ВІННИЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет комп’ютерних систем та автоматики
Кафедра комп’ютерних систем управління
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
Освітньо-професійна програма Інтелектуальні комп’ютерні системи

ЗАТВЕРДЖУЮ

Завідувач кафедри КСУ
Дубовой В.М.

«___» _____ 2019 року

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

Нейромережева система розпізнавання типів автомобілів на зображеннях
08-01.МКР.001.00.000

Студент групи 2АКІТ-18м Босенко Б.М.

Керівник к.т.н., доцент Ковтун .В.В

Рецензент к.т.н., доцент Довгалець С.М.

Вінниця ВНТУ – 2019

ЗМІСТ

АНОТАЦІЯ

Магістерська кваліфікаційна робота присвяча розробці нейромережевої системи для розпізнавання типів автомобілів.

Метою роботи є розробка, проектування і тестування удосконаленої нейронної мережі для полегшення процедури розпізнавання образів типів машин.

Наведені методи і способи фільтрації зображення. Розглянуто реалізації і функції технологій в області комп'ютерного бачення.

Розроблений алгоритм і архітектура програми, визначені основні переваги і недоліки. Реалізована програма загорткової нейромережі з використанням на основі Google Cloud Vision API. Проведено тестування для детальної демонстрації роботи програми, а також порівняння ефективності способу навчання нейронної мережі.

ABSTRACT

The thesis is devoted to the development of a neural network system for recognizing vehicle types.

The purpose of the work is either the development or design or or and testing of either an advanced or a neural network or for the sake of simplification or procedure or recognition or pattern of machine types.

The following are methods and methods of image filtering. Implementations and features of computer vision technologies are discussed.

The algorithm and architecture of the program have been developed, the main advantages and disadvantages have been identified. A neural network wrapper program is implemented using the Google Cloud Vision API. Testing was conducted to demonstrate in detail the operation of the program, as well as to compare the effectiveness of the neural network training method.

ВСТУП

Актуальність теми.Зростаючий інтерес до задач розпізнавання образів обумовлено необхідністю автоматизації, як функцій контролю і управління складними динамічними об'єктами в реальному часі, так і образних процесів комунікації в інтелектуальних системах. Тому досі продовжується пошук і реалізація ефективних принципів передачі розпізнавальної функції людини комп'ютеризованими системами. Один з перспективних напрямків вирішення даної проблеми ґрунтується на застосуванні штучних нейронних мереж і нейрокомп'ютерів, як найбільш адекватних по відношенню до класу задач розпізнавання образів. У наш час запропоновано велику кількість нейромережевих парадигм для вирішення задач розпізнавання образів. Значні важкості при розпізнаванні викликають образи, що були підвернені будь-якому спотворенню (зашумленню, зсуву, повороту, масштабуванню). Цю проблему вирішують шляхом вибору відповідної архітектури та способу навчання. Аналіз робіт показує що досі не існує такої моделі, яка б була не чутлива до всіх видів спотворень. Проблема достатньо добре вирішена відносно зміщених і зашумлених образів нейромережами зворотнього розповсюдження похибки. Однак, як і раніше, викликають труднощі такі види спотворень як зміна образу у розмірах і поворот. Перспективу в подоланні цих труднощів бачать у новій нейромережевій парадигмі - моделлю персептрона,яка використовує якісно нову архітектуру і неконтрольоване навчання. В основу архітектури персептрона покладена організація зорової системи людини.

Тема роботи – розробка нейромережевої системи оптимального розпізнавання типів автомобілів на зображенні.

Метою даної роботи є розробка, проектування і тестування удосконаленої нейронної мережі для полегшення процедури розпізнавання образів.

Методи дослідження. У даній роботі розглянуто різні підходи для вирішення задачі розпізнавання образів. Основну увагу приділено підходу до розпізнавання

образів за допомогою штучних нейронних мереж та методів їх навчання. Зокрема докладно розглянуто таку нейронну мережу як персептрон, наведені переваги і недоліки цієї нейронної мережі. В заключній частині роботи розроблено та запропоновано вдосконалену модель персептрона, після чого наведені результати розробленої комп'ютерної системи для задачі розпізнавання образів, зроблено порівняння з прототипом і вказані переваги та недоліки модифікованої нейронної мережі.

Практична цінність роботи полягає у можливості застосування отриманих результатів для ефективного використання нейронних мереж для задачі розпізнавання образів. Також, слід відмітити, що запропонована модель може бути змінена для досягнення кращих результатів розпізнавання.

Об'єкт дослідження – процес автоматичного розпізнавання об'єктів зображення з використанням нейронних мереж .

Предмет дослідження - методи автоматичного розпізнавання об'єктів зображення нейронною мережою.

Методи дослідження. У дослідженнях використовувались методи:

попередньої обробка зображення, метод адаптивної системи розпізнавання зображення, методи відстеження об'єктів, метод відстеження точок, метод визначення об'єктів.

Наукова новизна отриманих результатів:

1. Вперше запропоновано метод, механізму просторової локалізації у згортковій нейронній мережі, який на відміну від існуючих, відмінно масштабуються і можуть використовуватися для розпізнавання образів, якого завгодно великого масштабу, розроблено механізму просторової локалізації, завдяки чому пришвидшується робота обрахунку алгоритма, забезпечуючи особливі нелінійні фільтри, що дало змогу реагувати на все більше число пік селів, що покращує ефективність виявлення об'єктів зображення.

2. Удосконалено методи розпізнавання об'єктів, який на відміну від існуючих, більш гнучкий в розпізнаванні особливих точок, що дало змогу

підвищити точність та продуктивності системи розпізнавання об'єктів у зображення.

3. Розглянуто реалізації та функції засобів в області комп’ютерного бачення.

4. Як результат, запропонований новій метод – пошук об’єктів зображення у реальному часі, який на відміну від своїх аналогів, має більш гнучку структуру нейронної мережі , що підвищує продуктивності існуючої системи розпізнавання об'єктів у зображення.

Практичне значення одержаних результатів полягає у можливості використання запропонованої інформаційної технології для, відстеження руху об’єкта та оцінка положення, визначення особливих точок, визначення об’єкта.

Найбільшу практичну цінність мають такі одержані результати:

1. Розроблено алгоритм та архітектуру, згорткової нейронної мережі визначення об’єктів.

2. Досліджено сучасні методи та засоби для обробки зображення.

3. Розроблено алгоритм, який визначає руху об’єкта та оцінку положення.

4. Розроблено алгоритм, який визначає особливі точоки у зображенні.

5. Розроблено алгоритм, який визначає об’єкти у зображенні.

6. Проведенно тестування для детальної демонстрації роботи програми.

Апробація результатів роботи. Основні результати наукових досліджень доповідалися на 2 конференціях рівня факультету, зокрема, на таких: XLVII Науково-технічна конференція факультету комп'ютерних систем і автоматики (2018); XLVIII Науково-технічна конференція факультету комп'ютерних систем і автоматики (2019).

Публікації. Результати дисертаційного дослідження опубліковано в 2 наукових працях. Опублікована 1 стаття у науковому фаховому журналі «Вісник Вінницького політехнічного університету»; 1 стаття у виданні із міжнародної наукометричної бази Scopus.

1 АНАЛІЗ МЕТОДІВ РОЗПІЗНАВАННЯ ГРАФІЧНИХ ОБРАЗІВ

1.1 Опис предметної області

Досить довго завдання розпізнавання графічних образів розглядалося людиною з боку біологічних і психологічних аспектів. При цьому, вивченню піддавалися лише якісні характеристики, які не дозволяли точно описати механізм функціонування. Отримання функціональних залежностей було, як правило, пов'язане з дослідженням рецепторів органів слуху, дотику або зору. Однак принципи формування рішення залишалися загадкою. Вважається, що основним помилкою на зорі дослідження була думка про те, що мозок функціонує за певними алгоритмами, а отже, з'ясувавши цю систему правил, можна її відтворити за допомогою обчислювальних і технічних засобів.

Заснована Норбертом Вінером на початку XX століття нова наука, що отримала назву кібернетика (наука про загальні закономірності процесів управління і передачі інформації в машинах, живих організмах і суспільстві), дозволила в дослідження розпізнавання образів ввести кількісні методи. Іншими словами, представити процес розпізнавання образів (по суті - природне явище) математичними методами.

Створення пристроїв, що виконують функції розпізнавання різних об'єктів, в більшості випадків забезпечує можливість заміни людину спеціалізованим автоматом. Завдяки цьому, значно розширюються можливості складних систем, що виконують різні інформаційні, логічні, аналітичні завдання. Слід зазначити, що якість робіт, що виконуються людиною на робочому місці, залежить від багатьох факторів (кваліфікації, досвіду, сумлінності і т. д.). У той же час справний автомат діє одноманітно і забезпечує завжди однакову якість.Автоматичний контроль складних систем дозволяє вести моніторинг і забезпечувати своєчасне обслуговування, ідентифікацію перешкод і автоматичне застосування відповідних методів шумозаглушення, дозволяє підвищити якість передачіінформації. Також

зрозуміло, що використання автоматичних систем у ряді завдань може забезпечити неможливе для людини швидкодію.

Протягом досить тривалого часу проблема розпізнавання графічних образів привертає увагу фахівців в області прикладної математики, а потім і інформатики. Так, зокрема, відзначають роботи Р. Фішера, виконані в 20-х роках і що призвели до формування дискримінантного аналізу як одного з розділів теорії та практики розпізнавання. У 40-х роках А. Н. Колмогоровим і А. Я. Хінчіним поставлена задача про поділ суміші двох розподілів.

У 50-60-ті роки XX століття на основі великої кількості робіт з'явилася теорія статистичних рішень. В результаті цього було відкрито алгоритми, які забезпечують віднесення нового об'єкта до одного з заданих класів, що стало початком планомірного наукового пошуку і практичних розробок. В рамках кібернетики почало формуватися новий науковий напрям, пов'язаний з розробкою теоретичних основ і практичної реалізації пристроїв, а потім і систем, призначених для розпізнавання об'єктів, явищ, процесів. Нова наукова дисципліна отримала назву "Розпізнавання образів".

Таким чином, базою для вирішення завдань віднесення об'єктів до того чи іншого класу послужили, як це відзначається сьогодні, результати класичної теорії статистичних рішень. В її рамках будувалися алгоритми, що забезпечують на основі експериментальних вимірювань параметрів (ознак), що характеризують цей об'єкт, а також деяких апріорних даних, що описують класи, визначення конкретного класу, до якого може бути віднесений об'єкт.

Розпізнавання образів (об'єктів) - задача ідентифікації об'єкта або визначення будь-яких його властивостей по його зображенню (оптичне розпізнавання) або аудіозаписи (акустичне розпізнавання) та інші характеристики (рис 1.1).

Образ - класифікаційне угруповання в системі класифікації, яка об'єднує (виділяє) певну групу об'єктів за певною ознакою. Образи мають характерні властивості, які виявляються в тому, що ознайомлення з кінцевим числом явищ з одного і того ж безлічі дає можливість дізнаватися як завгодно велике число його

представників.

Методика віднесення елемента до якого-небудь образу називається вирішальним правилом. Ще одне важливе поняття - метрика - спосіб визначення відстані між елементами універсальної множини. Чим менший цей період, тим більш схожими є об'єкти (символи, звуки та ін.) нате, що ми розпізнаємо. Зазвичай елементи задаються у вигляді набору чисел, а метрика - у вигляді функції. Від вибору уявлення образів і реалізації метрики залежить ефективність програми, один алгоритм розпізнавання з різними метриками буде помилятися з різною частотою.

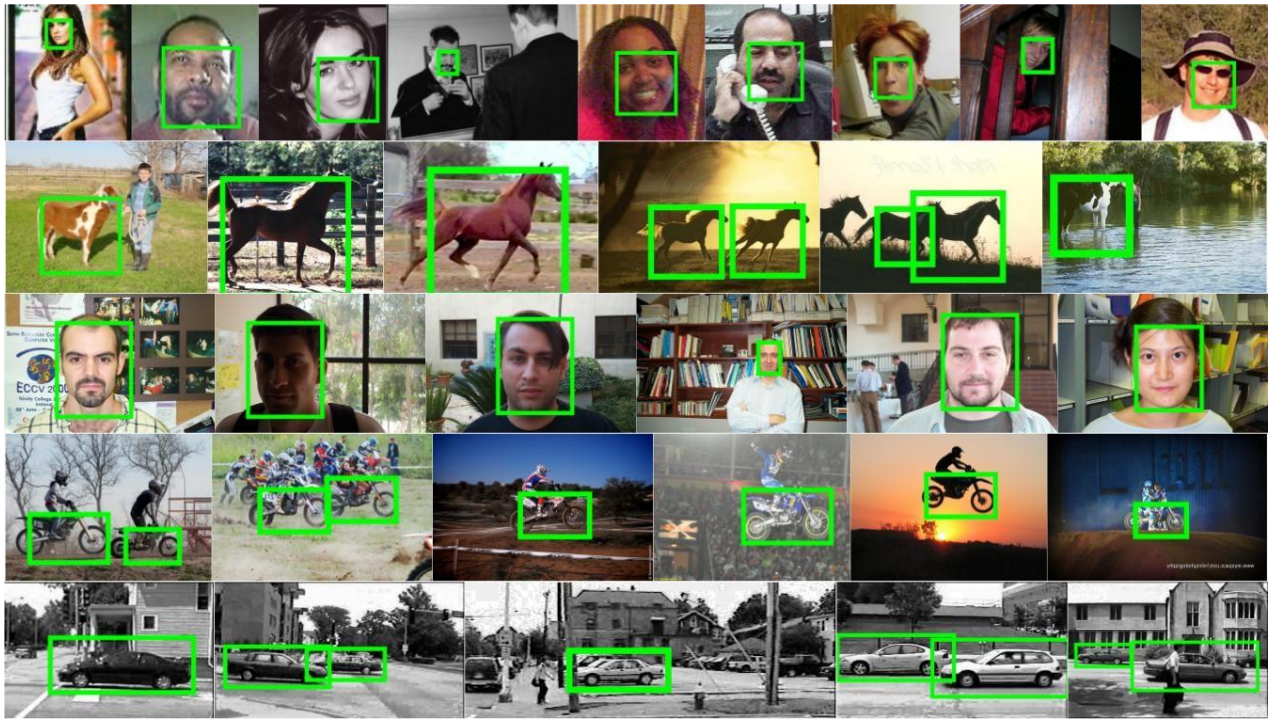


Рисунок 1.1 – Приклад розпізнавання графічних образів

Адаптація - це процес зміни параметрів і структури системи, а можливо - і керуючих впливів, на основі поточної інформації з метоюдосягнення певного стану системи при початковій невизначеності і мінливих умовах роботи.

Навчання - це процес, в результаті якого система поступово набуває здатності відповідати потрібними реакціями на певні сукупності зовнішніх впливів, а адаптація - це підстроювання параметрів і структури системи зметою досягнення необхідної якості управління в умовах безперервних змін зовнішніх умов.

Навчанням зазвичай називають процес вироблення в деякій системі тієї чи іншої реакції на групи зовнішніх ідентичних сигналів шляхом багаторазового впливу на систему зовнішньої коректування. Таку зовнішню коригування в навчанні прийнято називати "заохоченнями" і "покараннями". Механізм генерації цієї коригування практично повністю визначає алгоритм навчання. Самонавчання відрізняється від навчання тим, що тут додаткова інформація про вірність реакції системі не повідомляється.

1.2 Розпізнавання об’єкта методами машинного навчання

Машинне навчання є головною складовою теорії штучного інтелекту, глибокою математичною дисципліною, до якої відносяться науки математична статистика та теорія ймовірностей. Зазначимо два типи машинного навчання, що активно використовуються: індуктивне навчання та дедуктивне навчання. Індуктивне навчання базується на обробці емпіричних даних, а дедуктивне – на формалізації отриманих знань та подальшого їх упорядкування. Прийнято вважати, що область експертних систем включає дедуктивне навчання, тому і використовується в теорії і практиці машинного навчання загалом.

Розділ машинного навчання виник у результаті поділу науки про нейронні мережі в рамках науки про штучний інтелект на методи навчання мереж і види топологій архітектури мереж, увібравши в себе деякі інші області, такі як методи математичної статистики і теорію дискретного аналізу.

1.3 Теорія розпізнавання об’єкта

Теорія розпізнавання образів існує як розділ інформатики та суміжних дисциплін, що розвиває методи класифікації та ідентифікації об'єктів різної природи: сигналів, ситуацій, предметів, що характеризуються вичерпною кількістю деяких ознак . Проблема розпізнавання об’єктів також виділена в розділ

міждисциплінарних досліджень - в тому числі включаючи роботу зі створення штучного інтелекту, а також часто використовується при вирішенні практичних завдань у галузі комп'ютерного зору.

При постановці класичної задачі розпізнавання об’єктів заведено застосовувати математичну мову, ґрунтуючись на логічних міркуваннях і математичних принципах. В протилежність до цього підходу, існують методи розпізнавання об’єктів з використанням машинного навчання і штучних нейронних мереж, сформовані не настільки формалізованих підходах до розпізнавання, і демонструють не гірший, а в деяких випадках і значно кращий результат.

Часто можна зустріти хибне зіствлення термінів “розпізнавання” та “класифікація” де вони розглядаються як синоніми, але не є повністю взаємозамінюваними. Кожен з цих двох термінів може мати свої сфери застосування, в залежності від поставленої задачі. Розглянемо загальні елементи моделі класифікації.

Клас - множина об'єктів, що мають спільні властивості. Для об'єктів одного класу передбачається наявність «схожості». Для задачі розпізнавання може бути визначено довільну кількість класів, більше одного. Кількість класів позначається числом S. Кожен клас має свою ідентифікуючу мітку класу .

Класифікація - процес призначення міток класу об'єктам, відповідно до певного опису властивостей цих об'єктів. Класифікатор – це цнструмент для присвоєння міток класам, який в якості вхідних даних отримує перелік ознак об'єкта. До одного з найпоширеніших способів класифікації можна віднести спосіб, що базується на описі об'єктів з використанням ознак, де кожен об'єкт характеризується набором числових або нечислових ознак. Проте існують типи даних для яких відкриті ознаки не дають високої точності класифікації, наприклад, колір точок зображень або цифровий звуковий сигнал. Загальна класифікація зображень собак і автомобілів є дуже простою для людини і водночас складною для машини. Причиною цього є можливість людини сприймати «приховані ознаки»

недоступні для машини, такі як морда собаки або колеса автомобіля .

Верифікація - процес зіставлення досліджуваного об'єкта із однією моделлю об'єкта або описом класу.

Під образом будемо розуміти найменування області в просторі ознак, в якій відображається безліч об'єктів або явищ матеріального світу.

Ознакою можна назвати опис тієї чи іншої властивості, що має пряме відношення до предмета або явища.

Простір ознак - це N-вимірний простір, визначений для даної задачі розпізнавання, де N - фіксоване число ознак, що були виміряні для будь- яких об'єктів. Вектор з простору ознак x, відповідний об'єкту задачі розпізнавання, це N-вимірний вектор з компонентами (x_1, x_2, ... , x_N), що утворюють значення ознак для даного об'єкта .

Іншими словами, розпізнавання образів можна визначити, як віднесення вихідних даних до певного класу за допомогою виділення істотних ознак або властивостей, які характеризують ці дані, із загальної маси несуттєвих деталей.

Прикладами завдань класифікації є розпізнавання символів, встановлення медичного діагнозу, прогноз погоди, розпізнавання осіб, класифікація документів, тощо.

Найчастіше вихідним матеріалом служить отримане із камери зображення. Постановку завдання можна сформулювати, як одержання векторів, що складаються з ознак для кожного класу на зображенні. Процес можна розглядати як процес кодування, що полягає в присвоєнні значення кожної ознаки із простору ознак для кожного класу.

Якщо розглянути 2 класи об'єктів: легкова машина і грузовик . В якості значення ознак можна вибрати розмір і вагу (Рис. 1.2).

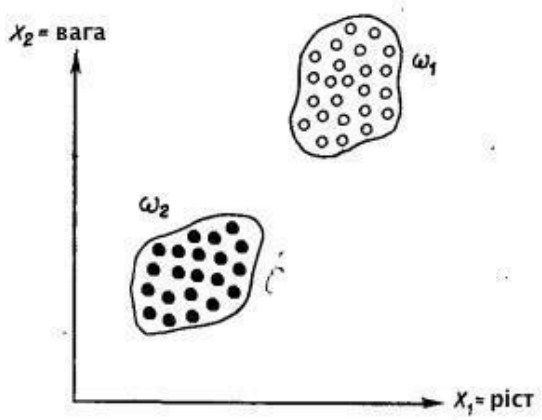


Рис 1.2 - Два непересічних класи

З рис. 1.2 видно, що ці два класи утворюють дві непересічні множини, що можна пояснити обраними ознаками. Проте не завжди є можливість вибрати правильні вимірювані параметри в якості ознак класів. Наприклад, вибрані параметри не підійдуть, щоб створити непересічні класи седана і купе.

Іншим завданням розпізнавання є виділення із вихідних зображень характерних ознак та/або властивостей. Це завдання можна віднести до попередньої обробки. Ознака повинна являти собою характерну властивість конкретного класу, при цьому загальну для цього класу.

Міжкласові ознаки – це ознаки, що визначають відмінності між класами. Загальні ознаки, що властиві усім класам, не несуть корисної інформації, тому для задачі розпізнавання об’єктів не розглядаються як характерні. Вибрати правильні ознаки - одна із важливих задач побудови систем розпізнавання.

1.4 Огляд існуючих методів

Методи розпізнавання графічних образів можна розділити на три групия: попередня фільтрація і підготовка зображення, логічна обробка результатів фільтрації, алгоритми прийняття рішень на основі логічної обробки. Межі між групами умовні. Для вирішення завдання далеко не завжди потрібно застосовувати

методи з усіх груп, буває достатньо двох, а іноді навіть одного.

Фільтрація. До цієї групи відносяться методи, які дозволяють виділити на зображеннях області, без їх аналізу. Велика частина цих методів застосовує єдине перетворення на всі точки зображення. На рівні фільтрації аналіз зображення не проводиться, але точки, які проходять фільтрацію, можна розглядати як області з особливими характеристиками.

Саме простіше перетворення - це бінаризація зображення по порогу. Для RGB зображення і зображення в градаціях сірого порогом є значення кольору. Зустрічаються ідеальні завдання, в яких такого перетворення досить.

Припустимо, потрібно автоматично виділити предмети на білому аркуші паперу: вибір порога, за яким відбувається бінаризація, багато в чому визначає процес самої бінаризації. В даному випадку, зображення було бінаризованим за середнім кольором. Зазвичай бінаризація здійснюється за допомогою алгоритму, який адаптивно вибирає поріг.

Бінаризація може дати дуже цікаві результати при роботі з гістограмами, втому числі в ситуації, якщо ми розглядаємо зображення не в RGB, а в HSV. Наприклад, сегментувати кольори. На цьому принципі можна побудувати як детектор мітки так і детектор шкіри людини.

Класична фільтрація: Фур'є, ФНЧ, ФВЧ. Класичні методи фільтрації з радіолокації і обробки сигналів можна з успіхом застосовувати в безлічізавдань Pattern Recognition.

Традиційним методом в радіолокації, який майже не використовується в зображеннях в чистому вигляді, є перетворення Фур'є (конкретніше - БПФ). Одне з небагатьох винятків, при яких використовується одновимірне перетворення Фур'є, - компресія зображень. Для аналізузображень одновимірного перетворення зазвичай не вистачає, потрібно використовувати куди більш ресурсоємних

$$G_{uw} = \frac{1}{NM} \sum_{n=1}^{N-1} \sum_{m=1}^{M-1} x_{mn} e^{-2\pi j}$$

двовимірне перетворення.

Мало хто його насправді розраховує, зазвичай, куди швидше і простіше використовувати згортку області з уже готовим фільтром, заточеним на високі (ФВЧ) або низькі (ФНЧ) частоти.

Такий метод, звичайно, не дозволяє зробити аналіз спектра, але в конкретному завданні відеоспостереження зазвичай потрібен не аналіз, а результат.

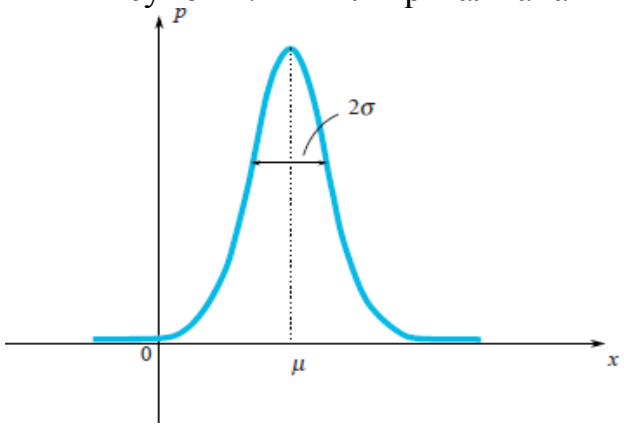
Найпростіші приклади фільтрів, що реалізують підкреслення низьких частот (фільтр Калмана (рис. 1.2)) і високих частот (Фільтр Габора).

Для кожної точки зображення вибирається вікно і перемножується з фільтром того ж розміру. Результатом такої згортки є нове значення точки.

При реалізації ФНЧ і ФВЧ виходять зображення. Але що якщо використовувати для згортки з сигналом якусь довільну характеристическую функцію?

Тоді це буде називатися "Вейвлет- перетворення". Це визначення вейвлетов не є коректним, але традиційно склалося, що в багатьох командах вейвлет-аналізом називається пошук довільного патерну на зображенні за допомогою згортки з моделлю цього патерну. Існує набір класичних функцій, використовуваних в вейвлет-аналізі. До них відносяться вейвлет Хаара, вейвлет Морлі, вейвлет мексиканський капелюх, і.т.д.

Рисунок 1.2 – Фільтр Калмана



Вище наведено 4 приклади класичних вейвлетов. 3х-мірний вейвлет Хаара, 2х-мірні вейвлет Мейєра, вейвлет Мексиканська Капелюх, вейвлет Добеши. Хорошим прикладом використання розширеної трактування вейвлетов є завдання пошуку відблиску в оці, для якої вейвлетом є сам відблиск. Класичні вейвлети зазвичай використовуються для стиснення зображень, або для їх класифікації (буде описано нижче).

