Науковий вісник НЛТУ України
Scientific Bulletin of UNFU
ISSN 1994-7836 (print)
ISSN 2519-2477 (online)
https://doi.org/10.36930/40340415
Article received 01.04.2024
Article accepted 23.05.2024
UDC 004.[2+8]
Correspondence author: P. B. Liashchynskyi petro.b.liashchynskyi@lpnu.ua

П. Б. Лящинський
Національний університет "Львівська політехніка", м. Львів, Україна

ПРОГРАМНИЙ ЗАСІБ ДЛЯ КЛАСИФІКАЦІЇ ТА СИНТЕЗУ БІОМЕДИЧНИХ ЗОБРАЖЕНЬ

Розроблено програмний засіб для класифікації та синтезу біомедичних зображень. Встановлено потребу штучного розширення наборів даних біомедичних зображень через їх обмежену доступність, що створює перешкоду для розвитку діагностичних інструментів. З'ясовано, що розроблений програмний засіб може вирішити цю проблему, генеруючи синтетичні, але реалістичні медичні зображення, що можуть слугувати додатковими даними для навчання класифікаторів. Розроблено функціональні вимоги до програмного засобу, а також його архітектуру, використовуючи сучасні технології програмування та проектування програмних засобів. Програмний засіб спроектовано, використовуючи модульну архітектуру, що дає змогу масштабувати кожен модуль незалежно від навантаження. Охарактеризовано закономірності архітектури програмного засобу, що містить клієнт-серверну взаємодію, базу даних MongoDB та використання брокера повідомлень RabbitMQ для асинхронного обміну даними між модулями програмного засобу. Основними модулями програмного засобу є: набори даних (відповідає за керування навчальними зображеннями), класифікатори (відповідає за навчання та використання згорткових нейронних мереж для класифікації зображень) та генератори (відповідає за навчання за використання генеративно-змагальних мереж для синтезу зображень). Програмний засіб розроблено, використовуючи різні мови програмування (Python, TypeScript) та сучасні технології (NodeJS, RabbitMQ, PyTorch, MongoDB, React). Також спроєктовано структуру бази даних, використовуючи логічну модель на підставі UML-діаграми класів. Показано ефективність використання згорткових нейронних мереж та генеративно-змагальних мереж для класифікації та синтезу біомедичних зображень, відповідно. Зроблено висновок про наукову новизну і практичну значущість розробленого програмного засобу, який відкриває нові можливості для медичної діагностики та досліджень, забезпечуючи гнучкість та масштабованість у синтезі та аналізі біомедичних зображень.

Ключові слова: нейронні мережі; нейромережеві методи; класифікація зображень; синтез зображень; архітектура програмного засобу.

Вступ / Introduction

Застосування інформаційних технологій у медицині відіграє вирішальну роль у покращенні діагностики, лікування та моніторингу стану пацієнтів. Цифровізація процесів, автоматизація оброблення даних та розроблення передових систем збирання та аналізу інформації дають змогу медичним фахівцям точніше і швидше ідентифікувати захворювання, а також спостерігати за динамікою стану здоров'я пацієнтів. Однією з ключових сфер, де інформаційні технології набули ширшого застосування, є комп'ютерні системи діагностування. Вони істотно спрощують процес аналізу біомедичних зразків, забезпечуючи високу точність та об'єктивність визначення діагнозу. Комп'ютерні системи діагностування використовують у багатьох сферах медицини, зокрема в гістології, цитології, мікробіології, для дослідження біопсій, клітин і мікроорганізмів [1].

Загалом інформаційні технології широко використовують для: зберігання та оброблення медичних даних, таких як історії хвороби, результати лабораторних досліджень, рентгенівські знімки та інші біомедичні зображення; діагностики захворювань за допомогою комп'ютерних програм та алгоритмів; планування лікування та ведення обліку пацієнтів; проведення наукових досліджень; телемедицини, яка дає змогу лікарям консультувати пацієнтів на відстані [6]. Використання інформаційних технологій допомагає лікарям ухвалювати більш обґрунтовані рішення, покращувати якість догляду за пацієнтами та знижувати витрати.

У сучасній медицині збільшення точності діагностики безпосередньо залежить від якості та доступності біомедичних зображень. Розвиток програмних засобів для класифікації та синтезу цих зображень відіграє ключову роль у вдосконаленні медичної діагностики та досліджень. Однак, незважаючи на значний прогрес, який було досягнуто у цій сфері, існують значні виклики та прогалини у розробленні та імплементації ефективних програмних засобів, зокрема для синтезу зображень, що є критично важливим для точної візуалізації патологічних станів. Подолання обмежень у точності та ефективності наявних програмних засобів для оброблення біомедичних зображень робить це дослідження актуальним. Це дослідження має на меті розробити новий програмний засіб, який покращить процеси класифікації та синтезу біомедичних зображень. Дослідження зосереджується на використанні сучасних методів машинного

Інформація про автора:
Лящинський Петро Борисович, магістр, аспірант, кафедра автоматизованих систем управління.
Email: petro.b.liashchynskyi@lpnu.ua; https://orcid.org/0000-0002-3920-6239

Цитування за ДСТУ: Лящинський П. Б. Програмний засіб для класифікації та синтезу біомедичних зображень. Науковий вісник НЛТУ України. 2024, т. 34, № 4. С. 120–127.
Citation APA: Liashchynskyi, P. B. (2024). Software tool for synthesis and classification of biomedical images. *Scientific Bulletin of UNFU*, 34(4), 120–127. https://doi.org/10.36930/40340415

навчання та штучного інтелекту.

Об'єкт дослідження – розроблення програмного засобу для класифікації та синтезу біомедичних зображень.

Предмет дослідження – методи та засоби розроблення програмного засобу для класифікації та синтезу біомедичних зображень з використанням штучних нейронних мереж.

Мета роботи – розробити програмний засіб для класифікації та синтезу біомедичних зображень з використанням методів і засобів штучного інтелекту, що дасть змогу підвищити точність та ефективність постановки діагнозу завдяки штучному розширенню наборів біомедичних зображень для навчання класифікаторів.

Для досягнення зазначеної мети визначено такі основні *завдання дослідження*:

1) проаналізувати основні літературні джерела щодо програмних систем і засобів медичної діагностики;
2) спроектувати архітектуру програмного засобу для класифікації та синтезу біомедичних зображень;
3) реалізувати програмний засіб, використовуючи сучасні мови програмування та технології;
4) зробити висновки та надати рекомендації щодо використання.

