COMPUTER SYSTEMS AND NETWORKS

Vol. 6, No. 2, 2024

DOI: https://doi.org/10.23939/csn2024.02.242

УДК 004.8, 004.932

ОЦІНКА ПРОДУКТИВНОСТІ ТА ОПТИМІЗАЦІЯ МОДЕЛЕЙ НЕЙРОННИХ МЕРЕЖ YOLOv8 ДЛЯ РОЗПІЗНАВАННЯ ЦІЛЕЙ

Б. С. Цюник, О. В. Муляревич

Національний університет "Львівська політехніка" ,
кафедра електронних обчислювальних машин
E-mail: bohdan.s.tsiunyk@lpnu.ua, oleksandr.v.muliarevych@lpnu.ua

Метою цього дослідження є проведення всебічного аналізу продуктивності різних типів моделей нейронних мереж (НМ) для розпізнавання цілей. Зокрема, це дослідження зосереджується на оцінці ефективності та продуктивності моделей yolov8n, yolov8s, yolov8m та YOLO у завданнях розпізнавання цілей. Використовуючи передові засоби, такі як OpenCV, Python та roboflow 3.0 FAST, дослідження спрямоване на розробку надійної методології для оцінки продуктивності цих моделей нейронних мереж. Методологія включає розробку та впровадження експериментів для вимірювання ключових показників, таких як точність, швидкість та використання ресурсів. Завдяки ретельному аналізу це дослідження надає демонстрацію сильних та слабких сторін кожної моделі, сприяючи обґрунтованому прийняттю рішень для практичних застосувань. У цій статті представлено процес розробки та проведення аналізу продуктивності, підкреслюючи обґрунтування вибору конкретних методів та засобів. Крім того, в дослідженні представлено отримані результати тестування прототипу та огляд перспектив для майбутнього розвитку систем розпізнавання цілей.

Ключові слова: нейронні мережі (NN), розпізнавання цілей, yolov8, YOLO, OpenCV, модель НМ.

Вступ

У сучасному світі підтримка безпеки наших спільнот стала важливою, як ніколи. Життя та загальний добробут мешканців залежать від того, наскільки захищеним є їхнє оточення. Цей аспект є одним із фундаментальних стовпів існування кожної людини, оскільки він підкріплює сутність миру та стабільності. Важливість вирішення питань домашньої безпеки глибоко відображається у впровадженні різних заходів безпеки, що охоплюють кіберфізичні системи, системи розумного будинку, охоронні системи тощо. У ситуаціях, коли існує ризик для здоров'я або життя людини, вкрай важливо попередити та захистити тих, хто може бути вразливим до надзвичайних ситуацій, вторгнень у домівки або крадіжок, тому питання моніторингу за потенційними динамічними загрозами є актуальним.

Для забезпечення задачі моніторингу за рухомими цілями було використано підхід на базі звичайної програмно-апаратної системи моніторингу руху, в яку входить набір камер спостереження. Для автоматизації цієї системи було розглянуто перспектива використання сучасних підходів з використанням моделей нейронних мереж (НМ), які орієнтовані на ефективний аналіз та розпізнавання цілей. В межах цієї статті було вирішено сконцентруватись на сімействі нейронних мереж YOLO, а саме моделі: yolov8n, yolov8s, yolov8m та YOLO, як найбільш популярні для

опрацювання набору відеофреймів. Було проведено аналіз ефективності використання тої чи іншої моделі при виконанні практичного завдання – розпізнавання цілей на відеозображеннях.

Згідно з ADCIS, класична система розпізнавання цілей [1] складається з поєднання програмних утиліт, компонентів та апаратного забезпечення, спеціально розроблених для задоволення вимог різних груп користувачів у визначеній програмній екосистемі. Усі вони розміщені у спільному сховищі користувачів. Асоціація автоматизованого зображення (англ. Automated Imaging Association, AIA) визначає комп'ютерний зір як інтеграцію апаратного та програмного забезпечення для керування пристроями у виконанні завдань на базі захоплення та обробки зображень у промислових та непромислових застосуваннях [2]. Хоча промисловий комп'ютерний зір використовує алгоритми, спільні з академічними та урядовими секторами, він стикається з унікальними обмеженнями. Системи комп'ютерного зору орієнтовані на цифрові сенсори в промислових камерах, поєднаних зі спеціалізованою оптикою, для захоплення зображень, які використовуються для подальшої обробки, аналізу та прийняття рішень [3]. Таким чином, промисловий комп'ютерний зір вимагає досягнення низької вартості, підтримання прийнятної точності та забезпечення високої надійності, а також стійкості до механічного та термічного впливу.

Протягом багатьох років дослідження в галузі штучного інтелекту (ШІ) технології опрацювання інформації на базі комп'ютерного зору набули широко розповсюдження [4]: від банкоматів, які самостійно зчитують чеки, до камер, що автоматично фокусуються на обличчях, і соціальних мереж, які автоматично позначають друзів на основі розпізнавання обличчя. Останніми роками відбувся значний прогрес в автоматичному аналізі візуальних даних за допомогою алгоритмів комп'ютерного зору, що значною мірою зумовлено розвитком згорткових нейронних мереж (ЗНМ). Успіх цих моделей можна частково пояснити їх дизайном, наближеним до реального прототипу, тобто класичний штучний нейрон є спрощеною версією свого біологічного аналога [5].