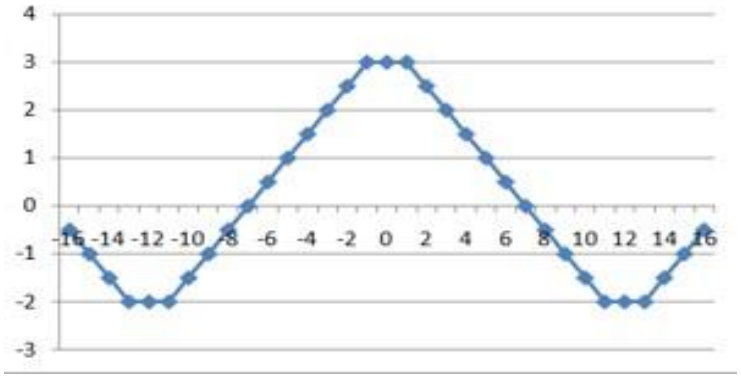
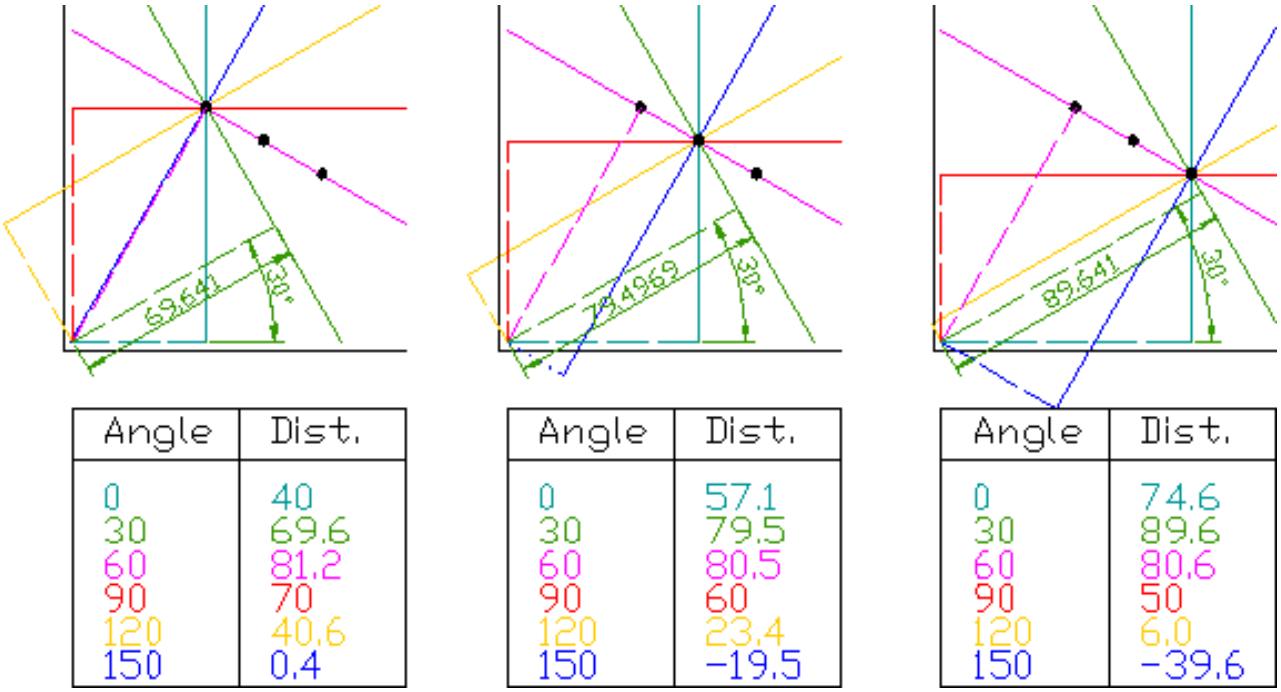


Рисунок 1.3 – Фільтр Гарбора

Після такої вільного трактування вейвлетов з мого боку варто згадати власне кореляцію, що лежить в їх основі. При фільтрації зображень це незамінний інструмент. Класичне застосування - кореляція відеопотоку для знаходження зрушень або оптичних потоків. Найпростіший детектор зсуву - теж в якомусь сенсі різницевий коррелятор. Там де зображення не корелюють - був рух.

Цікавим класом фільтрів є фільтрація функцій. Це чисто математичні фільтри, які дозволяють виявити просту математичну функцію на зображенні (пряму, параболу, коло). Будується зображення, в якому для кожної точки вихідного зображення промальовується безліч функцій, які її породжують. Найбільш класичним перетворенням є перетворення Хафа для прямих (рис. 1.4). У цьому перетворенні для кожної точки (x; y) отрісовивается безліч точок (a; b) прямий y = ax + b, для яких вірно рівність.



Angle	Dist.
0	40
30	69.6
60	81.2
90	70
120	40.6
150	0.4

Angle	Dist.
0	57.1
30	79.5
60	80.5
90	60
120	23.4
150	-19.5

Angle	Dist.
0	74.6
30	89.6
60	80.6
90	50
120	6.0
150	-39.6

Рисунок 1.4 – Перетворення Хафа для прямих

Перетворення Хафа дозволяє знаходити будь-які параметризовані функції. Наприклад окружності. Є модифіковане перетворення, яке дозволяє шукати будь-які фігури. Це перетворення страшенно люблять математики. Але ось при обробці зображень, воно, на жаль, працює далеко не завжди. Дуже повільна швидкість роботи, дуже висока чутливість до якості бінаризації. Навіть в ідеальних ситуаціях я вважав за краще обходитися іншими методами.

Аналогом перетворення Хафа для прямих є перетворення Радону. Воно обчислюється через БПФ, що дає виграш продуктивності в ситуації, коли точок дуже багато. До того ж його можна використовувати до не бінаризованими зображенню.

Окремий клас фільтрів - фільтрація кордонів і контурів. Контури дуже корисні, коли ми хочемо перейти від роботи з зображенням до роботи з об'єктами на цьому зображенні. Коли об'єкт досить складний, але добре виділяється, то часто єдиним способом роботи з ним є виділення його контурів. Існує цілий ряд алгоритмів, які вирішують задачу фільтрації контурів:

Найчастіше використовується саме Кенні, який добре працює. Логічна

обробка результатів фільтрації. Фільтрація дає набір придатних для обробки даних. Але найчастіше можна просто взяти і використовувати ці дані без їх обробки. У цьому розділі буде кілька класичних методів, що дозволяють перейти від зображення до властивостей об'єктів, або до самих об'єктів.

Переходом від фільтрації до логіки, є методи математичної морфології. По суті, це найпростіші операції нарощування і ерозії бінарних зображень. Ці методи дозволяють прибрати шуми з бінарного зображення, збільшивши або зменшивши наявні елементи. На базі математичної морфології існують алгоритми оконтурювання, але зазвичай користуються якимись гібридними алгоритмами або алгоритмами в зв'язці.

У підозділі про фільтрації вже згадувалися алгоритми отримання кордонів. Отримані кордону досить просто перетворюються в контури. Для алгоритму Кенні це відбувається автоматично, для інших алгоритмів потрібна додаткова бінаризація.

Контур є унікальною характеристикою об'єкта. Часто це дозволяє ідентифікувати об'єкт по контуру. Існує потужний математичний апарат, що дозволяє це зробити. Апарат називається контурним аналізом.

Особливі точки це унікальні характеристики об'єкта, які дозволяють зіставляти об'єкт сам з собою або зі схожими класами об'єктів. Існує кілька десятків способів дозволяють виділити такі точки. Деякі способи виділяють особливі точки в сусідніх кадрах, деякі через великий проміжок часу і при зміні освітлення, деякі дозволяють знайти особливі точки, які залишаються такими навіть при поворотах об'єкта. Почнемо з методів, що дозволяють знайти особливі точки, що не такі стабільні, зате швидко розраховуються, а потім підемо по зростанню складності:

Перший клас. Особливі точки, які є стабільними протягом секунд. Такі точки служать для того, щоб вести об'єкт між сусідніми кадрами відео, або для відомості зображення з сусідніх камер. До таких точок можна віднести локальні максимуми зображення, кути на зображенні (рис. 1.5), точки в яких досягається максимуми дисперсії, певні градієнти і.т.д.

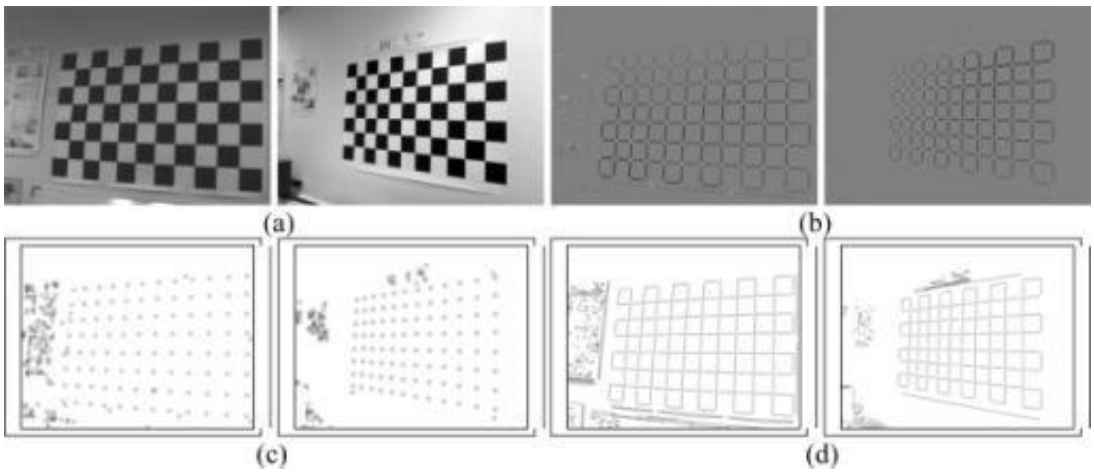


Рисунок 1.5 – Розпізнавання кутів на зображеннях

Другий клас. Особливі точки, які є стабільними при зміні освітлення і невеликих рухах об'єкта. Такі точки служать в першу чергу для навчання і подальшої класифікації типів об'єктів. Наприклад, класифікатор пішохода або класифікатор особи - це продукт системи, побудованої саме на таких точках. Деякі з раніше згаданих вейвлетов можуть є базою для таких точок. Наприклад, примітиви Хаара, пошук відблисків, пошук інших специфічних функцій. До

таких точок відносяться точки, знайдені методом гістограм спрямованих градієнтів (рис 1.6).

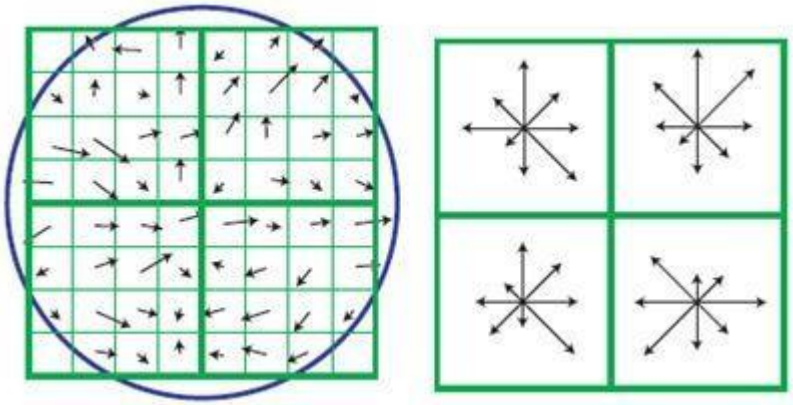


Рисунок 1.6 – Гістограм спрямованих градієнтів

Третій клас. Стабільні точки. Мені відомо лише про два методи, які дають повну стабільність і про їх модифікації. Це SURF і SIFT. Вони дозволяють

знаходити особливі точки навіть при повороті зображення. Розрахунок таких точок здійснюється довше в порівнянні з іншими методами, але досить обмежений час. На жаль ці методи запатентовані.

Навчання.Третя частина буде присвячена методам, які не працюють безпосередньо з зображенням, але які дозволяють приймати рішення. В основному це різні методи машинного навчання і прийняття рішень. Нещодавно Яндикс виклав на Хабре курс з цієї тематики, там дуже хороша підбірка. Ось тут воно є в текстовій версії. Для серйозного заняття тематикою настійно рекомендую подивитися саме їх. Тут я спробую окреслити кілька основних методів використовуваних саме в розпізнаванні образів.Є тестова вибірка, на якій є кілька класів об'єктів. Нехай це буде наявність / відсутність людини на фотографії. Для кожного зображення є набір ознак, які були виділені яким-небудь ознакою, будь то Хара, HOG, SURF або який-небудь вейвлет. Алгоритм навчання повинен побудувати таку модель, за якою він зможе проаналізувати нове зображення і прийняти рішення, який з об'єктів є на зображенні.

Як це робиться? Кожне з тестових зображень - це точка в просторі ознак. Її координати це вага кожного з ознак на зображенні. Нехай нашими ознаками будуть: «Наявність колес», «Наявність корпусу машини», «Наявність прецепа », і.т.д ... Всі ці ознаки ми виділимо існуючими у нас детекторами, які навчені на частинах машин. Для легкової в такому просторі буде коректною точка [1; 1; 1; 1; ..]. Для грузової [1; 0; 1; 0 ...].

Класифікатор навчається за вибіркою прикладів. Той, якого навчають класифікатор автомобілів автоматично розбиває простір ознак таким чином, щоб сказати: якщо перша ознака лежить в діапазоні 0.5 <x <1, другий 0.7 <y <1, і.т.д., тоді це легковий автомобіль. По суті мета класифікатора - відрисувати в просторі ознак області, характеристичні для об'єктів класифікації. Ось так буде виглядати послідовне наближення до відповіді для одного з класифікаторів (AdaBoost)в 2д.

1.5 Побудова нейронної мережі

Не менш важливим є завдання побудова мережі. Це питання вирішується в два етапи:

- вибір типу (архітектури) мережі;
- підбір ваг (навчання) мережі.

На першому етапі слід вирішити наступні питання:

- які нейрони ми хочемо використовувати (число входів, передавальні функції);
- яким чином слід з'єднати їх між собою;
- що взяти в якості входів і виходів мережі.

Це завдання на перший погляд здається складним, але необов’язково придумувати нейромережу – існує кілька десятків різних нейромережевих архітектур, причому ефективність багатьох з них доведена математичною статистикою. Найбільш популярні і вивчені архітектури – це багатошарови

персептрон, нейромережа із загальною регресією, мережі Кохонена, мережі Хопфілда, мережі Хеммінга та інші .

На другому етапі слід навчити обрану мережу, тобто підібрати такі значення її ваг, щоб мережа працювала належним чином. У використовуваних на практиці нейромережах кількість ваг може становити кілька десятків тисяч, тому навчання це дійсно складний процес. Для багатьох архітектур розроблені спеціальні алгоритми навчання, які дозволяють налаштувати ваги мережі певним чином.

У залежності від функцій, виконуваних нейронами в мережі, можна виділити три їх типи:

- вхідні нейрони – це нейрони, на які подається вхідний вектор, що кодує вхідний вплив чи образ зовнішнього середовища; в них зазвичай не здійснюється обчислювальні процедури, інформація передається з входу на вихід нейрона шляхом трансформаційних змін його активації;
- вихідні нейрони – це нейрони, вихідні значення яких представляють вихід мережі;
- проміжні нейрони – ці нейрони складають основу штучних нейронних

мереж.

У більшості нейронних моделей тип нейрона пов'язаний з його розташуванням у мережі. Якщо нейрон має тільки вихідні зв'язки, то це вхідний нейрон, якщо навпаки – вихідний нейрон. Однак може зустрітися випадок, коли вихід внутрішнього нейрона розглядається як частина виходу мережі. У процесі функціонування мережі здійснюється перетворення вхідного вектора у вихідний.

Сукупність штучних нейронів, суматора та порогового елемента називається штучною нейронною мережею (рис 2).

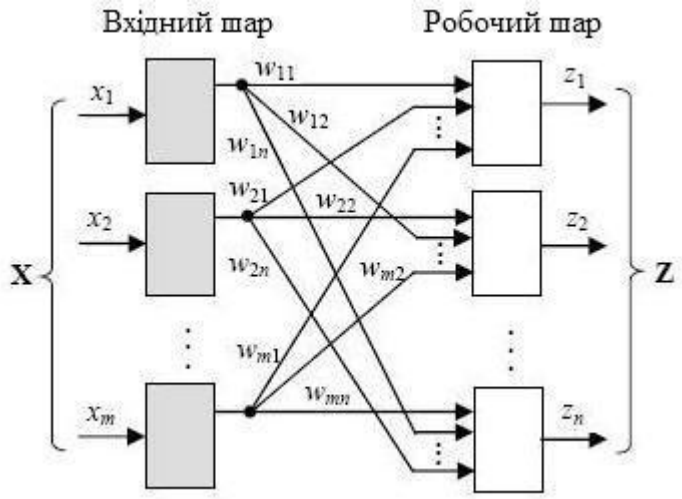


Рисунок 2 – Модель одношарової штучної нейронної мережі

Сигнал надходить на вхідний шар рецепторних нейронів, кожен з яких пов'язаний з усіма елементами вихідного шару з певними значеннями вагових коефіцієнтів w_{ij}. Їх підсумовування призводить до порушення тих нейронів робочого шару, значення активаційної функції яких перевищила порогове значення.

Кілька основних нейромережевих архітектур, такі, як багатошарові персептрони, мережі Хопфілда і карти Кохонена, роблять можливим вирішення широкого спектру завдань, найчастіше нерозв'язних класичними статистичними методами обробки даних.

Серед переваг нейронних мереж можна виділити такі:

- навчання на наборі прикладів;
- побудова нелінійної регресійної залежності або нелінійної роздільної поверхні без апріорного завдання виду нелінійної функції з точністю до значень параметрів;
- можливість рішення одночасно кількох завдань прогнозування чи класифікації однією нейронною моделлю з векторним виходом;
- цільова функція, оптимізується при навчанні нейронної мережі, не обмежена звичайним методом найменших квадратів (МНК) і може бути

робастною до викидів в даних, а також може включати в себе додаткові складові, наприклад, регулюючі рішення;

- побудова нелінійних головних компонент нейронною мережею з «вузьким горлом»
- при недостатності лінійних головних компонент для опису даних з потрібною точністю для їх подальшої візуалізації в просторі.

В літературі зустрічається значна кількість ознак, які повинна мати задача, щоб можна було ефективно застосувати ШНМ, наприклад: відсутність алгоритму або не відомі принципи вирішення завдань, але накопичено достатню кількість прикладів; проблема характеризується великими обсягами вхідної інформації; дані неповні, або надлишкові чи частково суперечливі.

ШНМ може розглядатися як спрямований граф зі зваженими зв'язками, в якому штучні нейрони є вузлами. За архітектурою зв'язків ШНМ можуть бути згруповані в два класи, в яких графи не мають петель, і рекурентні мережі, або мережі зі зворотними зв'язками.

Систематизація архітектур мереж прямого поширення та рекурентних (зі зворотним зв'язком).

У найбільш поширеному типі мереж першого класу, наприклад, багатошаровому персептроні, нейрони розташовані шарами і мають односпрямовані зв'язку між шарами.

Нейронні мережі можна розподілити на : а) мережі прямого розповсюдження:

1) одношаровий персептрон;
2) багатошаровий персептрон;
3) мережа радіальних базисних функцій.

б) рекурентні мережі (зі зворотним зв'язком) :

1) змагальні мережі;
2) мережа Хопфілда;
3) мережа Кохонена;
4) моделі ART (Adaptive Resonance Theory Network).

Мережі прямого поширення є статичними в тому сенсі, що на вхід вони поставляють одну сукупність вихідних значень, що не залежать від попереднього стану мережі. Рекурентні мережі є динамічними, тому що в силу зворотних зв'язків у них модифікуються входи нейронів, що призводить до зміни стану мережі.

1.6 Навчання штучних нейронних мереж.

У ШНМ процес навчання може розглядатися як налаштування архітектури мережі і ваг зв'язків для ефективного виконання різних типів задач.

Нейронна мережа повинна налаштувати ваги зв'язків по наявній навчальній вибірці. Функціонування мережі поліпшується у міру ітеративного налаштування вагових коефіцієнтів. Властивість мережі навчатися на прикладах робить їх більш привабливими в порівнянні з системами, які слідують певній системі правил функціонування, сформульованих експертами.

Для конструювання процесу навчання, перш за все, необхідно мати модель зовнішнього середовища, в якій функціонує нейронна мережа – знати доступну для мережі інформацію. Також необхідно визначити, як модифікувати вагові параметри мережі – які правила навчання управляють процесом налаштування. Алгоритм навчання означає процедуру, в якій використовуються правила навчання для налаштування ваг.

Найважливішою властивістю нейронних мереж є їх здатність навчатися на

основі даних навколишнього середовища і в результаті навчання підвищувати свою продуктивність. Підвищення продуктивності відбувається з часом у відповідності з певними правилами. Навчання нейронної мережі відбувається за допомогою інтерактивного процесу корегування синаптичних ваг і порогів. В ідеальному випадку нейронна мережа отримує знання про навколишнє середовище на кожній ітерації процесу навчання.

З поняттям навчання асоціюється досить багато видів діяльності, тому складно дати цьому процесу однозначне визначення. Більше того, процес

навчання залежить від точки зору на нього. Саме це робить практично неможливим появу будь-якого точного визначення цього поняття. Наприклад, процес навчання з точки зору психолога в корені відрізняється від навчання з точки зору шкільного вчителя. З позицій нейронної мережі, ймовірно, можна використовувати наступне визначення:

Навчання – це процес, у якому вільні параметри нейронної мережі настроюються за допомогою моделювання середовища, у яке ця мережа вбудована.

Тип навчання визначається способом підстроювання цих параметрів.

Це визначення процесу навчання нейронної мережі передбачає наступну послідовність подій:

1) у нейронну мережу надходять стимули із зовнішнього середовища;

2) у результаті реалізації першого пункту змінюються вільні параметри нейронної мережі;

3) після зміни внутрішньої структури нейронна мережа відповідає на штрафи вже іншим чином.

Вищевказаний список чітких правилвирішення проблеми навчання нейронної мережі називається алгоритмом навчання. Не існує універсального алгоритму навчання, відповідного для всіх архітектур нейронних мереж. Існує лише набір засобів, представлений безліччю алгоритмів навчання, кожен з яких має свої переваги. Алгоритми навчання відрізняються один від одного способом налаштування синаптичних ваг нейронів. Ще однією відмінною

характеристикою є спосіб зв'язку навченою нейронної мережі із зовнішнім світом. У цьому контексті говорять про парадигму навчання, пов'язану з моделлю навколишнього середовища, в якій функціонує дана нейронна мережа. Існують три способу навчання: з вчителем; без вчителя; змішана. Навчання нейронної мережі з учителем припускає, що для кожного вхідного вектора з навчальної множини існує необхідне значення вихідного вектора, званого цільовим. Ці вектора утворюють навчальну пару. Ваги мережі змінюють до тих пір, поки для кожного вхідного вектора не буде отриманий прийнятний рівень

відхилення вихідного вектора від цільового. Нейронна мережа має у своєму розпорядженні правильними відповідями (виходами мережі) на кожен вхідний приклад. Ваги налаштовуються так, щоб мережа виробляла відповіді, як можна більш близькі до відомих правильних відповідей. Посилений варіант навчання з учителем припускає, що відома тільки критична оцінка правильності виходу нейронної мережі, але не самі правильні значення виходу.

Навчання нейронної мережі без вчителя є набагато більш правдоподібною моделлю навчання з точки зору біологічних коренів штучних нейронних мереж. Навчальна множина складається лише з вхідних векторів. Алгоритм навчання нейронної мережі підлаштовує ваги мережі так, щоб виходили узгоджені вихідні вектори, тобто щоб пред'явлення досить близьких вхідних векторів давало однакові виходи. Навчання без вчителя не вимагає знання правильних відповідей на кожний приклад навчальної вибірки. У цьому випадку розкривається внутрішня структура даних, або кореляції між зразками в системі даних, що дозволяє розподілити зразки за категоріями. При змішаному навчанні частина ваг визначається за допомогою навчання з учителем, у той час як інша виходить за допомогою самонавчання.

Різні алгоритми навчання та пов'язані з ними архітектури мереж представлені у табл. 1. В останній колонці перераховані задачі, для яких можуть бути застосовані різні алгоритми. Кожен з алгоритмів навчання орієнтований на мережу певної архітектури і призначений для обмеженого класу задач.

Класифікація образів. Завдання полягає у вказівці приналежності вхідного образу (наприклад, «car» або «інше»), представленого вектором ознак, одному або декільком попередньо визначеним класам. Наприклад розпізнавання букв, розпізнавання машин та розподіл їх на класи.