Аналіз останніх досліджень та публікацій. Ідеї використання інформаційних технологій у медицині стали досить популярними за останні роки [5, 9, 14, 22]. Проте найбільше досліджень проведено в галузі класифікації різних видів раку [12, 13, 20]. За останні роки досить стрімко зросла кількість публікацій стосовно застосування технологій штучного інтелекту в медичній діагностиці. Так, у роботі [3] описано застосування технологій штучного інтелекту в медицині для класифікації та сегментації біомедичних зображень. У роботі [16] автори розробили метод детектування раку молочної залози на підставі генетичного алгоритму. У пропонованому методі генетичний алгоритм застосовують для визначення яскравих ділянок на зображенні.

Автори роботи [21] досліджують вплив генеративного штучного інтелекту в медицині та охороні здоров'я. Він пояснює, як його можна використовувати для підтримки клінічних рішень, залучення пацієнтів і електронного управління медичними записами. У роботі наведено приклади інструментів на підставі штучного інтелекту, таких як Microsoft Copilot і Nuance, які покращують продуктивність і спрощують ведення клінічних записів. Однак автори також обговорюють питання точності, надійності та моральних наслідків рішень, прийнятих за допомогою штучного інтелекту. Вони прогнозують, що генеративна ШІ покращить догляд за пацієнтами, а не замінить лікарів, і відіграватиме важливу роль у сфері охорони здоров'я.

Донг Ніє та його колеги розробили модель, що використовує GAN (англ. *Generative Adversarial Networks*) мережі для синтезу медичних зображень з різних модальностей. Модель використовує FCN (англ. *Fully Connected Network*) для зіставлення вихідних зображень з цільовими модальностями і містить функцію втрат на підставі градієнта різниці зображень для запобігання розмитості зображень. Також досліджують стратегію довготривалого залишкового навчання для задач зі схожими вхідними та вихідними зображеннями. Висока продуктивність моделі на трьох наборах даних свідчить про те, що вона може генерувати реалістичні і точні медичні зображення, потенційно зменшуючи потребу в реальному скануванні в клінічних застосуваннях [15].

Дослідження [11] спрямоване на розроблення неінвазивного методу ранньої діагностики раку легенів за допомогою системи аналізу дихання. Система використовує масиви газових сенсорів та алгоритми глибокого навчання для виявлення органічних сполук у повітрі, що видихається. У дослідженні проаналізовано 181 клінічний зразок дихання здорових людей та хворих на рак легенів, досягнувши точності 97,8 %. Автори зазначили, що використання алгоритмів глибокого навчання для аналізу складних патернів дихання відкриває нові шляхи для медичної діагностики, що потенційно може привести до створення доступніших і точніших діагностичних пристроїв для раку легенів та інших захворювань.

Також існує велика кількість програмних систем для медичної діагностики. Так, у дослідженні [10] автори порівнюють дві версії ImageChecker – комп'ютерної системи виявлення раку молочної залози на повнопольних цифрових мамограмах. Результати показують, що загальна чутливість системи покращилася з версії 3.1 до версії 8.3, зі значним збільшенням від 92,3 до 96,2 %. Чутливість також зросла при виявленні утворень – від 78,3 до 89,1 %. Дослідження робить висновок, що оновлення версії програмного забезпечення покращує продуктивність системи.

Автори дослідження [18] аналізують систему OncoDoc – це система підтримки прийняття рішень, розроблена для покращення лікування раку молочної залози способом оптимізації терапевтичних рекомендацій. Вона використовує базу знань, структуровану у вигляді дерева рішень, для побудови формального еквіваленту стану пацієнта. Система може запропонувати доказові терапевтичні варіанти або визначити клінічні випробування для конкретного пацієнта. Впроваджена в Інституті Гюстава Руссі система OncoDoc підвищила обізнаність лікарів про відкриті клінічні випробування, що призвело до збільшення кількості пацієнтів на 17 %. Однак дослідження показало, що психологічне небажання лікарів скеровувати пацієнтів на клінічні випробування не може бути повністю подолане за допомогою персоналізованої інформації про випробування.

Аналіз літератури підкреслює швидкий прогрес штучного інтелекту, який революціонізує сферу медичної візуалізації. Потужні класифікатори зображень, побудовані на підставі алгоритмів глибокого навчання, тепер здатні діагностувати захворювання за біомедичними зображеннями з високою точністю. Ці класифікатори значною мірою покладаються на великі обсяги навчальних даних, щоб навчатися та робити точні прогнози. Однак значна проблема виникає через дефіцит відкритих доступних наборів даних біомедичних зображень, які мають вирішальне значення для навчання цих систем. Обмежена доступність таких наборів даних створює істотну перешкоду для розроблення та вдосконалення діагностичних інструментів на підставі штучного інтелекту.

Дефіцит навчальних даних вказує на потребу штучного розширення цих наборів даних. Це розширення залежить не тільки від кількості, а й від різноманітності та якості, що гарантує, що моделі штучного інтелекту є надійними, універсальними та здатними до узагальнення для різних демографічних груп пацієнтів і проявів захворювань. Отже, розроблення програмних засобів для класифікації та синтезу біомедичних зображень є *актуальним*. Ці інструменти спрямовані на створення

синтетичних зображень, доповнюючи наявні набори даних, цим самим полегшуючи навчання потужніших і точніших класифікаторів.

Матеріали та методи дослідження. Дослідження проведено з використанням: методів математичної статистики; теорії нейронних мереж; методів машинного навчання; сучасних технологій програмування; бібліотек і фреймворків машинного навчання; методів проектування програмного забезпечення; мови програмування Python; технології Node.js та TypeScript; вебфреймворка NestJS та бібліотеки для створення інтерфейсів користувача React.

Результати дослідження та їх обговорення / Research results and their discussion

Формулювання функціональних вимог до програмного забезпечення. Програмний засіб повинен складатися із таких основних модулів: набори даних, класифікатори, генератори. Кожен із цих модулів має підтримувати створення, отримання, редагування і видалення ресурсів. У веброзробці ці операції описано однією абревіатурою CRUD (англ. *Create Read Update Delete*).

Модуль наборів даних призначений для додавання директорії зображень для навчання класифікаторів і генераторів. Основними вимогами до модуля є:

- можливість створювати нові набори даних, зазначаючи їх назви та шляхи до директорій з зображеннями;
- можливість імпорту зображень у різних форматах (наприклад, JPEG, PNG).
- можливість зміни назви набору даних;
- можливість зміни шляху директорії набору даних (заборонено за наявності пов'язаних класифікаторів чи генераторів);
- можливість видалення цілого набору даних із системи;
- можливість підтвердження (чи заборона) видалення для запобігання випадковій втраті даних за наявності пов'язаних класифікаторів чи генераторів.

