МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Навчально-науковий інститут інноваційних освітніх технологій

Кафедра комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Савченко А.С.

"_____"_____20__ р.

ДИПЛОМНА РОБОТА

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ "МАГІСТР"

Тема: «Методика розпізнавання зображень за допомогою нейронних мереж»

Студент: Омельченко Павло Олексійович

Керівник: к.т.н., доцент, Моденов Юрій Борисович

Нормоконтролер: к.т.н., доцент, Райчев І.Е.

Київ 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Навчально-науковий інститут інноваційних освітніх технологій

Кафедра комп'ютерних інформаційних технологій
Освітній ступінь: Магістр
Спеціальність: 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Савченко А.С.
"_____"_____ 20__р.

З А В Д А Н Н Я

на виконання дипломної роботи
студента Омельченка Павла Олексійовича

1. Тема: *Методика розпізнавання зображень за допомогою нейронних мереж*

затверджена наказом ректора від 22.11.2019 р. *№ 2701/ст.*

2. Термін виконання: з 25.11.2019 р. до 20.02.2020 р.

3. Вихідні дані: проаналізувати сучасні методики розпізнавання зображень; проаналізувати переваги та недоліки існуючих методик; розробити ефективну модель розпізнавання зображень; на основі моделі розробити програмне забезпечення, що дозволяє розпізнавати завантажені зображення.

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці):

1. Проаналізувати існуючі методи та засоби класифікації зображень.
2. Розробка алгоритмів класифікації зображень за допомогою нейронних мереж.
3. Огляд результатів тестування розроблених моделей.
4. Розробка програмного забезпечення класифікації зображень

КАЛЕНДАРНИЙ ПЛАН

виконання магістерської роботи

№ п/п	Етапи виконання бакалаврської атестаційної роботи	Термін виконання етапів	Примітка
1.	*Уточнення постановки задачі*	*25.11.2019-26.11.2019*	
2.	*Аналіз літературних джерел*	*26.11.2019-01.12.2020*	
3.	*Обґрунтування вибору рішення*	*01.01.2020-15.01.2020*	
4.	*Збір інформації*	*01.01.2020-15.01.2020*	
5.	*Аналіз сучасних методик та систем розпізнавання інформації*	*15.01.2020-18.01.2020*	
6.	*Розробка моделей*	*18.01.2020-20.01.2020*	
7.	*Тестування моделей*	*20.01.2020-22.01.2020*	
8.	*Розробка програмного модуля (Backend)*	*22.01.2020-25.01.2020*	
9.	*Розробка програмного модуля (Frontend)*	*25.01.2020-15.01.2020*	
10.	*Оформлення і друк пояснювальної записки*	*15.01.2020-10.02.2020*	
11.	*Оформлення презентації*	*10.02.2020-13.02.2020*	
12.	*Отримання рецензій від опонентів*	*13.02.2020-15.02.2020*	
13.	*Захист в ДЕК*	*15.02.2020-20.02.2020*	

Студент П.О. Омельченко

(підпис, дата)

Науковий керівник Ю.Б. Моденов

(підпис, дата)

РЕФЕРАТ

магістерської роботи Омельченка Павла Олексійовича на тему

«Методика розпізнавання зображень за допомогою нейронних мереж»

У магістерській роботі досліджується досліджується розпізнавання зображень за допомогою нейронних мереж, на основі даних аналізу мережевого Інтернет трафіку. Дана тема є актуальною, оскільки у рамках дослідження даної області є багато зображень, які легко описати вручну людині, проте це займає чимало часу і ресурсів, тому виникає необхідність описати і провести класифікацію оточуючих зображень автоматично за допомогою ПЕОМ.

Метою роботи є проведення аналізу ефективних засобів класифікацій зображень у прикладних системах, таких як аналізу мережевого Інтернет трафіку. Об'єктом дослідження є аналіз ефективних засобів використання нейронних мереж у програмних засобах та реалізація такого засобу у рамках мети роботи.

Було виконано огляд засобів класифікації – автоматичних систем категоризації зображень та зроблено висновки щодо їх недоліків та переваг при виконанні роботи. На основі проаналізованих засобів в підходів роботи нейронних мереж розроблено власний програмний модуль, який можна як використовувати у рамках проведення аналізу інтернет трафіку, так і модифікувати для використання у різноманітних моделях.

Науковою новизною даної роботи є розробка власної моделі, її оцінка, підбір параметрів та написання програмного модуля.

Загальний обсяг роботи: 74 сторінки, 25 рисунків, 20 таблиць та 24 бібліографічних найменувань.

Ключові слова: машинне навчання, нейронні мережі, класифікація зображень.

ABSTRACT

for master's thesis of Omelchenko Pavlo Olexiyovich

“Image classification using neural network”

The master's thesis investigates image recognition using neural networks, based on the analysis of network Internet traffic. This topic is relevant because there are many images that are easy to describe manually by a person in the study area, but it takes a lot of time and resources, so there is a need to describe and classify the surrounding images automatically using a PC.

The purpose of this work is to analyze the effective means of image classification in application systems, such as the analysis of network Internet traffic. The object of the study is to analyze the effective means of using neural networks in software and to implement such a tool within the purpose of the work.

The classification tools - automatic image categorization systems - were reviewed and conclusions were drawn about their disadvantages and advantages when performing the work. Based on the tools analyzed, neural network approaches have developed their own software module, which can be used as part of Internet traffic analysis and modified for use in various models.

The scientific novelty of this work is the development of its own model, its estimation, selection of parameters and writing of the program module.

A total volume of work: 74 pages, 25 figures, 20 tables and 24 bibliographic titles.

Keywords: machine learning, neural networks, classification of images.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

ГС	Градієнтний спуск
ДР	Дерева рішень
НБК	Наївний баєсовий класифікатор
НМ	Нейронна мережа
ПЕОМ	Персональна електронна обчислювальна машина
ПЗ	Програмне забезвечення
СГС	Стохастичний градієнтний спуск
ВЛ	Випадкові ліси

ВСТУП

Розвиток інформаційних галузей і технології у рамках людської діяльності в суспільстві на даний час супроводжується зростанням ролі комп'ютерних інформаційних технологій. Зараз обсяг інформаційних потоків досить сильно зріс і продовжує свій стрімкий ріст, і відповідно з'явилася значна необхідність проводити пошук способів зберігання, подання, формалізації і систематизації.

Розвиваються засоби автоматичної обробки інформації, проведення аналізу і класифікації її виявлень у різних сферах діяльності.

Ми живемо в часі, де все швидко змінюється та вдосконалюється. Це стосується й інформації. Її кількість з кожним роком все більше і більше зростає. Сюди входять різні дані – зображення, текст, аудіо, відео, які зберігаються та розширюються.

Різні сфери викликають різний попит, але загалом проблем структуризації, підбору рішень, та класифікації інформації викликають значний попит.

Вперше нейронні мережі привернули увагу 2012 року, коли Олексій Грижевський виграв конкурс ImagensNet, вигравши рекорд від 27% до 15% (майже вдвічі більше), і цього разу навколо. В даний час поглиблене навчання лежить в основі послуг багатьох компаній: Facials використовує нейронні мережі для алгоритмів автоматичного тегування, General - для пошуку тексту, пошуку тексту - для пошуку інфраструктури.

Завданням класифікації зображень є аналіз першоджерела зображення і виведення можливої або конкретної категорії. Наприклад, для зображення лева менш натренована система може вивести лише, що дане зображення відноситься до класу «Тварини». Більш натренована система може видати більш конкретну

назву, наприклад латинську класифікацію як «Panthera leo». Ця задача є досить тривіальною для людини, проте це досить складна проблема для автоматизації.

У дипломній роботі буде вирішуватись задача категоризації зображень у рамках аналізу інтернет трафіку.

Поставлена мета вимагає вирішення наступних задач:

- Оброббка
- підбір архітектур нейронної мережі
- оцi нка метрики алгоритму
- оцi нка точностi
- дослідження впливу параметрів

Обєктом досліджень у даній роботі є застосування нейронних мереж та її оптимальний варіант у рамках аналізу трафіку.

Метою дипломної роботи є розробка моделі та її практична реалізація у рамках програмного модуля, який можна використовувати для роботи під час проведення аналізу трафіку, та який за необхідності може стати більш складною частиною інших програмних модулів.

Досягнення поставленої мети реалізовано з використанням мови програмування JavaScript, фреймворк NodeJS, React для роботи безпосередньо з архітектурами нейронних мереж та імплементацією різних методів та бібліотек Tensorflow, HTML5, CSS3, фреймворк Express та мова JavaScript для створення сайту для відображення результатів.

Наукова новизна дипломної роботи полягає в тому, що було створено програмне забезпечення (Back-end, Front-end), яка за допомогою розробленої моделі проводить класифікацію з високою точністю зображень і за необходності

може бути дотренована. У розмірах є дуже малою, може бути використана як у браузері так і на телефоні. На даний момент аналогів даної системи не виявлено.

Потенційні застосування та практична цінність результатів дипломної роботи:

Модель може використовуватись на будь-якому компьютері, так і на смартфоні

1. АНАЛІЗ АКТУАЛЬНИХ ЗАБОСІВ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

1.1 Метод класифікації зображень з використанням комп’ютерного зору

Ознаки Хаара - є цифрове зображення, яке широко використовується в розпізнаванні образів. Образи мають інтуїтивну подібність з вейвлетами Хаара. Ознаки Хаара використовувалися в першому детекторі осіб, що працює в реальному часі.

Історично склалося, що алгоритми, які працюють лише з інтенсивністю зображення, мають велику обчислювальну складність, називаються на зображенні, тоді інтенсивність пікселів у регіонах підсумовується, після чого розраховується різниця між сумами. Ця різниця буде значенням певного атрибута, визначеним його розміром, певним актом, розміщеним на зображенні.

Наприклад, є база данних, що містить обличчя людей. Певні області в обличчях людей відрізняються за кольором і відтінкам. Тож певною ознакою Хаара для людей є певна кількість частин обличчя.

На етапі виявлення в методі Віоли - Джонса вікно встановленого розміру рухається по зображенню, і для кожної області зображення, над якою проходить вікно, розраховується ознака Хаара. Наявність або відсутність предмета в вікні визначається різницею між значенням ознаки і учнем порогом. Оскільки ознаки

Хаар не пі дходить для класифі кації (трохи вище, ні ж у випадку аномально розподі лених розмі рі в), для опису об'єкта з достатньою точні стю потрі бно бі льше. Це в методі Bioli - знаки Хара Янса, органі зовані в каскадному класифі каторі .

Через особливі сть Хаара швидкі сть є найвищою, порі вняно з і ншими знаками. При використанні і нтегрального зображення зображення знаки Хаара можуть бути обчислені прямолі ні йно (приблизно 60 і нструкці й попередньої обробки на знак двох областей).

1.2 Метод класифікації зображень за допомогою машинного навчання

Машинне навчання на даний час має використання в вирішенні проблем, повязаних з зором, медициною, керуванням транспортом та схожий областях. Основою для них служать методи та засоби AI, такі як класифікація, локалізація і виявлення. В останній час нейронні мережі дуже ефективно себе показали у задачах, де дані представлені у вигляді тексту, картинок, відео, тому зараз їх використовують досить часто для таких задач. Для задач класифікації можна використовувати класичні методи машинного навчання, такі як: наївний Баєсовий класифікатор, метод опорних векторів, логістичну регресію, випадкові ліси чи метод стимулювання градієнта.