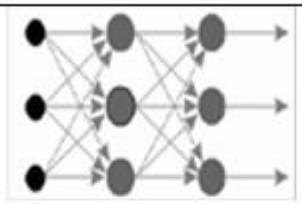
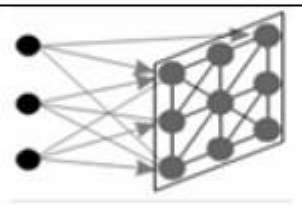
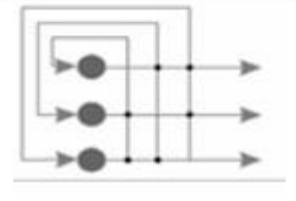
Таблиця 1 — Відомі алгоритми навчання нейронних мереж

Типи навчання	Правило навчання	Архітектура	Алгоритм навчання	Задача
З вчителем	Корекція помилки	Одношаровий і багатошаровий перцептрон	Алгоритми навчання перцептрона Зворотне поширення	Класифікація образів Аппроксимація функцій Передбачення, управління
	Больцман	Рекурентна	Алгоритм навчання Больцмана	Класифікація образів
	Хебб	Багатошарова прямого розповсюдження	Лінійний дискримінантний аналізанализ	Аналіз даних Класифікація образів
	Змагальні	Змагальння	Векторне квантування	Категоризація всередині класу Стиснення даних
		Мережа ART	ARTMap	Класифікація образів
Без вчителя	Корекція помилки	Багатошарова прямого розповсюдження	Проекція Саммона	Категоризація всередині класу Аналіз даних
	Хебб	Прямого поширення або змагання	Аналіз головних компонентів	Аналіз даних Стиснення даних
		Мережа Хопфілда	Навчання асоціативної пам'яті	Асоціативна пам'ять
	Змагальні	Змагальння	Векторне квантування	Категоризація Стиснення даних
		Мережа Кохонена	SOM Кохонена	Категоризація Аналіз даних
		Мережа ART	ART1, ART2	Категоризація
Змішані	Коррекция ошибки и соревнование	Мережа RBF	Алгоритм навчання RBF	Класифікація образів Апроксимація функцій Передбачення, управління

ШНМ добре підходять для розпізнавання образів і вирішення задач класифікації, оптимізації і прогнозування. Представимо деякі проблеми, які вирішуються з використанням нейронних мереж, які представляють інтерес для користувачів .

Зростання обсягів баз даних в техніці, бізнесі, медицині, екології і зростання вимог до точності рішення ставлять нові вимоги перед нейронними мережами. В табл. 3 приведені типи ШНМ та класи задач, які вони вирішують.

Таблиця 2 – Типи штучних мереж

Тип ШНМ	Архітектура ШНМ	Клас задач які вирішує ШНМ
Багатошаровий персептрон		Апроксимація функцій, класифікація.
Самоорганізаційна карта Кохонена		Стиснення даних, виділення при знаків.
Мережа Хопфілда		Асоціативна пам'ять, кластеризація даних.

Кластеризація (категоризація). При вирішенні задачі кластеризації, яка відома також як класифікація образів «без вчителя», відсутня навчальна вибірка з мітками класів. Алгоритм кластеризації заснований на подобі образів і розміщує близькі образи в один кластер. Відомі випадки застосування кластеризації для стиснення даних і дослідження властивостей даних.

Апроксимація функцій. Припустимо, що є навчальна вибірка (пари даних

вхід-вихід), яка генерується невідомою функцією F(x), спотвореної шумом. Завдання апроксимації полягає в знаходженні оцінки невідомої функції F(x). Апроксимація функцій необхідна при рішенні численних інженерних і наукових задач моделювання.

Прогнозування. Завдання полягає в передбаченні деякого значення у деякий майбутній момент часу. Передбачення мають значний вплив на прийняття рішень у бізнесі, науці і техніці. Передбачення цін на фондовій біржі і прогноз погоди є типовими програмами прогнозування.

Оптимізація. Численні проблеми в математиці, статистиці, техніці, науці, медицині та економіці можуть розглядатися як проблеми оптимізації. Завданням алгоритму оптимізації є знаходження такого рішення, яке задовольнить обмеженням системи і максимізує чи мінімізує цільову функцію. Задача комівояжера є класичним прикладом задачі оптимізації.

Пам'ять, що адресується за змістом. У моделі обчислень фон Неймана звертання до пам'яті доступно тільки за допомогою адресу, який не залежить від утримання пам'яті. Більш того, якщо допущена помилка в обчисленні адреси, то може бути знайдена зовсім інша інформація. Асоціативна пам'ять, чи пам'ять, що адресується за змістом, доступна за вказівкою заданого змісту. Зміст пам'яті може бути викликаний навіть по частковому входу. Асоціативна пам'ять надзвичайно ефективна при створенні мультимедійних інформаційних баз даних.

Програмні продукти на базі нейронних мереж використовують для контролю якості води та можуть знаходити пластикові бомби в багажі авіапасажирів. Фахівці інвестиційного банку за допомогою програмного нейропакета роблять короткострокові прогнози коливань курсів валют.

Проведення аналізу вирішення завдань на нейронних мережах дозволяє виділити основні переваги їх використання:

Рішення завдань при невідомих закономірностях. Використовуючи здатність навчання на безлічі прикладів, нейронна мережа здатна вирішувати завдання, в яких невідомі закономірності розвитку ситуації і залежності між вхідними і

вихідними даними. Традиційні математичні методи та експертні системи в таких випадках не використовують.

Стійкість до шумів у вхідних даних. Можливість роботи при наявності великого числа неінформативних, шумових вхідних сигналів. Немає необхідності робити їх попередній відсів, нейронна мережа сама визначить їх мало придатне для вирішення задачі та відкине їх.

Адаптація до змін навколишнього середовища. Нейронні мережі мають здатність адаптуватися до змін навколишнього середовища. Зокрема, нейронні мережі, навчені діяти в певному середовищі, можуть бути легко перевчити для роботи в умовах незначних коливань параметрів середовища. Більш того, для роботи в нестаціонарної середовищі (де статистика змінюється з плином часу, наприклад як котирування акцій на біржі) можуть бути створені нейронні мережі, вони перенавчаються в реальному часі.

Чим вище адаптивні здатності системи, тим більш стійкою буде її робота в нестаціонарної середовищі. При цьому слід зауважити, що адаптивність не завжди веде до стійкості; іноді вона призводить до зовсім протилежного результату. Наприклад, адаптивна системаз параметрами, що швидко змінюються в часі, може також швидко реагувати на сторонні порушення, що викличе втрату продуктивності.

Для того, щоб використовувати всі переваги адаптивності, основні параметри системи повинні бути досить стабільними, щоб можна було не враховувати зовнішні перешкоди, досить гнучкими, щоб забезпечити реакцію на істотні зміни середовища.

Нейронні мережі мають потенційну надвисоку швидкодію за рахунок використання масового паралелізму обробки інформації.

Нейронні мережі потенційно відмовостійкі при апаратній реалізації. Це означає, що за несприятливих умов їх продуктивність падає незначно.

Наприклад, якщо пошкоджений якийсь нейрон або його зв'язки, вилучення запам'ятовані інформації ускладнюється. Проте, враховуючи розподілений

характер зберігання інформації в нейронній мережі, можна стверджувати, що тільки серйозні пошкодження структури нейронної мережі суттєво вплинуть на її працездатність. Тому зниження якості роботи нейронної мережі відбувається повільно.

2. ЗГОРТКОВА НЕЙРОННА МЕРЕЖА

Архітектура згорткових нейронних мереж призначена і використовується для ефективного розпізнавання зображень, де чергуються згорткові шари (англ. convolutions) з нелінійними функціями активації (ReLU або гіперболічний тангенс tanh) і шари об'єднання або підвибірки (pooling layers).

На відміну від мережі прямого поширення, де кожен вхідний нейронз'єднується з вихідним нейроном в наступномушарі,в згортковихмережах для отримання вихіднихзначень застосовуються згорткинад кожним вхідним шаром.В операції згортки використовується матриця ваг невеликого розміру, яка проходить по всьому поточному шарі, формуючи після кожного зсуву сигнал активації для нейрона наступного шару з аналогічною позицією. Ця матриця називається ядром згортки; вона використовується для різних нейронів вихідного шару. Детально про згортковий шар описано у пункті 2.2 «Шари згортки» цієї роботи.

2.1 Шаблон архітектури згорткової нейронної мережі

Як відомо, багатошарові штучні нейронні мережі отримують вхідні дані (наприклад один вектор), після чого трансформують інформацію, проводячи її через ряд прихованих шарів. Кожен прихований шар складається з безлічі нейронів,декожнийнейронмаєсильнийзв'язокзусіманейронамив попередньомушарі і де нейрони в якості одного шару повністю незалежні один від одного і не мають спільних з'єднань. Останній повнозв'язну шар називається вихідним шаром, і в налаштуванях класифікації він демонструє число класів .

Звичайні штучні нейронні мережі (Рис. 2.1) погано масштабуються у випадку з зображеннями великих розмірів. Так, в системі комп'ютерного зору CIFAR-10, картинки становить [32 × 32 × 3] (32 - ширина, 32 - висота, 3 - канали кольорів), тому один повністю підключений нейрон в першому прихованому шарі звичайної

нейронної мережі має вагу 32 * 32 * 3 = 3 072. Здається, що це значення можна змінювати, але повнозв’язна структура не масштабується для великих зображень. Картинка обсягом більше, наприклад, [200 × 200 × 3], приведе до того, що повністю підключений нейрон буде важити 120 000. Крім того, потрібно залучити кілька таких нейронів, що призведе до додавання параметрів. Недоліком повнозв’язності буде величезна кількість параметрів, що швидко привести до перенавчання.

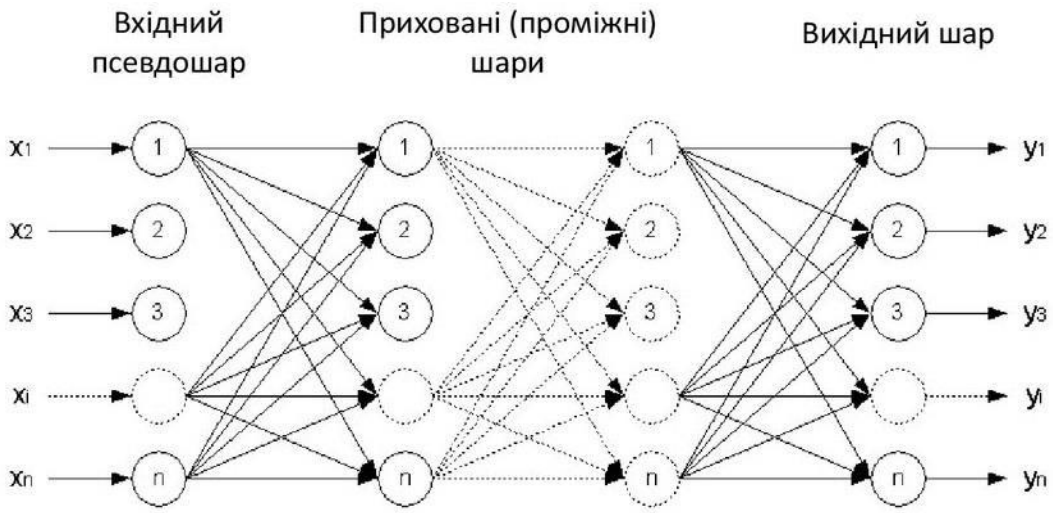


Рис. 2.1 - Багатошарова нейронна марежа

Згорткові нейронні мережі (Рис 2.2) користуються тим, що ввідні дані складаються з зображень, і вони обмежують побудову мережі більш розумним шляхом. На відміну від звичайної нейронної мережі, шари ЗНМ складаються з нейронів, розташованих в 3-х вимірах: ширині, висоті і глибині, тобто у вимірах, які формують об’єм.Зображення на вході є вхідними активаційними об’ємами, а об’єм сформований вимірами 32 × 32 × 3. Як буде описано далі, нейрони будуть підключені тільки до невеликої області шару. Крім того, результуючий вихідний шар для даної системи комп'ютерного зору складе 1x1x10, оскільки до кінця побудови ЗНМ зображення перетвориться в єдиний вектор оцінок класу, розташованих по вимірюванню глибини .

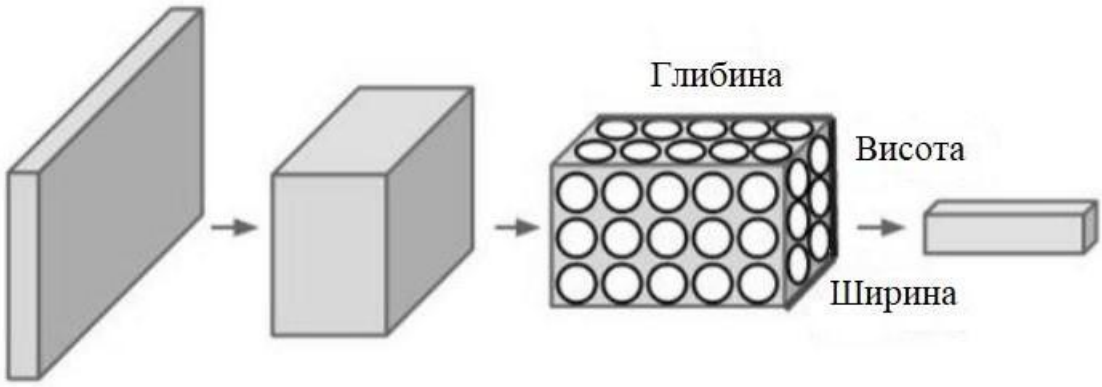


Рис. 2.2 Згорткова нейронна мережа

Отже основу згорткових нейромереж складають шари. Кожен шар характеризується простим набором задач: він перетворює вхідні дані у вигляді 3D-об'єму у вихідний 3D-об'єм з деякою диференційованою функцією, яка може мати або не мати параметри.

2.2 Шари згортки

Головним шаром згорткової нейронної мережі беззаперечно можна вважати шар згортки, робота якого є основою даної мережі. Параметри шару згортки складаються з набору фільтрів для навчання (також можна зустріти назву - ядра Кернела). Кожен фільтр має невеликі просторові габарити (ширину і висоту), але проходить по всій глибині вхідного об’єму. Наприклад, стандартний фільтр першого шару згорткової нейронної мережі може бути розміром [5x5x3]. Під час проходу вперед, виконується ніби ковзання кожного фільтру по ширині і висоті вхідних даних і обчислюється скалярний добуток між записами фільтра і входом у будь-яке положення. У міру проходження фільтра по ширині і висоті зображення, складається двомірну активаційна карта, яка надає відгук цього фільтра на кожній просторовій позиції (Рис. 2.3). Мережа навчає фільтри, що активуються при виявленні певної візуальної особливості. Це може бути грань деякої спрямованості, плямистість конкретного кольору на першому шарі кільцеподібні візерунки на більш високих рівнях мережі. Після мережа працює з цілим набором фільтрів в

кожному шарі згортки, і кожен з цих фільтрів буде формувати окрему двомірну карту ативації або карту ознак. Карти ознак складаються вздовж «глибини» вхідного об’єму і будуть формувати вихідний об’єм.

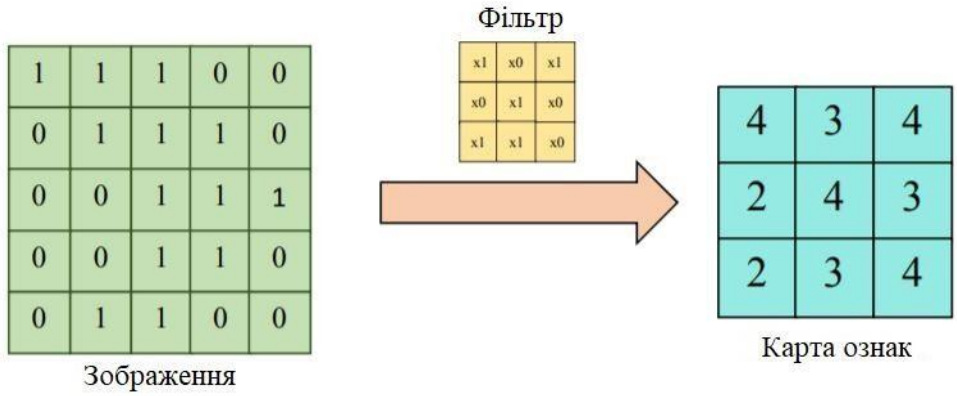


Рис 2.3 - Формування карти ознак. Операція згортки.

При роботі з вхідною інформацією високою розмірністю, установка зв'язку між нейронами і всіма нейронами з попереднім об’ємом є недоцільною. Замість цього потрібно застосовувати кожен нейрон тільки до локальної області вхідного об’єму. Просторова протяжність його зв'язку є гіперпараметром і називається рецептивним полем (полем сприйнятливості). Ідея рецептивного поля полягає в тому, щоб об'єднати нейрони прихованого шару лише з такими нейронами попереднього шару, які входять в деяку маленьку область n × n. Причому для кожного нейрона вибирається своя окрема область. Дана область і називається локальним рецептивним полем. Рецептивні поля виявляють елементарні візуальні ознаки такі, як кут або край, а їх комбінації дають можливість отримати складніші ознаки .

Легко здогадатися, що площа рецептивного поля дорівнює площі фільтра. Просторова протяжність уздовж осі глибини завжди еквівалента глибині вхідного об’єму. Слід ще раз підкреслити асиметрію в тому, як ми розглядаємо просторові виміри (ширину і висоту), а як - глибину: з’єднання є локальними по ширині і висоті, але завжди проходять через всю глибину вхідного об’єму.

Приклад вхідного об’єму прямокутник, параметри [32 × 32 × 3] зображення і можливий обсяг нейронів в першому шарі згорткової нейронної мережі показано на Рис 2.4.

Кожен нейрон в згортковому шарі з'єднаний тільки з локальною вхідною областю, де

просторова протяжність визначається шириною і висотою, але нейрон проходить через всю глибину, охоплює всі кольорові канали.

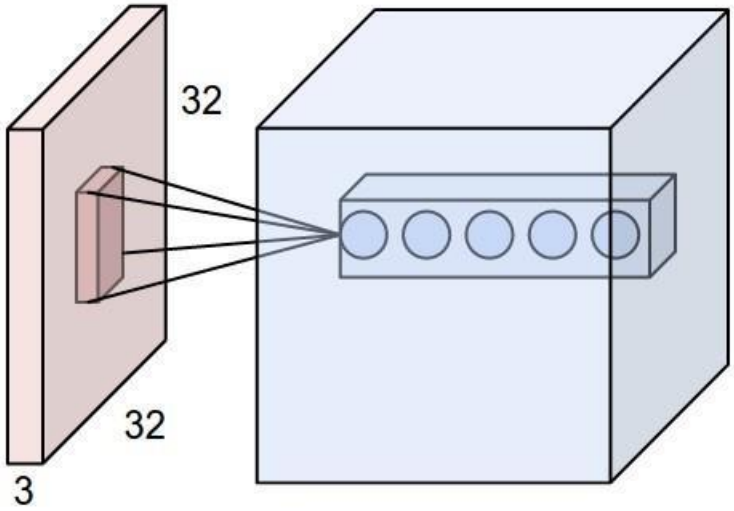


Рис 2.4 - З'єднання нейронів шару згортки з локальним вхідним об’ємом (частиною вхідного зображення)

Нейрони згорткових нейромереж залишаються незмінними: вони як і звичайні нейрони мереж обчислюють скалярний добутог їх ваги і вхідних даних з подальшою нелінійністю, але їх підключення обмежується локальними просторовими вимірами.

2.3 Параметри та гіперпараметри мережі

Існують 3 гіперпараметри мережі, які контролюють розмір вихідної інформації: глибина, крок і доповнення нулями .

Першим гіперпараметром вихідного об’єму є глибина. Гіперпараметр

еквівалентний числу фільтрів, які використовуються; кожен з фільтрів навчається пошуку різних ознак на вході. Наприклад, перший згортковий шар в якості вхідної інформації бере необроблене зображення, де різні нейрони, розташовані уздовж глибини, можуть активуватися в разі наявності, наприклад, згустків певного кольору. Позначимо безліч нейронів, які «дивляться» на одну ділянку вхідного об’єму як стовпець глибини.

Другим важливим гіперпараметром є крок, з яким фільтр виконує прохід. Розмір кроку (англ. stride): Розмір кроку визначає величину зсуву фільтру на кожному кроці. Чим більше крок, тим менше фільтр застосовується і тим менше розмір вихідної матриці. Зазвичай використовується крок, що дорівнює одиниці, проте більший крок може дозволити побудувати модель, поведінка якої буде нагадувати рекурсивну нейронну мережу (тобто згорткова мережа з великим кроком буде виглядати як дерево); це дозволить формувати менші вихідні об’єми просторових вимірів.

При проході фільтру виникають ситуації, коли зручно заповнювати кордон вхідного зображення нулями. Розмір цього доповнення нулями і є гіперпараметром. Ключовою особливістю доповнення нулями є те, що воно дозволяє контролювати просторовий розмір вихідних об’ємів (найчастіше ми використовуємо дану властивість, щоб зберегти просторовий розмір вхідної інформації з метою збереження вхідних та вихідних значень висоти і ширини однаковими). Без використання доповнення нулями отримуємо вузьку згортку.

Просторовий розмір вихідного об’єму можна обчислити як функцію від вхідного об’єму, розміру рецептивного поля нейронів згорткового шару, кроку, з яким вони переміщуються, і кількості заповнення нулями на кордоні вхідної картинки. Параметри згорткового шару [5]:

K - кількість фільтрів; F - розмір фільтра;

S - крок;

P - кількість заповнень нулями.

Розмір (об’єм) вихідних даних W2 × H2 × D2 обчислюється таким чином

(2.1, 2.2,2.3).

$$W_2 = (W_1 - F + 2 \times P) / + 1$$
(2.1)

$$H_2 = (H_1 - F + 2 \times P) / + 1$$
(2.2)

$$D_2 = K,$$
(2.3)

де W1 - ширина, H1 - висота, D1 - глибина вхідних даних.

Спільне використання параметрів використовується в шарах згортки з метою контролю кількості параметрів.

Ще одним параметром виключно згорткових мереж є шари об’єднання. Шари об'єднання допомагають скоротити розмірність вихідної інформації, при цьому зберігаючи саму помітну інформацію. Наприклад, якщо фільтр визначає, чи міститься явна ознака. Якщо десь присутня явна ознака, то результат застосування фільтра до цієї області зображення дасть велике значення і невелике для інших частин зображення.

Канали кольору (англ. channels): канали - це різні "погляди" на вхідні дані. Наприклад, в розпізнаванні зображень зазвичай три каналу - RGB.

2.4 Техніка batch-normalization

У багатошарових мережах кожен шар приймає дані з попереднього шару. Дані можуть бути представлені у вигляді будь-якого тензору, координати якого попередньо визначені. Це призводить до того, що параметри в подальших шарах мережі гіршають. Стає необхідним вибрати дуже низьку швидкість навчання, щоб відрегулювати той факт, що функції введення можуть різко змінюватися з часом.

Для шару, що приймає дані, будо б дуже зручним отримувати їх у вигляді тензорів саме з координатами із фіксованого розподілу, одного для всіх шарів, тоді

навчання перетворенню до параметрів розподілу стає непотрібним. Стає доречним між усіма шарами встановити обрахунки, які б оптимально нормалізували виходи попереднього шару, і знижували б тиск на наступний шар, що значно покращило б його роботу в мережі. Отже спосіб вирішення цієї проблеми полягає в нормалізації міні-партій (англ. batches).

Популярними на сьогоднішній день є дві техніки нормалізації : local response normalization та batch-normalization. Обидві нормалізації використовуються для запобігання зниженню швидкості навчання параметрів мережі. Локальна нормалізація (англ. local response normalization) відповідності є самостійним шаром мережі. Дана операція нормалізує кожне значення вхідної матриці по каналу.

Пакетна нормалізація (англ. batch-normalization), вперше введена в , застосовує стандартну нормалізацію до отриманих значень, а потім лінійно трансформує їх

$$y = \frac{x - \mu}{\sqrt{\sigma^2 - \epsilon}} \cdot \gamma + \beta, \tag{2.4}$$

тут ϵ, γ та β є параметрами, що підлягають налаштованню. μ та σ, середнє значення та дисперсія залежать від того на якому етапі знаходиться мережа. Якщо відбувається навчання мережі, то ці значення беруться від значень, отриманих тільки в поточний момент. Під час тестування мережі значення беруться з усієї зібраної статистики.

Питання коли саме застосовувати нормалізацію (шар нормалізації) й досі залишається відкритим. Для виконання dropout не має значення чи нормалізація застосовується до або після функції активації. З огляду на це, можливі варіанти застосування:

- Шар згортки / Класифікатор (повнозв’язний шар) → BN (нормалізація) → функція активації → Dropout → . . .
- Шар згортки / Класифікатор (повнозв’язний шар) → функці активації → BN → Dropout → . . .

- Шар згортки / Класифікатор (повнозв’язний шар) → функція активації → Dropout → BN → . . .
- Шар згортки / Класифікатор (повнозв’язний шар) → Dropout → BN → функція активації → . . .

Для побудови майбутньої мережі було вирішено виконувати BN одразу після застосування операції згортки і до застосування операції нелінійності (функції активації).

2.5 Класифікатор

Як і в звичайних нейронних мережах, в згорткових мережах в повнозв'язну шари (класифікатор) [9] нейрони мають зв’язок з усіма функціями активації з попереднього шаріу. Активації ж шарів класифікації можуть бути обчислені за допомогою множення матриць, що супроводжується зміщенням.

Відмінність між шаром класифікації і шаром згортки полягає у тому, що нейрони шару згортки з'єднані тільки з локальною областю на вході, і що нейрони цього шару можуть спільно використовувати параметри. Однак нейрони в обох шарах, незважаючи на свої особливості, підраховують скалярний добуток, тому їх функціональна форма ідентична.

Більш того, можна виконати конвертацію між повнозв'язним і згортковим шарами.Класифікатор повинен запам'ятовувати всі навчальні дані та зберігати їх для подальших порівнянь із даними з тестового запуску. Це дуже ресурсозатратно, оскільки набори даних можуть бути розміром у гігабайтах.

Лінійний класифікатор (Рис 2.5) дає оцінку класу як зважену суму всіх значень пікселів у трьох його кольорових каналах. Залежно від того, які значення встановлено для цих ваг, функція класифікатору має здатність позитивно або негативно оцінювати (залежно від знаку кожної ваги) певні кольори у певних положеннях зображення. Наприклад, людина стверджуватиме, що клас "машина" може бути більш імовірним, якщо з боків на зображенні є багато зеленого кольору

(що може відповідати деревам і траві за містом).

