

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Кваліфікаційна наукова
праця на правах рукопису

Денісов Ростислав Віталійович

УДК 621.397.6

ДИСЕРТАЦІЯ

СИСТЕМА РОЗПІЗНАВАННЯ ОБ'ЄКТІВ І ГОЛОСОВОГО СПОВІЩЕННЯ ДЛЯ ЛЮДЕЙ З ВАДАМИ ЗОРУ НА ОСНОВІ НЕЙРОННИХ МЕРЕЖ ТА МІКРОКОНТРОЛЕРІВ

17 - Електроніка та телекомунікації
171 Електроніка

Подається на здобуття наукового ступеня доктора філософії.

Дисертація містить результати власних досліджень. Використання ідей,
результатів і текстів інших авторів мають посилання на відповідне джерело
_____/Денісов Р.В.

Науковий керівник: Попович Павло Васильович, кандидат технічних наук,
доцент

Київ – 2025

АНОТАЦІЯ

Денісов Р.В. Система розпізнавання об'єктів і голосового сповіщення для людей з вадами зору на основі нейронних мереж та мікроконтролерів. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 171 "Електроніка". – Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського", МОН України, Київ, 2025.

В дисертаційній роботі вперше отримано наступні наукові результати:

1. Вперше досліджено можливості застосування мікроконтролерів та одноплатних комп'ютерів у поєднанні з нейронними мережами для створення систем розпізнавання об'єктів з подальшим голосовим сповіщенням для покращення орієнтації у просторі, та підвищення рівня мобільності людей з вадами розу.

2. Виконано розрахунок технічних параметрів, та виконано підбір компонентів на їх основі для варіацій систем, що базуються на різних мікроконтролерах та одноплатних комп'ютерах.

3. Вперше запропоновано метод попередньої обробки тренувальних даних для підвищення точності розпізнавання об'єктів, що мають надлишкову інформацію, або об'єктів з різних категорій, які можуть знаходитися поруч один з одним на тренувальних даних та при практичному використанні.

4. Виконано навчання і перевірку точності та швидкості розпізнавання нейронних мереж MobileNet для розпізнавання п'яти категорій об'єктів, а саме "windows", "door", "trees", "traffic lights", "crosswalk" з необробленими та змішаними тренувальними даними.

5. Вперше визначено умову застосування методу попередньої обробки для різних категорій об'єктів розпізнавання, яка полягає у порівнянні площі надлишкової інформації з корисною інформацією в межах кадру.

6. Розраховано час необхідний на проходження одного повного циклу розпізнавання-оголошення інформації з урахуванням особливостей Української

мови та мовлення, часу необхідного на оголошення різних комбінацій слів, швидкості реакції людини на голосову інформацію та часу розпізнавання об'єктів нейронними мережами.

Дисертаційна робота присвячена дослідженню та практичному опису можливості застосування системи розпізнавання об'єктів у режимі реального часу з подальшим голосовим сповіщенням для людей з вадами зору.

Дисертаційне дослідження представлене в чотирьох розділах, в яких обґрунтовані та представлені основні результати роботи.

У вступній частині обґрунтовано актуальність роботи, сформульовано мету та задачі дослідження, наведено методи дослідження, представлена інформація про наукову новизну, а також практичне значення результатів.

У першому розділі виконано огляд етапів процесу розпізнавання зображення, розглянуто як класичні, так і сучасні методи та алгоритми, які застосовуються під час попередньої обробки зображень, виділення ознак, сегментації об'єктів та пост обробці. Розглянуто архітектури сучасних нейронних мереж, а також доступні на ринку пристрої які призначені для людей з вадами зору.

У другому розділі представлено результати аналізу доступних складових системи розпізнавання об'єктів з подальшим голосовим виводом інформації для людей з вадами зору. Встановлено, що оптимальними нейронними мережами для розгортання на пристроях з обмеженими ресурсами є клас мереж MobileNet. Основними перевагами є можливість стиснення розмірів моделі до необхідного, завдяки коефіцієнтам ширини мережі і зміні розміру вхідного зображення, без значних втрат у швидкості і точності розпізнавання. У якості платформи для навчання та експортування нейронних мереж під обрані мікроконтролери, враховуючи функціонал та постійний розвиток обрано платформу Edge Impulse. У якості синтезатора мови оптимальним рішенням є eSpeak NG.

Встановлено, що мікроконтролери та одноплатні комп'ютери є оптимальним рішенням для створення необхідних систем. Вони мають невеликий форм фактор та вагу, а також мають достатньо потужності для

виконання задач з розпізнавання об'єктів в режимі реального часу. Для опису варіацій систем розпізнавання об'єктів з подальшим голосовим виводом для людей з вадами зору обрано такі плати як: ESP32-S3-EYE, Raspberry Pi 5 та NVIDIA Jetson Nano. Плати мають різні технічні характеристики і можуть забезпечити різний рівень швидкості розпізнавання об'єктів, і загальну продуктивність системи.

У третьому розділі виконано оцінку часу необхідного на один цикл процесу розпізнавання-оголошення інформації враховуючі швидкість реакції людини на голосову інформацію, та швидкість проходження різних етапів самого процесу. Встановлено час, необхідний на оголошення слів та їх комбінацій різної довжини з врахуванням особливостей Української мови та мовлення. Встановлено, що мінімальний час необхідний на оголошення одного слова становить 129 мс, у той час, як комбінація назва з трьох довгих слів може займати дві секунди на оголошення з урахуванням паузи між словами.

Виконано розрахунок і підбір компонентів для варіантів системи на платах ESP32-S3-EYE, Raspberry Pi 5 та NVIDIA Jetson Nano. Розраховано світловий потік який має забезпечити джерело світла, для нормального функціонування системи при поганому освітленні, та обрано світлодіод Cree XP-G3.

Для оголошення інформації обрано динаміки LD-SP-UM20/8A, що мають компактні розміри, сумісні з обраними платами, забезпечують необхідний рівень гучності. Також, було обрано звуковий підсилювач PAM8403 для підключення динаміків до системи. А також модулі камер для плат, в яких вона відсутня у базовій комплектації.

Розраховано загальну споживану потужність систем у різних комплектаціях з урахуванням роботи нейронних мереж, джерела світла і постійної роботи динаміків. На основі отриманої споживаної потужності було розраховано ємність джерела живлення та обрано оптимальні варіанти для забезпечення автономної роботи системи впродовж 3х годин.

У четвертому розділі проведено дослідження точності та швидкості розпізнавання 5 обраних категорій об'єктів, а саме "windows", "doors", "trees",

"traffic light", "crosswalk", нейронними мережами MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 для мікроконтролерів ESP32-S3-EYE, NVIDIA Jetson Nano та Raspberry Pi 5 було проведено декілька експериментів з необробленими та змішаними тренувальними зображеннями.

Для необроблених тренувальних даних отримано середні показники точності для мереж MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35 у 63% та 90%, а для мереж MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 отримано точність розпізнавання у 100%. При цьому, прогнозований час розпізнавання для мікроконтролера ESP32-S3-EYE у випадку моделей MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 становив більше 10 секунд, що є неприйнятним для використання у системах розпізнавання в режимі реального часу.

Для плат NVIDIA Jetson Nano та Raspberry Pi 5 середній прогнозований час розпізнавання становить від 3 мс для моделей MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, та від 14 мс до 33 мс у випадку мереж MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0.

Після змішування оброблених і необроблених тренувальних даних отримано приріст у точності розпізнавання для більшості категорій, але для категорії "windows" отримано погіршення точності розпізнавання. Після зворотної заміни комбінованих тренувальних даних у категорії "windows" на необроблені, отримано приріст у точності розпізнавання для мережі MobileNetV1 96x96 0.2 до 80 % для плат NVIDIA Jetson Nano та Raspberry Pi 5 та приріст до 73,3% для плати ESP32-S3-EYE. Для мережі MobileNetV2 96x96 0.35 було отримано точність розпізнавання у 96,7% для всіх плат.

Також встановлено мінімальний та максимальний час необхідних на проходження повного циклу розпізнавання-оголошення результатів з урахуванням особливостей Української мови та швидкості реакції людиною на слухову інформацію. Мінімальний час становить 379 мс у випадку назви об'єкту що складається з одного короткого слова, та 2198 мс у випадку для назви об'єкту з трьох довгих слів.

Практичне значення отриманих в дисертаційній роботі результатів можуть бути використані для проектування та створення адаптивної системи розпізнавання об'єктів у режимі реального часу з подальшим голосовим сповіщення користувача для людей з вадами зору або повною сліпотою. Отримано мінімальний та максимальний час проходження одного повного циклу розпізнавання-оголошення. Мінімальний час становить 379 мс для короткої назви об'єкту, що складається з одного слова, з урахуванням часу реакції людини на слухову інформацію, та максимальний час 2198 мс для назви об'єкту, що складається з трьох довгих слів, з урахуванням пауз між словами.

Виконано розрахунок необхідної сили світлового потоку та обрано світлодіод Cree XP-G3, для забезпечення роботи системи під час поганого освітлення, також обрано динаміки та підсилювач, розраховано загальну споживану потужність варіацій систем для плат ESP32-S3-EYE, Raspberry Pi 5 та NVIDIA Jetson Nano, розраховано необхідну ємність джерела живлення для забезпечення автономної роботи пристрою впродовж 3 годин.

Ключові слова: розпізнавання об'єктів, попередня обробка зображень, нейроні мережі, мікроконтролер, одноплатний комп'ютер, MobileNet, Edge Impulse, тренувальні дані, голосове сповіщення, мовна модель, точність розпізнавання, хибне розпізнавання, автономність.

SUMMARY

The following scientific results were obtained for the first time in the dissertation:

1. For the first time, the possibilities of using microcontrollers and single-board computers in combination with neural networks to create object recognition systems with subsequent voice notification to improve spatial orientation and increase the level of mobility of people with visual impairments have been investigated.

2. The technical parameters were calculated, and components were selected based on them for system variations based on different microcontrollers and single-board computers.

3. A method of preliminary processing of training data in Adobe Photoshop was proposed for the first time to increase the accuracy of recognizing objects with similar redundant information or objects from different categories that may be located close to each other in training data and during practical use.

4. Neural networks were trained and tested for recognition accuracy and speed for five categories of objects, namely "windows", "door", "trees", "traffic lights", "crosswalk" with raw and mixed training data.

5. For the first time, the condition for applying the preprocessing method for different categories of recognition objects has been defined, which consists in comparing the area of redundant information with useful information within the frame.

6. The time required for a complete recognition-announcement cycle was calculated, taking into account the peculiarities of the Ukrainian language and speech, the time needed to announce different word combinations, human reaction speed to voice information, and the time for object recognition by neural networks.

The dissertation is devoted to the research and practical description of the possibilities of applying an object recognition system in real-time with subsequent voice notification for visually impaired people.

The dissertation research is presented in four chapters, which substantiate and present the main results of the work.

The introduction substantiates the relevance of the work, formulates the goal and objectives of the research, presents the research methods, provides information on scientific novelty, and the practical significance of the results.

The first chapter reviews the stages of the image recognition process, considers both classical and modern methods and algorithms used during image preprocessing, feature extraction, object segmentation, and post-processing. The architectures of modern neural networks and available devices on the market designed for visually impaired people are reviewed.

The second chapter presents the results of the analysis of available components of the object recognition system with subsequent voice output of information for visually impaired people. It was established that the optimal neural networks for deployment on resource-limited devices are the MobileNet class. The main advantages are the ability to compress model sizes to the required level due to network width coefficients and changing the input image size without significant losses in recognition speed and accuracy. The Edge Impulse platform was chosen as the platform for training and exporting neural networks to selected microcontrollers, considering its functionality and continuous development. eSpeak NG was chosen as the optimal speech synthesizer.

It was established that microcontrollers and single-board computers are the optimal solution for creating the required systems. They have a small form factor and weight and are powerful enough to perform object recognition tasks in real-time. The object recognition system variations with subsequent voice output for visually impaired people were described using boards such as ESP32-S3-EYE, Raspberry Pi 5, and NVIDIA Jetson Nano. The boards have different technical characteristics and can provide different levels of object recognition speed and overall system performance.

The third chapter evaluates the time required for one cycle of the recognition-announcement process, considering human reaction speed to voice information and the speed of various process stages. The time required to announce words and their combinations of different lengths, taking into account the peculiarities of the Ukrainian language and speech, was determined. It was established that the minimum time

required to announce one word is 129 ms, while a combination of three long words may take up to two seconds to announce, considering pauses between words.

The components for system variants on ESP32-S3-EYE, Raspberry Pi 5, and NVIDIA Jetson Nano boards were calculated and selected. The luminous flux that the light source should provide for normal system operation in low-light conditions was calculated, and the Cree XP-G3 LED was chosen.

LD-SP-UM20/8A speakers, which have compact sizes, are compatible with the selected boards and provide the required sound level, were chosen for information announcement. A PAM8403 audio amplifier was selected to connect the speakers to the system, as well as camera modules for boards without a built-in camera.

The total power consumption of the systems in different configurations was calculated, taking into account the operation of neural networks, light sources, and continuous speaker operation. Based on the obtained power consumption, the battery capacity was calculated, and optimal options were chosen to ensure the autonomous operation of the system for 3 hours.

The fourth chapter conducted studies on the accuracy and speed of recognizing 5 selected categories of objects, namely "windows", "doors", "trees", "traffic light", "crosswalk" by MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, MobileNetV2 160x160 0.5, and MobileNetV2 160x160 1.0 neural networks for ESP32-S3-EYE, NVIDIA Jetson Nano, and Raspberry Pi 5 microcontrollers. Several experiments were conducted with raw and mixed training images.

For raw training data, the average accuracy indicators for MobileNetV1 96x96 0.2 and MobileNetV2 96x96 0.35 networks were 63% and 90%, respectively, while MobileNetV2 160x160 0.5 and MobileNetV2 160x160 1.0 achieved 100% recognition accuracy. However, the predicted recognition time for the ESP32-S3-EYE microcontroller with MobileNetV2 160x160 0.5 and MobileNetV2 160x160 1.0 models exceeded 10 seconds, which is unacceptable for real-time recognition systems.

For NVIDIA Jetson Nano and Raspberry Pi 5 boards, the average predicted recognition time ranges from 3 ms for MobileNetV1 96x96 0.2 and MobileNetV2

96x96 0.35 models, to 14 ms - 33 ms for MobileNetV2 160x160 0.5 and MobileNetV2 160x160 1.0 networks.

After mixing processed and unprocessed training data, an improvement in recognition accuracy was observed for most categories, but a significant decrease in recognition accuracy was recorded for the "windows" category. After reverting the combined training data for the "windows" category to the unprocessed data, recognition accuracy for the MobileNetV1 96x96 0.2 network increased to 80% for NVIDIA Jetson Nano and Raspberry Pi 5 boards and to 73.3% for the ESP32-S3-EYE board. Recognition accuracy for the MobileNetV2 96x96 0.35 network reached 96.7% for all boards.

Additionally, the minimum and maximum times required to complete a full recognition-announcement cycle, taking into account the peculiarities of the Ukrainian language and the human reaction speed to auditory information, were determined. The minimum time is 379 ms for an object name consisting of one short word, while the maximum time is 2198 ms for an object name with three long words.

The practical significance of the results obtained in the dissertation can be used for designing and creating an adaptive real-time object recognition system with subsequent voice notifications for visually impaired or completely blind individuals. The results include calculations of the time required for a full recognition-announcement cycle, as well as the time needed for each of its separate elements, considering the peculiarities of the Ukrainian language and speech, and the human reaction speed to auditory information.

The required luminous flux was calculated, and the Cree XP-G3 LED was selected to ensure system operation in low-light conditions. Speakers and an amplifier were also selected. The total power consumption of various system configurations for ESP32-S3-EYE, Raspberry Pi 5, and NVIDIA Jetson Nano boards was calculated, and the required battery capacity was determined to ensure autonomous operation of the device for 3 hours.

Keywords: object recognition, image preprocessing, neural networks, microcontroller, single-board computer, MobileNet, Edge Impulse, training data, voice notification, language model, recognition accuracy, false recognition, autonomy.

Список публікацій здобувача

1. Денісов Р.В. Оникієнко Ю.О. Особливості розпізнавання зображень нейронними мережами на прикладі MobileNetV1 та MobileNetV2 в системах на мікроконтролерах. «Технології та інжиниринг» , Випуск 2(13), 2023 15-27 сторінки, DOI: 10.30857/2786-5371.2023.2.2, (фахове видання категорії Б)
2. Денісов Р.В., Попович П. В., Особливості попередньої обробки та групування тренувальних даних нейронної мережі для підвищення точності розпізнавання об'єктів на основі MobileNetV2. «Технології та інжиниринг» , Випуск 5(16), 2023 9-21 сторінки, DOI: 10.30857/2786-5371.2023.5.1, (фахове видання категорії Б)
3. Денісов Р.В., Попович П. В., Особливості застосування систем розпізнавання об'єктів у режимі реального часу на мікроконтролерах з подальшим голосовим виводом інформації для людей з вадами зору. «Технології та інжиниринг» , Випуск 3(20), 2024 21-30 сторінки, DOI: 10.30857/2786-5371.2024.3.2, (фахове видання категорії Б)

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ.....	14
ВСТУП.....	18
РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ РОЗПІЗНАВАННЯ ОБ’ЄКТІВ У ЕЛЕКТРОНИХ СИСТЕМАХ ТА СИСТЕМАХ ДЛЯ ЛЮДЕЙ З ВАДАМИ ЗОРУ	26
1.1 Загальна структура алгоритмів розпізнавання об’єктів.....	26
1.2 Традиційні методи розпізнавання об’єктів	45
1.3 Розпізнавання об’єктів на основі глибокого навчання.	51
1.4 Системи розпізнавання об’єктів для людей з вадами зору.....	60
Висновки до розділу 1	63
РОЗДІЛ 2. АНАЛІЗ ПРОГРАМНИХ ТА АПАРАТНИХ ЗАСОБІВ ДЛЯ РОЗПІЗНАВАННЯ ОБ’ЄКТІВ ТА ГЕНЕРУВАННЯ ГОЛОСУ	64
2.1 Дослідження нейронних мереж різних класів для розпізнавання об’єктів на пристроях з обмеженими можливостями.	64
2.2 Дослідження обґрунтування вибору мікроконтролера в системах розпізнавання об’єктів.	69
2.3 Вибір платформи для навчання і роботи з нейронними мережами.	77
2.4 Вибір програмного забезпечення для генерації тексту у мову.	80
Висновки до розділу 2	81
РОЗДІЛ 3. СИСТЕМА РОЗПІЗНАВАННЯ ОБ’ЄКТІВ З ГОЛОСОВИМИ ПІДКАЗКАМИ ДЛЯ ЛЮДЕЙ З ВАДАМИ ЗОРУ ТА ПОВНОЮ СЛІПОТОЮ	83
3.1 Структура та принцип роботи запропонованої мною системи.	83
3.2 Актуальність модифікації систем розпізнавання об’єктів для людей з вадами зору.	83
3.3 Оцінка часу необхідного на проходження одного повного циклу розпізнавання-оголошення інформації.	86
3.4 Попередня обробка і групування тренувальних даних для підвищення точності розпізнавання об’єктів у складних ситуаціях.....	91

3.5 Комплектації варіантів систем розпізнавання об'єктів з подальшим голосовим сповіщенням для людей з вадами зору.	92
Висновки до розділу 3	101
РОЗДІЛ 4. ДОСЛІДЖЕННЯ МОЖЛИВОСТІ ВИКОРИСТАННЯ НЕЙРОНИХ МЕРЕЖ У СИСТЕМАХ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ З ПОДАЛЬШИМ ГОЛОСОВИМ ВИВОДОМ ІНФОРМАЦІЇ ДЛЯ ЛЮДЕЙ З ВАДАМИ ЗОРУ.	102
4.1 Метод попередньої обробки і групування тренувальних даних для покращення розпізнавання об'єктів у складних ситуаціях.	102
4.2 Перевірка точності розпізнавання обраних категорій після змішування оброблених і необроблених тренувальних даних.	118
4.3 Представлення зовнішнього вигляду системи.	127
Висновки до розділу 4	127
ВИСНОВКИ	129
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	132

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

ВООЗ	– Всесвітня організація охорони здоров'я
МБ	– Мегабайт
ОЗП	– Оперативний запам'ятовуючий пристрій
ОС	– Операційна система
РК	– Рідкокристалічний
США	– Сполучені Штати Америки
УМБ	– Універсальна мобільна батарея
ІІІ	– Штучний інтелект
3D	– 3-Dimensional
AI	– Artificial intelligence - Штучний інтелект
ARM	– Advanced RISC Machine - Розширена RISC-архітектура
BiFPN	– Bi-directional Feature Pyramid Network - Двонаправлена мережа піраміди ознак
BLE	– Bluetooth low energy - Bluetooth з низьким енергоспоживанням
CDF	– Cumulative Distribution Function - Кумулятивна функція розподілу
CMOS	– Complementary Metal-Oxide-Semiconductor - Комплементарний метал-оксид-напівпровідник
CPU	– Central processing unit - Центральний процесор
CSPN	– Cross Stage Partial Network - Перехресна часткова мережа
CUDA	– Compute Unified Device Architecture - Обчислювальна уніфікована архітектура пристроїв
cuDNN	– CUDA Deep Neural Network library - Бібліотека глибокої нейронної мережі CUDA
dBFS	– Decibels relative to Full Scale - Децибели відносно повної шкали
Det	– Determinant - Детермінант
DoG	– Difference of Gaussian - Різниця Гауса

eMMC	- Embedded Multimedia Memory Card - Вбудована мультимедійна карта пам'яті
FOMO	- Faster Objects, More Objects - Швидші об'єкти, більше об'єктів
GPU	- Graphics processing unit - Графічний процесор
GPIO	- General-purpose input/output - Введення/виведення загального призначення
HAT	- Hardware Attached on Top – Обладнання прикріплене зверху
HDMI	- High Definition Multimedia Interface - Мультимедійний інтерфейс високої чіткості
HDR	- High Dynamic Range - Високий динамічний діапазон
HOG	- Histogram of Oriented Gradients - Гістограма орієнтованих градієнтів
IoT	- Internet of Things – Інтернет речей
I2C	- Inter-Integrated Circuit - Міжінтегральна схема
iOS	- iPhone Operating System - Операційна система iPhone
LED	- Light-emitting diode - Світлодіод
LPDDR	- Low Power Double Data Rate - Низька потужність подвійної швидкості передачі даних
LBP	- Local Binary Patterns - Локальні бінарні шаблони
ML	- Machine learning – Машинне навчання
MCU	- Microcontroller unit - Блок мікроконтролера
MIMO	- Multiple Input Multiple Output - Множинний вхід, Множинний вихід
MIN	- Minimum - Мінімум
MIPI	- Mobile Industry Processor Interface - Інтерфейс мобільного промислового процесора
MSE	- Mean Squared Error - Середня квадратична помилка
NCC	- Normalized Cross-Correlation - Нормована крос-кореляція
NG	- Next Generation - Наступне покоління
NoIR	- No Infrared - Без інфрачервоного випромінювання

NNAPI	- Neural Networks Application Programming Interface - Інтерфейс прикладного програмування нейронних мереж
PCIe	- Peripheral Component Interconnect Express - Експрес з'єднання периферійних компонентів
Pro	- Professional - Професійний
PWM	- Pulse-width modulation - Широтно-імпульсна модуляція
PSRAM	- Pseudo-Static Random Access Memory - Псевдостатична оперативна пам'ять
QSPI	- Quad Serial Peripheral Interface – Чотириканальний послідовний периферійний інтерфейс
RAM	- Random Access Memory - Оперативна пам'ять
R-CNN	- Region-based Convolutional Neural Networks - Регіональні згорткові нейронні мережі
ReLU	- Rectified Linear Unit - Випрямлена лінійна одиниця
ResNet	- Residual neural network - Залишкова нейронна мережа
RGB	- Red, Green, Blue – Червоний, зелений, синій
ROI Pooling	- Region of Interest Pooling – Об'єднання регіональних шарів
RPN	- Region Proposal Network - Мережа пропозицій регіонів
RST	- Reset - Скинути
SIFT	- Scale-Invariant Feature Transform - Масштабно-інваріантне перетворення ознак
SDK	- Software Development Kit - Набір для розробки програмного забезпечення
SD	- Secure Digital – Цифрова безпека
SDIO	- Secure Digital Input Output - Безпечний цифровий вхід і вихід
SDRAM	- Synchronous Dynamic Random Access Memory - Синхронна динамічна оперативна пам'ять
SAD	- Sum of Absolute Differences - Сума абсолютних різниць
SVM	- Support vector machines - Опорні векторні машини

SMD	- Surface mount technology - Технологія поверхневого монтажу
SNR	- Signal-to-noise ratio - Відношення сигнал/шум
SoC	- System-on-a-chip - Система-на-чипі
SoM	- System-on-Module – Модульна система
SSD	- Single Shot Multibox Detector - Одиночний детектор шарів
SSD	- Solid-state drive - Твердотільний накопичувач
SURF	- Speeded Up Robust Features - Прискорені надійні функції
TTS	- Text-to-speech – Генератор тексту у мову
TOPS	- Tera Operations Per Second - Операції Тера за секунду
TPU	- Tensor processing unit - Блок обробки тензорів
UART	- Universal Asynchronous Receiver-Transmitter - Універсальний асинхронний приймач-передавач
USB	- Universal Serial Bus - Універсальна послідовна шина
VGG	- Visual Geometry Group - Група візуальної геометрії
YOLO	- You Only Look Once - Дивишся лише раз

ВСТУП

Актуальність роботи. У сучасному світі технології відіграють вирішальну роль у підвищенні якості життя людей з обмеженими можливостями. Однією з найбільш актуальних проблем є створення адаптивних систем для людей із вадами зору, які дозволяють їм краще орієнтуватися у просторі та ефективніше взаємодіяти з навколишнім середовищем. За даними Всесвітньої організації охорони здоров'я, на сьогодні у світі понад 1 мільярд людей страждає від порушень зору, з яких понад 43 мільйони є незрячими. Розробка систем розпізнавання об'єктів із подальшим голосовим сповіщенням відкриває нові можливості для забезпечення автономності та безпеки таких людей. Поєднання мікроконтролерів та нейронних мереж дозволяє створювати компактні, енергоефективні та високопродуктивні пристрої для обробки зображень у реальному часі. Мікроконтролери та одноплатні комп'ютери, у поєднанні з невеликими моделями нейронних мереж, забезпечують високу швидкість обробки даних і мінімальну затримку між розпізнаванням об'єкта та голосовим сповіщенням.

Значний внесок у дослідження та розвиток у алгоритми і системи розпізнавання об'єктів зробили такі іноземні та вітчизняні вчені як Девід Марр, Томас Хаунг, Шимон Ульман, Такое Канаде, Ян Лекун, Джеффри Гінтон, Андрій Карпатов, Фей-Фей Лі, Джозеф Редмон, Росс Гіршік, Кріс Пінто, Алекс Кризевський, Сержіо Есколано, Джошуа Бенжіо. Проте аналіз праць, що стосується систем розпізнавання об'єктів для людей з вадами зору та повною сліпотою, свідчить про існування низки невирішених питань, стосовно можливості створення систем розпізнавання об'єктів для людей з вадами зору та повною сліпотою на основі нейронних мереж та мікроконтролерів, з подальшим голосовим озвученням назви розпізнаного об'єкту, а також стосовно часу необхідного на проходження одного повного циклу розпізнавання-оголошення з урахуванням особливостей української мови та мовлення.

У результаті проведеного аналізу встановлено, що системи розпізнавання об'єктів у режимі реального часу з подальшим голосовим сповіщенням можуть

бути створені на основі сучасних мікроконтролерів та нейронних мереж, і спроможні покращити орієнтування у просторі, підвищити рівень безпека та полегшити взаємодію з зовнішнім світом для людей з вадами зору та повною сліпотою.

Розраховано та підібрано компоненти системи, а саме джерело світла, динаміки, підсилювач для динаміків, джерело живлення та камеру, для комплектацій, що базуються на таких мікроконтролерах та одноплатних комп'ютерах як ESP32-S3-EYE , Raspberry Pi 5 та NVIDIA Jetson Nano, а також час необхідний на оголошення комбінацій слів різної довжини.

У результаті проведених досліджень та експериментів розраховано час необхідний на проходження одного повного циклу розпізнавання-оголошення результатів, з урахуванням особливостей Української мови та мовлення, різної довжини та комбінації слів, різного часу розпізнавання об'єктів такими нейронними мережами як MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, MobileNetV2 160x160 0.5 та нестиснута версія MobileNetV2 160x160 1.0, а також швидкості реакції людини на голосову інформацію.

У результаті використання власного методу попередньої обробки тренувальних даних методом видалення надлишкової інформації встановлено, що для об'єктів площа яких значно більша за площу надлишкової інформації, або об'єктів з різних категорій, що можуть знаходитися на тренувальних даних один біля одного, запропонований метод підвищує точність розпізнавання для категорій об'єктів "door", "trees", "traffic lights", "crosswalk", підвищує стійкість системи, зменшуючи кількість хибних розпізнавань. Для об'єктів з категорії "windows", площа яких значно менша, такий метод може призвести зниження точності розпізнавання об'єктів категорії "windows".

Отримані результати досліджень можуть бути використані для створення системи розпізнавання об'єктів у режимі реального часу з голосовим сповіщенням для людей з вадами зору та повною сліпотою.

Мета і завдання дослідження. Метою дисертаційної роботи є удосконалення систем розпізнавання об'єктів для людей з вадами зору та повною

сліпотою шляхом застосування нейронних мереж на мікроконтролерах, та введення голосового сповіщення, а також розрахунок технічних параметрів окремих елементів системи та підбір компонентів на їх основі.

Об'єкт дослідження – є процес розпізнавання об'єктів та час необхідний на голосове сповіщення з урахуванням особливостей Української мови та мовлення, а також швидкості реакції людини на слухову інформацію на основі мікроконтролерів і нейронних мереж.

Предмет дослідження – нейроні мережі, мікроконтролери та одноплатні комп'ютери, платформи для навчання нейронних мереж та програмне забезпечення для генерації мови.

Для досягнення поставленої мети необхідно було вирішити такі завдання:

1. Провести аналіз сучасних методів та алгоритмів розпізнавання об'єктів, які можна було б застосувати в системах для людей з вадами зору.
2. Провести дослідження архітектур нейронних мереж та можливості їх застосування в системах розпізнавання об'єктів. Обґрунтувати вибір мікроконтролерів та нейронних мереж MobileNet для розпізнавання об'єктів, провести навчання та тестування обраних мереж.
3. Розробити комбіновану адаптивну систему розпізнавання об'єктів з подальшим голосовим сповіщенням для людей з вадами зору та повною сліпотою.
4. Дослідити платформи для тренування та перевірки нейронних мереж, та обґрунтувати вибір платформи.
5. Дослідити можливості застосування голосових генеративних моделей в запропонованій системі. Обґрунтувати вибір голосової генеративної моделі.
6. Провести розрахунок часу необхідно для проходження одного повного циклу розпізнавання об'єкту, генерування повідомлення та його оголошення. Дослідити вплив запропонованого методу попередньої обробки зображень на точність розпізнавання обраних категорій об'єктів.

7. Виконати підбір компонентів для запропонованої системи та розрахунок загальної споживаної потужності і на його основі розрахувати ємність джерела живлення для забезпечення автономної роботи системи на 2-3 години.

Методи дослідження. Для вирішення поставлених завдань під час теоретичних досліджень було використано порівняльний аналіз існуючих методів розпізнавання об'єктів в електронних системах та системах для людей з вадами зору, порівняльний аналіз нейронних мереж доступних для розгортання на приладах з обмеженими обчислювальними можливостями, виконано порівняння платформ для навчання нейронних мереж, використано доступні для роботи з нейронними мережами мікроконтролери та одноплатні комп'ютери, а також програмне забезпечення для генерації тексту в мову. Під час проведення експериментальних досліджень використано метод попередньої обробки тренувальних зображень, шляхом видалення надлишкової інформації в тренувальних даних за допомогою прикладних програмних пакетів. Застосовано метод змішування оброблених та необроблених тренувальних даних для підвищення точності розпізнавання зображень, метод тренування і перевірки нейронних мереж на змішаних та незмішаних тренувальних даних у програмному середовищі Edge Impulse. Виконано розрахунок часу необхідного на проходження одного повного циклу розпізнавання-оголошення, шляхом розрахунку мінімального, середнього та максимального часу необхідного на проходження кожного окремого етапу.

Наукова новизна одержаних результатів.

В дисертації представлено такі наукові результати:

1. Вперше запропоновано комбіновану адаптивну систему для людей з вадами зору та повною сліпотою, яка відрізняється від існуючих рішень поєднанням розпізнавання об'єктів за допомогою нейронних мереж MobileNet на мікроконтролері та формуванням повідомлення про розпізнаний об'єкт за допомогою генеративної моделі мовлення, з урахуванням особливостей української мови та мовлення, а також швидкості реакції людини на голосову інформацію.

2. Вперше запропоновано метод попередньої обробки тренувальних даних, який полягає у видаленні надлишкової інформації з зображень та змішуванні необроблених та оброблених тренувальних даних для підвищення точності розпізнавання категорій об'єктів "door", "trees", "traffic lights", "crosswalk", "windows".

3. Вперше визначено умову застосування методу попередньої обробки для різних категорій об'єктів розпізнавання, яка полягає у порівнянні площі надлишкової інформації з корисною інформацією в межах кадру.

Особистий внесок здобувача. Результати досліджень, що наведені у дисертаційній роботі та винесені на захист, отримані особисто автором або ж за його активної участі та опубліковані у спеціалізованих фахових виданнях України.

В рамках роботи [1], що опублікована в співавторстві у фаховому виданні України, здобувачем особисто виконано аналіз впливу гіперпараметрів обробки зображення та архітектури нейронної мережі MobileNet на об'єм використаних ресурсів мікроконтролера, виконано порівняння результатів точності та швидкості розпізнавання моделей нейронних мереж MobileNetV1 та MobileNetV2.

