Проєкт на тему: “Детект ворожої наземної техніки”

...

Виконали студенти КН-322б, КН-323сб:
Єлізов Сергій Станіславович
Олексієнко Дмитро Віталійович
Волобуєв Дмитро Романович
Вакуленко Артем Володимирович
Наздрач Єгор Геннадійович

Научний керівник:
Бохан Констянтин Олександрович

Склад команди

Научний керівник - Бохан Констянтин Олександрович

DevOps - Єлізов Сергій Станіславович

Frontend - Вакуленко Артем Володимирович

Data Scientist - Волобуєв Дмитро Романович

Backend - Олексієнко Дмитро Віталійович

Project manager - Наздрач Єгор Геннадійович

Завдання проєкту

Завданням проєкту “Детект ворожої наземної техніки” є створення комплексної системи для автоматичного виявлення ворожих наземних об’єктів на основі відеопотоків з дронів, супутників або тепловізійних камер. Для цього було розроблено архітектуру проєкту, що охоплює всі ключові компоненти системи — від обробки відео до виводу результатів. Було налаштовано середовище Docker для забезпечення стабільної та кросплатформеної роботи. Основу системи становить модель, натренована на базі YOLOv11, здатна точно і швидко виявляти техніку на зображеннях. Також було реалізовано бекенд для обробки запитів і керування моделлю та фронтенд для візуалізації детекції у зручному інтерфейсі. У результаті всі елементи було інтегровано в єдину ефективну систему виявлення.

Девопс частина проєкту

Було реалізовано контейнеризацію всієї системи за допомогою Docker.

До складу середовища входять окремі сервіси, зокрема RTSP-стример для трансляції відеопотоків та база даних PostgreSQL для збереження результатів детекції.

Налаштовано повноцінне Python-середовище з усіма необхідними залежностями для роботи з моделлю YOLOv11, обробки відео та взаємодії з базою даних.

Усі компоненти об’єднані за допомогою docker-compose, що забезпечує централізоване керування, мережеву взаємодію та збереження даних через іменовані томи.

Для зручності управління командами запуску та зупинки сервісів, а також для виконання типових операцій з контейнерами використовувався Makefile.

Репозиторій проєкту розміщено на GitHub, що дало можливість організувати спільну роботу, зберігати історію змін і спростити розгортання системи на інших пристроях.

Труднощі при налаштуванні середовища

Основна складність виникла при налаштуванні стримінгу RTSP з контейнера через протокол UDP. Це пов’язано з архітектурними обмеженнями Docker на системах macOS та Windows, де використовується віртуалізоване мережеве середовище. У таких умовах передача UDP-пакетів може бути нестабільною або зовсім недоступною.

На Linux-системах ця проблема не виникає, оскільки Docker має прямий доступ до мережевого стеку. Якщо замість емульованого стримінгу використовувати реальне джерело (наприклад, камеру або зовнішній RTSP-сервер), система працює стабільно без додаткових змін.

Фронтенд частина проєкту

За використанням фреймворку Streamlit було створено інтерактивну панель для виводу відеопотоку:

- запит на джерело відеопотоку
- підключення до джерела
- оновлення панелі з актуальним джерелом
- ручне оновлення джерела
- виведення статистики для перевірки роботи нейронної мережі

Труднощі розробки фронтенду

Під час розробки Streamlit-сторінки для виведення відеопотоку з AI-розпізнаванням та статистики, я зіткнувся з кількома ключовими проблемами. Насамперед, оптимізація продуктивності була справжнім викликом. Одночасна обробка відео в реальному часі, виконання алгоритмів AI-розпізнавання та оновлення інтерфейсу Streamlit вимагали значних ресурсів, що призводило до затримок і зависань.

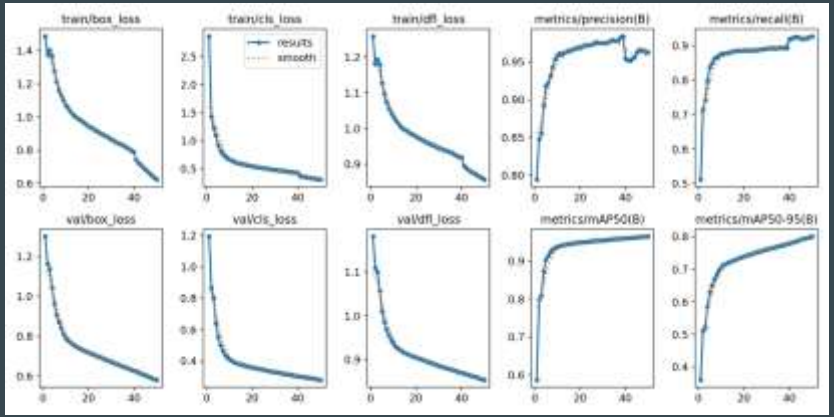
Крім того, виникли труднощі з керуванням станом застосунку. Підтримка синхронізації між відеопотоком, даними розпізнавання та елементами UI Streamlit, особливо при зміні користувачем параметрів, виявилася непростим завданням. Це вимагало ретельного планування архітектури даних та механізмів їх оновлення.