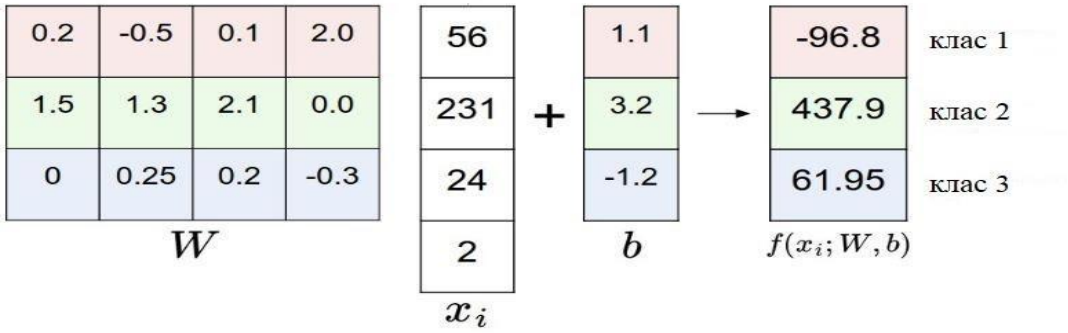


Рис 2.5 - Лінійний класифікатор

Ще один спосіб організації класифікатору – шаблонізація. Особливість способу стосується ваг W і полягає в тому, що кожен рядок W відповідає шаблону (іноді також називається прототипом) для одного з класів. Оцінка кожного класу для зображення отримується шляхом порівняння кожного шаблону, один за одним, із зображенням за допомогою внутрішньої ознаки (або точки-ознаки), щоб знайти те, що "підходить" найкраще. За допомогою такого порівняння лінійний класифікатор знаходить відповідність шаблону.

2.6 Показники нейронної мережі: точність, втрата та якість

Епоха - прямий і зворотний прохід по всім тренувальним прикладам.

Розмір серії (batch) - кількість тренувальних прикладів для однієї ітерації прямого і зворотного проходів. Кількість ітерацій - кількість проходів: кожен прохід використовує приклади (batch). Один прохід дорівнює прямому проходу та зворотньому проході. Тобто маючи 1000 прикладів, batch = 500, нам буде потрібно зробити дві ітерації, щоб завершити одну епоху.

З математичної точки зору, навчання нейронних мереж - багатопараметрична задача нелінійної оптимізації.

Процес навчання мереж описують за допомогою графіку кривої, де горизонтальна вісь відображає кількість навчальних зразків, що отримує мережі, а

вертикальна вісь відображає значення помилки, що отримано при класифікації вхідних даних.

На рисунку 2.6 зображено два криві: криву значень помилок на навчальному наборі (величина якої задана горизонтально) та криву значень помилок на тестовому або перевірочному наборі (яка має фіксований розмір). Чим більше даних використовується для навчання, тим більше помилок видасть архітектура мережі, що відповідатимуть навчальним даним. У той же час навчальні дані стають більш схожими на справжній розподіл даних, які слід зафіксувати за навчальними даними. У певний момент часу помилка на навчальному та тестовому наборах повинна бути приблизно однаковою.

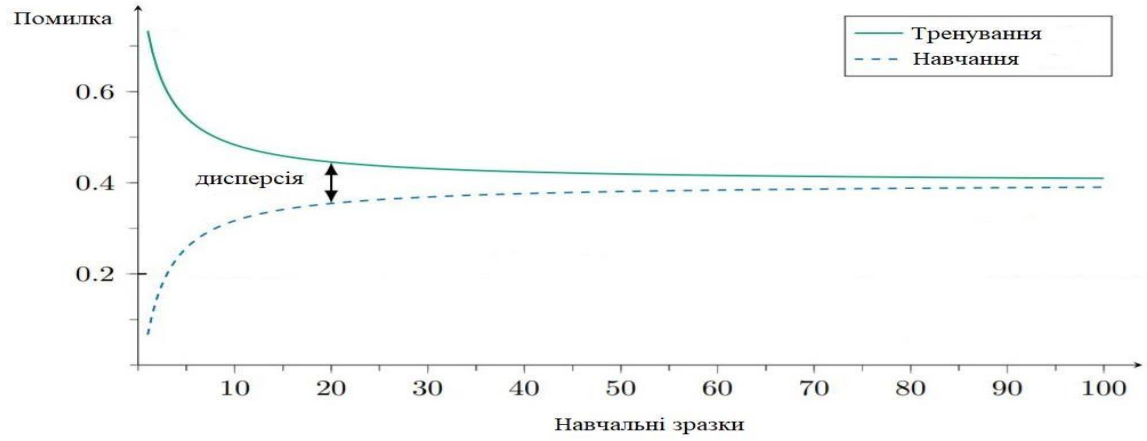


Рис 2.6 - Процес навчання мережі

Крива процесу навчання для встановлення є індикатором правильності роботи мережі, якщо велика кількість навчальних даних без будь-яких змін, що в свою чергу покращує ефективність мережі. Побудувавши криву навчального набору, можна оцінити, чи здатна архітектурна модель мережі відповідати даним для потрібної класифікації помилки. Значення помилки при тестовому наборі ніколи не повинне бути значно нижчим, ніж значення помилки навчального набору. Якщо помилка на навчальному наборі занадто висока, то збільшення даних не допоможе вирішити проблему

Натомість стане зрозуміло, що сама архітектура моделі або алгоритм її навчання потребують коригування. У випадку, коли на графіку крива навчального

набору значно перевищує криву тестового (перевірочного) набору даних необхідно або зменшити роздільну здатність зображень, або ввести додаткові навчальні зразки даних, або збільшити регуляризацію для вирішення даної проблеми.

Показники мережі слугують для обчислення та відображення гіперпараметрів (наприклад, навчальної епохи). Показники можна подати у вигляді графіків гіперпараметрів на горизонтальній осі та значення якості на вертикальній осі. Точність розпізнавання, представлена як помилка = (1) або втрата є типовими показниками якості. У випадку, коли кількість навчальних епох розглядається як головний гіперпараметр, показники діють як індикатор довгого часу тренування та продуктивності мережі. Побудувавши графік значень помилки на наборі тренувань, а також значень помилки на наборі перевірки, можна також оцінити, чи є загрозливим перенавчання(Рис 2.7).

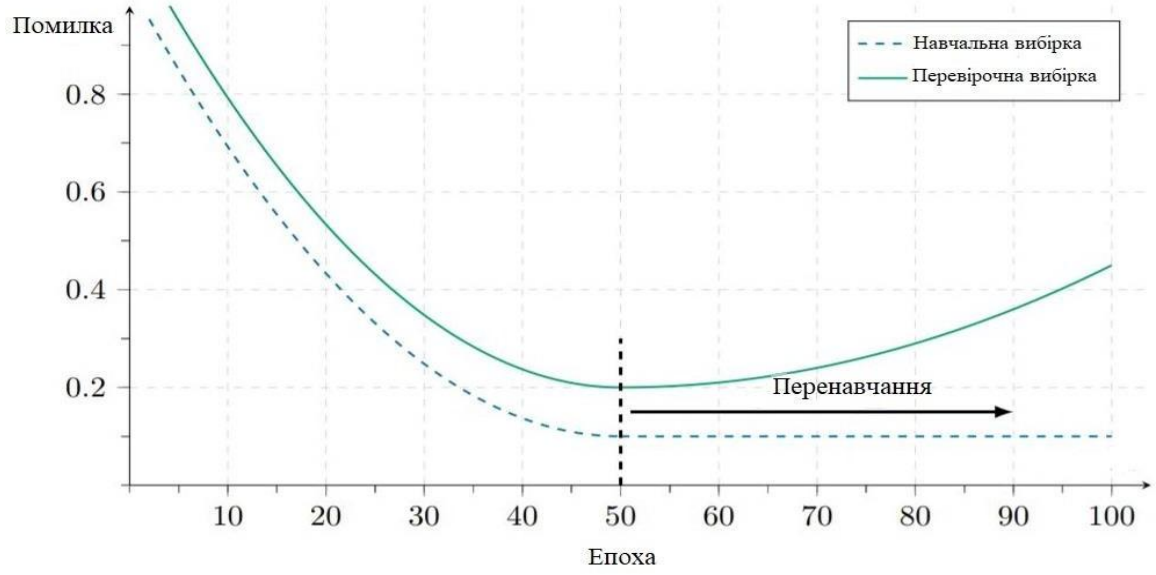


Рис. 2.7 - Проблема перенавчання мережі

Розглянемо рисунок 2.7 детальніше де зображено типову ситуацію перенавчання.

Кількість епох навчання виступає як гіперпараметр мережі, а оцінка якості розпізнавання виражено через значення помилки. Чим довше буде навчатися мережа, тим краще вона звикне до тренувального набору даних.

У якийсь момент мережа добре розпізнаватиме лише для навчальні дані і втратить здатність до узагальнення.

Потенціал перенавчання залежить не лише від кількостей параметрів та даних, але й від відповідності структури моделі формі даних, та величини похибки моделі в порівнянні з очікуваним рівнем шуму або похибки в даних.

Навіть коли пристосована модель не має надмірного числа параметрів, слід очікувати, що пристосований взаємозв'язок працюватиме на новому наборі даних не так добре, як на наборі, використаному для пристосовування.

Можливість перенавчання існує тому, що критерій, який застосовується для тренування моделі, відрізняється від критерію, який застосовується для оцінки її ефективності.

Зокрема, значення коефіц детермінації відносно первинних тренувальних даних скорочуватиметься[en].

Щоби уникати перенавчання, необхідно використовувати додаткові методики (наприклад, перехресну перевірку, регуляризацію, ранню зупинку, обрізку[en], баєсові апріорні параметрів або порівняння моделей), які можуть вказувати, коли подальше тренування не даватиме кращого узагальнення. Основою деяких методик є або

(1) явно штрафувати занадто складні моделі, або (2) перевіряти здатність моделі до узагальнення шляхом оцінки її продуктивності на наборі даних, не використаному для тренування, який вважається наближенням типових небачених даних, з якими стикатиметься модель.

На цьому етапі криві якості навчальної та перевірочної виборок на графіку розходяться. Поки класифікатор продовжує вдосконалювати показник якості на тренувальному наборі, він навпаки погіршує якість розпізнавання при застосуванні тестового набору. При ситуації, коли показник якості не поліпшується при достатній кількості епох варто змінити значення ініціалізованих ваг нейронів на початку мережі, або застосувати нормалізацію моделі.

2.7 Функція втрат

Застосовані у лінійних класифікаторах функції визначення оцінки приналежності об’єкту до певного класу не надають певного контролю за вхідними даними, адже вони параметризовані значенням набором ваги нейронів W. Тому варто встановити контроль над цими вагами нейронів, і потрібно зробити це їх таким чином, щоб прогнозовані оцінки класів відповідали значенню з навчальних виборок.

Практичне застосування класифікаторів показало, що не завжди оцінка класу, що дійсно відповідає правильній ідентифікації об’єкта є вірною, тому рішенням стане обчислення не позитивного результату розпізнавання, а навпаки – негативного. Для цього і використовується функція втрат або цільова функція

Інтуїтивно, значення функції втрат буде високим, якщо роботу з класифікації навчальних даних завершилася помилками, і значення буде низьким, якщо все працюватиме вірно.

Функція втрат (англ. loss function) , також називається функцією помилки або функцією вартості - це функція, яка вираховує дійсне значення для складної події, як-от передбачення приналежності до класу вхідного вектора. Вона використовується для визначення цільової

функції.

У пролематиці класифікації функція втрат подається як перехресна ентропія з L_1або L_2 регуляризацією (2.6). С точки зору теорії інформації, навчанням є мінімізація значень перехресної ентропії стосовно реальних класів і гіпотезами моделі .

$$E_{CE}(W) = \sum_{x \in X} \sum_{k=1} [t_k^x \log(o_k^x) + (1 - t_k^x) \log(1 - o_k^x)]$$

Перехресна ентропія

$$+ \lambda_1 \cdot \underbrace{\sum_{w \in W} |w|}_{L_1} + \lambda_2 \cdot \underbrace{\sum_{w \in W} w^2}_{L_2}$$

Значення втрат мережі

(2.5)

У формулі 2.5 параметр W - вага, X - це набір навчальних даних,

$K \in N \geq 0$ це число класів, t^x вказує чи x є прикладом з класу k. o^x є вихідним значенням алгоритму класифікації, який залежить від ваг. λ1, λ2 ∈ [0, ∞) вага регуляризації, зазвичай це значення менше за 0,1.

Значенні втрати даних є позитивним, коли класифікація є неправильною, але значення втрат є числом більшим для більш складних моделей мережі. Якщо дві моделі мереж розпізнають однаковий набір даних, то кращою визнається модель архітектура якої є простішою.

Приводом для зображення саме функції втрат методом кривої на графіку замість інших показників якості є те, що тоді графік

міститиме більше інформації про якість мережі. Головним недоліком функції втрати є те, що вона не має верхньої межі, як точність, і її важко інтерпретувати. Функція втрати лише показує відносний прогрес навчання, тоді як точність показує абсолютний прогрес.

Існують критерії покращення використання функції втрат, які можуть допомогти у вдосконаленні мережі.

Якщо втрати протягом кількох епох не зменшуються, то темп навчання може бути занизьким. Процес оптимізації також може бути зафіксований у місцевому мінімумі.

Неправильне значення функції втрат у вигляді числа, що не існує, можебути пов'язана із надто високими темпами навчання. Іншою причиною може бути поділ на нуль або логарифм нуля.

В обох випадках додавання невеликої константи, як от 10^{-7}, вирішує проблему.

Якщо крива функції втрат в залежності від кількості епох напочатку досягла межі, погані значення ініціалізації ваги може бути цьому причиною.

2.8 Точність, як критерій якості навчання нейронної мережі

Серед архітектурних моделей нейронних мереж, робота яких спрямована на класифікацію зображень, існують декілька критеріїв оцінки якості даної роботи. Більшість критеріїв базуються на так званій матриці передбачень, що позначається c_{ij}, діагональ якої містить кількість правильних прогнозів. Припустимо нехай $t_i = \sum_{j=1}^{k} c_{ij}$ буде числом навчальних зразків для класу i, тоді найбільш узагальненим критерієм якості є точність (англ. accuracy) виражена такою формулою

$$accuracy(\mathrm{c}) = \frac{\sum_{j=1}^{k} c_{ii}}{\sum_{j=1}^{k} t_i} \in [0,1] \quad (2.6)$$

Одна з проблем точності, як критерію якості - це відхилення від оцінки класу. Якщо один клас є більш узагальненим за інші, то найпростішим способом оцінення будь-якого іншого класу високим балом буде його постійна класифікація як узагальненого. Для вирішення цієї проблеми можна використовувати усереднене значення точності.

Однак окрім такого критерію як точність класифікації, на практиці важливими є також і інші критерії якості:

- швидкість оцінки та аналізу нових зображень, що надаються мережі;
- затримка часу навчання;
- ресурсозатратність;
- стійкість до (не) випадкових відхилень у навчальних даних;
- розмірність архітектури мережі.

Помітно, що подання значення числа точності з плаваючою комою дозволяє обробляти більше даних на одному пристрої.

2.9 Тестування програми

Для ефективності навчання був використаний набір даних для тестування "Автомобілів" - це набір даних "Stanford Cars", що містить 16 185 зображень різних розмірів та типів.

На сьогоднішній день потрібно зосередитися не тільки продуктивності, але також на розмірах та обчислювальній потужності.

Перейшовши на нижчу на вищу роздільну здатність ми покращуємо ефективність розпізнавання типів автомобілів, але це також напряму залежить від продуктивності програми та обчислювальної потужності комп'ютера .

Графік залежності помилки мережі від кількості ітерацій навчання зображений на рис. 4.

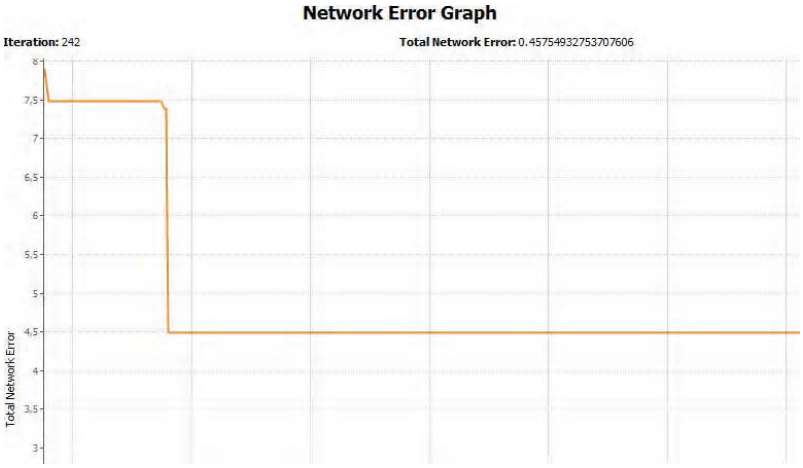


Рис.4. Графік залежності помилки нейронної мережі від кількості ітерацій при її тестуванні за допомогою нормалізованої вибірки

Мінімальна помилка склала $\delta_{st} = 0,45$, що становить

$$\varepsilon_{st} = \frac{0,45}{5} \cdot 100\% = 9,14\%$$

Розглянем навчання нейромережі за допомогою нормалізованого набору параметрів суб'єктивного тестування з роздільною здатністю зображення 400 ppi. Графік залежності помилки мережі від кількості ітерацій навчання зображений на рис. 5.

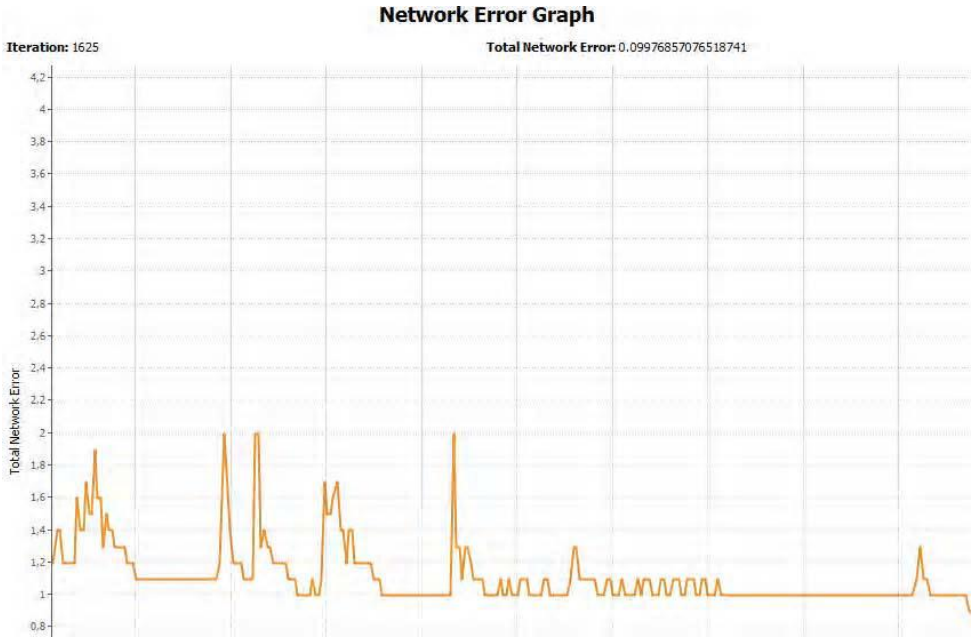


Рис.5. Графік залежності помилки нейронної мережі від кількості ітерацій при її тестуванні за допомогою нормалізованої вибірки

В процесі навчання мінімальна помилка нейронної мережі склала $\delta_{mod} = 0{,}1$, що становить $\varepsilon_{mod} = \frac{0{,}1}{5} \cdot 100\% = 2\%$

Можна зробити висновок, що використання роздільної здатності зображення в 400 ppi дозволяє збільшити точність визначення типу автомобіля.

3 РОЗРОБКА КОМП'ЮТЕРНОЇ СИСТЕМИ РОЗПІЗНАВАННЯ ГРАФІЧНИХ ОБРАЗІВ

3.1 Постановка задачі і складності, пов'язані з її реалізацією

Адаптивний алгоритм навчання вибирає невелике число ознак низького рівня з більш ніж 180 тис. ознак. Існує багато мотивацій для використання ознак, а не окремих пікселів. Найважливіша причина: ознака може кодувати різні значення, які важко визначати і вивчати, використуючи кінцеву безліч навчальних даних. Ознаки обчислюються набагато швидше, ніж піксельні величини, що детально буде розглянуто далі. В алгоритмі Віола-Джонса використується три виду ознак: два прямокутника, три прямокутника і чотири, хоча насправді число видів ознак може бути і більше. Область очей більше темна в порівнянні з середньою областю особи і перенісся. Примітиви цієї конфігурації і розмірів найбільш кращим чином характеризують дане зображення. На основі таких класифікаторів з відібраними найбільш ефективними примітивами будується каскад класифікаторів. Кожен наступний елемент каскаду має більш жорсткі умови успішного проходження, ніж попередній (використується більше примітивів). Тим самим до кінця доходять тільки самі правильні.

3.2 Розробка системи розпізнавання

Навчання базується на ідеї бустінга. Для успішного навчання потрібно кілька тисяч позитивних і негативних зображень (залежно від того, є на ньому особа, чи ні). На рисунку 3.1 показаний алгоритмбустінга:

Бустінг використується для вибору невеликої кінцевої безлічі ознак високого рівня й для тренування класифікатора. Для навчання нейромережевого класифікатора застосується простий (англ. weak) алгоритм навчання.

1) Дана навчальна вибірка у форматі $(x_1, y_1), \ldots, (x_n, y_n)$, де

$$y_i = \begin{cases} 1, & \text{cars} \\ 0, & \text{інше} \end{cases}$$

Ініціалізувати ваги:

$$w_{1,i} = \begin{cases} 0.5/m, \text{якщо } y_i = 0 \\ 0.5/l, \text{якщо } y_i = 1 \end{cases}$$

2) Для t = 1, …, T

Нормалізувати ваги: w_t – ймовірнісно розподілені.

$$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$

Для кожної ознаки j тренирувати класифікатор h_j, який обмежен застосуванням однієї ознаки. Помилка тренування ε_j при цьому знаходиться відносно w_t:

$$\varepsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$$

a) Обрати класифікатор h_t з найменшою помилкою ε_t.

b) Відновити ваги:

$w_{t+1,i} = w_{t,i}\beta_t^{1-\varepsilon_i}$, де $\varepsilon_i = 0$, якщо приклад x_i класифікувався коректно;

$\varepsilon_i = 1$, інакше та $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$.

c) Результуючий класифікатор дорівнює:

$$h(x) = \begin{cases} 1, \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t, \text{де } \alpha_t = \log \frac{1}{\beta_t} \\ 0, \text{інакше} \end{cases}$$

Рисунок 3.1 – Алгоритм бустінгу

В даний час одним з актуальних питань інформаційних технологій є проблема розпізнавання образів з застосуванням ефективних методів нейрон-обробці інформації. Стандартні схеми аналізу зображень та оцінки даних, не завжди приносять бажаних результатів, тому що вони негнучкі і прив'язані до певного не адаптивного алгоритму.

Нейронні мережі успішно використовують у вирішенні багатьох проблем розпізнавання образів: розпізнавання символів, розпізнавання об'єктів і багатьох інших. Проблема виявлення образу особи дуже важка через велику різноманітність викривлень, таких як різний вираз обличчя, умови зйомки та інших. Перевага використання нейронних мереж для виявлення обличчя – здатність до навчання системи для виділення ключових характеристик обличчя з навчальних вибірок. В даний час найбільш часто в завданнях розпізнавання та ідентифікації зображень використовують класичні нейромережеві архітектури (багатошаровий персептрон, мережі з радіально-базисної функцією та інші), але,

як показує аналіз даних робіт, застосування класичних нейромережевих архітектур до даної задачі є неефективним з наступних причин:

- до даного завдання зазвичай застосується ансамбль нейронних мереж (2-3 нейронні мережі, навчені з різними початковими значеннями синаптичних коефіцієнтів і порядком пред'явлення образів), що негативно позначається на обчислювальної складності розв'язання завдання;
- як правило, класичні нейромережеві архітектури використуються в сукупності з допоміжними методами виділення сюжетної частини зображення (сегментація за кольором шкіри, виділення контурів та іншими), які вимагають якісної й кропіткої передобробки навчальних і робочих даних, що не є ефективним;
- нейромережеві архітектури є вкрай чутливими до впливу різних зовнішніх факторів (зміни умов зйомки, присутність індивідуальних особливостей на зображенні, зміна орієнтації).

Додатково виникають труднощі застосування традиційних нейронних мереж до реальних задач розпізнавання та класифікації зображень.

По-перше, як правило, зображення мають велику розмірність, відповідно зростає розмір нейронної мережі (кількість нейронів, вагові коефіцієнти). Велика кількість параметрів збільшує місткість системи і відповідно вимагає більшої тренувальної вибірки, що збільшує час і обчислювальну складність процесу навчання.

По-друге, недоліком повнозв'язної архітектури є те, що топологія введення повністю ігнорується. Вхідні змінні можуть бути представлені в будь-якому порядку, не зачіпаючи мета навчання.

На подолання цих недоліків спрямована архітектура згортковох нейронних мереж. Згорткові нейронні мережі (ЗНМ) забезпечують часткову стійкість до змін масштабу, зсувів, поворотам, зміні ракурсу та інших спотворень.

Вирішенню цих проблем були присвячені роботи американського ученого французького походження Яна Ле Куна, пов'язані з розробкою і дослідженням згорткових нейронних мереж. Ідея згорткових нейронних мереж полягає в чергуванні згорткових шарів (С-шар), шарів підвибірки (S-шар) та використанні на виході полносвязного шару нейронів, які в цілому утворюють ансамбль спеціалізованих нейромереж. В основі згорткових мереж лежать три механізми, що використуються для досягнення інваріантності до переносу, масштабуванню, незначним спотворень:

1. Локальне вилучення ознак. Кожен нейрон отримує вхідний сигнал від локального рецептивного поля в попередньому шарі, витягуючи, таким чином, його локальні ознаки. Як тільки ознаку витягнуто – його точне розташування вже не має значення, оскільки встановлено його місцезнаходження щодо інших ознак.

