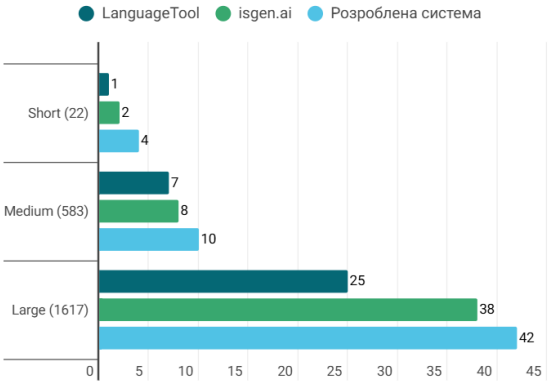

Рисунок 5 – Порівняння ефективності виявлення помилок

Ключові переваги: прискорення у 2.5-2.7 рази, 100% виявлення на малих/середніх текстах, детальні пояснення помилок (проти "чорної скриньки" isgen.ai), адаптивність через вибір конфігурації (рис. 6).

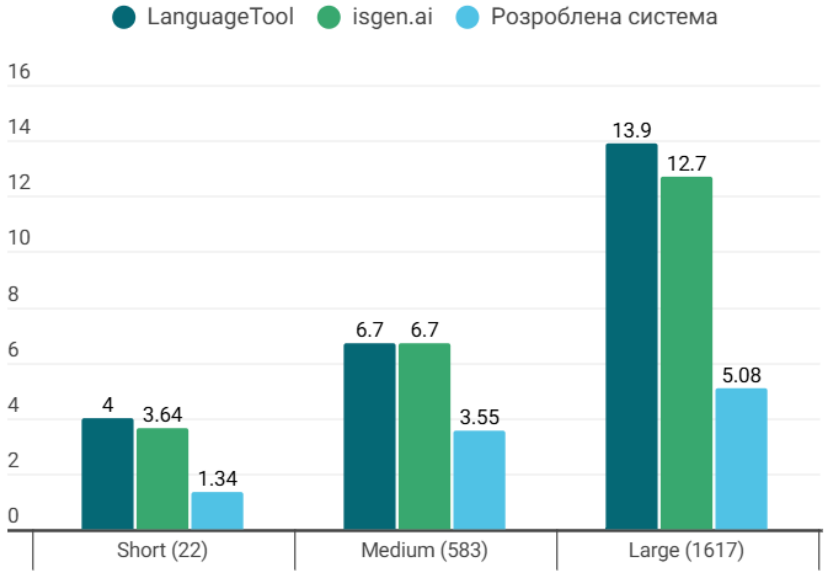

Рисунок 6 – Порівняння швидкодії систем

ВИСНОВКИ

У магістерській роботі розв'язано актуальну науково-прикладну задачу підвищення ефективності автоматизованої перевірки текстів українською мовою шляхом розробки комп'ютерної системи з використанням технологій штучного інтелекту.

Отримано такі результати:

1. Проведено аналіз предметної області від алгоритму Soundex (1918) до трансформерних архітектур. Встановлено, що наявні рішення не забезпечують балансу між якістю (50-76%) та швидкодією (12-14 с), що зумовило розробку нового підходу на основі LLM;

2. Розроблено багаторівневу архітектуру (клієнт, сервер, API) та метод гранулярного виявлення помилок на чотирьох рівнях, що забезпечує точність 78-81%;

3. Розроблено стратегію промпт-інженірингу (Few-Shot +4-6%, Chain-of-Thought +7%, JSON Schema виключає помилки парсингу). Визначено конфігурації: базова (1.2 с, 76.6%) та поглиблена (2.5-3.5 с, 78.6%);

4. Проведено тестування на 50 зразках: Recall 81.2% проти 76% та 50%, швидкодія 5.08 с проти 12.7 с та 13.9 с (прискорення у 2.5-2.7 рази), 100% виявлення на малих текстах, F0.5 Score 88.0% для критичних помилок.

Система поєднує переваги LLM (глибоке розуміння контексту) з оптимізованою архітектурою. Досягнуті показники дозволяють рекомендувати систему для освіти, бізнесу, видавництв та державних установ.

Перспективи: підтримка інших мов, інтеграція Claude/Gemini, навчання на зворотному зв'язку, мобільні додатки.

Таким чином, усі завдання виконано, а мету досягнуто.