В рамках роботи [2], що опублікована в співавторстві у фаховому виданні України, здобувачем особисто виконано дослідження можливостей підвищення точності та варіативності розпізнавання різних груп об'єктів зі схожою надлишковою інформацією нейронною мережею після попередньої обробки та групування тренувальних зображень для подальшого використання на мікроконтролерах. Перевірено вплив видалення зайвої інформації у тренувальних даних на практичні значення точності розпізнавання різних категорій об'єктів на основі архітектури MobileNet V2.

В рамках роботи [3], опублікованій у співавторстві, здобувачем особисто виконано дослідження мінімального і максимального часу необхідного для виконання одного повного циклу розпізнавання-оголошення назви об'єкту з урахуванням різної довжини слів, різної швидкості розпізнавання об'єктів, а

також фізичних особливостей людей з вадами зору для систем розпізнавання об'єктів у режимі реального часу на мікроконтролерах з подальшим голосовим виведенням.

Практичне значення одержаних результатів:

Отримані результати можуть бути використані для створення систем розпізнавання об'єктів з подальшим голосовим озвученням інформації на базі нейронних мереж та мікроконтролерів для людей з вадами зору.

1. Проведене дослідження існуючих моделей нейронних мереж та доступних мікроконтролерів і одноплатних комп'ютерів дає змогу визначити перелік необхідних програмно-апаратних засобів для практичної реалізації системи, які можуть забезпечити високу точність та швидкість розпізнавання.

2. Проведені розрахунки параметрів складових системи дають можливість здійснити вибір елементів, що відповідають програмно-апаратним вимогам, а також сумісні з платами ESP32-S3-EYE, Raspberry Pi 5 та NVIDIA Jetson Nano. До складу системи на основі розрахунків доцільно включити динаміки, підсилювач для динаміків, джерело світла та джерело живлення.

3. Розраховано мінімальний та максимальний час проходження одного повного циклу розпізнавання-оголошення. Мінімальний час становить 379 мс для короткої назви об'єкту, що складається з одного слова, з урахуванням часу реакції людини на слухову інформацію, та максимальний час 2198 мс для назви об'єкту, що складається з трьох довгих слів, з урахуванням пауз між словами.

4. Встановлено, що для категорій об'єктів "door", "trees", "traffic lights", "crosswalk" , площа яких більша за площу надлишкової інформації в кадрі застосування методу попередньої обробки дає приріст у точності розпізнавання. Для категорії "windows", де площа самого об'єкту значно менша за надлишкову інформацію в межах кадру, такий метод може призвести до погіршення точності розпізнавання для цієї категорії об'єктів.

5. Експериментально встановлено, що після тренування нейронних мереж MobileNetV1 96x96 0.2 та MobileNetV2 96x96 0.35 на змішаних тренувальних даних у категоріях "door", "trees", "traffic lights", "crosswalk" та незмішаних

тренувальних даних у категорії "windows" отримано приріст у точності розпізнавання об'єктів. Для категорії "crosswalk" з 80% до 100%, для категорії "doors" з 66.7 % до 83.3%, для категорії "traffic lights" з 42.9% до 71.4%, та з 71.4% до 85.7 % для категорії "trees" для мережі MobileNetV1 96x96 0.2. Для мережі MobileNetV2 96x96 0.35 у категорії "traffic lights" отримано підвищення точності з 71.4% до 100%.

Зв'язок роботи з науковими програмами, планами, темами.

Роботу виконано на кафедрі акустичних та мультимедійних електронних систем Національного технічного університету України "Київський політехнічний інститут імені Ігоря Сікорського" .

Наукові дослідження виконано здобувачем в рамках держбюджетної НДР № 2704-п "Генезис мінно-вибухових травм і розробка мобільного електроакустичного апарату для діагностики і лікування ушкоджень слуху військовослужбовців" (№ДР 0124U00087) під керівництвом доцента кафедри акустичних та мультимедійних електронних систем, к.т.н. Поповича Павла Васильовича.

Викладені у дисертації нові теоретичні та практичні результати досліджень можуть використовуватися під час розробки та проєктування електронних пристроїв та систем на мікроконтролерах, а також у освітньому процесі кафедри акустичних та мультимедійних електронних систем за спеціальністю 171 Електроніка, в освітній програмі "Електронні системи мультимедіа та засоби Інтернету речей" Національного технічного університету України "Київський політехнічний інститут імені Ігоря Сікорського".

Апробація результатів дисертації. Матеріали дисертаційних досліджень обговорювалися на міжнародних конференціях:

VII Міжнародна науково-практична конференція «SCIENCE AND TECHNOLOGY: CHALLENGES, PROSPECTS AND INNOVATIONS», 26-28.02.2025, Осака, Японія.

Публікації. За результатами досліджень опубліковано 3 наукові публікації за спеціальністю 171 Електроніка, галузь знань Електроніка та телекомунікації, з них 3 статті в наукових фахових виданнях України категорії Б.

Структура та обсяг дисертації. Робота складається зі вступу, чотирьох розділів, висновків, переліку джерел посилань. Робота містить 81 рисунок та 7 таблиць. Загальний обсяг дисертаційної роботи становить 139 сторінок.

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ У ЕЛЕКТРОНИХ СИСТЕМАХ ТА СИСТЕМАХ ДЛЯ ЛЮДЕЙ З ВАДАМИ ЗОРУ

1.1 Загальна структура алгоритмів розпізнавання об'єктів

Розпізнавання об'єктів – ключовий напрямок комп'ютерного зору, направлений на автоматизацію процесів отримання, аналізу та розуміння візуальної інформації з цифрових зображень або відео у режимі реального часу. Основною метою є створення систем, здатних ідентифікувати й інтерпретувати об'єкти в зображеннях по аналогії з людським зором. Розпізнавання об'єктів стало однією з ключових технологій у багатьох сферах, таких як робототехніка, автономне управління та керування, медицина, промислове виробництво - де автономні системи взаємодіють із навколишнім середовищем. Системи автопілотування автомобілів та безпілотних літальних апаратів, виявлення злоякісних пухлин на рентгенівських знімках, інклюзивні технології для людей з вадами зору, що дозволяють їм отримувати інформацію про навколишні об'єкти за допомогою голосових підказок, можуть бути прикладом практичного застосування.

Технології комп'ютерного зору почали розвиватися в середині 20 століття. Спочатку це були прості алгоритми для розпізнавання країв об'єктів на зображеннях, однак з розвитком технологій з'явилися більш потужні методи на основі глибокого навчання, здатні розпізнавати об'єкти у складних умовах.

Типова структура алгоритму розпізнавання об'єктів складається з таких етапів:

- Попередня обробка зображень (Preprocessing)
- Виділення ознак (Feature Extraction)
- Сегментація об'єктів (Object Segmentation)
- Пост-обробка (Post-processing)

Попередня обробка зображень (Preprocessing) – застосовується для стандартизації, уніфікації та групування зображень, що дозволяє покращити якість даних перед їх подальшою обробкою та аналізом. Метою цього етапу є

підготовка зображень для оптимальної роботи алгоритмів глибокого навчання або інших методів аналізу.

До основних методів попередньої обробки належать:

- Нормалізація та масштабування зображень
- Фільтрація та видалення шуму
- Робота з контрастом зображення
- Конверсія кольору у відтінки сірого

Нормалізація та масштабування даних — це дві ключові техніки попередньої обробки зображень, що використовуються для підготовки даних перед застосуванням алгоритмів машинного або глибокого навчання. Нормалізація полягає у приведенні значень пікселя зображення до одного діапазону, наприклад $[0,1]$, що допомагає усунути відмінності в яскравості та контрастності різних зображень, які можуть бути спричинені різними варіаціями умов освітлення, налаштуваннями камери та іншими факторами. Нормалізація дозволяє уникнути помилок при змішуванні різних груп зображень та покращує роботу алгоритму. До основних методів нормалізації відносяться Min-Max Scaling та Z-score standardization. Також, для виконання нормалізації даних, можна використовувати бібліотеки такі як PyTorch, або бібліотеки Scikit-learn на Python.

Min-Max Scaling — метод використовується для трансформації значень пікселів зображень у певний діапазон, зазвичай від 0 до 1, що дозволяє рівномірно масштабувати пікселі, незалежно від їх початкових значень. Загальний вигляд формули нормалізації за методом Min-Max Scaling має вигляд [4]:

$$X_{\text{normalized}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}},$$

де X — випадкове значення ознаки, яке має бути нормалізовано. X_{\min} — мінімальне значення ознаки в наборі даних, а X_{\max} — максимальне значення ознаки.

Метод особливо ефективний, коли важливо зберегти розподіл даних у вихідній формі, але при цьому привести всі значення до єдиного діапазону, що спрощує обробку даних для алгоритмів машинного навчання.

Z-score standardization (Стандартизація) — передбачає трансформацію значень так, щоб середнє значення зображення стало рівним 0, а стандартне відхилення дорівнювало 1. Це особливо корисно для моделей, що передбачають нормальний розподіл даних, таких як лінійні моделі та опорні вектори. Загальний вигляд формули нормалізації за методом Z-score standardization має вигляд [4]:

$$Z_{\text{standertized}} = \frac{X - \mu}{\sigma},$$

де μ - середнє значення, а σ - стандартне відхилення.

Метод особливо корисний під час роботи з алгоритмами, які передбачають нормально розподілені дані, наприклад багато лінійних моделей. На відміну від техніки мінімально-максимального масштабування, значення ознак не обмежуються певним діапазоном у техніці стандартизації. Даний метод нормалізації в основному представляє ознаки з точки зору кількості стандартних відхилень, які лежать далеко від середнього.

Масштабування змінює розміри зображення або його об'єктів, щоб привести їх до певного стандартного розміру. Це важливо для моделей, які приймають на вхід зображення фіксованого розміру, таких як CNN. Масштабування дозволяє зменшувати або збільшувати зображення відповідно до вимог моделі, зберігаючи при цьому співвідношення сторін. До соновних методів маштабування відносяться Resize (зміна розміру), та Cropping (обрізання зображення).

Resize (зміна розміру) — це процес зміни розмірів зображення (ширини і висоти), для приведення зображення до стандартного розміру, необхідного для навчання моделей машинного навчання [4]. Зміна розміру кожного пікселя здійснюється за допомогою інтерполяції. Найбільш поширені методи — це білінійна і бікубічна інтерполяція.

Білінійна інтерполяція — це метод зміни розміру зображень, який використовує лінійну інтерполяцію у двох напрямках (по ширині і висоті) для визначення значення нового пікселя на основі його найближчих сусідніх пікселів. Білінійна інтерполяція розраховує новий піксель як зважену суму чотирьох сусідніх пікселів в оригінальному зображенні. Формула білінійної інтерполяції має вигляд [5]:

$$I'(x',y') = (1 - dx) \times (1 - dy) \times I(x_1,y_1) + dx \times (1 - dy) \times I(x_2,y_1) + (1 - dx) \times dy \times I(x_1,y_2) + dx \times dy \times I(x_2,y_2),$$

де: $I(x_1,y_1)$, $I(x_2,y_1)$, $I(x_1,y_2)$, $I(x_2,y_2)$ — це значення сусідніх пікселів, x' та y' — нові координати пікселя в масштабованому зображенні, dx і dy — дробові частини нових координат, $I'(x',y')$ — значення нового пікселя у масштабованому зображенні після інтерполяції.

Бікубічна інтерполяція використовується для зміни розміру зображень та полягає у визначенні значення нового пікселя на основі 16 сусідніх пікселів оригінального зображення [6]. Вона враховує не лише значення пікселів, але й їхні похідні, що дозволяє створити більш плавний результат у порівнянні з білінійною інтерполяцією. Формула бікубічної інтерполяції:

$$I'(x',y') = \sum_{i=0}^3 \sum_{j=0}^3 \omega(x' - x_i) \omega(y' - y_j) I(x_i, y_j),$$

де $I(x_i,y_j)$ — це значення пікселя в оригінальному зображенні, x_i і y_j — це сусідні пікселі, що оточують новий піксель, w — це вагова функція, яка базується на кубічній функції.

Вагова функція $w(d)$ [6]:

$$\omega(d) = \begin{cases} (1.5|d|^3 - 2.5|d|^2 + 1), & \text{якщо } |d| \leq 1 \\ (-0.5|d|^3 + 2.5|d|^2 - 4|d| + 2), & \text{якщо } 1 < |d| \leq 2 \\ 0, & \text{якщо } |d| > 2 \end{cases}$$

Вагова функція $w(d)$ визначає вплив кожного сусіднього пікселя на новий піксель, враховуючи його відстань від нового пікселя.

Cropping — техніка обрізання частини зображення для виділення основного об'єкта або видалення небажаних частин зображення [7]. Використовується для фокусування на ключових елементах зображення або для

приведення зображення до стандартного розміру, зберігаючи при цьому співвідношення сторін. Формула може бути представлена у вигляді:

$$I'(x',y') = I(x_0 + x', y_0 + y'),$$

де, $I(x,y)$ — це значення пікселя в оригінальному зображенні, x_0, y_0 — координати початкової точки обрізання, x', y' — нові координати після обрізання, $I'(x',y')$ — значення пікселя у новому зображенні після обрізання.

Фільтрація та видалення шуму - це процес усунення небажаного шуму із зображення, шляхом застосування фільтра, який регулює значення пікселів зображення залежно від типу фільтра. Вони можуть бути розроблені для видалення певних типів шуму, наприклад шуму Гауса. У загальному випадку виділяють три основні методи фільтрації та видалення шуму - Box Filter, Гауссовий фільтр, медіанний фільтр.

Box Filter — це один із найпростіших фільтрів для згладжування зображень, що використовується для видалення шуму [7]. Box Filter працює шляхом заміни кожного пікселя на середнє значення його сусідів у вікні розміру $K_{height} \times K_{width}$. Усі пікселі всередині цього вікна мають однакову вагу. Тобто, для кожного пікселя обчислюється середнє значення інтенсивностей пікселів, що знаходяться поруч. Значення пікселя після застосування фільтра можна отримати за формулою:

$$I'(x,y) = \frac{1}{K_{height} \times K_{width}} \sum_{i=-\frac{K_{height}}{2}}^{\frac{K_{height}}{2}} \sum_{j=-\frac{K_{width}}{2}}^{\frac{K_{width}}{2}} I(x + i, y + j),$$

де $I(x,y)$ — інтенсивність пікселя в оригінальному зображенні, $K_{height} \times K_{width}$ — розмір ядра (зазвичай квадрат, наприклад, 3×3 або 5×5), $I'(x,y)$ — інтенсивність пікселя після згладжування.

Box Filter часто застосовується як попередня обробка для усунення випадкового шуму перед виконанням більш складних операцій, таких як сегментація або розпізнавання об'єктів. До переваг можна віднести : простоту у реалізації, та ефективність при усуненні дрібного шуму, приклад роботи бокс фільтру представлено на рисунку 1.1.

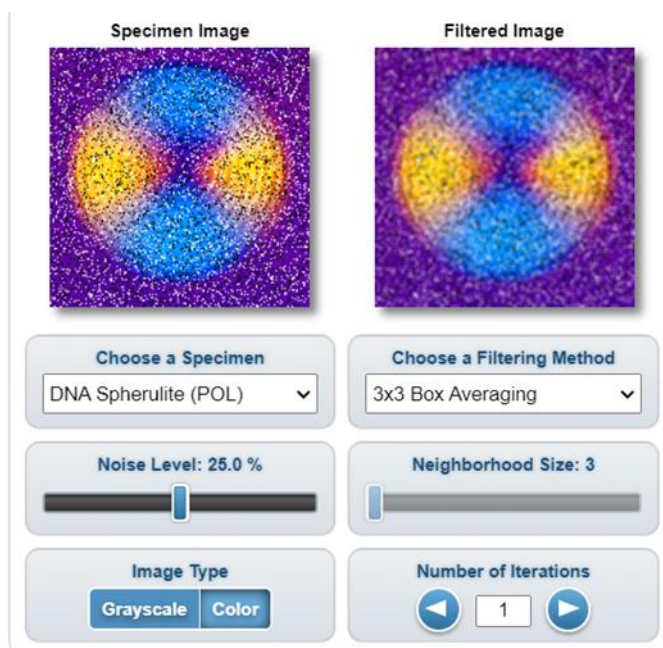


Рисунок 1.1 - Приклад роботи бокс фільтру [8]

До недоліків - зниження різкості зображення, через рівномірне згладжування всіх пікселі, включно з краями, що може призвести до їхнього розмивання.

Гаусовий фільтр — це лінійний фільтр, що використовується в обробці зображень для зменшення шуму та розмиття зображень [7]. Його основна мета — згладити зображення, зберігаючи при цьому краї та деталі. До характеристик Гаусівського фільтра відносять : Гаусову функцію, конволюцію, та параметр σ .

Гаусовий фільтр заснований на Гаусовій функції, яка має форму дзвоноподібної кривої. У двох вимірах її можна описати формулою:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}},$$

де (x,y) — координати пікселя, а σ — стандартне відхилення, що визначає ступінь розмиття.

Фільтрація зображення відбувається шляхом конволюції Гауссового ядра з пікселями зображення. Кожен піксель нового зображення обчислюється як зважена сума пікселів навколо нього, де ваги визначаються Гаусовою функцією.

Значення σ контролює ступінь розмиття: невелике σ призводить до меншого розмиття, зберігаючи більше деталей. Велике σ призводить до сильнішого розмиття, з більшою втратою деталей рисунку 1.2.

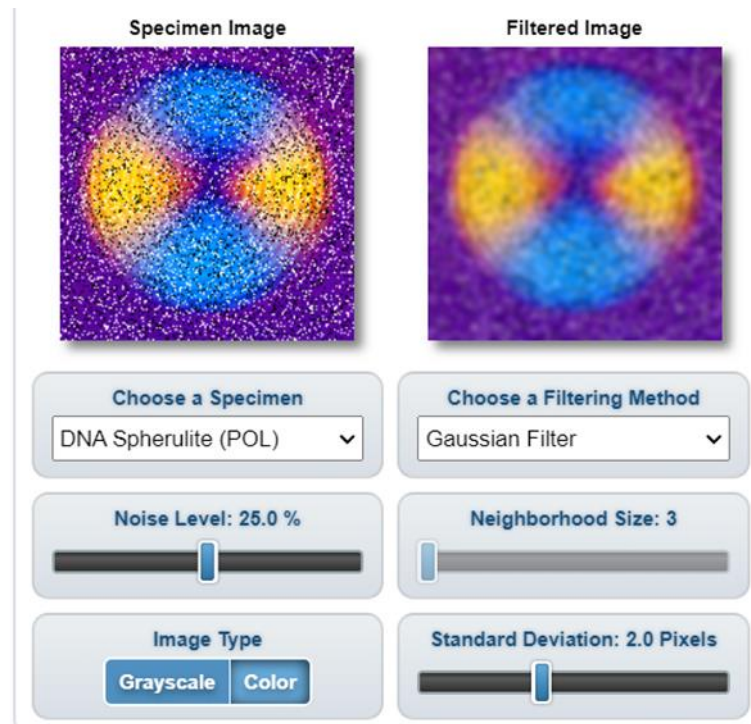


Рисунок 1.2 - Приклад роботи фільтру Гауса [8]

До переваг можна віднести: ефективне зменшення високочастотного шуму, плавне розмивання зображення, що не викликає артефактів, які можуть виникати при використанні інших фільтрів.

Медіанний фільтр — це нелінійний фільтр, який широко використовується в обробці зображень для зменшення шуму, особливо імпульсного (шум «сіль-перець»), при збереженні країв і деталей зображення [7].

$$I'(x,y) = \text{Median} (I(x+i,y+j))$$

Медіанний фільтр працює на основі статистичної функції медіани. Для кожного пікселя в зображенні розглядаються сусідні пікселі, формується з них набір, а потім замінюється значення центрального пікселя на медіану цього набору. Для фільтрації зображення зазвичай використовується квадратне або кругле вікно (ядро) певного розміру (наприклад, 3x3, 5x5).

У процесі фільтрації вибирається вікно навколо кожного пікселя, яке визначається його сусідами. Усі значення пікселів у вікні сортуються за

зростанням. Після чого значення медіани замінює значення центрального пікселя. Приклад роботи медіанного фільтру представлено на рисунку 1.3.

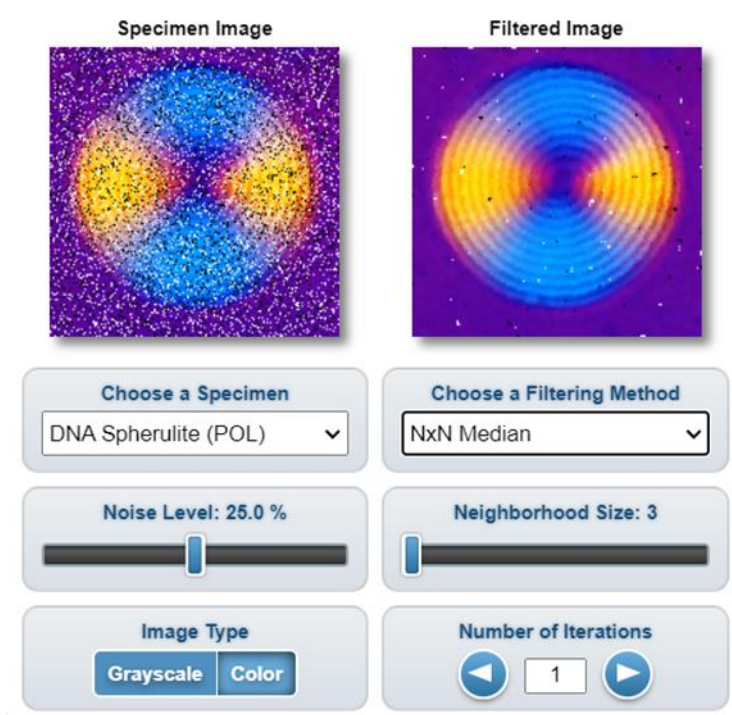


Рисунок 1.3 – Приклад роботи медіанного фільтру [8]

До переваг можна віднести захист країв та високу стійкість до шуму, особливо до імпульсного. До недоліків - можливе розмиття деталей при великому вікні фільтру, та вищу обчислювальну складність, у порівнянні з лінійними фільтрами.

Корекція контрасту зображення призначена для покращення візуального сприйняття зображення, та підвищення його якості і видимості деталей. До основних методів корекції контрасту відносять : лінійну корекцію, нелінійну корекцію, метод гістограм(вирівнювання гістограми) та метод розширення контрасту.

Лінійна корекція контрасту — це проста та ефективна техніка покращення контрасту зображень під час попередньої обробки, що базується на застосуванні лінійного перетворення до кожного пікселя зображення і може бути описана формулою:

$$g(i,j) = \alpha \cdot f(i,j) + \beta,$$

де $g(i,j)$ — нове значення пікселя; $f(i,j)$ — вихідне значення пікселя; α — коефіцієнт контрасту (значення більше за 1 збільшують контраст, менші за 1 — зменшують); β — зміщення яскравості (зміна середнього значення пікселів).



Рисунок 1.4 – Приклад лінійної корекції контрасту [9]

Нелінійна корекція - використовує функції, такі як гамма-корекція, для зміни яскравості зображення. Гамма-корекція дозволяє налаштовувати яскравість у певних діапазонах, що може бути корисним для підвищення видимості деталей в темних або світлих областях. Формула для гамма-корекції має вигляд [10]:

$$O = \left(\frac{I}{255}\right)^\gamma \times 255,$$

де I — початкове значення яскравості пікселя (від 0 до 255); O — нове значення яскравості пікселя після корекції; γ — гамма-коефіцієнт, який визначає рівень корекції (значення більше 1 затемнюють зображення, менше 1 — освітлюють); 255 — максимальне значення яскравості в 8-бітному зображенні для нормалізації.

Цей метод дозволяє керувати підвищенням або зниженням контрасту, зокрема для дуже темних або яскравих зображень, коригуючи відображення тіней і яскравості окремо.

До переваг гамма-корекції можна віднести – те, що нелінійний характер корекції дозволяє точніше налаштовувати контраст окремих діапазонів яскравості, зокрема тіней та світлих областей. Покращення деталей в затемнених або переосвітлених частинах зображення. До недоліків можна віднести

потенційне спотворення кольорів через некоректні значення гамма-кофіцієнту, та можливу втрату інформації у високо контрастних областях.

Вирівнювання гистограми — це метод глобального покращення контрастності, який перерозподіляє інтенсивність пікселів для досягнення майже рівномірної гистограми, що передбачає обчислення кумулятивної функції розподілу (CDF), гистограми зображення та відображення початкових значень пікселів на нові значення на основі CDF [11]. Вирівнювання гистограми ефективно розподіляє найбільш часті значення інтенсивності, підвищуючи загальний контраст зображення рис.1.5. Техніка особливо корисна для зображень із поганою контрастністю або коли важлива інформація зосереджена у вузькому діапазоні значень інтенсивності.

Кумулятивна функція розподілу (CDF, Cumulative Distribution Function) для гістограмної корекції контрасту обчислюється за наступною формулою:

$$\text{CDF}(i) = \sum_{j=0}^i p(j),$$

де $\text{CDF}(i)$ — кумулятивна функція розподілу для рівня інтенсивності i ; $p(j)$ — ймовірність появи рівня яскравості j , яка визначається як кількість пікселів з цією інтенсивністю, поділена на загальну кількість пікселів у зображенні.

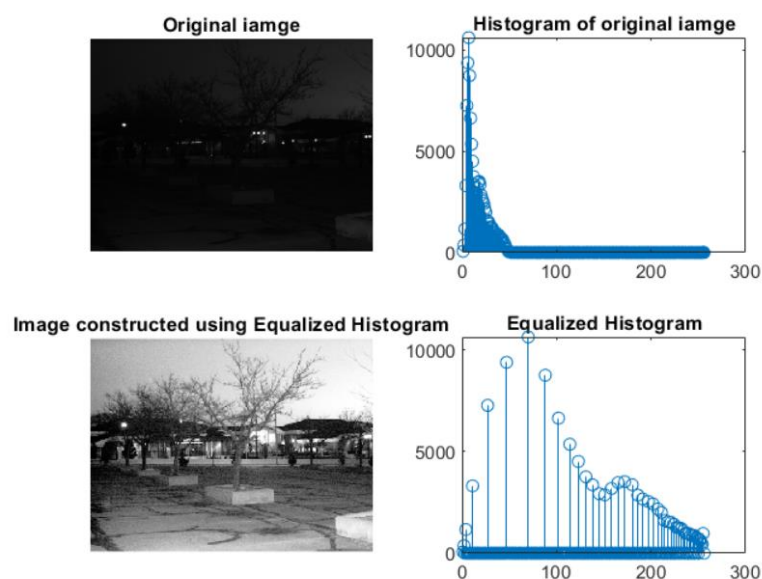


Рисунок 1.5 – Приклад вирівнювання гистограми [12]

Метод розширення контрасту (або *stretching contrast*) — це техніка покращення контрасту шляхом масштабування рівнів яскравості зображення для

охоплення всього діапазону доступних значень інтенсивності (наприклад, від 0 до 255 для 8-бітних зображень) рис.1.6. Спершу відбувається визначення мінімуму та максимуму інтенсивності (I_{\min} та I_{\max}) в оригінальному зображенні. Після чого кожне значення пікселя I_{old} перетворюється на нове значення I_{new} за допомогою лінійної формули [13]:

$$I_{\text{new}} = \frac{I_{\text{old}} - I_{\min}}{I_{\max} - I_{\min}} \times (I_{\max_new} - I_{\min_new}) + I_{\min_new},$$

де I_{old} — вихідне значення пікселя; I_{\min} і I_{\max} — відповідно мінімальні та максимальні інтенсивності у вихідному зображенні; I_{\min_new} і I_{\max_new} — мінімальне та максимальне значення нового діапазону (наприклад, 0 і 255 для 8-бітного зображення).

В результаті всі значення пікселів після перетворення будуть розподілені по новому діапазону, що покращить видимість деталей у зображенні.

Розширення контрасту дозволяє краще візуалізувати деталі зображення, які могли бути непомітними через низький контраст. Метод також є простим у реалізації та може бути ефективним для широкого кола зображень.

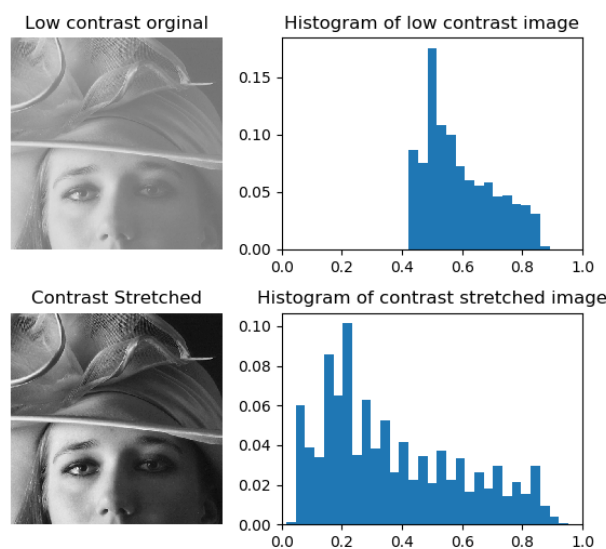


Рисунок 1.6 – Приклад методу розширення контрасту [13]

До недоліків можна віднести потенційне підвищення шуму в зображенні, особливо коли початковий контраст був низьким та втрату деталей у крайніх областях.

Конверсія кольору у відтінки сірого — це ключовий етап попередньої обробки зображень, який спрощує аналіз та зменшує обсяг даних для обробки. Відтінки сірого зображення не містять колірної інформації, а лише рівні яскравості пікселів, що робить зображення одночасно простішим для аналізу та менш ресурсоємним. Існує декілька основних підходів до конверсії кольору у відтінки сірого.

Середнє значення каналів RGB, при якому кожен піксель відтінку сірого обчислюється як середнє значення компонент кольору (червоного, зеленого і синього) і визначається за формулою:

$$I_{\text{gray}} = \frac{R+G+B}{3},$$

Але такий підхід не враховує, що людське сприйняття кольору є нерівномірним.

Зважене середнє каналів RGB. Оскільки око людини більш чутливе до зеленого кольору, менше — до червоного, і ще менше — до синього, використовують зважене середнє для кращої відповідності сприйняттю:

$$I_{\text{gray}} = 0,299 \times R + 0,587 \times G + 0,114 \times B,$$

Ця формула відображає людське сприйняття світла та забезпечує більш точне відтворення яскравості, оскільки зелений колір вносить найбільший вклад, а синій — найменший.

Метод максимальної інтенсивності. В цьому випадку сірий піксель обчислюється за максимальною інтенсивністю компонент кольорів:

$$I_{\text{gray}} = \max(R, G, B),$$

часто використовується в комп'ютерному зорі для фільтрації важливих об'єктів на зображенні.

Метод середнього по яскравості використовується для обчислення сірого значення на основі яскравості пікселів, яка може бути більш релевантною для специфічних застосувань, таких як супутникові зображення або медичні знімки.

До переваг конверсії в сірі відтінки можна віднести: зменшення розмірів даних, зниження обчислювальних витрат та полегшення аналізу зображень рис.1.7.

До недоліків: втрату інформації, адже перехід у відтінки сірого призводить до втрати кольорової інформації, яка може бути важливою для деяких застосувань (наприклад, для обробки медичних зображень, де колір може вказувати на патологічні зміни).

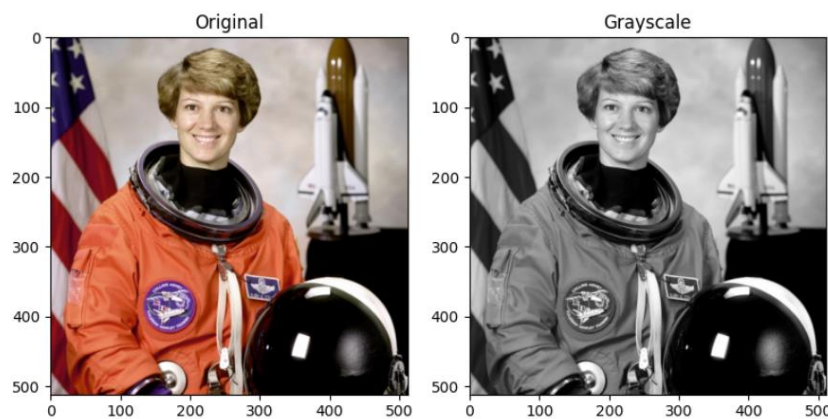


Рисунок 1.7 – Приклад конверсії зображення у відтінки сірого [14]

Виділення ознак — це критично важливий етап у комп'ютерному зорі та розпізнаванні образів, який полягає в перетворенні зображення в набір значень (ознак), які характеризують об'єкт. Цей процес дозволяє спростити ідентифікацію та класифікацію об'єктів, зберігаючи при цьому найважливішу інформацію про них. Ознаки можуть включати: контури, текстури та колір. До основних методів виділення контурів відносять – метод детектора Канні, метод Собеля, метод Лапласа та детектор градієнтів.

Виділення текстур — задача, яка включає аналіз та опис структурних характеристик зображень. Текстура визначає, як світло відбивається від поверхні об'єкта, що може бути корисним для класифікації об'єктів, сегментації зображень та інших застосувань.

До методів виділення текстур відносять - метод статистичних ознак, який передбачає аналіз статистичних характеристик зображення. Він може включати обчислення таких параметрів, як:

- Гістограми - вивчення розподілу яскравості пікселів у зображенні. Гістограми можуть допомогти виявити різноманітні текстури.
- Матриця співвідношення. Створюється для оцінки просторових відносин між пікселями. Матриця обчислюється шляхом аналізу яскравості сусідніх пікселів.

Фільтри Габора є одним з найпопулярніших методів для виділення текстур. Вони поєднують в собі інформацію про частоту і просторову локалізацію, що робить їх ефективними для виявлення текстур у зображеннях. Основна ідея полягає в застосуванні фільтрів до зображення на різних частотах та орієнтаціях рис. 1.8.