2. Формування шарів у вигляді набору карт-ознак. Кожен обчислювальний шар складається з безлічі карт-ознак – площин, на яких всі нейрони повинні використувати один і той же безліч синаптичних ваг. Така форма ускладнює структуру мережі, однак має дві важливі переваги: інваріантність до зсуву, яке досягається за допомогою згортки з ядром невеликого розміру, і скорочення числа вільних параметрів, яке досягається за рахунок спільного використання синаптичних ваг нейронами однієї і тієї ж карти.

3. Підвибірка. За кожним шаром згортки йде обчислювальний шар, який здійснює локальне усереднення і підвибірку. За рахунок цього досягається зменшення дозволу для карт ознак. Така операція призводить до зниження чутливості вихідного сигналу оператора відображення ознак до незначного зсуву та іншим видам деформації. В якості такого оператора виступає одна з сигмоїдальних функцій, використованих при побудові нейронних мереж (наприклад, гіперболічний тангенс)

Слід зауважити, що послідовне застосування згортки і підвибірки призводить до так називемого підвищенню рівня ознак: якщо перший шар витягує локальні ознаки з областей зображення, то наступні шари витягують загальні ознаки, які називаються ознаками високого порядку. На рисунку 3 показана мережа згортки, що реалізує розпізнавання вхідного зображень. Розглянемо її архітектуру, особливості функціонування та параметричне опис.

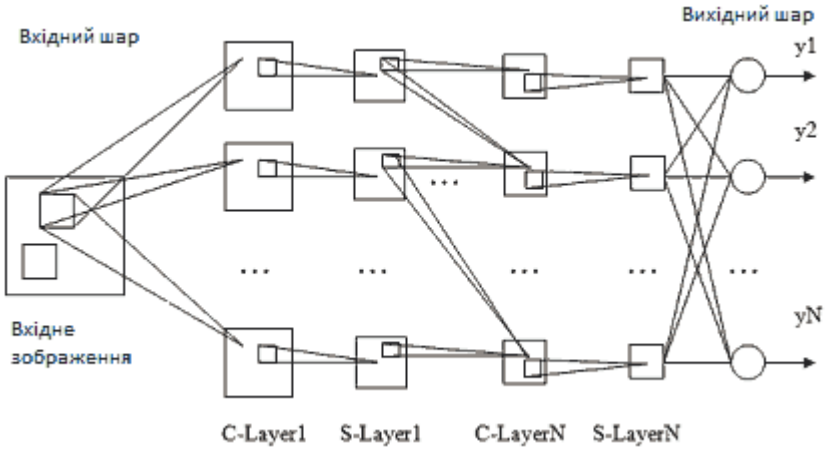


Рисунок 3.2 – Архітектура згорткової нейронної мережі

На теоретико-множинному рівні модель згорткової нейронної мережі CNN (Convolution Neural Network) представлена у вигляді кортежу :

$$CNN=<M,\{N^{\eta}\},\{<\{C_{(kl),m}^{(\eta,p^{\eta},(ij))}\},f^{\eta},\{S_{(qt),n}^{(\eta,p^{\eta},(i',j'))}\},\varphi^{\eta}>\},N,\{w_{i^*}^{j'}\},\psi>$$

де M – кількість шарів в нейронної мережі; η – номер шару; $1\le\eta\le M$; N^{η} – кількість карт ознак у η шару; C, S – вагові коефіцієнти C і S шарів відповідно, f^{η},φ^{η} – функція активації нейрона η шару, C і S шарів відповідно;

N – кількість нейронів в останньому шарі нейронної мережі; w – матриця вагових коефіцієнтів у вихідному шарі; ψ – функція активації нейронів вихідного шару.

Параметрична модель виявила і впорядкувала основні абстракції ЗНМ як об'єкта проектування, але вона не відображає поведінковий (функціональний) аспект елементів нейронної мережі. Цей недолік усуває логічна модель у вигляді

діаграми класів, яка містить основні об'єкти з параметричної моделі ЗНМ, наділені властивостями та операційної можливістю нейронної мережі.

Побудова логічної моделі починається з визначення сутностей (класів і об'єктів) системи. На рисунку 4 можна бачити реалізацію системи у вигляді діаграми класів. Виходячи з топології згорткової нейронної мережі, виділені наступні класи системи: Neuron – зберігає в собі всю необхідну інформацію для роботи нейрона (Входи, вагові коефіцієнти, виходи); абстрактний клас BasePlane – описує механізми роботи всіх типів площин (вхідна, С-площина, S-площина, вихідні), оголошує методи, відповідальні за режими навчання та розпізнавання.

Також в цьому класі реалізовані структури зв'язків між картами сусідніх шарів. Абстрактний клас SubPlane (розширення класу BasePlane) – реалізує методи, специфічні для згорткової й подвиборочної площин. Клас ImagePlane інкапсулює в собі роботу з зображенням, тобто переводить його в двовимірний масив вхідних значень і зберігає в собі зображення. Нейрони на даному етапі ще не потрібні. Клас ConvolutionalPlane реалізує функціональність С-шару мережі. Клас SubSamplingPlane – це реалізація S-шару мережі. Клас OutputPlane – шар нейронів, який повністю пов'язаний з попереднім шаром мережі (S2), генерує вихідні значення мережі. Клас NeuralNet являє собою Чорний ящик, тобто на вході потрібно зображення, на виході генерується вихідні сигнали. У загальному випадку, шари можуть взаємодіяти між собою, на що вказує зв'язок карт ознак.

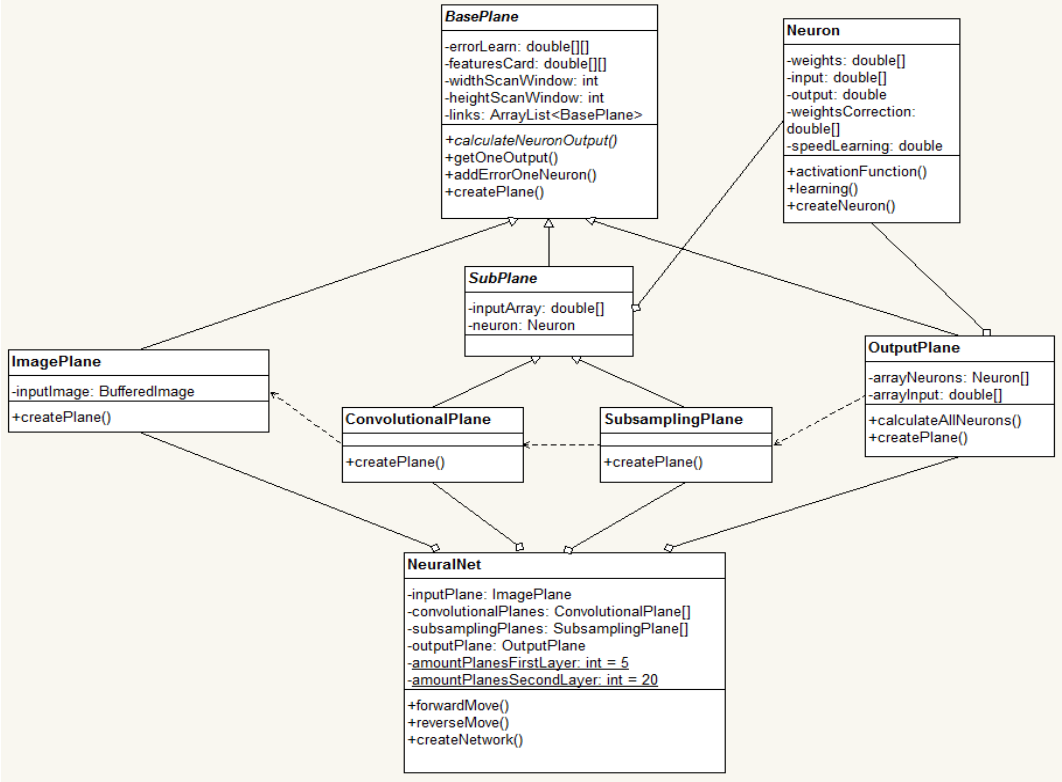


Рисунок 3.3 – Логічна модель згорткової нейр-мережі у вигляді діаграми UML

3.3 Реалізація комп'ютерної системи розпізнавання графічних образів

Комп'ютерна система розпізнавання графічних образів реалізована на основі Google Cloud Vision API. Дане програмне забезпечення швидко класифікує зображення на тисячі категорій (наприклад, "вітрильник", "лев", "Ейфелева вежа"), виявляє окремі об'єкти і особи в межах зображень, а також знаходить і читає надруковані слова, що містяться в зображеннях. Існує можливість створиювати метадані каталогу зображень, помірне образливого змісту, або включити нові маркетингові сценарії за допомогою аналізу зображень. Аналіз зображення, завантажені в запиті або інтегруватися з зберігання зображень на Google Cloud Storage.

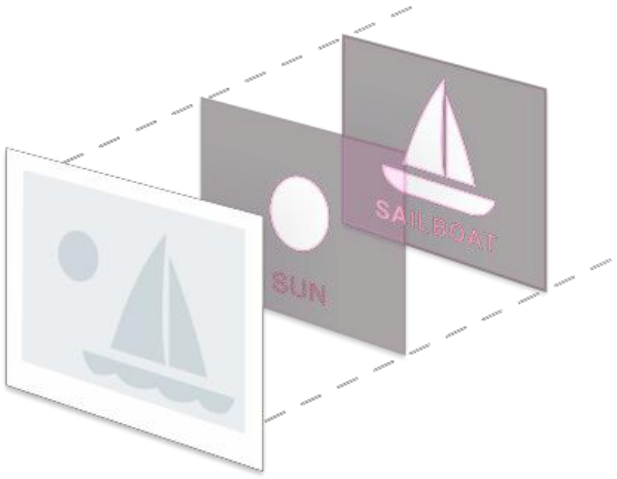


Рисунок 3.4 Приклад класифікацій графічних образів

Задопомогоюданоїкомп'ютерноїсистемилегковиявитиоб'єктина зображені, наприклад квіти, тварини, або тисячі інших категорій об'єктів, які зазвичай зустрічаються в зображеннях.VisionAPI з плином часу, водить нові поняття вводяться і точність підвищується.

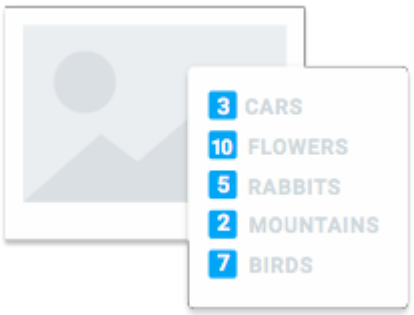


Рисунок 3.5 Приклад виявлення графічних об'єктів

Розроблена комп'ютерна складається з наступних модулів:

1. LabelDetection. Виявляє широкі набори категорій об'єктів в межах зображення, починаючи від видів транспорту до тварин.
2. ExplicitContentDetection. Виявлення явного змісту на зображенні.
3. LogoDetection. Виявлення популярних логотипів продукту в межах зображення.
4. LandmarkDetection.Виявлення популярних природних і штучних споруд в межах зображення.
5. OpticalCharacterRecognition. Виявлення і вилучення тексту всередині

зображення, з підтримкою широкого спектру мов, поряд з підтримкою автоматичної ідентифікації мови.

6. CarDetection. Виявлення декількох машин в межах зображення. Розпізнавання типу автомобіля не підтримується.

7. ImageAttributes. Виявити загальні атрибути зображення, такі як домінуючий колір.

Щоб використовувати Cloud API, потрібно налаштувати облікові дані для вашого програмного продукту для перевірки автентичності особистості до певного APIта отримати дозвіл на виконання завдання. (Ці облікові дані, пов'язані з механізмами схеми авт.).

При отриманні доступу до APIGoogleCloudPlatform, рекомендовано встановити ключ API для тестування, а також створити обліковий запис служби для використання продукції.

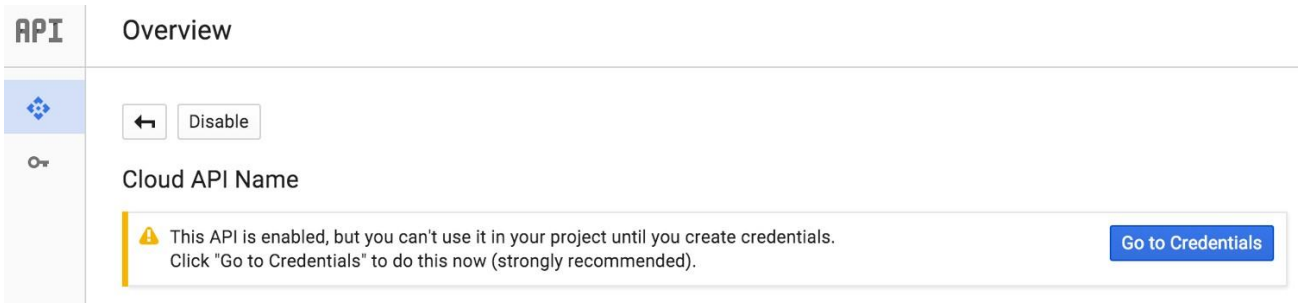


Рисунок 3.6 - Настройки проекту

Для того щоб додати облікові дані потрібно натиснути кнопку “GotoCredentials”.

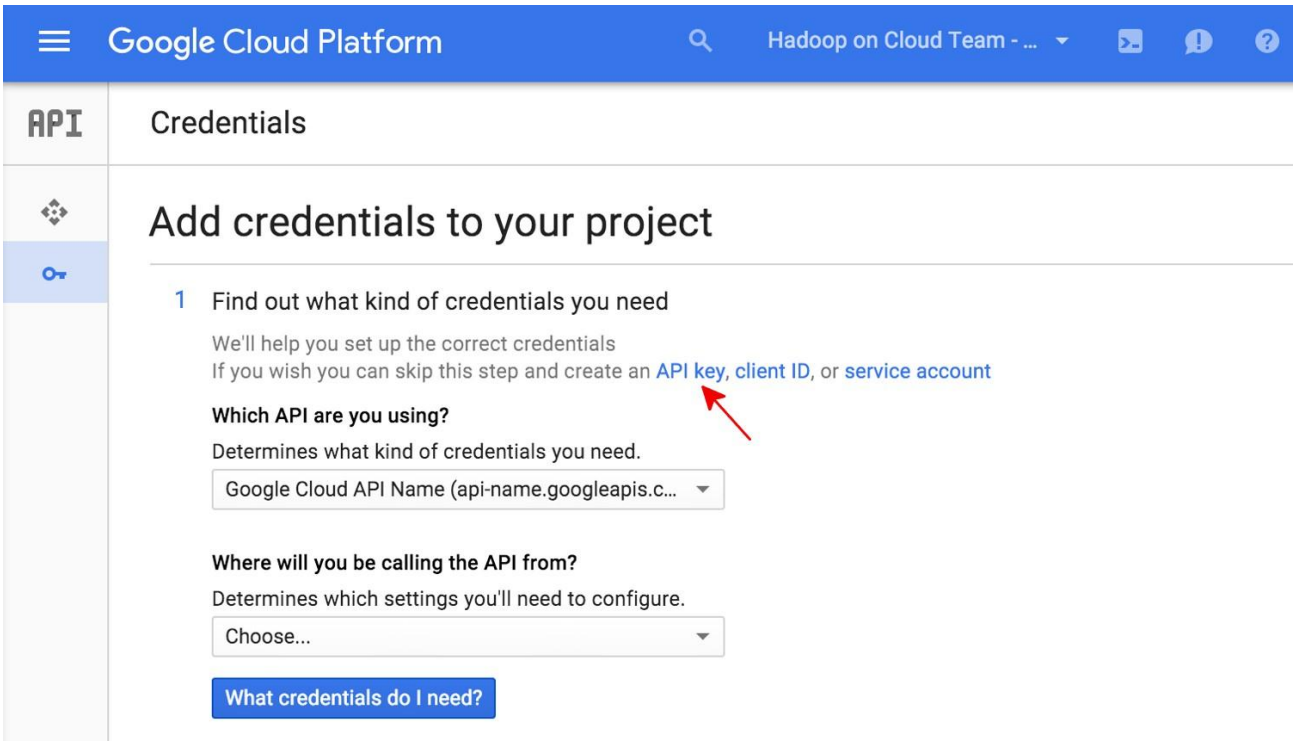


Рисунок 3.7 - Додавання облікових даних

Google Cloud Platform APIаутентифікації і авторизації (зазвичай згруповані разом як "авт") зазвичай виконується за допомогою облікового запису служби. Рахунок дозволяє код, щоб відправити дані програми безпосередньо на Cloud API. Службова обліковий запис, як обліковий запис Користувача, який представляє собою адресу електронної пошти. На відміну від облікового запису користувача, однак, обліковий запис служби відноситься тільки до додатку, і може бути використаний для доступу до API, для якого він був створений. В якості прикладу, ми покажемо, як створити обліковий запис служби за допомогою консолі платформи Google Cloud.

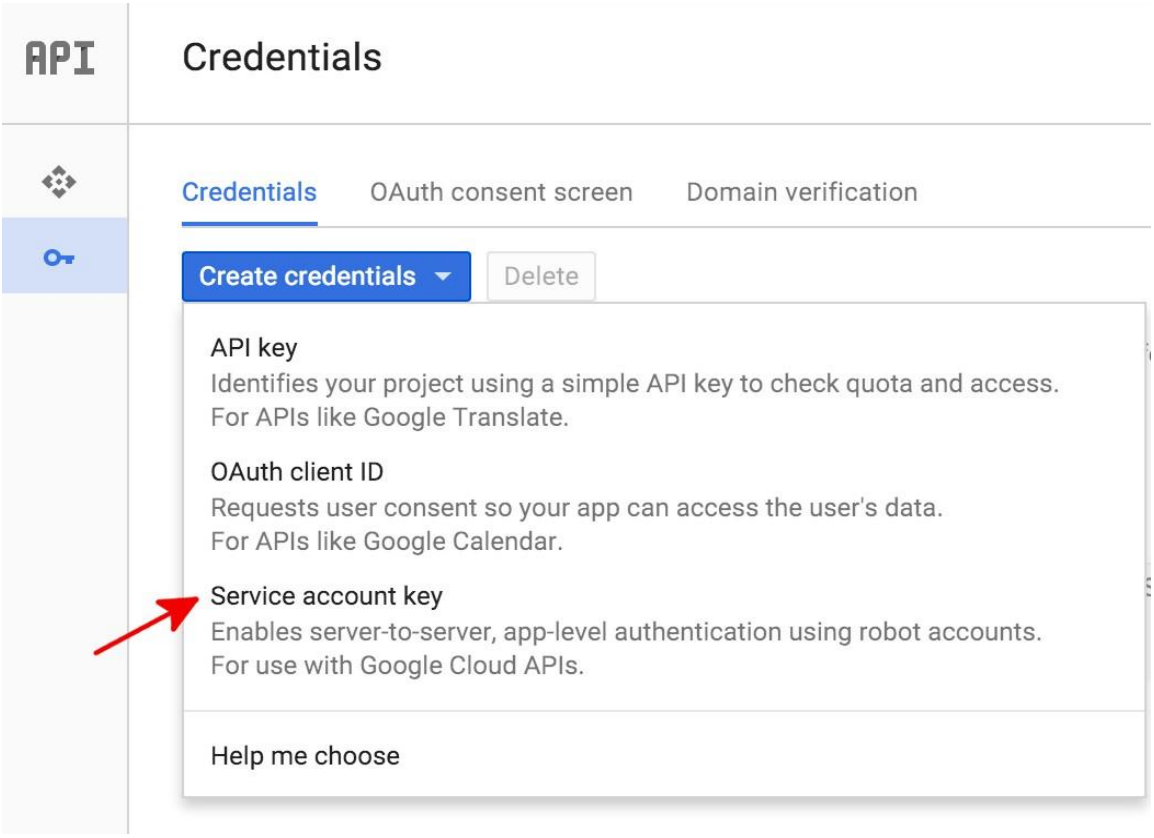


Рисунок 3.8 - Створення APIключа

Розглянемо приклад реалізації CarDetection. Щоб отримати доступ до APIGoogle, за допомогою офіційного клієнта SDKs, потрібно створити об'єкт сервісу, заснований на відкриття документа API, який описує API в SDK. Потрібно отримати його від служби VisionAPI's, використовуючи свої облікові дані:

```
useGoogle\Cloud\Vision\VisionClient;

// $projectId = 'YOUR_PROJECT_ID';
// $path = 'path/to/your/image.jpg'

$vision =newVisionClient([
    'projectId'=>
    $projectId,
]);
```

Щоб створити запит до API бачення, насамперед звернетеся до документації по API. У цьому випадку, ви будете питати зображень ресурс, щоб коментувати зображення. Запит на цей API приймає форму об'єкта зі списком запитів. Кожен елемент цього списку містить два біта інформації:

```
$image = $vision-
>image(file_get_contents($path),['CAR_DETECTION']);
$result = $vision->annotate($image);
```

У відповідь на наше прохання особи Анотація включає в себе велику кількість метаданих по виявлених машинах, які включають в себе координати багатокутника, що його охоплює.

3.4 Аналіз отриманих результатів

Комп'ютерна система розпізнавання графічних образів дозволяє розпізнавати різного типу об’єктів на графічних зображеннях використовуючи нейронні мережі. Реалізовано розпізнавання позначок, тексту, кольорів, автомобілів і т.д. Завантажуємо відео послідовність.

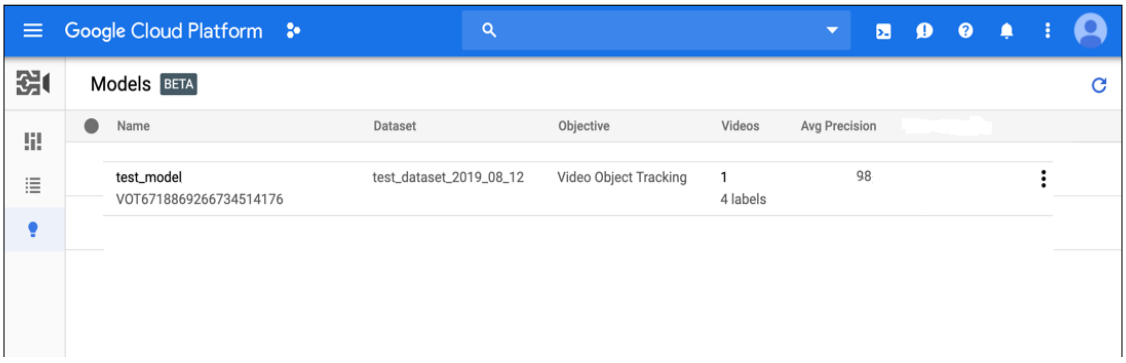


Рисунок 3.9

Після зчитування кадру, клас Detection відповідає за визначення і відстеження об’єктів. Обробку зображення, фільтрацію, створення моделі заднього фону, морфологічні операції та визначення об’єктів. Кожен об’єкт відео представляє собою екземпляр класу Object

Для оцінки результатів роботи будемо розглядати наскільки точно програма розпізнає об’єкти, що рухаються. Тобто кількість правильно визначених об’єктів і шуму.

Оскільки програма працює тільки з послідовностями зображень, які мають статичний фон. Деякі зміни у фоні, наприклад, коливання дерев чи снігопад зумовлюють їх розпізнавання як об’єктів переднього фону. Неякісне зображення може привести до помилкових визначень об’єктів, що також дає негативний результат. Отже, проаналізуємо результати роботи програми.

Розглянемо розпізнавання об'єктів на даному графічному зображенню:

Рисунок 3.10 -Вихідне графічне зображення для тестування

Програма виводить оригінальне зображення, знайдені границі. Визначені об’єкти позначаються прямокутниками синього кольору і визначаються як «cars». Результати програми повністю співпадають з вимогами: знайдені об’єкти – це легкові машини, що відповідно рухаються з постійною швидкість.

На наступнмх кадрах розподіл об’єктів синя рамка «car» і зелена рамка «truck»

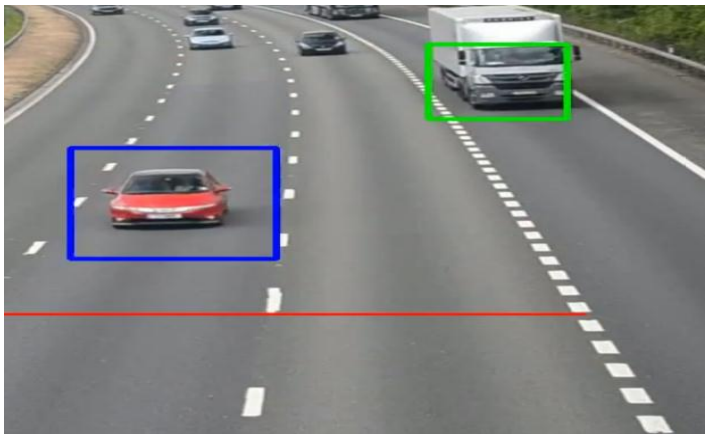


Рисунок 3.11

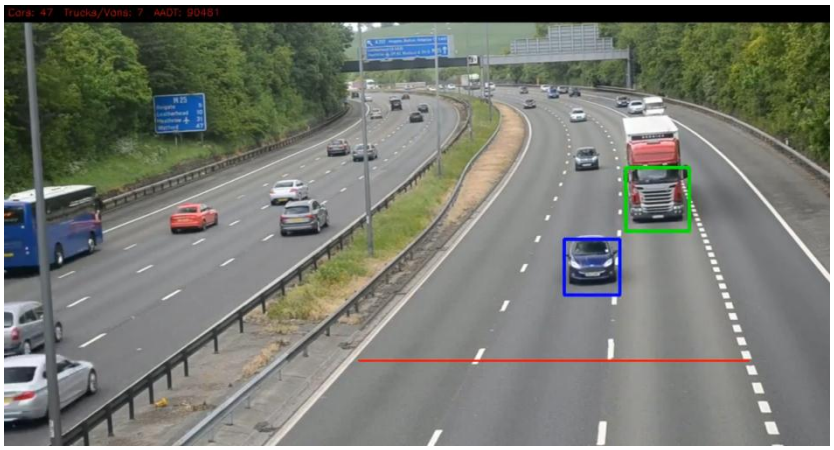


Рисунок 3.12

Трекінг особливих точок полягає у відстежування індивідуальних точок протягом кадрів у відео. Перевагою є те, що не потрібно визначати особливіточки в кожному кадрі. Ми можемо визначити їх один раз і відстежувати після цього. Це більш ефективний підхід, ніж визначення точок в кожному кадрі.