MNIST - це велика база зразкі в зразкових цифр рукописного тексту. База даних - це стандарт, затверджений Наці ональним і нститутом стандарті в і технологі й для калі брування та ві дображення методі в розпі знавання зображень для забезпечення машинного навчання. Дані складаються і з заздалегі дь визначених прикладі в ві зуалі заці ї, на яких можна тренувати

та тестувати системи. База даних MNIST мі стить 60 000 зображень для тренувань та 10 000 зображень для тестування. На цьому наборі даних метод наївного Байєса працює досить точно, а саме бі ля джерела [5], для того щоб отримати 85% точності. Так як наївний баєсовий класифікатор — модель, що базується на теоремі Баєса для визначення ймовірності приналежності екземпляра до одного з класів C за умови того, що залежні змінні приймають задані значення. Це означає що, якщо на основі відомих змінних можна точно визначити, до якого класу належить екземпляр, баєсовий класифікатор підсумує, що ймовірність приналежності до даного класу рівна 1. У інших випадках, коли екземпляр може з різною ймовірністю належати до різних класів, результатом роботи класифікатора буде набір ймовірностей приналежності до певного класу [12]. Виходячи з визначення і самого набору даних, можна пояснити чому ця модель дала прийнятну точність, а саме, що зазвичай люди пишуть схоже і так як зображення на вхід ми подаємо у вигляді пікселів, то основна частина буде розміщатись у одного типу цифр в тих же місцях.

Але, на реальних даних, зокрема і в цій роботі, краща точність була отримана за допомогою згорткових нейронних мереж. Детальніше про ці алгоритми у наступному розділі.

Отож, класичні методи машинного навчання, такі як наївна байєсівська класика, вектори методів, логістична регресія, випадкові ліси або методи стимуляції градієнта та методи глибокого навчання дають нам результати класифікації. Звичайно, метод слід вибирати виходячи зі специфіки даних та інших характеристик, таких як - незадовільна точність алгоритму, час прогнозування тощо. Зазвичай реальні дані сприяють конволюційним нейронним мережам, оскільки класичні методи не в змозі виявити всіх необхідні зразки.

1.3 Комбінація методів комп’ютерного зору та нейронних мереж

Класифікація цифрових зображень, що зберігаються в базах даних, з використанням традиційних алгоритмів машинного навчання, характеризується високою міцністю, великою кількістю деталей, великим класом якості зображення та великою кількістю особливостей зображення. Крім того, їх класифікація за допомогою цих алгоритмів займає багато часу. Існуючі системи зберігання зображень, такі як QBIC [5] та VisualSEEK [4], обмежують методи класифікації списків на зображення за основним форматом, текстурою та кольором [6].

Один із методів, який ми використовуємо для розповсюдження, класифікації та накопичення, - це метод на основі нейронів. Для зменшення чистих нейронних меж введення необхідна система класифікації зрошення, щоб перетворити її на стадію. Одним з ребер перенапруги цифрового зображення є перетворення вейвлета. Час В даний час вейвлет перекидання - це метод типу ширпо, який затримує зоряність та часові характеристики, такі як кадр та текст.

Алгоритм, заснований на комбінації вейвлет-перетворення Хаара і нейронної мережі для класифікації цифрових зображень з бази даних. Кольорове зображення ділиться на три RGB-компоненти. Моменти кольору першого порядку і коефіцієнти розкладання вейвлет перетворення Хаара [7] трьох RGB-компонентів зображень є вхідним вектором багатошарової нейронної мережі, навченої алгоритмом зворотного поширення помилки.

Подання змісту цифрових зображень. Зміст і контури цифрових зображень зазвичай використовуються в класифікації зображень. У алгоритмі моменти

кольору і вейвлет-коефіцієнти розкладання використовуються для подання змісту цифрових зображень.

Моменти вибору. Ключові моменти використовуються у багатьох системах відновлення зображень [8], особливо якщо зображення містить лише один об'єкт. Моменти першого, другого та третього порядку ефективні для забезпечення розподілу зображення.

Перевірка вейвлетів, яку втягувач входить у систему відновлення ковзання. На кожному рівні вейвлет-перетворення сигналу накладається на чотири піддіапазони часто (боковини) LLn, LHn, HLn, HHn (рис. 1.1), де L низька частота; H - фільтровий привал; n - рівень розкладання. На рис. 1.1 представлені стандартні показники для відбору рівня перехідного збору: LL, LH, HL, HH. LAD - це обробка низької роздільної здатності (cAn), HLn - деталь збирання вертикальної деталі (cVn), LNn - це деталі зацікавлення зображення із зображенням (cHn), HHn - діагностична інформація.

Отримані коефіцієнти вейвлетів розміщуються на вхідній нейронній середині.

У алгоритмі використовується вейвлет-перетворення Хаара [8] по шести рівням розкладання.
Отримані вейвлет-коефіцієнти подаються на входи нейронної мережі.

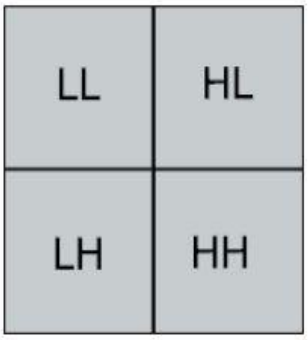


Рисунок 1.1 - Одноразове застосування двовимірного вейвлет-перетворення до квадратному зображенню [5]

У роботі [10] зазначалось, що максимальна отримана точність класифікації була 88%.

1.4 Аналіз актуальних методів класифікації

За останнє десятиліття було опубліковано значну кількість статей і наукових робіт про алгоритми класифікації зображень у різних областях застосування. Крім того, було запропоновано наступні підходи: класифікація зображень, де на виході ми використовуємо лише мітку класу або ж використання ймовірності того, що на даному зображені клас з певною міткою. Останній підхід буде частіше використовуватись у системах, де помилки є не припустимими, наприклад, при класифікації і нтернети контенту.

У статті[10] було описано, що класичні підходи до вирішення таких задач на наборі даних, який вони використовували дали 85% точності. У роботі[11], де потрібно було класифікувати 53 класи на їхньому наборі даних використовували специфічну обробку даних за рахунок чого їм і вдалось отримати 87% точності. Також була розглянуто рішення[12], де було поєднано два вище зазначених підходи, що дозволило покращити результати. Звісно були і статті, результати

яких відтворити не вдалось, так як деякі з них використовували закритий набір даних чи дуже багато обчислювальних потужностей.

1.5 Висновок

У першому розділі було проведено аналіз різних методів і підходів виконання задач класифікації зображень. Було описано як класичні методи комп’ютерного зору так і методи машинного навчиння. Як ми бачимо, на даний момент найкращі результати класифікації зображень було отримано за допомогою використання нейронних мереж, а саме згорткових нейронних мереж. Що і буде далі досліджуватись в даній роботі.

2. АНАЛІЗ АЛГОРИТМІВ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ ТА РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ

2.1 Аналіз поняття багатошарого перцептрону

“Перцептрон - це одна з перших моделей нейронних мереж. Незважаючи на свою простоту, ця модель здатна навчатися і вирішувати досить складні завдання. Основна математична задача, з якою він справляється, - це лінійне розділення будь-яких нелінійних множин, так зване забезпечення лінійної сепарабельності [12]”. Типова архітектура багатошарового перцептрону схематично зображена на рис. 2.1.

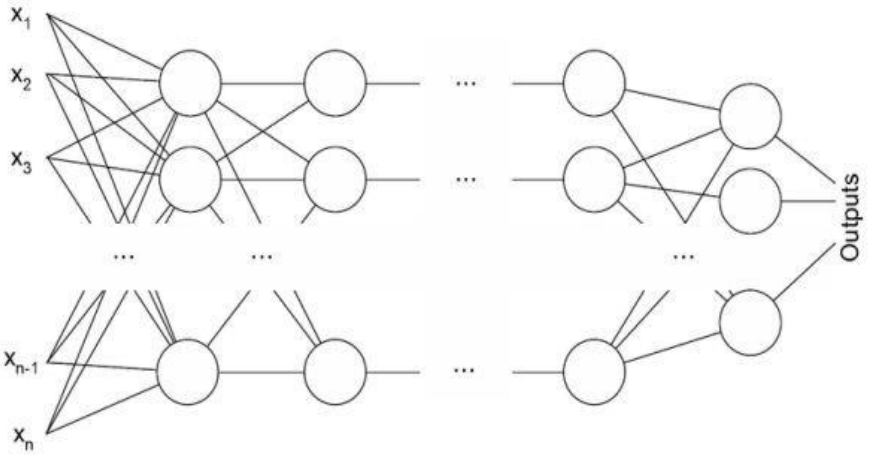


Рисунок 2.1 – Модель базового перцептрона [5]

Отже, отримується певна послідовність асоціацій між вхідними даними та інформацією виході. В біологічному плані це відповідає перетворенню, наприклад, зорової інформації у фізіологічну відповідь рухових нейронів. Перцептрон застосовується для вирішення класичних задач машинного (класифікація, регресія) навчання як окрема модель, так і в складі більш складних моделей.[2]

2.2 Аналіз поняття згорткові нейронні мережі

“Нейронні мережі, що складаються, - це одне з найвпливовіших нововведень у галузі комп’ютерного зору. Вперше нейронні мережі привернули загальну увагу в 2012 році, коли Олексій Грижевський виграв змагання ImaginationNet (грубо кажучи, це щорічна Олімпіада)[16]”. Сьогодні глибинне навчання лежить в основі послуг багатьох компаній: Facebook використовує нейронні мережі для алгоритмів автоматичного створення тегів, Google - для пошуку серед фотографій користувача, Amazon - для генерації рекомендацій товарів, Pinterest - для персоналізації домашньої сторінки користувача, а Instagram - для пошукової інфраструктури.

Завданням класифікації зображень – є аналіз початкового зображення і призначення конкретного класу або припустимого, яка найкраще характеризує зображення. Для людей це один з перших навичок, який вони починають освоювати з народження. На рис. 2.2 показано як сприймає картинки комп’ютер.