Системи комп'ютерного зору широко використовують в різних сферах повсякденного життя. Вони забезпечують візуальне сприйняття та моделювання за допомогою обчислювальних інструментів. Ці системи використовують одно-, стерео- або багатокамерні налаштування для виконання візуальних завдань [6]. Стереосистеми зору, що використовують подвійні або багатокамерні конфігурації, дозволяють виконувати складні операції комп'ютерного зору. Системи візуалізації є важливими інструментами зображення, які використовують у в широкому спектрі застосувань, включаючи портативну автономну робототехніку, 3D-вимірювання, відстеження об'єктів, індустрію розваг, доповнену реальність та розпізнавання об'єктів [7]. Ці системи, подвійні за своєю природою, залежать від багатовимірного зору, відіграють ключову роль у покращенні просторового сприйняття та сприяють виконанню різноманітних візуальних завдань у різних секторах.

Огляд літературних джерел

Python виділяється як мова програмування, що втілює простоту та потужність, роблячи її улюбленим вибором серед розробників. Її інтуїтивна природа дозволяє користувачам зосередитися на вирішенні проблем, а не на боротьбі зі складним синтаксисом та структурою. Офіційне введення до Python підкреслює її простоту та ефективність, описуючи її як потужну мову програмування з високорівневими структурами даних та простим підходом до об'єктно-орієнтованого програмування. Елегантність Python полягає в її мінімалістичному дизайні, що нагадує англійську мову зі строгими правилами. Її зручний синтаксис робить її надзвичайно доступною навіть для початківців. З Python розробники звільняються від турбот про деталі низького рівня, такі як управління пам'яттю, завдяки її відкритому середовищу. Універсальність Python поширюється на багато платформ, включаючи GNU/Linux, Windows, macOS та інші, що робить її загальнодоступним засобом для розробки програмного забезпечення [8].

Програма, написана мовою компіляції, такою як C або C++, проходить процес, у якому вона перекладається з вихідного коду в двійковий код (0 та 1), який комп'ютер може розуміти, використовуючи компілятор разом із різними прапорцями та параметрами. Згодом програмне

забезпечення зв'язування/завантаження переміщує програму з жорсткого диска в пам'ять і починає її виконання після запуску. На відміну від цього, Python не вимагає бінарної компіляції. Замість цього ви можете безпосередньо виконувати програму з її вихідного коду. Внутрішньо Python перекладає вихідний код у проміжну форму, відому як байткоди, яка потім перетворюється в рідну мову комп'ютера перед виконанням. Привабливість Python полягає в його захопливому поєднанні продуктивності та функціональності.

Переваги технології на основі 3D-зору поширюються на численні галузі, включаючи 3D-вимірювання форми, візуальний огляд, медичну візуалізацію, керування роботами тощо. Вона слугує основним компонентом калібрування промислових камер для отримання внутрішніх та зовнішніх параметрів для подальших завдань вимірювання або виявлення. Точність цих вимірювань залежить від точності калібрування камери, особливо у високоточних промислових застосуваннях. Поширені методи калібрування камер, такі як пряма лінійна трансформація (DLT), метод Цая та планарний метод калібрування Чжана, використовуються для ефективної реконструкції локальної або повної 3D-профільної інформації. Хоча DLT створює базову перспективну модель камери, використовуючи 3D-точки в просторі, він часто не враховує спотворення лінз. Метод Цая, з іншого боку, включає радіальні обмеження вирівнювання для обчислення зовнішніх параметрів камери, після чого виконується нелінійна оптимізація для врахування радіальних спотворень. Метод Чжана, відомий своєю гнучкістю, накладає мінімальні обмеження на просторове розташування калібрувального об'єкта та цінується за свою адаптивність та ефективність у завданнях калібрування камер [9].

OpenCV (Open-Source Computer Vision Library) – це вільно доступна відкрита програмна бібліотека, призначена для завдань комп'ютерного зору та машинного навчання. Вона була започаткована з метою створення єдиної платформи для застосунків комп'ютерного зору та прискорення інтеграції машинного сприйняття в комерційні продукти. З ліцензією Apache 2, OpenCV пропонує компаніям простий спосіб використовувати та змінювати їх кодову базу. З більш ніж 2500 оптимізованими алгоритмами бібліотека містить широкий спектр як традиційних, так і найсучасніших алгоритмів комп'ютерного зору та машинного навчання. Ці алгоритми дозволяють користувачам виконувати різноманітні завдання, такі як виявлення та розпізнавання облич, ідентифікація об'єктів, класифікація дій людини у відео, відстеження руху камери, відстеження руху об'єктів, 3D-моделювання об'єктів, створення 3D-точкових хмар на основі стереокамери, об'єднання зображень для панорамних сцен, пошук подібностей зображень у базах даних, усунення ефекту червоних очей при зйомці зі спалахом, відстеження руху очей, розпізнавання сцен та встановлення маркерів для впливу на поведінку алгоритму. Із спільнотою користувачів понад 47000 та оціненими понад 18 мільйонами завантажень [10], OpenCV стала базовим інструментом, який використовують компанії, дослідницькі установи та урядові агенції по всьому світу. Більшість програм комп'ютерного зору включають введення та виведення зображень. Інтерактивні застосунки можуть вимагати камеру як джерело введення інформації та вікно відображення як місце призначення. Файли зображень, відеофайли та необроблені байтові дані можуть слугувати джерелами введення та виведення, що дозволяє реалізувати різноманітні сценарії застосування, включаючи мережеву передачу або генерацію процедурної графіки.