Використання каскаду Хаара , який вибирає невелике число ознак і об'єднує їх у ознаки більш високого рівня. В результаті отримуємо каскад з 38 кроків на великому наборі даних. Машини розпізнаються зі швидкістю 10 обчислень ознак у секунду для кожного вікна. Результуючий детектор ковзає по зображенню в різних масштабах, при цьому збільшується не масштаб зображення, а масштаб самого вікна. Таким чином отримується інваріантність до віддалення машини від камери.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки нейромережевої системи розпізнавання типів автомобілів на зображеннях.

Для проведення технологічного аудиту було залучено 3-х анонімних незалежних експертів Вінницького національного технічного університет, кафедри комп'ютерних систем та управління. За допомогою таблиці 4.1 за п'ятибальною шкалою використовуючи 12 критеріїв оцінки комерційного потенціалу розробки експерти надали свої оцінки.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів

Продовження табл. 4.1

5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої	Необхідно отримання великої кількості дозвільних документів на виробництво та	Процедура отримання дозвільних документів для виробництва та реалізації	Необхідно тільки повідомлення відповідним органам про виробництво	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

	кількості дозвільних документів на виробництво та реалізацію продукту	реалізацію продукту, що вимагає значних коштів та часу	продукту вимагає незначних коштів та часу	та реалізацію продукту	

Таблиця 4.2 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

В таблиці 4.3 наведено результати оцінювання експертами комерційного потенціалу розробки.

Таблиця 4.3 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	Анонім 1.	Анонім 2.	Анонім 3.
	Бали, виставлені експертами:		
1	2	2	1
2	3	2	3
3	1	2	1
4	4	5	5
5	1	1	1
6	4	5	4
7	0	0	0
8	5	5	5
9	4	5	4
10	5	4	5
11	5	5	5
12	1	2	1
Сума балів	$СБ_1$=35	$СБ_2$=38	$СБ_3$=35
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{35+38+35}{3} = 36$		

Середньоарифметична сума балів, розрахована на основі висновків експертів

склала 36, що згідно таблиці 4.2 вважається, що рівень комерційного потенціалу розробки є вище середнього.

Дана програма може бути використана в системах безпеки, навігації машин, стеження за підозрілою активністю.

Порівняємо програму, яка розробляється з аналогами, які існують на ринку. В якості аналога для розробки було обрано програмний засіб Car Go Enterprise .

Основними недоліками аналога є: метод фільтрації зображень та ціна. Також до недоліків можна віднести: не завжди точна класифікація за типами автомобілів.

У розробці дана проблема вирішується шляхом проектування і тестуванням удосконаленої нейронної мережі для полегшення процедури розпізнавання, що дозволило розпізнвати спецефічні позначки на зобрженнях, а також підвищити швидкодію та зменшити апаратні затрати.

В таблиці 4. 4 наведено порівняння аналога і нової розробки.

Таблиця 4.4 – Порівняння технічних характеристик нового методу та аналога

Технічні показники	Новий метод	Аналог	Відношення
Точність, %	99,8	75	1,3
Надійність, %	99	70	1,4
Величина затрати часу, хв..	5	90	0,06

Отже, з таблиці видно, що по всім технічним характеристикам новий розроблюваний метод має набагато кращі характеристики, що свідчить про доцільність його подальшої розробки та впровадження.

Проведемо оцінку якості продукції, яка є найефективнішим засобом забезпечення вимог споживачів та порівняємо її з аналогом.

Визначимо відносні одиничні показники якості по кожному параметру за формулами (4.1) та (4.2) і занесемо їх у відповідну колонку табл. 4.5.

$$q_i = \frac{P_{Hi}}{P_{Бi}} \qquad (4.1)$$

або

$$q_i = \frac{P_{Бi}}{P_{Нi}} \tag{4.2}$$

де $P_{Нi}$, $P_{Бi}$ – числові значення *i*-го параметру відповідно нового і базового виробів.

Таблиця 4.5 – Основні параметри нової розробки та товару-конкурента

Показник	Варіанти		Відносний показник якості	Коефіцієнт вагомості параметра
	Базовий (товар-конкурент)	Новий (інноваційне рішення)		
1	2	3	4	5
Точність, %	75	99,8	1,3	45%
Надійність, %	70	99	1,4	10%
Величина затрати часу, хв..	90	5	18	45%

$$q_1 = \frac{99,8}{75} = 1,3;$$

$$q_2 = \frac{99}{70} = 1,4;$$

$$q_3 = \frac{90}{5} = 18.$$

Відносний рівень якості нової розробки визначаємо за формулою:

$$К_{я.в.} = \sum_{i=1}^{n} q_i \cdot \alpha_i, \tag{4.3}$$

$$К_{я.в.} = 1,3 \cdot 0,45 + 1,4 \cdot 0,1 + 18 \cdot 0,45 = 8,83$$

Відносний коефіцієнт показника якості нової розробки більший одиниці, отже нова розробка якісніший базового товару-конкурента.

Наступним кроком є визначення конкурентоспроможності товару. Конкурентоспроможність товару є головною умовою конкурентоспроможності підприємства на ринку і важливою основою прибутковості його діяльності.

Однією із умов вибору товару споживачем є збіг основних ринкових характеристик виробу з умовними характеристиками конкретної потреби покупця. Такими характеристиками найчастіше вважають нормативні та технічні параметри, а також ціну придбання та вартість споживання товару.

Приблизна ціна нового товару складе 1500 грн. Занесемо ці та інші показники (взяті з попередніх розрахунків) до табл. 4.6.

Таблиця 4.6 – Нормативні, технічні та економічні параметри інноваційного рішення і товару-виробника

Показники	Варіанти	
	Базовий (товар-конкурент)	Новий (інноваційне рішення)
1	2	3
1. Нормативно-технічні показники		
Точність, %	75	99,8
Надійність, %	70	99
Величина затрати часу, хв..	90	5
2. Економічні показники		
Ціна придбання, грн	2600	1500

Загальний показник конкурентоспроможності інноваційного рішення (К) з урахуванням вищезазначених груп показників можна визначити за формулою:

$$K = \frac{I_{m.n.}}{I_{e.n.}}, \tag{4.4}$$

де $I_{m.n.}$ – індекс технічних параметрів; $I_{e.n.}$ – індекс економічних параметрів.

Індекс технічних параметрів є відносним рівнем якості інноваційного рішення. Індекс економічних параметрів визначається за формулою (4.5)

$$I_{e.n.} = \frac{\sum_{i=1}^{n} P_{Нei}}{\sum_{i=1}^{n} P_{Бei}}, \tag{4.5}$$

де $P_{Нei}$, $P_{Бei}$ – економічні параметри (ціна придбання та споживання товару)

відповідно нового та базового товарів.

$$I_{е.п.} = \frac{1500}{2600} = 0{,}58;$$

$$К = \frac{8{,}83}{0{,}58} = 15{,}22.$$

Зважаючи на розрахунки, можна зробити висновок, що нова розробка буде конкурентоспроможніше, ніж конкурентний товар на.

4.2 Прогнозування витрат на виконання науково-дослідної роботи

1. Основна заробітна плата – винагорода за виконану роботу відповідно до встановлених норм праці. Вона встановлюється у вигляді тарифних ставок (окладів) і відрядних розцінок для робітників та посадових окладів для службовців. Стаття «Основна заробітна плата робітників» містить витрати на виплату основної заробітної плати робітникам, зайнятим виробництвом продукції.

Основна заробітна плата кожного із розробників (дослідників) *З* розраховується за формулою:

$$З = \frac{M}{T_P} \cdot t \ ,[грн.], \tag{4.6}$$

де *М* – місячний посадовий оклад конкретного розробника.

T_P – число робочих днів, $Т_р = 22$;

t – число днів роботи розробника.

Розрахунки основної заробітної плати зведемо в таблицю 4.7:

Таблиця 4.7 – Розрахунок основної заробітної плати розробників

Найменування посади	Місячний посадовий оклад,грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник	9500	431,8	5	2159
Програміст	6000	272,7	30	8182
Всього				10341

2. До статті «Додаткова заробітна плата» відносяться витрати на виплату виробничому персоналу підприємства додаткової заробітної плати за працю понад установлені норми, заохочувальні виплати за поточну виробничу діяльність, компенсаційні виплати тощо. Звичайно, ці витрати встановлюються у відсотках до основної заробітної плати на підставі відповідних розрахунків на підприємстві:

$$З_д = 11\% \cdot З_\Sigma, \tag{4.7}$$

$$З_д = 11\% \cdot (10341) = 1137,5(грн.).$$

3. Витати на соціальні заходи виникають внаслідок здійснення обов'язкової сплати єдиного внеску на загальнообов'язкове державне соціальне страхування. Відрахування на соціальні заходи здійснюється від суми всіх витрат на оплату праці робітників, зайнятих безпосередньо виробництвом продукції:

$$В_{сз} = (З_\Sigma + З_д) \cdot \frac{\beta}{100\%}, \tag{4.8}$$

де в – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, %.

З 1.01.2016 року ставка єдиного внеску на загальнообов'язкове державне соціальне страхування встановлена залежно від класу професійного ризику виробництва і для бюджетної сфери β=22,0% .

$$В_{сз} = (10341 + 1137,5) \cdot \frac{22,0\%}{100\%} = 2525,3(грн.).$$

4. У спрощеному вигляді амортизаційні відрахування у загальному можуть бути розраховані за формулою:

$$А = \frac{Ц \cdot Т}{Ткор \cdot 12} \quad [грн], \tag{4.9}$$

де Ц – балансова вартість даного виду обладнання (приміщень), грн.;

$Т_{кор}$ – час користування;

Т – термін використання обладнання (приміщень), цілі місяці.

Згідно пункта 137.3.3 Податкового кодекса амортизація нараховується на

основні засоби вартістю понад 2500 грн.

$$А = \frac{12000 \cdot 1}{2 \cdot 12} = 500 \text{ грн.}$$

5. Норма витрат матеріалу – це плановий показник, який визначає максимально допустимі затрати відповідних ресурсів на виробництво одиниці продукції в умовах певного рівня техніки і організації виробництва.

Витрати на матеріали М, що були використані під час виконання даного етапу роботи, розраховуються по кожному виду матеріалів за формулою:

$$М = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i - \sum_{1}^{n} В_i \cdot Ц_в \text{ грн.,} \quad (4.10)$$

де $Н_i$ – витрати матеріалу *і*-го найменування, кг;

$Ц_i$ – вартість матеріалу *і*-го найменування, грн./кг.;

$К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15);

$В_i$ – маса відходів матеріалу *і*-го найменування, кг;

$Ц_в$ – ціна відходів матеріалу *і*-го найменування, грн/кг;

n – кількість видів матеріалів.

Інформацію про використані матеріали подамо у вигляді табл. 4.8.

Таблиця 4.8 – Матеріали, що використані на розробку

Найменування матеріалу	Ціна за одиницю, грн.	Витрачено	Вартість витраченого матеріалу, грн.
Флешка	120	1	120
Папір	75	1	75
Ручка	10	1	10
CD диск	10	1	10
Всього			215
З врахуванням коефіцієнта транспортування			236,5

6. До статті «Паливо та енергія на технологічні цілі» відносяться витрати на всі види палива й енергії, що безпосередньо використовуються у процесі

виробництва продукції. У даному випадку будемо враховувати лише витрати на електроенергію. Витрати на енергію визначаються на основі витрат на одиницю продукції та тарифів на енергію за допомогою залежності:

$$В_е = В \cdot П \cdot Ф \cdot К_п, \quad (4.11)$$

де В – вартість 1 кВт енергії, грн. В = 8,44 грн/кВт*год;

П – установлена потужність обладнання, кВт. При паяні використовується паяльник потужність П =400 Вт або П = 0,4 кВт;

Ф – фактична кількість годин роботи обладнання, год. Ф = 100 год;

$К_п$ – коефіцієнт використання потужності, $К_п$ = 0,65.

$$В_е = 8,44 \cdot 0,4 \cdot 100 \cdot 0,65 = 219,44 (грн).$$

7. Інші витрати $B_{ін}$ охоплюють: витрати на управління організацією, оплата службових відряджень, витрати на утримання, ремонт та експлуатацію основних засобів, витрати на опалення, освітлення, водопостачання, охорону праці тощо.

Інші витрати $B_{ін}$ можна прийняти як (100…300)% від суми основної заробітної плати розробників та робітників, які виконували дану МКНР, тобто:

$$B_{ін} = (1..3) \cdot (З + З_р). \quad (4.12)$$

$$B_{ін} = 1 \cdot (10341) = 10341 (грн.)$$

Сума всіх попередніх статей витрат дає витрати, які безпосередньо стосуються даного розділу МКНР

В = 10341+1137,5+2525,3+500+236,5+219,44+10341=25300,5 грн.

Загальна вартість всієї МКНР визначається за формулою:

$$B_{заг} = \frac{B}{\alpha} \quad (4.13)$$

$$B_{заг} = \frac{25300,5}{1} = 25300,5 (грн.)$$

Прогнозування загальних втрат ЗВ на виконання та впровадження

результатів виконаної МКНР здійснюється за формулою:

$$ЗВ = \frac{В}{\beta},$$

(4.14)

де $\beta -$ коефіцієнт, який характеризує стадію виконання даної НДР.

Оскільки, робота знаходиться на стадії розробки дослідного зразка, то коефіцієнт β = 0,9.

Звідси:

$$ЗВ = \frac{25300,5}{0,9} = 28112\,(грн.).$$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

У даному підрозділі кількісно спрогнозуємо, яку вигоду, зиск можна отримати у майбутньому від впровадження результатів виконаної наукової роботи. Розрахуємо збільшення чистого прибутку підприємства $ДП_i$, для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, за формулою

$$\Delta П_i = \sum_1^n \left(\Delta Ц_o \cdot N + Ц_o \cdot \Delta N\right)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\upsilon}{100}\right) \qquad (4.15)$$

де $\Delta Ц_0$ – покращення основного оціночного показника від впровадження результатів розробки у даному році.

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки:

Цо – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки:

л – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт *л* = 0,8333.

р – коефіцієнт, який враховує рентабельність продукту. р = 0,25;

х – ставка податку на прибуток. У 2019 році – 18%.

Припустимо, що при впроваджені результатів наукової розробки покращується якість, що дозволяє підвищити ціну його реалізації на 400 грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року на 120 шт., протягом другого року – на 150 шт., протягом третього року на 180 шт. Реалізація продукції до впровадження розробки складала 1 шт., а її ціна 1500 грн. Розрахуємо прибуток, яке отримає підприємство протягом трьох років.

$$\Delta П_1 = \left[400 \cdot 1 + (1500 + 400) \cdot 120\right] \cdot 0,833 \cdot 0,25 \cdot \left(1 + \frac{18}{100}\right) = 39016,8 (\text{грн.})$$

$$\Delta П_2 = \left[400 \cdot 1 + (1500 + 400) \cdot (120 + 150)\right] \cdot 0,833 \cdot 0,25 \cdot \left(1 + \frac{18}{100}\right) = 88034 (\text{грн.})$$

$$\Delta П_3 = \left[400 \cdot 1 + (1500 + 400) \cdot (120 + 150 + 180)\right] \cdot 0,833 \cdot 0,25 \cdot \left(1 + \frac{18}{100}\right) = 146457 (\text{грн.})$$

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Розрахуємо основні показники, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності. Теперішню вартість інвестицій PV, що вкладаються в наукову розробку приймемо рівну загальним витратам PV = ЗВ = 28112 грн.

Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$ згідно наступної формули:

$$E_{абс} = (ПП - PV) \tag{4.16}$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки, грн;

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t},$$

(4.17)

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДЦКР, грн;

Т – період часу, протягом якою виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,2;

t – період часу (в роках).

$$ПП = \frac{39016,8}{(1+0,2)^1} + \frac{88034}{(1+0,2)^2} + \frac{146457}{(1+0,2)^3} = 178797,92\,(грн.)\,.$$

$$\text{Е}_{абс} = (178797,92 - 28112) = 150686,24(грн.).$$

Оскільки $E_{абс} > 0$ то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього користуються формулою:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.20)$$

$T_ж$ – життєвий цикл наукової розробки, роки.

$$E_в = \sqrt[3]{1 + \frac{150686,24}{71768}} - 1 = 0,85 = 85\%$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді

визначається за формулою:

$$\tau = d + f,$$

(4.18)

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2019 році в Україні d= (0,14…0,2);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,1).

.

Так як $E_в > \tau_{\min}$ то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в} \qquad (4.19)$$

$$T_{ок} = \frac{1}{0,85} = 1,2 \text{ (роки)}$$

Так як $T_{ок} \leq$ 3...5-ти років, то фінансування даної наукової розробки в принципі є доцільним.

ВИСНОВКИ

Дипломна робота присвячена розробці і дослідженню розпізнаванню типів автомобілів нейромережевою системою, а саме розробку згорткової нейромережі на основі Google Cloud Vision API.

В дипломній роботі була поставлена мета суттєвого підвищення ефективності процесів розпізнавання об’єктів, збільшення точності та покращення методів фільтрації зображення.

Для досягнення поставленої мети були поставлені і виконані такі задачі:

– проведено аналіз існуючих методів розпізнавання об’єктів нейромережевими системами.

– вперше запропоновано метод, механізму просторової локалізації у згортковій нейронній мережі, який на відміну від існуючих, відмінно масштабуються і можуть використовуватися для розпізнавання образів, якого завгодно великого масштабу.

Було проаналізовано методи та технології навчання нейронних мереж. Розглянуто особливості архітектури мереж та принципи їх тренування. Проаналізовано ефективність архітектур і методів навчання для вирішення конкретних задач.

Проведено тестування для детальної демонстрації роботи програми, а також порівняння ефективності способу навчання нейронної мережі.

В економічній частині було оцінено економічний потенціал розробки нейромережевої системи розпізнавання типів автомобілів на зображеннях. Наведено порівняння нової розробки з аналогом, яке показало, що новий виріб буде набагато краще за аналог по технічним і економічним параметрам.

Оцінка якості і конкурентоспроможності показали, що нова розробка є більш якісною і конкорунтноспроможнішою ніж аналоги, які існують на ринку.

Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 25300,5 грн. Загальна ж величина витрат на виконання та впровадження результатів даної НДР буде складати 28112 грн.

Вкладені інвестиції в даний проект окупляться через 1,2 роки при прогнозованому прибутку 141275,68 грн. за три роки.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016. Mode of access : http://www.deeplearningbook.org.

2. Борисов Ю.И. Нейросетевые методы обработки информации и средства их программно-аппаратной поддержки / Борисов Ю.И, Кашкаров В.М., Сорокин С.А. // Открытые системы. – 1997.– № 4. – С. 38 – 40.

3. Simon Haykin . Neural Network a comprehensive foundation(2nd edition) / Simon Haykin - Prentice Hall, 842 pages, 2013

4. *Brownlee, J.* (2016, July 1). Object Recognition with Convolutional Neural Networks in the Keras Deep Learning Library. From Machine Learning Mastery: http://machinelearn-ingmastery.com/object- recognition-convolutional-neural-networks-keras-deeplearning-library.

5. *Eleonora Cagli1, Cecile Dumas, Emmanuel Prouff*. Convolutional Neural Networks with Data Augmentation against Jitter-Based Countermeasure International on Conference on Multimedia, pages 6-7. ACM, 2014.

6. Rodrigo Benenson. [Electronic resource] // What is the class of this image?, Last updated on 2016-02-22: – Mode of access: WWW.URL: http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_r esults.html#43494641522d3130.

7. Принципи ущільнення та перетворення зображення: Монографія/ В. П. Кожем‟ко, А. С. Васюра, Н. В. Сачанюк-Кавецька, О. В. Кириченко. – Вінниця : ВНТУ, 2010. – С.261.

8. Kyrychenko O. V. Transformation and compression of images in optoelectronic logic and time environments / Kozhemiako V. P., Maidaniuk V. P., Kyrychenko O. V. // Naukaistudia. – № 21 (89). – 2013. – P. 35–43. − ISSN 1561- 6894.

9. Кожем‟яко В. П. Реализация методов определения расстояний в системах технического зрения / В. П. Кожем‟яко, С. М. Білан // Электронное моделирование. – 1997. – № 4. – С. 52-63.

10. Грузман И. С. Алгоритмы распознавания объектов, устойчивые к геометрическим искажениям: сдвигу, масштабу, повороту / И. С. Грузман, В. Г. Никитин // Автометрия. – 2004. – № 3. – С. 46-53.

11. Прэтт У. Цифровая обработка зображений. / У. Прэтт. – М . : Мир, 1982. – 790 с.

12. Русин Б. П. Системи синтезу, обробки та розпізнання складноструктурованих зображень / Б. П. Русин. – Львів : Вертикаль, 1997. – 262 с.

13. Охоцинский Д. Е. Системы технического зрения / Д. Е. Охоцинский, В. М. Златкис. – М. : Наука, 1991. – 201 с.

14. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. / А.Н. Горбань, Д.А. Россиев – Н.: Наука, 2006. – 276с.

15. Hong S.G., Kim S.W. and Lee J.J., 2015. The Minimum Cost Path Finding Algorithm Using a Hopfield Type Neural Network, Proceedings IEEE International Conference on Fuzzy Systems 4, 719–726.

16. Лисе А.А., Степанов М.В. Нейронные сети и нейрокомпьютеры. / А.А. Лисе, М.В. Степанов // Учеб. пособие. ГЭТУ. - СПб., 2009. 64 с.

17. Тархов Д.А. Нейронные сети. Модели и алгоритмы. – М.: Радиотехника, 2010.– 82 с.

18. Царегородцев В.Г. Перспективы распараллеливания программ нейросетевого анализа и обработки данных / В.Г. Царегородцев // Материалы III Всерос. конф. «Математика, информатика, управление – 2008». – Иркутск, 2014. – С. 110-117.

19. Рутковская Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский. - М. Горячая линия-Телеком 2004. – 112 с.

20. Галушкин А.И. Теория нейронных сетей. / А.И. Галушкин - М.: ИПРЖР, 2010. – 348 с.

21. Боголюбов Д. П., Чанкин А. А., Стемиковская К. В. Реализация алгоритма обучения самоорганизующихся карт Кохонена на графических

процесорах // Промышленные АСУ и контроллеры. 2012. № 10. С. 30-35

22. Asanovic, Krste et al. (Jan 18, 2016). The Landscape of Parallel Computing Research: A View from Berkeley University of California, Berkeley. Technical Report No. UCB/EECS-2016-183

23. Barbara Chapman, Gabriele Jost, Ruud van der Pas. Using OpenMP: portable shared memory parallel programming (Scientific and Engineering Computation). Cambridge, Massachusetts: The MIT Press., 2008. - 353 pp.

24. Нейромережеве відображення дійсності. – Режим доступу: http://studies.in.ua/mpd_seminar/1313-neyrmerezh.html – Дата доступу: 29.10.2019

25. Моделі нейронних мереж. – Режим доступу: https://studme.com.ua/1246122010028/neural/models.htm – Дата доступу: 27.10.2019

26. Cavallaro A. Video tracking/ A.,E. Maggio, Cavallaro A. – Chichester,West Sussex, UK: Wiley, 2010.

27. Challa S. Fundamentals of object tracking/Chall S., Mark R. Morelande, Darko Mušicki // Cambridge UK: Cambridge University Press, 2011

28. Martínez-Martín E. Robust motion detection in real-life scenarios./ A. Pobil, E. Martínez-Martín// – London: Springer, 2012.

29. Компьтерное зрение.Современний подход/Форсайт Д., Понс Ж.,:Пер. С англ. – М.:Издательский дом «Вильямс», 2004. – 928с.

30. Yilmaz A. Object tracking: A survey /A. Yilmaz, O. Javed and M. Shah//ACM Comput. Surv, 2006, vol. 38, no. 4,

31. Борисенко Д. И. Методы поиска угловых особенностей наизображениях // Молодой ученый, 2011. — №5. Т.1. — С. 120-123.

32. Цифровая обработка видеоизображений / А. А.Лукьяница, А.Г.Шишкин. – М.: «Ай-Эс-Эс Пресс», 2009. – 518с.

33. Joshi Prateek, Opencv by Example/ Pratek J. – S.l.: Packt Limited, 2016.

34. Федяев О.И., Махно Ю.С. Система распознавания зашумлённых и искажённых графических образов на основе нейронной сети типа неокогнитрон // Одиннадцатая национальная конференция по искусственному интеллекту с

международным участием КИИ-2008: Труды конференции. Т. 3. – М .: ЛЕНАНД, 2008. – 464 с.

35. OpenCV – библиотека компьютерного зрения с открытым исходным кодом [Электронный ресурс]. – Режим доступа: http://software.intel.com/en-us/articles/...– Дата доступу: 27.10.2019

36. Viola P, JonesM(2001) Rapid object detection using a boosted cascade of simple features. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition. pages 511–518, Kauai, HI

37. Rainer Lienhart, Alexander Kuranov, Vadim Pisarevsky. Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection.: Intel

38. Мальцев А. Использование каскада Хаара для сравнения изображений [Электронный ресурс] / Мальцев Антон. — Режим доступа: https://habrahabr.ru/post/198338/ Дата доступу: 27.10.2019

39. Зуєв А.А. Методи виділення контурів на зображеннях / А.А. Зуєв, Г.Є. Нечепоренко // Міжнародна наукова конференція MicroCAD: Секція № 8 - Мікропроцесорна техніка в автоматіці та пріладобудуванні - НТУ ХПІ, 2012. - С. 108.

40. Дэвид А. Форсайт, Джин Понс [Computer Vision: A Modern Approach Компьютерное зрение. Современный подход]. — М.: «Вильямс», 2004. — 928 с.

41. Джордж Стокман, Линда Шапиро [Computer Vision Компьютерное зрение]. — М.: Бином. Лаборатория знаний, 2006. — 752 с.