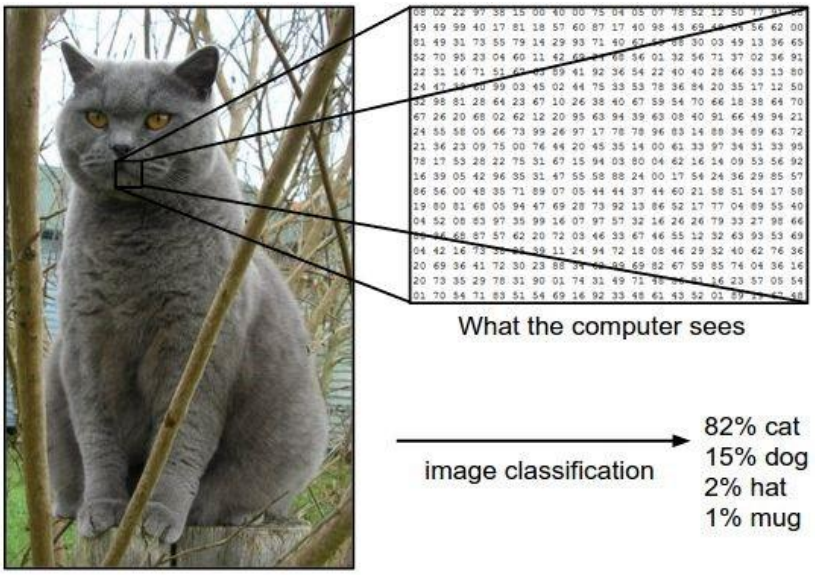


Рисунок 2.2 – Зображення у вигляді байтів [5]

Коли комп'ютер бачить зображення (приймає дані на вхід), він бачить масив пікселів. Залежно від дозволу і розміру зображення, наприклад, розмір масиву може бути 3x3x4 (де 3 - це значення каналів RGB). Щоб було зрозуміліше, давайте уявимо, у нас є кольорове зображення у форматі JPG, і його розмір 580x580. Відповідний масив буде. Кожному з цих в**і** дпов**і** дн**і** значення у д**і** апазон**і** від 0 до 255, яке описує інтенсивність пікселя в цій точці. Ці цифри, залишаючись безглуздими для нас, коли ми визначаємо що на зображенні, є єдиними вступними даними, доступними комп'ютера.

Ідея нейронних мереж конструкцій запозичені з біології: нейронні мережі імітують процес обробки нейронами головного мозку сприймаються з навколишнього середовища образів і участь цих нейронів в прийнятті рішень [10]. Принцип роботи окремо взятого штучного нейрона по суті дуже простий: він обчислює зважену суму всіх елементів вхідного вектора $\underline{x}$, використовуючи вектор ваг $\underline{w}$ (а також адитивну складову зсуву w_0), а потім до результату може застосовуватися функція активації σ.

“Класичними реалізаціями ЗНМ, що стали проривом в індустрії, є LeNet5 та AlexNet [6]”(рис. 2.3).

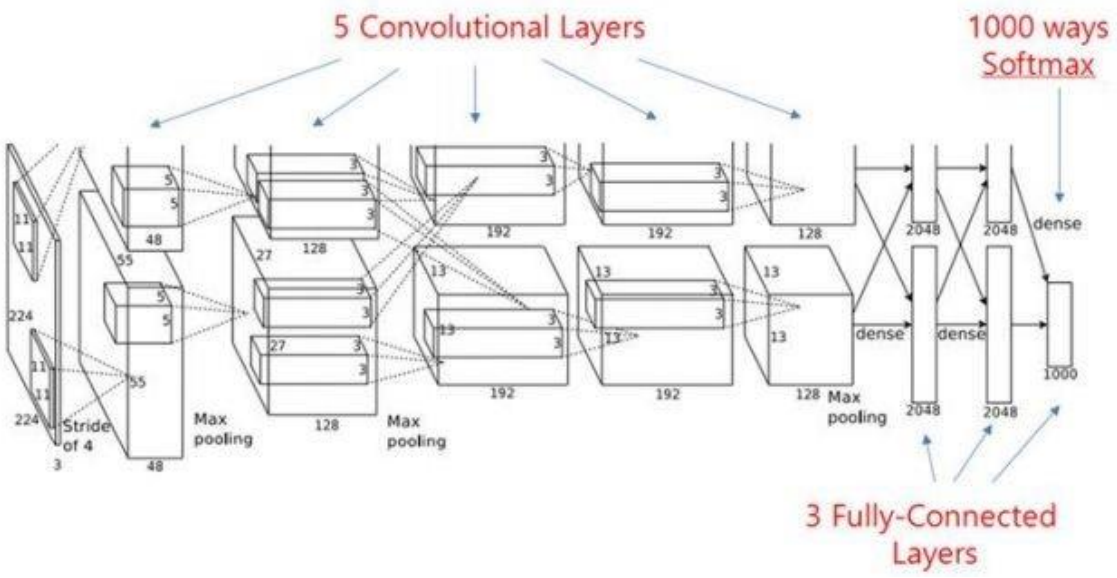


Рисунок 2.3 – Архітектура AlexNet [13]

Кажуть, що програма вивчає досвід E для кожного класу задач T у розумінні міри якості L, оскільки при вирішенні задачі T якість, виміряна мірою L, збільшується з демонстрацією нового досвіду E [5].

Рисунок 2.4 - Загальна класифікація задач машинного навчання [8]

Два основні класи задач машинного навчання - це задачі навчання з учителем (supervised learning) і навчання без вчителя (unsupervised learning).

Під час навчання з викладачем набір навчальних прикладів, який зазвичай називають навчальним набором даних про навчання, подається як вхідні дані, а завдання - вийти за рамки того, що зазвичай виражається у формі даних. Основне припущення тут полягає в тому, що доступні для тренінгу дані будуть дещо збільшені з даними, за якими потім застосовується тренувальна модель, інакше ніяке узагальнення не буде неможливим.

Завдання підготовки вчителів зазвичай поділяються на класифікаційні та регресійні завдання. У задачі класифікації об'єкт повинен бути призначений на вхід для визначення в одному з (зазвичай кінцевих) класів. І в задачі регресії нам потрібно передбачити значення якоїсь функції, яка зазвичай може мати нескінченно багато різних значень. Наприклад, щоб виростити людину, передбачити її вагу, зробити перерву на завтра тощо.

Наприклад, системи пошуку та ранжирування часто стикаються із завданням навчання за ранжуванням. Він ставиться так: згідно з наявними даними (у пошуковій системі це будуть тексти документів та минула поведінка користувачів) у відповідь на цей запит).

Якщо між набором даних і конкретним завданням є невідповідність, не існує простих даних, яким потрібно «знайти якийсь сенс», щоб виникали завдання навчання без вчителя. Типовим прикладом завдання навчання не вчителя є кластеризація: потрібно певною мірою розбити дані на невидимі класи, щоб точки, присвоєні одному і тому ж кластеру, були максимально близькими один до одного, наскільки кластери були б якомога далі від одного з них, як можна менше.Нейронні мережі, можуть використовуватись для всіх цих задач, але потрібно підібрати правильно функцію втрат.

Використання нейронних мереж має такі переваги:

1. Можуть знайти складніші закономірності в даних, в порівнянні із класичними методами машинного навчання, добре себе показали для задач з картинками і текстами;

2. Можна донавчати, коли з’являються нові дані;

3. Зазвичай добре модель, для якої вдалоо підібрані гіпер параметри та достатньо даних буде краще працювати, за класичні алгортми машинного навчання;

4. Нейронна мережа сама шукає закономірності в даних і певні признаки, по яким і навчається, робить передбачення, в той час як для класичних методів потрібно самому придумати всі ці ознаки, щоб модель краще навчилась.

Хоча використання нейронних мереж має і ряд недоліків:

1. Для того, щоб отримати хороші результати потрібно мати набагато більше даних, аніж для класичних методів;

2. Для тренування потрібно мати більше потужностей, наприклад одну або декілька відеокарт (GPU), в той час як для класичних методів достатньо виконувати обчислення на процесорі;

3. Час тренування виходячи з попередніх двох пунктів також забирає більше часу, в порівнянні з класичними методами (хоча на це можна впливати використовуючи додаткові потужності, хмарні рішення);

4. Легше перенавчаються, важче впроваджувати в реальну систему, на меленьких пристроях, застосовувати в реальному часі.

2.3 Аналіз архітектури для класифікації зображень

У поточному часі є чимала кількість багато архітектур згорткових нейронних мереж. Щорічно зявляються свіжі наукові статті, модернізуються підходи, виникають новітні архітектури. В даній праці було використано подальші архітектури: MobileNet, DenseNet, InceptionV3, Squeezenet. Дві перших архітектури досить об’ємні і навчання займало досить багато часу, але і точність була краща. Останні є меншими за розмірами , проте мали перевагу у швидкості навчання, оскільки вони розраховані для використання в пристроях, де наявно менше розрахункових можливостей.

На рис 2.5 зображена архітектура DenseNet, як бачимо вона складається з трьох блоків, приклад одного такого блоку зображено на рис 2.6.

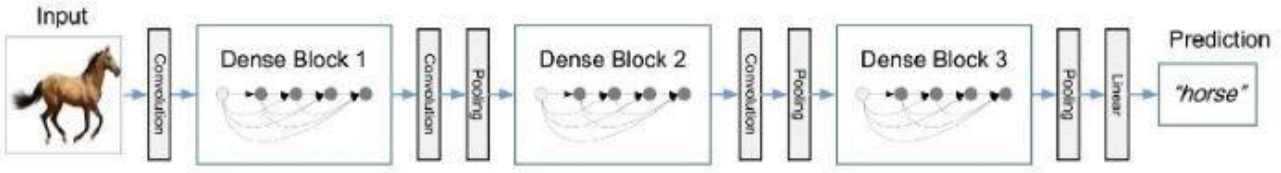


Рисунок 2.5 - Архітектура DenseNet [7]

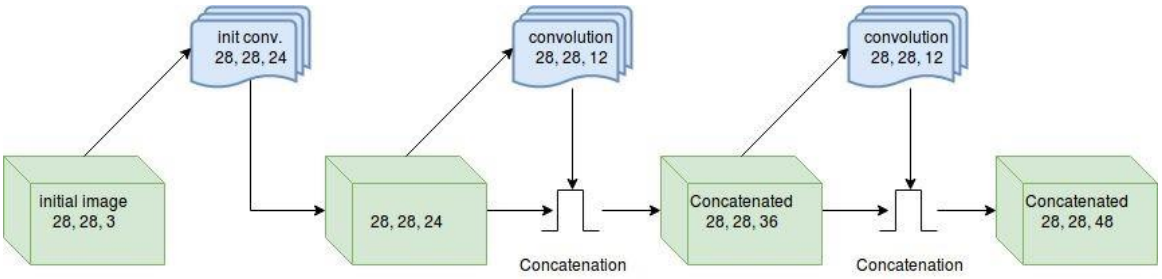


Рисунок 2.6 - Блок з архітектури DenseNet [7]

На даний момент ця архітектура показує одну з найменших помилок у відкритих наборах даних CIFAR / SVHN

Таблиця 2.1 - Результати оцінки якості різних алгоритмів на CIFAR/SVHN наборах даних [10]

Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [31]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [33]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	21	38.6M	7.33	4.60	28.20	23.73	1.87
ResNet [11]	110	1.7M	-	6.61	-	-	-
ResNet (reported by [13])	110	1.7M	13.63	6.41	44.74	27.22	2.01
ResNet with Stochastic Depth [13]	110	1.7M	11.66	5.23	37.80	24.58	1.75
	1202	10.2M	-	4.91	-	-	-
Wide ResNet [41]	16	11.0M	-	4.81	-	22.07	-
	28	36.5M	-	4.17	-	20.50	-
with Dropout	16	2.7M	-	-	-	-	1.64
ResNet (pre-activation) [12]	164	1.7M	11.26*	5.46	35.58*	24.33	-
	1001	10.2M	10.56*	4.62	33.47*	22.71	-
DenseNet ($k = 12$)	40	1.0M	**7.00**	5.24	**27.55**	24.42	1.79
DenseNet ($k = 12$)	100	7.0M	**5.77**	**4.10**	**23.79**	**20.20**	1.67
DenseNet ($k = 24$)	100	27.2M	**5.83**	**3.74**	**23.42**	**19.25**	**1.59**
DenseNet-BC ($k = 12$)	100	0.8M	**5.92**	4.51	**24.15**	22.27	1.76
DenseNet-BC ($k = 24$)	250	15.3M	**5.19**	**3.62**	**19.64**	**17.60**	1.74
DenseNet-BC ($k = 40$)	190	25.6M	-	**3.46**	-	**17.18**	-

Архітектура Incéption V3, відображена Googlé, була відображена перед DannesNet, а також дає дуже хороші результати на тих же відкритих наборах даних CIFAR. Як ви бачите на малюнку, модель складається з t модулів, вихід яких надається на вхід наступного.