Останнім і вирішальним кроком у розробці застосунків є тестування. Воно дозволяє нам виявляти та виправляти поширені помилки на ранніх етапах розробки. Зазвичай цей вид роботи виконують тестувальники. Вони повинні симулювати всі можливі сценарії робочого процесу користувача, що дозволяє пройти через усі функції, які пропонує застосунок, та протестувати їх. Тестування продуктивності розробленої програми вимагає спеціальних інструментів. Одним із таких інструментів є Desktop Studio Profiler, який полегшує моніторинг використання ресурсів ЦП, споживання оперативної пам'яті, мережевої активності та її впливу на час роботи батареї [11]. Основною характеристикою моделі НМ для розпізнавання цілей є обсяг пам'яті, який повинен становити до 20 ГБ загального виділення пам'яті.

Постановка задачі

У контексті систем розпізнавання цілей існує критична потреба ретельно аналізувати продуктивність різних типів моделей нейронних мереж (НМ). Основним предметом цього дослідження є проведення глибокого вивчення ефективності та продуктивності різних архітектур НМ для завдань розпізнавання цілей. Це вимагає систематичної оцінки характеристик продуктивності моделей НМ, включаючи такі фактори, як точність, використання ресурсів та обчислювальна ефективність.

Розглянута проблема полягає у виборі та оптимізації моделей НМ для досягнення високої продуктивності в застосуваннях розпізнавання цілей. З огляду на різноманітність доступних архітектур НМ, відсутнє повне розуміння їх порівняльної продуктивності в реальних сценаріях. Тому основною метою цього дослідження є усунення цієї прогалини шляхом проведення детального аналізу моделей НМ, зосереджуючись на їх продуктивності та придатності до завдань розпізнавання цілей.

Основні виклики включають оптимізацію розподілу ресурсів, таких як використання пам'яті та навантаження на ЦП, щоб забезпечити ефективну роботу моделей НМ у системах розпізнавання цілей. Крім того, необхідно встановити еталонні показники для оцінки продуктивності, враховуючи такі фактори, як точність виявлення, швидкість обробки та масштабованість у різних середовищах. Систематично оцінюючи та порівнюючи продуктивність різних моделей НМ, це дослідження прагне надати цінну інформацію, яка допоможе в процесах прийняття рішень щодо вибору та розгортання систем розпізнавання цілей на основі НМ. Метою є визначення найбільш ефективної та продуктивної моделі НМ для розпізнавання цілей, що сприятиме прогресу в галузі спостереження, безпеки та суміжних сферах.

Мета роботи

Метою цієї роботи є систематична оцінка та порівняння ефективності та продуктивності різних моделей нейронних мереж (НМ) у сфері розпізнавання цілей. Завдання полягає у визначенні найбільш ресурсоефективної та дієвої моделі НМ для завдань розпізнавання цілей. Це дослідження прагне надати цінну інформацію про характеристики продуктивності різних архітектур НМ, таких як yolov8n, yolov8s та yolov8m, зосереджуючись на їх здатності точно та ефективно розпізнавати цілі в різних середовищах. Завдяки ретельному аналізу та експериментам метою є інформування процесів прийняття рішень щодо вибору та розгортання моделей НМ у реальних застосуваннях розпізнавання цілей, що в кінцевому підсумку сприятиме прогресу систем спостереження та безпеки.

Критичним аспектом для системи є оптимізація програми, особливо через інтенсивні обчислення та відтворення відеопотоку. Використання оперативної пам'яті, розподіл пам'яті та навантаження на ЦП часто становлять виклик у таких сценаріях. Зазвичай стандартна модель НМ для розпізнавання цілей виділяє до 20 ГБ пам'яті. Крім того, швидкий час обчислення є суттєвим, при цьому виявлення рухомих об'єктів зазвичай здійснюється протягом 0,1 мс. Результатом науково-дослідної роботи є знаходження найкращого рішення моделі НМ, яка відповідає всім вищезазначеним вимогам і забезпечує найкращий результат розпізнавання цілей.