42. Стокман Дж. Компьютерное зрение - Computer Vision / Джордж Стокман, Линда Шапиро. - М.: Бином. Лаборатория знаний, 2006. - 752 с.

43. Фомин Я. А. Распознавание образов: теория и применения / Я. А. Фомин. - М.: ФАЗИС, 2012. - 429 с.

44. Применение нейронных сетей для задач классификации / Интернет ресурс – Режим доступа: http://www.basegroup.ru Дата доступу: 27.10.2019

45. Николенко С.И. Курс лекций по машинному обучению — слайды.

Електронний ресурс. Режим доступу: http://logic.pdmi.ras.ru/~sergei/index.php?page=mlaptu09 Дата доступу: 29.10.2019

46. Умяров Н.Х., Федяев О.И. Параметрическая модель свёрточной нейронной сети // VI международная научно-техническая конференция студентов, аспирантов и молодых научных работников «Информатика и компьютерные технологии»: Т. 2 – Донецк, ДонНТУ, 2010. – 292с.

47. Умяров Н.Х., Федяев О.И. Логическое и физическое представление архитектуры свёрточной нейронной сети // Інформаційні управляючі системи та комп'ютерний моніторинг (ІУС КМ – 2011) : ІІ Всеукраїнська науково-технічна конференція студентів, аспірантів та молодих вчених, 11-13 квітня 2011 р., м. Донецьк : зб. доп. у 3 т./ Донец. національ. техн. ун-т; редкол.: Є.О. Башков (голова) та ін. – Донецьк: ДонНТУ, 2011. – Т.3, с. 81-85

48. Битачевский Е.А. Использование нейронных сетей для распознавания визуальных образов [Материалы IV РНТ конференции «Вузовская наука Северо-Кавказского региона»: Сборник докладов.] / Е.А. Битачевский, В.В. Козуб - 2000. - С. 52-54.

49. Уоссермен Ф. Нейрокомпьютерная техника. Теория и практика. – М.: Мир, 1992

50. Кириченко О. В. Тимченко Л.І. Розробка методів для реалізації квантронного оброблення зображень / Тимченко Л. І., Кириченко О. В. // Приднепровский научный вестник. – №8 (144). – 2013. – С. 15–26. − ISSN 1561-6940.

51. Кириченко А. В. Аппаратная реализация квантронной обработки зображений / Кириченко А. В. Тимченко Л. И. // Современный научный вестник. – 2013. – №32 (171). – С. 31–42. − ISSN 1561-6886.

ДОДАТКИ

Додаток А

(обов’язковий)

ВНТУ

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

НЕЙРОМЕРЕЖЕВА СИСТЕМА РОЗПІЗНАВАННЯ ТИПІВ АВТОМОБІЛІВ НА ЗОБРАЖЕННЯХ

Студент групи 2АКІТ-18м Босенко Б.М.

“___” _____ 2019 р.

Керівник д.т.н., доцент Ковтун В.В.

“___” _____ 2019 р.

Вінниця 2019

Назва та галузь застосування

1.1. Назва – Розробка нейромережевої системи розпізнавання типів автомобілів.

1.2. Галузь застосування – Комп'ютеризовані системи відео спостереження.

2. Підстава для проведення розробки.

Тема магістерської кваліфікаційної роботи затверджена наказом по ВНТУ № 254 від "_02__" ____10____ 20 19 р.

3. Мета та призначення розробки.

Метою магістерської кваліфікаційної роботи є розробка та підвищення якості розпізнавання обєктів на зображеннях .

4. Вихідні дані для проведення розробки.

Магістерська кваліфікаційна робота виконується вперше. В ході проведення розробки повинні використовуватись такі документи:

1. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016. Mode of access : http://www.deeplearningbook.org.
2. Борисов Ю.И. Нейросетевые методы обработки информации и средства их программно-аппаратной поддержки / Борисов Ю.И, Кашкаров В.М., Сорокин С.А. // Открытые системы. – 1997.– № 4. – С. 38 – 40.
3. Simon Haykin . Neural Network a comprehensive foundation(2nd edition) / Simon Haykin - Prentice Hall, 842 pages, 2013

5. Вимоги до розробки.

5.1. Перелік головних функцій:

Програма повинна здійсювати розпізнавання типу автомобіля та здійснювати візуалізацію отриманого результату.

5.2. Основні технічні вимоги до розробки.

5.2.1. Вимоги до програмної платформи:

- WINDOWS 2007\XP;
- Visual Studio 2010.

5.2.2. Умови експлуатації системи:

- робота на стандартних ПЕОМ в приміщеннях зі стандартними умовами;
- можливість цілодобового функціонування системи;
- текст програмного забезпечення системи є цілком закритим.

6. Економічні показники

До економічних показників входять:

- термін окупності не більше 1,2 років;
- економічний ефект не менше 15068 грн..;
- інші економічні переваги у порівнянні з аналогами.

7. Стадії та етапи розробки.

7.1 Пояснювальна записка:

1	Аналіз методів оцінювання ризику прийняття керівних рішень при управлінні розгалужено-циклічними технологічними процесами. Постановка задач дослідження	12.09.2019р.
2	Удосконалення технології прийняття рішень при управлінні розгалужено-циклічними технологічними процесами	22.09.2019р.
3	Практична реалізація та аналіз отриманих результатів	3.10.2019р.
4	Підготовка економічної частини	12.11.2019р.
5	Апробація результатів дослідження	22.11.2019р.
6	Публікації	
7	Оформлення пояснювальної записки, графічного матеріалу і презентації	30.11.2019р.
8	Захист МКР	17.12.2019р.

7.2 Графічні матеріали:

- алгоритм прийняття рішення при управлінні РЦТП: «01 » 12. 2019 р.
- UML діаграма варіантів використання: «01 » 12. 2019 р.
- UML-діаграма діяльності: «03 » 12. 2019 р..
- UML-діаграма класів: «03 » 12. 2019 р.
- вигляд екрану «Головне меню ІТ «ТР Modeling»»: «06 » 12. 2019 р.
- вигляд екрану «Total Risk»: «06 » 12. 2019 р.
- 8. Порядок контролю і приймання.

8.1. Хід виконання магістерської кваліфікаційної роботи контролюється керівником роботи, консультантами з економічної частини. Рубіжний контроль провести до «7» грудня 2019 р.

8.2. Атестація проекту здійснюється на попередньому захисті. Попередній захист магістерської кваліфікаційної роботи провести до «10» грудня 2019 р.

8.3. Підсумкове рішення щодо оцінки якості виконання магістерської кваліфікаційної роботи приймається на засіданні ДЕК. Захист магістерської кваліфікаційної роботи провести «17» грудня 2019 р.

Додаток Б

(обов´язковий)

лістинг програми

Файл analize_type_car

```
import logging
import sys
import keras.layers.convolutional
import keras.layers.normalization
from keras import backend as K
from keras.models import
load_model import seaborn as sns
sns.set_style("whitegrid")
import matplotlib.pyplot as plt
import operator
from functools import reduce
import numpy as np
import yaml
import imp
from run_training import make_paths_absolute
import os
import pprint
import scipy.misc
import scipy.stats
import glob
logging.basicConfig(format='%(asctime)s %(levelname)s %(message)s',
                    level=logging.DEBUG,
                    stream=sys.stdout)
from msthesis_utils import make_mosaic
def get_activations(model, layer_index, X_batch):
    """Get activation of one layer for one input."""
    get_activations = K.function([model.layers[0].input, K.learning_phase()],
                                 [model.layers[layer_index].output, ])
    activations = get_activations([X_batch, 0])
    return activations
def show_conv_act_distrib(model, X, show_feature_maps=False):
    """Show the distribution of convolutional layers for one input."""
    X_train = np.array([X])
    layer_index = 0
    activations_by_layer = []
    labels = []
    for layer_index in range(len(model.layers)):
        act = get_activations(model, layer_index, X_train)[0]
        if show_feature_maps:
            scipy.misc.imshow(X_train[0])
```

```
            mosaik = make_mosaic(act[0], 8, 4)
            scipy.misc.imshow(mosaik)
        data = act[0].flatten()
        if isinstance(model.layers[layer_index],
                      keras.layers.convolutional.Conv2D):
            print("\tlayer {}: len(data)={}".format(layer_index, len(data)))


      activations_by_layer.append(data)
      labels.append(layer_index)
                    layer_index += 1

    # Activations
    for label, fw in enumerate(activations_by_layer):
        print("99% filter weight interval of layer {}: [{:.2f}, {:.2f}]"
              .format(label, np.percentile(fw, 0.5), np.percentile(fw, 99.5)))

    f, ax1 = plt.subplots(1, 1)
    p = sns.violinplot(data=activations_by_layer, orient="v",
                       palette=sns.color_palette(palette="RdBu", n_colors=1),
                       ax=ax1)
    p.tick_params(labelsize=16)
    ax1.set_xticklabels(labels)
    ax1.set_title('Convolution activations by layer')
    sns.plt.show()
def show_conv_weight_dist(model, small_thres=10**-6):
    """Show the distribution of conv weights of model."""
    layer_index = 0
    filter_weights = []
    bias_weights = []
    filter_weight_ranges = []
    labels = []
    for layer in model.layers:
        if not isinstance(layer, keras.layers.convolutional.Conv2D):
            layer_index += 1
            continue
        weights = layer.get_weights()

        # Filter
        print("{}: {} filter weights in {}th layer"
              .format(weights[0].shape,
                      reduce(operator.mul, weights[0].shape, 1),
                      layer_index))

        # Measure distribution (replacement by 1x1 filters)
        ranges = []
        w, h, range_i, range_j = weights[0].shape
```

```
    if w > 1 or h > 1:
        for i in range(range_i):  # one filter, but different
            channel for j in range(range_j): # different filters
                elements = weights[0][:, :, i, j].flatten()
                ranges.append(elements.max() - elements.min())


        ranges = np.array(ranges)
    # measure distribution
    data = weights[0].flatten()
    labels.append(layer_index)
    filter_weights.append(data)
    data_small = np.array([el for el in data if abs(el) < small_thres])
    print("< {}: {}".format(small_thres, len(data_small)))
    # Bias
    if len(weights) > 1:
        print("{}: {} bias weights in {}th layer"
              .format(weights[1].shape,
                      reduce(operator.mul, weights[1].shape, 1),
                      layer_index))
        data = weights[1].flatten()
        bias_weights.append(data)
        data_small = np.array([el for el in data if abs(el) < small_thres])
        print("< {}: {}".format(small_thres, len(data_small)))
    else:
        print("No bias in layer {}".format(layer_index))
    layer_index += 1

labels = [1, 3, 5, 7, 9, 11, 13, 15]  # baseline

# Filter weight ranges
f, ax1 = plt.subplots(1, 1)
p = sns.violinplot(data=filter_weight_ranges, orient="v",
                   palette=sns.color_palette(palette="RdBu", n_colors=1),
                   ax=ax1)
p.tick_params(labelsize=16)
p.set_xlabel('Layer', fontsize=20)
p.set_ylabel('Weight range', fontsize=20)
ax1.set_xticklabels(labels)
ax1.set_title('Filter weight ranges by layer')
sns.plt.show()

# Filter weights
for label, fw in enumerate(filter_weights):
    print("99% filter weight interval of layer {}: [{:.2f}, {:.2f}]"
          .format(label, np.percentile(fw, 0.5), np.percentile(fw, 99.5)))
```

```
    f, ax1 = plt.subplots(1, 1)
    p = sns.violinplot(data=filter_weights, orient="v",
                       palette=sns.color_palette(palette="RdBu", n_colors=1),
                       ax=ax1)
    p.tick_params(labelsize=16)
    p.set_xlabel('Layer', fontsize=20)
    ax1.set_xticklabels(labels)
    ax1.set_title('Filter weight distribution by layer')


    sns.plt.show()
    # Bias weights
    for label, fw in enumerate(bias_weights):
        print("99% bias weight interval of layer {}: [{:.2f}, {:.2f}]"
              .format(label, np.percentile(fw, 0.5), np.percentile(fw, 99.5)))

    f, ax1 = plt.subplots(1, 1)
    p = sns.violinplot(data=bias_weights[:], orient="v",
                       palette=sns.color_palette(palette="RdBu", n_colors=1),
                       ax=ax1)
    p.tick_params(labelsize=16)
    p.set_xlabel('Layer', fontsize=20)
    ax1.set_xticklabels(labels[:])
    ax1.set_title('Bias weight distribution by layer')
    sns.plt.show()


def show_batchnorm_weight_dist(model):
    """Show the distribution of batch norm weighs for one model."""
    # analyze
    gamma_weights = []
    beta_weights = []
    layer_index = 0
    labels = []
    for layer in model.layers:
        if isinstance(layer, keras.layers.normalization.BatchNormalization):
            labels.append(layer_index)
            weights = layer.get_weights()
            data = weights[0].flatten()
            gamma_weights.append(data)
            data = weights[1].flatten()
            beta_weights.append(data)
        layer_index += 1

    labels = [2, 4, 6, 8, 10, 12, 14, 16]  #
```

```
    baseline # Gamma weights
    if len(gamma_weights) > 0:
        for label, fw in zip(labels, gamma_weights):
            print("99% gamma interval of layer {}: [{:.2f}, {:.2f}]"
                  .format(label,
                          np.percentile(fw, 0.5),
                          np.percentile(fw, 99.5)))

        f, ax1 = plt.subplots(1, 1)
        p = sns.violinplot(data=gamma_weights, orient="v",
                           palette=sns.color_palette(palette="RdBu",

                                                     n_colors=1),
                           ax=ax1)
        p.tick_params(labelsize=16)
        p.set_xlabel('Layer', fontsize=20)
        ax1.set_xticklabels(labels)
        ax1.set_title('Gamma distribution by layer')
        sns.plt.show()

    # beta weights
    if len(beta_weights) > 0:
        for label, fw in zip(labels, beta_weights):
            print("99% beta interval of layer {}: [{:.2f}, {:.2f}]"
                  .format(label,
                          np.percentile(fw, 0.5),
                          np.percentile(fw, 99.5)))

        f, ax1 = plt.subplots(1, 1)
        p = sns.violinplot(data=beta_weights, orient="v",
                           palette=sns.color_palette(palette="RdBu",
                                                     n_colors=1),
                           ax=ax1)
        p.tick_params(labelsize=16)
        p.set_xlabel('Layer', fontsize=20)
        ax1.set_xticklabels(labels)
        ax1.set_title('Beta distribution by layer')
        sns.plt.show()


def main(config, data_module, model_path, image_fname):
    """Run analysis."""
    model = load_model(model_path)

    print("## Activation analyzation")
```

```
    if image_fname is not None:
        image = scipy.misc.imread(image_fname)
    else:
        data = data_module.load_data(config)
        X_train = data['x_train']
        X_test = data['x_test']
        # y_train = data['y_train']
        # y_test = data['y_test']

        X_train = data_module.preprocess(X_train)
        X_test = data_module.preprocess(X_test)
        image = X_train[0]

    print("## Weight analyzation")
    show_conv_weight_dist(model)
    print("## BN analyzation")
    show_batchnorm_weight_dist(model)


def get_parser():
    """Get parser object for script xy.py."""
    from argparse import ArgumentParser, ArgumentDefaultsHelpFormatter
    parser = ArgumentParser(description=__doc__,
                            formatter_class=ArgumentDefaultsHelpFormatter)
    parser.add_argument("-f", "--file",
                        dest="filename",
                        help="Experiment yaml file",
                        required=True,
                        metavar="FILE")
    parser.add_argument("--model",
                        dest="model_path",
                        help="Model h5 file",
                        metavar="FILE")
    parser.add_argument("--image",
                        dest="image_fname",
                        help="A single image",
                        metavar="FILE")
    return parser


if __name__ == "__main__":
    args = get_parser().parse_args()
    # Read YAML experiment definition file
    with open(args.filename, 'r') as stream:
        config = yaml.load(stream)
    # Make paths absolute
```

```
config = make_paths_absolute(os.path.dirname(args.filename),
                             config)

# Print experiment file
pp = pprint.PrettyPrinter(indent=4)
pp.pprint(config)

# Load data module
dpath = config['dataset']['script_path']
sys.path.insert(1, os.path.dirname(dpath))
data = imp.load_source('data', config['dataset']['script_path'])

if args.model_path is not None:
    model_path = args.model_path
else:
    artifacts_path = config['train']['artifacts_path']
    model_path = os.path.basename(config['train']['artifacts_path'])
    model_paths = glob.glob("{}/*.h5".format(artifacts_path))
    model_paths = [m for m in model_paths if "_chk.h5" not in m]
    model_path = model_paths[0]
logging.info("Take {}".format(model_path))
main(config, data, type_path, args.image_fname)
```

Додаток В
(Обов'язковий)

ЗАТВЕРДЖУЮ
Завідувач кафедри КСУ
д.т.н., проф. В.М. Дубовой
_____
« _____» _____2019 р.

ПЕРЕЛІК
ГРАФІЧНИХ МАТЕРІАЛІВ

для захисту магістерської кваліфікаційної роботи
на тему

РОЗРОБКА НЕЙРОМЕРЕЖЕВОЇ СИСТЕМИ РОЗПІЗНАВАННЯ ТИПІВ АВТОМОБІЛІВ НА ЗОБРАЖЕННЯХ

1. Мета і задачі дослідження
2. Модель одношарової штучної нейронної мережі.
3. Порівняння алгоритмів навчання нейромереж
4. Багатошарова нейронна мережа
5. Формування карти ознак. Операція згортки
6. Процес навчання мережі
7. Проблема перенавчання мережі
8. Графік залежності помилки нейронної мережі від кількості ітерацій при її тестуванні за допомогою нормалізованої вибірки
9. Архітектура згорткової нейронної мережі
10. Логічна модель згорткової нейр-мережі у вигляді діаграми UML
11. Вихідне графічне зображення для тестування
12. Розподіл об'єктів машин на типи синя рамка «car» і зелена рамка «truck»

Розробив: Босенко Б.М.
_____ _____
(підпис) (дата)
Перевірив: Ковтун В.В.
_____ _____
(підпис) (дата)
Рецензент: Довгалець С.М.
_____ _____
(підпис) (дата)

Вінниця 2019

1 МЕТА ТА ЗАВДАННЯ ДОСЛІДЖЕННЯ

МЕТА ТА ЗАВДАННЯ ДОСЛІДЖЕННЯ

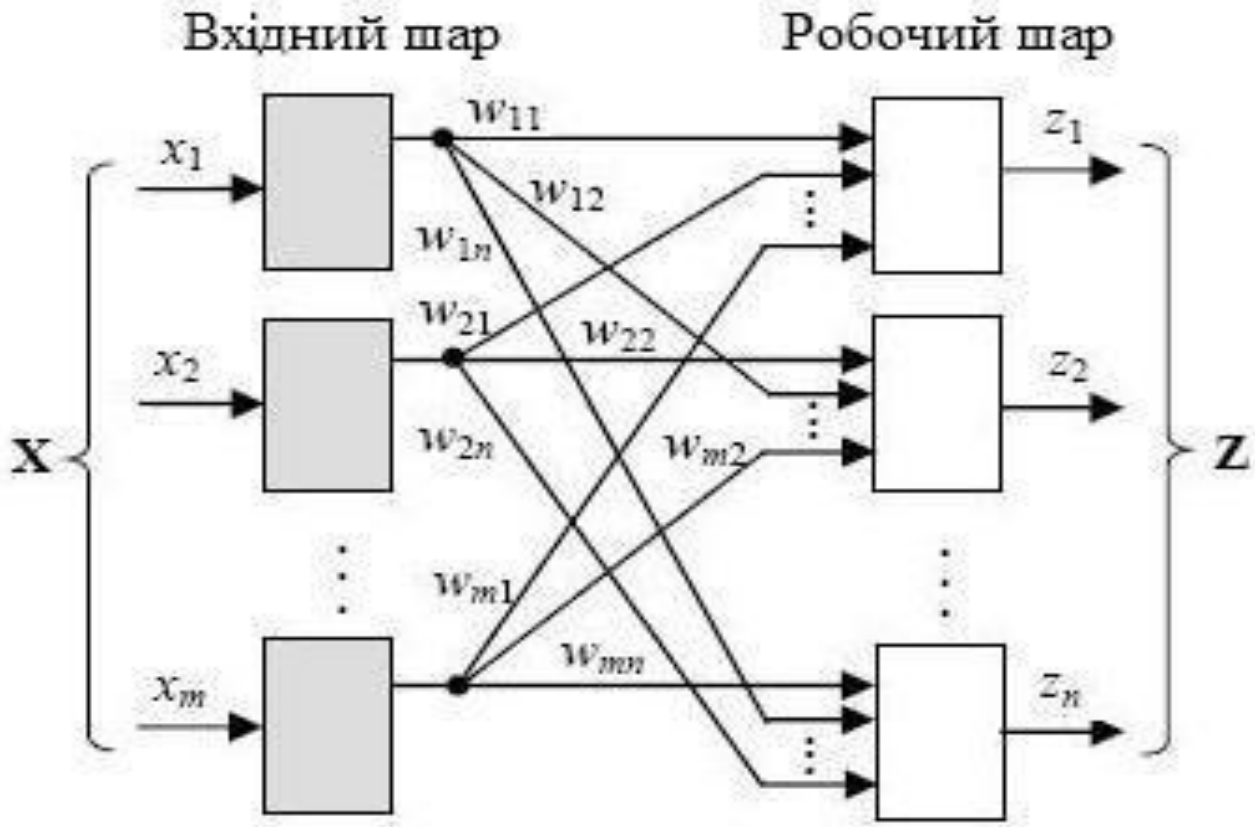
Мета – є розробка, проектування і тестування удосконаленої нейронної мережі для полегшення процедури розпізнавання образів

Завдання:

- Аналіз об'єкту дослідження
- Розв'язання науково-технічної задачі покращення методів розпізнавання типів автомобілів на зображенні.
- Вибір алгоритму навчання нейромережі
- Розробка програмного забезпечення системи
- Тестування програми

3

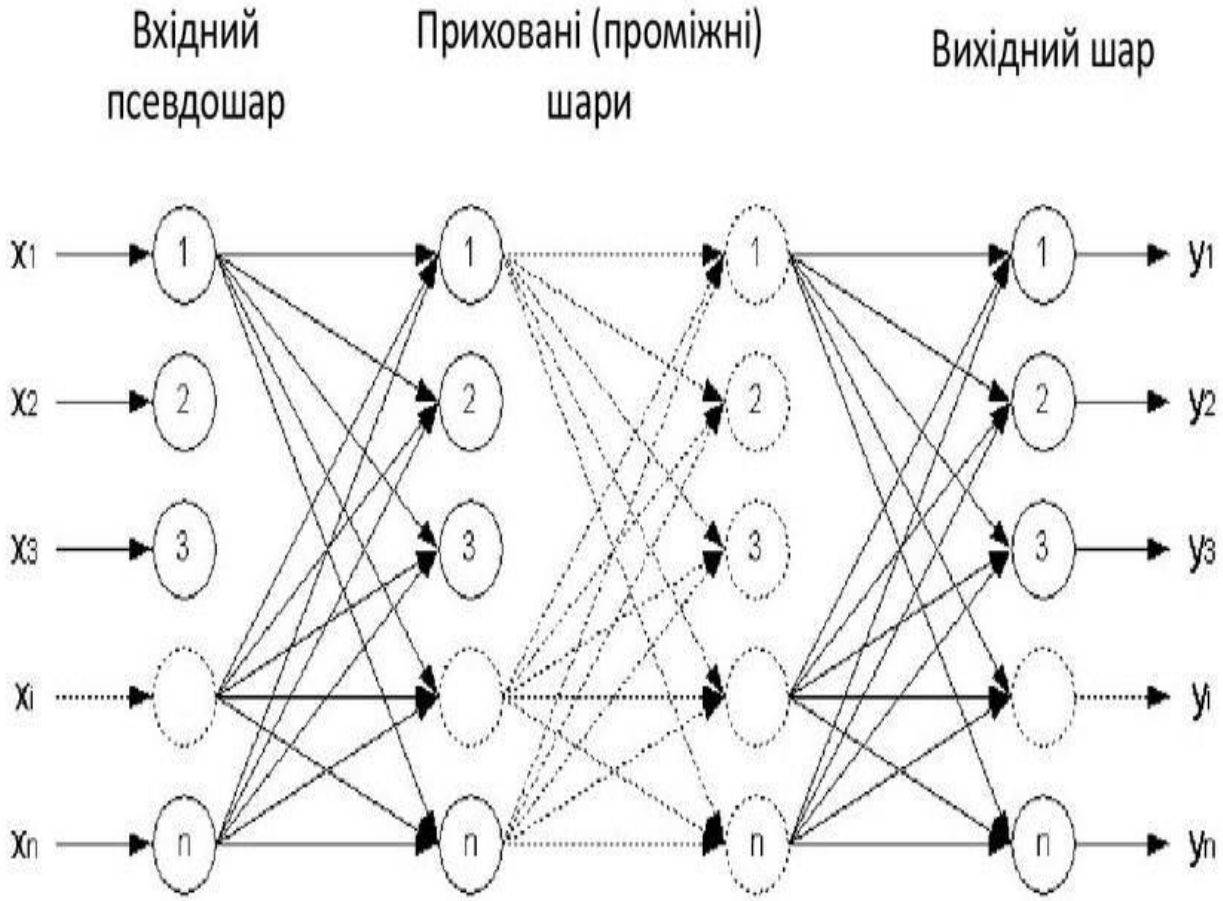
2 МОДЕЛЬ ОДНОШАРОВОЇ ШТУЧНОЇ МЕРЕЖІ



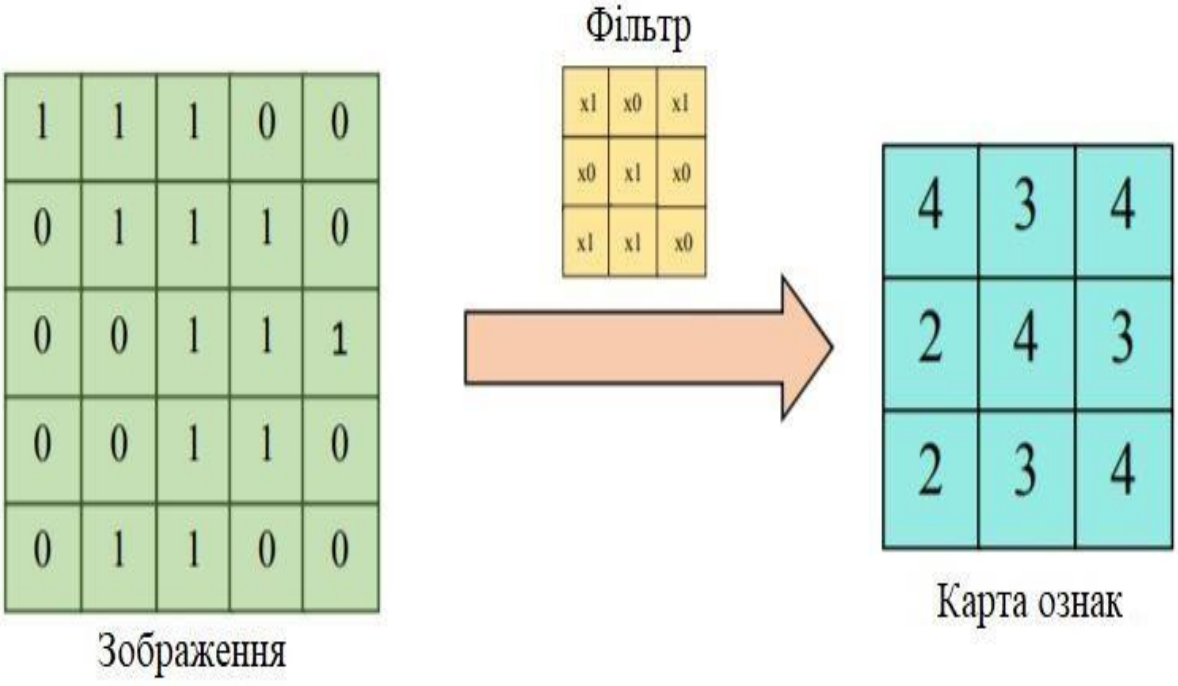
3 ПОРІВНЯННЯ АЛГОРИТМІВ НАВЧАННЯ НЕЙРОМЕРЕЖ

