https://doi.org/10.31891/2219-9365-2025-81-50
УДК 004.4'277.2.056.55

ХОРОШУН Владислав
Вінницький національний технічний університет
https://orcid.org/0009-0002-6492-1514
e-mail: vlad119b@gmail.com
АЗАРОВА Анжеліка
Вінницький національний технічний університет
https://orcid.org/0000-0003-3340-5701
e-mail: azarova.angelika@gmail.com
МУРАЩЕНКО Олександр
Вінницький національний технічний університет
https://orcid.org/0009-0005-9314-2095
e-mail: murachenko@vntu.edu.ua
ЗОРЯ Ірина
Вінницький національний технічний університет
https://orcid.org/0009-0002-7151-3455
e-mail: iryna.zoria03@gmail.com

ПРОГРАМНО-АПАРАТНИЙ КОМПЛЕКС ДЛЯ ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗОВАНОГО КЕРУВАННЯ АВТОМОБІЛЕМ ЗАСОБАМИ НЕЙРОМЕРЕЖЕВОГО МОДЕЛЮВАННЯ ТА ГЕНЕТИЧНИХ АЛГОРИТМІВ

У статті представлено програмно-апаратний комплекс для забезпечення автопілотажу засобами нейромережевого моделювання та генетичних алгоритмів. У комплексі використано бібліотеку YOLO v6 для розпізнавання об'єктів у реальному часі та для подальшого прийняття рішень щодо ситуації на дорозі авторами запропоновано використання багатошарового перцептрону, для оптимізації навчання якого було застосовано генетичний алгоритм, що дозволило суттєво зменшити тривалість його навчання. Розроблена система використовує DirectX 11 для передавання зображень із будь-якої камери, що дозволяє інтегрувати різноманітні візуальні джерела даних. Основний науковий результат полягає в тому, що запропоновано удосконалений підхід до автоматичного управління автомобілем із використанням багатошарового перцептрону, прискорене навчання якого відбувається засобами генетичного алгоритму, що уможливлює значне зростання продуктивності системи.

Ключові слова: автопілот, нейромережеве моделювання, YOLO v6, багатошаровий перцептрон, генетичний алгоритм, активаційні функції, оптимізація навчання.

KHOROSHUN Vladyslav, AZAROVA Anzhelika,
MURASHCHENKO Oleksandr, ZORYA Iryna
Vinnytsia National Technical University

SOFTWARE-HARDWARE COMPLEX FOR AUTOMATED VEHICLE CONTROL USING NEURAL NETWORK MODELING AND GENETIC ALGORITHMS

The article presents a hardware-software complex for providing autopiloting using neural network modeling and genetic algorithms. The complex uses the YOLO v6 library for real-time object recognition and for further decision-making regarding the road situation. The authors propose the use of a multilayer perceptron, the training of which was optimized using a genetic algorithm, which significantly reduced the duration of its training. The developed system uses DirectX 11 to transmit images from any camera, which allows integrating various visual data sources. The main scientific result is that an improved approach to automatic vehicle control using a multilayer perceptron has been proposed, the accelerated training of which is carried out using a genetic algorithm, which allows a significant increase in system performance. One of the key advantages of the approach proposed by the authors is the ability to adapt to changes. As the software-hardware complex for automated driving of a car collects more data about the environment, the genetic algorithm can update the weights of the neural network, increasing the accuracy of decisions and the safety of the system. The collected data is used to further improve the system based on new generations of GA.

The study shows that the combination of genetic algorithms and MLP allows to significantly accelerate learning and increase the accuracy of decision-making in autopilot systems. The use of YOLO v6 for real-time object detection provides high performance and adaptability of the system to changing conditions.

Keywords: autopilot, neural network modeling, YOLO v6, MLP, genetic algorithm, activation functions, training optimization.

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

Сучасні технології, зокрема штучний інтелект та нейронні мережі, активно впливають на різні сфери життя разом із автоматизованими системами керування. Використання нейромережевих моделей, таких як YOLO v6, та генетичних алгоритмів для оптимізації процесу навчання стає ключовим елементом розвитку автономних систем керування.