Опис алгоритмів, що використовувались в межах розробленої системи

Головною метою та керівним принципом розробки реалізованої програмної системи було використання сучасних технологій комп'ютерного зору. Python був обраний як мова програмування для розробки всієї системи, а PyCharm IDE служить середовищем розробки. Після ретельного вивчення переваг та недоліків різних технологій було вирішено прийняти такий стек технологій для реалізації моделей нейронних мереж (НМ): Python, OpenCV, YOLO та PyCharm.

Алгоритм перетворення відео на основі потоку моделі нейронної мережі можна описати за схемою, наведеною на рис.1. На схемі видно шлях до конвертації потоку в дані для подальшої обробки за наступними послідовними кроками: підготовка даних для тренування (Custom Dataset),

тренування нейронної мережі для обробки відеопотоку на базі підготовлених даних (NN Model Training), використання отриманої NN Model для розпізнавання відеопотоку (Video Stream), яке в процесі обробки (CV Video Capture) перетворюється в об'єкти OpenCV (CV Out) для подальшого розпізнавання цілей.

Рис. 1. Схема конвертації відео потоку

Типовий алгоритм навчання моделі нейронної мережі (етап на схемі NN Model Training) розпочинається із збору та попередньої обробки різноманітного набору зображень, що містять релевантні цільові об'єкти для завдання розпізнавання. Наступним кроком потрібно додати обмежувальні рамки до набору даних, щоб вказати місцезнаходження та класові мітки цільових об'єктів. Після чого необхідно розділити набір даних на тренувальні, валідаційні та тестові набори для оцінки навченої моделі нейронної мережі. У межах проведених досліджень було побудовано модель на базі YOLOv8 – 8-ї версії базової архітектури YOLO.

Наступний етап – це імпорт попередньо навчених ваг в модель, щоб скористатися вивченими особливостями з великого набору даних. Потім настає період тонкого налаштування та адаптації попередньо навченої моделі YOLO до завдання розпізнавання конкретних цілей. Потрібно визначити, чи потрібно зафіксувати певні шари на зображенні або виконати повне навчання.

Потрібно вибрати відповідну функцію втрат, налаштовану для завдання розпізнавання цілей, та налаштувати функцію втрат, щоб підкреслити певні аспекти розпізнавання цілей. Для цього потрібно переглянути мініпакети зображень та відповідні анотації з тренувального набору.

Після чого потрібно виконати прямий прохід: подайте зображення в модель YOLO та обчисліть передбачувані обмежувальні рамки та класові ймовірності. Після обчислення втрати між передбаченими та фактичними анотаціями потрібно виконати зворотний прохід: оновити ваги моделі, використовуючи оптимізацію градієнтного спуску.

Процес навчання необхідно повторювати протягом кількох ітерацій. У процесі яких відбувається вдосконалювання гіперпараметрів, такі як швидкість навчання, розмір пакета та параметри оптимізатора. У підсумку проводиться експеримент для визначення оптимальних конфігурацій гіперпараметрів.

В результаті проводиться оцінка навченої моделі YOLO на валідаційному наборі даних, які було залишено на початку відбору даних. Проводиться оцінка метрик продуктивності, таких як точність (accuracy), прецизійність (precision), повнота (recall) та середня точність (mAP). Продуктивність оцінюється у різних сценаріях розпізнавання цілей. У межах проведених досліджень було проаналізовано продуктивність різних варіантів моделей YOLO для порівняння їх роботи. Акцент при вимірюванні відбувався на таких метриках, як швидкість виведення, використання пам'яті та точність.

Створення користувацького набору даних на базі реальних фото

Створення користувацького набору даних на основі реальних фотографій та його імпорт у процес навчання моделі нейронної мережі (НМ) є критичним кроком в аналізі продуктивності кількох типів моделей НМ для розпізнавання цілей. У зв'язку з бойовими діями щораз популярнішими стають методи розпізнавання класів та видів військової техніки, а саме виявлення та супровід цілей за допомогою нейронних мереж, а для цього необхідно створити датасет зображень.

Розпочинається із збирання різноманітного набору реальних фотографій, що відповідають завданням розпізнавання цілей. Необхідно переконатись, що зображення охоплюють широкий спектр сценаріїв та умов, з якими модель НМ може стикатися в реальних застосуваннях. Потрібно використовувати інструменти анотації для позначення об'єктів або цілей, що становлять інтерес на кожному зображенні.

Необхідно призначити відповідні мітки або класи анотованим об'єктам, щоб полегшити навчання під час тренування. Важливим є зміна розміру усіх зображень до стандартного розміру 640 x 640 пікселів для забезпечення однорідності та сумісності з архітектурою моделі НМ. У цьому дослідженні було використано понад 300 реальних фотографій військової техніки, а саме фотографії з різних ракурсів БТР-3, на базі яких було створено датасет для тренування моделі.

Крім того, потрібно виконати попередню обробку зображень за потреби, такі як нормалізація або аугментація, щоб підвищити надійність моделі та здатність до узагальнення. Розділити анотовані зображення на тренувальний, валідаційний та тестовий набори. Виділити значну частину для тренувального набору, щоб забезпечити достатній доступ до різноманітних прикладів. Валідаційний набір використовується для налаштування гіперпараметрів та оцінки моделі, тоді як тестовий набір оцінює кінцеву продуктивність моделі на невидимих даних.