Типи навчання	Правило навчання	Архітектура	Алгоритм навчання	Задача
З вчителем	Корекція помилки	Одношаровий і багатошаровий перцептрон	Алгоритми навчання перцептрона Зворотне поширення	Класифікація образів Аппроксимація функцій Передбачення, управління
	Больцман	Рекурентна	Алгоритм навчання Больцмана	Класифікація образів
	Хебб	Багатошарова прямого розповсюдження	Лінійний дискримінантний аналізанализ	Аналіз даних Класифікація образів
	Змагальні	Змагальння	Векторне квантування	Категоризація всередині класу Стиснення даних
		Мережа ART	ARTMap	Класифікація образів
Без вчителя	Корекція помилки	Багатошарова прямого розповсюдження	Проекція Саммона	Категоризація всередині класу Аналіз даних
	Хебб	Прямого поширення або змагання	Аналіз головних компонентів	Аналіз даних Стиснення даних
		Мережа Хопфілда	Навчання асоціативної пам'яті	Асоціативна пам'ять
	Змагальні	Змагальння	Векторне квантування	Категоризація Стиснення даних
		Мережа Кохонена	SOM Кохонена	Категоризація Аналіз даних
		Мережа ART	ART1, ART2	Категоризація
Змішані	Коррекция ошибки и соревнование	Мережа RBF	Алгоритм навчання RBF	Класифікація образів Апроксимація функцій Передбачення, управління

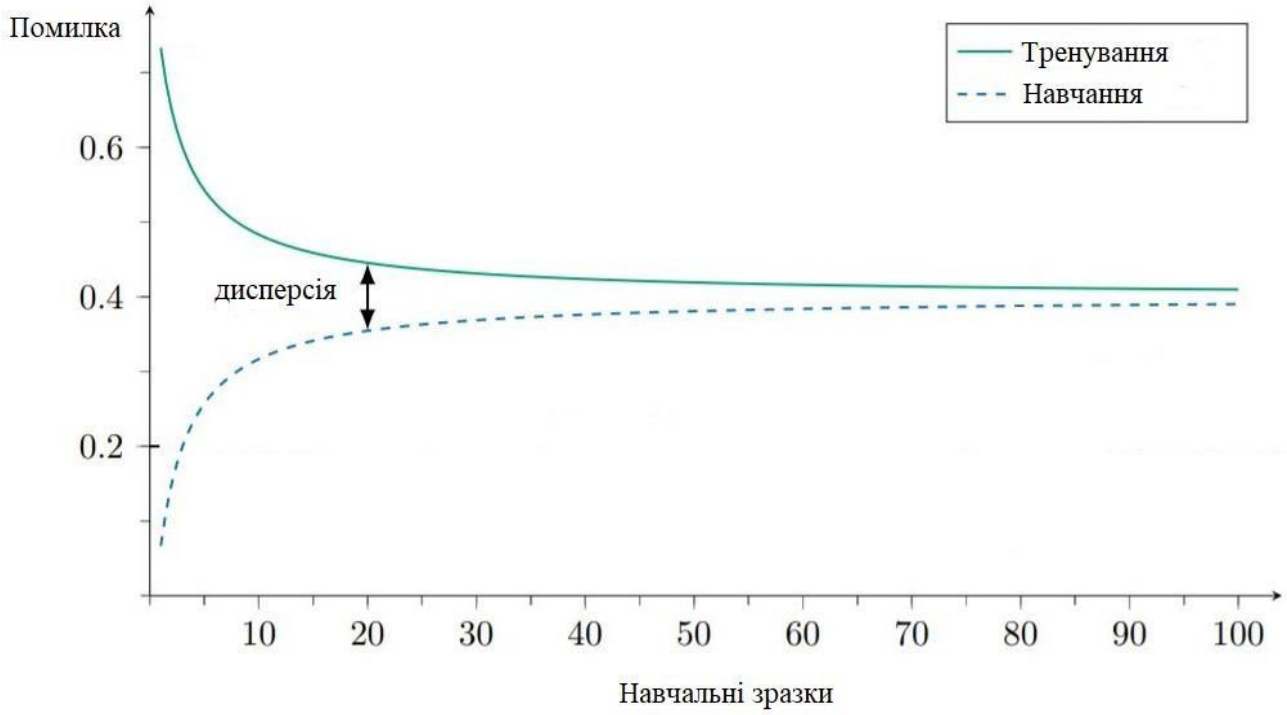
4 БАГАТОШАРОВА НЕЙРОННА МЕРЕЖА



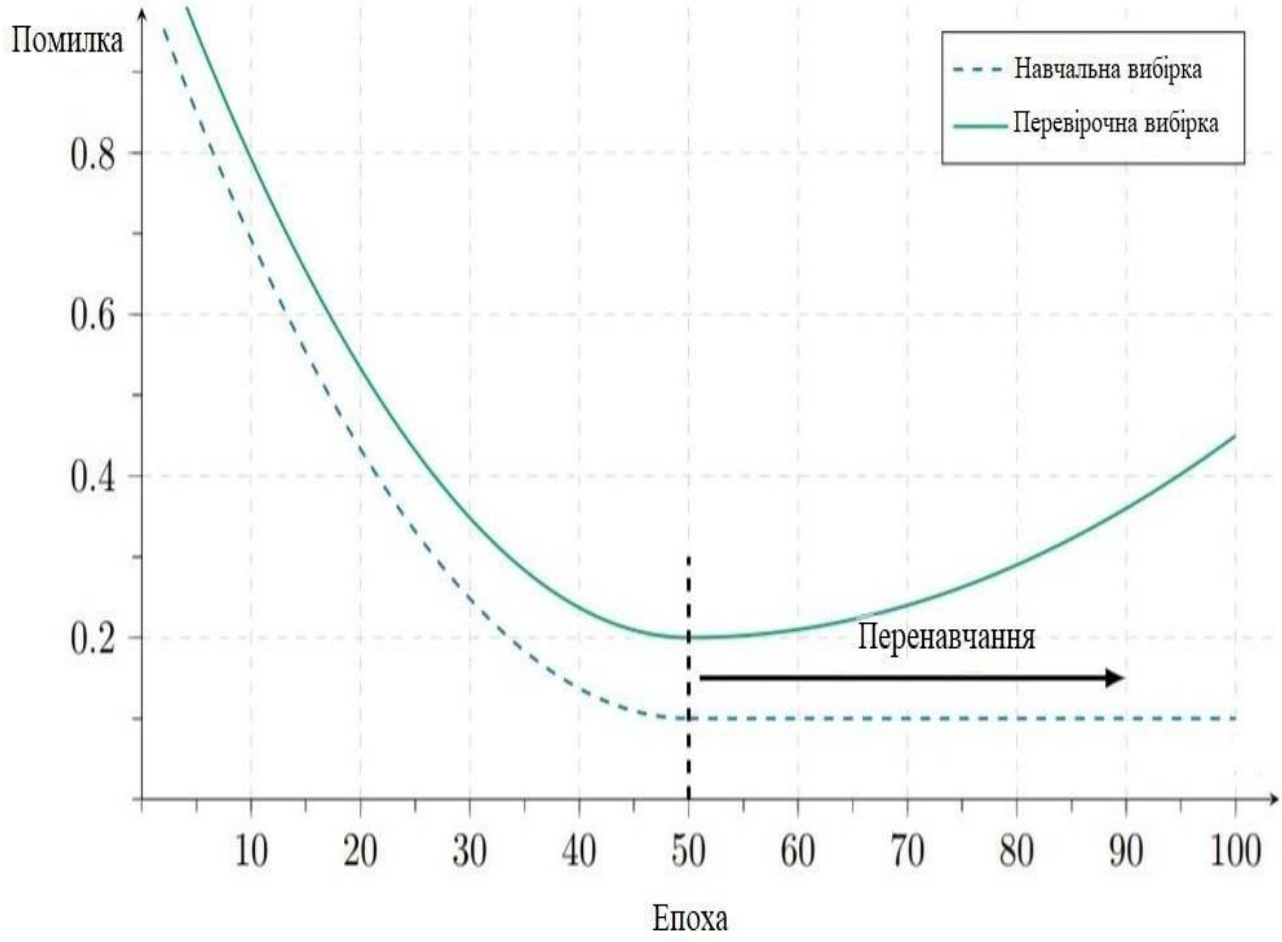
5 ФОРМУВАННЯ КАРТИ ОЗНАК. ОПЕРАЦІЯ ЗГОРТКИ



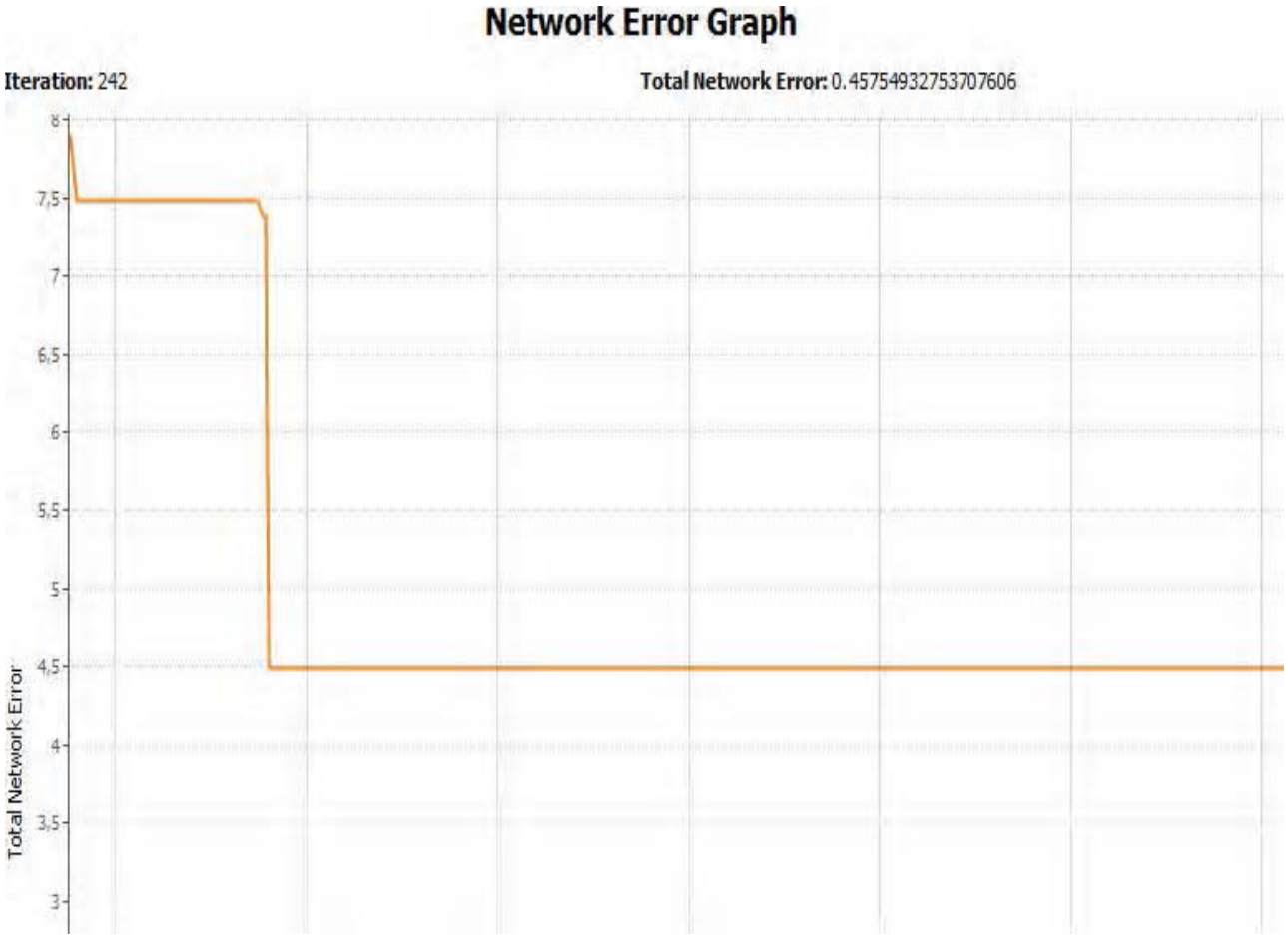
6 ПРОЦЕС НАВЧАННЯ МЕРЕЖІ



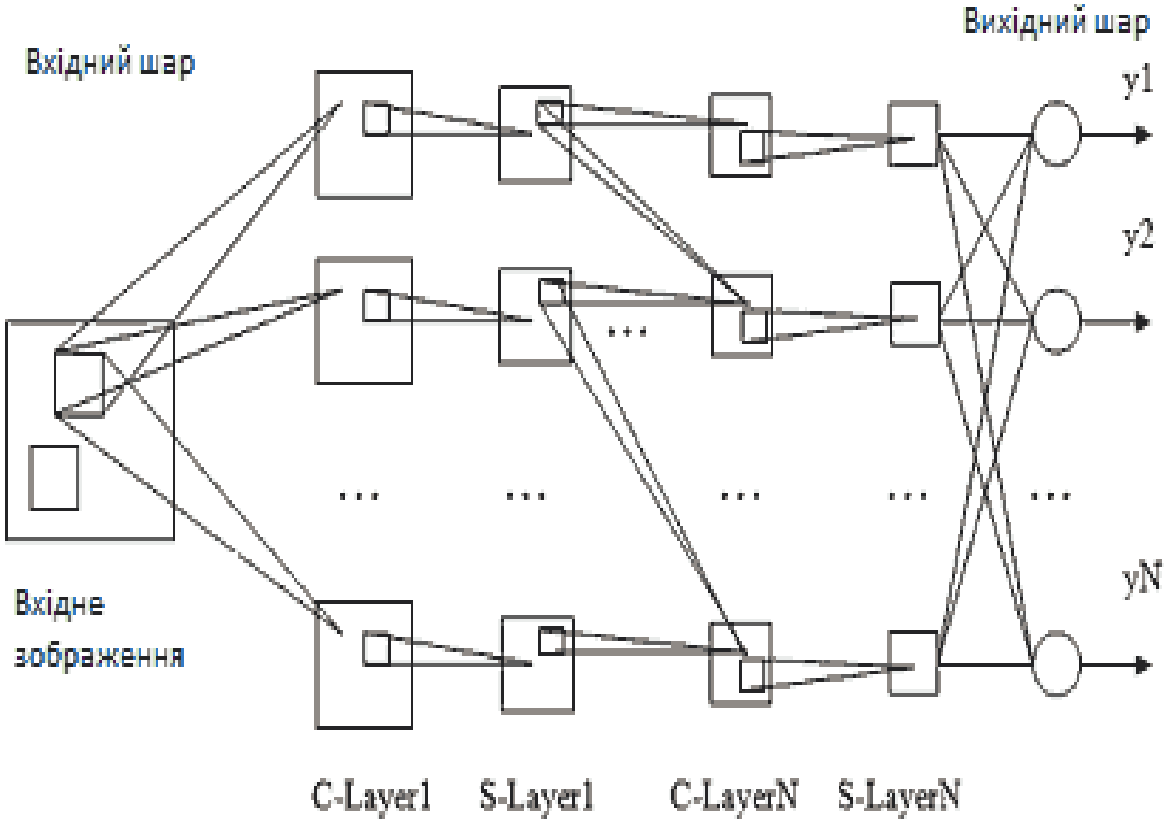
7 ПРОБЛЕМА ПЕРЕНАВЧАННЯ МЕРЕЖІ



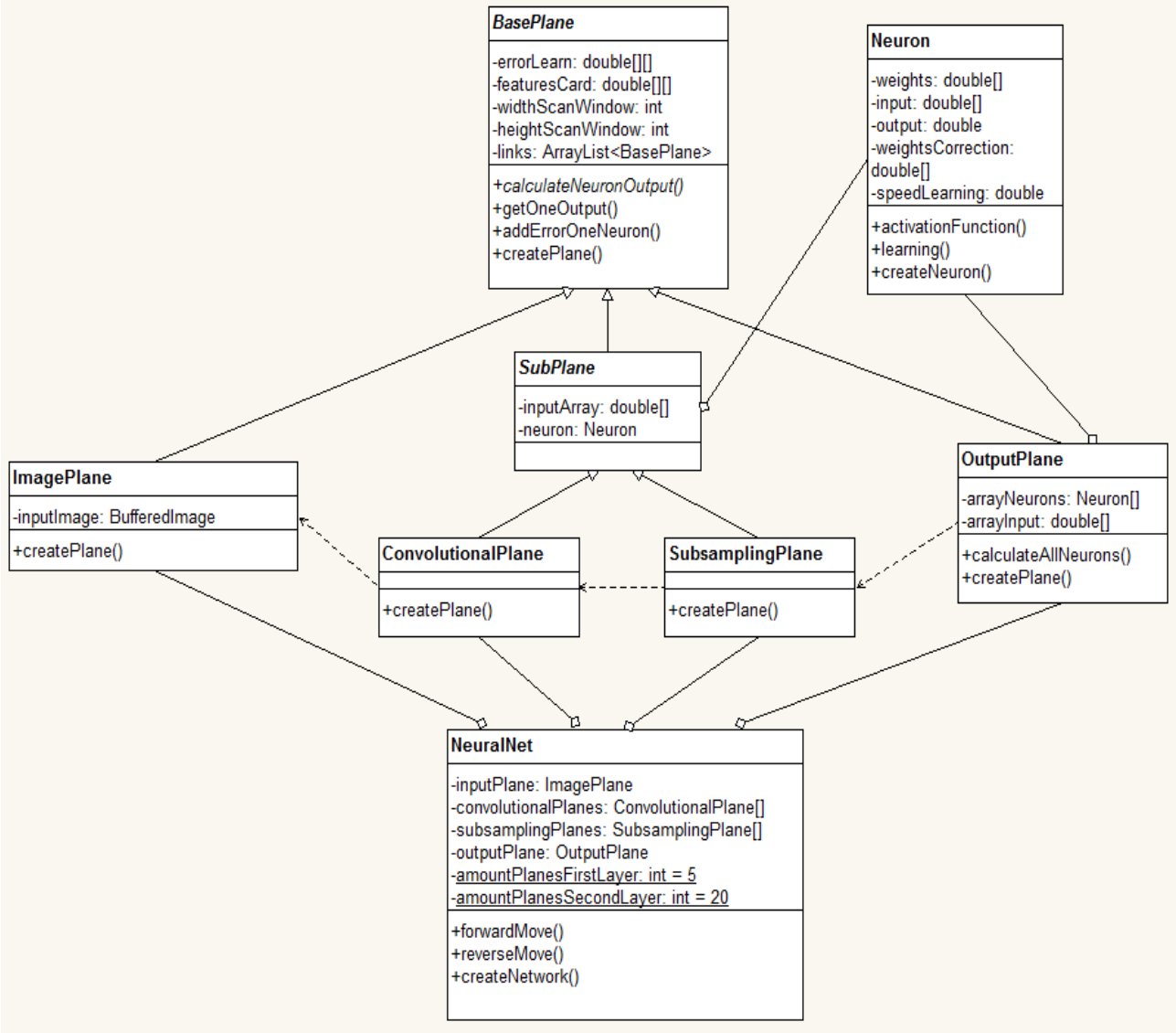
8 ГРАФІК ЗАЛЕЖНОСТІ ПОМИЛКИ НЕЙРОННОЇ МЕРЕЖІ ВІД КІЛЬКОСТІ ІТЕРАЦІЙ ПРИ ЇЇ ТЕСТУВАННІ ЗА ДОПОМОГОЮ НОРМАЛІЗОВАНОЇ ВИБІРКИ



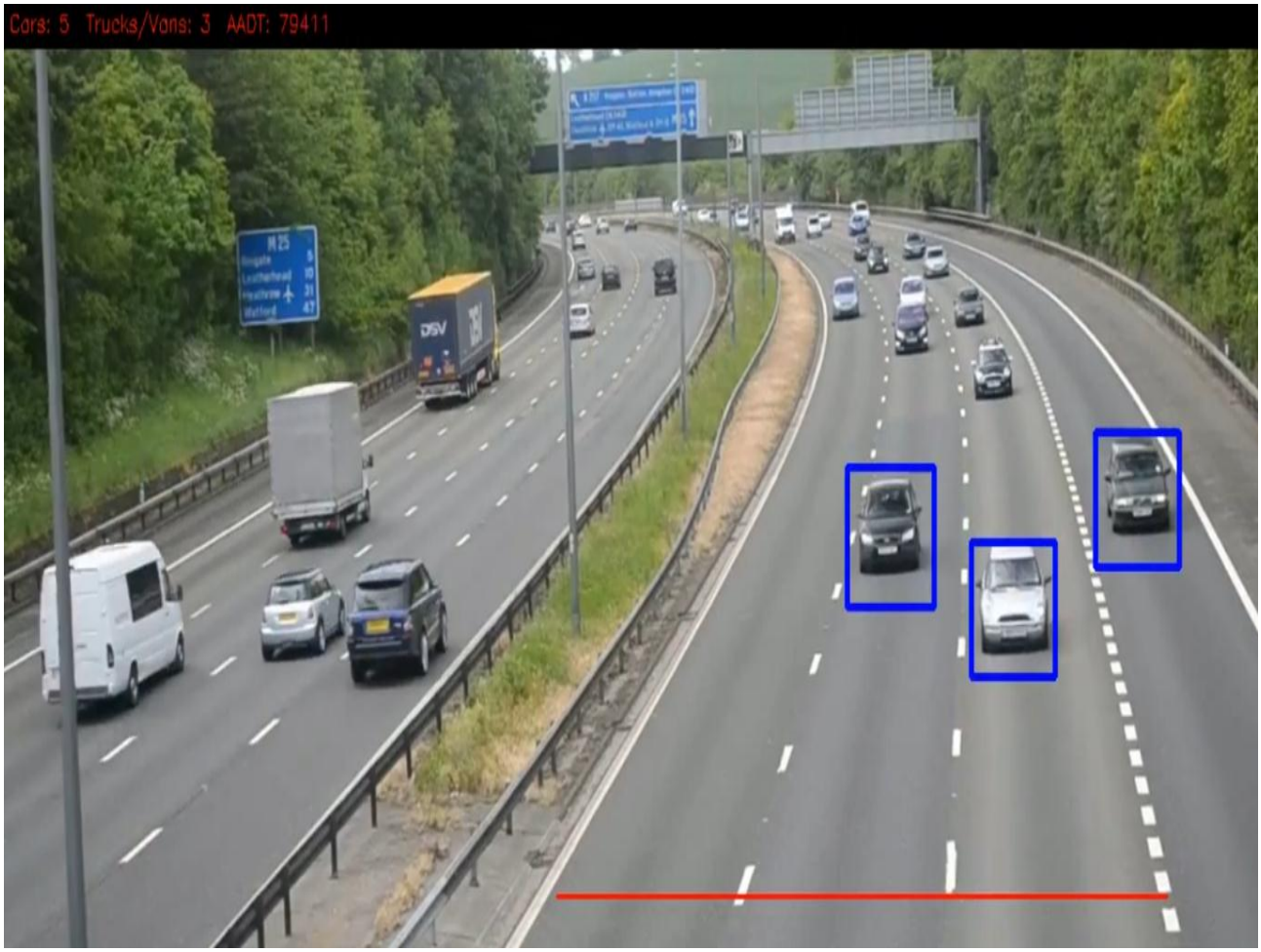
9 АРХІТЕКТУРА ЗГОРТКОВОЇ НЕЙРОННОЇ МЕРЕЖІ



10 ЛОГІЧНА МОДЕЛЬ ЗГОРТКОВОЇ НЕЙРОМЕРЕЖІ У ВИГЛЯДІ ДІАГРАМИ UML



11 ВИХІДНЕ ГРАФІЧНЕ ЗОБРАЖЕННЯ ДЛЯ ТЕСТУВАННЯ



12 РОЗПОДІЛ ОБ´ЄКТІВ МАШИН НА ТИПИ: СИНЯ РАМКА «CAR» І ЗЕЛЕНА РАМКА «TRUCK»