Новітні системи автопілотажу є важливою частиною розвитку автономного транспорту, однак вони стикаються з викликами, що обмежують їх ефективність і безпеку. Однією з головних проблем є низька

швидкість навчання нейронних мереж через високу обчислювальну складність, що впливає на їхню здатність швидко адаптуватися до змінюваних умов. Також існують труднощі з адаптацією моделей до реальних умов, що негативно впливає на стабільність роботи систем. Вирішення цих проблем є важливим завданням для підвищення ефективності автоматизованих систем керування. Дослідження в цьому напрямку можуть значно покращити продуктивність автономних технологій і підвищити їхню безпеку в різних середовищах експлуатації.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

Дослідження у сфері штучного інтелекту, нейромережевого моделювання та еволюційних алгоритмів активно проводяться провідними науковцями. Зокрема, значний внесок у розвиток нейронних мереж здійснили І. Гудфеллоу, Я. Лекун, А. Карпати, Д. Сільвер, а у сфері генетичних алгоритмів – К. Голдберг, Д. Фогель, М. Мітчелл та ін.

У науковій літературі широко висвітлено переваги нейронних мереж для вирішення завдань розпізнавання образів, а також важливість еволюційних алгоритмів для оптимізації процесів навчання. Проте, недостатньо уваги приділено питанням прискорення навчання нейронних мереж, що є вельми актуальним завданням, особливо для підвищення швидкодії реагування систем автопілотажу на постійно змінювану дорожню обстановку. Крім того, варто дослідити можливість застосування комбінованих методів оптимізації, які могли б забезпечити більш збалансовану продуктивність і стійкість до різних видів шуму та змінних умов експлуатації.

Незважаючи на значну кількість досліджень у галузі нейронних мереж та штучного інтелекту, питання оптимізації навчання багатошарового перцептрону (MLP) шляхом використання генетичних алгоритмів залишається недостатньо вивченим. Зокрема, потребує детальнішого аналізу вплив генетичних алгоритмів на швидкість і точність навчання нейромережевих моделей, особливо в умовах обмежених обчислювальних ресурсів. Крім того, слабко досліджено питання використання комбінованих методів еволюційного та градієнтного навчання, що може забезпечити ефективніше використання обчислювальних ресурсів.

Дослідження цих аспектів є важливим для подальшого вдосконалення автономних систем керування, зокрема у сфері автомобільного транспорту, безпілотних літальних апаратів та робототехнічних комплексів. Крім того, це відкриває можливості для створення більш гнучких та адаптивних систем, здатних ефективно працювати в умовах невизначеності та змінних параметрів середовища.

ФОРМУЛЮВАННЯ ЦІЛЕЙ СТАТТІ

Метою статті є оптимізація процесу навчання багатошарового перцептрону засобами генетичних алгоритмів для підвищення ефективності автономних систем керування. Особлива увага приділяється розробленню підходу, що дозволяє зменшити кількість ітерацій та тривалість навчання нейронної мережі, що є критично важливим для автономних систем, які працюють у режимі реального часу, зокрема автопілотажу.

Крім того, у статті розглядаються можливості адаптації розроблених методів для різних типів автономних систем, а також потенціал використання оптимізованих моделей у промислових додатках, зокрема для роботизованих виробничих ліній та інтелектуальних систем контролю якості. Результати дослідження можуть стати основою для подальшої інтеграції штучного інтелекту в сучасні системи керування, що сприятиме їхній продуктивності та надійності в реальних умовах експлуатації.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

Застосування архітектури YOLO для моніторингу дорожньої обстановки

YOLO v6 є однією з найновіших і покращених версій оригінального YOLO, що представляє собою бібліотеку мови Python та C++, що засобами нейронних мереж глибокого навчання уможливлює розпізнавання об’єктів. Оригінальна версія цієї мережі отримала такі ключові нововведення, як Path Aggregation Network (PANet) та Feature Pyramid Network (FPN), що підвищують точність і швидкість виявлення об'єктів [1].

Архітектура удосконаленої мережі-гібрида складається з кількох рівнів згорткових шарів (convolutional layers), які витягують ознаки із зображення. Використовуються блоки таких шарів, як CBL (Convolutional, BatchNorm, Leaky ReLU) для покращення процесу зниження розмірності та кількості параметрів. Важливим елементом є FPN (Feature Pyramid Networks) для багаторівневої детекції об'єктів.