Наступним кроком є організація анотованих зображень та відповідних міток у структурованому форматі, сумісному з популярними фреймворками глибокого навчання, такими як TensorFlow або PyTorch. Зазвичай це передбачає створення окремих директорій для тренувального, валідаційного та тестового наборів, причому кожна директорія містить піддиректорії для різних класів або міток. Необхідно реалізувати завантажувачі даних або генератори для ефективного завантаження пакетів зображень та міток під час процесу навчання, а також переконатись, що конвеєр завантаження даних оптимізований для продуктивності, щоб зменшити кількість втрат кадрів під час навчання моделі.

Після чого можна навчати модель НМ з користувацьким набором даних, створеним з реальних фотографій. Для цього використовуються найсучасніші архітектури, такі як YOLO (You Only Look Once), для ефективного та точного розпізнавання цілей. Демонстрацію стану розпізнавання військової техніки по зображенню з відеопотоку можна побачити на рис. 2.

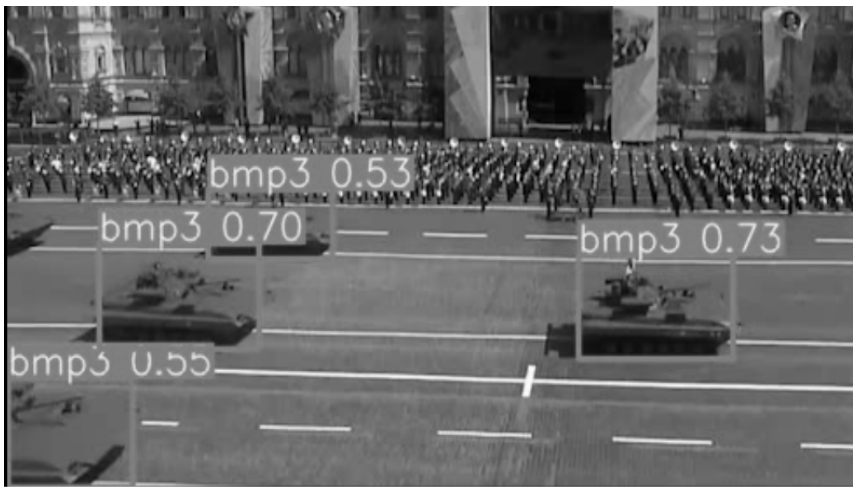

Рис. 2. Візуалізація стану розпізнавання рухомих цілей за допомогою YOLOv8S моделей

Тонке налаштування параметрів (калібрування моделі) та архітектури на основі метрик продуктивності, отриманих під час навчання, – один з фінальних етапів, який потребує оцінки продуктивності навченої моделі на валідаційних та тестових наборах, щоб отримати її точність, прецизійність, повноту та інші релевантні метрики. При потребі процес навчання можна повторити,

щоб підвищити продуктивність моделі та її здатність до вирішення практичної задачі. Метрики на основі різних моделей yolov8 для розпізнавання цілей зображено на рис. 3.

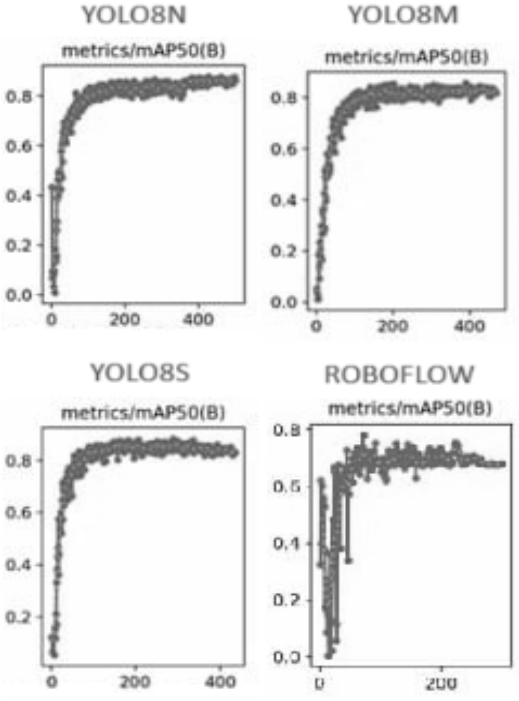

Рис. 3. Метрики нейронних мереж YOLOv8S моделей

Налаштування нейронної моделі на базі основних метрик

Процес налаштування починається з вибору відповідних метрик продуктивності для оцінки роботи моделі YOLOv8 у розпізнаванні цілей. Поширені метрики включають прецизійність, точність, повноту, F1-міру та середню точність (mAP). Ці метрики надають уявлення про здатність моделі точно виявляти та класифікувати цілі в різних середовищах. Проводиться навчання базової моделі YOLOv8, використовуючи стандартизований набір даних та гіперпараметри за замовчуванням. Це слугує початковим еталоном для оцінки продуктивності моделі перед етап тонкого налаштування.