Модуль класифікаторів призначений для класифікації зображень на підставі готових або власних моделей згорткових нейронних мереж. Основними вимогами до модуля є:

- можливість створення нових класифікаторів із зазначенням їх архітектури (на підставі готових моделей чи власних) та набору даних для навчання;
- можливість зміни архітектури і набору даних класифікаторів;
- можливість навчання класифікаторів на нових наборах даних;
- можливість задати параметри (норма навчання, алгоритм оптимізації) перед початком навчання класифікатора;
- можливість видалення існуючих класифікаторів з системи;
- можливість підтвердження видалення для запобігання випадковому видаленню.

Модуль генераторів призначений для синтезу зображень на підставі готових або власних архітектур GAN-мереж. Основними вимогами до модуля є:

- можливість створення нових генераторів із зазначенням їх архітектури (на підставі готових моделей чи власних) та набору даних для навчання;
- можливість зміни архітектури і набору даних генераторів;
- можливість навчання генераторів на нових наборах даних;
- можливість задати параметри (норма навчання, алгоритм оптимізації) перед початком навчання генератора;
- можливість видалення наявних генераторів із системи;
- можливість підтвердження видалення для запобігання випадковому видаленню.

Проектування архітектури програмного засобу. Для реалізації цього етапу дослідження використано онлайн-сервіс Diagrams.Net. Архітектура програмного забезпечення базується на клієнт-серверній технології (рис. 1), що в майбутньому дасть змогу масштабувати розроблену програму відповідно до потреб навантаження.

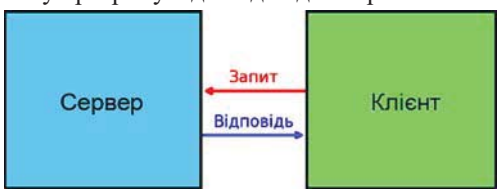

Рис. 1. Клієнт-серверна взаємодія / Client-server interaction

На рис. 2 зображено архітектуру клієнт-серверної взаємодії у системі.

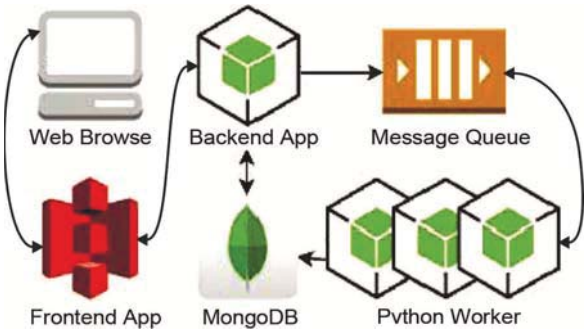

Рис. 2. Клієнт-серверна архітектура системи / Client-server system architecture

Архітектура системи містить такі компоненти:

- Web Browser (веббраузер): Представляє користувача, який взаємодіє із системою через вебінтерфейс. Веббраузер надсилає запити до фронтенду та отримує від нього відповіді;
- Frontend App (Фронтенд-додаток): Це інтерфейс, з яким взаємодіє користувач (через веббраузер). Він приймає вхідні дані від користувача та відправляє запити до бекенду. Зазвичай це набір вебсторінок, що містять HTML, CSS та JavaScript;
- Backend App (бекенд-додаток): Це серверна частина вебдодатку, яка обробляє запити, надіслані з фронтенду, виконує логіку додатку, звертається до бази даних за необхідною інформацією, а потім повертає відповіді до фронтенду;
- MongoDB: Це система управління базами даних, яка зберігає дані програмної системи. MongoDB – це NoSQL база даних, оптимізована для великої кількості операцій читання та запису та горизонтального масштабування;
- Message Queue (черга повідомлень): Механізм для асинхронного обміну повідомленнями між різними частинами додатку. Це може допомагати у вирішенні завдань, які не потребують миттєвого оброблення, та врівноважувати навантаження між різними воркерами або сервісами.
- Python Worker (пітон-працівник): Це сервери або процеси, які обробляють завдання асинхронно. Вони можуть отримувати завдання з черги повідомлень та виконувати фонове оброблення, таке як операції з базою даних, складні обчислення або інтеграція із зовнішніми сервісами.

Розроблена архітектура відображає сучасний вебзастосунок з чітко розділеними компонентами для фронтенду, бекенду, бази даних, оброблення черги повідомлень та фонового оброблення. Це частково демонструє використання шаблону проектування "мікросервісів", де кожен компонент може бути розміщений та масштабований незалежно. Такий підхід дасть змогу в майбутньому масштабувати систему горизонтально.

Проектування структури бази даних. Для реалізації програмної системи потрібно спроектувати структуру бази даних типу NoSQL. Логічну модель бази даних у вигляді UML-діаграми класів наведено на рис. 3.

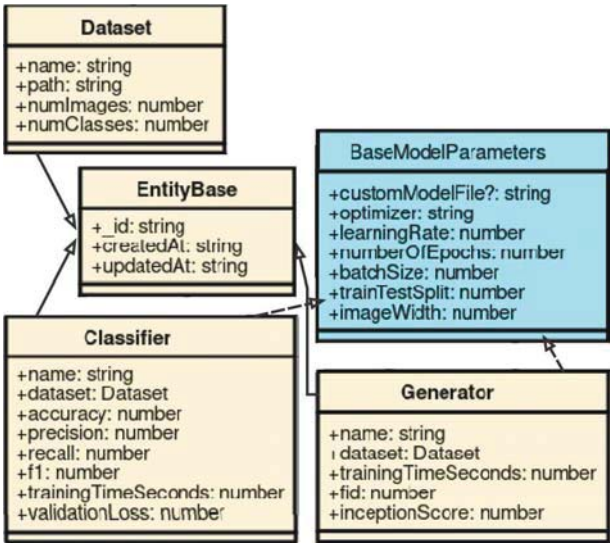

Рис. 3. Логічна модель бази даних / Database logical model

Основними сутностями бази даних є: Dataset (набір даних) – відображає директорію із навчальними зображеннями, що поділені на класи; Classifier (класифікатор) – відображає модель нейронної мережі класифікатора, його навчальні параметри та фінальні метрики; Generator (генератор) – відображає модель нейронної мережі генератора, його навчальні параметри та фінальні метрики.

Як СУБД обрано нереляційну базу даних MongoDB, яка є досить потужним інструментом, а найголовніше дуже добре піддається горизонтальному масштабуванню. Перевагою є те, що документи зберігаються у форматі JSON (англ. *JavaScript Object Notation*), що значно спрощує взаємодію з ними із серверного додатку.