Базовим компонентом бібліотеки YOLO є згорткові нейронні мережі (Convolutional Neural Networks (CNN)) – це тип глибокої нейронної мережі, спеціально розробленої для оброблення та аналізу візуальних даних, таких як зображення та відео. CNN є основною архітектурою для розв'язання задач комп'ютерного зору, таких як класифікація зображень, детекція об'єктів, сегментація тощо.

Розглянемо основні компоненти CNN, що містить багатошаровий перцептрон для створення системи автоматизованого керування автомобілем:

1. Згорткові шари (Convolutional layers) – це ключовий компонент CNN. Згортковий шар

використовує спеціальні фільтри (ядра), які переміщуються по всьому зображенню, виконуючи операцію згортки (convolution). Це дозволяє автоматично виділяти важливі ознаки зображення, такі як краї, текстури та інші локальні деталі, з мінімальною кількістю ручного втручання.

2. Шар об'єднання (Pooling layer) зменшує просторові розміри зображень, зберігаючи найважливіші ознаки, що допомагає зменшити обчислювальні витрати та знижує ризик перенавчання. Найпоширенішими типами є Max Pooling, який вибирає найбільше значення з фрагмента, і Average Pooling, який обчислює середнє значення.

3. Активаційні функції (Activation functions). Після кожної згорткової операції застосовують нелінійні функції активації, такі як ReLU (Rectified Linear Unit), щоб модель могла вивчати більш складні шаблони й розпізнавати не лише лінійні ознаки.

4. Повнозв'язані шари (Fully connected layers). Після кількох згорткових і шарів об'єднання результат подається на один або кілька повнозв'язаних шарів. Це класичні нейронні шари, де кожен нейрон зв’язаний із кожним нейроном попереднього шару. Вони використовуються для завершення класифікації на основі витягнутих ознак.

5. Вихідний шар надає результат прогнозу. У задачах класифікації зображень зазвичай використовують Softmax для багатокласових задач або Sigmoid для бінарних задач [2].

Ключовою ідеєю FPN є побудова піраміди ознак, яка дозволяє моделі отримувати інформацію з різних рівнів абстракції. Схема її архітектури показано на рис. 1.

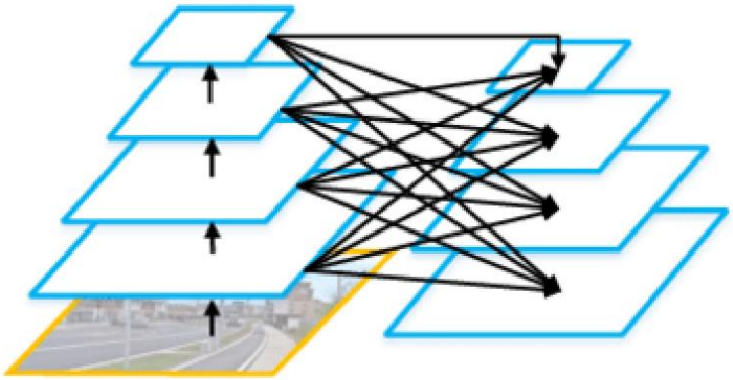

Рис. 1. Схематичне зображення архітектури Feature Pyramid Network

Нижчі рівні мережі мають високу роздільну здатність, але менше абстрактних ознак, тоді як вищі рівні містять більше абстракцій, але з меншою деталізацією. Поєднання цих рівнів дозволяє моделі розпізнавати як дрібні, так і великі об'єкти на зображенні.

Розглянемо основні етапи роботи FPN (Feature piramid network — мережева піраміда ознак), що застосовується для створення системи автоматизованого керування автомобілем:

Побудова базової нейронної мережі: спочатку модель пропускає зображення через стандартну глибоку нейронну мережу, щоб отримати кілька рівнів ознак.

Після цього відбувається «зворотне» передавання ознак, де інформація з вищих рівнів передається на нижчі рівні, що допомагає деталізувати інформацію з абстрактніших ознак.

Ознаки з різних рівнів поєднуються, щоб отримати більш точні представлення об'єктів на різних масштабах. Це дозволяє моделі краще ідентифікувати як великі, так і дрібні об'єкти на зображенні [1].