Проводиться оцінка продуктивність базової моделі, використовуючи обрані метрики. Під час якого визначаються області для покращення та конкретні сценарії розпізнавання цілей, де модель може бути недостатньою. Згідно з отриманими результатами оцінки та аналізу, виконується тонке налаштування (калібрування) гіперпараметрів моделі YOLOv8 для оптимізації її продуктивності в розпізнаванні цілей за визначеними сценаріями. Це може включати налаштування таких параметрів, як швидкість навчання, розмір пакета, вибір оптимізатора, розмір зв'язуючих блоків та методи аугментації.

Наступним крок – реалізація стратегії аугментації даних для покращення здатності моделі розпізнавати нечіткі контури та чатково невидимі об'єкти, що дозволяє підвищити продуктивність в різних сценаріях розпізнавання цілей. Поширені методи аугментації включають випадкове обрізання, обертання, масштабування та варіації кольорів. Рекомендовано використовувати техніки перенесення навчання для ініціалізації моделі YOLOv8 з попередньо навченими вагами з більшого набору даних або спорідненого завдання. Важливо провести тонке налаштування (калібрування) цільової моделі нейронної мережі на наборі даних розпізнавання цілей, щоб адаптувати її особливості до специфічних характеристик цільових об'єктів. Слід використовувати техніки перехресної валідації для оцінки стійкості тонко налаштованої моделі YOLOv8 на різних підмножинах набору даних. Це допомагає забезпечити послідовність та надійність моделі в різних розподілах даних.

Після чого можна оцінити продуктивність налаштованої моделі YOLOv8, використовуючи обрані метрики валідації та тестові набори даних. Порівняти результати з базовою моделлю, щоб кількісно оцінити поліпшення в точності розпізнавання цілей та інших релевантних метриках. Ітерації процесу тонкого налаштування продовжуються, в доповненні з налаштуванням гіперпараметрів, стратегії аугментації та інших факторів на основі отримуваних результатів зміни оціночної продуктивності. Для цього потрібно постійно моніторити продуктивність моделі та вдосконалювати її конфігурацію для досягнення оптимальних результатів.

Важливим є крок документування процедури тонкого налаштування, детально описуючі обрані гіперпараметри, методи аугментації та метрики продуктивності. Результат цього етапу – детальний звіт з висновками та поліпшеннями, досягнутими шляхом тонкої настройки моделі YOLOv8 для розпізнавання цілей.

Результати досліджень

Для оцінки поточної системи було обрано ручне тестування через його гнучкість та надійність, що дозволяє всебічно виявляти та усувати помилки. Тестування продуктивності проводилося за допомогою інструменту Studio Profiler, інтегрованого в середовище розробки, що виключало потребу в зовнішніх інструментах.

Для всебічної оцінки продуктивності застосунку було проведено стрес-тестування, метою якого було оцінити стабільність системи при максимальному навантаженні. Цей тип тестування передбачає виконання операцій, що штовхають використання ресурсів застосунку до межі.

Під час стрес-тестування ресурсомісткі операції включали обчислення місцезнаходження рухомих об’єктів у всьому зображенні та перетворення живих кадрів з відеопотоку. Максимальне навантаження на ЦП становило 75 %, а використання оперативної пам’яті досягло 20 ГБ (див. рис. 4). Крім того, швидкість перетворення кадрів досягала до 0,0125 мілісекунди на кадр. Швидкість та ефективність цих моделей були майже ідентичними, тому загальна ефективність зводиться до ефективності використання ресурсів.

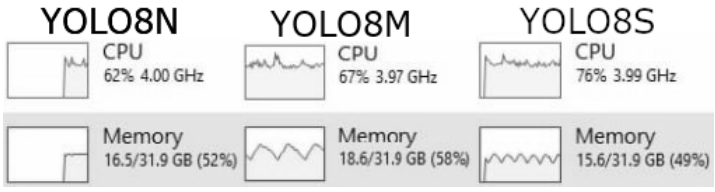

Рис. 4. Використання ресурсів нейронними моделями під час стрес-тестування

Ефективність ЦП можна знайти, віднявши від 100 % відсоток навантаження на ЦП:

$$E_{Ci} = 100\% - L_{CPUi}, \quad (1),$$

де L_{CPUi} – це навантаження ЦП, що спричиняє модель і. Використовуючи навантаження на ЦП у відсотках, поданих на рис. 4, отримуємо такі значення для моделей НМ: $E_{C8n} = 100 - 62 = 38$ %, відповідно: $E_{C8m} = 100 - 67 = 33$ %, і аналогічно для третьої моделі: $E_{C8s} = 100 - 76 = 24$ %.

Ефективність використання пам’яті можна виразити такою формулою аналогічно:

$$E_{Mi} = 100\ \% - L_{MEMi}, \quad (2),$$

де L_{MEMi} – це відсоток використання пам’яті, що потребує модель і. Відповідно до рис. 4, ми можемо представити такі значення для досліджуваних моделей НМ: $E_{M8n} = 100 - 52 = 48$ %, $E_{M8m} = 100 - 58 = 42$ % та $E_{M8s} = 100 - 49 = 51$ %, відповідно.

