УДК 004.5:004.8

Оврамець І. В., здобувач 2 курсу
ОС «Магістр» спеціальності
F3 Комп'ютерні науки,
Антонов Ю. С., канд. фіз.-мат. наук, доцент,
доцент кафедри інформаційних технологій

АВТОМАТИЗОВАНА СИСТЕМА ГЕНЕРАЦІЇ ВІДПОВІДЕЙ НА ПОВІДОМЛЕННЯ З ВИКОРИСТАННЯМ DOCKER КОНТЕЙНЕРІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. У сучасних умовах цифровізації обсяг електронних комунікацій зростає експоненційно. Користувачі щодня обмінюються мільярдами повідомлень через месенджери, корпоративні чати та системи підтримки клієнтів [1]. Це створює потребу в автоматизації процесів обробки та генерації відповідей, які б враховували контекст діалогу та зберігали природність спілкування.

Сучасні технології штучного інтелекту, зокрема великі мовні моделі (LLM) та методи обробки природної мови (NLP), дають змогу створювати інтелектуальні системи, здатні аналізувати історію переписки, визначати наміри користувачів і формувати релевантні відповіді. Однак для ефективного впровадження таких систем необхідно забезпечити їх незалежність від конкретної інфраструктури, можливість швидкого розгортання та масштабування.

Контейнеризація на основі Docker стала стандартом де-факто для побудови сучасних розподілених систем. Вона забезпечує ізоляцію середовища виконання, спрощує процеси розгортання та дає змогу реалізувати мікросервісну архітектуру, в якій кожен компонент функціонує автономно.

Основний текст. Архітектура автоматизованої системи генерації відповідей побудована на принципах модульності, ізоляції компонентів та незалежності від конкретного середовища виконання. Ключовою особливістю реалізації є виділення модуля генерації відповідей в окремий Docker-контейнер, що забезпечує його портативність і можливість використання як самодостатнього мікросервісу.

Система складається з кількох функціональних блоків, кожен з яких виконує чітко визначену роль. Центральним елементом є REST API, що забезпечує взаємодію між клієнтськими застосунками та серверною частиною. API структуровано за доменними модулями: авторизація (api/login), управління навчальними даними (Learning), робота з повідомленнями (Messages) та генерація рекомендацій (Recommendations). Усі ендпоінти документовані за допомогою Swagger/OpenAPI, що значно спрощує інтеграцію та тестування системи.

Особливу увагу приділено механізмам безпеки та валідації даних. Система використовує токенову автентифікацію (JWT), яка забезпечує захищений доступ до всіх ресурсів, окрім ендпоінту входу. Це унеможливлює несанкціонований доступ до внутрішніх даних та модифікацію інформації. Додатково реалізовано багаторівневу валідацію запитів, що блокує некоректні, небезпечні або спотворені дані на етапі їх надходження до системи.

Обробка вхідних повідомлень здійснюється через два послідовні підмодулі. Перший, MessageSanitizer, відповідає за видалення чутливої інформації – номерів банківських карток, email-адрес, телефонів, IP-адрес, IBAN та інших персональних даних. Це забезпечує відповідність вимогам GDPR та захищає конфіденційність користувачів. Другий підмодуль, AnalysingModule, виконує лінгвістичну обробку тексту: нормалізацію, токенізацію, виділення ключових слів, визначення намірів користувача та побудову структурованого представлення діалогу.

Для генерації відповідей використовується готова велика мовна модель (LLM) з бібліотеки Hugging Face Transformers. Реалізовано сервіс LLMService, який інкапсулює логіку завантаження моделі та генерації тексту. Модель та токенайзер завантажуються один раз під час першого виклику, що дає змогу уникнути повторного створення ресурсоємних об'єктів. Генерація відбувається з параметрами do_sample = True, top_k = 30 та temperature = 0.7, що забезпечує баланс між креативністю та релевантністю відповідей.

Ключовою перевагою запропонованого підходу є використання Docker-контейнеризації. Система упакована у Docker-контейнер разом з усіма залежностями, конфігураційними файлами та бібліотеками, що гарантує однакову роботу у будь-якому середовищі – від локального комп'ютера до хмарних платформ. Використання Docker Compose допомагає одночасно запускати декілька контейнерів (вебсервер, базу даних PostgreSQL, модуль рекомендацій) з налаштованими зв'язками між ними.