Отже, автори статті пропонують для моніторингу дорожньої обстановки розробити ПЗ, що імплементує можливості бібліотеки DirectX 11 для передавання зображень з будь-якої камери, що дозволяє синхронізоване зчитування отриманої інформації засобами бібліотеки YOLO v6. Особливістю такого процесу є інтеграція різноманітних візуальних джерел даних.

Побудова багатошарового перцептрону для управління дорожньою ситуацією

Застосування бібліотеки YOLO v6 уможливлює розпізнавання об'єктів з ідентифікацією таких параметрів для кожного з них [1]:

– bounding box coordinates: координати (x, y, ширина, висота);

– клас об'єкта: тип об'єкта (машина, пішохід, дорожній знак тощо);

– confidence score: ймовірність (впевненість) у визначенні об'єкта.

Щоб розрахувати кількість вхідних, прихованих та вихідних нейронів у багатошаровому перцептроні (MLP) для автопілота на основі даних від YOLO v6, скористаємося наданими параметрами та загальними рекомендаціями.

Отже, вхідні дані для MLP складаються з інформації від YOLO v6 і додаткових сенсорних даних від автомобіля:

– загальна кількість параметрів на один об'єкт: 6 (4 координати bounding box + 1 клас об'єкта + 1 ймовірність);

– максимальна кількість об'єктів на кадр: 10.

Оскільки багатошаровий перцептрон складається з кількох шарів: вхідного шару, одного або кількох прихованих шарів і вихідного шару та кожен нейрон у шарах з'єднаний з усіма нейронами наступного шару (повнозв'язна архітектура), то:

1. Розрахуємо кількість вхідних нейронів для MLP від YOLO v6:

$$N_{YOLO} = 6 \cdot 10 = 60.$$

Розглянемо додаткові сенсорні дані**:**

поточна швидкість: 1 нейрон;

– поточний кут керма: 1 нейрон;

– лівий поворотник увімкнено: 1 нейрон;

– правий поворотник увімкнено: 1 нейрон;

– нахил дороги: 1 нейрон;

– поточне прискорення/сповільнення: 1 нейрон;

– загальна кількість додаткових вхідних нейронів:

$$N_{extra} = 1 + 1 + 1 + 1 + 1 + 1 = 6;$$

– загальна кількість вхідних нейронів**:**

$$N_{input} = N_{YOLO} + N_{extra} = 60 + 6 = 66.$$

2. Вихідні нейрони. Для автопілота необхідно приймати рішення щодо управління такими функціями, як:

– рульове управління (поворот керма): 1 нейрон;

– прискорення/гальмування: 1 нейрон;

– лівий поворотник: 1 нейрон;

– правий поворотник: 1 нейрон.

Загальна кількість вихідних нейронів:

$$N_{output} = 4.$$

3. Приховані шари. Вибір кількості нейронів у прихованих шарах може базуватися на емпіричних правилах та експериментальних результатах. Один із поширених підходів — використовувати кількість нейронів у прихованих шарах як середнє арифметичне між кількістю вхідних та вихідних нейронів.

Отже перший прихований шар містить відповідно:

$$N_{hidden1} = 32 \cdot N_{input} + N_{output} \cdot 32 \cdot 66 + 4 \approx 48 \text{ (нейронів).}$$

Другий прихований шар містить таку кількість нейронів**:**

$$N_{hidden2} = 32 \cdot N_{hidden1} \approx 32 \cdot 48 \approx 32.$$

Отже, архітектура багатошарового перцептрону описується такими параметрами:

– вхідний шар: 66 нейронів;

– прихований шар 1: 48 нейронів;

– прихований шар 2: 32 нейрони;

– вихідний шар: 4 нейрони.

Для прихованих шарів найпоширенішою функцією активації є ReLU (Rectified Linear Unit), яка забезпечує ефективне навчання та уникає проблеми зникнення градієнта, а для вихідних нейронів – гіперболічний тангенс керування кермом і швидкістю (оскільки можуть бути як позитивні, так і негативні значення), а для керування поворотниками застосуємо функцію сигмоїду [3].