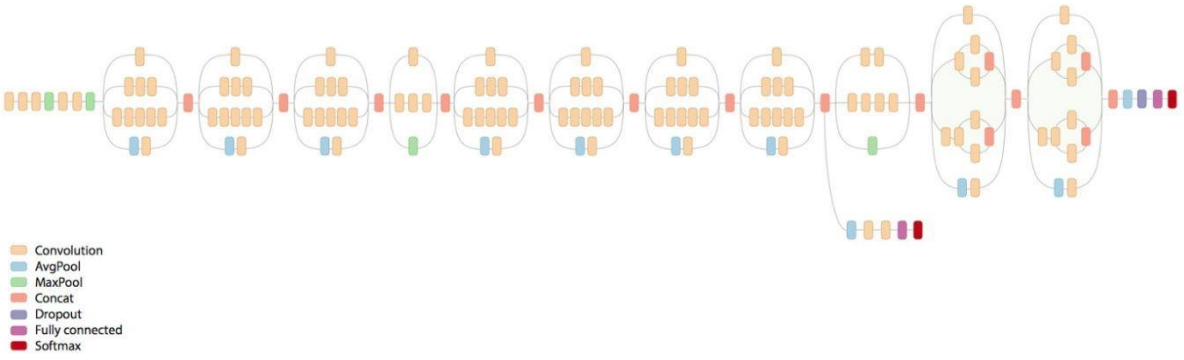


Рисунок 2.7 - Архітектура InceptionV3 [11]

На рис. 2.8 зображено архітектура SquzeHet. Робота SqueezeNet забезпечує розумовий підхід. Для такої ж порівняльної точності розрахунків архітектура SqueezeNet виграє в швидкодії більш ніж в три рази.

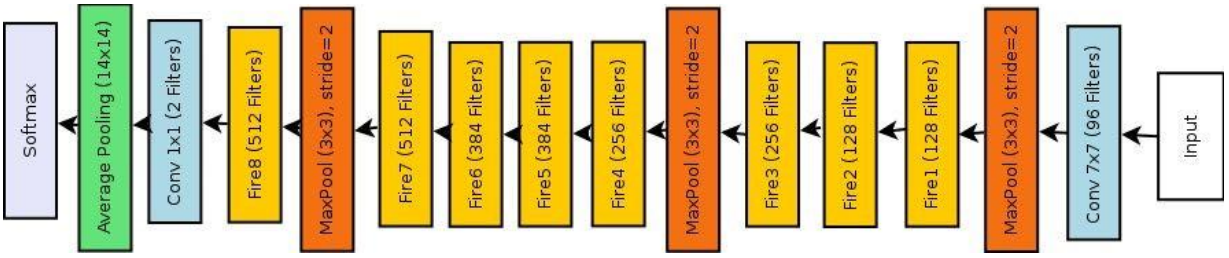


Рисунок 2.8 - Архітектура Squezenet[18]

У таблиці 2.2 зображено переваги SqueezeNet над моделю A1exNet(на наборі даних ImageNet), оскільки ми можемо досягти однакової точності, роблячи це, модель в 50 разів менша.

Таблиця 2.2 - Результати оцінки якості SqeezeNet та A1exNet [18]

CNN architecture	Compression Approach	Data Type	Original → Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD (Denton et al., 2014)	32 bit	240MB → 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning (Han et al., 2015b)	32 bit	240MB → 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression (Han et al., 2015a)	5-8 bit	240MB → 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	**50x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB → 0.66MB	**363x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB → 0.47MB	**510x**	57.5%	80.3%

На рис. 2.9 зображено архітектуру MobileNet, яка показує схожі результати до попередньої. Зазвичай ці архітектури використовують для мобільних пристроях чи інших пристроях з обмеженою пам’яттю.

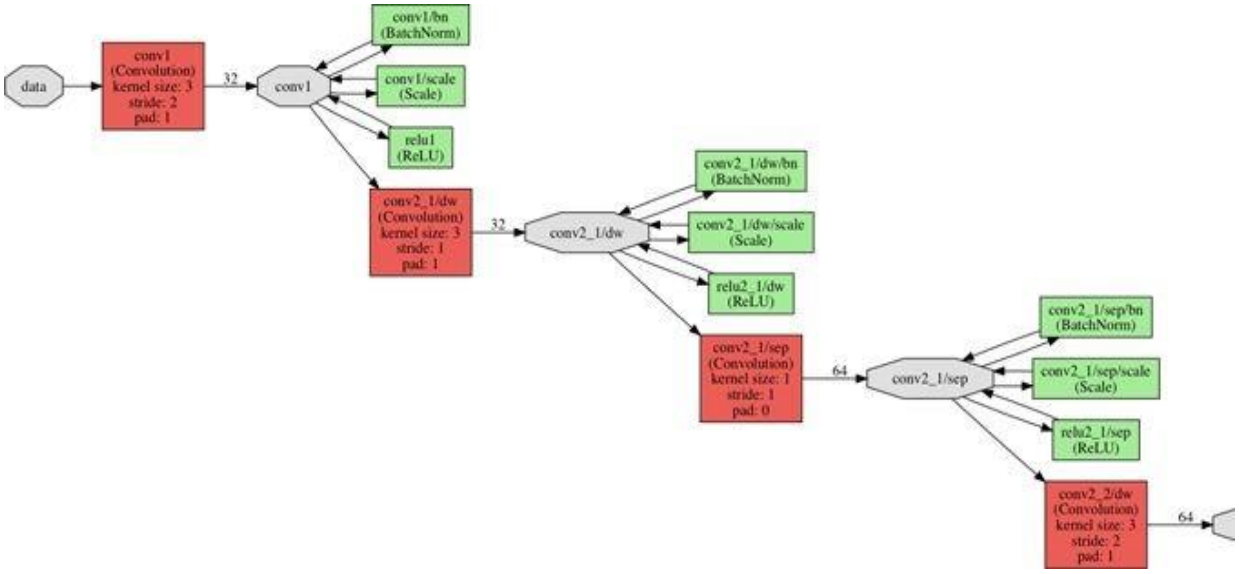


Рисунок 2.9 - Архітектура MobileNet [20]

Отже, проаналізувавши попередні архітектури можна зробити висновок, що більші архітектури такі як DenseNet, InceptionV3 навчаються набагато довше в порівнянні з двома іншими, але дають трохи більшу точність. На деяких наборах даних можна отримати точність, яка буде відрізнятись до 5% (якщо порівнювати всі попередні архітектури), але для задач які вирішуються в реальному житті ця точність не критична, а останні дві моделі мають більший спектр застосування.

2.4 Функції активації у штучній нейронній мережі

“Функція активації (активаційна функція, функція збудження) - функція, що обчислює вихідний сигнал штучного нейрона. Як аргумент приймає сигнал Y, одержуваний на виході вхідного суматора. Найбільш часто використовуються наступні функції активації [5]”.

2.4.1. Жорстка порогова функція активації

Ця функці я є простою і зрозумі лою функці єю. Якщо вхі дне значення менше вхі дного значення, значення функці ї активаці ї дорі внює

мі німальному виходу, в ішому випадку - максимальному виходу. 2.10 - графік цієї функції.

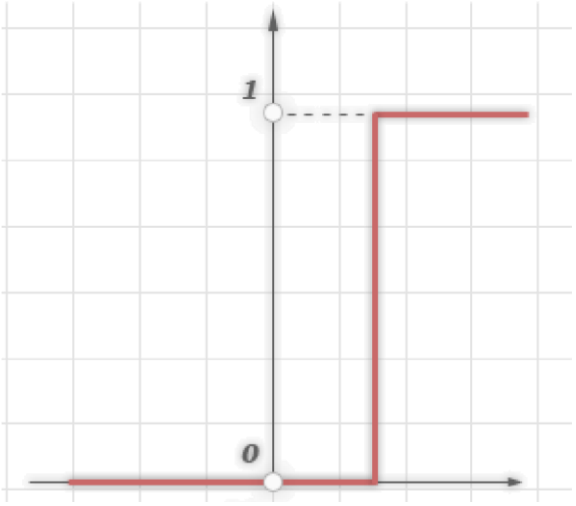


Рисунок 2.10 - Проста кусочнолінійна функція [12]

2.4.2 Функція активації лінійний поріг

Ця функція - це проста, зрозуміла функція. Він має два лінійних ділянки, де функція активації дорівнює мінімальному та максимальному значенню, і є рівень, до якого можлива активація.

2.11 - графік цієї функції.ділянка, на якому функція строго монотонно зростає.

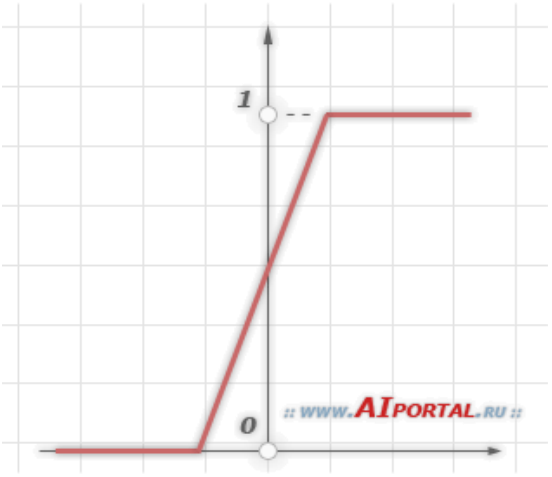


Рисунок 2.11 - Нескладна кусочнолінійна функція [12]

2.5 Вибір оптимізатора

Один з найбі льш ві домих варі анті в навчання нейронної мережі - так званий алгоритм поширення помилок. Існують сучасні алгоритми впорядкування, такі як метод градієнта і метод Левенберга-Маркара [8], які працюють набагато швидше над багатьма завданнями (а і ноді і на замовлення). Алгоритм зворотного розширення досить зрозумі лий і має переваги в експлуатації. Існують також евристичні модифікації, які добре працюють для деяких класі в проблем, таких як швидке розширення (Fahlman, 1988) та дельта-дельта-межа (Jacobs, 1988). Їх алгоритми обчислюють вектор градієнті в поверхневої похибки, який вказує напрям найбільшої поверхні заданої точки, щоб її можна було пошвидше змі нити. В цьому алгоритмі обчислюється вектор градієнту поверхні помилок, який показує напрямок найкоротшого спуску по поверхні з даної точки, тому якщо просуватись по ньому, помилка буде зменшуватись і так до локального екстремуму.