Програмна реалізація. Для написання коду використано Visual Studio Code IDE. Відповідно до спроектованої архітектури програми її реалізовано окремими модулями, кожен з яких відповідає за певні завдання. Розглянемо детальніше кожен з них.

Frontend App. Цей додаток відповідає за взаємодію користувача із системою. Користувач має змогу створювати, редагувати та видаляти дані, як описано у функціональних вимогах. Приклад сторінки фронтенд додатку показано на рис. 4.

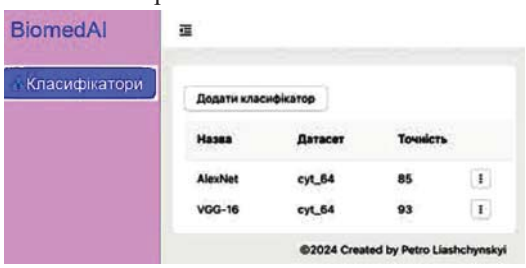

Рис. 4. Приклад сторінки інтерфейсу користувача / User interface example page

Для реалізації цієї частини програми використано мову програмування TypeScript, бібліотеку React та бібліотеку готових UI-компонентів Ant Design. Загалом цей додаток поєднує React з Ant Design для компонентів інтерфейсу, використовує React Router для навігації, Redux для управління станами, власні SVG (англ. *Scalable Vector Graphics*) іконоки та Axios для HTTP-запитів, дотримуючись загальної структури (рис. 5) для сучасного розроблення вебдодатків.

Backend App. Для реалізації серверної частини додатку використано фреймворк NestJS та мову програмування TypeScript. NestJS – це прогресивна платформа, яка використовує Node.js для створення ефективних і масштабованих серверних додатків. Насамперед платформа відома завдяки використанню TypeScript, що покращує процес розроблення та забезпечує безпеку типів. NestJS також добре працює з такими середовищами розроблення, як Visual Studio Code, а додатки, написані з використанням платформи, прості в обслуговуванні та масштабуванні завдяки модульній архітектурі. Загалом платформа дає змогу використовувати рішення та шаблони проектування з інших парадигм програмування, таких як функційне програмування (ФП), об'єктно-орієнтоване програмування (ООП) і функційно-реактивне програмування (ФРП).

Серверний додаток побудований на підставі модульної архітектури. Оскільки відповідно до функціональних вимог у системі є три модулі (набори даних, класифікатори, генератори), то кожен модуль винесений в окрему директорію. Структуру директорій зображено на рис. 5.

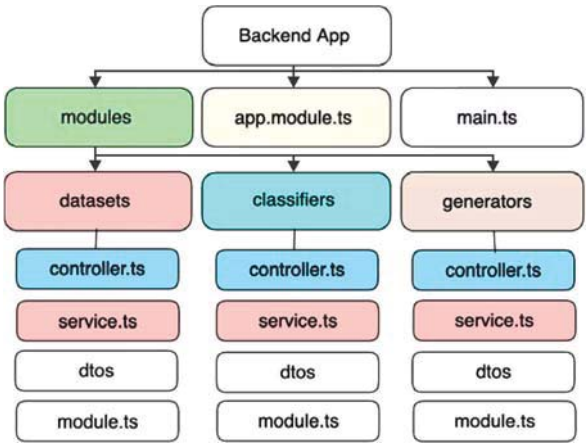

Рис. 5. Структура директорій додатку / App directory structure

Вхідною точкою додатку є файл *main.ts*, де описано логіку запуску вебсервера та основні його конфігурації. У цьому файлі створюється головний модуль (AppModule) та запускається вебсервер. У файлі *app.module.ts* описано головний модуль, який імпортує всі інші модулі системи.

Далі йде директорія із всіма додатковими модулями. Наприклад, розглянемо модуль наборів даних. У папці *dtos* знаходяться класи Data Transfer Object, які описують в якому форматі вебсервер може приймати дані, коли до нього звертаються за певними URL адресами. Файл *controller.ts* описує клас HTTP роутера, який маршрутизує запити з конкретних URL адрес вебсервера на певні обробники, які якраз описані в файлі *service.ts*. Файл *module.ts* є вхідною точкою будь-якого модуля. У ньому імпортують та експортують всі залежності. Саме цей файл підключається до головного модуля програми AppModule.

Однією з переваг NestJS є те, що користувач практично з мінімальними затратами може налаштувати генерування OpenAPI документації для описаних контролерів. Наприклад, документацію API для модуля наборів даних показано на рис. 6.

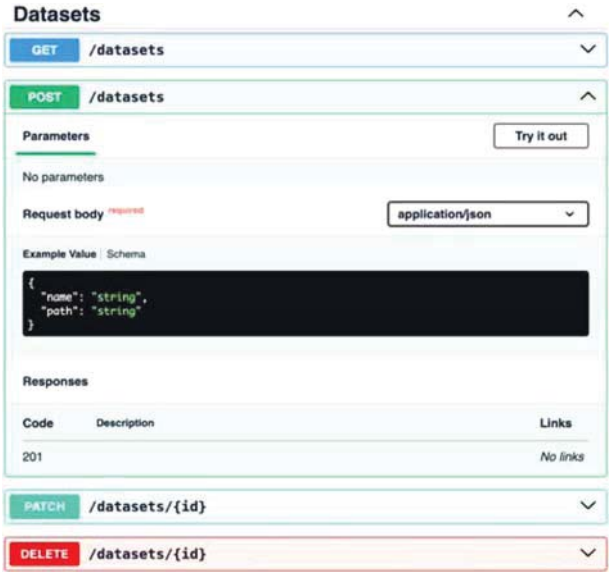

Рис. 6. Приклад OpenAPI документації для модуля наборів даних / Example of OpenAPI documentation for datasets module

Message Queue. Черга повідомлень, також відома як просто queue, є функцією програми, яка дає змогу обмінюватися повідомленнями між різними її частинами. Її можна порівняти з поштовою скринькою, куди надсилаються повідомлення. Після цього отримувач може взяти їх, коли йому зручно. Основним концептом черги повідомлень є:

- асинхронний зв'язок: означає, що людина, яка відправляє повідомлення, не зобов'язана чекати, доки воно буде отримано одержувачем. Це збільшує масштабованість і ефективність системи;
- надійність: черга повідомлень гарантує безпечну доставку. Повідомлення можна повторно спробувати обробити пізніше, навіть якщо виникає збій під час оброблення;
- взаємодія незалежних програм: дає змогу різним частинам програми спілкуватися між собою, незважаючи на те, що вони написані на різних мовах або працюють на різних комп'ютерах.