Загальну кількість параметрів моделі можна розрахувати як суму вагових коефіцієнтів та зміщень (bias):

– між вхідним шаром і першим прихованим шаром:

$$P_{input\text{-}hidden1} = N_{input} \cdot N_{hidden1} + N_{hidden1} = 66 \cdot 48 + 48 = 3216;$$

– між першим і другим прихованими шарами:

$$P_{hidden1\text{-}hidden2} = N_{hidden1} \cdot N_{hidden2} + N_{hidden2} = 48 \cdot 32 + 32 = 1568;$$

– між другим прихованим шаром і вихідним шаром:

$$P_{hidden2\text{-}output} = N_{hidden2} \cdot N_{output} + N_{output} = 32 \cdot 4 + 4 = 132;$$

– загальна кількість параметрів:

$$P_{total} = 3216 + 1568 + 132 = 4916.$$

Отже, модель багатошарового перцептрону має таку архітектуру, що представлено на рис. 2.

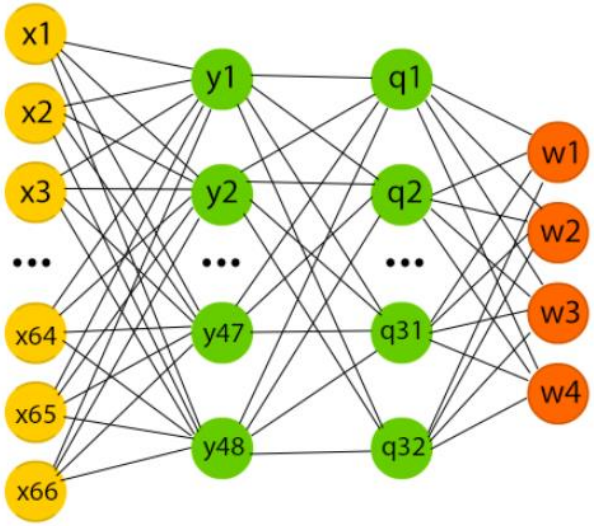

Рис. 2. Багатошаровий перцептрон для ПР щодо керування авомобілем

На рис. 2 позначено нейрони так:
$x_1,\ldots, x_{66}$ – вхідний шар;
$y_1,\ldots, y_{48}$ – прихований шар;
$q_1,\ldots, q_{32}$ – другий прихований шар;
$x_1,\ldots, x_4$ – вихідний шар.

Процес удосконалення навчання багатошарового перцептрону засобами ГА для управління дорожньою ситуацією

Поєднання генетичних алгоритмів з багатошаровим перцептроном дозволяє прискорити процес навчання нейронних мереж, зокрема в складних задачах, де традиційні методи, такі як зворотне поширення помилки (backpropagation), можуть бути повільними або неефективними, наприклад алгоритм зворотного поширення помилки має такі недоліки: потрібна велика кількість обчислень при навчанні, особливо для великих мереж; схильність до потрапляння у локальні мінімуми або плато функції помилок; повільне сходження під час оптимізації складних задач [3].

На думку авторів статті саме генетичні алгоритми є оптимальним методом для прискорення процесу навчання нейронних мереж, в основі якого є такі процедури, як селекція, кросовер і мутація, що імітують природні механізми розвитку популяцій організмів. Генетичні алгоритми можуть використовуватися для пошуку глобальних мінімумів у багатовимірних просторах, що робить їх корисними для задач оптимізації.

Отже, основна задача автопілота – це прийняття рішень на основі отриманих даних. Це полягає у виконанні таких дій, як об'їзд перешкод, уповільнення руху перед пішоходом або перетином перехрестя на зелене світло. Для цих завдань важливі не лише алгоритми розпізнавання об'єктів (YOLO v6), але й рішення, прийняті на основі отриманих даних. Саме тут багатошаровий перцептрон вступає в дію. Він буде використовуватися для прийняття рішень на основі даних, отриманих від YOLO v6. Наприклад, після того, як YOLO v6 виявить перешкоду (інший автомобіль або пішохода), багатошаровий перцептрон може бути використаний для визначення таких дій, як потреба у пригальмовуванні, об'їзді перешкоди або у повній зупинці.