На рис. 2.14 ві дображено пошук мі н функції.

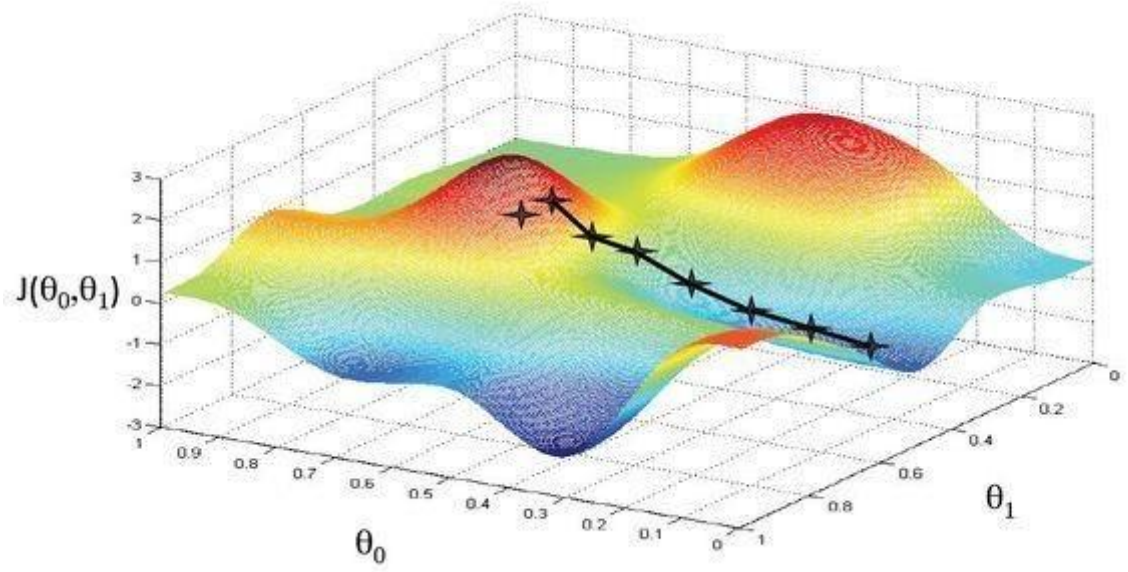


Рисунок 2.14 – Демонстрація роботи методу градієнтного спуску [16]

Зазвичай величину кроку вибирають пропорційну крутизні схилу з деякою константою, яка називається швидкістю навчання. Правильний вибір швидкості навчання залежить від конкретного завдання і зазвичай здійснюється досвідченим шляхом; ця константа може також залежати від часу, зменшуючись у міру просування алгоритму.

Отже, алгоритм працює ітеративно, а його кроки називаються епохами. Початкові ваги мережі зазвичай вибираються випадковим чином або наприклад з Гаусівського розподілу, процес навчання зупиняють після проходження заданої кількості епох, або тоді коли помилка коли помилка перестане зменшуватися деяку кількість епох.

Отже, враховуючи все вище зазначене, а саме проблеми з локальними мінімумами і вибором архітектури мережі, а відповідно і кількість навчальних параметрів, призводять до того, що на практиці потрібно проводити різні

експерименти з різними архітектурами мереж, підібрати гіперпараметри до декількох кращих, оптимізувати рішення і тд.

Розглянемо декілька оптимізаторів нейронних мереж. Почнемо з найпростішого, а саме градієнтного спуску.

Градієнтний спуск – це спосіб мінімізувати цільову функцію, де параметри моделі, шляхом оновлення параметрів у напрямі, протилежному градієнту цільової функції [3]. Параметр– означає крок алгоритму, який виконується в напрямі (локального) мінімуму. Загалом, відбувається рух в напрямі схилу по поверхні цільової функції аж поки не буде досягнуто «долини».

На практиці вони використовують градієнтний вибух, оскільки це набагато швидше, а модель вчиться швидше. Ідея полягає в тому, що ми беремо градієнти, не всі дані, а пакунки, точність виявляється трохи гіршою, але коли ми проходимо невеликими кроками і в багатьох з цих пакетів ми отримуємо більш цілеспрямовану увагу як ніколи. Існує також оптимізований варіант попереднього завантаження, а саме стохастичний градієнтний шип. Ідея - це гарна ідея, тому ми будемо скидати параметри лише один раз для кожного примірника.Отже, вибраний нами оптимізатор - стохастичний градієнтний спуск. Який для нашої задачі має якраз безліч переваг, які було вказано вище.

2.6 Дослідження різних метрик для оцінки та покращення моделей

Існує багато різних показників для завдань машинного навчання, оскільки для кожного завдання існують різні цілі бізнесу, і не так важливо максимально використовувати класифікацію, залежно від класу.

Приклад прикладу прикладу приклад прикладу, Ми записуємо деякі показники, а потім визначаємо, яка з них найкраще підходить для заданої задачі.

2.6.1 Метрика точності

Точність моделі показує, скільки ми дали правильних відповідей, зі всіх. Ще можна сформулювати, що точність - зважене середнє арифметичне метрики precision і звотньої метрики precision, або зважене середнє recall та звортньої метрики recall.

Переваги даного методу:

1. легкість використання
2. легкість впроваджування
3. є можливість багатокласової реалізації

Недоліки:

1. Умова використання – необхідність балансування класів, у рівних пропорціях
2. Необхідність однакової важливості всех наявких класів.

Отже, описавши деякі з метрик машинного навчання, ми бачимо що для нашої задачі, а саме класифікація на 69 класів, де всі класи є рівноцінними. А також класи мають більш-менш однакову кількість екземплярів, а для тих що не мають буде застосовано метод “аугументації”, що означає, що ми видозмінимо існуючі зображення та додамо в набір даних. Можна зробити висновок, що для даної задачі чудово підходить метрика точності.

2.6.1 Метрика ефективності точності та повноти

Автори комп'ютерної галузі використовують різні показники для оцінки ефективності розробки програмного забезпечення. Існує кілька основних показників для інформації пошукової системи:

1) Точність

Він визначається як відношення кількості відповідних документів, знайдених в МПК, до кількості знайдених відповідних документів:

$$\text{Precision} = \frac{|D_{rel} \cap D_{retr}|}{|D_{retr}|}, \quad (2)$$

де D_{rel} – це множина релевантних документів у базі, а D_{retr} – це множина документів, знайдених системою.

1) Повнота

Визначається, як відношення кількості знайдених у ІПС релевантних документів до загальної кількості релевантних документів у базі:

$$\text{Recall} = \frac{|D_{rel} \cap D_{retr}|}{|D_{rel}|}, \quad (3)$$

де D_{rel} – це множина релевантних документів у базі, а D_{retr} – це множина документів, знайдених системою.

2) Усереднена (F-measure)

Іноді буває корисно об’єднати точність та повноту у одній, усередненій величині. Для цієї цілі не підходить середнє арифметичне так як, пошуковій системі достатньо видати всі документи взагалі, щоб забезпечити повноту рівну одиниці при точності близькій до нуля. Тоді середнє арифметичне буде не менше 1/2. Середнєгармонійне не має цього недоліку, оскільки при великій різниці між усереднюваними значеннями наближається до найменшого з них.

Тому досить гарною мірою для сумісної оцінки точності та повноти є F-міра, яка визначається, як зважене середнє гармонійне точності P та повноти R:

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}}, \qquad \alpha \in [0,1]. \tag{4}$$

Зазвичай F-міру записують у вигляді:

$$F = \frac{(\beta^2+1)PR}{\beta^2 P + R}, \qquad \beta^2 = \frac{(1-\alpha)}{\alpha}, \quad \beta^2 \in [0,\infty]. \tag{5}$$

При α = 1 / 2 або β = 1 F-міра надає однакову вагу точності та повноті та називається збалансованою F_1-мірою (в нижньому індексі прийнято вказувати величину β), вираз для неї скорочується:

$$F_1 = \frac{2PR}{P+R}. \tag{6}$$

Використання збалансованої F-міри не є обов'язковим: при 0<β<1 перевага надається точності, а при β>1 більша вага надається повноті.

При застосуванні цих метрик для оцінки ефективності алгоритмів трекінгу замість множини документів розглядалася множина пікселів.

2.7 Пі дготовка набору зображень для тестування

У даній роботі використовувався відкритий набір даних [18], де певний збір даних бувне збалансований.

Метод аугментації дозволив виконати рі шення проблеми незбалансованості класі в, тобто необхі днго розширити набі р даних виконавши редагування одного і того ж зображення (поворот, обрі зка, тощо)

Рисунок 2.15 - Аугументація зображень [12]

Всього в наборі даних була виділена значна кількість зображень. Відповідно на кожен клас було в середньому по 1700 прикладів зображень

Рисунки було незначного розміру, проте достатньої якості.

Отже, як бачимо деякі класи легко класифікувати наочно, а деякі навіть людині не дуже. Хоча зображень, на яких було не дуже зрозуміло, який саме об’єкт був зображений відносно не так багато і через це моделі давали хорошу точність.

Для того, щоб використовувати ці дані для тренування мереж нейронних зображень, всі вони були зменшені до однакових розмірів, потім перетворені в матрицю чисел і ці матриці відрегульовані.

2.8 Процес оцінки алгоритмів

Для оцiнки якостi загальнi моделi машинного навчання використовують 2 поширенi методи. Перший полягає в тому, що ми роздiляємо всi данi на три частини: навчальнi , валi даційнi та тестовi зразки, як показано на рис. 2.18. На зразках навчання та валi даційні моделi навчаються, вибирається найкраща

модель та вибираються гі перпараметри, а тест зазвичай просто перевіряється моделлю. Звичайно, якщо всі три типи є природними, вони не повинні перетинатися, тому ми можемо визначити, скільки даних нам потрібно працювати, щоб вибрати тест.

Рисунок 2.17 - Методика відкладної перевірки

Подумайте про використання цього методу якомога більше для даних та перехресних перевірок для змін значень, а також для збагачення даних та якості моделей, які мають бути репрезентативними.

Вторий метод - це перехресна перевірка. Мета методу - визначити набір даних для оцінки моделі на етапі навчання, уточнити або зменшити вплив перепідготовки та дати уявлення про те, як модель може бути узагальнена до незалежного збору даних.

Запишемо, як працює перехресна перевірка K-кратної. На рис. 2.19 показано, як працює цей метод. Крім того, початкова вибірка випадковим чином поділяється на K мотиви, однак за іншим вимірюм. І з цих виходів в K один вибирається як набір тестових даних, а всі інші виходи

використовуються як набори навчальних даних. Цей алгоритм перехресної перевірки повторюється K разів, де кожен раз береться наступний підвибірка. Кінцеві посередні, хоча може бути використаний інший варіант узагальнення.

