УДК 004.032.26:004.93’1 **DOI:** 10.31673/2412-9070.2025.061207
М. М. ІЛАЩУК, аспірант;
ORCID: 0009-0002-7996-6176
С. В. МЕЛЬНИЧУК, д-р фіз.-мат. наук,
ORCID: 0009-0001-4325-2342
Чернівецький національний університет імені Юрія Федьковича

МЕТОДИКА РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ ОБЛИЧ У РЕАЛЬНОМУ ЧАСІ ЗА ДОПОМОГОЮ ЗГОРТКОВОЇ НЕЙРОННОЇ МЕРЕЖІ MOBILENETV3

У статті описано розроблену методику для детектування та розпізнавання облич у реальному часі. В основу розробленої методики покладено алгоритмічний конвеєр для паралельної обробки даних та їх обміну між трьома потоками: зчитування зображення з відеокамери, обробки нейронною мережею, візуалізації розпізнаного зображення. Основне покращення обробки даних отримано внаслідок використання двох моделей згорткової нейронної мережі (ЗНМ) з архітектурою MobileNetV3. Модель ЗНМ №1 була натренована для детектування облич, а модель ЗНМ №2 натренована для розпізнавання облич. На основі запропонованої методики розроблено програму на мові Python. Ключову увагу приділено оптимізації методики для досягнення мінімальних затримок при обробці кадрів. Проведені експерименти продемонстрували ефективність розробленої методики, яка забезпечує стабільну роботу на персональному комп’ютері навіть без графічного процесора. Отримано середню швидкодію 5.5 кадрів за секунду із затримкою 0.72 секунди. Запропонована методика є гнучкою до змін навантаженості операційної системи.

Ключові слова: розпізнавання зображень облич; згорткові нейронні мережі; обробка зображень; MobileNetV3, Python; машинне навчання; детектування; алгоритм; комп’ютерний зір.

Вступ

Задача автоматичного детектування обличчя на зображеннях та у відеопотоці є однією з ключових для комп’ютерного зору [1]. Її дослідження бере початок ще з 1960–1970-х років, коли перші алгоритмічні підходи ґрунтувалися на простих геометричних ознаках. Сучасний етап розвитку пов’язаний із використанням глибоких нейронних мереж, які, завдяки своїй здатності автоматично виділяти релевантні ознаки, значно підвищили точність і надійність систем детектування. Це дало змогу застосовувати такі алгоритми не лише у сфері безпеки та біометричної ідентифікації, а й у взаємодії людини з комп’ютером, медичних технологіях, доповненій реальності та багатьох інших галузях.

У даній роботі розглядається методика розпізнавання зображень облич, яка полягає у побудові цілісного алгоритмічного конвеєра для отримання відеопотоку з відеокамери, попередньої обробки кадрів, детектування та розпізнавання облич за допомогою штучних нейронних мереж, візуалізації результатів розпізнавання у реальному часі.

Постановка задачі

Дана робота має за мету розробку методики розпізнавання зображень облич у реальному часі за допомогою згорткової нейронної мережі (ЗНМ) з архітектурою MobileNetV3 [2], де початкові зображення облич зчитуються з відеокамер. Зчитування початкових зображень, детектування ділянок облич, розпізнавання облич та візуалізація результатів повинні виконуватися як етапи цілісного алгоритмічного конвеєра, який реалізується у вигляді багатопотокової програми. Дослідження передбачає використання попередньо натренованих моделей ЗНМ, зміну структури та тренування ЗНМ №1 для детектування ділянок облич на зображеннях, зміну структури та тренування ЗНМ №2 для розпізнавання облич певних осіб на виділених ділянках зображень. Для обробки зображень у реальному часі потрібно мінімізувати часові затримки

між моментом зчитування зображення з відеокамери та моментом візуалізації розпізнаного зображення. Також однією із задач дослідження є максимізація частоти обробки зображень, тобто параметра FPS (Frames Per Second - кількість оброблених зображень за секунду). Оскільки задача розпізнавання зображень за допомогою ЗНМ є доволі ресурсозатратною з точки зору використання центрального процесора, тому розроблена методика має бути гнучкою до зміни завантаженості операційної системи комп'ютера. Тобто, якщо інші програми будуть більше навантажувати операційну систему, то передбачається автоматичне зменшення показника FPS для розробленої програми розпізнавання облич.