Габоровий фільтр визначається як добуток Гауссової функції і комплексної експоненти [15]. Основна формула для Габорового фільтру виглядає наступним чином:

$$g(x,y;\lambda,\theta, \varphi,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2+y^2}{2\sigma^2}} \times e^{j(\frac{2\pi}{\lambda} \times x \times \cos\theta + \frac{2\pi}{\lambda} \times y \times \sin\theta + \varphi)},$$

де - (x,y) — координати пікселя; λ — довжина хвилі хвилі; θ — орієнтація фільтру; φ — фаза (додаткова зміщення); σ — стандартне відхилення Гауссової функції, яке визначає ширину фільтру.

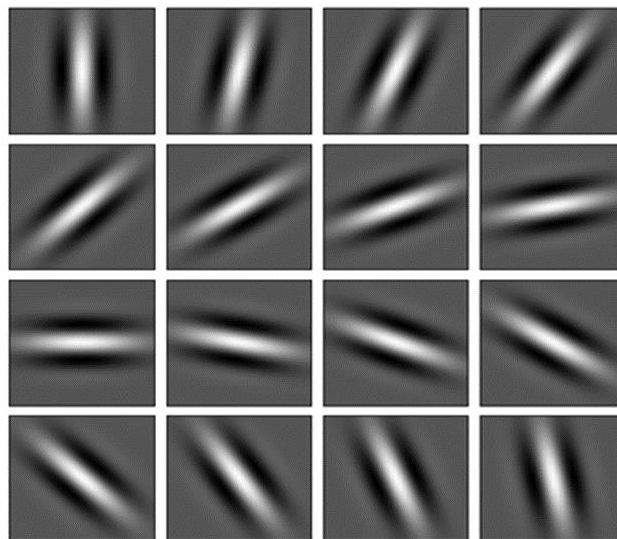


Рисунок 1.8 – Приклад фільтрів Габора [16]

Локальні бінарні патерни (Local Binary Patterns - LBP) — це простий і ефективний метод для опису текстур, що базується на порівнянні значень

пікселів з їх сусідами [17]. Алгоритм працює наступним чином: для кожного пікселя зображення визначається, чи є сусідні пікселі світлішими або темнішими за цей піксель рис.1.9. Результат порівняння представляється у вигляді бінарного числа, яке далі перетворюється на десяткове.

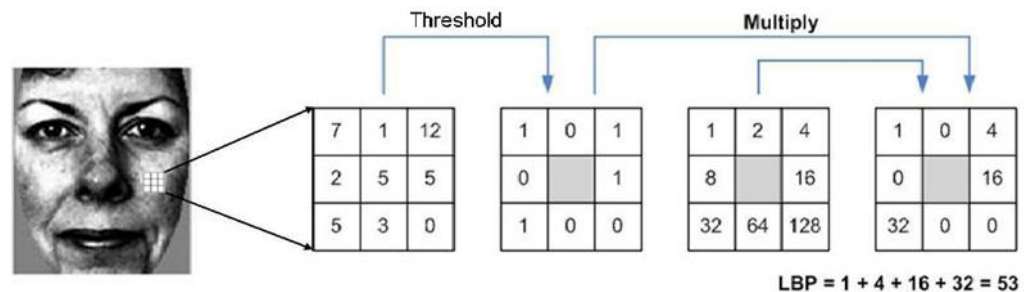


Рисунок 1.9 – Приклад методу бінарних патернів [17]

Метод широко використовується в розпізнаванні об'єктів та обличч, оскільки він є простим і швидким в обчисленні.

Глибокі нейронні мережі

З появою глибокого навчання, виділяти ознаки почали за допомогою згорткових нейронних мереж (CNN). Такі мережі автоматично вчаться розпізнавати різні патерни з великих наборів даних, що значно покращує результати в задачах розпізнавання.

Сегментація об'єктів (Object Segmentation)

Сегментація об'єктів — це етап, при якому зображення розділяється на кілька частин або "сегментів", де кожен сегмент відповідає окремому об'єкту або області. Метою сегментації є полегшення аналізу зображення, методом відокремлення об'єктів від фону та один від одного, що полегшує їх подальшу ідентифікацію та аналіз, а також підвищує виявлення та розпізнавання об'єктів у зображеннях, особливо в складних сценах, де багато об'єктів або фонових елементів.

До основних методів сегментизації належать: порогова сегментація (Thresholding), сегментація на основі кластеризації (Clustering-based Segmentation), сегментація на основі графів (Graph-based Segmentation), метод активних контурів (Active Contours або Snakes), семантична сегментація (Semantic Segmentation), масштабна сегментація (Instance Segmentation).

Порогова сегментація (Thresholding): Один з найпростіших підходів до сегментації. Працює на основі вибору порогу яскравості, після якого пікселі поділяються на дві категорії: ті, що мають значення вище порогу (наприклад, об'єкт), і ті, що мають значення нижче порогу (фон). Основна формула [7]:

$$I(x,y) = \begin{cases} 1, & \text{якщо } I(x,y) \geq T \\ 0, & \text{якщо } I(x,y) < T \end{cases}$$

де $I(x,y)$ — інтенсивність пікселя в точці (x,y) , а T — порогове значення.

Сегментація на основі кластеризації (Clustering-based Segmentation): використовує алгоритми кластеризації для групування пікселів на основі їх подібних властивостей, таких як колір або інтенсивність. Наприклад, пікселі одного кольору чи текстури будуть віднесені до одного кластеру, а інші до іншого. Це дозволяє розділити зображення на кілька областей, які можуть відповідати різним об'єктам. Один з популярних алгоритмів — k-means — кластеризує пікселі на k груп. Формула для оновлення центрів кластерів виглядає так [4]:

$$C_i = \frac{1}{|S_i|} \sum_{x \in S_i} x,$$

де: C_i — центр i -го кластеру, S_i — множина пікселів, які належать до кластеру i , x — піксель у кластері S_i .

Алгоритм мінімізує функцію втрат [7]:

$$\sum_{i=1}^k \sum_{x \in S_i} \|x - C_i\|^2$$

де $\|x - C_i\|$ — відстань між пікселем x та центром кластеру C_i .

Сегментація на основі графів (Graph-based Segmentation): В цьому методі пікселі зображення представляються як вузли графа, а схожість між ними як ваги ребер графа. Графовий підхід часто використовує алгоритм Мінкут, де пікселі — це вузли графа, а зв'язки між ними мають ваги, що відповідають схожості пікселів. Основна мета — мінімізувати функцію енергії [7]:

$$E(A) = \text{Cut}(A,B) + \lambda \sum_{i \in A, j \in B} R(i,j),$$

де A і B — сегменти, $\text{Cut}(A,B)$ — сума ваг ребер між сегментами, $R(i,j)$ — різниця між пікселями i та j , λ — коефіцієнт регулювання між внутрішньою і між сегментною енергією.

Активні контури (Active Contours або Snakes): метод використовує замкнуту криву, яка поступово деформується, наближаючись до краю об'єкта в зображенні. Алгоритм шукає мінімум енергетичної функції, яка залежить від особливостей зображення (наприклад, градієнтів), щоб обмежити форму об'єкта. Алгоритм показує високу ефективність для складних об'єктів з нерівними межами. Формула енергії для кривої $C(s)$ виглядає так [18]:

$$E_{\text{snake}} = \int_0^1 (\alpha |C'(s)|^2 + \beta |C''(s)|^2 + E_{\text{ext}}(C(s))) ds,$$

де $C(s)$ — контур, що описує об'єкт, α — коефіцієнт для контролю пружності (наскільки легко контур може розтягуватися), β — коефіцієнт для контролю жорсткості (наскільки контур є гладким), $E_{\text{ext}}(C(s))$ — зовнішня енергія, що враховує градієнти зображення.

Семантична сегментація (Semantic Segmentation) : Це метод, який використовує нейронні мережі, зокрема глибокі згорткові нейронні мережі (CNN), щоб класифікувати кожен піксель зображення як частину певного об'єкта або класу об'єктів (наприклад, людина, автомобіль, дерево) [19]. Такий підхід дозволяє не лише розділити зображення на сегменти, але й дати кожному сегменту певну категорію.

Вихідна карта для кожного пікселя формується за допомогою згортки:

$$y = \sigma (W * x + b),$$

де W — ваги згорткових фільтрів, x — вхідне зображення, b — зсув (bias), $*$ — операція згортки, σ — функція активації (наприклад, softmax для класифікації пікселів).

Функція втрат для сегментації часто є перехресною ентропією:

$$L = -\sum_{i=1}^N \sum_{c=1}^C y_{i,c} \log(\hat{y}_{i,c}),$$

де N — кількість пікселів, C — кількість класів, $y_{i,c}$ — істинна мітка для пікселя i і класу c , $\hat{y}_{i,c}$ — ймовірність передбаченого класу для пікселя i .

Масштабна сегментація (Instance Segmentation): Цей метод схожий на семантичну сегментацію, але з додатковим кроком виділення окремих об'єктів одного класу. Наприклад, якщо на зображенні є кілька людей, метод сегментації повинен ідентифікувати кожну людину окремо, а не просто віднести всі пікселі до одного класу "людина". Для цього використовується, наприклад, архітектура Mask R-CNN [20]. Алгоритм складається з двох основних частин: виявлення об'єктів (Region Proposal Network - RPN) з використанням функції втрат:

$$L=L_{cls} + L_{box} ,$$

де L_{cls} — втрати класифікації, L_{box} — втрати визначення меж об'єкта (регресія рамки).

Друга частина - це маска додатковий сегментаційний блок для передбачення маски об'єкта.

До переваг сегментації об'єктів можна віднести :

- Має високу точність ідентифікації об'єктів, їх меж та форм.
- Сегментація фокусується на конкретних ділянках зображення, що зменшує обсяг обробки і поліпшує точність.
- Підвищення точності аналізу сцени, особливо у складних умовах, таких як перекриття об'єктів або складні тіні.

До недоліків можна віднести :

- Складність у випадку нерівномірного освітлення – у випадку якщо освітлення нерівномірне або об'єкти мають схожі текстури або кольори з фоном, сегментація може бути неточною.
- Висока обчислювальна складність: деякі методи, особливо на основі глибоких нейронних мереж, потребують великих обчислювальних ресурсів.
- Чутливість до шуму.

Пост-обробка – етап, що включає поліпшення результатів обробки зображень і корекцію помилок, які могли виникнути на попередніх стадіях. Основна мета пост-обробки — зробити результати більш точними та відповідними вимогам задачі. До етапів пост-оброки належать : фільтрація

артефактів та шумів, заповнення пропусків та корекція контурів, обчислення метрик і фільтрація об'єктів за розміром, згладжування та усунення шумів, а також перетворення результатів у необхідний формат.

Фільтрація артефактів і шумів необхідна для того, щоб видалити "шумові" пікселі або дрібні сегменти, які не відповідають об'єктам і можуть залишатися після сегментації. Для цього використовують морфологічні операції, такі як ерозія та дилатація. Ерозія - зменшує об'єкти на зображенні, видаляючи пікселі по краях. Формула має вигляд [7]:

$$A \ominus B = \{z \in E | B_z \subseteq A\},$$

де A — вихідне зображення, B — структурний елемент, а B_z — трансляція B на точку z .

Дилатація - збільшує об'єкти, додаючи пікселі по краях:

$$A \oplus B = \{z \in E | (B^s)_z \cap A \neq \emptyset\},$$

де $(B^s)_z$ — структурний елемент відображений і зсунутий на z .

Заповнення пропусків та корекція контурів необхідні для того, щоб дозаповнити контури, які можуть бути неповними після сегментації. Для цього використовують методи заповнення отворів (наприклад, методи заливки) і інтерполяцію контурів, що дозволяє зробити межі об'єктів більш чіткими і точними.

Через те, що деякі об'єкти можуть бути занадто малими або занадто великими для поставленої задачі. Використовують метрики, як-от площа, периметр, круглість, для фільтрації небажаних об'єктів.

Під час згладжування та усунення шумів використовуються фільтри для зменшення різких переходів між сегментами або шумами, такі як середньозважені фільтри або медіанні фільтри, для забезпечення більш плавного вигляду зображення без втрати деталей. Після сегментації та очищення зображення результати можуть потребувати перетворення у формат, зручний для наступних кроків.

1.2 Традиційні методи розпізнавання об'єктів

Традиційні методи розпізнавання об'єктів використовують підходи на основі ознак, що отримуються з зображень за допомогою фіксованих алгоритмів. До класичних методів розпізнавання належать: Scale-Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF) та Histogram of Oriented Gradients (HOG).

Scale-Invariant Feature Transform (SIFT) — це алгоритм комп'ютерного зору, розроблений Девідом Лоу у 1999 році для виявлення та опису локальних особливостей в зображеннях. Основна мета SIFT полягає в тому, щоб виявляти ключові точки (особливості), які є інваріантними до масштабів, обертання та деяких змін у освітленні. Основні етапи методу SIFT складаються з виявлення та локалізації ключових точок, обчислення орієнтацій на основі градієнтів інтенсивності, опису ключових точок та знаходження відповідностей між ключовими точками з різних зображень [7].

Виявлення ключових точок - відбувається за допомогою методу "Difference of Gaussian" (DoG). Ключові точки визначаються там, де DoG досягає максимуму або мінімуму. Формула використовує різницю гаусіан:

$$D(x,y,\sigma) = L(x,y,k,\sigma) - L(x,y,\sigma),$$

де $D(x,y,\sigma)$ — це різниця гаусіан у масштабному просторі, а $L(x,y,\sigma)$ — згортка зображення з гауссовим фільтром на масштабі σ .

Для кожної ключової точки визначається її точне місцезнаходження та масштаб, що допомагає видалити слабкі точки, які можуть бути спотворені шумом. Після чого, для кожної ключової точки обчислюється орієнтація на основі градієнтів інтенсивності, що дозволяє зробити ключові точки обертовими інваріантними. Градієнт інтенсивності в пікселі (x,y) можна обчислити за допомогою операторів Собеля або центральних відмінностей:

$$I_x(x,y) = I(x+1,y) - I(x-1,y)$$

$$I_y(x,y) = I(x,y+1) - I(x,y-1),$$

де I — значення інтенсивності пікселя.

Орієнтація ключової точки визначається за допомогою градієнтів, що дозволяє отримати інваріантність до обертання. Вона розраховується за формулою:

$$\theta = \text{atan2}(I_x, I_y)$$

Опис ключової точки створюється з векторів градієнтів у околі точки. Зазвичай область розбивається на сітку 4×4 , і для кожної комірки підраховуються градієнти. Формула для опису виглядає так:

$$\text{Descriptor}(i,j) = \sum_{x,y \in \text{cell}_{i,j}} \sqrt{I_x(x,y)^2 + I_y(x,y)^2},$$

де $\text{cell}_{i,j}$ — комірка у сітці.

Відповідність між ключовими точками з різних зображень може визначатися за допомогою евклідичної відстані:

$$d(a,b) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2},$$

де a і b — вектори опису ключових точок.

SIFT має широке застосування в різних галузях, таких як обробка зображень, комп'ютерне бачення, робототехніка та 3D-модельовання, завдяки своїй здатності до роботи з зображеннями, які можуть бути обернені, змінені в масштабі або зазнавати змін в освітленні, але чутливий до змін, таких як зашумлення або перешкоди на зображенні.

Speeded Up Robust Features (SURF) — це вдосконалений алгоритм для виявлення та опису ключових точок в зображеннях, який був розроблений як швидша альтернатива SIFT. SURF використовує гессіанові матриці та апроксимацію гаусових фільтрів для швидшого розрахунку, а також інтегральні зображення для прискорення обчислень. Основні етапи алгоритму SURF: виявлення ключових точок, локалізація ключових точок за шкалою, орієнтація ключових точок та формування дескрипторів [21].

Для виявлення ключових точок SURF використовує гессіанову матрицю, оскільки вона надає надійну інформацію про локальні особливості.

Гессіанова матриця в точці (x,y) для масштабу σ визначається як:

$$H(x,y,\sigma) = \begin{pmatrix} L_{xx}(x,y,\sigma) & L_{xy}(x,y,\sigma) \\ L_{xy}(x,y,\sigma) & L_{yy}(x,y,\sigma) \end{pmatrix},$$

де - $L_{xx}(x,y,\sigma)$, $L_{yy}(x,y,\sigma)$, $L_{xy}(x,y,\sigma)$ — другі похідні зображення по осях x і y після згладжування гаусовим фільтром з масштабом σ .

Для визначення наявності ключових точок використовується детермінант гессіанової матриці:

$$\det(H) = L_{xx}L_{yy} - (0.9L_{xy})^2$$

Така апроксимація дозволяє швидше знаходити потенційні ключові точки зображення.

Щоб зробити ключові точки інваріантними до обертання, SURF обчислює орієнтацію на основі напрямків градієнтів в околі ключової точки. Для цього використовується фільтр Хаара. Градієнти обчислюються як суми вікон Хаара в області радіуса r , визначеного масштабом ключової точки.

Фільтр Хаара (Haar filter) — це простий фільтр для обчислення градієнтів інтенсивності в зображеннях. Фільтри Хаара використовуються для виявлення особливостей, таких як крайові, кутові або текстурні елементи зображень, і часто застосовуються в методах розпізнавання облич або інших об'єктів [22].

Фільтр Хаара для горизонтальних країв дозволяє виявляти горизонтальні зміни в інтенсивності:

$$H_x(x,y) = I(x+1,y) - I(x,y)$$

Це дві прямокутні області, одна світла (позитивна), а інша темна (негативна), розташовані поруч по горизонталі.

Фільтр Хаара для вертикальних країв використовується для виявлення вертикальних змін.

Фільтр Хаара для кутів або діагональних змін використовує більш складний шаблон, який складається з чотирьох областей (дві світлі та дві темні), розташованих як квадрати, що дозволяє виявляти кутові або діагональні зміни інтенсивності.

Орієнтація ключової точки обчислюється за допомогою найбільш значущого напрямку градієнта:

$$\theta = \text{atan2}(\sum_i w_i \times g_i^y, \sum_i w_i \times g_i^x),$$

де: g_i^x та g_i^y — градієнти по осях x і y , а w_i — ваги на основі відстані до ключової точки.

Кожна ключова точка описується за допомогою вектора дескриптора, який складається з інформації про інтенсивність та напрямок градієнтів у місцевій області.

Опис ключової точки формують таким чином - область навколо ключової точки ділиться на підрегіони 4×4 . Після чого, для кожного підрегіону обчислюються суми градієнтів Хаара за напрямками x і y (та їх абсолютні значення):

$$v = (\sum_x, \sum_y, \sum_{|x|}, \sum_{|y|})$$

Отриманий вектор дескриптора містить 64 елементи (для сітки 4×4).

Як і в SIFT, відповідність ключових точок між зображеннями визначається на основі евклідидної відстані між дескрипторами.

Переваги SURF:

1. Підвищення швидкості - використання інтегральних зображень та апроксимація гауссових похідних значно прискорює алгоритм порівняно з SIFT.

2. SURF є інваріантним до масштабу, повороту, і частково стійким до зміни освітлення та перспективних трансформацій.

До недоліків можна віднести меншу точність порівняно з SIFT при роботі зі складними об'єктами та меншу стійкість до шуму і змін освітлення.

Histogram of Oriented Gradients (HOG) — це метод для виділення ознак зображення, що полягає у використанні градієнтів інтенсивності зображення для опису локальних текстурних ознак у вигляді гістограми напрямків градієнтів. Він розбиває зображення на менші ділянки і для кожної ділянки будує гістограму напрямків градієнтів [23]. HOG широко застосовується в задачах розпізнавання об'єктів, зокрема пішоходів.

До основних етапів алгоритму належать:

- Обчислення градієнтів зображення.

- Побудова гістограм орієнтованих градієнтів для локальних ділянок зображення.
- Нормалізація гістограм для кожного блоку.
- Формування вектора ознак на основі цих гістограм.

Обчислення горизонтальних (G_x) і вертикальних (G_y) градієнтів для кожного пікселя виконується за допомогою фільтрів Собеля або шляхом розрахунку різниць інтенсивностей між сусідніми пікселями.

Формули для градієнтів:

$$G_x = I(x+1,y) - I(x-1,y)$$

$$G_y = I(x,y+1) - I(x,y-1),$$

де $I(x,y)$ — інтенсивність пікселя в точці (x,y) , G_x і G_y — компоненти градієнта в горизонтальному та вертикальному напрямках відповідно.

Після обчислення горизонтальних і вертикальних градієнтів для кожного пікселя, можна обчислити величину градієнта та його орієнтацію:

$$\text{Magnitude} = \sqrt{G_x^2 + G_y^2}$$

$$\text{Orientation} = \theta = \tan^{-1}\left(\frac{G_y}{G_x}\right),$$

де Magnitude — величина градієнта, а θ — його напрямок.

Після чого відбувається побудова гістограм градієнтів. Зображення ділиться на малі регіони, відомі як блоки або клітинки (звичайно розміром 8×8 або 16×16 пікселів). Для кожної клітинки будується гістограма напрямків градієнтів. Напрямки розбиваються на діапазони (наприклад, на 9 бінів з кроком у 20 градусів для діапазону від 0° до 180°). Кожен піксель клітинки вносить свій внесок до відповідного біна, пропорційно величині градієнта.

Для підвищення стійкості до змін освітлення, гістограми клітинок нормалізуються. Нормалізація виконується для блоків, що складаються з декількох клітинок (наприклад, блоки 16×16 пікселів, що містять чотири клітинки). Це робиться для згладжування локальних змін освітлення.

Формула нормалізації блоку:

$$V_{\text{norm}} = \frac{v}{\sqrt{\|v\|^2 + e^2}}$$

Після нормалізації гістограм блоків вектори для кожного блоку об'єднуються у довгий вектор ознак, що описує все зображення. Цей вектор подається на вхід моделі класифікації (наприклад, SVM), що приймає рішення про наявність об'єкта.

Переваги HOG:

1. Інваріантність до змін освітлення: нормалізація допомагає зменшити вплив локальних змін освітлення.
2. Стійкість до геометричних трансформацій: HOG добре працює для виявлення об'єктів з подібними контурами, навіть якщо вони трохи зміщені або обернені.

Недоліки HOG:

1. Для об'єктів різних розмірів потрібно змінювати параметри блоку або виконувати багато масштабний аналіз.
2. Високі обчислювальні витрати.

Традиційні методи, як SIFT, SURF і HOG, мають такі переваги: легка інтерпретація та простота у створенні та використанні, а також добре працюють на задачах з невеликою кількістю даних та обмеженими обчислювальними ресурсами.

Але у порівнянні з сучасними методами, які базуються на нейронних мережах, вони мають недостатню масштабованість та меншу точність.

Методи на основі шаблонів використовуються для пошуку об'єктів на зображенні шляхом порівняння його з заданим еталоном або шаблоном (template). Ці методи часто застосовуються в задачах комп'ютерного зору для виявлення або локалізації об'єктів, коли є відомий зразок, який потрібно знайти на нових зображеннях. Один із класичних підходів — це Template Matching.

Template Matching — це один з найпростіших методів. Він полягає в тому, що відомий шаблон зображення ковзає по всьому зображенню, і в кожній позиції обчислюється подібність між шаблоном і поточним регіоном зображення. Цей

метод використовує кількісні метрики для порівняння. Для вимірювання подібності використовуються [24]:

Нормалізована крос-кореляція (Normalized Cross-Correlation, NCC):

$$R(xy) = \frac{\sum_{i,j} [T(i,j) \times I(x+i, y+j)]}{\sqrt{\sum_{i,j} T(i,j)^2 \times \sum_{i,j} I(x+i, y+j)^2}},$$

де $T(i,j)$ — пікселі шаблону, а $I(x+i, y+j)$ — пікселі вікна зображення, з якими порівнюється шаблон. Така операція нормалізує крос-кореляцію, щоб компенсувати зміну освітлення і масштабу яскравості.

Середньоквадратичне відхилення (Mean Squared Error, MSE) вимірює різницю між пікселями шаблону і пікселями зображення і визначається за формулою:

$$MSE(x,y) = \frac{1}{N} \sum_{i,j} (T(i,j) - I(x+i, y+j))^2,$$

де N — кількість пікселів у шаблоні.

Абсолютна різниця (Sum of Absolute Differences, SAD) обчислює суму абсолютних різниць між пікселями шаблону та зображення:

$$SAD(x,y) = \sum_{i,j} |T(i,j) - I(x+i, y+j)|$$

Template Matching проста у реалізації та підходить для завдань, де шаблон чітко визначений і відомий, і має непогані показники точності для об'єктів з незмінними розмірами, положенням і орієнтацією.

До недоліків можна віднести чутливість до масштабування та обертання об'єктів на зображенні, чутливість до зміни освітлення, а також великі обчислення, особливо для великих зображень та шаблонів.

1.3 Розпізнавання об'єктів на основі глибокого навчання.

Згорткові нейронні мережі (CNN) — це один із основних типів глибоких нейронних мереж, спеціалізований для роботи із зображеннями. CNN добре підходять для задач комп'ютерного зору, таких як розпізнавання об'єктів, завдяки своїй здатності автоматично виділяти ключові ознаки зображень, можливості до перенавчання та масштабування.

Згортковий шар — основний будівельний блок CNN. У ньому використовуються невеликі фільтри (ядра), які «проходять» через зображення, виконуючи операцію згортки (конволюції). Це дозволяє моделі автоматично виявляти ключові ознаки, такі як границі, кути, текстури тощо, без ручного втручання. Типова структура згорткової нейронної мережі представлена на рисунку 1.10.

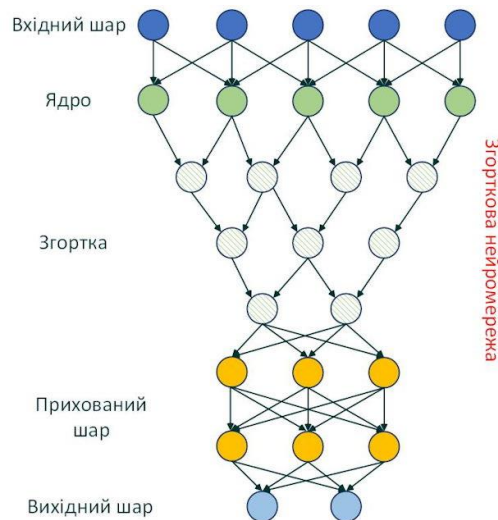


Рисунок 1.10 – Типова структура згорткової нейронної мережі [26]

Процес згортки включає:

Фільтрацію зображень - ядро (зазвичай 3x3 або 5x5) переміщується по зображенню (піксель за пікселем), виконуючи операцію множення елементів ядра на відповідні пікселі зображення, і підсумовуючи їх. В результаті отримується «карта ознак» (feature map), яка містить інформацію про певні особливості (наприклад, границі об'єктів) [7]. Нехай X — вхідне зображення, K — ядро, i, j — індекси пікселів зображення, а m, n — індекси елементів ядра. Операція згортки для отримання значення у позиції i, j на карті ознак Y виглядатиме так:

$$Y(i, j) = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} X(i + m, j + n) \times K(m, n),$$

де $X(i+m, j+n)$ — піксель зображення, $K(m, n)$ — елемент ядра розміром $k \times k$, $Y(i, j)$ — відповідний елемент на карті ознак.

Встановлення кроку та рамки (Stride та Padding) : Stride визначає, наскільки сильно зсувається ядро при кожному кроці (зазвичай 1 або 2). Padding додає

рамку навколо зображення, щоб зберегти розмірність карти ознак після конволюції, оскільки без нього розміри зменшуються.

Після кожного згорткового шару застосовується активаційна функція, зазвичай ReLU (Rectified Linear Unit) [26]. ReLU встановлює всі негативні значення в карті ознак рівними нулю, що додає нелінійність до моделі, допомагаючи мережі краще виявляти складні взаємозв'язки між пікселями. Формула для ReLU така:

$$f(x) = \max(0, x),$$

де x — значення після згортки (конволюції).

Після конволюційних шарів часто застосовується шар зниження розмірності або pooling-шар, наприклад, MaxPooling [27]. Він зменшує просторові розміри карти ознак, зберігаючи при цьому важливі ознаки. MaxPooling - вибирає максимальне значення з групи пікселів (наприклад, з блоку 2×2), таким чином зменшуючи розмір карти ознак, але зберігаючи найбільш важливі ознаки. Нехай Y — карта ознак після конволюції. Тоді для кожного блоку розміром $p \times p$:

$$Y_{\text{pool}}(i, j) = \max(Y(i, j), Y(i+1, j), \dots, Y(i+p-1, j+p-1))$$

AveragePooling -використовується рідше, і бере середнє значення з блоку пікселів, замість максимального.

Після кількох шарів згортки і pooling, CNN створює багато «карт ознак», кожна з яких містить певну інформацію про різні аспекти зображення. На ранніх рівнях мережа виділяє прості ознаки, такі як границі та кути, а на глибших рівнях вона починає розпізнавати більш складні структури, такі як форми і об'єкти.

Мережа переводить отримані карти ознак в одновимірний вектор і подає його на повнозв'язний шар. Вихід кожного нейрона цього шару визначається наступною формулою:

$$z = \sum_{i=1}^n w_i \times x_i + b,$$

де w_i — ваги нейронів, x_i — вхідні значення (елементи сплющеного вектора), b — зміщення (bias).

Це як класична нейронна мережа, де кожен нейрон пов'язаний з кожним наступним нейроном. Тут мережа використовує всі виділені ознаки для прийняття рішення, до якого класу належить зображення.

Для класифікації об'єктів застосовують Softmax або Sigmoid функції в залежності від кількості класів [28]. Ці функції перетворюють виходи повнозв'язного шару в ймовірності, які вказують, до якого класу належить зображення (наприклад, автомобіль, кішка або собака). Формула для Softmax виглядає так:

$$P(y = k|x) = \frac{e^{z_k}}{\sum_{j=1}^K e^{z_j}},$$

де z_k — активація k -го нейрона в повнозв'язному шарі, K — кількість класів.

Функція Sigmoid використовується для бінарної класифікації:

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Вона перетворює вихід у значення від 0 до 1, що можна інтерпретувати як ймовірність приналежності до певного класу.

До основних переваг згорткових нейронних мереж можна віднести автоматичне виділення ознак зображень, стійкість до зсуву, масштабування та обертання об'єктів на зображенні, компактність та можливість застосування на пристроях з обмеженими обчислювальними можливостями, висока точність розпізнавання, підтримка великих баз даних та бібліотек, можливість перенавчання мережі під нові задачі.

До недоліків можна віднести – великі потреби у обчислювальних потужностях, у випадку застосування дуже глибоких моделей з великою кількістю шарів, необхідність у навчальних даних та потенційні проблеми при неправильному балансуванні даних.

Сучасні архітектури для розпізнавання об'єктів, такі як Faster R-CNN, YOLO (You Only Look Once) та SSD (Single Shot Multibox Detector), MobileNet стали основними інструментами у задачах виявлення та локалізації об'єктів на

зображеннях. Кожна з цих архітектур має свої унікальні характеристики, що дозволяють досягати ефективних і точних результатів.

Faster R-CNN (Region-Based Convolutional Neural Networks).

Faster R-CNN є поліпшеною версією попередніх моделей R-CNN і Fast R-CNN. Вона використовує ідею регіональних пропозицій, але є набагато швидшою завдяки спрощенню та інтеграції ключових компонентів. Архітектура Faster R-CNN складається з регулярного CNN (backbone), мережі пропозицій регіонів (RPN), операції ROI Pooling, класифікації та регресії рамок [29].

Спочатку вхідне зображення обробляється базовою CNN для отримання карти ознак (зазвичай використовуються ResNet або VGG-16) РИС 1.11. Далі застосовується мережа пропозицій регіонів (RPN), яка генерує потенційні області (регіони) для об'єктів. Ці регіони є пропозиціями, які вказують на ймовірні області наявності об'єктів. Після чого виділені регіони на карті ознак подаються на рівномірні квадрати через операцію ROI pooling. Кожна область пропозицій обробляється класифікатором для визначення об'єкта та регресором для уточнення координат об'єктної рамки.

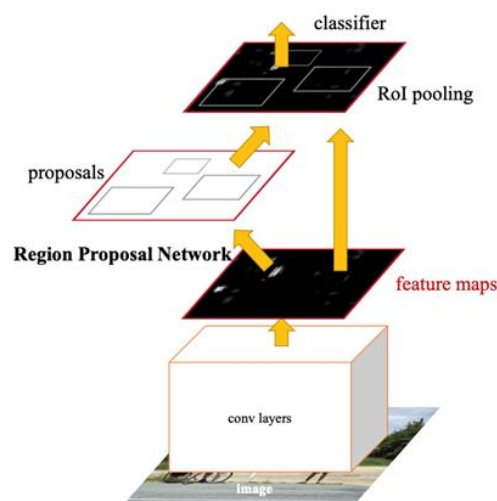


Рисунок 1.11 – Типова структура R-CNN [29]

До переваг Faster R-CNN можна віднести високу точність завдяки двоетапному підходу, який спочатку виділяє потенційні області, а потім уточнює їх та гнучкість для виявлення різноманітних об'єктів у зображенні.

До недолік можна віднести швидкість роботи, у порівнянні з іншими архітектурами, такими як YOLO або SSD та MobileNet, оскільки обробка відбувається в два етапи (спочатку пропозиції регіонів, потім їх класифікація).

YOLO (You Only Look Once).

YOLO — це один з найшвидших підходів для виявлення об'єктів у реальному часі [30]. Основна ідея полягає в тому, щоб не ділити завдання на кілька етапів (як у Faster R-CNN), а одночасно виконувати як класифікацію об'єктів, так і локалізацію їхніх рамок за один прохід через мережу. YOLO розбиває зображення на сітку (наприклад, 7×7) і для кожної клітинки передбачає можливість наявності об'єкта та відповідні координати рамки (bounding box). Кожна клітинка відповідає за передбачення об'єкта, його координат та ймовірностей класу.

Мережа навчається визначати не тільки класи, але й регресувати координати рамок. Загальна втрата (loss) містить кілька складових — для правильності класифікації, для точності рамок і для виявлення наявності об'єкта.