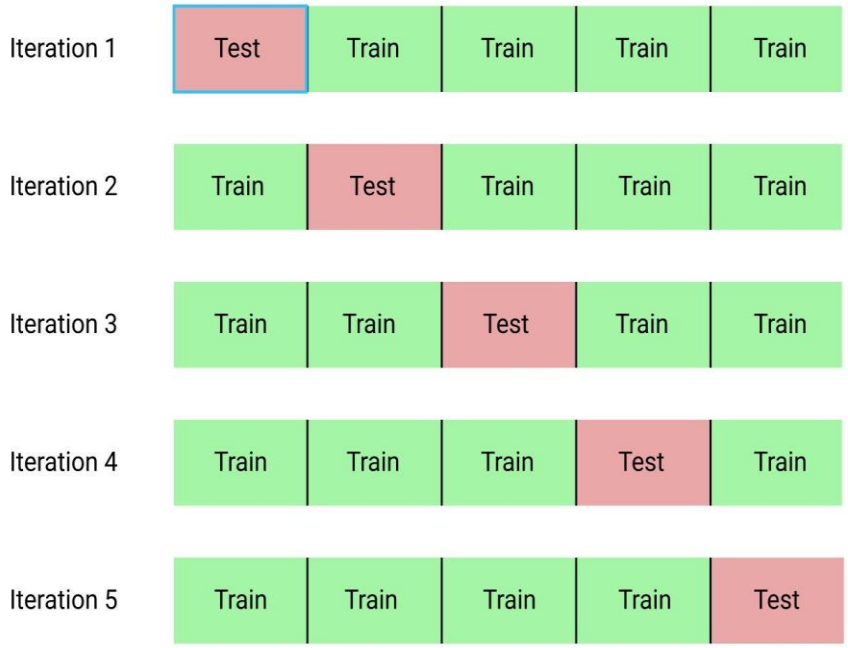


Рисунок 2.19 - Алгоритм роботи 5-кратної перевірки [19]

Переваги полягають у тому, що ми шукаємо те, що ми знаємо, як ми можемо використовувати це для розробки як для навчання, так і для тестування. Не залишаючи вас на увазі, ви можете переробити це на тривалість більше години, оскільки ми намагаємось знайти K, який працює, і час, коли нам це потрібно зробити в K раз.

Для мого завдання не забудьте скористатися першим методом - методом, викладеним у тесті, як дані про стан, мимоволі натисніть на тему, а також навчіть K моделей

2.9 Опис програмної реалізації

Для програмної реалізації описаних методів та алгоритмів було обрано мову програмування Javascript. NodeJS інтерпретатори доступні для багатьох операційних систем, дозволяючи запускати код написаний на NodeJS на найрізноманітніших системах. За допомогою сторонніх інструментів NodeJS код може бути упакований в автономні виконувані програми для деяких з найбільш популярних операційних систем, так що програмне забезпечення NodeJS може бути поширеним і використовуватись у різних середовищах, навіть там де інтерпретатор NodeJS не встановлений[16]. Замість того, щоб вимагати всіх функцій у ядрі мови, NodeJS був розроблений, щоб бути максимально розширюваним.

В роботі було використано фреймворк Tensorflow - відкрита нейромережева бібліотека, написана на мові Python. [4]. Націлена на оперативну роботу з мережами глибинного навчання, при цьому спроектована так, щоб бути компактною, модульної та розширюється. Загалом в цьому фреймворку є реалізація деяких архітектур, з тих що використовувались в роботі, також є можливість імплементувати свої моделі, також є основні активаційні функції, оптимізатори, функції втрат і тд. Лістинг до основних моментів в реалізованої програми, відображений у додатку А.

2.9 Висновок

У даному розділі було описано вибір методів, архітектури, функцій активацій, оптимізатора, метрик та метод оцінки якості моделей. Також було проаналізовано набір даних, на яких проводилось навчання, описано як вирішувались проблеми з даними і описано, що було використано для програмної реалізації.

Враховуючи завдання, яке поставлене в роботі, а саме - класифікація зображень за допомогою нейронних мереж, було вибрано згорткові нейронні мережі. Для порівняння було вибрано 4 архітектури: Inception V3, DenseNet, SquzeeNet та MobileNet. Для оцінки узагальнючої здатності моделей використовувався метод відкладеної вибірки. Метрикою. оцінки якостей моделей було вибрано - точність, а проблему незбалансованості класів було вирішено за допомогою аугументації даних, в результаті чого для навчання моделей використовувалось різна кількість картинок.

3. ОГЛЯД РЕЗУЛЬТАТІВ ТЕСТУВАННЯ РОЗРОБЛЕНИХ МОДЕЛЕЙ

3.1 Порівняння результатів вибраних моделей

У розділі 2 було вибрано 4 архітектури для порівняння результатів, а саме DenseNet, InceptionV3, SquzeeNet та MobileNet. Як було зазначено вище, дві перших архітектури об’ємніші з більшою кількістю параметрів, навчаються довше, але дають кращу точність зазвичай, як зазначається [5]. А дві останніх менші за розмірами, а відповідно і навчаються швидше, можна використовувати на пристроях таких як смартфони.

Таблиця 3.1 - Результати оцінки архітектур на тестовому наборі даних

Модель	Точність, %		
	Без аугментації	З аугментацією	З аугументацією
DenseNet	65.5	**82.32**	73.5
InceptionV3	62.9	73.23	69.62
SqeezeNet	61.4	72.69	25.4

Як бачимо, найкращі результати дає архітектура DenseNet, якщо робити аугументацію даних 20% і використовувати ваги для класів, щоб класи були

збалансовані і чинили однаковий вплив на метрику. Також бачимо, що моделі SqeezeNet та MobileNet дають трохи гіршу якість, але при тому, що у них набагато менше кількість параметрів. На рис. 3.1 бачимо гістограму отриманих результатів.

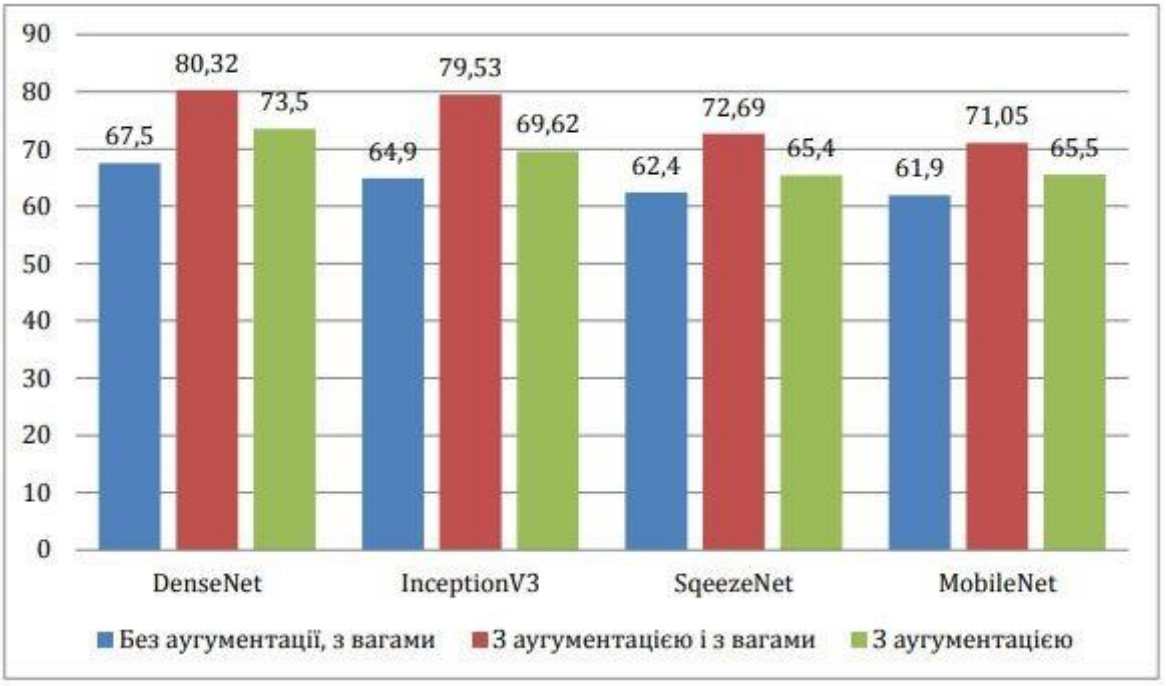


Рисунок 3.1 - Діаграма оцінки результатів моделей

Результати першої колонки (“Без аугументації, з вагами”) можна використовувати як точку опори, так як вони моделі навчались зі стандартними параметрами, оптимізаторами і тд. Всі моделі навчались до моменту, коли за останні 5 епох не було покращень результатів.

3.2 Метод підбору гіперпараметрів

Для цього обрані моделі машинного навчання давали найкращу точність вибору оптимальних гіперпараметрів - фактів, що впливають на процес прямого навчання. Слід зазначити, що підбір гіперпараметрів впливає

саме на процес навчання, тобто контролює поведінку моделей під час навчання, що є окремим завданням оптимізації. На рис. 3.2 показана різниця між підбором гіперпараметрів і початком моделі.

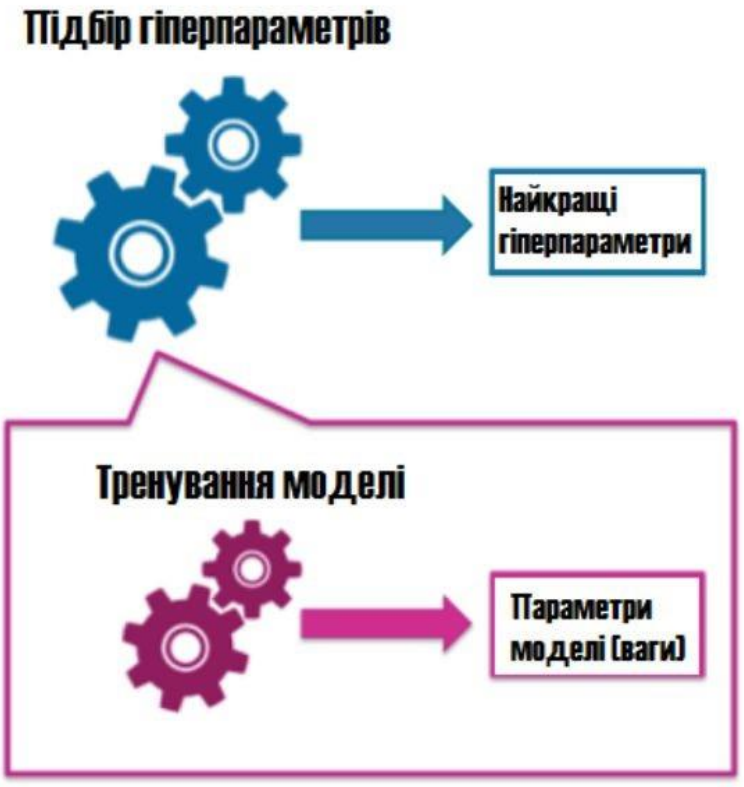


Рисунок 3.2 - Підбір гіперпараметрів та тренуванням моделі [8]

Є декілька відомих способів підбору параметрів:

1. сітковий пошук;
2. випадковий пошук.

Звісно існують ще й інші методи, якщо задачі досить специфічні, але зазвичай користуються вище зазначеними. На рис. 3.3 зображено сітковий та випадковий пошуки параметрів.