Основна частина

Центральним ядром розробленої методики розпізнавання обличчя є згорткова нейронна мережа (CNN – convolution neural network) з архітектурою MobileNetV3, яка попередньо натренована для задачі розпізнавання облич. MobileNetV3 була відібрана як оптимальна для даної задачі, оскільки, порівняно з іншими моделями, вона потребує менше обчислювальних ресурсів на стадії розпізнавання (передбачення) [3]. За рахунок низьких вимог моделі MobileNetV3 до обчислювальних ресурсів вона може бути використана навіть на робочому комп'ютері, який не містить графічного процесора.

Проведено донавчання двох моделей згорткових нейронних мереж (CNN №1 та CNN №2) з архітектурою MobileNetV3. Модель CNN №1 призначена для детектування (виявлення) облич на зображенні. На виходах моделі CNN №1 отримуються координати прямокутних рамок, які обмежують зображення. Попереднє навчання моделі CNN №1 виконано на наборі даних ImageNet [4]. Для донавчання та валідації було використано 5791 та 1469 зображень відповідно з набору даних [5]. Тренування здійснювалося впродовж двадцяти епох. У результаті було збережено ЗНМ з найвищими показниками точності детектування обличчя.

Модель CNN №2 призначена для розпізнавання особи за зображенням обличчя. Особливістю CNN №2 є фіксація шарів нейронів, які відповідають за визначення положення об'єкту на зображенні. Вихідний шар CNN №2 призначений для класифікації осіб, тобто кожен його нейрон відповідає певній особі. У даній моделі для навчання використано зображення облич 30 осіб, які дали на це згоду. Тренування CNN №2 тривало протягом 50 епох. У результаті збережено модель CNN №2 з найкращими показниками точності розпізнавання на валідаційному наборі даних.

Програмну реалізацію розробленої методики виконано на мові програмування Python, а донавчання моделей CNN №1 та CNN №2 проведено за допомогою фреймворку TensorFlow [6]. Для доступу до загальних архітектур нейронних моделей комп'ютерного зору використано бібліотеку Keras-cv [7]. Навчання мереж виконувалося у хмарному онлайн-середовищі Google Colab, використовуючи графічний процесор Nvidia Tesla T4 з 15 ГБ відеопам'яті. Робоче середовище також мало 12.7 ГБ оперативної пам'яті. Натреновані моделі CNN №1 та CNN №2 у подальшому досліджено під час розпізнавання зображень у реальному часі на персональному комп'ютері з процесором AMD Ryzen 7 5700U, операційною системою Ubuntu 22.04, 16 Гб оперативної пам'яті та без використання графічного процесора.

Для обробки зображень, окрім нейронних мереж, також було розроблено програмний код, який у реальному часі зчитував зображення з веб-камери, відображав на відповідних кадрах результати детектування та розпізнавання за допомогою ЗНМ. На зображеннях-результатах будувався прямокутник навколо обличчя з підписом особи (якщо нейронній мережі вдалося встановити особу) або з надписом про те, що особа є невідомою.

Магістраль потоку даних у реальному часі

Оскільки моделі згорткових нейронних мереж, використані у даній роботі, вимагають значних ресурсів для обробки зображень, то це спричиняє складнощі під час використання моделей у реальному часі. Час обробки одного зображення є більшим, ніж час між зчитуванням кадрів відеопотоку. Оскільки більшість операцій, які виконуються нейронною мережею, є матричними, тому обробка зображень групами (banch) дозволяє значно пришвидшити розраху-

нки. Тобто, час на обробку одного зображення у групі буде у загальному менший, ніж якби кожне зображення оброблялося індивідуально. На рис. 1 показано залежність часу програмної обробки зображення (за розробленою методикою) від кількості зображень.

На рис. 1 видно, що час, який необхідний для обробки одного зображення, є більшим у випадку використання моделі ЗНМ для передбачення лише для однієї фотографії, а також суттєво зменшується у випадку групування фотографій. Час передбачення є найменшим для групи з трьох зображень, що становить 0.48 секунд для передбачення всієї групи і, відповідно, 0.16 секунд для передбачення одного зображення. Звідси можна оцінити максимальне значення для кількості зображень, що можуть бути оброблені за одну секунду (FPS параметр), як 6.25. Однак, у дійсності це значення може бути дещо меншим, оскільки у коді програми використовуються додаткові обчислення, які також вимагають ресурсів центрального процесора. Тому під час програмної обробки аналізується кожен 10 кадр відеопотоку. Отримані результати, що описують швидкодію обробки (рис. 1), можуть дещо відрізнятися у випадку застосування інших процесорів.