Тренування моделі

Модель була натренована на датасеті військової наземної техніки в якій присутні такі види техніки: артилерія, БТР, ЗРК, танк, військова наземна техніка.

Тренування моделі проходило впродовж 50 епох, що в общому обсязі зайняло приблизно 16 годин.

В результаті модель знаходить ворожу техніку з точністю 96%.



Труднощі з тренуванням моделі

Під час тренування моделі зінктувся із такими труднощами.

Датасет був розділений на 2 директорії, але YOLO11 потребує дані розподіленні на 3 дерикторії, а саме: train(дані для тренування), test(дані для тестування), val(дані для валідації).

Бекенд частина проєкту

У межах бекенд-частини проєкту була розроблена серверна система, яка забезпечує обробку відеопотоку та інтеграцію моделі YOLOv11 для виявлення ворожої наземної техніки.

Сервер отримує відео зі стороннього джерела через протокол HTTP (або RTSP), зчитує кадри в окремому потоці та зберігає їх у спільній змінній.

Для обробки зображень реалізовано два окремих маршрути: один транслює оригінальний потік, інший — оброблений мережею.

Функція detect() викликає модель YOLOv11, яка аналізує кожен кадр, знаходить об’єкти та повертає зображення з нанесеними рамками.

Уся логіка обробки кадрів побудована на генераторах потоків, що дозволяє підтримувати трансляцію у режимі реального часу без затримок.

Технічні труднощі при розробці бекенд-частини проєкту

Під час реалізації бекенд-частини проєкту виникло кілька технічних труднощів.

По-перше, були проблеми зі стабільним отриманням відеопотоку з зовнішнього джерела — деякі формати потоків не підтримувалися стандартним інтерфейсом OpenCV, що вимагало налаштування альтернативних джерел або проксі-серверів.

По-друге, виникали затримки при обробці кадрів у реальному часі, особливо під час одночасної обробки та трансляції, що потребувало оптимізації коду та перенесення частини логіки в окремі потоки.

Також довелось вирішити питання із синхронізацією доступу до кадрів, щоб уникнути збоїв при одночасному зчитуванні й обробці.

Координація проєкту

Проєкт складався з декількох важливих напрямків: бекенд, фронтенд, датасет, девопс. Завдання полягало в тому, щоб ці частини не існували окремо, а ефективно працювали разом як єдина система.

Було організовано регулярні синхронізації (регулярні робочі зустрічі, технічні брейнсторми), визначено критичні точки проєкту, сформовано графік з ключовими віхами — тренування моделі, запуск відеопотоку, вивід результатів, підготовка MVP.

Загальний вигляд робочого сайту

Детектор об'єктів з трекінгом

Топ-10 об'єктів

	Об'єкт	Кількість
0	tank	12
1	btr	8
2	vehicle	1

Висновки

У ході реалізації проєкту було створено комплексну систему виявлення ворожої наземної техніки на відео, що поєднує в собі передові технології комп’ютерного зору, контейнеризацію та веб-інтерфейс для користувача. Модель YOLOv11, натренована на спеціалізованому датасеті, продемонструвала високу точність (96%) при розпізнаванні об’єктів, а вся система стабільно працює в режимі реального часу.

Інфраструктура була реалізована з використанням Docker і docker-compose, що забезпечило зручне розгортання та сумісність між усіма компонентами. За допомогою Streamlit реалізовано інтерфейс, який дозволяє не лише переглядати відео з детекцією, а й аналізувати статистику виявлених об’єктів. Система успішно обробляє кілька потоків одночасно, а результати фіксуються у базі даних PostgreSQL.

Проєкт став результатом командної роботи, де кожен учасник відповідав за окрему частину — від збору та підготовки датасету до реалізації бекенду, фронтенду та налаштування інфраструктури. Завдяки спільній координації вдалося досягти високого рівня інтеграції між усіма модулями системи та вийти на стабільне MVP, яке вже зараз може застосовуватись у реальних умовах.