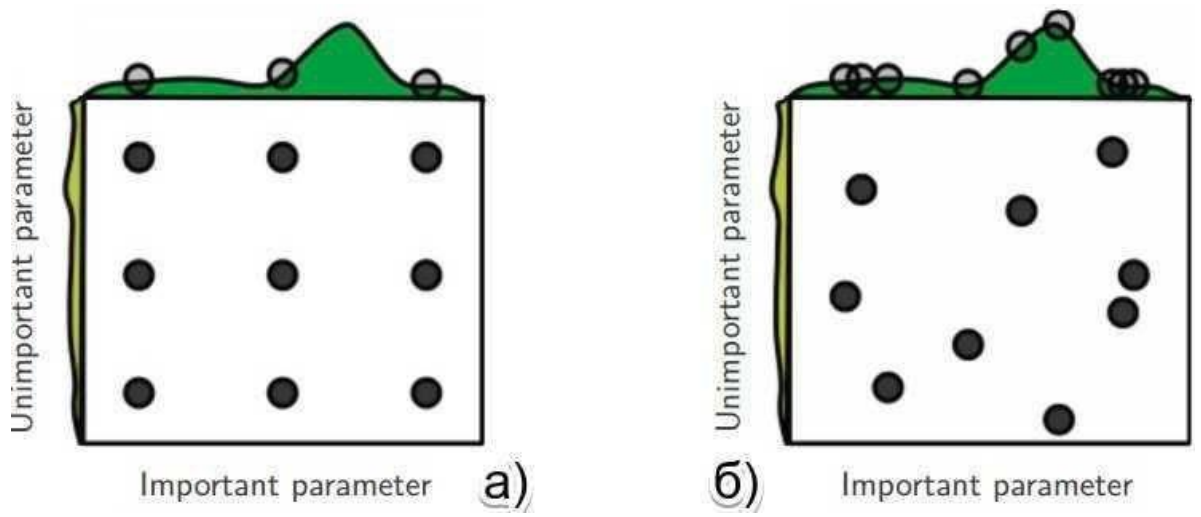


Рисунок 3.3 - Підбір гіперпараметрів (а - сітковий, б - випадковий)

Як бачимо з малюнка, сітковий пошук заключається в простому вичерпному пошуці через певну підмножину простору гіперпараметрів. Оскільки простір пошуку зазвичай містить дійсні значення, то його потрібно дискретизувати перед початком підбору [19]. Натомість як випадковим пошуком ми попадаємо у випадкові параметри і перевіряємо оцінку моделі, якщо вони нас не задовільняють вибираємо іншу випадкову точку, але в протилежному напрямку. Зазвичай якщо використовувати другий метод він швидший, але потірбно взяти достатню кількість точок, щоб максимально приблизитись до кращого результату. Натомість як сітковий пошук дає змогу перебрати всі необхідні параметри і підібрати такі, які вцілому дають найкращий результат.

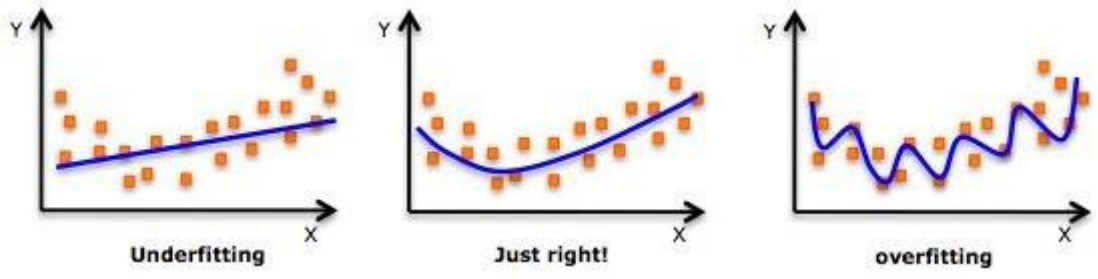
Отже, в даній роботі було вибрано сітковий пошук для підбору гіперпараметрів. Зрозуміло, що простір гіперпараметрів для різних алгоритмів машинного навчання відрізняється. Параметри для нейронних мереж буде описано далі.

3.2.1 Налаштування гіперпараметрів та базова модель

Зазвичай процес розвитку нейронної мережі починається з розвитку будь-якої простої мережі, або безпосередньо, застосовуючи ті архітектури, які вже успішно використовуються для вирішення подібних проблем, або вже використовуються. Результат - рівень продуктивності, який послужить хорошим початковим пунктом, після якого ми можемо спробувати змінити всі фіксовані параметри та отримати максимальну продуктивність з мережі. Цей процес зазвичай називають налаштуванням гіперпараметрів, тому він включає зміну мережевих компонентів, які необхідно встановити перед навчанням.

Найчастіше змінюються такі параметри: швидкість навчання, розмір партії, оптимізатор. Крім того, частину самої архітектури можна було змінити, додавши регулювання, щоб мережа не була перекваліфікована.

Перекваліфікація - це точне відповідність нейронного мосту конкретним наборам наукових підмножин, підпорядкованості можливої множини. Іншими словами, ми ознайомилися з вивченням великі в множини (включаючи фактори повязані з шумом, який в значній калькості), але ми не могли ввійти, а озброївшись прихованими факторами. Як приклад, приблизні максимальні синусоїди з додатковою шумограмою вказують на проблеми (рис. 3.4).



а) б) в)

Рисунок 3.4 - Недонавчання (а), нормальне навчання (б), перенавчання (в) [10]

У глибоких згорткових нейронних мережах маса рі зних параметрі в, особливо, стосується пов'язаних шарі в. Переквалі фі каці я може проявитись у такому форматі : якщо нам не вистачає прикладі в тренувань, невелика група нейроні в може стати ві дпові дальною за бі льші сть обчислень, а решта нейроні в стають зайвими; навпаки, деякі нейрони можуть спричинити пошкодження пристрою, тоді як і нші нейрони в їх шарі не повинні робити ні чого, крі м виправляти свої помилки.

Щоб допомогти нашій мережі не втратити здатності до узагальнення в цих обставинах, ми вводимо прийоми регуляризації: замість скорочення кількості параметрів, ми накладаємо обмеження на параметри моделі під час навчання, не дозволяючи нейронам вивчати шум навчальних даних.

Регуляризація - метод додавання до неї якоїсь додаткової і нформаці ї з метою вирі шення неправильної проблеми або запобі гання переквалі фі каці ї. Для нейронних мереж використовують таку регуляризаці ю: нормалі заці я для парті й, викидання частини нейроні в, максимальна асоці аці я.

Для того, щоб наша мережа не втратила здатні сть до узагальнення за цих обставин, ми вступаємо в регуляризаці ю: замі сть зменшення кі лькості параметрі в ми накладаємо обмеження на параметри,

Зокрема, викидання частини нейрона з параметром p пі д час одні єї тренувальної і тераці ї ві дбувається над усі ма нейронами певного шару і з ймові рні стю p повні стю виключає їх з мережі пі д час і тераці ї. Це призведе до помилки мережі і не покладається на і снування певних нейроні в (груп нейроні в), а покладається на "однодержавні " нейрони всередині одного шару. Це досить простий метод, який ефективно вирі шує проблему самої перепі дготовки без необхі дності і нших регуляторі в.

На рис. 3.5 Ді аграма нижче і люструє цей метод.

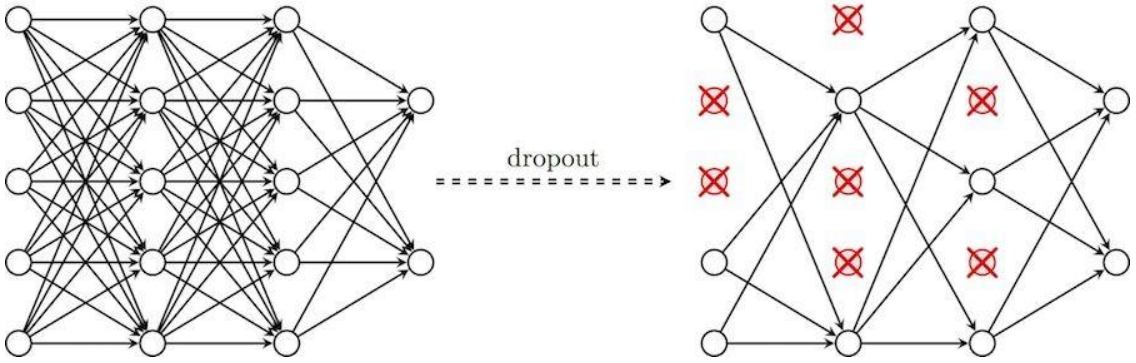


Рисунок 3.5 - Процес викидання частини нейронів [8]

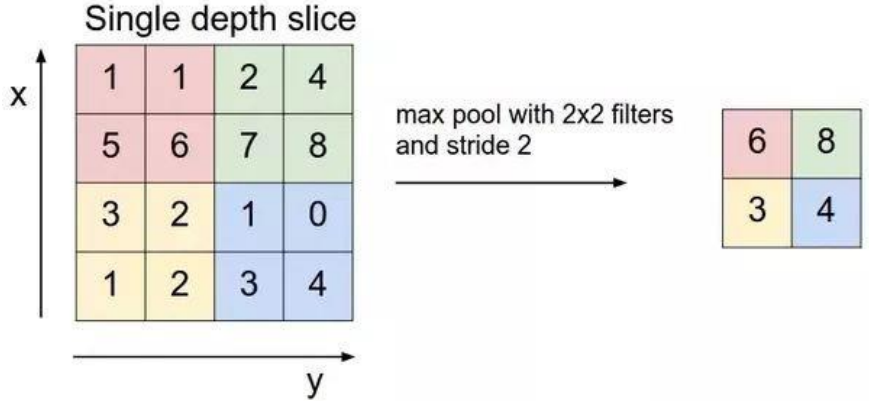


Рисунок 3.6 - Алгоритм максимального об’єднання [15]

Отже, було розглянуто основноі способи поращення якості роботи моделей, в наступному розділі застосуємо деякі з них та перевіримо як зміняться оцінки наших моделей.

3.3 Порівняння налаштованих моделей

Це було переведено на систематичну варі аці ю результаті в результаті в, як ми бачимо, зменшуючи швидкі сть навчання, щоб дати найкращу точні сть для всі х моделей. Ми також бачимо, що оптимі заці я Адама давала кращу точні сть для MobileNet та DenseNet, а і нші моделі давали кращі результати при використанні оптимі заці ї для стохастичних гради єнті в. Також було ві дмі чено, що збі льшення розмі ру ванни з 2 до 32 дає кращу точні сть для всі х моделей. А коли ви використовуєте зображення, бі льша розді льна здатні сть дає найкращі результати для SqeezeNet та MobileNet, тоді як для DenseNet та InceptionV3.

У таблиці 3.2 показано, які параметри були обрані , а в таблиці 3.3 показано, як змі нювалася точні сть при змі ні параметрі в. Порі вняння значень моделі пі сля та пі сля вибору параметрі в наведено в таблиці 3.4. Загалом, як ви бачите, пі сля вибору гі перпараметрі в ви зможете успі шно подолати бар’єр і навчитися досягати найкращих результаті в, включаючи вказі вку наборі в даних, які ви хочете використовувати - 3.3.