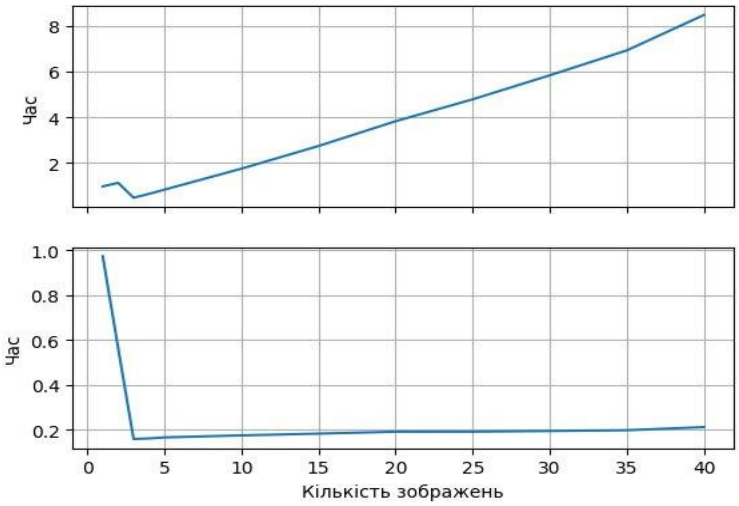

Рис. 1. Час (у секундах), який необхідний для обробки групи зображень (верхній графік), та середній час, який потрібний для обробки одного зображення (нижній графік)

Основною задачею розробленого цілісного алгоритмічного конвеєра обробки даних є забезпечення мінімальних затримок при відтворенні розпізнаного зображення (рис. 2). Для цього використано три паралельні потоки (thread) програми, які виконували різні задачі:

1. Потік камери. У даному потоці проводилося зчитування зображень веб-камерою через певні проміжки часу та зберігання їх у чергу (queue).

2. Потік штучних нейронних мереж. У даному потоці виконувалося вивільнення зображень з черги та їх обробка за допомогою моделей ЗНМ. На момент початку циклу в даному потоці програма вивільняє всі зображення у черзі, формує групи зображень та подає кожну групу на вхід CNN №1, а потім в CNN №2. Після закінчення роботи моделі нейронної мережі CNN №2 виконується візуалізація даних за допомогою бібліотеки OpenCV [8]. На кожному зображенні відображаються прямокутники навколо облич та надписи, які ідентифікують людей або вказують, що дана особа невідома (не міститься в базі). Найбільш складні обчислення відбуваються саме у потоці нейронних мереж, тому час між зчитуваннями кадрів у потоці відеокамери програма підбирає так, щоб дані були своєчасно оброблені. Після обробки фотографій вони додаються до черги оброблених зображень у тому ж порядку, у якому вони були взяті з початкової черги. За рахунок цього зберігається черговість відображення кадрів.

3. Потік візуалізації. У даному потоці зображення беруться з черги оброблених зображень та відображаються у графічному вікні через певні проміжки часу (зазвичай через такі ж проміжки часу, які були між знімками). Це забезпечує плавність зміни кадрів.

Мінімальні затримки при обробці кадрів забезпечуються навіть при зміні завантаженості операційної системи за рахунок адаптивного встановлення часу між знімками. Якщо операційна система більш завантажена і потік нейронної мережі не встигає обробити зображення, то кількість знімків у черзі стає більшою за оптимальний розмір групи, а час між знімками збільшується. Тобто в такому випадку для синхронізації частоти візуалізації з роботою нейронної мережі збільшується часовий проміжок між знімками, що призводить до зменшення показника FPS. І навпаки, якщо кількість зображень в черзі знімків стає меншою за оптимальний розмір групи, то показник FPS зростає.

На рис. 2 показана загальна структура роботи цілісного алгоритмічного конвеєра обробки даних. З рисунку видно, що між моментом зйомки та відображенням відповідного обробленого знімку на часовій шкалі розташовуються два періоди обробки групи зображень потоком нейронної мережі. Попередньо визначено, що час аналізу нейронною мережею однієї групи з трьох зображень рівний 0.48 секунд. У такому разі варто очікувати затримку між знімком та зображенням приблизно у 2 рази більшою за час обробки групи. Це пов'язано з тим, що у момент зчитування зображення починається обробка нейронною мережею попередньої групи, після якої розпочинається обробка поточної групи (в якій знаходиться дане зображення). Лише після аналізу зображень нейронною мережею відбувається їх візуалізація. У випадку використання груп з більшою кількістю зображень час обробки групи потоком нейронної мережі зростає пропорційно (рис. 1), а відповідно зростає затримка між зчитуванням зображення з камери та його візуалізацією. Проте, це не вплине на частоту відображення кадрів (FPS), оскільки всі потоки виконуються паралельно. Послідовні кроки обробки кожного зображення наведені на рис. 3.