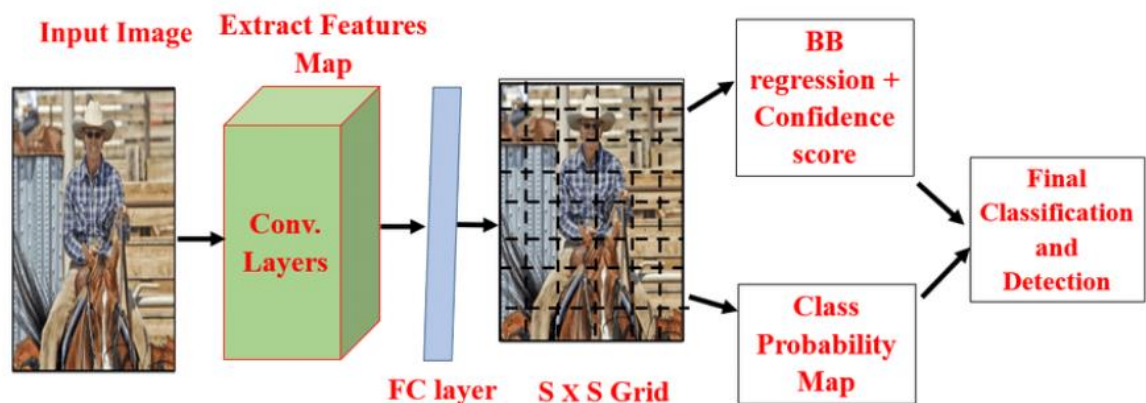


Рисунок 1.12 – Типова архітектура YOLO [30]

До переваг можна віднести високу швидкість розпізнавання та можливість працювати з зображеннями у режимі реальному. Однак, точність локалізації може бути обмежена, оскільки система розбиває зображення на фіксовану сітку.

Архітектура SSD (Single Shot Multibox Detector).

SSD —архітектура для швидкого виявлення об'єктів, яка використовує одноетапний підхід, але при цьому досягає більшої точності, ніж YOLO. SSD здійснює класифікацію та локалізацію об'єктів у різних масштабах одночасно [31].

В основі SSD лежить стандартна нейронна мережа (VGG-16 або ResNet), яка генерує карти ознак для зображення. SSD робить передбачення об'єктів на основі кількох шарів різних розмірів карт ознак, що дозволяє виявляти об'єкти різних масштабів рис 1.13. Кожен шар прогнозує не тільки класи об'єктів, але й координати рамок.

Для кожного рівня карти ознак використовуються так звані anchor boxes (або попередньо визначені рамки), що відповідають різним співвідношенням сторін. Кожен такий box регресується на основі фактичної позиції об'єкта.

SSD досягає кращого балансу між швидкістю і точністю, ніж YOLO, завдяки використанню багатошарових карт ознак для обробки об'єктів різного розміру, а також може працювати в реальному часі, зберігаючи високу точність виявлення. SSD, як і інші одноетапні моделі, може мати труднощі з точним виявленням дуже малих об'єктів через масштабність передбачень.

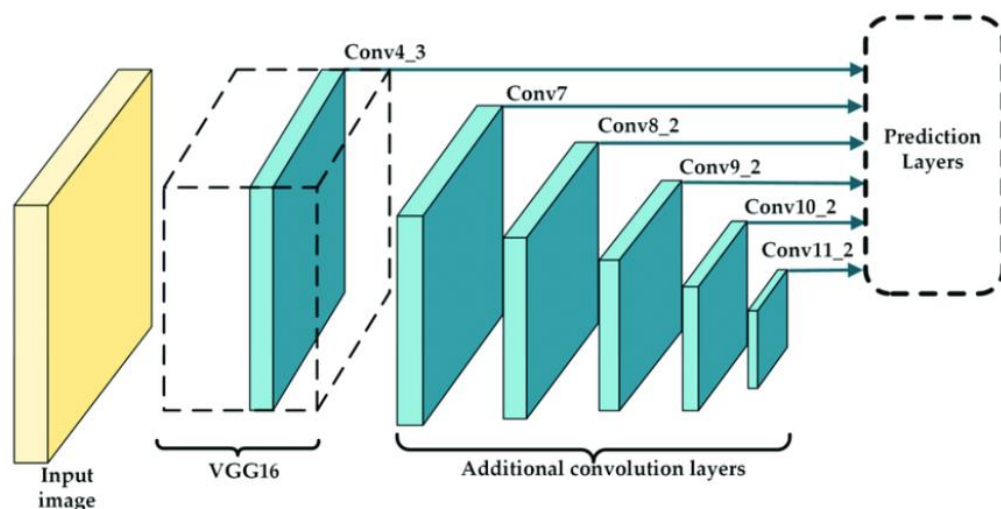


Рисунок 1.13 - Типова модель Single-Shot MultiBox Detector (SSD), що складається з VGG і додаткових шарів згортки [31]

MobileNet

MobileNet — це архітектура згорткових нейронних мереж (CNN), яка була розроблена для ефективного розпізнавання об'єктів на мобільних пристроях та пристроях з обмеженими обчислювальними можливостями, такими як мікроконтролери. Її головна перевага полягає в тому, що вона зберігає баланс між точністю і продуктивністю, забезпечуючи менші вимоги до обчислювальних ресурсів і енерговитрат. Одна з основних інновацій MobileNet полягає у використанні глибинно-сепарабельних згорток замість стандартних згорткових операцій [32]. Це дозволяє значно зменшити кількість обчислень і параметрів, що робить модель більш легкою та придатною для мобільних пристроїв.

Глибинно-сепарабельні згортки розбивають стандартну згорткову операцію на два окремих кроки:

1. Depthwise Convolution: Застосовується окремий фільтр для кожного каналу вхідного зображення, що дозволяє обробляти глибину окремо для кожного каналу.
2. Pointwise Convolution (1x1 Convolution): Застосовується згортка 1x1 для комбінування результатів з кожного каналу після попередньої операції.

Для забезпечення гнучкості та можливості налаштування продуктивності, MobileNet використовує два гіперпараметри: Width Multiplier (α) що визначає, наскільки вузькою або широкою буде модель. Цей параметр зменшує кількість каналів на кожному шарі мережі, зменшуючи обчислювальні вимоги. Наприклад, якщо ширина множника становить 0.5, то кількість каналів на кожному шарі буде половиною від початкової кількості. Значення множника змінюється в діапазоні від 0 до 1. Чим менше значення, тим легша модель, але й нижча точність. Resolution Multiplier (ρ) - змінює розмір вхідного зображення. Зменшення роздільної здатності зображення знижує вимоги до обчислень і пам'яті. Менша роздільна здатність вхідних зображень може призвести до незначного зниження точності, але значно знижує обчислювальні витрати.

Загальна структура MobileNet складається з набору глибинно-сепарабельних згорткових шарів, які замінюють стандартні згорткові шари, зменшуючи обчислювальну складність, послідовність Depthwise і Pointwise

згорток, після яких додаються операції активації (ReLU) та нормалізації (Batch Normalization). Наприкінці використовуються стандартні шари класифікації (Fully Connected Layers) для передбачення класів об'єктів. Типова мережа складеться з :

- Вхідного шару для обробки зображення.
- Серії згорток по глибині для виділення ознак.
- Шарів максимального об'єднання (Max Pooling) для зменшення розмірів ознак.
- Фінального повнозв'язного шару для передбачення класів.

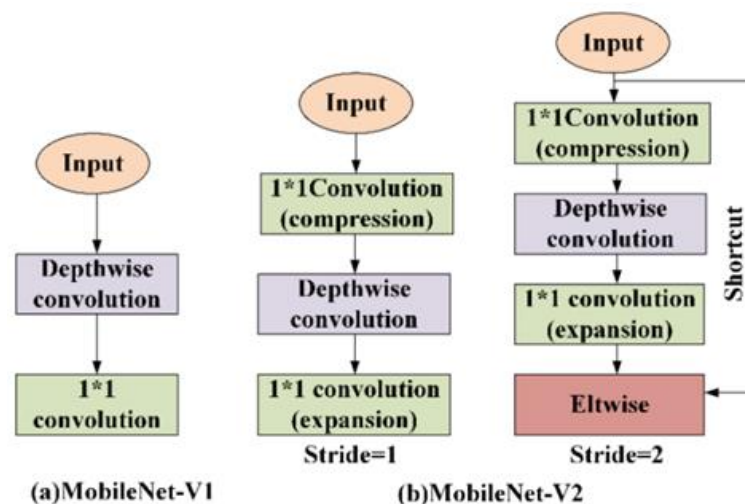


Рисунок 1.14 - Типова структура архітектур MobileNet-V1 (a), та MobileNet-V2 (b) [33]

До переваг мережі MobileNet можна віднести високу швидкість та точність розпізнавання, при відносно невеликих обчислювальних потребах, у порівнянні з іншими архітектурами. Підходить для розгортання на пристроях з обмеженими обчислювальними можливостями. Гнучкі налаштування між точністю, продуктивністю та розміром моделі отримуються завдяки кофіцієнтам ширини мережі та можливості задання розміру зображення. При цьому, мережа може мати труднощі з точним розпізнаванням дуже малих об'єктів через зменшену кількість ознак.

1.4 Системи розпізнавання об'єктів для людей з вадами зору.

Технології розпізнавання об'єктів стали важливими для людей з вадами зору, за даними Всесвітньої організації охорони здоров'я, на сьогодні у світі понад 1 мільярд людей страждає від порушень зору, з яких понад 43 мільйони є незрячими. Використання систем розпізнавання зображень на основі нейронних мереж відкриває нові можливості для покращення можливостей орієнтації у просторі та доступу до інформації. Ці системи, здатні ідентифікувати об'єкти в реальному часі та надавати голосові підказки, виконувати зчитування тексту, розпізнавати валюту та обличчя. Системи та мобільні додатки такі як Seeing AI, OrCam, Be My Eyes та Airpoly Vision використовують комп'ютерний зір для допомоги орієнтації у просторі незрячим людям.

Seeing AI — це безкоштовний мобільний додаток від Microsoft, створений для допомоги незрячим і людям з вадами зору орієнтуватися в навколишньому середовищі [34]. Додаток доступний для iOS, використовує штучний інтелект і технології комп'ютерного зору для розпізнавання об'єктів, тексту, облич, валюти та інших елементів, що можуть бути корисними для користувача в реальному часі. До основних можливостей входить: розпізнавання короткого тексту, режим читання документу, розпізнавання продуктів та валюти, розпізнавання та опис деяких об'єктів.

OrCam — це компанія, яка створила пристрої для людей з вадами зору та слуху, серед яких найпопулярніші OrCam MyEye та OrCam Read. Ці портативні пристрої працюють на основі штучного інтелекту, дозволяючи користувачам зчитувати текст, розпізнавати обличчя, ідентифікувати продукти та розпізнавати купюри. Пристрої OrCam 2 Pro використовують штучний інтелект і технології комп'ютерного зору, що дозволяє їм працювати без підключення до інтернету — усі дані обробляються локально, забезпечуючи високу приватність та швидкість роботи [35]. OrCam MyEye 2 Pro — це невеликий, легкий пристрій, який кріпиться до оправ окулярів. Він має камеру та динамік і може читати текст, розпізнавати обличчя, ідентифікувати продукти, розпізнавати валюту рис 1.15. Ціна варіюється в межах від 4400 до 5000 доларів США в залежності від моделі.



Рисунок 1.15 – Зовнішній вигляд OrCam MyEye 2 Pro [35]

OrCam Read — це портативний пристрій у формі ручки, призначений для людей з вадами читання, наприклад, для дислексиків або людей зі слабким зором рис 1.16 [36]. Він може зчитувати текст зі сторінки або екрана, допомагаючи користувачам читати книги, документи чи меню. При цьому OrCam Read має дві режими читання: перший для вибіркового тексту і другий — для повного. Ціна варіюється в межах від 1500 до 2000 доларів США в залежності від моделі.



Рисунок 1.16 – Зовнішній вигляд OrCam Read [36]

Be My Eyes — це безкоштовний мобільний додаток, що з'єднує незрячих або слабозорих людей із волонтерами з усього світу для надання зорової допомоги в режимі реального часу [37]. Ідея програми полягає в тому, щоб користувачі з вадами зору могли швидко отримати допомогу у вирішенні побутових чи інших візуальних задач за допомогою відеодзвінка. Користувач з вадами зору відкриває додаток і натискає кнопку «Попросити про допомогу». Це надсилає запит у мережу волонтерів. Перший доступний волонтер отримує сповіщення і підключається до користувача через відеодзвінок. Через камеру телефону волонтер може бачити те, що показує користувач, і допомагати йому

виконувати різні завдання. Ве My Eyes працює у понад 150 країнах і підтримує більш ніж 180 мов. Додаток створив інклюзивну спільноту, яка дозволяє незрячим людям бути незалежнішими та отримувати допомогу у повсякденному житті.

Aipoly Vision — це мобільний додаток, створений для допомоги людям із вадами зору, що використовує штучний інтелект для розпізнавання об'єктів та кольорів у реальному часі, а всі обчислення виконуються безпосередньо на пристрої [38].

Aipoly Vision не вимагає натискання кнопки для ідентифікації об'єкта — він автоматично аналізує все, що потрапляє в поле зору камери та розпізнає широкий спектр об'єктів: від побутових предметів і меблів до продуктів харчування та озвучує назву. Aipoly Vision доступний для пристроїв на iOS та Android. Інтерфейс додатка розроблено так, щоб він був максимально доступним для користувачів з вадами зору, з підтримкою великих шрифтів, голосових команд та інших функцій.

Існують також різноманітні саморобні носимі пристрої, що використовують ультразвукові сенсори для виявлення перешкод та вільного простору. Наприклад, спеціальні рукавички, що передають тактильні сигнали, і вказують на напрямок руху. Інформація, отримана з датчиків, перетворюється на вібрацію, що дозволяє незрячому користувачеві визначати наявність об'єктів навколо себе. Розумні окуляри - це один з найпопулярніших форм факторів приладів допомоги для незрячих користувачів. Вони обладнані датчиками, що за допомогою вібрації сповіщають про відстань до об'єкту в режимі реального часу [39].

Більшість робіт, які розглядають використання систем розпізнавання об'єктів для людей з вадами зору зосередженні на навчанні нейронних мереж і отриманні високих показників точності розпізнавання об'єктів, при цьому не вирішуючи практичних проблем, таких як частота оновлення інформації та оголошення інформації з урахуванням того, що слова різної довжини потребують різного часу на їх оголошення та інших практичних задач.

Висновки до розділу 1

У результаті проведеного аналізу встановлено, що на даний момент існують декілька підходів до розпізнавання об'єктів на зображеннях. Для створення системи розпізнавання об'єктів з подальшим голосовим виводом інформації для людей з вадами зору ефективніше використовувати систему на основі згорткових нейронних мереж.

Нейронні мережі мають високу швидкість і точність розпізнавання об'єктів на зображеннях, здатні працювати у режимі реального часу, гнучкі у налаштуваннях та досить стійкі до різних видів деформації та шумів, у порівнянні з класичними методами. Також, однією з головних переваг є можливість до масштабування нейронних мереж, що дозволяє використовувати їх на пристроях з обмеженими потужностями, такими як мікроконтролери та одноплатні комп'ютери. Наявність відкритих баз тренувальних зображень, можливість до перенавчання, а також велика кількість відкритих бібліотек, роблять нейронні мережі більш зручними у використанні.

Існуючі пристрої, що призначені для полегшення життя людей з вадами зору здебільшого вміють читати текст, і розраховані на людей, які мають поганий зір, але абсолютно не розраховані на людей які його ніколи не мали. Існуючі пристрої на базі детектора відстані не вирішують питань, і не використовуються на практиці. Тому створення системи розпізнавання з подальшим голосовим оголошенням інформації є актуальним і не вирішеним завданням на сьогоднішній день.

РОЗДІЛ 2. АНАЛІЗ ПРОГРАМНИХ ТА АПАРАТНИХ ЗАСОБІВ ДЛЯ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ ТА ГЕНЕРУВАННЯ ГОЛОСУ

2.1 Дослідження нейронних мереж різних класів для розпізнавання об'єктів на пристроях з обмеженими можливостями.

Нейронні мережі стали одним із ключових інструментів у системах розпізнавання об'єктів завдяки своїй здатності обробляти великі обсяги даних та знаходити складні патерни, які важко виявити традиційними методами розпізнавання. Сучасні системи на основі нейронних мереж здатні ефективно аналізувати візуальні дані у режимі реального часу, виділяючи важливі характеристики об'єктів та розпізнаючи їх з високою точністю навіть у випадках, коли зображення містить шуми, перешкоди або об'єкти мають різні кути нахилу, розміри чи освітлення. Ця здатність дозволяє нейронним мережам перевершувати традиційні алгоритми у багатьох складних ситуаціях, які раніше вимагали значних людських зусиль для обробки. До основних переваг нейронних систем можна віднести:

- Можливість розгортання на приладах з обмеженими ресурсами, таких як мікроконтролери
- Висока точність та швидкість розпізнавання об'єктів
- Масштабованість та можливість до перенавчання

Завдяки використанню нейронних мереж, системи стають здатними більш природно реагувати на складні завдання, адаптуючись під індивідуальні потреби користувачів. У випадках з адаптивними системами для людей з вадами зору, нейронні мережі допомагають швидко ідентифікувати об'єкти та передавати цю інформацію у вигляді голосових підказок, що значно підвищує мобільність і безпеку незрячих людей у повсякденному житті.

Не всі моделі нейронних мереж для розпізнавання об'єктів можна розгорнути на мікроконтролерах. На даний момент для роботи на мікроконтролерах, підтримуються і доступні такі моделі мереж як : MobileNetV1/V2, Tiny-YOLO, EfficientNet-Lite та Edge Impulse FOMO.

MobileNet

MobileNetV1 — це архітектура нейронної мережі, розроблена для мобільних та вбудованих систем, що використовує згортки по глибині. Цей підхід замінює звичайні згортки двома окремими операціями — глибинною та точковою згорткою. Це знижує кількість обчислень і кількість параметрів, роблячи мережу ефективною для завдань розпізнавання зображень і об'єктів на пристроях з обмеженими ресурсами.

MobileNetV2 удосконалена версія, в яку додали інвертований залишок із лінійним вузьким місцем [40]. Цей модуль приймає як вхідні дані низькорозмірне стиснене представлення, яке спочатку розширюється до високого розміру та фільтрується за допомогою згортки по глибині. Згодом об'єкти проектується назад у низьковимірне представлення з лінійною згорткою. Модель також має компресовані шари з активацією ReLU6 для кращого узагальнення в обмежених ресурсах. Ці вдосконалення підвищують точність розпізнавання об'єктів при менших потребах у потужності пристрою для розгортання.

Основна ідея полягає в заміні повного згорткового оператора факторизованою версією, яка розбиває згортку на два окремі шари. Перший рівень називається згорткою по глибині, він виконує легку фільтрацію шляхом застосування одного згорткового фільтра на вхідний канал. Другий шар - це згортка 1×1 , яка називається поточною згорткою, і відповідає за створення нових функцій шляхом обчислення лінійних комбінацій вхідних каналів[] рис 2.1. Згортки, що розділяються по глибині, є заміною стандартних згорткових шарів.

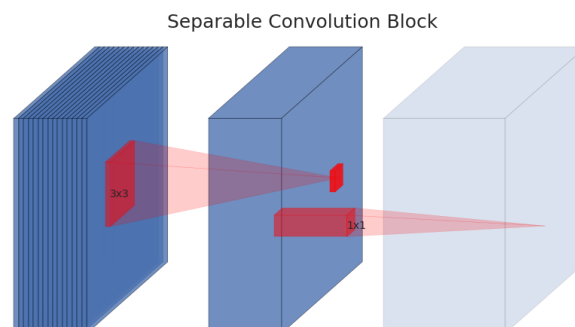


Рисунок 2.1 – Роздільний блок згортки [40]

Tiny-YOLO

Tiny-YOLO (You Only Look Once) — це спрощена версія YOLO, що підходить для розпізнавання у реальному часі завдяки низьким вимогам до обчислювальних ресурсів. Модель розпізнає об'єкти шляхом одночасного обчислення координат обмежувальних рамок і ймовірності об'єкта, що дозволяє працювати зі значно меншою затримкою, особливо на мобільних пристроях і вбудованих системах.

Tiny-YOLO має менше шарів та знижений розмір моделі в порівнянні з повною версією YOLO, що дозволяє ефективно працювати з пристроями з обмеженою пам'яттю. Попри свою компактність, вона здатна виконувати детекцію на великих зображеннях з прийнятною точністю, що робить її популярною в задачах реального часу, таких як відеоспостереження та автономне керування транспортними засобами.

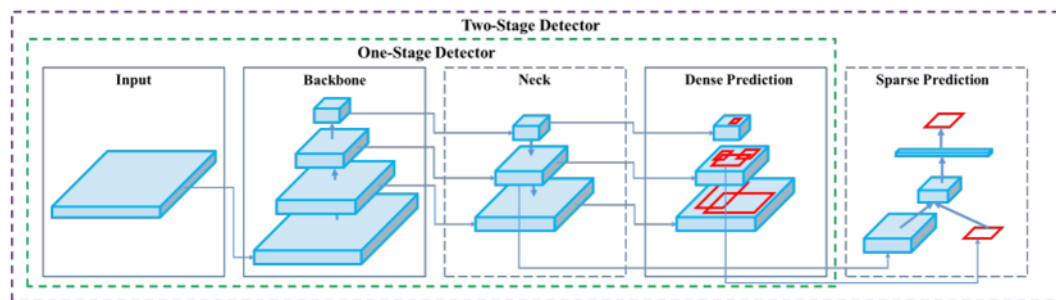


Рисунок 2.2 - Приклад блоків Tiny-YOLOv4 [41]

Однією з останніх версій є YOLOv4-tiny. Це компактна версія YOLOv4, оптимізована для швидшої обробки зображень на менш потужних пристроях. Її структура складається з базових згорткових шарів, які виявляють та класифікують об'єкти в одному проході [41]. Модель використовує дві основні детекторні головки, які обробляють особливості на різних рівнях, що дозволяє їй ефективно розпізнавати об'єкти різних розмірів. YOLOv4-tiny підтримує CSPDarknet53 як базову архітектуру, що знижує кількість параметрів, зберігаючи при цьому високу точність і швидкість.

CSPDarknet53 — це вдосконалена архітектура глибоких нейронних мереж, розроблена для швидкої та ефективної обробки зображень. Вона побудована на

основі Darknet53, але з розширеною структурою Cross Stage Partial (CSP), яка розділяє основний блок на дві частини: одна частина обробляється шарами згортки, а інша залишається незмінною. Це зменшує кількість параметрів і дублювання обчислень. CSPDarknet53 підтримує баланс між точністю та швидкістю, що робить її особливо придатною для реального часу задач, як у YOLOv4-tiny рисунок 2.3 [42].

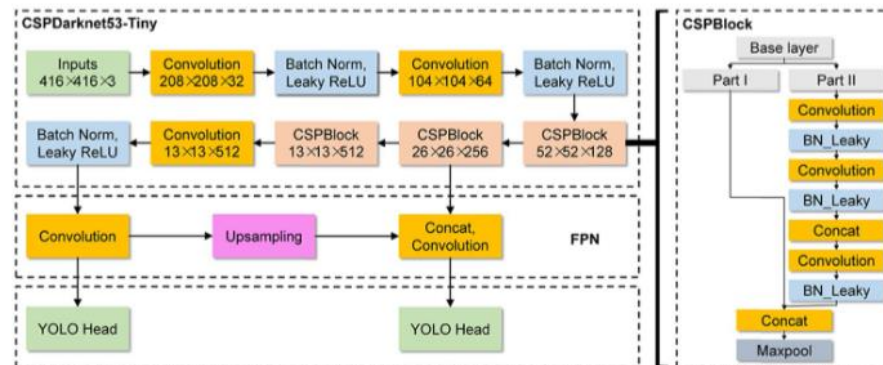


Рисунок 2.3 - Архітектура YOLOv4-tiny Network [42]

EfficientNet-Lite

EfficientNet-Lite є легкою версією EfficientNet, адаптованою для мобільних та вбудованих систем, і використовує методи масштабування моделі для оптимального співвідношення точності та швидкодії. Замість додавання шарів або нейронів, EfficientNet-Lite балансує глибину, ширину і роздільну здатність моделі для досягнення максимальної ефективності за обмежених ресурсів рисунок 2.4 [43].

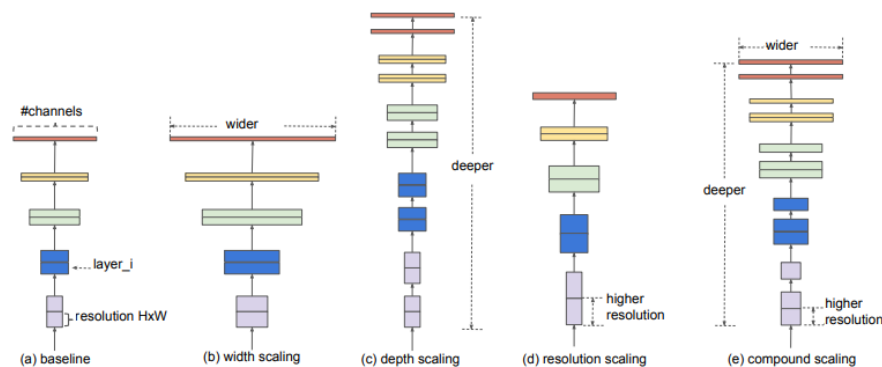


Рисунок 2.4 – Приклад різних варіантів масштабування моделі EfficientNet-Lite [43]

Завдяки архітектурі на основі біфокальних мереж (BiFPN), модель може ефективно обробляти багаторівневу інформацію для покращення результатів у задачах класифікації та розпізнавання об'єктів. EfficientNet-Lite застосовується у системах, що потребують швидкої обробки та низького енергоспоживання, як-от смартфони, розумні камери і IoT-пристрої.

Edge Impulse FOMO — це легка модель для розпізнавання об'єктів, розроблена спеціально для мікроконтролерів і малопотужних пристроїв [44]. Вона здатна працювати на платформах із пам'яттю всього до 20 КБ, що робить її ідеальною для IoT-застосунків. FOMO використовує об'єктне розпізнавання для відстеження декількох об'єктів в реальному часі. Модель розроблена для економії енергії та пам'яті, а також підтримує високу швидкодію завдяки оптимізації алгоритмів обробки зображень. FOMO використовується у задачах моніторингу навколишнього середовища, індустриальних системах спостереження та інших IoT-додатках, де є обмеження по обчислювальним ресурсам.

Базова архітектура нейронної мережі складається з ряду згорткових шарів. FOMO використовує таку саму архітектуру, але відсікає останні шари стандартної моделі класифікації зображень і замінює цей шар картою ймовірності класу регіону (наприклад, карта 4x4 у прикладі вище). Потім він має спеціальну функцію втрати, яка змушує мережу повністю зберігати локальність на останньому рівні. Цей процес створює «теплову карту» розташування об'єктів. Роздільна здатність теплової карти визначається тим, де відрізати шари мережі. FOMO сумісний з будь-якою моделлю MobileNetV2.

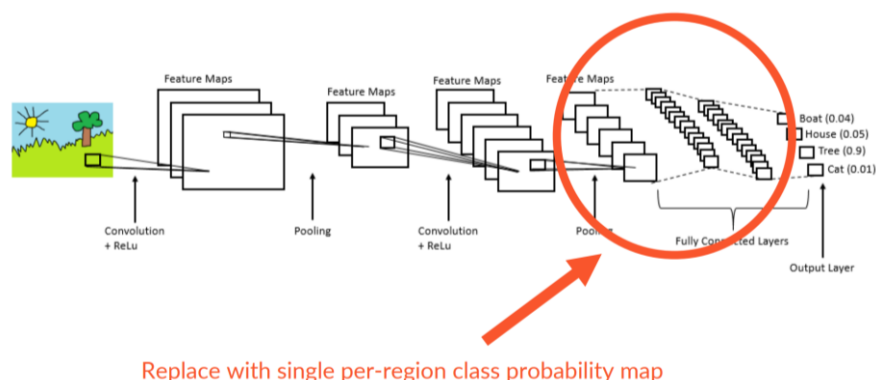


Рисунок 2.5 - Приклад заміни шарів у архітектурі FOMO [45]

2.2 Дослідження обґрунтування вибору мікроконтролера в системах розпізнавання об'єктів.

У сучасних системах розпізнавання об'єктів нейронні мережі активно впроваджуються на рівні мікроконтролерів, що дозволяє створювати ефективні рішення для носимих та вбудованих систем. Вибір мікроконтролера для таких задач залежить від обчислювальних потреб нейронної мережі, можливостей обробки зображень, підтримки сумісних бібліотек та платформ, а також енергоспоживання та вимог до продуктивності, а також вартості мікроконтролера. Під час вибору важливо врахувати, як мікроконтролер справлятиметься з поставленими задачами розпізнавання, для забезпечення оптимальної роботи нейронної мережі в режимі реального часу [46-49]. Також, варто враховувати можливість підключення додаткових датчиків та сенсорів, для створення повноцінної системи з голосовим сповіщення.

На даний момент, для задачі розпізнавання об'єктів основні доступні лінійки моделей таких виробників мікроконтролерів як Espressif, Arduino, Raspberry, а також окремі спеціалізовані для роботи з нейронними мережами одноплатні комп'ютери як Google Coral Dev Board та NVIDIA Jetson Nano.

Espressif

ESP32 – серія багатофункціональних мікроконтролерів компанії Espressif, із вбудованими підключенням Wi-Fi, Bluetooth, для виконання широкого спектру задач, включно з задачами розпізнавання об'єктів нейронними мережами [50]. ESP32 здатний надійно функціонувати в промислових умовах з діапазоном робочих температур від -40°C до $+125^{\circ}\text{C}$. Завдяки розширеним схемам калібрування ESP32 може динамічно усувати дефекти зовнішніх схем і адаптуватися до змін зовнішніх умов. Розроблений для мобільних пристроїв, переносної електроніки та додатків Інтернету речей, ESP32 забезпечує наднизьке енергоспоживання за допомогою комбінації кількох типів власного програмного забезпечення. ESP32 також включає в себе найсучасніші функції, такі як точне синхронізація, різні режими живлення та динамічне масштабування потужності. Може працювати як повноцінна автономна система або як підлеглий пристрій

для головного MCU, зменшуючи накладні витрати стека зв'язку на головному прикладному процесорі. ESP32 може взаємодіяти з іншими системами для забезпечення функцій Wi-Fi і Bluetooth через інтерфейси SPI / SDIO або I2C / UART. Для систем розпізнавання об'єктів є модуль з інтегрованою камерою ESP32-S3-EYE.

ESP32-S3-EYE — це модуль розроблений для завдань комп'ютерного зору та розпізнавання голосу на базі мікроконтролера ESP32-S3. Він має 2-мегапіксельну камеру, РК-дисплей і мікрофон, які використовуються для розпізнавання зображень і обробки звуку рисунок 2.6 [51]. ESP32-S3-EYE пропонує великий обсяг пам'яті з 8 МБ Octal PSRAM і 8 МБ флеш-пам'яті. Також підтримує передачу зображень через Wi-Fi і налагодження через порт Micro-USB. Плата ESP32-S3-EYE складається з двох частин: основної плати (ESP32-S3-EYE-MB), яка об'єднує модуль ESP32-S3-WROOM-1, камеру, слот для SD-карти, цифровий мікрофон, USB-порт і функцію кнопки; і додаткову плату (ESP32-S3-EYE-SUB), яка містить РК-дисплей. Основна плата та допоміжна плата з'єднані через контактні роз'єми.



Рисунок 2.6 – Зовнішній вигляд ESP32-S3-EYE [51]

Камера OV2640 з 2 мільйонами пікселів має кут огляду $66,5^\circ$ і максимальну роздільну здатність 1600×1200 , а також можливість змінити роздільну здатність під час розробки програм. На платі є шість функціональних кнопок, які можна назначити на будь-які функції, крім кнопки RST. На додаток до інтегрованої 8

МБ Octal SPI PSRAM, запропонованої SoC, модуль також постачається з 8 МБ флеш-пам'яті, що забезпечує швидкий доступ до даних. Також підтримується модуль ESP32-S3-WROOM-1U. Цифровий мікрофон I2S MEMS має SNR 61 дБ і чутливість -26 dBFS, працює при напрузі 3,3 В.

Arduino

Arduino — це серія мікроконтролерних плат і екосистема з відкритим вихідним кодом, розроблена для створення електронних проектів. Arduino пропонує різні плати, адаптовані до різних потреб, від початкових проектів до складних застосувань в IoT, робототехніці та комп'ютерному зору.

Для задач з розпізнавання об'єктів найкраще підходять такі плати компанії як, Arduino Nicla Vision, Arduino Portenta H7 та Arduino Nano 33 BLE Sense Rev2.

Nicla Vision поєднує потужний процесор STM32H747AIІ6 Dual ARM® Cortex® M7/M4 IC з 2-мегапксельною кольоровою CMOS камерою, яка підтримує TinyML має кут огляду 80° , розумний 6-осьовий датчик руху, вбудований мікрофон і датчик відстані, підтримує MicroPython, а також пропонує підключення Wi-Fi і Bluetooth® Low Energy рис 2.7 [49]. 2 МБ флеш-пам'яті / 1 МБ ОЗП, 16 МБ QSPI Flash для зберігання, робочу температуру від -20°C до $+70^\circ\text{C}$, та процесор STM32H747AIІ6 Dual Arm® Cortex® M7/M4 IC.