Така архітектура забезпечує низку переваг: масштабованість через можливість запуску кількох екземплярів контейнерів, ізоляцію середовища виконання, що мінімізує конфлікти залежностей, простоту розгортання на різних платформах, включно з AWS, Azure та Google Cloud, а також можливість швидкого оновлення окремих компонентів без зупинки всієї системи.

Експериментальна перевірка системи показала здатність генерувати контекстно релевантні відповіді з дружньою тональністю. Система успішно аналізує історію переписки, враховує попередні повідомлення та формує логічні продовження діалогу. Використання відносно невеликої моделі дає змогу забезпечити прийнятну швидкість генерації за обмежених обчислювальних ресурсів, що робить рішення придатним для демонстраційних та тестових сценаріїв.

Висновки. За результатами проведеного дослідження розроблено автоматизовану систему генерації відповідей на повідомлення, реалізовану у вигляді окремого Docker-контейнера. Система поєднує сучасні технології обробки природної мови, великі мовні моделі та принципи мікросервісної архітектури.

Запропонована архітектура забезпечує модульність, масштабованість та незалежність від конкретного середовища виконання. Реалізовано REST API з повною документацією через Swagger/OpenAPI, багаторівневі механізми безпеки та валідації даних, систему видалення чутливої інформації та аналітичну обробку повідомлень.

Використання Docker-контейнеризації дає змогу швидко розгортати систему на різних платформах, легко масштабувати її у разі зростання навантаження та інтегрувати з наявними комунікаційними системами – корпоративними месенджерами, службами підтримки клієнтів або вебчатами.

Експериментальна перевірка підтвердила працездатність системи та її здатність генерувати релевантні відповіді на основі контексту діалогу. Подальші дослідження можуть бути спрямовані на інтеграцію більш потужних мовних моделей, впровадження механізмів донавчання на спеціалізованих датасетах та розробку системи автоматичного оцінювання якості згенерованих відповідей.

Список використаних джерел

1. The Effect of WhatsApp Usage on Employee Innovative Performance at the Workplace: Perspective from the Stressor – Strain – Outcome Model. URL: https://ojs.srce.hr/index.php/entrenova/article/view/30184 (дата звернення: 07.09.2025).

2. Prieur B. Traitement automatique du langage naturel avec Python: le NLP avec spaCy et NLTK. Paris: ENI, 2024. 277 p.

3. Kane S. P., Matthias K. Docker: Up & Running. Sebastopol: O'Reilly Media, 2023. 327 p.

4. Corinne C. Hugging Face Transformers: A Step-by-Step Guide to Building NLP Applications with Python. Sebastopol: O'Reilly Media, 2025. 317 p.

УДК 004.422.5

Кудирка М. Ю., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Павлюк О. А., канд. техн. наук, старший викладач кафедри доцент, доцент кафедри інформаційних технологій

АЛГОРИТМИ ВИЗНАЧЕННЯ МАКСИМАЛЬНОЇ ПРОПУСКНОЇ ЗДАТНОСТІ В ПОТОКОВИХ МЕРЕЖАХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Задача визначення максимальної пропускної здатності в потокових мережах є однією з фундаментальних оптимізаційних задач у теорії графів. Потокова мережа – це абстрактна модель, що являє собою орієнтований граф, де кожне ребро (канал) має певну пропускну здатність, а два вузли виділені як джерело та сток. Мета полягає у знаходженні максимального обсягу «потоку», який можна передати від джерела до стоку. Цей підхід широко застосовується для моделювання систем у логістиці, телекомунікаціях, енергетиці та економіці. Теоретичною основою для розв'язання цієї задачі є теорема про максимальний потік та мінімальний розріз, яка стверджує, що максимальна величина потоку дорівнює мінімальній пропускній здатності серед усіх можливих розрізів мережі [1; 2].

Основоположним методом розв'язання задачі є підхід Форда–Фалкерсона, який полягає в ітеративному нарощуванні потоку. Його ідея базується на пошуку так званих доповнюючих шляхів – маршрутів від джерела до стоку в залишковій мережі, якими ще можна направити додатковий потік [1]. Залишкова мережа динамічно відображає вільну пропускну здатність кожного каналу. Процес повторюється