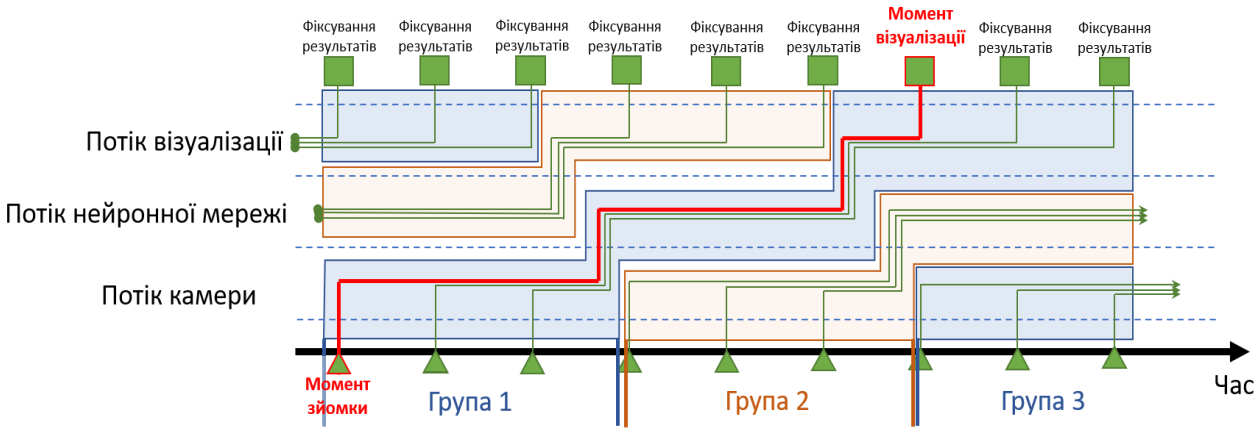

Рис. 2. Схематичне зображення цілісного алгоритмічного конвеєра. Зеленими лініями відображено рух зображень. Помаранчевим та блакитним фоновим кольором розділено різні групи зображень

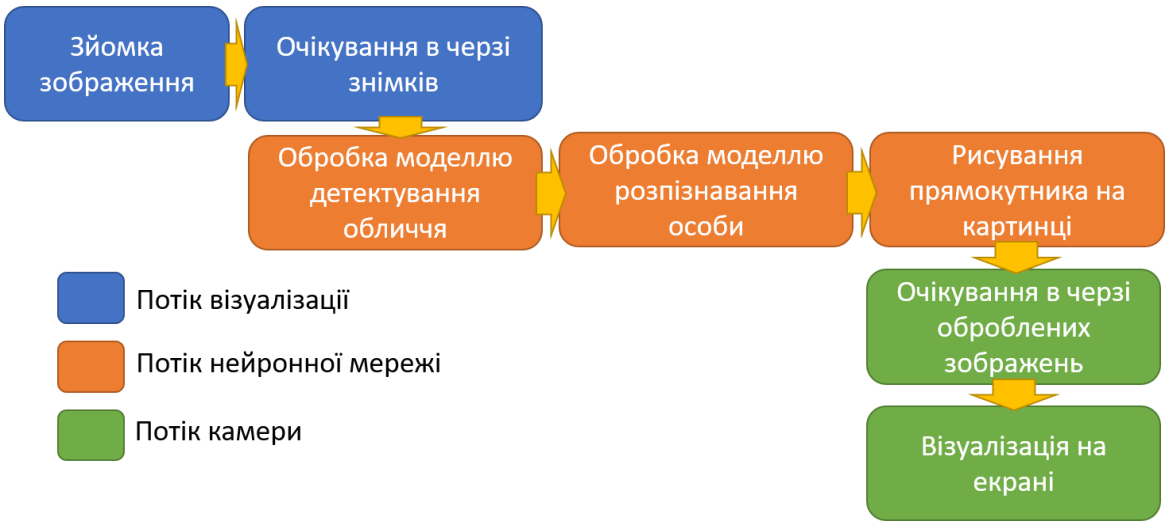

Рис. 3. Послідовне відображення кроків, які проходить кожне зображення від моменту зйомки до моменту його відображення на екрані