Отже, поєднання генетичного алгоритму та багатошарового перцептрону дозволяє вирішити такі завдання:

– швидке навчання: генетичний алгоритм забезпечує гарну початкову точку для навчання нейронної мережі, що дозволяє зменшити час сходження;

– покращена оптимізація: ГА допомагає уникати проблем з локальними мінімумами, що дозволяє знайти глобально оптимальні параметри для MLP;

– зниження помилок: генетичний алгоритм допомагає зменшити кількість помилок на тестових вибірках, що є важливим для точності автопілота.

Отже, розглянемо етапи поєднання MLP та ГА:

1. Кодування ваг нейронної мережі. Кожна хромосома у популяції відповідає за набір ваг і зміщень MLP. Наприклад, якщо мережа має 100 вагових параметрів, то кожна хромосома матиме довжину 100. Всі ваги мережі зберігаються у векторі, який кодується у вигляді рядка (генів) для подальшої оптимізації ГА.

Кодування ваг нейронної мережі виконується за таким алгоритмом:

— **ініціалізація популяції**. Створюється початкова популяція хромосом, кожна з яких містить випадкові значення ваг і зміщень для MLP. Ці випадкові значення можуть бути отримані за допомогою гаусового розподілу або інших підходів для кращої ініціалізації;

— **оцінка придатності**. Для кожної хромосоми запускається MLP із відповідними вагами. Після запуску на навчальному наборі даних обчислюється значення функції помилок (наприклад, середнє квадратичне помилки або крос-ентропії). Це значення використовують як показник придатності для кожної хромосоми.

— **селекція, кросовер і мутація**. На основі показників придатності відбираються найкращі хромосоми для створення наступного покоління. Під час кросовера об'єднуються частини ваг від двох батьків, створюючи нові рішення. Мутація застосовується до деяких ваг з певною ймовірністю, щоб підтримувати варіативність і уникати локальних мінімумів.

2. Навчання гібридної моделі. Після кількох поколінь, коли генетичний алгоритм знаходить якісну початкову точку для ваг, ці ваги можуть бути передані класичному алгоритму зворотного поширення помилки для точнішого і швидшого навчання. Це комбінує можливості ГА для глобальної оптимізації з локальною оптимізацією, яку забезпечує алгоритм backpropagation.

3. Завершення навчання. Генетичний алгоритм продовжує пошук до тих пір, поки не буде досягнуто прийнятного рівня помилок або інших критеріїв зупинки (наприклад, максимальна кількість поколінь). Після цього можна завершити процес навчання або доопрацювати модель, використовуючи градієнтні методи.

Поєднання генетичних алгоритмів, багатошарового перцептрону (MLP) і архітектури YOLO v6 для створення автопілота є складним, але перспективним підходом. Мета цієї інтеграції полягає в тому, щоб використати переваги кожної технології: генетичні алгоритми (ГА) для оптимізації процесу навчання, багатошаровий перцептрон для прийняття рішень, і YOLO v6 для швидкого та точного виявлення об'єктів.

Розглянемо процес поєднання YOLO v6 та MLP з генетичними алгоритмами для автопілота. Він полягає у виконанні таких етапів.

Перший етап — це використання YOLO v6 для виявлення об'єктів на зображеннях або відео, отриманих з камер автопілота. YOLO v6 забезпечить швидке і точне виявлення об'єктів, таких як пішоходи, автомобілі, дорожні знаки і світлофори.

Другий етап — після виявлення об'єктів багатошаровий перцептрон використовує ці дані для прийняття рішень. Наприклад, якщо YOLO v6 виявить автомобіль попереду на відстані 50 метрів, перцептрон визначить, чи потрібно знизити швидкість або об'їхати цей автомобіль.

Третій етап полягає у застосуванні генетичного алгоритму для оптимізації параметрів багатошарового перцептрону. Наприклад, якщо автопілот працює в складних умовах, генетичний алгоритм може автоматично налаштовувати ваги MLP, щоб він швидше адаптувався до швидкозмінюваної дорожньої обстановки. Це зменшує потребу в довготривалому процесі навчання традиційними методами, такими як градієнтний спуск, який може бути неефективним у великих системах з багатьма параметрами.