Загалом, черга повідомлень є потужним інструментом, який може допомогти створити програми, які більш масштабовані та гнучкі.

У нашому випадку чергу повідомлень застосовують у модулях класифікаторів та генераторів у серверному додатку. Процес навчання моделі нейронної мережі часто є довготривалим. Для того щоб користувач не чекав поки він завершиться, ми додаємо повідомлення в чергу, а користувачу показуємо сповіщення, що процес навчання класифікатора або генератора розпочато. Цей підхід дає змогу не блокувати взаємодію користувача із додатком, тому користувач може виконувати інші завдання паралельно (наприклад, додавати чи редагувати набори даних). Для імплементації черги повідомлень у нашій програмі ми використали платформу RabbitMQ. В основі цієї технології лежить концепція продюсерів і споживачів. Продюсер публікує повідомлення в чергу, а споживач його отримує та обробляє.

Python Worker. Ці сервіси фактично є отримувачами (споживачами) повідомлень із RabbitMQ, а відповідно один отримувач відповідає за навчання моделі нейронної мережі. Серверний додаток є продюсером, який публікує повідомлення про потребу запуску навчання мережі в чергу. Після того як навчання завершене, споживач створює нотифікацію в базі даних, після чого користувач бачить відповідне повідомлення у системі. Для імплементації споживачів використано мову програмування Python та фреймворк машинного навчання PyTorch (для створення та навчання моделей нейронних мереж).

Оскільки навчання мережі потребує досить багато системних ресурсів, то споживачі повідомлень також дуже добре піддаються масштабуванню. Наприклад, вони можуть бути винесені на окремі потужні комп'ютери, щоб пришвидшити навчання моделей.

Загалом взаємодія двох сервісів через чергу повідомлень є досить простою (рис. 7).

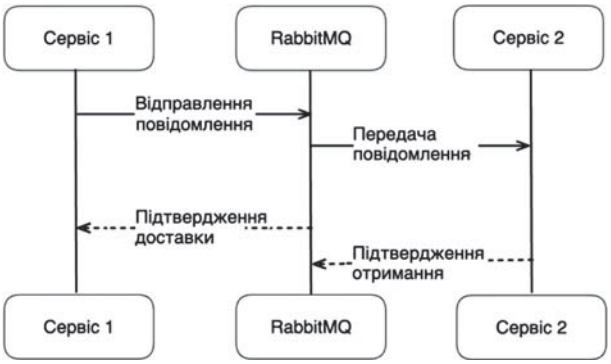

Рис. 7. Взаємодія двох сервісів використовуючи RabbitMQ / Interaction between two services using RabbitMQ

Отже, програмний засіб реалізовано із використанням клієнт-серверної технології використовуючи такі мови програмування, як TypeScript та Python. Програмний засіб складається із клієнтської (реалізовано з використанням бібліотеки React) та серверної (реалізовано використовуючи фреймворк NestJS) частин, а також із обробника повідомлень на базі RabbitMQ та Python. Функціональними особливостями системи є можливість класифікації та синтезу біомедичних зображень. Систему реалізовано із використанням розподіленої архітектури, що задовольняє таку нефункціональну вимогу, як гнучкість та масштабованість.

Результати дослідження. Як експеримент проведемо класифікацію та синтез цитологічних зображень. Приклад цитологічних зображень показано на рис. 8.

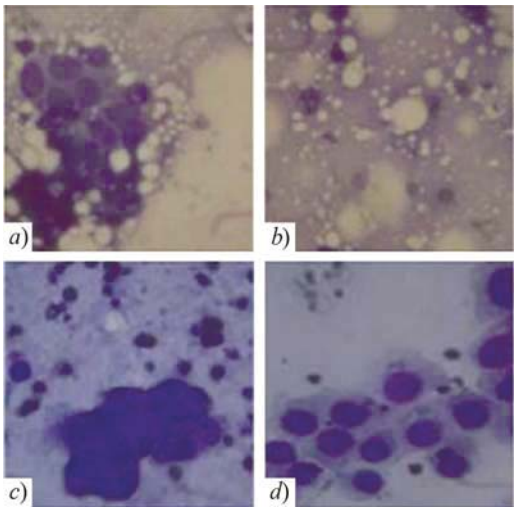

Рис. 8. Приклад цитологічних зображень / Example of cytological images

Для класифікації обрано архітектуру згорткової нейронної мережі AlexNet. Норму навчання встановлено у значення 0,001, а як оптимізатора обрано алгоритм Adam [2]. Вхідні зображення конвертовано у роздільну здатність 64×64 пікселі. Внаслідок проведення експери-

менту мережа досягнула точності класифікації 85 %. ROC-криву зображено на рис. 9,*a*.

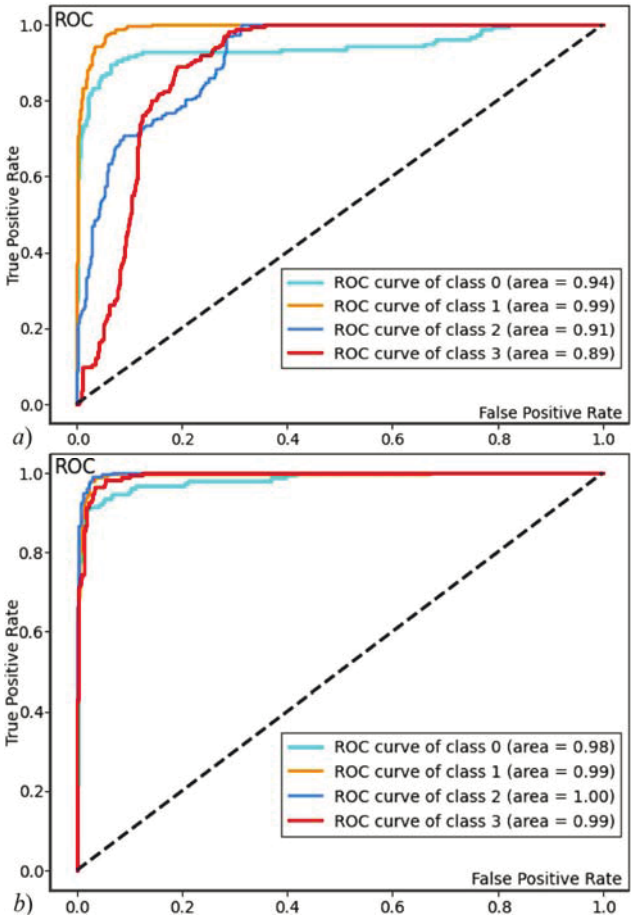

Рис. 9. ROC-крива / ROC curve of the: класифікатора AlexNet (*a*); розробленого класифікатора / developed classifier (*b*)

Також проведено ще один експеримент використовуючи розроблену архітектуру згорткової мережі. Усі параметри навчання мережі встановлено у ті самі значення, що й у попередньому експерименті. Архітектуру розробленої мережі показано на рис. 11. Внаслідок проведення експерименту мережа досягнула точності 98,3 %. ROC-криву показано на рис. 9,*b*.