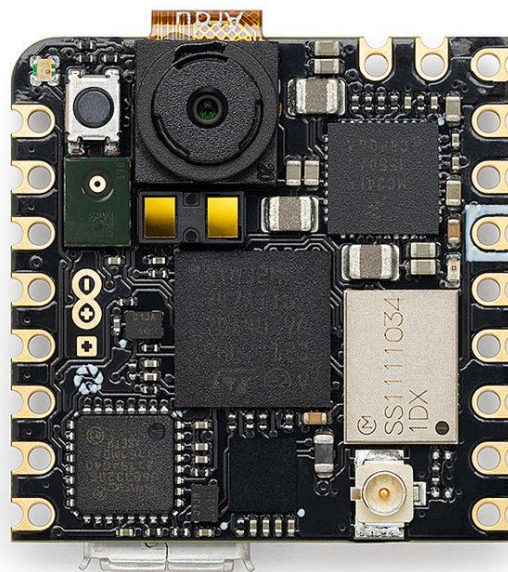


Рисунок 2.7 – Зовнішній вигляд Nicla Vision [52]

Portenta H7 модуль з двоядерним процесором (ARM Cortex-M7 і Cortex-M4) і підтримкою TensorFlow Lite, що дозволяє виконувати базові задачі з розпізнавання об'єктів [53]. Плата одночасно виконує код високого рівня разом із завданнями в реальному часі. Конструкція включає два процесори, які можуть виконувати завдання паралельно. Основним є двоядерний STM32H747, включаючи Cortex® M7, що працює на частоті 480 МГц, і Cortex® M4, що працює на частоті 240 МГц. Два ядра взаємодіють за допомогою механізму Remote Procedure Call, який дозволяє безперешкодно викликати функції іншого процесора. Наприклад, можна виконувати скомпільований код Arduino разом із кодом MicroPython і мати обидва ядра для зв'язку одне з одним. Інтерфейс Bluetooth® підтримує Bluetooth Classic і Bluetooth® Low Energy. Також можна підключити серію різних дротових інтерфейсів, таких як UART, SPI або I2C, як через деякі з роз'ємів у стилі MKR, так і через нову пару промислових 80-контактних роз'ємів Arduino. Пара 80-контактних роз'ємів забезпечує додаткові функції, включаючи Ethernet. Має робочу температуру від -40 °C до +85 °C, 8 MB SDRAM та 16 MB QSPI Flash. Ціна 99 євро.

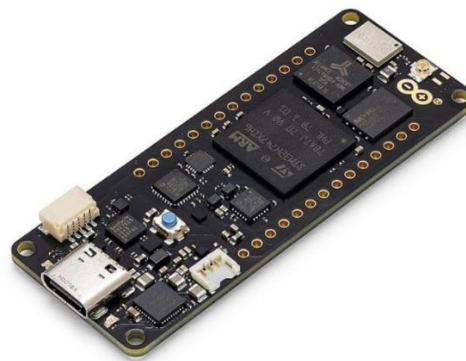


Рисунок 2.8 – Зовнішній вигляд Portenta H7 [53]

Arduino Nano 33 BLE Sense Rev2 — це плата Arduino з підтримкою штучного інтелекту на 3,3 В у найменшому доступному форм-факторі з набором датчиків [54]. Вона побудована на базі мікроконтролера nRF52840 від Nordic Semiconductor, який працює на частоті 64 МГц, з 256 КБ RAM та 1 МБ флеш-пам'яті рис 2.9. Arduino Nano 33 BLE Sense оснащена різними сенсорами для вимірювання навколишнього середовища, включно з акселерометром,

гіроскопом, магнітометром, барометром, мікрофоном і сенсором температури та вологості. Вона підтримує Bluetooth Low Energy і сумісна з TensorFlow Lite для використання нейронних мереж, а також є можливість запуску на ній додатків Edge Computing (AI) за допомогою TinyML. Ціна плати 38 євро.

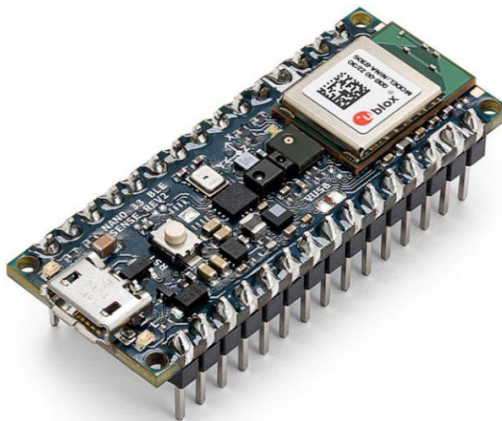


Рисунок 2.9 - Зовнішній вигляд Arduino Nano 33 BLE Sense Rev2 [55]

Raspberry Pi 4 — це потужний одноплатний комп'ютер, розроблений для проєктів, які потребують більшої продуктивності, ніж попередні моделі Raspberry Pi рис 2.10 [56]. Його процесор — чотириядерний ARM Cortex-A72, що працює на частоті 1.5 ГГц, забезпечує достатню обчислювальну потужність для багатьох задач, включаючи обробку зображень, відео та запуск нейронних мереж. В залежності від версії, Raspberry Pi 4 доступний з 1 ГБ, 2 ГБ, 4 ГБ або 8 ГБ оперативної пам'яті LPDDR4, що дозволяє запускати одночасно більше програм та обробляти більше даних. Плата також оснащена двома портами USB 3.0 та двома портами USB 2.0, що дозволяє підключати різноманітні периферійні пристрої з високою швидкістю передачі даних. Підключення до мережі підтримується за допомогою Gigabit Ethernet, а також вбудованого модуля Wi-Fi (802.11ac) та Bluetooth 5.0. Живлення пристрою забезпечується через порт USB-C з потужністю 5 В / 3 А.



Рисунок 2.10 - Зовнішній вигляд Raspberry Pi 4 [56]

В основі Raspberry Pi 5 лежить 64-розрядний чотирьохядерний процесор Arm Cortex-A76, що працює на частоті 2,4 ГГц, що забезпечує вражаюче 2-3-кратне збільшення продуктивності ЦП порівняно з Raspberry Pi 4. Цей процесор підтримується Broadcom BCM2712 SoC і новим процесором. Система пам'яті LPDDR4X, доступна в конфігураціях 2 ГБ, 4 ГБ або 8 ГБ RAM. Пам'ять LPDDR4X забезпечує підвищену енергоефективність, а процесор також включає спеціальні криптографічні розширення, що покращує його придатність для завдань безпеки та шифрування.

Графічна продуктивність Raspberry Pi 5 була значно покращена завдяки графічному процесору VideoCore VII 800 МГц, який підтримує Vulkan 1.2 і подвійні виходи дисплея 4Кр60 HDMI. Розширені можливості вводу/виводу включають подвійні порти USB 3.0, подвоєну пропускну здатність USB і додавання підтримки PCIe 2.0, що забезпечує підключення до периферійних пристроїв з високою пропускну здатністю, таких як SSD, через додатковий M.2 HAT або AI HAT. Крім того, Raspberry Pi 5 має подвійні чотирисмугові порти MIPI для використання до двох модулів камери або дисплея, зовнішній вигляд претсавлено на рисунку 2.11 [57].

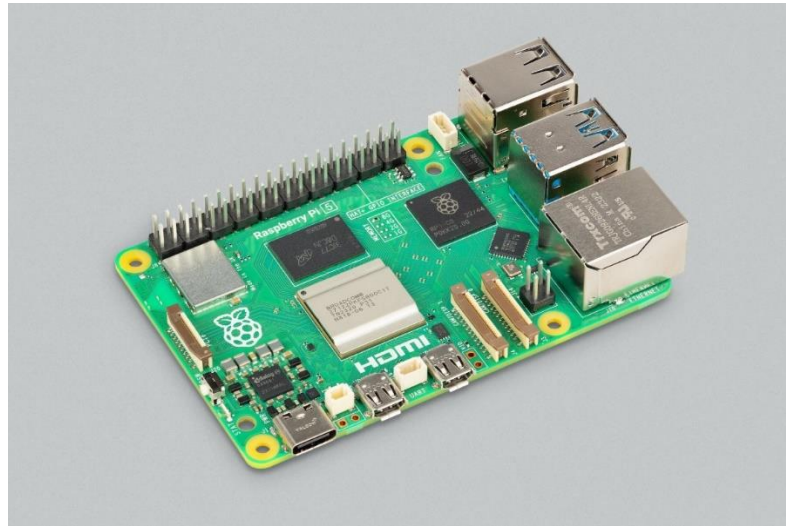


Рисунок 2.11 - Зовнішній вигляд Raspberry Pi 5 [57]

Google Coral Dev Board — це потужний одноплатний комп'ютер (SBC), розроблений для програм високошвидкісного машинного навчання (ML), особливо на периферії рис 2.12 [58]. Центральним елементом його можливостей є співпроцесор Google Edge TPU, який забезпечує швидкий висновок із швидкістю 4 трильйони операцій за секунду (TOPS) із енергоефективністю 2 TOPS на ват. Плата підтримує моделі TensorFlow Lite, які можна скомпільувати для ефективної роботи на Edge TPU, що робить її доступною для створення прототипів програм ML без масштабного перепроектування моделі.

Плата Dev Board оснащена чотирьохядерним процесором NXP i.MX 8M, вбудованим графічним процесором GC7000 Lite, 1 ГБ або 4 ГБ оперативної пам'яті LPDDR4 і 8 ГБ пам'яті eMMC, яку можна розширити за допомогою слота MicroSD. Для підключення він включає Wi-Fi (2x2 MIMO), Bluetooth 4.1, USB Type-C і Gigabit Ethernet. Він також підтримує аудіо (роз'єм 3,5 мм і роз'єми для стереодинаміків), вихід HDMI і вхід для камери через роз'єм MIPI-CSI2. Coral Dev Board працює на Mendel Linux, полегшеній ОС на основі Debian, оптимізований для запуску робочих навантажень ML безпосередньо на пристрої, що забезпечує бездоганну інтеграцію стандартних інструментів Linux і сумісність з існуючими робочими процесами розробки.

Модульна конструкція Coral Dev Board, яка відокремлює System-on-Module (SoM) від базової плати, забезпечує масштабоване розгортання від

прототипування до готових до виробництва додатків. Сам SoM можна видалити та інтегрувати в спеціальне обладнання для більш широкого розгортання ML. Плата також сумісна з різними периферійними пристроями, включаючи дисплеї та додаткові датчики, що робить її надійною платформою для завдань периферійного машинного навчання в різноманітних середовищах.



Рисунок 2.12 - Зовнішній вигляд Google Coral Dev Board [58]

NVIDIA Jetson Nano — це компактна потужна обчислювальна платформа штучного інтелекту, розроблена спеціально для вбудованих додатків ШІ [596]. NVIDIA Jetson Nano оснащений 128-ядерним графічним процесором Maxwell, чотирьохядерним процесором ARM Cortex-A57 і 4 ГБ пам'яті LPDDR4, що забезпечує високу продуктивність для таких завдань, як розпізнавання зображень, виявлення об'єктів і обробка мовлення в установках з низьким енергоспоживанням рис 2.13. Jetson Nano має високу енергоефективність, споживаючи лише 5 Вт, що робить його придатним для проектів штучного інтелекту, що живляться від батарейок, і для портативних проектів.

Однією з ключових особливостей Jetson Nano є підтримка NVIDIA JetPack SDK, який включає CUDA, cuDNN і TensorRT — основні інструменти для глибокого навчання та комп'ютерного зору. Цей SDK надає розробникам доступ до різноманітних попередньо навчених моделей і спрощує розгортання власних нейронних мереж. Пристрій має широкі можливості підключення, включаючи виходи HDMI і DisplayPort, чотири порти USB 3.0 і підтримку Gigabit Ethernet. Крім того, він має подвійні роз'єми камери MIPI CSI, що робить його ідеальним

для проектів, які вимагають вхідного відео з високою роздільною здатністю, наприклад для відеоспостереження в реальному часі та робототехніки.



Рисунок 2.13 - Зовнішній вигляд NVIDIA Jetson Nano [60]

2.3 Вибір платформи для навчання і роботи з нейронними мережами.

З розвитком технологій штучного інтелекту (ШІ) з'являється все більше можливостей для використання нейронних мереж у пристроях з обмеженими ресурсами, таких як мікроконтролери. Платформи для навчання нейронних мереж на мікроконтролерах забезпечують ефективні методи створення, оптимізації та розгортання моделей на малопотужних пристроях. Популярні платформи, як TensorFlow Lite, Edge Impulse, PyTorch Mobile, а також спеціалізовані SDK, наприклад, для NVIDIA Jetson, дозволяють адаптувати нейронні мережі для виконання завдань розпізнавання об'єктів, обробки звуку та аналізу зображень на мікроконтролерах.

Edge Impulse — це потужна платформа для створення моделей машинного навчання (ML) безпосередньо на периферійних пристроях, що робить її чудовим інструментом для завдань розпізнавання об'єктів у реальних програмах [61]. Це особливо підходить для пристроїв з обмеженими обчислювальними ресурсами, таких як мікроконтролери та одноплатні комп'ютери. Інтерфейс Edge Impulse спрощує процес навчання, розгортання та тестування моделей на різних пристроях, включаючи ESP32, Arduino Nicla Vision і Raspberry Pi, шляхом автоматизації більшої частини конвеєра ML, від збору даних до оцінки моделі.

Для виявлення об'єктів Edge Impulse використовує такі моделі, як MobileNet та FOMO (швидші об'єкти, більше об'єктів), які оптимізовані для пристроїв із малою пам'яттю та обчислювальною потужністю. Вебінтерфейс Edge Impulse підтримує інтерактивне середовище, що полегшує візуалізацію результатів, налаштування параметрів навчання та швидке тестування моделей на реальних пристроях. Платформа пропонує простий спосіб збору й обробки даних із сенсорів, що підключені до мікроконтролерів, підтримує різноманітні типи даних, включаючи аудіо, зображення та показники з акселерометрів та інших сенсорів, що особливо важливо для навчання моделей розпізнавання об'єктів, аналізу звуків і розпізнавання рухів.

Edge Impulse має інструменти для підготовки та оптимізації моделей, які працюють безпосередньо на пристроях IoT, а також автоматично адаптує моделі для ефективної роботи на пристроях з обмеженими ресурсами. Платформа підтримує повний цикл обробки даних: від їхнього збирання та очищення до тренування моделей і експорту моделі та інтегрується з різними апаратними платформами, включаючи Arduino, STM32, Nordic Semiconductor, Espressif та інші. Платформа надає гнучкі налаштування для пристрою, на який буде завантажено нейронну мережу для подальшої задачі розпізнавання рисунків 2.14.

The screenshot shows a web interface titled "Configure your target device and application budget". It is divided into two main sections: "Target device" and "Application budget".

Target device section:

- Target device:** A dropdown menu showing "Espressif ESP-EYE (ESP32 240MHz)".
- Processor family:** A dropdown menu showing "ESP32".
- Clock rate:** A text input field with "240" and a unit selector showing "MHz". Below the input is a "Max" label.
- Custom device name (optional):** An empty text input field.

Application budget section:

- RAM:** A text input field with "4" and a unit selector showing "MB". Below the input is a "Max" label.
- ROM:** A text input field with "4" and a unit selector showing "MB". Below the input is a "Max" label.
- Latency:** A text input field with "100" and a unit selector showing "ms". Below the input is a "Max" label.

At the bottom of the form, there are three buttons: "Reset to default settings" (grey), "Cancel" (blue), and "Save" (blue).

Рисунок 2.14 - Приклад вікна налаштувань для обраного девайсу

Платформа дозволяє створювати моделі для аналізу аудіоданих, таких як розпізнавання голосових команд, детекція шумів та аналіз звукових сигналів. Це корисно для додатків, які реагують на звук, наприклад, у системах безпеки або голосових помічниках. Завдяки підтримці даних із сенсорів, таких як акселерометри та гіроскопи, Edge Impulse може створювати моделі для розпізнавання рухів і жестів.

TensorFlow Lite — це версія TensorFlow, оптимізована для мобільних пристроїв та мікроконтролерів.

Версія LiteRT для мікроконтролерів написана на C++ 17 і вимагає 32-бітної платформи, ретельно протестована з багатьма процесорами на базі архітектури Arm Cortex-M Series і була портована на інші архітектури, включаючи ESP32. Фреймворк доступний як бібліотеки Arduino. LiteRT також може створювати проекти для середовищ розробки, таких як Mbed, має відкритий вихідний код і може бути включений до будь-якого проекту C++ 17.

TensorFlow Lite дозволяє конвертувати стандартні TensorFlow моделі у спеціальний формат .tflite, оптимізований для мобільних і вбудованих пристроїв [62]. Інструмент TFLite Converter також підтримує квантування моделей, що дозволяє зменшити розмір моделі і прискорити її роботу, зокрема зменшення точності (від 32-бітного представлення до 8-бітного). TensorFlow Lite забезпечує низький рівень затримки при виконанні моделей, завдяки чому він підходить для додатків реального часу, таких як розпізнавання об'єктів, обробка звуку та аналіз жестів. Платформа підтримує спеціалізовані апаратні прискорювачі, такі як Edge TPU від Google, що дозволяє виконувати складні нейронні мережі швидше та з меншим енергоспоживанням на пристроях на базі Google Coral. Крім того, підтримуються графічні прискорювачі на Android та iOS, що використовують NNAPI та Core ML для підвищення продуктивності. Інтерпретатор TFLite дозволяє використовувати моделі безпосередньо на пристрої, не потребуючи підключення до хмарного сервера, що підвищує приватність та швидкість виконання. Для класифікації та розпізнавання об'єктів підтримуються моделі на

базі MobileNet SSD та YOLO, MobileNet (V1/V2), EfficientNet Lite, Inception та NasNet.

PyTorch Mobile — це версія фреймворку PyTorch, спеціально адаптована для розгортання моделей на мобільних пристроях та пристроях з обмеженими обчислювальними можливостями, таких як Raspberry Pi або NVIDIA Jetson. Фреймворк підтримує зменшені версії популярних моделей, які можна адаптувати для реального часу та низької затримки в додатках, таких як розпізнавання об'єктів і класифікація зображень [63]. PyTorch Mobile включає інструменти для оптимізації моделей, такі як квантування та вирізання шарів (pruning), що дозволяє зменшити розмір моделей і знизити потреби у пам'яті та енергоспоживанні. PyTorch Mobile легко інтегрується з основним фреймворком PyTorch, що дозволяє створювати і тренувати модель у PyTorch і згодом конвертувати її у формат, придатний для мобільного застосування. Процес конвертації підтримує автоматичне виконання на CPU або GPU, що дає можливість ефективніше використовувати апаратні ресурси пристрою. PyTorch Mobile підтримує основні мобільні платформи, такі як iOS і Android. За допомогою спеціальних API розробники можуть інтегрувати нейронні мережі безпосередньо у додатки, дозволяючи виконувати задачі ШІ без підключення до хмари. PyTorch Mobile підтримує різні моделі для задач виявлення об'єктів, включаючи SSD, YOLO та MobileNet SSD, а також MobileNet, ResNet, EfficientNet та NasNet.

2.4 Вибір програмного забезпечення для генерації тексту у мову.

Для озвучення розпізнаного об'єкту і надання інформації людям з вадами зору, необхідний генератор тексту у мову, з підтримкою Української мови, можливістю завантаження на мікроконтролер, та низьким споживанням пам'яті для роботи. На даний момент для застосування на мікроконтролері доступні такі як : eSpeak NG, Voxxygen TTS, Google TTS.

eSpeak NG (Next Generation) — це відкритий синтезатор мовлення, що підтримує понад 100 мов, включаючи українську [64]. Його можна використовувати на мікроконтролерах з достатніми обчислювальними

ресурсами, таких як ESP32 або Raspberry Pi. eSpeak NG використовує форму синтезу, що називається формантним синтезом, яка генерує синтезоване мовлення за допомогою параметричних моделей звуків мови. Це дозволяє створювати мовлення досить природного звучання навіть на пристроях з обмеженими ресурсами, а також має високу швидкість генерації мовлення та невеликі вимоги до пам'яті. Використання командного рядка або інтеграція з програмним забезпеченням здійснюється через доступні бібліотеки, такі як eSpeakService.

Voxugen TTS — це комерційна платформа синтезу мовлення, що пропонує інтеграцію з мікроконтролерами для генерації голосового виходу. Цей синтезатор підтримує українську мову і здатний генерувати природніший голос завдяки алгоритмам побудови тексту на базі нейронних мереж [65]. Вимоги до ресурсів залежать від вибраного рівня якості, але Voxugen може бути інтегрований у пристрої на базі мікропроцесорів. Для доступу до Voxugen TTS на мікроконтролерах може знадобитися підтримка додаткових зовнішніх сервісів або зв'язок із сервером через API для зниження навантаження на процесор. Можливості локального розгортання обмежені.

Google TTS підтримує українську мову і доступний через API [66]. Хоча Google TTS вимагає підключення до інтернету, його можна використовувати на пристроях з Wi-Fi або мобільним зв'язком, наприклад, ESP32 з модулем Wi-Fi або Raspberry Pi. Сервіс перетворює текст на українській мові у високоякісне мовлення, доступне для прослуховування на самому пристрої.

Висновки до розділу 2

У результаті аналізу доступних складових системи розпізнавання об'єктів з подальшим голосовим виводом інформації для людей з вадами зору та повною сліпотою встановлено, що оптимальним нейронними мережами для розгортання на пристроях з обмеженими ресурсами є клас мереж MobileNet. Основними перевагами є можливість стиснення розмірів моделі до необхідного, завдяки коефіцієнтам ширини мережі і зміні розміру вхідного зображення, без значних втрат у швидкості і точності розпізнавання. У якості платформи для навчання та

експортування нейронних мереж під обрані мікроконтролери, враховуючи функціонал, постійний розвиток, а також зручність у використанні обрано платформу Edge Impulse.

У якості програмного забезпечення для голосового сповіщення обрано eSpeak NG, завдяки можливості автономної роботи без необхідності підключення до мережі, підтримці Української мови, гнучким налаштуванням параметрів генерації мови, та невеликих потребах у пам'яті.

Встановлено, що мікроконтролери та одноплатні комп'ютери є оптимальним рішенням для створення необхідних систем. Вони мають невеликий форм фактор та вагу, а також мають достатньо потужності для виконання задач з розпізнавання об'єктів режимі реального часу. Для опису варіацій систем розпізнавання об'єктів з подальшим голосовим виводом для людей з вадами зору обрано такі плати як: ESP32-S3-EYE , Raspberry Pi 5 та NVIDIA Jetson Nano. Плати мають різні технічні характеристики і можуть забезпечити різний рівень швидкості розпізнавання об'єктів, і загальну продуктивність системи. Варто врахувати, що для більш потужних систем необхідне потужніше джерело живлення та корпус із системою охолодження.

РОЗДІЛ 3. СИСТЕМА РОЗПІЗНАВАННЯ ОБ'ЄКТІВ З ГОЛОСОВИМИ ПІДКАЗКАМИ ДЛЯ ЛЮДЕЙ З ВАДАМИ ЗОРУ ТА ПОВНОЮ СЛІПОТОЮ

3.1 Структура та принцип роботи запропонованої мною системи.

Запропонована система складається з мікроконтролера, або одноплатного комп'ютера, джерела живлення, джерела світла, динаміків та підсилювача для динаміків. Інформація з камери надходить на нейронну мережу MobileNet, що попередньо завантажено на мікроконтролер, нейронна мережа виконує розпізнавання об'єкту на зображенні з камери, результат розпізнавання у вигляді назви об'єкту передається на генератор тексту у мову eSpeak NG, після чого формується голосовий сигнал та виконується сповіщення користувача через динаміки про об'єкт який знаходиться попереду нього. Узагальнена схема проходження процесу розпізнавання-оголошення представлена на рисунку 3.1.

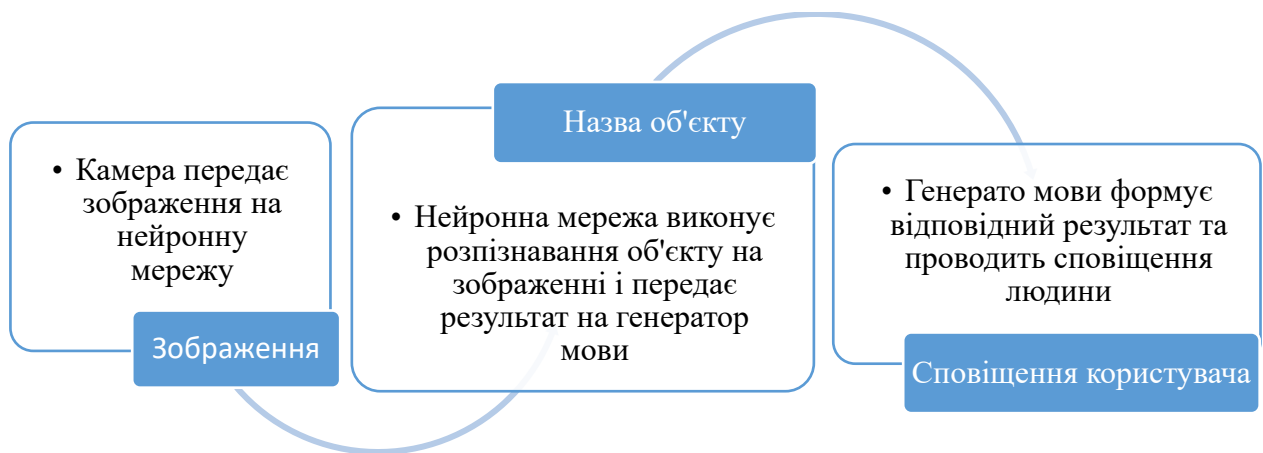


Рисунок 3.1 – Узагальнена схема роботи запропонованої системи

3.2 Актуальність модифікації систем розпізнавання об'єктів для людей з вадами зору.

Сучасні технології комп'ютерного зору та нейронних мереж відкривають нові можливості для поліпшення якості життя людей з порушеннями зору. Розробка систем розпізнавання об'єктів для цієї категорії користувачів дозволяє частково компенсувати відсутність зорового сприйняття, покращити орієнтацію у просторі та взаємодію з навколишнім середовищем. За даними Всесвітньої

організації охорони здоров'я (ВООЗ), у світі близько 2,2 мільярда людей мають проблеми з зором. Із них приблизно 43 мільйонів є повністю незрячими, тобто мають повну втрату зору. Ще близько 295 мільйонів людей страждають від різного ступеня часткової втрати зору, що значно обмежує їхню здатність бачити навколишнє середовище, навіть із корекцією [67].

За допомогою мобільних пристроїв або портативних систем, які озвучують ідентифіковані об'єкти, люди з вадами зору можуть значно легше переміщатися в міському просторі, орієнтуватися у незнайомих приміщеннях, розпізнавати обличчя або предмети, які використовуються у повсякденному житті. Такі системи можуть бути налаштовані під конкретні потреби користувача, що підвищує їхню ефективність та адаптивність. Вони сприяють створенню більш інклюзивного середовища та дозволяють користувачам з обмеженим зором вести незалежніший та активніший спосіб життя.

В залежності від ступеню втрати зору, існують дві основних групи людей з вадами зору, люди з повною сліпотою, або з частковою втратою зору.

Люди з повною втратою зору не мають здатності до зорового сприйняття. Їм недоступне сприйняття світла, форми або кольору. У цьому випадку тактильні, слухові та нюхові відчуття є основними способами взаємодії з навколишнім середовищем. Для таких людей системи розпізнавання об'єктів із голосовим озвученням є надзвичайно корисними, оскільки вони можуть оперативно отримувати інформацію про предмети навколо та діяти на основі отриманої заздалегідь інформації.

Часткова втрата зору (залишковий зір): до цієї категорії належать люди, які здатні частково бачити світло, кольори або форми, проте їхнє зорове сприйняття значно обмежене. Залишковий зір може включати бачення плям або розмитих силуетів, що дозволяє лише частково орієнтуватися у просторі. Візуальні системи розпізнавання об'єктів із додатковою функцією підсилення контрастності або кольору можуть допомогти таким людям краще розпізнавати об'єкти та контури. У цьому випадку застосування комбінованих методів,

зокрема голосового озвучення та зорових підказок, дозволяє їм ефективніше використовувати наявні можливості зору.

Причини, через які люди можуть втратити зір є очні захворювання, такі як: катаракта, глаукома, макулярна дегенерація, а також системні захворювання, такі як: діабетична ринотерапія та судинні захворювання, генетичні та вродженні захворювання або травми, та інфекційні захворювання.

Зараз, орієнтування у просторі для людей з повною втратою зору базується на використанні слуху, дотику, нюху, пам'яті, а також спеціальних пристосувань та технологій. Слух є основним джерелом інформації для людей із повною сліпотою. Вони вміють сприймати зміни відлуння від звуків, що допомагає визначати відстань до об'єктів. Наприклад, можуть помітити наближення до стіни або перешкоди через зміну звучання кроків або шуму навколо. Розпізнавання звуків, що видають об'єкти, наприклад, шурхіт дерев, шум води або сигналізації, дозволяє їм будувати своєрідну «картину» простору за звуками.

Одним з головних пристроїв, що допомагає орієнтуватися у просторі є тростина рис 3.2. Людина використовує тростину як продовження руки для виявлення предметів перед собою та отримання інформації про структуру середовища, предмети на шляху, зміну поверхні під ногами (асфальт, трава, сходи тощо) та розташування меж дороги або тротуару.



Рисунок 3.2 – Тростина для незрячих людей [68]

Тактильна плитка та тактильні схеми, які можуть бути у деяких громадських місцях допомагають ознайомитися зі структурою приміщення, обрати маршрут та надати попередження про небезпеку. Також, існують собаки-поводирі, навчені допомагати людині орієнтуватися у просторі, уникати перешкод, слідувати певним маршрутам.

Один з основних способів переміщення повністю незрячих людей це запам'ятовування маршрутів з якими доводиться часто стикатися. Наприклад, дорога від дому до магазину чи роботи запам'ятовується за допомогою багатьох деталей, таких як кількість кроків, поворотів, особливих тактильних ознак чи предметів розташованих на маршрутах, специфічних звуках чи запахах, а також за зміною поверхні під ногами.

Основні проблеми які виникають під час пересування пов'язані зі зміною орієнтирів у навколишньому середовищі, технічними роботами, через які звичний маршрут може бути недоступний, погодними умовами, сніг, наприклад може приховати специфічні орієнтири на маршрутах і тд.

У загальному випадку система розпізнавання об'єктів з голосовими підказками для людей з вадами зору або повною сліпотою функціонує наступним чином – камера передає отримане зображення на мікроконтролер на який завантажено перередньо навчану на розпізнавання різних об'єктів нейронну мережу. Розпізнавання об'єктів відбувається у режимі реального часу, після чого, результат розпізнавання надходить на генератор мови, і відбувається сповіщення людини.

Після проходження повного циклу розпізнавання-оголошення інформації, процес повторюється. Один цикл може займати різний час в залежності від часу необхідного на проходження його різних етапів.

3.3 Оцінка часу необхідного на проходження одного повного циклу розпізнавання-оголошення інформації.

Для оцінки часу необхідного для проходження циклу розпізнавання-оголошення інформації спершу необхідно встановити основні складові цього

процесу, а також їх особливості. У загальному випадку цей процес складається з таких складових :

- Розпізнавання об'єкту нейроною мережею
- Передача розпізнаної інформації до генератора мови
- Генерація і оголошення назви об'єкту
- Отримання інформації людиною і реакція на неї.

Кожна складова процесу залежить від певних факторів, які впливають на час проходження кожного етапу.

Швидкість та точність розпізнавання об'єктів нейроною мережею залежить від:

- Якості тренувальних даних для нейроної мережі
- Обраної нейроної мережі
- Апаратних можливостей мікроконтролера
- Якості освітлення та погодних умов

Час необхідний на передачу, генерацію і оголошення результату залежить від :

- Кількості букв і слів у назві об'єкту
- Швидкості генерації мовлення
- Паузи між словами

В залежності від кількості букв та слів у назві об'єкту буде змінюватися і час необхідний на його оголошення. Швидкість генерації мови та час паузи між словами параметри змінні, і можуть бути вказані вручну під час налаштування роботи генератора мови.

Для розрахунку кількості часу необхідного для оголошення назви об'єкту, треба розглянути можливі комбінації назв об'єктів в залежності від кількості слів та їх довжини. Середній темп мовлення в Україні складає 200 слів на хвилину, включаючи паузи між словами, оскільки паузи є природною частиною мовного потоку, середня довжина Українських слів становить 6-8 символів. Паузи необхідні для сприйняття, інтонації та ритму мовлення. В середньому, затримка (пауза) між словами складає 100-200 мс [69]. Якщо виключити паузи, то

швидкість звучання лише слів (без проміжків) буде вищою. Наприклад, замість 200 слів на хвилину (з паузами) це могло б дорівнювати 240–260 слів за хвилину (без пауз).

Можливі комбінації назви об'єкту :

- Назва з одного слова
- Назва з двох слів
- Назва з трьох і більше слів

Для зручності розрахунків швидкість мовлення була переведена з слів на хвилину до символів на секунду з урахуванням особливостей української мови.

Швидкість мовлення 200 слів на хвилину означає, що одне слово займає:

$$\frac{60 \text{ секунд}}{200 \text{ слів}} = 0.3 \text{ секунди (або 300мс)}$$

Тривалість включає: час вимови слова - приблизно 200 мс, час паузи між словами - приблизно 100 мс.

При швидкості 200 слів на хвилину, кількість слів на секунду буде дорівнювати:

$$\frac{200 \text{ слів}}{60 \text{ секунд}} \approx 3.3 \text{ слова на секунду}$$

При середній довжині слова в 7 символів, отримуємо швидкість близько 23,1 символа на секунду.

Тривалість, за яку оголошується один символ, обчислюється як:

$$\frac{1 \text{ секунда}}{23,1 \text{ символа}} \approx 0.043 \text{ секунди (або 43 мс)}$$

Отже, середній час необхідний на оголошення одного символу дорівнює 43 мс. Враховуючи цю інформацію, розраховано час необхідний на оголошення слів різної довжини. У випадку коли назва об'єкту складається з одного слова, воно може бути коротким, середньостатистичним(середнім) або довгим. Коротким будемо вважати слово довжиною у 3-5 символи, середнє з довжиною у 6-8, та довге з довжиною у 9-12 символів. У ситуації коли назва об'єкту

складається з двох слів - це можуть бути комбінації з двох коротких слів (к-к), короткого та середнього (к-с), короткого та довгого (к-д), двох середніх (с-с), середнього та довгого (с-д), або двох довгих (д-д).

У таблиці 3.1 представлено мінімальний, максимальний та середній час необхідний на оголошення назви об'єкту, що складається з одного слова. Також, при розрахунку назв, які складаються з двох і більше слів, слід враховувати паузу між словами, що в середньому становить 200 мс для української мови.