В алгоритмі програми також враховано випадок, коли часові проміжки між знімками камери є рівними часовим проміжкам між моментами візуалізації, що приводить до синхронізації з роботою потоку нейронної мережі. У такому випадку алгоритмічний конвеєр перебуває у стаціонарному стані. Однак, якщо до настання цього стану черга оброблених зображень є більшою, ніж черга початкових знімків, то це спричинить сталу затримку. Для уникнення такої

затримки програма буде зменшувати проміжок між змінами кадрів візуалізації до того моменту, поки ці черги не зрівняються. Проміжок між знімками при цьому залишиться без змін. Таким чином, черга оброблених зображень буде звільнятися швидше, ніж наповнюватися, тому кількість зображень у ній зменшиться. Для забезпечення плавної роботи конвеєра (мінімальних затримок кадрів при їх візуалізації) черга початкових знімків і черга оброблених зображень повинні бути приблизно однаковими. У такому випадку часові проміжки між візуалізацією та між зчитуванням знімків також мають бути рівними. Даний метод уникнення сталої затримки кадрів реалізовано у коді розробленої програми.

Тестування розробленої програми за параметрами точності та швидкодії

Точність детектування та розпізнавання зображень облич за допомогою розробленої прог-рами, яка реалізує запропоновану методику, перевірено на персональному комп'ютері з вико-ристанням веб-камери та без застосування графічного процесора. Відеокамера забезпечувала зчитування 30 кадрів за секунду. Під час тестування програма забезпечувала незначні затри-мки кадрів, що візуально сприймалося як плавна зміна кадрів, а детектування та розпізнавання облич у більшості випадків виконувалося правильно (рис. 4). Програма автоматично синхроні-зувала роботу конвеєра потоку даних, який досягнув сталого режиму роботи при показнику FPS = 5.5. У такому випадку затримка між зйомкою кадру та відображенням була рівна 0.72 секунди, а черги знімків та оброблених зображень в усталеному режимі містили від двох до чотирьох зображень.

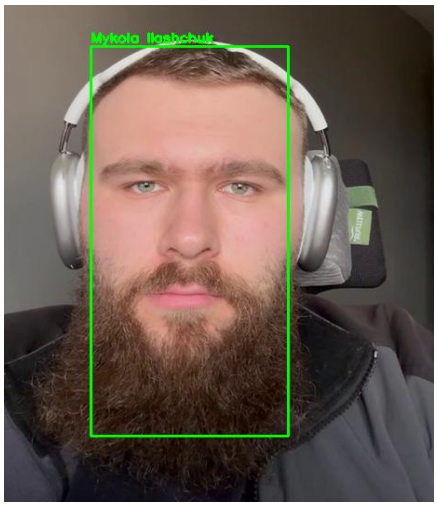

Рис. 4. Приклад візуалізації програмою оброблених знімків (розміром 2048 × 1536 пікселів) у реальному часі

Отримані результати порівняно з результатами попередніх досліджень [9], в яких вико-ристано бібліотеку OpenCV, як основне ядро детектування та розпізнавання обличчя. Порівняльні характеристики досліджень відображені у таблиці.

Порівняльні характеристики методів та методик розпізнавання облич

Кількість даних, відео	Точність розпізнавання, %	Час підготовки до розпізнавання	Якість візуалізації відео з розпізнаним обличчям
Результати розпізнавання облич за допомогою методів OpenCV [9]			
10	100	1 хвилина	висока
20	90,5	4 хвилини	висока
30	81	7 хвилин	висока
Результати розпізнавання облич за допомогою розробленої методики			
30	98,26	0,72 секунди	висока

Висновки

Дана робота дозволила підвищити точність та швидкодію обробки кадрів відеопотоку, яка полягала у детектуванні облич та розпізнаванні осіб у реальному часі. В основу розробленої методики покладено алгоритмічний конвеєр для паралельної обробки даних та їх обміну між трьома потоками: зчитування зображення з відеокамери, обробки нейронною мережею, візуалізації розпізнаного зображення. Основне покращення обробки даних отримано внаслідок використання моделей згорткової нейронної мережі з архітектурою MobileNetV3. Модель ЗНМ №1 була натренована для завдань детектування облич, а модель ЗНМ №2 - для розпізнавання облич. Розроблена методика розпізнавання облич може в подальшому застосовуватися як частина системи комп'ютерного зору, призначеної для розпізнавання емоцій людини.