Процедура використання такої автоматизованої системи керування автомобілем працює в режимі реального часу. Поєднання YOLO v6 і оптимізованого MLP дозволяє швидко обробляти інформацію та реагувати на різноманітні обставини: від різкої появи пішоходів на дорозі до необхідності перетнути складні перехрестя або об'їхати бар'єри.

Для оцінки прискорення навчання з використанням ГА можна застосувати формулу, яка враховує кількість ітерацій в обох підходах. Формула для оцінки прискорення комбінованого підходу виглядатиме так:

$$T_{BP} = (1 - T_{GA}) * T_{baseline}$$

TGA – час навчання за допомогою генетичного алгоритму для отримання початкових ваг.

TBP – час доопрацювання ваг методом зворотного поширення помилки.

*T*Baseline – час навчання методом зворотного поширення помилки без використання генетичного алгоритму (тобто базовий підхід).

EGA – ефективність генетичного алгоритму в наближенні до оптимальних ваг (від 0 до 1, де 1 означає ідеальні ваги).

Генетичний алгоритм знижує необхідну кількість ітерацій для зворотного поширення помилки залежно від точності початкових ваг. Якщо генетичний алгоритм забезпечує початкові ваги з ефективністю EGA , то необхідний час для доопрацювання методом зворотного поширення помилки зменшується на цей коефіцієнт:

$$T_{BP} = (1 - T_{GA}) * T_{baseline}$$

Підставимо це у формулу для прискорення:

$$S = \frac{T_{baseline}}{T_{GA} + (1 - T_{GA}) * T_{baseline}}.$$

Якщо EGA = 1 (генетичний алгоритм знаходить майже оптимальні ваги), то TBP = 0, і навчання стає майже таким швидким, як TGA

Якщо EGA = 0 (генетичний алгоритм не знаходить корисних ваг), то S = 1і виграш у швидкості зникає.

Нехай:

– TGA=1 година (генетичний алгоритм працює швидко);

– TBaseline=6 годин;

– EGA=0.8 (генетичний алгоритм дає 80% наближення до оптимальних ваг).

$$T_{BP} = (1 - 0.8) * 6 = 1.2 \text{ години}$$
$$S = \frac{6}{1 + 1.2} = \frac{6}{2.2} = 2.73$$

Це означає, що комбінований підхід зробив навчання приблизно у 2.73 рази швидшим.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

Однією з ключових переваг запропонованого авторами підходу є здатність адаптуватися до змін. Мірою того, як програмно апаратний комплекс для забезпечення автоматизованого керування автомобілем збирає більше даних про оточення, генетичний алгоритм може оновлювати ваги нейронної мережі, підвищуючи точність рішень та безпеку системи. Зібрані дані використовуються для подальшого вдосконалення системи на базі нових поколінь ГА.

Дослідження показало, що поєднання генетичних алгоритмів і MLP дозволяє значно прискорити навчання та підвищити точність прийняття рішень у системах автопілотажу.

Використання YOLO v6 для детекції об'єктів у реальному часі забезпечує високу продуктивність і адаптивність системи до змінних умов.

Література

1. Meituan YOLOv6 *Ultralytics YOLO Docs*: веб-сайт. URL: https://docs.ultralytics.com/models/yolov6/ (Дата звернення: 20.11.2024).

2. Neural network zoo *Asimov Institute*: веб-сайт. URL: https://www.asimovinstitute.org/neural-network-zoo/ (Дата звернення: 20.11.2024).

3. Темчур В.С., Баган Т.Г Методи глибокого навчання моделей для прогнозного обслуговування. *Інформатика, обчислювальна техніка та автоматизація.* 2023. URL: https://tech.vernadskyjournals.in.ua/journals/2023/6_2023/23.pdf

References

1. Meituan YOLOv6 *Ultralytics YOLO Docs*: website. URL: https://docs.ultralytics.com/models/yolov6/ (Accessed: 20.11.2024).

2. Neural Network Zoo *Asimov Institute*: website. URL: https://www.asimovinstitute.org/neural-network-zoo/ (Accessed 20.11.2024).

3. Temchur V.S., Bagan T.G. Methods of Deep Learning Models for Predictive Maintenance *Informatics, Computing and Automation.* 2023. URL : https://tech.vernadskyjournals.in.ua/journals/2023/6_2023/23.pdf