Таблиця 3.1 - Час необхідний на оголошення одного слова різної довжини.

Назва об'єкту	Мінімальний час, мс	Максимальний час, мс	Середній час, мс
Коротка	129	215	172
Середня	258	344	301
Довга	387	516	452

У таблиці 3.2 наведено мінімальний, максимальний та середній час необхідний на оголошення назви об'єкту, яка складається з двох слів та їх можливі комбінації. Час вказаний з врахуванням паузи між словами у 200 мс.

Таблиця 3.2 - Час необхідний на оголошення двох слів різної довжини.

Комбінація слів	Мінімальний час, мс	Максимальний час, мс	Середній час, мс
К-К	458	630	544
К-С	587	759	673
К-Д	716	931	823
С-С	716	888	802
С-Д	845	1060	953
Д-Д	974	1232	1103

Також, було виконано розріхунок для назв, що які складаються з трьох слів, враховуючи паузи між словами (по 200 мс) та час на озвучення кожного слова залежно від його довжини. Результати представлено у таблиці 3.3. Швидкість реакції людини на голосову інформацію зазвичай становить приблизно 200–400 мілісекунд. Це середній час, за який мозок обробляє голосове повідомлення та реагує на нього. Реакція на прості голосові команди, наприклад "стоп" або "йди", може бути швидшою, оскільки вони легко інтерпретуються і вимагають меншого обсягу обробки. Більш складні інструкції можуть потребувати більше часу.

Таблиця 3.3 - Час необхідний на оголошення трьох слів різної довжини.

Комбінація слів	Мінімальний час, мс	Максимальний час, мс	Середній час, мс
К-К-К	787	1045	916
К-К-С	916	1174	1045
К-К-Д	1045	1346	1196
К-С-С	1045	1303	1174
К-С-Д	1174	1475	1324
К-Д-Д	1303	1647	1475
С-С-С	1174	1432	1303
С-С-Д	1303	1604	1454
Д-Д-Д	1561	1948	1755

Люди з повною сліпотою або значними порушеннями зору можуть мати дещо вищу швидкість реакції на голосові команди порівняно з тими, хто має нормальний зір. Це пояснюється тим, що у людей з вадами зору слухові здібності часто більш розвинені через підвищену залежність від слуху для отримання інформації. Така адаптація покращує сприйняття звуків і дозволяє більш швидко реагувати на звукові сигнали. У людей із вадами зору можуть бути кращі показники в розрізненні швидких змін у звуках або їх напрямку, що може прискорювати реакцію на голосову інформацію. Тим не менше, різниця в часі реакції на звуки зазвичай не є дуже значною, але вона може відігравати важливу роль у ситуаціях, що потребують швидкого прийняття рішень, як-от орієнтування у просторі або уникнення перешкод.

Середня швидкість руху у людей з вадами зору становить приблизно 0,8–1,0 метр за секунду, що трохи менше від середньої швидкості руху людей із нормальним зором (близько 1,4 метра за секунду). Проте швидкість руху може суттєво змінюватись залежно від факторів, таких як умови навколишнього середовища, рівень знайомства з маршрутом, рівень залишкового зору, а також допоміжні засоби (наприклад, тростина або собака-поводир). Люди, які користуються тростиною, зазвичай рухаються повільніше, адже їм потрібно сканувати простір перед собою для визначення перешкод. Темрява чи сутінки можуть значно уповільнити рух, особливо для людей із залишковим зором. Дощ чи сніг також уповільнюють швидкість, адже слизька поверхня або накопичений сніг ускладнюють пересування, підвищуючи ризик падіння.

Отже, загальний час ($t_{\text{ц}}$) необхідний на один повний цикл процесу розпізнавання-оголошення назви розпізнаного об'єкту можна розрахувати за формулою:

$$t_{\text{ц}} = t_{\text{роз}} + t_{\text{назви}} + t_{\text{реакції}}$$

3.4 Попередня обробка і групування тренувальних даних для підвищення точності розпізнавання об'єктів у складних ситуаціях.

У сучасних системах комп'ютерного зору, що використовують нейронні мережі, якість розпізнавання об'єктів значною мірою залежить від тренувальних даних. Особливо це стосується об'єктів, що мають схожу сторонню інформацію на зображеннях, наприклад, небо, дерева чи інші фонові сцени, які можуть створювати помилкові розпізнавання моделлю. Ефективна попередня обробка тренувальних даних є ключовим етапом, який дозволяє мінімізувати ці проблеми та підвищити точність розпізнавання.

Попередня обробка допомагає нейронній мережі краще інтерпретувати візуальні ознаки об'єктів, ізолюючи їх від фону або інших небажаних елементів. Для зображень, де об'єкти розташовані на фоні, що часто повторюється, як-от небо, важливо уникнути ситуації, коли модель починає асоціювати об'єкт не з його характерними ознаками, а з властивостями фону. Це може призвести до помилкових класифікацій, наприклад, коли всі об'єкти на фоні неба сприймаються як "птахи" або "літаки".

Одним із основних завдань попередньої обробки є забезпечення різноманітності даних і видалення потенційно зумовлених кореляцій. Використання методів, як-от нормалізація зображень, зменшення шуму чи сегментація об'єктів, дозволяє підвищити стійкість моделі до сторонньої інформації. Це стає особливо актуальним у застосуваннях, де точність і надійність розпізнавання є критично важливими, як-от у системах для допомоги людям з вадами зору.

У випадках, коли одну нейронну мережу необхідно навчити розпізнавати об'єкти, які можуть знаходитися на тренувальних зображеннях поруч один з

одним, правильний набір і групування тренувальних даних має підвищити точність розпізнавання, та підвищити стійкість системи до хибних розпізнавань.

Таким чином, попередня обробка тренувальних даних не лише покращує якість навчання нейронних мереж, але й формує основи для створення адаптивних і надійних систем розпізнавання, які здатні ефективно працювати в умовах високої мінливості середовища.

3.5 Комплектації варіантів систем розпізнавання об'єктів з подальшим голосовим сповіщенням для людей з вадами зору.

Запропонована мною система складається з мікроконтролера (одноплатного комп'ютера), камери, динаміків, джерела світла для додаткового освітлення у темний період доби, елементів живлення, системи охолодження (для більш потужних варіацій, а також додаткової периферії для функціонування системи у різних комплектаціях.

Крім того, необхідно визначитися з форм фактором фінального пристрою, адже він має бути зручний для використання і носіння, мати оптимальну ергономіку, і невелику вагу, щоб не створювати надлишкове навантаження на користувача, а також зручний корпус з елементами охолодження.

Систему описано для трьох різних варіантів мікроконтролерів (одноплатних комп'ютерів), а саме ESP32-S3-EYE , Raspberry Pi 5 та NVIDIA Jetson Nano, з підбором комплектуючих компонентів.

Якщо система використовується в умовах недостатнього освітлення, необхідно підключити джерело світла для забезпечення нормальної роботи камери та розпізнавання об'єктів.

Для цього можна обрати такі варіанти джерел освітлення: LED-модулі з високою яскравістю, SMD світлодіоди, компактні LED-стрічки або окремі модулі.

Для визначення необхідного джерела світла, варто визначитися з ефективною відстанню освітлення, кутом розсіювання, силою світла та його енергоспоживання. Для роботи у запропонованій системі розпізнавання об'єктів, ефективна відстань освітлення (d) має складати 2.5 – 3 м, кут розсіювання має

складати 60-70°, щоб покривати необхідну площу для ефективного розпізнавання об'єктів.

Щоб визначити силу світла (I) для освітлення певної площі, використано формулу [70]:

$$E = \frac{I}{d^2},$$

де E — освітленість у люксах (лк), яку потрібно забезпечити, I — сила світла в канделах (кд), d — відстань від джерела до поверхні освітлення (м).

При певному куті розсіювання (θ) світло розподіляється в межах конуса. Світловий потік, що поширюється на площу (A) освітленої поверхні, пов'язаний із силою світла:

$$A = \pi d^2 \cdot \tan^2\left(\frac{\theta}{2}\right),$$

де A — площа освітлення (м²), d — ефективна відстань освітлення, θ — кут розсіювання.

Площа освітлення буде дорівнювати:

$$A = \pi \cdot 3^2 \cdot (0.637)^2 = \pi \cdot 9 \cdot 0.404 \approx 11.46 \text{ м}^2.$$

Взявши освітленість E = 50 лк, для хорошої видимості в умовах низької освітленості розраховано силу світла I:

$$I = E \cdot d^2$$

$$I = 50 \cdot 3^2 = 50 \cdot 9 = 450 \text{ кд.}$$

Отже, для забезпечення умов, необхідних для роботи системи розпізнавання в умовах низького освітлення, обране джерело світла має відповідати таким вимогам :

- Ефективна відстань освітлення у 3м
- Кутом розсіювання від 60-70 °
- Силою світла від 450 кд
- Потужністю до 6 В
- Можливістю підключення до обраних мікроконтролерів

Світлодіод Cree XP-G3 має нейтральний білий колір з температурою 5000 К, кут світіння 115° з можливістю встановлення лінзи для керування кутом рис 3.2 [71].

Максимальна потужність 6 Вт, максимальний струм – 2000 мА, напругу 2.85-3.6 В, та світловий потік у 777 лм.

Світлодіод Cree XP-G3 є сумісним з усіма трьома платами. Основна сумісність досягається завдяки використанню стандартних інтерфейсів живлення та керування (GPIO, PWM або драйвери через I2C/SPI). ESP32-S3-EYE - може керувати світлодіодами через GPIO або PWM. Cree XP-G3 можна інтегрувати через відповідний драйвер або через підключення до транзисторного каскаду для керування струмом. Це важливо, оскільки світлодіод працює на струмі до 1 А, що перевищує можливості прямого підключення до GPIO ESP32-S3.



Рисунок 3.3 – Зовнішній вигляд світлодіоду Cree XP-G3 [71]

Raspberry Pi 5 - завдяки наявності GPIO і підтримці PWM, може керувати світлодіодом через драйвер, як-от MOSFET-транзистори. Потрібен зовнішній блок живлення для підтримки потужності Cree XP-G3. NVIDIA Jetson Nano - має підтримку GPIO, SPI та I2C, а також сумісність із драйверами, як-от Adafruit Blinka. Це забезпечує просте налаштування світлодіодів і драйверів на основі Python, включаючи Cree XP-G3.

Для виводу голосової інформації необхідно обрати динаміки. Враховуючи компактність системи, оптимальним варіантом будуть невеликі динаміки, потужністю до 10 Вт (для чіткого звуку), а також вони мають бути сумісні з обраними мікроконтролерами. Також, для підключення необхідно обрати підсилювач.

Наприклад, динамік LD-SP-UM20/8A LOUDITY має такі характеристики рис 3.4 [72]:

- АЧХ : 100 – 20000 Гц
- Опір 8 Ом
- Максимальну потужність : 2 Вт
- Рівень звуку 83 дБ
- Загальний зовнішній діаметр 20 мм
- Висоту 5,8 мм
- Резонансну частоту 700 Гц
- Робочу температуру – 20...60 °C

Вартість такого динаміку становить від 215 до 300 гривень.



Рисунок 3.4 - Зовнішній вигляд LD-SP-UM20/8A [72]

ESP32-S3-EYE для роботи з динаміком необхідний зовнішній підсилювач звуку, оскільки вихідна потужність аудіокодека недостатня для прямого підключення. До Raspberry Pi 5 динамік можна підключити через аудіовихід 3.5 мм або за допомогою підсилювача, якщо потрібна більша потужність. NVIDIA Jetson Nano потрібен зовнішній аудіоінтерфейс або підсилювач через відсутність вбудованого аналогового аудіовиходу.

Також, для роботи у умовах підвищеної вологості або дощу, можна обрати динамік з захистом від вологи, наприклад LD-SP-MSI50-53 рис 3.5 [73]. Він має потужність 4 Вт, імпеданс 4 Ом, розміри 50*18,5 мм, клас захисту IP 67, і також сумісний з обраними платами. Вартість такого динаміку від 270 до 350 гривень.



Рисунок 3.5 - Зовнішній вигляд LD-SP-MSI50-53 [73]

Для підключення динаміків до системи необхідно використати підсилювач.

Враховуючи технічні характеристики обраних динаміків і сумісність з необхідними платами, обрано звуковий підсилювач D-класу PAM8403 [74]. Як підсилювач класу D забезпечує мінімальні втрати енергії у порівнянні з підсилювачами класу АВ. Вбудовані функції захисту від короткого замикання та перегріву дозволяють уникнути пошкоджень у разі несправностей в підключених компонентах, вихідну потужність 2*3 Вт та розміри 21x18x3.5 мм.

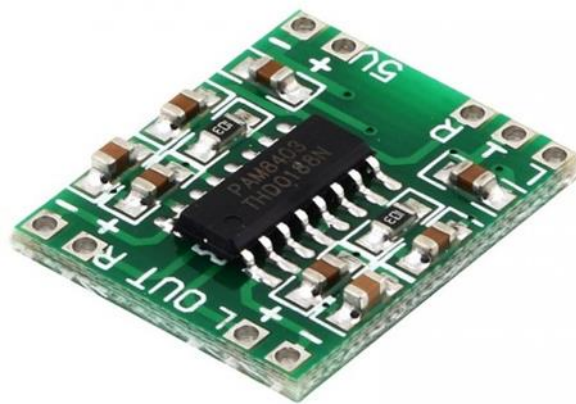


Рисунок 3.6 – Зовнішній вигляд підсилювача PAM8403 [74]

Для функціонування системи необхідно обрати джерело живлення, з врахуванням споживання енергії мікроконтролером у режимі розпізнавання об'єктів у реальному часу, роботи динаміків та підсилювача, а також джерела світла, і при цьому забезпечити автономність функціонування системи у активному режимі на 2-3 години.

Для системи на основі мікроконтролера ESP32-S3-EYE, з власною камерою, джерелом світла - Cree XP-G3, динаміків LD-SP-UM20/8A та

підсилювача РАМ8403 загальну споживану потужність (P_3) розраховано наступним чином:

$$P_3 = P_m + P_{дс} + P_d + P_n,$$

де, P_m – споживана потужність мікроконтролера у режимі роботи розпізнавання об’єктів нейроною мережею, $P_{дс}$ – споживана потужність джерела світла, P_d – споживана потужність динаміків, P_n – споживана потужність підсилювача. Потужність розраховано при максимальних навантаженнях на мікроконтролер у режимі розпізнавання об’єктів з подальшою передачею на генератор мови, і постійною роботою динаміків та підсилювача, для того, щоб обрати оптимальне джерело живлення для максимальних навантажень.

Споживана потужність кожного окремого елемента розраховано за формулою:

$$P = U \cdot I,$$

де U – напруга при максимальному навантаженні, I – струм при максимальному навантаженні.

Для ESP32-S3-EYE максимальна потужність споживання залежить від активованих функцій і периферійних пристроїв. За умови роботи на максимальній частоті (240 МГц), увімкненого Wi-Fi, активного режиму камери, розпізнавання нейронною мережею об’єктів та генерації звуку, загальна споживана потужність буде дорівнювати добутку суми струмів та напруги. Базове споживання ESP32-S3 становить до 120 мА при пікових навантаженнях під час передачі Wi-Fi або інтенсивного процесу обробки, камера та переферія 80-100 мА, генерація мови в залежності від режиму роботи – 50-100 мА, при робочій напрузі у 5В. Таким чином, загальна споживана потужність мікроконтролера ESP32-S3-EYE з переферією буде дорівнювати:

$$P_m = 5 \cdot (0.120 + 0.100 + 0.100) = 1.6 \text{ Вт}$$

Максимальна потужність світлодіоду Cree XP-G3 становить 6 Вт, максимальна потужність динаміку LD-SP-UM20/8A становить 2 Вт, враховуючи, що динаміків 2, максимальна споживана потужність буде 4 Вт, максимальне

споживання при нормальному навантаженні РАМ8403 становить 0.5 Вт. Отже загальна споживана потужність (P_3) системи на базі мікроконтролера ESP32-S3-EYE буде дорівнювати:

$$P_3 = 6 + 4 + 1,6 + 0,5 = 12,5 \text{ Вт}$$

Для автономної роботи системи на 3 години відповідно необхідно джерело живлення з ємністю, що забезпечить 32,5 Вт*год сумарної енергії (W).

Ємність акумулятора (A) розраховано за формулою:

$$A = W/V$$

$$A = 37.5/5 = 7.5 \text{ А*год}$$

Таким чином, необхідне джерело живлення з напругою в 5 В, та ємністю у 7500 мА*год. Одним з найкращих і універсальних рішень буде обрати потужний Power Bank із підтримкою швидкого заряджання. На ринку існує багато варіантів що задовольняють вимоги і мають достатню ємність для автономної роботи системи на 3 години.

Наприклад, УМБ Andowl Q-CD6500 має ємність 68000 мА·год, вихідну напругу 5-12 В, вихідну силу струму 1.5-5 А, 4 порти USB Type A, та 1 порт USB Type C, вагу 210 грм [75]. Вартість становить від 3000 до 3500 тисяч гривень в залежності від магазину.



Рисунок 3.7 - Зовнішній вигляд УМБ Andowl Q-CD6500 [75]

Для системи на основі одноплатного комп'ютера Raspberry Pi 5, з джерелом світла - Cree XP-G3, динаміків LD-SP-UM20/8A та підсилювача РАМ8403 спершу необхідно обрати камеру, адже у комплекті з самою платою її немає.

Для Raspberry Pi 5 оптимальною камерою є Camera Module 3 рис 3.8 [76]. Це офіційна камера для цієї плати. Вона доступна в декількох версіях, зокрема стандартна, ширококутна, NoIR (без інфрачервоного фільтру для нічного бачення), і NoIR Wide. Основні характеристики:

- Сенсор - Sony IMX708, 12 МП, задньопідсвітка CMOS.
- Роздільна здатність: до 4608x2592 пікселів.
- Відео - підтримка до 1080p при 50 кадрах/с.
- Кут огляду: 75° (стандартна версія) або 120° (ширококутна версія).
- Автофокус з підтримкою фазового виявлення.
- HDR-режим з розміром до 3 МП вихідного зображення.

Ця камера сумісна з новим Raspberry Pi Open Source Camera System, що надає доступ до налаштувань через бібліотеку `libcamera`, а також має підтримку Python-бібліотеки `Picamera2` для інтеграції у проекти машинного навчання або робототехніки.

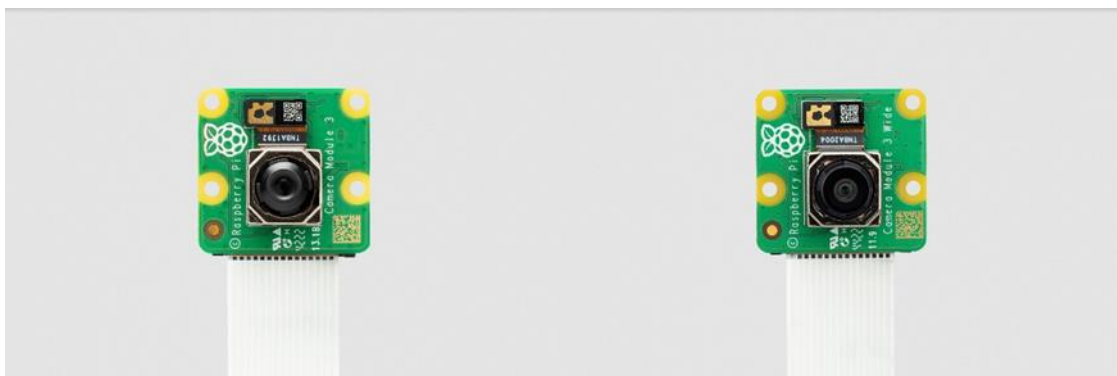


Рисунок 3.8 - Зовнішній вигляд Camera Module 3 - стандартний (ліворуч) і ширококутний (праворуч) [76]

В якості джерела живлення Raspberry Pi 5 виробник рекомендує застосовувати джерела, які можуть забезпечити стабільний струм 5 А при напрузі +5 В. Наприклад, УМБ Promate Titan-130 20000 mAh 2xUSB-C PD USB-A QC3.0 може забезпечити вихідну напругу у 5-20 в, вихідний струм 2-5 А,

максимальну вихідну потужність 130 Вт рис 3.9 [77]. Вартість становить від 3000 до 4000 тисяч гривень.



Рисунок 3.9 - Зовнішній вигляд Promate Titan-130 20000 mAh
[77]

Для системи на основі одноплатного комп'ютера NVIDIA Jetson Nano, з джерелом світла - Cree XP-G3, динаміками LD-SP-UM20/8A та підсилювачем РМ8403 варто обрати камеру. NVIDIA Jetson Nano сумісний із різними камерами, що використовують інтерфейси MIPI CSI або USB. До них належать камери Raspberry Pi Camera Module V2 і Camera Module 3, e-CAM30_CUNANO, IMX219. Порівняльна таблиця для камер IMX219 [78], Raspberry Pi Camera Module V2, Camera Module 3 та e-CAM30_CUNANO [79]:

Таблиця 3.4 - Порівняння камер для NVIDIA Jetson Nano.

Параметр	IMX219 (8 MP)	Raspberry Pi Camera Module V2 (8 MP)	Camera Module 3 (12 MP)	e-CAM30_CUNANO (3.4 MP)
Роздільна здатність	8 MP	8 MP	12 MP	3.4 MP
Кут огляду	~155°	~62°	~63°	~118°
Вольтаж	3.3 В	3.3 В	3.3 В	3.3 В
Споживаний струм	150-200 мА	250 мА	250 мА	350 мА
Якість запису відео	1080p @ 30fps	1080p @ 30fps	4K @ 60fps	1080p @ 30fps
Максимальна споживана потужність	0.66 Вт	0.825 Вт	0.825 Вт	1.155 В
Вага	3.5 г	3.5 г	6.3 г	6 г
Вартість	40 доларів	40 доларів	50 доларів	75-80 доларів

У якості джерела живлення виробник рекомендує використовувати джерела, що можуть забезпечити 5 А при 5 В, одже, можна застосувати обраний раніше УМБ Promate Titan-130 20000 mAh 2xUSB-C PD USB-A QC3.0.

Висновки до розділу 3

Виконано оцінку часу необхідного на один цикл процесу розпізнавання-оголошення інформації враховуючі швидкість реакції людини на голосову інформацію, та швидкість проходження різних етапів самого процесу. Встановлено час, необхідний на оголошення слів та їх комбінацій різної довжини з врахуванням особливостей Української мови та мовлення. Встановлено, що мінімальний час необхідний на оголошення одного слова становить 129 мс, у той час, як комбінація назва з трьох довгих слів може займати дві секунди на оголошення з урахуванням паузи між словами.

Виконано розрахунок і підбір компонентів для варіантів системи на платах ESP32-S3-EYE , Raspberry Pi 5 та NVIDIA Jetson Nano. Розраховано світловий потік який має забезпечити джерело світла, для нормального функціонування системи при поганому освітленні, та обрано світлодіод Cree XP-G3.

Для оголошення інформації обрано динаміки LD-SP-UM20/8A, що мають компактні розміри, сумісні з обраними платами, забезпечують необхідний рівень гучності. Також, було обрано звуковий підсилювач PAM8403 для підключення динаміків до системи. А також модулі камер для плат, в яких вона відсутня у базовій комплектації.

Було розраховано загальну споживану потужність систем у різних комплектаціях з урахуванням роботи нейронних мереж, джерела світла і постійної роботи динаміків. На основі отриманої споживаної потужності було розраховано ємність джерела живлення та обрано оптимальні варіанти для забезпечення автономної роботи системи впродовж 3х годин.

РОЗДІЛ 4. ДОСЛІДЖЕННЯ МОЖЛИВОСТІ ВИКОРИСТАННЯ НЕЙРОНИХ МЕРЕЖ У СИСТЕМАХ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ З ПОДАЛЬШИМ ГОЛОСОВИМ ВИВОДОМ ІНФОРМАЦІЇ ДЛЯ ЛЮДЕЙ З ВАДАМИ ЗОРУ

4.1 Метод попередньої обробки і групування тренувальних даних для покращення розпізнавання об'єктів у складних ситуаціях.

Однією з найбільш складних ситуацій під час розпізнавання об'єктів лишається проблема пов'язана з наявністю сторонніх об'єктів які можуть знаходитися біля цільового об'єкту розпізнавання, тим самим «плутаючи» нейронну мережу. Особливо у випадку, коли одну нейронну мережу використано для розпізнавання декількох категорій об'єктів, які можуть знаходитися на зображенні разом. Наприклад, у вікні може відбиватися відображення автомобіля, в наслідок чого можна отримати в результаті розпізнавання «автомобіль», хоча основним об'єктом для розпізнавання у даному випадку мало бути вікно. Аналогічна ситуація може траплятися з об'єктами які часто трапляються у повсякденному житті викликаючи хибні результати розпізнавання.



Рисунок 4.1 – Приклад хибного розпізнавання цільового об'єкту

Попередня обробка та правильне групування тренувальних даних можуть підвищити точність розпізнавання об'єктів у складних ситуаціях роблячи нейронну мережу більш гнучкою та стійкою. Для попередньої обробки зображень було обрано Adobe Photoshop.

Adobe Photoshop — це потужний програмний продукт для професійної обробки зображень, розроблений компанією Adobe Systems [80]. Програма широко використовується у сферах графічного дизайну, фотографії, веб-дизайну, а також у створенні цифрових ілюстрацій. Програма надає широкий набір інструментів для редагування і обробки зображень.

Основний принцип попередньої обробки тренувальних даних полягає у видаленні надлишкової інформації або зайвих об'єктів з зображення за допомогою інструменту «Гумка» з подальшим змішуванням оброблених тренувальних даних з необробленими.

Інструмент "Гумка" використовується для видалення пікселів на шарі, перетворюючи їх на прозорі (для шарів з прозорістю) або змінюючи колір на фон (для фонового шару). Інструмент обрано для того, щоб краї довкола основних об'єктів після видалення зайвих елементів були не ідеальними.

Інструмент функціонує за принципом зміни значення пікселів у межах його дії. При роботі на звичайному шарі пікселі змінюються на прозорість (альфа-канал отримує значення 0). При роботі на фоні або шарі без прозорості видалені пікселі замінюються кольором фону. Кожен піксель після дії інструменту "Гумки" розраховується за формулою:

$$P_{\text{new}} = P_{\text{original}} \cdot (1 - \alpha) + P_{\text{background}} \cdot \alpha,$$

де P_{new} - фінальне значення пікселя, P_{original} - початкове значення пікселя, $P_{\text{background}}$ - колір фону (для шарів без прозорості), α - рівень прозорості, який залежить від параметрів Opacity та Flow.

Непрозорість (Opacity) встановлює інтенсивність видалення пікселів (часткове або повне видалення). Потік (Flow) визначає швидкість нанесення ефекту при натисканні. Якщо "Гумка" працює по прозорому шару, пікселі

зменшують свою непрозорість вдвічі. Якщо шар непрозорий, колір фону впливає на видалені пікселі із врахуванням формули.

Для проведення експерименту створено 5 груп тренувальних даних із назвами "windows", "door", "trees", "traffic lights", "crosswalk" для 5 категорій об'єктів відповідно. Обрані категорії об'єктів постійно зустрічаються у повсякденному житті, їх розпізнавання необхідне для забезпечення безпечного пересування людей з вадами зору або повною сліпотю як на вулиці, так і в домашніх умовах.

Кожна група містить по 30 тренувальних необроблених зображень. Половину з необроблених тренувальних даних для кожної категорії об'єктів було оброблено у Adobe Photoshop методом видалення зайвого за допомогою інструменту «Гумка». Приклад необроблених та оброблених тренувальних зображень представлено на рисунку 4.2.

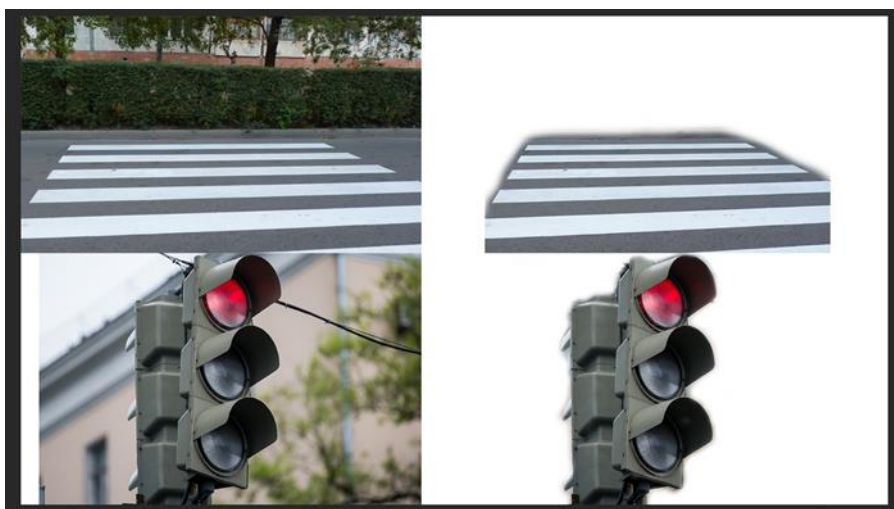


Рисунок 4.2 – Приклад необроблених та оброблених тренувальних даних

Для тренування нейронних мереж обрано платформу Edgeimpulse. На базі платформи створено проект для навчання моделей нейронних мереж для подальшого завантаження на плати ESP32-S3-EYE, Raspberry Pi 5 та NVIDIA Jetson Nano. У якості таргету для завантаження нейронних мереж спершу обрано ESP32-S3-EYE. У налаштуваннях вказано доступні для застосування ресурси мікроконтролера (рис 4.3).

Після налаштувань мережі для вихідного девайсу, завантажено по 30 необроблених тренувальних зображень для кожної з категорій (рис 4.4.), та 65

тестувальних зображень зроблених при різних умовах освітленості, кутах нахилу та погодних умовах. Після завантаження даних створено «Імпульс» .

Configure your target device and application budget

Target device
Define your target device requirements to inform model optimizations and performance calculations. No device yet? Use the default settings which you can change at any time.

Target device: Espressif ESP-EYE (ESP32 240MHz)

Processor family: ESP32

Clock rate: 240 MHz

Custom device name (optional):

Application budget
Specify the available RAM and ROM for the model's operation, along with the maximum allowed latency for your specific application. Not sure yet? Start with the defaults and modify them later on.

RAM: 4 MB

ROM: 4 MB

Latency: 100 ms

[Reset to default settings](#) [Cancel](#) [Save](#)

Рисунок 4.3 - Налаштування проекту під ESP32-S3-EYE

В Edge Impulse "Імпульс" — це набір алгоритмів та послідовностей обробки даних, які визначають, як дані від датчиків (зображення, звук, рух тощо) трансформуються в придатну форму для нейронної мережі. Він охоплює всі етапи підготовки даних, побудови моделі та прогнозування.

Dataset

Training (150) Test (65)

Apply filters [Clear all filters](#)

By label
Deselect all
☒ Crosswalk (30)
☒ Doors (30)
☒ Traffic_light (30)
☒ Trees (30)
☒ Windows (30)

By name or ID
 Enter a sample name or ID

By signature validity
Valid & invalid signatures

Enabled & disabled samples
Enabled & disabled samples

Рисунок 4.4 - Завантажені для тренування та тестування мережі данні

Імпульс складається з таких складових як: блок налаштувань вхідних даних, у випадку зображень це Image data, блоку який відповідає за навчання

мережі для обраних категорій (Transfer Learning), та блоку який генерує остаточні ознаки обраних категорій об'єктів (Output features) рисунок 4.5.

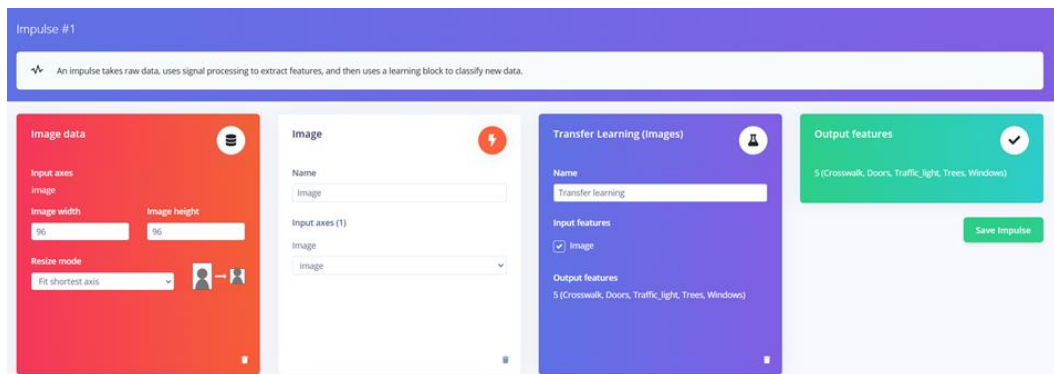


Рисунок 4.5 – Створений імпульс для обраних категорій об'єктів

В якості налаштувань вхідних параметрів зображень обрано вхідний розмір 96 * 96 пікселів для першого тесту, та 160 * 160 пікселів для другого тесту. В якості кольорового простору для зображень обрано RGB. Після чого для обраних категорій зображень було виділено ознаки. Візуальний розподіл ознак – це графічне представлення виділених ознак з кожної групи зображень. Кожна категорія об'єктів має свої характерні ознаки, які в подальшому будуть використовуватися для навчання мережі. Отриманий розподіл для необроблених тренувальних даних обраних 5 категорій представлено на рисунку 4.6.