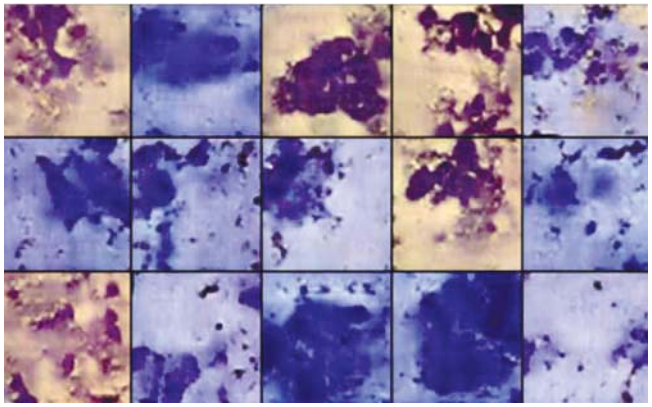

Рис. 11. Приклад синтезованих зображень / Example of synthetic images

Для синтезу зображень оберемо архітектуру генеративно-змагальної мережі DCGAN. Норму навчання мережі також встановлено у значення 0,001, а як оптимізатора обрано алгоритм Adam. Внаслідок проведення експерименту мережа може генерувати цитологічні зображення роздільною здатністю 64×64 пікселі. Приклад синтезованих зображень наведено на рис. 10.

Розроблений програмний засіб дає змогу класифікувати та синтезувати біомедичні зображення як на комп'ютері користувача, так і використовуючи віддалені сервери. Оскільки архітектура системи побудована з використанням клієнт-серверної технології, то кожен з описаних сервісів може бути розгорнутий як локально, так і використовуючи технології хмарних обчислень.

Навчання нейронних мереж часто потребує потужних графічних процесорів, які не кожен користувач має в наявності, тому можливість перенесення цих обчислень на хмару є перевагою розробленої системи. Проте варто відзначити, що при використанні системи для синтезу зображень використовуючи складні моделі нейронних мереж чи велику роздільну здатність зображень необхідно буде винести модуль *Python Worker* на окрему хмарну інфраструктуру, що може призвести до високих грошових витрат. В основному витрати залежать від складності моделі та часу її навчання.

Обговорення результатів дослідження. У більшості публікацій автори використовують уже відомі архітектури нейронних мереж або використовують власні архітектури, зроблені на основі відомих. Так, у роботі [19] автори застосували розроблену згорткову мережу на основі архітектури AlexNet для класифікації цитологічних зображень. У дослідженні автори використовували зображення більшої роздільної здатності (227×227 пікселів), а також їм вдалося отримати точність класифікації 93,3 %.

У роботі [4] автори застосували GAN мережі для розширення набору даних цитологічних зображень з метою покращення класифікації. Автори заявляють, що після розширення оригінального набору даних за допомогою синтетичних зображень їм вдалося досягнути точності класифікації 94 %. Варто зазначити, що автори використовують зображення у відтінках сірого, що значно полегшує навчання класифікаторів і генераторів.

У роботі [8] автори розробили автоматизовану CAD-систему (англ. *Computer-aided Diagnosis*) використовуючи технології глибинного машинного навчання. У ролі наборів даних автори використали зображення пухлин головного мозку та набір даних COVID-19. Обидва набори даних містять зображення у відтінках сірого. Основними алгоритмами та технологіями, що використані у розробленій системі, є: згорткові нейронні мережі (AlexNet, GoogleNet), метод опорних векторів, дерева прийняття рішень та інші. Автори стверджують, що їх систему можуть впевнено використовувати лікарі для діагностики пацієнтів, що хворіють на COVID-19 або мають пухлини головного мозку.

У роботі [17] автори роблять порівняння декількох архітектур згорткових мереж для класифікації цитологічних зображень. У порівняння включено такі архітектури, як AlexNet, ResNet, VGG. Авторам вдалося досягнути максимальної точності класифікації 94,4 %. У роботі [7] автори розробили систему для діагностування раку молочної залози використовуючи генеративні мережі для розширення навчальних наборів даних, а потім згорткові мережі для класифікації зображень. Автори досягнули точності класифікації 96 %.

Отже, за результатами виконаної роботи можна сформулювати такі наукову новизну та практичну значущість результатів дослідження.

Наукова новизна отриманих результатів дослідження – розроблено методологію класифікації і синте-

зу біомедичних зображень на підставі використання згорткових та генеративно-змагальних нейронних мереж, що дасть змогу значно підвищити точність діагностики захворювань, зокрема, створенням якісних синтетичних зображень для розширення обмежених наборів даних та покращення навчальних вибірок для класифікаторів. Також під час дослідження розроблено модель згорткової мережі для класифікації біомедичних (цитологічних) зображень на підставі використання глибинного машинного навчання.

Практична значущість результатів дослідження – розроблено програмний засіб для класифікації та синтезу біомедичних зображень, який використано в комп'ютерній системі діагностування раку молочної залози.