На основі запропонованої методики розроблено програму на мові Python, яку протестовано у реальних умовах. Програма працювала стабільно на персональному комп'ютері з процесором AMD Ryzen 7 5700U та 16 Гб оперативної пам'яті під контролем Desktop версії операційної системи Ubuntu 22.04. Контрольні заміри ефективності показали, що розроблена програма здатна обробляти 5.5 зображень на секунду із затримкою 0.72 секунди між подією та її проаналізованим відображенням. Дані обмеження швидкодії спричинені фізичним лімітом обчислювальної потужності центрального процесора, оскільки в даному випадку графічний процесор не використовувався. Розроблена методика забезпечує адаптивність до змін завантаженості операційної системи і показала здатність змінювати показник FPS залежно від кількості наявних обчислювальних ресурсів, зберігаючи при цьому стабільність роботи.

Список літератури

1. S. Popereshnyak, R. Skoryk, D. Kuptsov, and R. Kravchenko, "Software Tool for Recognition of Human Face in Video Stream," UkrPROG, Kyiv, Ukraine, May 14-15, 2024, vol. 3806, CEUR-WS, ISSN 1613-0073, 2024.

2. A. Howard et al., "Searching for MobileNetV3," in Proc. IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 1314–1324.

3. J. A. Rachman, O. Santoso, M. Destianti and R. I. Wahyuningtyas, "Lightweight Face Anti-spoofing for Improved MobileNetV3," Journal of Information, Communication and Computer Technology, vol. 7, no. 1, pp. 117-131, Dec. 2024.

4. O. Russakovsky et al., "ImageNet Large Scale Visual Recognition Challenge," Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, 2015.

5. F. Elmenshawii, "Face-Detection-Dataset," Kaggle, [Онлайн]. Available: https://www.kaggle.com/datasets/fareselmenshawii/face-detection-dataset. [Доступ: Jul. 29, 2025].

6. M. Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," arXiv preprint arXiv:1603.04467, 2016.

7. M. Watson et al., "KerasCV and KerasNLP: Vision and Language Power-Ups," J. Mach. Learn. Res., vol. 25, no. 375, pp. 1–10, 2024.

8. G. Bradski, "The OpenCV Library," Dr. Dobb's Journal: Software Tools for the Professional Programmer, vol. 25, no. 11, pp. 120-123, 2000.

9. М. Ілащук, І. Кушнір, С. Мельничук "Розпізнавання облич в реальному часі за допомогою бібліотеки OpenCV та мови програмування Python," Herald of Khmelnytskyi Natl. Univ., no. 341, pp. 5–21, 2024, ISSN 2307-5732, doi: 10.31891/2307-5732-2024-341-5-21.

M. Ilashchuk, S. Melnychuk

REAL-TIME FACE IMAGE RECOGNITION METHOD USING CONVOLUTIONAL NEURAL NETWORK MOBILENETV3

This article describes a developed methodology for real-time face detection and recognition. The methodology is based on an algorithmic pipeline for parallel data processing and exchange between three threads: video camera image acquisition, neural network processing, and recognized image visualization.

The study involved the development of a full real-time data processing pipeline that integrates image acquisition from a webcam, preprocessing of frames, neural network inference, and visual display of recognition results. The MobileNetV3 model was selected due to its balance between recognition accuracy and computational efficiency. . The main improvement in data processing is achieved through the use of two Convolutional Neural Network (CNN) models with the MobileNetV3 architecture. CNN model #1 was trained for face detection, and CNN model #2 was trained for face recognition. Based on the proposed methodology, a software application was developed using the Python

language. Key attention was focused on optimizing the methodology to achieve minimal latency during frame processing.

To overcome the latency inherent to CPU-only systems, a multi-threaded software architecture was designed, consisting of three independent threads responsible for image capture, neural network inference, and visualization. The conducted experiments demonstrated the effectiveness of the developed methodology, which ensures stable operation on a personal computer even without a Graphics Processing Unit (GPU). An average performance of 5.5 frames per second (FPS) with a latency of 0.72 seconds was obtained. The proposed methodology is flexible to changes in the operating system load.

Keywords: facial image recognition; convolutional neural networks; image processing; MobileNetV3; Python; machine learning; detection; algorithm; computer vision.