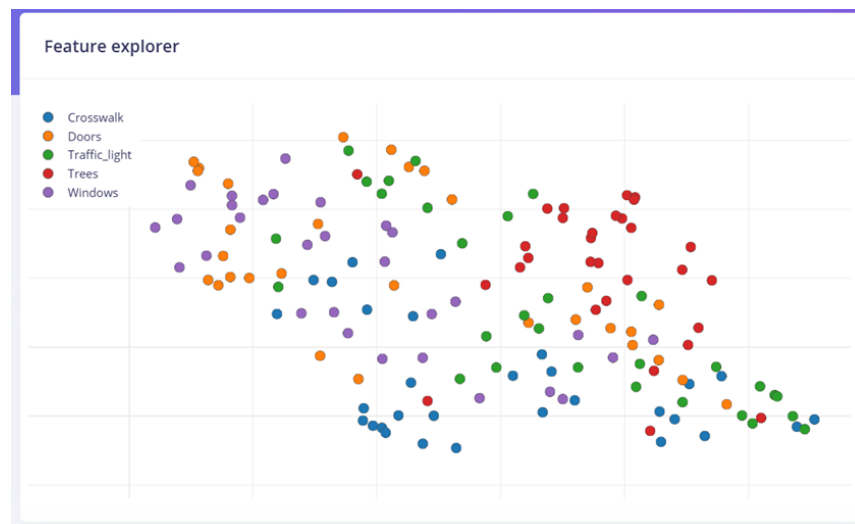


Рисунок 4.6 – Отриманий візуальний розподіл ознак для необроблених тренувальних даних

Проаналізувавши результати розподілу можна зробити висновок, що у обраних категоріях зображень досить частим є явище присутності декількох категорій об'єктів на одному зображенні та схожої надлишкової інформації, наприклад, схожий задній фон, що створює нерівномірний розподіл, та може викликати помилки при розпізнаванні.

Після виділення ознак виконано налаштування для тренування мережі та обрано модель. На платформі є можливість обрати мережі MobileNet V1 та V2 з розміром входного зображення 96*96 або 160*160 пікселів та різними коефіцієнтами ширини мережі, а також модель EfficientNet B0, що розроблена ком'юніті. EfficientNet B0 має значно більші потреби у потужності та розрахована на пристрої Linux, тому не була обрана для подальших експериментів. Список доступних для роботи з зображеннями моделей у Edge Impulse представлено на рисунку 4.7.

MobileNetV1 96x96 0.25 OFFICIALLY SUPPORTED	A pre-trained multi-layer convolutional network designed to efficiently classify images. Uses around 105.9K RAM and 301.6K ROM with default settings and optimizations.	Edge Impulse	Add
MobileNetV1 96x96 0.2 OFFICIALLY SUPPORTED	Uses around 83.1K RAM and 218.3K ROM with default settings and optimizations. Works best with 96x96 input size. Supports both RGB and grayscale.	Edge Impulse	Add
MobileNetV1 96x96 0.1 OFFICIALLY SUPPORTED	Uses around 53.2K RAM and 101K ROM with default settings and optimizations. Works best with 96x96 input size. Supports both RGB and grayscale.	Edge Impulse	Add
MobileNetV2 96x96 0.35 OFFICIALLY SUPPORTED	Uses around 296.8K RAM and 575.2K ROM with default settings and optimizations. Works best with 96x96 input size. Supports both RGB and grayscale.	Edge Impulse	Add
MobileNetV2 96x96 0.1 OFFICIALLY SUPPORTED	Uses around 270.2K RAM and 212.3K ROM with default settings and optimizations. Works best with 96x96 input size. Supports both RGB and grayscale.	Edge Impulse	Add
MobileNetV2 96x96 0.05 OFFICIALLY SUPPORTED	Uses around 265.3K RAM and 162.4K ROM with default settings and optimizations. Works best with 96x96 input size. Supports both RGB and grayscale.	Edge Impulse	Add
MobileNetV2 160x160 1.0 OFFICIALLY SUPPORTED	Uses around 1.3M RAM and 2.6M ROM with default settings and optimizations. Works best with 160x160 input size. Supports RGB only.	Edge Impulse	Add
MobileNetV2 160x160 0.75 OFFICIALLY SUPPORTED	Uses around 1.3M RAM and 1.7M ROM with default settings and optimizations. Works best with 160x160 input size. Supports RGB only.	Edge Impulse	Add
MobileNetV2 160x160 0.5 OFFICIALLY SUPPORTED	Uses around 700.7K RAM and 982.4K ROM with default settings and optimizations. Works best with 160x160 input size. Supports RGB only.	Edge Impulse	Add
MobileNetV2 160x160 0.35 OFFICIALLY SUPPORTED	Uses around 683.3K RAM and 658.4K ROM with default settings and optimizations. Works best with 160x160 input size. Supports RGB only.	Edge Impulse	Add
EfficientNet B0 COMMUNITY	Transfer learning model based on efficientnetb0_notop.h5 weights. This is a much larger model than MobileNet for Linux devices and accelerators.	Community blocks	Add

Рисунок 4.7 – Список доступних для роботи з зображеннями моделей у Edge Impulse

MobileNet використовує параметр α (альфа), який відомий як множник ширини, для оптимізації моделі. Цей параметр дозволяє зменшити обсяг нейронної мережі та підвищити швидкість обчислень за допомогою рівномірного зменшення кількості каналів у кожному шарі. Множник ширини α дозволяє зменшити кількість обчислювальних операцій та кількість параметрів приблизно в квадраті (α^2). Цей метод застосовується до будь-якої архітектури нейронної мережі, щоб створити менш об'ємну модель з новим рівнем точності, швидкістю та розміром, де $\alpha = 1$ – це базова MobileNet, а $\alpha < 1$ – стиснуті моделі MobileNets.

Для навчання обрано декілька моделей з різними коефіцієнтами стиснення та вхідним розміром зображення, а саме MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, MobileNetV2 160x160 0.5 та нестиснута версія MobileNetV2 160x160 1.0.

MobileNetV1 96x96 0.2 — це швидка, малогабаритна модель, яка ефективно працює при мінімальному використанні ресурсів завдяки низькому значенню множника ширини. Після тренування мережі під ESP32-S3-EYE виконано тестування та було отримано точність розпізнавання для кожної з груп об'єктів і представлено на рисунку 4.8.



Рисунок 4.8 – Результати точності розпізнавання мережі MobileNetV1 96x96 0.2 для ESP32-S3-EYE

З отриманих результатів можна зробити висновки, що у обраних категоріях досить часто на тренувальних зображеннях присутні об'єкти з суміжних категоріях, що створює хибні розпізнавання мережею MobileNetV1 96x96 з

кофіцієнтом стиснення 0.2. Найвища точність розпізнавання у категорії «crosswalk» і складає 80% для обраної мережі з данним набором тестувальних та тренувальних даних, та 42,9 % найнижча точність для категорії «Trafick_light». Середня точність мережі становить 63.3%, а теоретичний час розпізнавання після завантаження мережі на мікроконтролер становить 990 ms.

Графічне преставлення результатів – це візуальний розподіл тестування мережі на усіх зображеннях у мережі, як тренувальних так і тестувальних. Графічне преставлення результатів та теоретичні потреби у ресурсах мікроконтролера представлено на рисунку 4.9.



Рисунок 4.9 – Графічне представлення результатів розпізнавання

Після отримання результатів виконано перенавчання мережі на тих самих тренувальних та тестових зображеннях для моделей MobileNetV2 96x96 0.35 MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 під плату ESP32-S3-EYE.

Для MobileNetV2 96x96 0.35 отримано результати з середньою точністю 90% рис. 4.10. Найнижча точність розпізнавання у категорії «Trafick_light».

Last training performance (validation set)

ACCURACY 90.0%

LOSS 0,35

Confusion matrix (validation set)

	CROSSWALK	DOORS	TRAFFIC_LIGHT	TREES	WINDOWS
CROSSWALK	100%	0%	0%	0%	0%
DOORS	0%	83.3%	0%	0%	16.7%
TRAFFIC_LIGHT	14.3%	0%	71.4%	0%	14.3%
TREES	0%	0%	0%	100%	0%
WINDOWS	0%	0%	0%	0%	100%
F1 SCORE	0.91	0.91	0.83	1.00	0.83

Рисунок 4.10 - Результати точності розпізнавання мережі MobileNetV2 96x96 0.35 для ESP32-S3-EYE

Графічне представлення результатів та теоретичні потреби у ресурсах мережі MobileNetV2 96x96 0.35 для ESP32-S3-EYE відображено на рисунку 4.11.



Рисунок 4.11 – Графічне представлення результатів розпізнавання мережі MobileNetV2 96x96 0.35 для ESP32-S3-EYE

Теоретичний час розпізнавання становить 1598 мс, що є досить великим значенням для системи розпізнавання у режимі реального часу.

Для тренування мереж з вхідним розміром зображень 160*160 встановлено відповідні налаштування «Імпульсу» та виконано виділення ознак, а також тренування та тестування мережі для MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0. Візуальне відображення отриманих ознак для мережі MobileNetV2 160x160 0.5 зображено на рисунку 4.12.

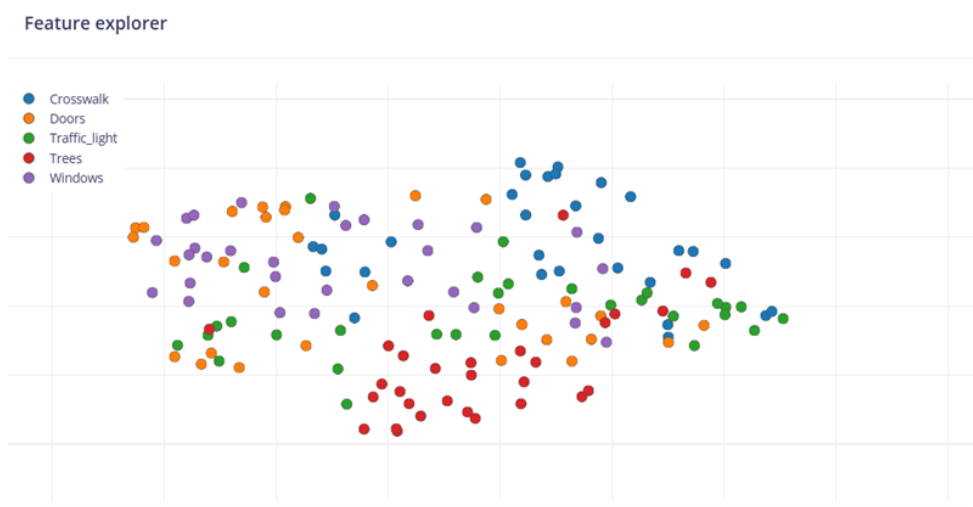


Рисунок 4.12 – Візуальний розподіл ознак з вхідним розміром зображень 160*160

Після перенавчання мережі було виконано тестування мережі MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.

Результати точності розпізнавання мережі MobileNetV2 160x160 0.5 представлено на рисунку 4.13.

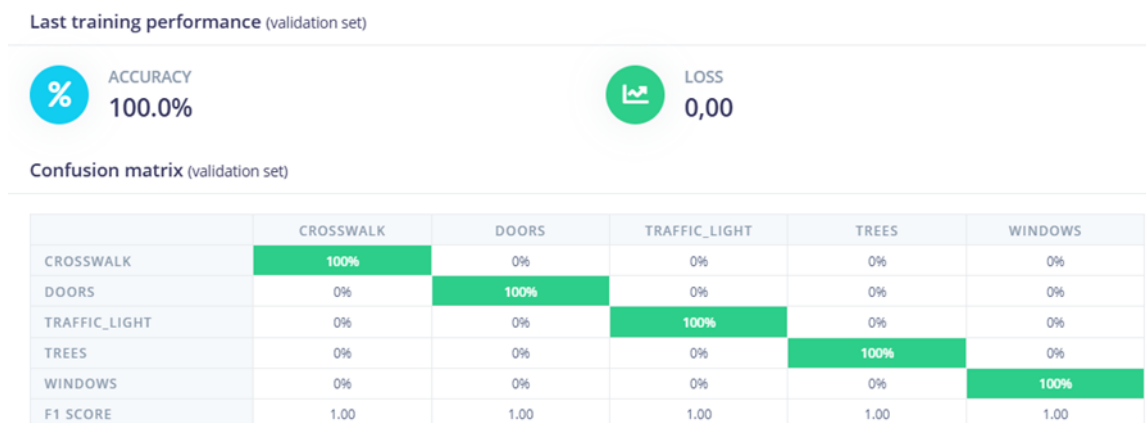


Рисунок 4.13 – Результати точності розпізнавання мережі MobileNetV2 160x160 0.5 для ESP32-S3-EYE

Отримано 100% точність розпізнавання об'єктів у всіх категоріях. Але , при цьому, теоретична швидкість розпізнавання об'єктів становить близько 4.5 секунд, що є дуже великим часом для систем розпізнавання у режимі реального часу рисунок 4.14.



Рисунок 4.14 - Візуальний розподіл результатів розпізнавання мережею MobileNetV2 160x160 0.5

Результати точності розпізнавання мережі MobileNetV2 160x160 1 представлено на рисунку 4.15.

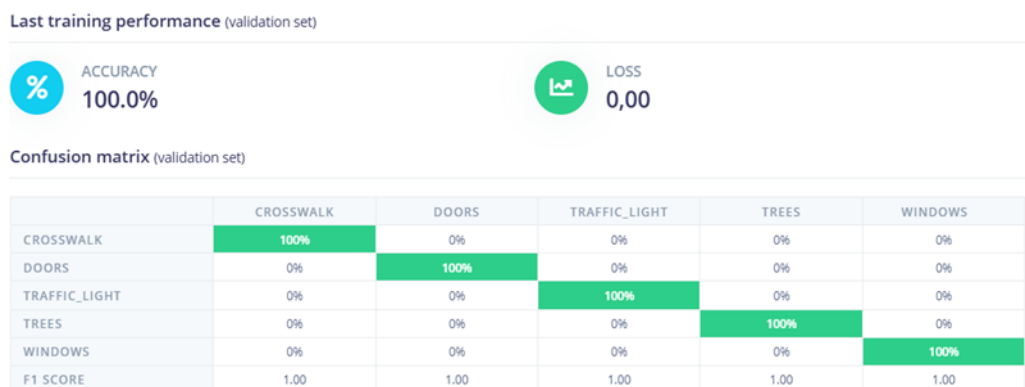


Рисунок 4.15 - Результати точності розпізнавання мережі MobileNetV2 160x160 1 для ESP32-S3-EYE

Точність розпізнавання має показник 100%, але прогнозований час розпізнавання перевищує 13 секунд, що є неприпустимим для використання у системах розпізнавання у режимі реального часу.

Проаналізувавши отриманий візуальний розподіл результатів можна зробити висновок, що чим менше стиснута мережа та чим більший розмір вхідного зображення, тим вища точність розпізнавання, більш рівномірний візуальний розподіл отриманих результатів та більший час необхідний на розпізнавання рис. 4.16.



Рисунок 4.16 - Візуальний розподіл результатів розпізнавання мережею MobileNetV2 160x160 1 для ESP32-S3-EYE

Аналогічно було виконано навчання та тестування мереж MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 для Raspberry Pi 5 та NVIDIA Jetson Nano на тих самих необроблених тренувальних і тестових зображеннях.

Для мережі MobileNetV1 96x96 0.2 на базі плати Raspberry Pi 5 було отримано аналогічні результати точності розпізнавання, що відображені на рисунку 4.17.



Рисунок 4.17 - Точність розпізнавання мережі MobileNetV1 96x96 0.2 на базі плати Raspberry Pi 5.

При цьому, отримано теоретичну швидкість розпізнавання у 3 мс, що є дуже високим показником швидкості розпізнавання рисунок 4.18.

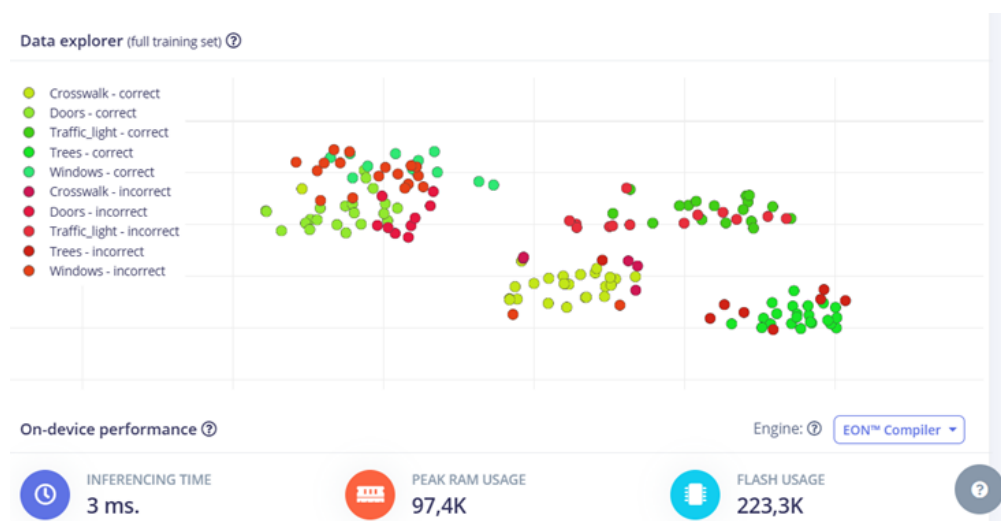


Рисунок 4.18 - Візуальний розподіл розпізнавання мережі MobileNetV1 96x96 0.2 на базі плати Raspberry Pi 5

Для моделі MobileNetV2 96x96 0.35 отримано аналогічний результат точності розпізнавання (рис.4.19), однак теоретична швидкість розпізнавання становить 5 мс, що є значним меншим показником ніж на базі плати ESP32-S3-EYE (рис.4.20).

Last training performance (validation set)



ACCURACY
90.0%



LOSS
0,35

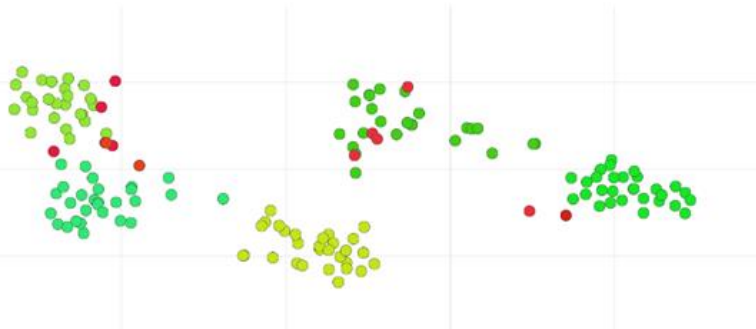
Confusion matrix (validation set)

	CROSSWALK	DOORS	TRAFFIC_LIGHT	TREES	WINDOWS
CROSSWALK	100%	0%	0%	0%	0%
DOORS	0%	83.3%	0%	0%	16.7%
TRAFFIC_LIGHT	14.3%	0%	71.4%	0%	14.3%
TREES	0%	0%	0%	100%	0%
WINDOWS	0%	0%	0%	0%	100%
F1 SCORE	0.91	0.91	0.83	1.00	0.83

Рисунок 4.19 - Точність розпізнавання мережі MobileNetV2 96x96 0.35 на базі плати Raspberry Pi 5.

Data explorer (full training set) ?

- Crosswalk - correct
- Doors - correct
- Traffic_light - correct
- Trees - correct
- Windows - correct
- Doors - incorrect
- Traffic_light - incorrect
- Trees - incorrect
- Windows - incorrect



On-device performance ?

Engine: ?

EON™ Compiler



INFERRING TIME
5 ms.



PEAK RAM USAGE
334,7K



FLASH USAGE
575,6K

Рисунок 4.20 - Візуальний розподіл результатів розпізнавання моделі MobileNetV2 96x96 0.35 на базі плати Raspberry Pi 5

Для моделі MobileNetV2 160x160 0.5 отримано 100% точність розпізнавання з теоретичною швидкістю розпізнавання у 6 мс рис.4.21.



Рисунок 4.21 - Візуальний розподіл результатів розпізнавання моделі MobileNetV2 160x160 0.5 на базі плати Raspberry Pi 5

Після тренування і тестування мережі отримано 100% точність розпізнавання. Теоретична швидкість розпізнавання становить 33 мс, що є дуже високим показником, і відмінно підходить для систем розпізнавання об'єктів у режимі реального часу рисунок 4.22.



Рисунок 4.22 - Візуальний розподіл результатів розпізнавання моделі MobileNetV2 160x160 1 на базі плати Raspberry Pi 5

Для плати NVIDIA Jetson Nano було виконано аналогічні навчання і тестування моделей мереж MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0.

Для моделі MobileNetV1 96x96 0.2 отримано середню точність у 80%, та теоретичну швидкість розпізнавання у 2 мс рисунок 4.23 та 4.24.



Рисунок 4.23 - Точність розпізнавання мережі MobileNetV1 96x96 0.2 на базі плати NVIDIA Jetson Nano.

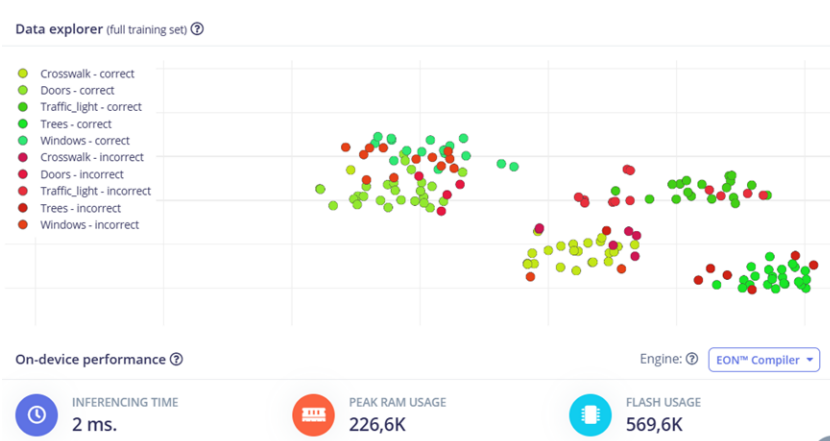


Рисунок 4.24 - Візуальний розподіл результатів розпізнавання моделі MobileNetV1 96x96 0.2 на базі плати NVIDIA Jetson Nano

Для моделі MobileNetV2 96x96 0.35 отримано середню точність 90%, при теоретичній швидкості розпізнавання 3 мс рис.4.25.

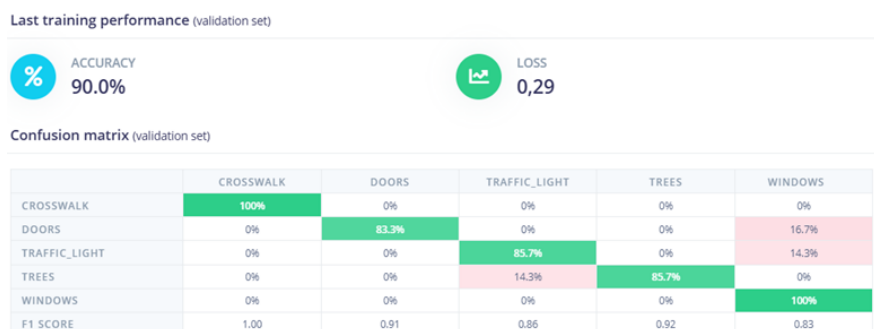


Рисунок 4.25 - Точність розпізнавання мережі MobileNetV2 96x96 0.35 на базі плати NVIDIA Jetson Nano

Для мережей MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 отримано точність розпізнавання у 100% рис.4.26. Та швидкість розпізнавання у 9 мс та 27 мс відповідно рисунок 4.27 та рисунок 4.28.



Рисунок 4.26 - Точність розпізнавання мережей MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 на базі плати NVIDIA Jetson Nano



Рисунок 2.27 - Теоретичний час та необхідні ресурси для мережі MobileNetV2 160x160 0.5 на базі NVIDIA Jetson Nano



Рисунок 4.28 - Візуальний розподіл результатів розпізнавання мережею MobileNetV2 160x160 1 та прогнозований час розпізнавання

При однаковій точності розпізнавання у мереж MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0, візуальний розподіл результатів більш чіткий і рівномірний у випадку нестиснутої мережі. При цьому, прогнозований час

розпізнавання у 27 мс є невеликим і відмінно підходить для застосування у системах розпізнавання у режимі реального часу.

Прогнозований час розпізнавань для обраних мереж та плат представлено у таблиці 4.1.

Таблиця 4.1 – прогнозований час розпізнавання.

Плати/прогнозований час розпізнавання для мереж	ESP32-S3-EYE	Raspberry Pi 5	NVIDIA Jetson Nano
MobileNetV1 96x96 0.2	990 мс	3 мс	2 мс
MobileNetV2 96x96 0.35	1598 мс	5 мс	3 мс
MobileNetV2 160x160 0.5	4451 мс	6 мс	9 мс
MobileNetV2 160x160 1	13727 мс	33 мс	27 мс

Точність розпізнавання для обраних плат у мереж MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 в усіх випадках становила 100%, але у випадку мережі MobileNetV2 160x160 без стиснення візуальний розподіл результатів був більш рівномірний та чіткий.

4.2 Перевірка точності розпізнавання обраних категорій після змішування оброблених і необроблених тренувальних даних.

У другому експерименті було оброблено половину тренувальних зображень методом видалення надлишкової інформації у Photoshop, після чого оброблені і необроблені зображення було змішано навпіл і використано для тренування нейронних мереж MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 під плати ESP32-S3-EYE, Raspberry Pi 5 та NVIDIA Jetson Nano.

Візуальний розподіл ознак для об'єднаних тренувальних даних зображено на рисунку 4.29.

З отриманого розподілу ознак можна зробити висновки, що тепер, для кожної категорії об'єктів виділено більш чіткі і окремі групи ознак, що має підвищити гнучкість мережі, особливо для невеликих мереж з великим коефіцієнтом стиснення.

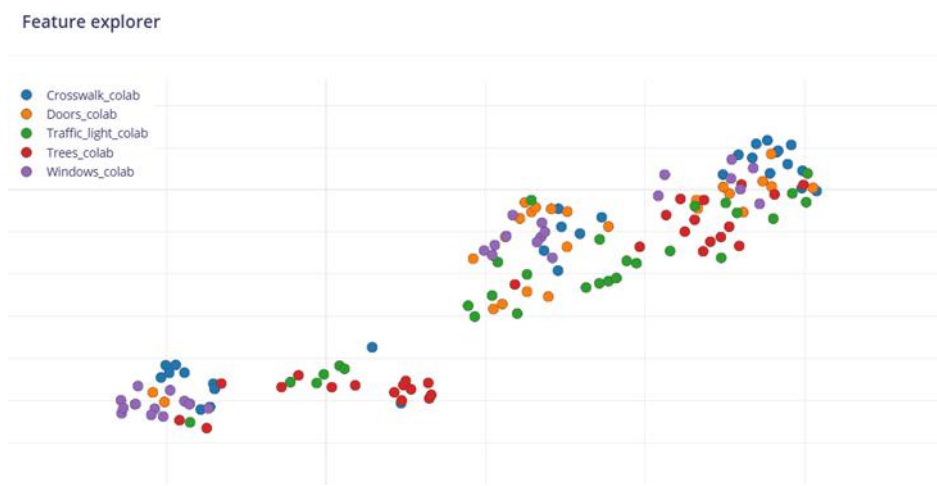


Рисунок 4.29 - Візуальний розподіл ознак для об'єднаних тренувальних даних

Після тренування і тестування мережі MobileNetV1 96x96 0.2 з поєднаними обробленими і необробленими тренувальними зображеннями, було отримано середню точність розпізнавання у 70%, і покращення результату у всіх категоріях крім категорії «Windows». Результати точності розпізнавання мережі MobileNetV1 96x96 0.2 для плати ESP32-S3-EYE зі змішаними тренувальними даними представлено на рисунку 4.30.

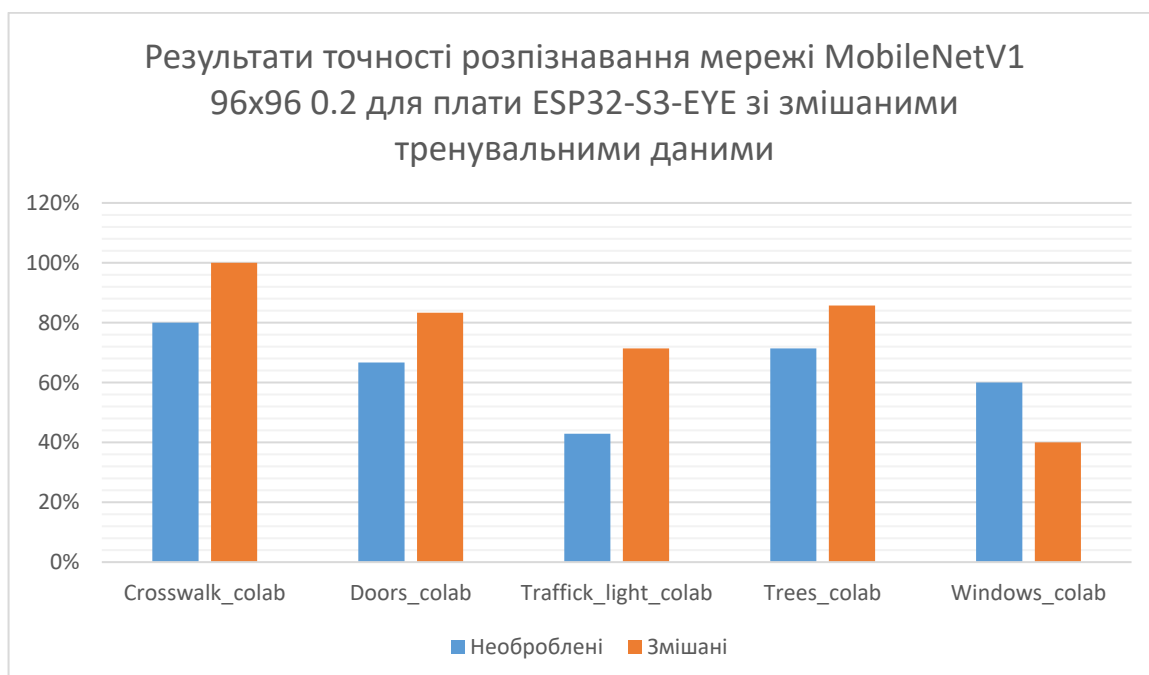


Рисунок 4.30 - Результати точності розпізнавання мережі MobileNetV1 96x96 0.2 для плати ESP32-S3-EYE зі змішаними тренувальними даними

Проаналізувавши отриманий результат можна зробити висновки, що для категорій об'єктів, де площа корисної інформації в межах кадру ($S_{\text{корисна}}$) більша за площу надлишкової інформації ($S_{\text{надлишкова}}$), метод видалення надлишкової інформації і тренування мережі на основі поєднання оброблених і необроблених даних 1 до 1 дають приріст у точності розпізнавання обраних категорій. У випадку, коли площа цільового об'єкту значно менша, за площу сторонньої інформації, яку було видалено під час етапу попередньої обробки, можуть виникати хибні розпізнавання, як у випадку з вікнами. Для автоматизації процесу у майбутньому, дану умову можна описати у вигляді залежності:

Якщо $S_{\text{надлишкова}} > S_{\text{корисна}}$, попередню обробку не виконувати;

У випадку $S_{\text{надлишкова}} < S_{\text{корисна}}$, попередню обробку виконувати.

Встановлено, що у випадку, коли біля світлофору може висіти знак пішохідного переходу стиснуті варіанти мережі, можуть його розпізнати як пішохідний перехід, і може виникнути хибне розпізнавання і оголошення хибного результату рис.4.31.

u-vinnici-na-vulici-privokzalniy-nezaba...
Label: Traffic_light_colab
Predicted: Crosswalk_Colab
[View sample](#)



Рисунок 4.31 – Приклад хибного розпізнавання цільового об'єкту

Тому, для категорії «Windows» було повернуто повністю необроблені тренувальні данні і виконано перенавчання мережі.

В результаті було покращено точність розпізнавання моделі, в тому числі у порівнянні з необробленими тестувальними зображеннями для всіх категорій окрім категорії «Windows». Це пов'язано з схожими патернами які існують у візуальному відображенні вікон і дверей, наприклад, коли і вікна і двері мають максимально наближений вигляд і стиль, а також з інформацією яка може відбиватися чи просвічуватися крізь віконне скло. Результати для моделі MobileNetV1 96x96 0.2 представлено на рисунку 4.32.

Last training performance (validation set)

ACCURACY
80.0%

LOSS
0,87

Confusion matrix (validation set)

	CROSSWALK_COLAB	DOORS_COLAB	TRAFFIC_LIGHT_COLAB	TREES_COLAB	WINDOWS
CROSSWALK_COLAB	100%	0%	0%	0%	0%
DOORS_COLAB	0%	83.3%	0%	0%	16.7%
TRAFFIC_LIGHT_COLAB	0%	0%	71.4%	0%	28.6%
TREES_COLAB	0%	0%	14.3%	85.7%	0%
WINDOWS	20%	0%	20%	0%	60%
F1 SCORE	0.91	0.91	0.71	0.92	0.55

Рисунок 4.32 – Результати точності розпізнавання мережі MobileNetV1 96x96 0.2 для плати NVIDIA Jetson Nano зі змішаними тренувальними даними, після повернення необроблених даних у категорію «Windows».

Для моделі MobileNetV2 96x96 0,35 було отримано покращення результатів у порівнянні з необробленими тренувальними зображеннями рис.4.33.

Last training performance (validation set)

ACCURACY
96.7%

LOSS
0,16

Confusion matrix (validation set)

	CROSSWALK_COLAB	DOORS_COLAB	TRAFFIC_LIGHT_COLAB	TREES_COLAB	WINDOWS
CROSSWALK_COLAB	100%	0%	0%	0%	0%
DOORS_COLAB	0%	83.3%	0%	0%	16.7%
TRAFFIC_LIGHT_COLAB	0%	0%	100%	0%	0%
TREES_COLAB	0%	0%	0%	100%	0%
WINDOWS	0%	0%	0%	0%	100%
F1 SCORE	1.00	0.91	1.00	1.00	0.91

Рисунок 4.33 – Результати точності розпізнавання мережі MobileNetV2 96x96 0,35 для плати NVIDIA Jetson Nano зі змішаними тренувальними даними, після повернення необроблених даних у категорію «Windows».

Результати точності розпізнавання необроблених і комбінованих тренувальних зображень на базі плати NVIDIA Jetson Nano представлено у таблиці 4.2.