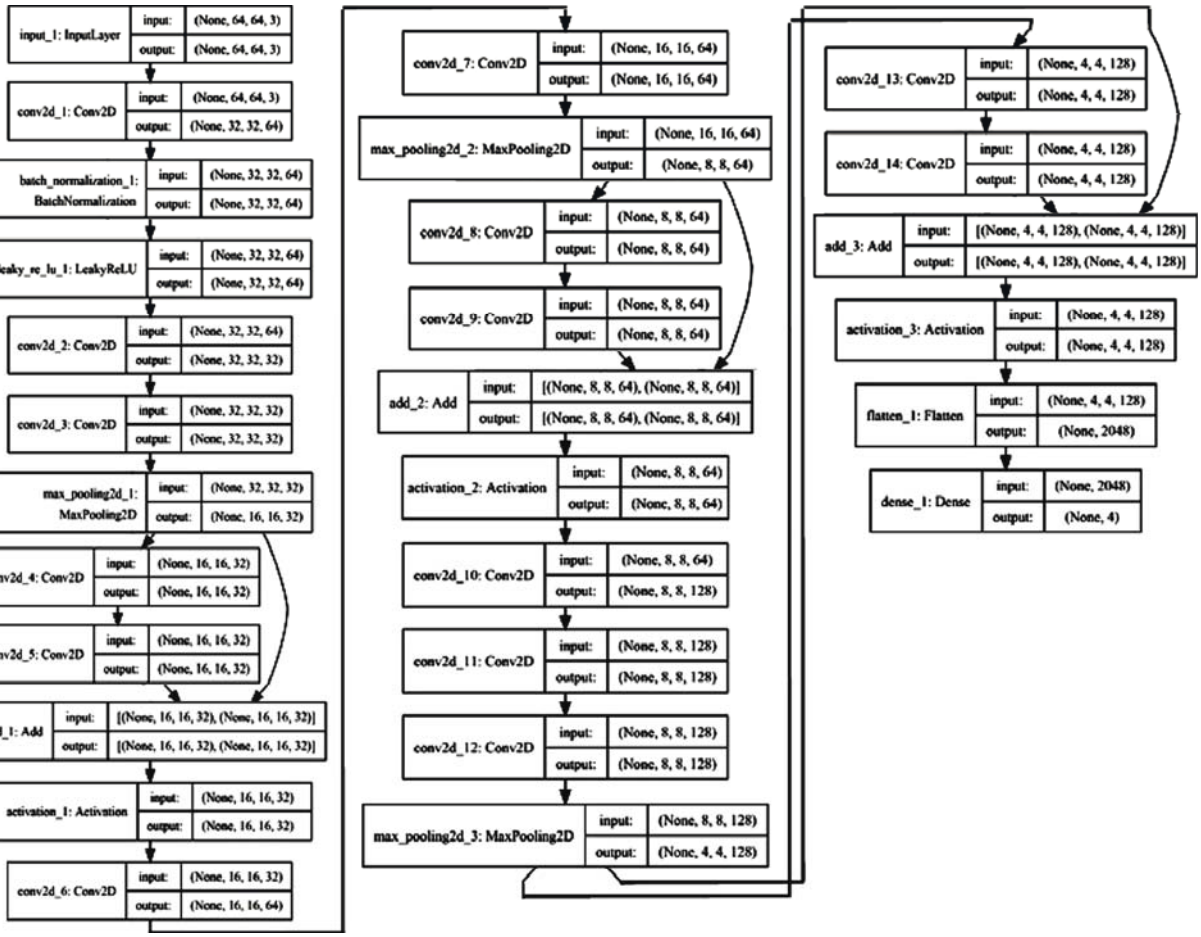

Рис. 10. Архітектура розробленої згорткової мережі / Architecture of the developed convolutional network

Висновки / Conclusions

Як результат цього дослідження розроблено програмний засіб для класифікації та синтезу біомедичних зображень. Із дослідження можна зробити такі висновки:

1. Аналіз літератури показав актуальність проблеми розроблення програмних засобів для класифікації та синтезу біомедичних зображень, що дає змогу підвищити ефективність та точність медичної діагностики, забезпечуючи при цьому ширші можливості для навчання класифікаторів на підставі збільшеного і різноманітнішого набору даних, що відіграє ключову роль у вдосконаленні технологій штучного інтелекту в медицині.
2. Спроектовано архітектуру програмного засобу використовуючи сучасні, надійні технології та підходи у сфері розроблення програмного забезпечення, що дасть змогу легко масштабувати систему в майбутньому відповідно до навантаження та обсягів даних, забезпечити високий рівень безпеки медичної інформації та оптимізувати взаємодію між модулями системи, що сприятиме ефективній інтеграції розробленого засобу в медичні інформаційні системи та дасть змогу медичним фахівцям використовувати новітні досягнення в галузі штучного інтелекту для покращення діагностики.
3. Розроблено структуру організації бази даних використовуючи UML-діаграму класів для надійного зберігання даних системи, яка забезпечить ефективне управління великими обсягами біомедичних зображень і відповідних метаданих, оптимізацію процесів пошуку та вибірки необхідної інформації для аналізу, а також сприятиме забезпеченню цілісності та конфіденційності медичних даних, що є критично важливим для дотримання нормативних вимог у сфері охорони здоров'я.
4. Розроблено програмний засіб використовуючи різні мови програмування та технології. Основним функціоналом програми є менеджмент наборів даних, класифікаторів та генераторів. Користувач має змогу навчати класифікатори і генератори на власних наборах даних, а також генерувати нові зображення за допомогою навчених генераторів.

References

1. Alenoghena, C. O., Ohize, H. O., Adejo, A. O., Onumanyi, A. J., Ohihoin, E. E., Balarabe, A. I., & Alenoghena, B. (2023). Telemedicine: A survey of telecommunication technologies, developments, and challenges. *Journal of Sensor and Actuator Networks*, *12*(2), 20 p. https://doi.org/10.3390/jsan12020020
2. Berezsky, O., Liashchynskyi, P., Pitsun, O., Liashchynskyi, P., & Berezkyy, M. (2022). Comparison of Deep Neural Network Learning Algorithms for Biomedical Image Processing. *The 5th International Conference on Informatics & Data-Driven Medicine* (*IDDM 2022*) *will be held in University Lumière Lyon 2, Lyon, France*, 135–145. URL: https://www.researchgate.net/publication/367190420_Comparison_of_Deep_Neural_Network_Learning_Algorithms_for_Biomedical_Image_Processing