Порівняльну ефективність використання ресурсів можна знайти, знайшовши різницю між ефективністю за ЦП та пам’яттю за такою формулою:

$$E_{ij} = E_i - E_j, \quad (3),$$

де E_i – це ефективність моделі і, а E_i – це ефективність моделі j. Використовуючи тестові дані з рис. 4, отримуємо таку порівняльну ефективність використання ресурсів: $E_{C8nm} = 38 - 33 = 5$ %,

тобто yolo8n є на 5 % більш ефективним за ЦП, ніж модель yolo8m. Водночас $E_{C8ns} = 38 – 24 = 14$ %, та $E_{C8ms} = 33 – 24 = 9$ %.

Стосовно споживання пам'яті отримаємо: $E_{M8nm} = E_{M8n} – E_{M8m} = 48 – 42 = 6$ %, $E_{M8ns} = 48 - 51 = -3$ %, $E_{M8ms} = 42 – 51 = -9$ %.

Отже, модель yolo8m є на 9 % менш ефективною за споживанням пам'яті, ніж модель yolo8s. Порівняльна ефективність є двонаправленою, тому yolo8s також на 9 % більш ефективний у використанні пам'яті, ніж yolo8m. Таким чином, ці значення є всім, що нам потрібно для складання повного набору даних.

Порівняльну ефективність використання ресурсів можна знайти, усередниви ефективність ЦП та пам'яті:

$$E_{ij} = \frac{E_{Cij} + E_{Mij}}{2}, \quad (4),$$

де E_{Cij} – це порівняння ефективності за ЦП моделі по відношенню до моделі j та E_{Mij}, це порівняння ефективності за пам'яттю моделі, і по відношенню до моделі j.

Як результат можемо обчислити порівняння розглянутих моделей НМ:

$E_{8nm} = (5 + 6) / 2 = 5{,}5$ %, $E_{8ns} = (14 + 3) / 2 = 5{,}5$ %, $E_{8ms} = (9 + -9) / 2 = 0$ %.

Тобто моделі yolo8m та yolo8s мають практично ідентичну ефективність, в той час як модель yolo8n є на 5,5 % загалом більш ефективною порівняно з іншими двома моделями.

Висновки

В результаті цієї роботи були проведені всебічні дослідження різних моделей нейронних мереж (НМ) для розпізнавання цілей. Зокрема, yolov8n, yolov8s та yolov8m були ретельно протестовані, щоб визначити їх продуктивність у цій сфері.

Швидкість та ефективність цих моделей були майже ідентичними, тому загальна ефективність зводиться до ефективності використання ресурсів. Yolov8n в цілому був на 5,5 % більш ресурсоефективним, ніж інші варіанти моделей. Він також має на 5 % кращу ефективність використання ЦП порівняно з yolov8m $E_{C8nm} = 5$ %, та на 14 % кращу порівняно з yolov8s, $E_{C8ns} = 14$ %. Модель yolov8m використовує занадто багато пам'яті для незначного покращення ефективності ЦП: $E_{C8mn} = -5$ %, $E_{C8ms} = 9$ %, $E_{M8mn} = -6$ %, $E_{M8ms} = -9$ %. І має таку ж загальну ефективність, як yolo8s. У той час як yolo8s є найбільш ефективним за використанням пам'яті варіантом, але програє в ефективності за використанням ЦП іншим моделям: $E_{M8sm} = 9$ %, $E_{M8sn} = 3$ %, $E_{C8sm} = -9$ %, $E_{C8sn} = -14$ %.

Серед цих моделей yolov8s виявилася оптимальним вибором в межах цього дослідження, головним чином через її ефективність використання пам'яті. Оскільки тести проводили на універсальному ЦП, різниця у використанні ЦП була більшою, ніж була б на спеціалізованому ЦП з кращою багатопоточністю. Навпаки, оновлення типу пам'яті не вплинуло б на загальне споживання пам'яті. Отже, YOLOv8 була визнана оптимальним вибором над yolo8n, яка мала на 5,5 % кращу загальну ефективність. Прийняття кросплатформенного підходу та використання фреймворків Python сприяло бездоганній інтеграції та взаємодії з користувачем. Інтеграція сервісів OpenCV розширила функціональність, дозволяючи виконувати авторизацію та моніторинг у режимі реального часу.

Ключовим аспектом дизайну системи була її адаптивність, що дозволяє користувачам налаштовувати свої конфігурації моніторингу та відстеження. Ця гнучкість підкреслює корисність системи та орієнтованість на користувача. Крім того, були впроваджені суворі заходи оптимізації для усунення обмежень ресурсів, особливо щодо використання оперативної пам'яті при відтворенні відеопотоку.

Поточні моделі НМ відповідають вимогам до стандартних моделей розпізнавання цілей. Загалом усі моделі yolov8 відповідають основним вимогам щодо використання пам'яті, але найбільш ефективною для цього завдання є модель типу yolov8s.