В результаті було покращено точність розпізнавання моделі, в тому числі у порівнянні з необробленими тестувальними зображеннями для всі категорії окрім категорії «Windows». Це пов'язано з схожими патернами які існують у візуальному відображенні вікон і дверей, наприклад, коли і вікна і двері мають максимально наближений вигляд і стиль, а також з інформацією яка може відбиватися чи просвічуватися крізь віконне скло.

Таблиця 4.2 – Результати точності розпізнавання необроблених і комбінованих тренувальних зображень на базі плати NVIDIA Jetson Nano.

Назва мережі/ категорії	Crosswalk	Doors	Traffic_light	Trees	Windows
MobileNetV1 96x96 0.2 необроблені	80%	66,7%	42,9%	71,4%	60%
MobileNetV1 96x96 0.2 комбіновані	100%	83,3%	71,4%	85,7%	60%
MobileNetV2 96x96 0.35 необроблені	100%	83,3%	71,4%	100%	100%
MobileNetV2 96x96 0.35 комбіновані	100%	83,3%	100%	100%	100%

Для моделей мереж MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 було отримано точність 100%, і прогнозований час розпізнавання у 15 та 31 мс відповідно.

На базі плати ESP32-S3-EYE для мережі MobileNetV1 96x96 0.2 після тренування на комбінованих даних також було отримано приріст у точності розпізнавання. Для категорії «Windows» було отримано результат точності у 40%, що є нижчим показником, ніж було при всіх необроблених тренувальних даних. В той самий час, для всіх інших категорії було отримано приріст у точності розпізнавання. Встановлено, що велика кількість об'єктів, які можуть відбиватися у відображенні вікон, або просвічуватися крізь віконне скло, може створювати хибні розпізнавання, якщо в одній нейронній мережі знаходяться категорії об'єктів, яка можуть бути у цих відображеннях у вікнах для невеликих

моделей мереж з високим коефіцієнтом стиснення, а саме MobileNetV1 96x96 0.2 та MobileNetV2 96x96 0.35 з розміром вхідного зображення 96*96 пікселів.

Last training performance (validation set)



Confusion matrix (validation set)

	CROSSWALK_COLAB	DOORS_COLAB	TRAFFIC_LIGHT_COLAB	TREES_COLAB	WINDOWS
CROSSWALK_COLAB	80%	0%	0%	0%	20%
DOORS_COLAB	0%	100%	0%	0%	0%
TRAFFIC_LIGHT_COLAB	0%	0%	71.4%	14.3%	14.3%
TREES_COLAB	0%	0%	0%	85.7%	14.3%
WINDOWS	0%	40%	0%	20%	40%
F1 SCORE	0.89	0.86	0.83	0.80	0.40

Рисунок 4.34 - Результати точності розпізнавання мережі MobileNetV1 96x96 0.2 для плати ESP32-S3-EYE зі змішаними тренувальними даними

On-device performance

Engine: **EON™ Compiler**

INFERENCE TIME
921 ms.

PEAK RAM USAGE
97,4K

FLASH USAGE
223,3K

Рисунок 4.35 - Прогнозований час розпізнавання та потреби у пам'яті мережі MobileNetV1 96x96 0.2 для ESP32-S3-EYE зі змішаними тренувальними даними

Last training performance (validation set)



Confusion matrix (validation set)

	CROSSWALK_COLAB	DOORS_COLAB	TRAFFIC_LIGHT_COLAB	TREES_COLAB	WINDOWS
CROSSWALK_COLAB	100%	0%	0%	0%	0%
DOORS_COLAB	0%	83.3%	0%	0%	16.7%
TRAFFIC_LIGHT_COLAB	0%	0%	100%	0%	0%
TREES_COLAB	0%	0%	0%	100%	0%
WINDOWS	0%	0%	0%	0%	100%
F1 SCORE	1.00	0.91	1.00	1.00	0.91

Рисунок 4.36 - Результати розпізнавання мережі MobileNetV2 96x96 0.35 після тренування на комбінованих даних для ESP32-S3-EYE



Рисунок 4.37 - Прогнозований час розпізнавання та потреби у пам'яті мережі MobileNetV2 96x96 0.35 для ESP32-S3-EYE

Для моделей мереж MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 було отримано точність 100%, та прогнозований час розпізнавання у 5537 мс та 15046 мс відповідно рисунок 4.38 та рисунок 4.39. Такий час є неприпустимими для розпізнавання об'єктів у режимі реального часу.



Рисунок 4.38 - Прогнозований час розпізнавання та потреби у пам'яті мережі MobileNetV2 160x160 0.5 для ESP32-S3-EYE



Рисунок 4.39 - Прогнозований час розпізнавання та потреби у пам'яті мережі MobileNetV2 160x160 1 для ESP32-S3-EYE

Для плати Raspberry Pi 5 було отримано аналогічні значення точності розпізнавання як для NVIDIA Jetson Nano, прогнозований час розпізнавання для мереж MobileNetV2 160x160 0.35 та MobileNetV2 160x160 1, а також споживані ресурси зображено на рисунках 4.40 та 4.41 відповідно.



Рисунок 4.40 - Прогнозований час розпізнавання та потреби у пам'яті мережі MobileNetV2 160x160 0.5 для Raspberry Pi 5



Рисунок 4.41 - Прогнозований час розпізнавання та потреби у пам'яті мережі MobileNetV2 160x160 1 для Raspberry Pi 5

Результати точності розпізнавання необроблених і комбінованих тренувальних зображень на базі плати ESP32-S3-EYE представлено у таблиці 4.3.

Таблиця 4.3 - Результати точності розпізнавання необроблених і комбінованих тренувальних зображень на базі плати ESP32-S3-EYE

Назва мережі/ категорії	Crosswalk	Doors	Traffick_light	Trees	Windows
MobileNetV1 96x96 0.2 необроблені	80%	66,7%	42,9%	71,4%	60%
MobileNetV1 96x96 0.2 оброблені	100%	83,3%	71,4%	71,4%	40%
MobileNetV2 96x96 0.35 необроблені	100%	83,3%	71,4%	100%	100%
MobileNetV2 96x96 0.35 оброблені	100%	83,3%	100%	100%	100%

Також мережі MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 було протестовано за допомогою телефону у режимі реального часу у вуличних умовах. На основі телефону Samsung Galaxy S21 FE з процесором Qualcomm Snapdragon 888, графічним процесором Adreno 660 та 8 Гб оперативної пам'яті отримано швидкість розпізнавання 12-14 мс для моделі MobileNetV2 160x160 1.0. Приклади роботи розпізнавання у режимі реального часу представлено на рисунку 4.42.

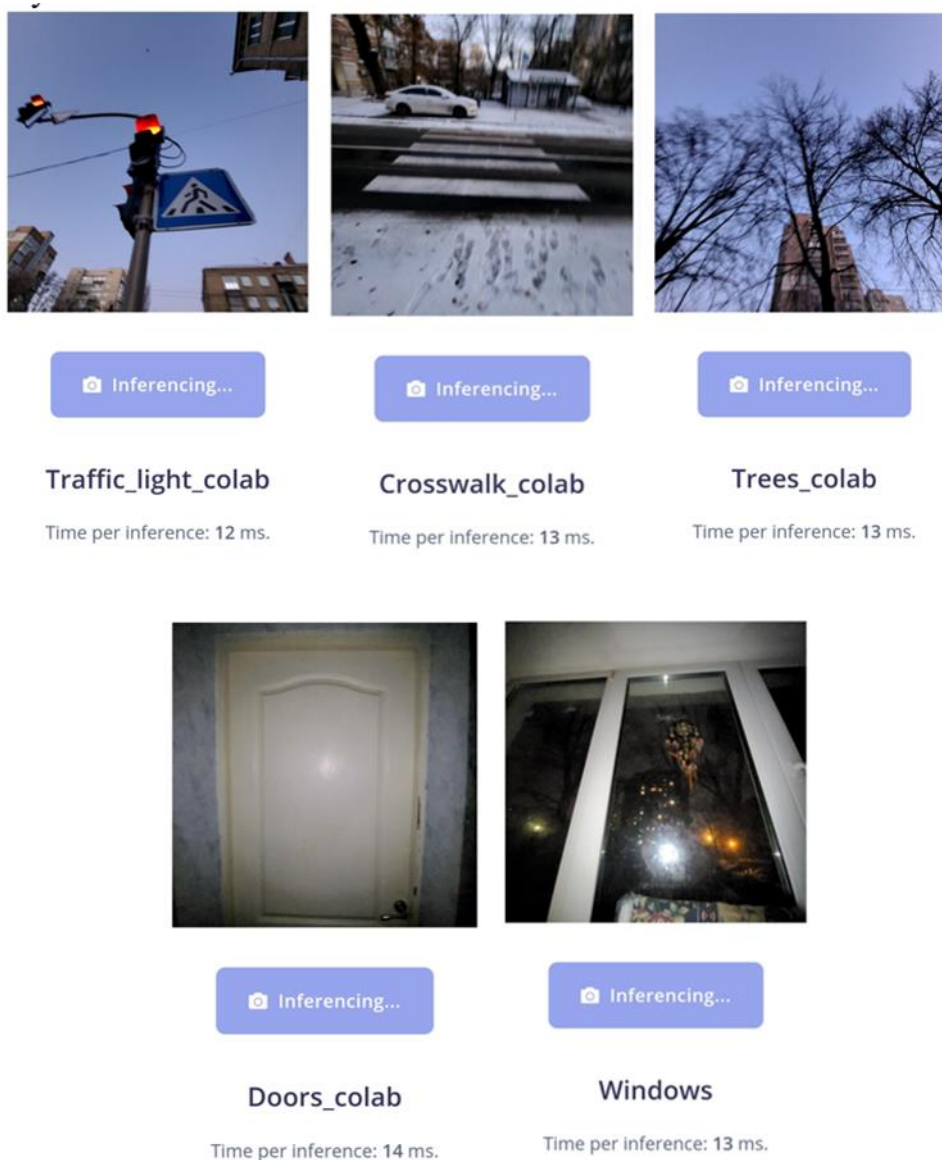


Рисунок 4.42 - Результати розпізнавання мережі MobileNetV2 160x160 1.0 в режимі реального часу

Для визначення максимального і мінімального часу проходження циклу обрано середній час розпізнавання у 50 мс для плат Raspberry Pi 5 та NVIDIA Jetson Nano, для ESP32-S3-EYE прогнозований час розпізнавання для мереж MobileNetV2 160x160 становить більше 10 секунд, що є неприйнятним для використання у системах розпізнавання в режимі реального часу.

Мінімальний і максимальний час проходження одного повного циклу розпізнавання-оголошення результатів з урахуванням особливостей Української мови та швидкості реакції людини на звукову інформацію розраховано за формулою:

$$t_{\text{ц}} = t_{\text{роз}} + t_{\text{назви}} + t_{\text{реакції}}$$

$$t_{\text{цмін}} = 50 + 129 + 200 = 379 \text{ мс}$$

$$t_{\text{цмакс}} = 50 + 1948 + 200 = 2198 \text{ мс}$$

Таким чином, мінімальний час на проходження одного циклу становить 379 мс, максимальний час проходження одного повного циклу становить 2198 мс.

4.3 Представлення зовнішнього вигляду системи.

Для комфортного використання системи розпізнавання об'єктів з подальшим голосовим сповіщенням необхідно врахувати потреби людей які будуть нею користуватися. У якості місця для кріплення камери обрано окуляри. Основний корпус буде мати прямокутну форму з округлими кутами, для зручності носіння. Габарити корпусу складатимуть 15 см ширини, 30 см довжини, та 15 см висоти, для нормального розміщення усіх елементів. Всередині корпус матиме внутрішні кріплення для фіксації електронних компонентів. В якості основного матеріалу для корпусу обрано полікарбонат або ABS-пластик. Полікарбонат є легким, міцним і ударостійким матеріалом. ABS-пластик забезпечує міцність і легкість водночас. Сплав PC-ABS поєднує обидві переваги: високу ударостійкість і термостійкість.

Вентиляційні отвори на задній та верхній панелях. Вентиляційні решітки захищають від пилу. З обох боків передбачені спеціальні кріплення під регульований ремінь.

Кришка на верхній частині корпусу застосовується для доступу всередину корпусу. Для забезпечення захисту та створення герметичності, необхідно додати герметичні гумові прокладки.

Для організації внутрішніх та зовнішніх дротів передбачено отвори для кабельних каналів.

Висновки до розділу 4

В результаті дослідження точності та швидкості розпізнавання 5 обраних категорій об'єктів, а саме "windows", "doors", "trees", "traffic light", "crosswalk",

нейронними мережами MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 для мікроконтролерів ESP32-S3-EYE, NVIDIA Jetson Nano та Raspberry Pi 5 було проведено декілька експериментів з необробленими та змішаними тренувальними зображеннями.

Для необроблених тренувальних даних було отримано середні показники точності для мереж MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35 у 63% та 90 %, а для мереж MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 отримано точність розпізнавання у 100%. При цьому, прогнозований час розпізнавання для мікроконтролера ESP32-S3-EYE у випадку моделей MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0 становив більше 10 секунд, що є неприйнятним для використання у системах розпізнавання в режимі реального часу.

Для плат NVIDIA Jetson Nano та Raspberry Pi 5 середній прогнозований час розпізнавання становить від 3 мс для моделей MobileNetV1 96x96 0.2, MobileNetV2 96x96 0.35, та від 14 мс до 33 мс у випадку мереж MobileNetV2 160x160 0.5 та MobileNetV2 160x160 1.0.

Після змішування оброблених і необроблених тренувальних даних було отримано приріст у точності розпізнавання для категорій "doors", "trees", "traffic light", "crosswalk", для категорії "windows" було отримано погіршення точності розпізнавання. Після зворотної заміни комбінованих тренувальних даних у категорії "windows" на необроблені, було отримано приріст у точності розпізнавання для мережі MobileNetV1 96x96 0.2 до 80 % для плат NVIDIA Jetson Nano та Raspberry Pi 5 та приріст до 73,3% для плати ESP32-S3-EYE. Для мережі MobileNetV2 96x96 0.35 було отримано точність розпізнавання у 96,7% для всіх плат.

Також, було встановлено мінімальний та максимальний час необхідних на проходження повного циклу розпізнавання-оголошення результатів з урахуванням особливостей Української мови та швидкості реакції людиною на слухову інформацію. Мінімальний час становить 379 мс у випадку назви об'єкту що складається з одного короткого слова, та 2198 мс у випадку для назви об'єкту з трьох довгих слів.

ВИСНОВКИ

У дисертаційній роботі проведено удосконалення систем розпізнавання об'єктів для людей з вадами зору та повною сліпотою шляхом застосування нейронних мереж на мікроконтролерах, та введення голосового сповіщення, а також розрахунок технічних параметрів окремих елементів системи та підбір компонентів на їх основі.

Проведено літературні огляди актуальних досліджень в цій галузі, запропоновано власний варіант системи розпізнавання об'єктів з подальшим голосовим сповіщенням, проведено відповідні розрахунки та отримано такі результати:

1. Проведено аналіз існуючих методів та алгоритмів які можна застосувати у системах розпізнавання об'єктів для людей з вадами зору та повною сліпотою. На основі проведеного аналізу для системи розпізнавання об'єктів обрано методи на основі глибокого навчання, а саме згорткові нейронні мережі. Згорткові нейронні мережі спеціалізовані під задачі з розпізнавання об'єктів і можуть бути розгорнуті на пристроях з обмеженими ресурсами як мікроконтролери та одноплатні комп'ютери.
2. Проведено дослідження архітектур нейронних мереж та можливості їх застосування у системах на мікроконтролерах та одноплатних комп'ютерах, а також дослідження мікроконтролерів та одноплатних комп'ютерів, які підтримують розгортання нейронних мереж. В результаті аналізу серед існуючих архітектур обрано клас мереж MobileNet, що здатні забезпечити високу точність розпізнавання об'єктів, можуть бути розгорнуті на пристроях з обмеженими ресурсами завдяки коефіцієнту ширини мережі, та змінному вхідному розміру зображення, мають активну підтримку та постійний розвиток, а також підтримуються платформами для навчання та тестування нейронних мереж.

3. В результаті аналізу існуючих мікроконтролерів та одноплатних комп'ютерів, які підтримують можливість розгортання нейронних мереж, обрано ESP32-S3-EYE, Raspberry Pi 5 та NVIDIA Jetson Nano для використання у системі розпізнавання об'єктів з голосовим сповіщенням. Обрані мікроконтролери та одноплатні комп'ютери здатні працювати з нейронними мережами в режимі реального часу, мають можливість підключення динаміків, підтримують програмне забезпечення для генерації тексту у мову, а також мають невеликі розміри та високу енергоефективність.
4. Розроблено адаптивну комбіновану систему розпізнавання об'єктів з подальшим голосовим сповіщенням для людей з вадами зору та повною сліпотою. Описано принцип роботи системи, її зовнішній вигляд, а також необхідні матеріали та складові компоненти для її реалізації.
5. В результаті проведеного дослідження існуючих платформ для тренування нейронних мереж обрано платформу Edge Impulse. Данна платформа підтримує обрані мережі та мікроконтролери, має постійний розвиток, зручний для роботи інтерфейс та повний цикл для навчання та тестування нейронних мереж.
6. Досліджено можливості застосування голосових генеративних моделей для запропонованої системи. В результаті дослідження обрано програмне забезпечення eSpeak NG, що може працювати в офлайн режимі, підтримує українську мову, має відкриту ліцензію, а також гнучкі налаштування швидкості генерації мови та паузи між словами.
7. Виконано розрахунок часу ,необхідного на виконання одного повного циклу розпізнавання-оголошення, та часу необхідного на оголошення назв об'єктів різної довжини для голосового сповіщення про об'єкти. Встановлено, що час, необхідний на оголошення одного символу становить 43 мс з урахуванням особливостей української мови та мовлення. Мінімальний час оголошення назви становить 129 мс, максимальний – 1948 мс з урахуванням пауз між словами. Мінімальний

час проходження повного циклу розпізнавання-оголошення з урахуванням швидкості реакції людини на голосову інформацію становить 379 мс, максимальний час - 2198 мс.

8. В результаті дослідження впливу запропонованого методу попередньої обробки зображень встановлено, що для категорій об'єктів, де площа корисної інформації в межах кадру більша за площу надлишкової інформації в кадрі метод видалення надлишкової інформації і тренування мережі на основі поєднання оброблених і необроблених даних у співвідношенні 1:1 дають приріст точності розпізнавання обраних категорій у більше ніж 10%. У випадку, коли площа цільового об'єкту значно менша, за площу сторонньої інформації в межах кадру, яку було видалено під час етапу попередньої обробки, можуть виникати хибні розпізнавання, як у випадку з вікнами.
9. Виконано підбір компонентів для запропонованої системи на основі плат ESP32-S3-EYE, Raspberry Pi 5 та NVIDIA Jetson Nano та розрахунок загальної споживаної потужності і ємності джерела живлення для забезпечення автономної роботи системи на 2-3 години. Також розраховано необхідну інтенсивність світла, та на її основі як джерело світла, для нормального функціонування системи при поганому освітлені обрано світлодіод Cree XP-G3. Для голосового сповіщення користувача обрано динаміки LD-SP-UM20/8A. Для підключення та роботи динаміків обрано підсилювач класу D - PAM8403. Як джерело живлення, для забезпечення автономної роботи системи протягом щонайменше 3 годин обрано УМБ Promate Titan-130 20000 mAh 2xUSB-C PD USB-A QC3.0.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Денісов Р.В. Оникієнко Ю.О. Особливості розпізнавання зображень нейронними мережами на прикладі MobileNetV1 та MobileNetV2 в системах на мікроконтролерах. «Технології та інжиниринг», Випуск 2(13), 2023 15-27 сторінки, DOI: 10.30857/2786-5371.2023.2.2
2. Денісов Р.В., Попович П. В., Особливості попередньої обробки та групування тренувальних даних нейронної мережі для підвищення точності розпізнавання об'єктів на основі MobileNetV2. «Технології та інжиниринг», Випуск 5(16), 2023 9-21 сторінки, DOI: 10.30857/2786-5371.2023.5.1
3. Денісов Р.В., Попович П. В., Особливості застосування систем розпізнавання об'єктів у режимі реального часу на мікроконтролерах з подальшим голосовим виводом інформації для людей з вадами зору. «Технології та інжиниринг», Випуск 3(20), 2024 21-30 сторінки, DOI: 10.30857/2786-5371.2024.3.2
4. Sinan Ozdemir, Divya Susarla, Feature Engineering Made Easy: Identify unique features from your dataset in order to build powerful machine learning systems. Packt Publishing 2018, P.133-145.
5. Jason D. Bakos. Embedded Systems: ARM Programming and Optimization. Morgan Kaufmann, 2016, P. 135-140. <https://doi.org/10.1016/B978-0-12-800342-8.00003-1>.
6. Khaledyan, Donya ; Amirany, Abdollah ; Jafari, Kian ; Moaiyeri, Mohammad Hossein ; Zargari Khuzani, Abolfazl ; Mashhadi, Najmeh. Low-Cost Implementation of Bilinear and Bicubic Image Interpolation for Real-Time Image Super-Resolution. September 2020. DOI: 10.48550/arXiv.2009.09622.
7. Rafael C. Gonzalez, Richard E. Woods. Digital Image Processing. Pearson. 2018. The 4th Edition. P.165-167,627-630, 638-641, 742-745, 771-773,780-790, 881-883, 964-970.
8. Median Filters for Digital Images. URL: <https://micro.magnet.fsu.edu/primer/java/digitalimaging/processing/medianfilter/index.html>

9. Linear Contrast Enhancement. URL:
https://www.imageprocessing.com/2017/11/linear-contrast-enhancement.html#google_vignette
10. Changing the contrast and brightness of an image! URL:
https://docs.opencv.org/3.4/d3/dc1/tutorial_basic_linear_transform.html
11. Image preprocessing and enhancement. URL:
<https://library.fiveable.me/geospatial-engineering/unit-4/image-preprocessing-enhancement/study-guide/L5fa2eI0Xk8chrKd>
12. Quick Guide to Histogram Equalization for Clearer Images. URL:
<https://www.analyticsvidhya.com/blog/2022/01/histogram-equalization/>
13. Contrast Stretching. URL : <https://samirkhanal35.medium.com/contrast-stretching-f25e7c4e8e33>
14. Manipulating exposure and color channels. RGB to grayscale. URL:
https://scikit-image.org/docs/stable/auto_examples/color_exposure/plot_rgb_to_gray.html
15. Bařina, David. (2011). Gabor Wavelets in Image Processing. 10.48550/arXiv.1602.03308.
16. Through The Eyes of Gabor Filter. URL:
https://medium.com/@anuj_shah/through-the-eyes-of-gabor-filter-17d1fdb3ac97
17. Hadid, A. (2008). The Local Binary Pattern Approach and its Applications to Face Analysis. 2008 First Workshops on Image Processing Theory, Tools and Applications, 1-9.
18. Fabien Pierre, Mathieu Amendola, Clémence Bigeard, Timothé Ruel, Pierre-Frédéric Villard. Segmentation with Active Contours. Image Processing On Line, 2021, 11, pp.120 - 141. {10.5201/ipol.2021.298}. {hal-03235096}
19. Garcia-Garcia, A., Orts, S., Oprea, S., Villena-Martinez, V., & Rodríguez, J.G. (2017). A Review on Deep Learning Techniques Applied to Semantic Segmentation. ArXiv, abs/1704.06857.
20. What is instance segmentation? URL:
<https://www.ibm.com/think/topics/instance-segmentation>

21. Bay, Herbert & Tuytelaars, Tinne & Van Gool, Luc. (2006). SURF: Speeded up robust features. Computer Vision-ECCV 2006. 3951. 404-417. 10.1007/11744023_32.
22. Moghimi, Mohammad & Nayeri, Maryam & Pourahmadi, Majid & Moghimi, Mohammad Kazem. (2018). Moving Vehicle Detection Using AdaBoost and Haar-Like Feature in Surveillance Videos. International Journal of Imaging and Robotics. 10.48550/arXiv.1801.01698.
23. T. Surasak, I. Takahiro, C. -h. Cheng, C. -e. Wang and P. -y. Sheng, "Histogram of oriented gradients for human detection in video," 2018 5th International Conference on Business and Industrial Research (ICBIR), Bangkok, Thailand, 2018, pp. 172-176, doi: 10.1109/ICBIR.2018.8391187.
24. What is Template Matching? An Introduction. URL: <https://blog.roboflow.com/template-matching/>
25. Принципи побудови нейронних мереж. URL: <https://itmaster.biz.ua/programming/vision/neural-networks-principles.html>
26. Agarap, Abien Fred. (2018). Deep Learning using Rectified Linear Units (ReLU). 10.48550/arXiv.1803.08375.
27. Kim, Hannah & Jeong, Young-Seob. (2019). Sentiment Classification Using Convolutional Neural Networks. Applied Sciences (Switzerland). 9. 10.3390/app9112347.
28. Chen, Feng & Xu, Junlin. (2023). Deep learning algorithm-based wearable device in basketball motion dynamic analysis. Applied Mathematics and Nonlinear Sciences. 9. 10.2478/amns.2023.2.00120.
29. S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017, doi: 10.1109/TPAMI.2016.2577031
30. Murthy, Ch & Hashmi, Mohammad Farukh & Bokde, Neeraj & Geem, Zong Woo. (2020). Investigations of Object Detection in Images/Videos Using Various Deep

Learning Techniques and Embedded Platforms—A Comprehensive Review. *Applied Sciences*. 10.3390/app10093280.

31. Li, Qing & Lin, Yingcheng & He, Wei. (2021). SSD7-FFAM: A Real-Time Object Detection Network Friendly to Embedded Devices from Scratch. *Applied Sciences*. 11. 1096. 10.3390/app11031096.

32. Howard, G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T. et al. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv*, 17 Apr 2017, P. 1–9. DOI: 10.48550/arXiv.1704.04861.

33. Wang, Y., Yan, J., Sun, Q., Li, J., Yang, Z. (2019). A MobileNets Convolutional Neural Network for GIS Partial Discharge Pattern Recognition in the Ubiquitous Power Internet of Things Context: Optimization, Comparison, and Application. In: *IEEE Access*, Vol. 7, P. 150226–150236. DOI: 10.1109/ACCESS.2019.2946662.

34. Seeing AI. An app for visually impaired people that narrates the world around you. URL : <https://www.microsoft.com/en-us/garage/wall-of-fame/seeing-ai/>.

35. OrCam MyEye. URL: <https://www.orcam.com/en-us/orcam-myeye-2-pro>.

36. OrCam Read. URL: <https://www.orcam.com/en-us/orcam-read>.

37. Be My Eyes. See the world together. URL: <https://www.bemyeyes.com/>.

38. Vision AI for the Blind and Visually Impaired. URL: <https://www.aipoly.com/>.

39. Mukhiddinov M, Cho J. Smart Glass System Using Deep Learning for the Blind and Visually Impaired. *Electronics*. 2021; 10(22):2756. <https://doi.org/10.3390/electronics10222756>.

40. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. -C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 2018, pp. 4510-4520, doi: 10.1109/CVPR.2018.00474.

41. Bochkovskiy, Alexey & Wang, Chien-Yao & Liao, Hong-yuan. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. 10.48550/arXiv.2004.10934.
42. Saponara, Sergio & Elhanashi, Abdussalam & Qinghe, Zheng. (2022). Developing a real-time social distancing detection system based on YOLOv4-tiny and bird-eye view for COVID-19. *Journal of Real-Time Image Processing*. 19. 1-13. 10.1007/s11554-022-01203-5.
43. A Look at the EfficientNet/EfficientNet-Lite Architecture. URL: <https://nathanbaileyw.medium.com/efficientnet-efficientnet-lite-blog-b850b94c965d>.
44. FOMO: Object detection for constrained devices. URL: <https://docs.edgeimpulse.com/docs/edge-impulse-studio/learning-blocks/object-detection/fomo-object-detection-for-constrained-devices>.
45. FOMO: Real-Time Object Detection on Microcontrollers. URL: https://www.edge-ai-vision.com/wp-content/uploads/2022/05/E3W04_Jongboom_Edge_Impulse_2022.pdf
46. Liberis, Edgar & Lane, Nicholas. (2021). Differentiable Network Pruning for Microcontrollers. 10.48550/arXiv.2110.08350.
47. Weerasekara, D. T., Gamage, M. P. A. W., Kulasooriya, K. S. A. F. (2021). Combined Approach of Supervised and Unsupervised learning for Dog Face Recognition. 2021 6th International Conference for Convergence in Technology (I2CT), 2021, P. 1–5. DOI: 10.1109/I2CT51068.2021.9418175.
48. de Vita, F., Nocera, G., Bruneo, D., Tomaselli, V., Giacalone, D., Das, S. K. (2020). Quantitative Analysis of Deep Leaf: a Plant Disease Detector on the Smart Edge. 2020 IEEE International Conference on Smart Computing (SMARTCOMP), 2020, P. 49–56. DOI: 10.1109/SMARTCOMP50058.2020.00027.
49. Keshavamurthy, Mariyam, S. J., Meghamala, M., Meghashree, M., Neha (2019). Automatized Food Quality Detection and Processing System Using Neural Networks. 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), 2019, P. 1442–1446. DOI: 10.1109/RTEICT46194.2019. 9016919.

50. ESP32-S3 Series. URL: https://www.espressif.com/sites/default/files/documentation/esp32-s3_datasheet_en.pdf
51. ESP32-S3-EYE v2.2. URL: https://github.com/espressif/esp-who/blob/master/docs/en/get-started/ESP32-S3-EYE_Getting_Started_Guide.md#11-overview.
52. Nicla Vision. URL: <https://store.arduino.cc/products/nicla-vision?srsId=AfmBOooCMq0VJjX-4ovNmzjUi2r8JRc0bYPK2Nin4EgxlUa23M-lRuTh>.
53. Portenta H7. URL: https://store.arduino.cc/products/portenta-h7?srsId=AfmBOoo2HUIjPDfev4knBXRn6R9nT2RrQuddcv3tBrFLe5Sj_xDS0Th
54. Nano 33 BLE Sense Rev2. URL: <https://docs.arduino.cc/hardware/nano-33-ble-sense-rev2/>
55. Arduino Nano 33 BLE Sense Rev2 with headers. URL: <https://arduino.ua/prod6293-arduino-nano-33-ble-sense-rev2-with-headers>.
56. Raspberry Pi 4. URL: <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>.
57. Raspberry Pi 5. URL: <https://www.raspberrypi.com/products/raspberry-pi-5/>.
58. Google Coral Dev Board. URL: <https://coral.ai/products/dev-board/>.
59. NVIDIA Jetson Nano. Bringing the power of modern AI to millions of devices. URL: <https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/product-development/>.
60. Sharpen Your Edge AI and Robotics Skills with the NVIDIA Jetson Nano Developer Kit. URL: <https://developer.nvidia.com/blog/sharpen-your-edge-ai-and-robotics-skills-with-the-nvidia-jetson-nano-developer-kit/>.
61. Edge impulse. edgeimpulse.com. URL: <https://www.edgeimpulse.com/>.
62. TensorFlow. URL: <https://www.tensorflow.org/>
63. PyTorch Mobile. URL: <https://pytorch.org/mobile/home/>
64. eSpeak NG. URL: <https://github.com/espeak-ng/espeak-ng?tab=readme-ov-file>

65. Voxygen TTS. Voice Generation for Generation Voice. URL: <https://www.voxygen.fr/en/home>
66. Google Text-to-Speech AI. URL: <https://cloud.google.com/text-to-speech>
67. Global Estimates of Vision Loss. URL: <https://www.iapb.org/learn/vision-atlas/magnitude-and-projections/global/>
68. Алюмінієва тростина для сліпих OSD-BL590200. URL: <https://osd.ua/ua/products/kostyli-hodunki-trostri/trostri/alyuminievaya-trost-dlya-nezryachix-osd-bl590200.html>
69. Oleksandr Serhiyevich Ishchenko. Vowel sounds of the Ukrainian language depending on the tempo of speech: monograph. - K.: Institute of the Ukrainian Language of the National Academy of Sciences of Ukraine, 2012. - P. 109-119.
70. McCluney, William. (2015). Introduction to Radiometry and Photometry 2nd ed: Title Page & Table of Contents.
71. Світлодіод Cree XP-G3 холодний білий 5000K. URL: <https://forled.com.ua/cree-xp-g3-holodniy-beliy-5000k-20-mm-ua>
72. LD-SP-UM20/8A. URL: <https://elaso.com.ua/products/2-elektronnye-komponenty/11-istochniki-zvuka/name/6859-ld-sp-um20-8a>
73. LD-SP-MSI50-53 LOUDITY. URL: <https://www.tme.eu/ua/ru/details/ld-sp-msi50-53/dinamiki/loudity/>
77. Підсилювач звуковий D-класу PAM8403 2x 3Вт. URL: <https://www.mini-tech.com.ua/ua/usilitel-zvuka-d-klass-pam8403-2x3w>
75. Повербанк Andowl Q-CD6500 68000 mAh швидка зарядка QC3.0 PD30W. URL: <https://epicentrk.ua/ua/shop/mplc-poverbank-andowl-q-cd6500-68000-mah-svidka-zaradka-qc3-0-pd30w-cervonij-3755f6fe-1ee823cc-06a6-661a-8397-c7bc4ed85326.html>
76. Camera Module 3. About the Camera Modules. URL: <https://www.raspberrypi.com/documentation/accessories/camera.html>
77. УМБ Promate Titan-130 20000 mAh 2xUSB-C PD USB-A QC3.0 (titan-130).

78. 8MP IMX219. URL: <https://docs.arducam.com/Nvidia-Jetson-Camera/Native-Camera/imx219/>

79. e-CAM30_CUNANO - 3.4 MP Camera for NVIDIA® Jetson Nano™/ NVIDIA® Jetson Xavier™ NX. URL: https://www.e-consystems.com/nvidia-cameras/jetson-nano-cameras/e-CAM30_CUNANO.asp#module-features

80. Adobe Photoshop. URL: <https://www.adobe.com/ua/products/photoshop.html>