3. Berezsky, O., Pitsun, O., Liashchynskyi, P., Derysh, B., & Batryn, N. (2022). Computational Intelligence in Medicine. In *International Scientific Conference "Intellectual Systems of Decision Making and Problem of Computational Intelligence"*, 488–510. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-16203-9_28
4. Dhivya, S., Mohanavalli, S., Karthika, S., Shivani, S., & Mageswari, R. (2020). GAN based data augmentation for enhanced tumor classification. In *2020 4th international conference on computer, communication and signal processing* (*ICCCSP*), 1–5. https://doi.org/10.1109/ICCCSP49186.2020.9315189
5. Doo, F. X., Kulkarni, P., Siegel, E. L., Toland, M., Paul, H. Y., Carlos, R. C., & Parekh, V. S. (2024). Economic and Environmental Costs of Cloud Technologies for Medical Imaging and Radiology Artificial Intelligence. *Journal of the American College of Radiology*, *21*(2), 248–256. https://doi.org/10.1016/j.jacr.2023.11.011
6. Habuza, T., Navaz, A. N., Hashim, F., Alnajjar, F. S., Zaki, N., Serhani, M. A., & Statsenko, Y. (2021). AI applications in robotics, diagnostic image analysis and precision medicine: Current limitations, future trends, guidelines on CAD systems for medicine. Informatics in Medicine Unlocked, 24. https://doi.org/10.1016/j.imu.2021.100596
7. Inan, M. S. K., Hossain, S., & Uddin, M. N. (2023). Data augmentation guided breast cancer diagnosis and prognosis using an integrated deep-generative framework based on breast tumors morphological information. *Informatics in Medicine Unlocked*, *37*. https://doi.org/10.1016/j.imu.2023.101171
8. Kadhim, Y. A., Khan, M. U., & Mishra, A. (2022). Deep learning-based computer-aided diagnosis (cad): applications for medical image datasets. *Sensors*, *22*(22). https://doi.org/10.3390/s22228999
9. Khang, A., Abdullayev, V., Ali, R. N., Bali, S. Y., Mammadaga, G. M., & Hafiz, M. K. (2024). Using Big Data to Solve Problems in the Field of Medicine. In *Computer Vision and AI-Integrated IoT Technologies in the Medical Ecosystem*, 407–418. CRC Press. https://doi.org/10.1201/9781003429609-23
10. Kim, S. J., Moon, W. K., Kim, S. Y., Chang, J. M., Kim, S. M., & Cho, N. (2010). Comparison of two software versions of a commercially available computer-aided detection (CAD) system for detecting breast cancer. *Acta Radiologica*, *51*(5), 482–490. https://doi.org/10.3109/02841851003709490
11. Lee, B., Lee, J., Lee, J., Hwang, Y., Bahn, H., Park, I., Jheon, S., & Lee, D. (2024). Breath Analysis System with Convolutional Neural Network (CNN) for Early Detection of Lung Cancer. Sensors and Actuators B: Chemical. https://doi.org/10.1016/j.snb.2024.135578
12. Mahoro, E., & Akhloufi, M. A. (2024). Breast cancer classification on thermograms using deep CNN and transformers. *Quantitative InfraRed Thermography Journal*, *21*(1), 30–49. https://doi.org/10.1080/17686733.2022.2129135
13. Majumder, S., Gautam, N., Basu, A., Sau, A., Geem, Z. W., & Sarkar, R. (2024). MENet: A Mitscherlich function based ensemble of CNN models to classify lung cancer using CT scans. *Plos one*, *19*(3). https://doi.org/10.1371/journal.pone.0298527
14. Nazirova, T. O., & Kostenko, O. B. (2018). Neural network information technology for the processing of medical data. *Scientific Bulletin of UNFU*, *28*(8), 141–145. https://doi.org/10.15421/40280828
15. Nie, D., Trullo, R., Lian, J., Wang, L., Petitjean, C., Ruan, S., Wang, Q., & Shen, D. (2018). Medical Image Synthesis with Deep Convolutional Adversarial Networks. IEEE Transactions on Biomedical Engineering, 65, 2720–2730. https://doi.org/10.1109/TBME.2018.2814538
16. Peng, Y., Yao, B., & Jiang, J. (2006). Knowledge-discovery incorporated evolutionary search for microcalcification detection in breast cancer diagnosis. Artificial intelligence in medicine, 37(*1*), 43–53. https://doi.org/10.1016/j.artmed.2005.09.001
17. Remya, R., & Raimond, K. (2023). Cervical cell classification using Deep Learning Techniques. 2023 International Conference on Computer Communication and Informatics (ICCCI), 1–5. https://doi.org/10.1109/ICCCI56745.2023.10128238
18. Séroussi, B., & Bouaud, J. (2001). Using OncoDoc as a computer-based eligibility screening system to improve accrual onto breast cancer clinical trials. Artificial intelligence in medicine, 29, 1-2, 153–67. https://doi.org/10.1016/s0933-3657(03)00040-x
19. Wu, M., Yan, C., Liu, H., Liu, Q., & Yin, Y. (2018). Automatic classification of cervical cancer from cytological images by using convolutional neural network. *Bioscience reports*, *38*(6). https://doi.org/10.1042/BSR20181769
20. Wu, R., Qin, K., Fang, Y., Xu, Y., Zhang, H., Li, W., & Li, Q. (2024). Application of CNN in the diagnosis for the invasion depth of gastrointestinal cancer: a systematic review and meta-analysis. *Journal of Gastrointestinal Surgery*. https://doi.org/10.1016/j.gassur.2023.12.029
21. Zhang, P., & Boulos, M. N. (2023). Generative AI in Medicine and Healthcare: Promises, Opportunities and Challenges. *Future Internet*, 15, 286 p. https://doi.org/10.3390/fi15090286
22. Zhang, W., Cai, M., Lee, H. J., Evans, R., Zhu, C., & Ming, C. (2024). AI in Medical Education: Global situation, effects and challenges. *Education and Information Technologies*, *29*(4), 4611–4633. https://doi.org/10.1007/s10639-023-12009-8

P. B. Liashchynskyi

Lviv Polytechnic National University, Lviv, Ukraine

SOFTWARE TOOL FOR SYNTHESIS AND CLASSIFICATION OF BIOMEDICAL IMAGES

The paper presents a software tool for classification and synthesis of biomedical images that has been developed in the course of research. The need for artificial expansion of data sets of biomedical images is established due to their limited availability, which creates an obstacle to the development of diagnostic tools. It was found that the developed software tool can solve this problem by generating synthetic, but realistic medical images, which can serve as additional data for the training of classifiers. Functional requirements for the software tool and its architecture were developed using modern programming and software design technologies. The software tool is designed using a modular architecture that allows each module to be scaled independently according to the load. The regularities of the software architecture are characterized, including client-server interaction, the MongoDB database and the use of the RabbitMQ message broker for asynchronous data exchange between the software modules. The main modules of the software tool are as follows: datasets (responsible for managing training images), classifiers (responsible for training and using convolutional neural networks for image classification), and generators (responsible for training and using generative adversarial networks for image synthesis). The software tool was developed using various programming languages such as Python, TypeScript and modern technologies, in particular, NodeJS, RabbitMQ, PyTorch, MongoDB, React. The database structure is also designed using a logical model based on a UML class diagram. The effectiveness of using convolutional neural networks and generative-adversarial networks for classification and synthesis of biomedical images, respectively, is shown. The conclusion is made about the scientific novelty and practical significance of the developed software tool, which represents new opportunities for medical diagnostics and research, providing flexibility and scalability in the synthesis and analysis of biomedical images.

Keywords: neural networks; neural networking methods; image classification; image synthesis; software architecture.