Список літератури

1. *Das S., Saha S., Coello Coello, C. A., Bansal J. C. (2023). "Deep Neural Network Based Performance Evaluation and Comparative Analysis of Human Detection in Crowded Images Using YOLO Models". In International Conference on Advances in Data-Driven Computing and Intelligent Systems. ADCIS. Lecture Notes in Networks and Systems, vol 893. Springer, Singapore., p. 508–509. DOI: 10.1007/978-981-99-9518-9 37.*
2. *Delleji T., Slimeni F., Fekih H., (2022). "An Upgraded-YOLO with Object Augmentation: Mini-UAV Detection Under Low-Visibility Conditions by Improving Deep Neural Networks". Oper. Res. Forum 3, 60, p. 3–5. DOI: 10.1007/s43069-022-00163-7.*
3. *Tattari J., Donthi V. R., Mukirala D., Komar Kour S. (2021). "Deep Neural Networks Based Object Detection for Road Safety Using YOLO-V3". In Smart Computing Techniques and Applications. Smart Innovation, System and Technologies, vol 225. Springer, Singapore., p. 731–733. DOI: 10.1007/978-981-16-0878-0 71.*
4. *Poskar, B., Iskierka G., Krot K. (2024). "Logistics 4.0 – Monitoring of Transport Trolley in the Factory Through Vision Systems Using the YOLO Model Based on Convolution Neural Networks", In International Conference on Intelligent Systems in Production Engineering and Maintenance III. ISPEM 2023. Lecture Notes in Mechanical Engineering, Springer, Cham., p. 348–350. DOI: 10.1007/978-3-031-44282-7 27.*
5. *Priyankan K. and Fernando T.G.I., (2021). "Mobile Application to Identify Fish Species Using YOLO and Convolutional Neural Networks." In Proceedings of International Conference on Sustainable Expert Systems: ICSES 2020, volume 176. Springer, Singapore., p. 304–308. DOI: 10.1007/978-981-33-4355-9 24.*
6. *Ayob A. F., Khairuddin K., Mustafah Y. M., Salisa A. R., Kadir K. (2021). "Analysis of Pruned Neural Networks (MobileNetV2-YOLOv2) for Underwater Object Detection". In: Proceedings of the 11th National Technical Seminar on Unmanned System Technology 2019. NUSYS 2019. Lecture Notes in Electrical Engineering vol 666. Springer, Singapore., p. 87–88. DOI: 10.1007/978-981-15-5281-6 7.*
7. *A Byte of Python, (2022). [Electronic resource]. Available at: https://homepages.uc.edu/ ~becktl/byte of python.pdf. (Accessed: March 29, 2024).*
8. *Bansal J. C. and Uddin M. S. (2023). "Computer Vision and Machine Learning in Agriculture, Volume 3". In: Algorithms for Intelligent Systems., p. 120–125. DOI: 10.1007/978-981-99-3754-7.*
9. *Learning OpenCV, (2022). [Electronic resource]. – Available at: https://www.bogotobogo.com/ cplusplus/files/OReilly%20Learning%20OpenCV.pdf. (Accessed: March 29, 2024).*
10. *Huang D. S., Premaratne P., Jin B., Qu B., Jo K. H. and Hussain A., (2023). "Advanced Intelligent Computing Technology and Application". Springer, Singapore., p. 83–88. DOI: 10.1007/978-981-99-4742-3*
11. *Lys R., Opotyak Y. (2023). "Development of a Video Surveillance System for Motion Detection and Object Recognition". Advances in Cyber-Physical Systems, 8(1), p. 50–53, DOI: 10.23939/acps2023.01.050.*

PERFORMANCE EVALUATION AND OPTIMIZATION OF YOLOv8 NEURAL NETWORK MODELS FOR TARGET RECOGNITION

B. Tsiunyk, O. Muliarevych

Lviv Polytechnic National University,
Department of Computer Engineering
E-mail: : bohdan.s.tsiunyk@lpnu.ua, oleksandr.v.muliarevych@lpnu.ua

The objective of this research is to conduct a comprehensive performance analysis of various types of neural network (NN) models for target recognition. Specifically, this study focuses on evaluating the effectiveness and efficiency of yolov8n, yolov8s, yolov8m, and YOLO models in target recognition tasks. Leveraging cutting-edge technologies such as OpenCV, Python, and roboflow 3.0 FAST, the research aims to develop a robust methodology for assessing the performance of these NN models. The methodology includes the design and implementation of experiments to measure key metrics such as accuracy, speed, and resource utilization. Through meticulous analysis, this study aims to provide insights into the strengths and weaknesses of each model, facilitating informed decision-making for practical applications1. This paper presents the process of designing and conducting the performance analysis, highlighting the rationale behind the selection of specific technologies and methodologies. Furthermore, the study discusses the implications of the findings for future developments in target recognition systems.

Keywords: yolov8, YOLO, OpenCV, NN model.