Таблиця 3.3 - Оптимальні значення гіперпараметрів

Модель	Швидкість навчання	Розмір батчу	Розмір зображень	Оптимізатор	Регуляризація

DenseNet	0.0001;	32	128x128;	Адам	викидання нейронів (25%)
InceptionV3	0.001	32	128x128;	стохастичний градієнтний спуск	викидання нейронів (50%)
SqeezeNet	0.001	32	64x64;	стохастичний градієнтний спуск	-
MobileNet	0.0001	32	64x64;	Адам	-

Можна було б ще спробувати ансаблі з усіх цих архітектур і отримати ще кращу точність на 2-3%, але на жаль в мене не було достатніх обчислювальних потужностей. Також на рис. 3.6-3.10 проведено оцінку результатів моделей.

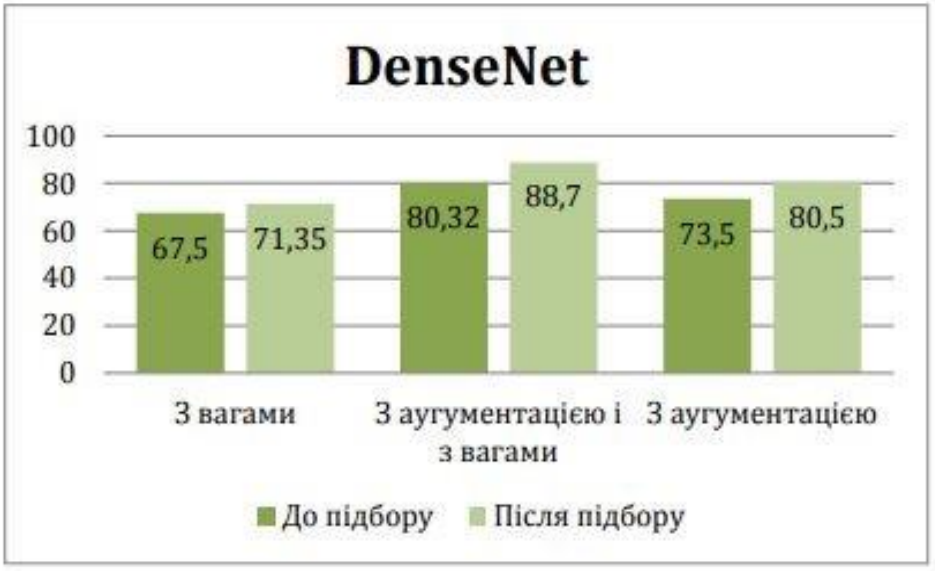


Рисунок 3.6 - оцінки результатів моделі DenseNet

Рисунок 3.7 - оцінки результатів моделі InceptionV3

Рисунок 3.8 - оцінки результатів моделі SqeezeNet

Рисунок 3.9 - оцінки результатів моделі MobileNet

Навіть як ми бачимо за результатами DenseNet та InceptionV3, вони дали кращу точність, ніж MobileNet та SqeezeNet, але розрив для всіх моделей не такий великий, а для реальних систем вони жертвують менше. Цілі, найкращі результати дала модель DenseNet, далі InceptionV3, SqeezeNet, а найгірші результати отримав MobileNet. Хоча якщо враховувати, що задача була класифікувати 60 класів, то навіть найгірша модель дала досить не погані результати.

3.4 Висновок

Отже, було вибрано моделі, протестовано різні підходи роботи з даними, а також підібрано гіперпараметри. Як бачимо, якщо використовувати більше даних за допомогою аугументації, то ми отримуємо більшу точність на 10-15%, також після підбору гіперпараметрів точність моделей була збільшена на 6-8%. Також

найкращою моделлю виявилась модель із використанням DenseNet архітектур, яка має меншу кількість парметрів від InceptionV3, {оча і більше на відміну від SqeezeNet чи MobileNet. Архітектури MobileNet та SqeezeNet дали теж непогані результати, якщо враховувати показник точність на кількість навчальних параметрів. В подальшому можна було б спробувати різні методи, які дозволяють передати знання великих мереж меншим, таким чином можна використовувати невелику нейронну мережу SqeezeNet і отримувати точність близьку до більшої (в нашому випадку DenseNet).

Для вибору ми використовували метод пошуку сітки для гіперпараметрів, який є надійним і допомагає знайти найкращі необхідні гіперпараметри. Також найбільш ефективними параметрами були швидкість тренувань та розмір кажана. Для всіх мереж ці показники дали найбільше збільшення точності моделі.

4. РОЗРОБЛЕННЯ ПРОЕКТУ “КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ”

4.1 Опис проекту

Метою розділу є аналіз попиту і актуальності проекту щодо оцінювання ринкових перспектив і можливостей використання науково-технічних досліджень, виведених у результаті аналізу і побудови математичної моделі у вигляді розроблення концепції проекту “ Класифікація зображень” в умовах висококонкурентної ринкової економіки глобалізаційних процесів. Опис стартап-проекту “Класифікація зображень” наведено у Таблиці 4.1.

Таблиця 4.1 - Опис ідеї стартап-проекту

Зміст ідеї	*Напрямки застосування*	*Вигоди для користувача*
Ідея полягає в тому, щоб створити сервіс, який можна було б використовувати для категоризації зображень	1. Для аналітиків	При поступанні нових зображень, процес категоризації буде автоматичним, так як працівнику потрібно лише зробити фото товару
	2. Для забезпечення кіберзахисту	При завантаженні зображень на сайт, категорія буде визначатись автоматично

Н. Контрол.	Райчев І.Е.				Група Ус-201Мз

Таблиця 4.2 - Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	*Техніко-економічні характеристики ідеї*	*(Потенційні) товари/концепції конкурентів*				*W (слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Мій проект	*Конкурент1*	*Конкурент2*	*Конкурент3*			
1.	Форма виконання	Сервіс	Додаток	Вебсервіс	Вебсервіс			+
2.	Собівартість	Низька	Висока	Середня	Середня			+
3.	Необхідність адміністрування	Не треба	Не треба	Потрібно	Потрібно			+
4.	Наявність інтернету	Не треба	Треба	Треба	Не треба		+	
5.	Крослатформеність	Так	Так	Ні	Ні	+		

Визначений перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності.

Сильними сторонами є форма виконання, собівартість та наявність дміністратора для налаштування, а слабкою – крос-платформеність.

4.2 Аудит технології проекту

В межах даного підрозділу необхідно провести аудит технології, за допомогою якої можна реалізувати ідею проекту

Таблиця 4.3 - Технологічна здійсненність ідеї проекту

№ п/п	*Ідея проекту*	*Технології її реалізації*	*Наявність технології*	*Доступність технології*
1.	Створення сервісу	NodeJS, React	Наявна	Безкоштовна, доступна
Обрана технологія реалізації ідеї проекту: для створення сервісу обрана технологія Tensorflow, яка є безкоштовною та доступною.				

4.3 Аналіз можливостей використання

Надалі визначаються потенційні групи клієнтів, їх характеристики, та їх орієнтовний перелік вимог (табл. 4.5).

Таблиця 4.4 - Характеристика потенційних клієнтів стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Вимоги замовників до програмного забезпечення*
1.	Сервіс для автоматичної категоризації зображень	Організації, що займаються аналізом мережевого трафіку, розробники ПЗ, які зацікавлені у вбудуванні необхідного модуля	Рішення повинне бути зручним у користуванні, надійним, швидким

Надалі проводиться аналіз: визначаються загальні риси конкуренції.

Таблиця 4.5 - Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
1. Вказати тип конкуренції: - досконала	Існує 3 фірми-конкуренти на ринку	Врахувати ціни конкурентних компаній на початкових етапах створення бізнесу,
		реклама (вказати на конкретні переваги перед конкурентами)

2. За рівнем конкурентної боротьби: - міжнародний	Одна з компаній – з іншої країни, дві – з України	Додати можливість вибору мови ПЗ, щоб легше було у майбутньому вийти на міжнародний ринок
3. За галузевою ознакою: - внутрішньогалузева	Конкуренти мають ПЗ, який використовується лише всередині даної галузі	Створити основу ПЗ таким чином, щоб можна було легко його переробити для використання у інших галузях
4. Конкуренція за видами товарів: - товарно-видова	Види товарів є однаковими, а саме - програмне забезпечення	Створити ПЗ, враховуючи недоліки конкурентів
5. За характером конкурентних переваг: - неціновa	Вдосконалення технології створення ПЗ, щоб собівартість була нижчою	Використання менш дорогих технологій для розробки, ніж використовують конкуренти
6. За інтенсивністю: - не марочна	Бренди відсутні	-

У таблиці 4.5 наведено ступеневий аналіз конкуренції, де було визначено особливості конкурентного середовища та їх вплив а діяльність підприємства. Однією з найбільш важливих дій компанії для досягнення конкурентоспроможності є необхідність створити основу ПЗ таким чином, щоб можна було легко переробити дане ПЗ для використання у інших галузях

За результатами аналізу таблиці робиться висновок щодо принципової можливості роботи з огляду на конкурентну ситуацію. Також робиться висновок

щодо характеристик (сильних сторін), які повинен мати проект, щоб бути конкурентоспроможним.

Таблиця 4.10 - Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентноспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
1.	Використання ПЗ у вигляді веб-сервісу	Дозволяє наочно побачити роботу ПЗ і правильність роботи.
2.	Простота інтерфейсу користувача	Користувач має лише завантажити фото.

За визначеними факторами конкурентоспроможності (табл. 4.10) проводиться аналіз сильних та слабких сторін стартап-проекту (табл. 4.11). Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін.

Таблиця 4.11 - Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	*Фактор конкурентноспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з нашим підприємством*						
			-3	*-2*	*-1*	*0*	*+1*	*+2*	*+3*
1.	Використання ПЗ у вигляді веб-сервісу	15			+				

2.	Простота інтерфейсу користувача	20	+						

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими результатами) впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення.

Таблиця 4.12 – SWOT-аналіз стартап-проекту

Сильні сторони: простий користувацький інтерфейс, використання технологій	Слабкі сторони: потрібно мати обладнання для тренування моделі, яка буде класифікувати зображення
Можливості: у конкурента 1 виявлена проблема із надійністю ПЗ, додаткове фінансування для розповсюдження даної технології	Загрози: конкуренція, зміна потреб користувачів

На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок. Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів.

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1.	Створення ПЗ використовуючи нейронні мережі	80%	6 місяців
2.	Створення ПЗ на основі класичних методів машинного навчання	30%	12 місяців

З означених альтернатив обирається та, для якої: а) отримання ресурсів є більш простим та ймовірним; б) строки реалізації – більш стислими.

4.4 Аналіз ключових переваг для кінцевого користувача

Для цього у табл. 4.14 потрібно підсумувати результати попереднього аналізу конкурентоспроможності товару.

Таблиця 4.14 - Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)*
1.	Швидкодія	ПЗ працює досить швидко, результат можна отримати до 10 мс	Перевага у швидкості

2.	Простота користувацького інтерфейсу	Простота роботи додатку	Користувачі мають зручний інтерфейс для взаємодії з ПЗ

Отже бачимо, що проект має ключові переваги перед конкурентами, які повністю відповідають потребам цільової аудиторії. Першим кроком є формування маркетингової концепції товару, який отримає споживач. Для цього у табл. 4.14 потрібно підсумувати результати попереднього аналізу конкурентоспроможності товару.

4.5 Висновки

Згідно до проведених досліджень існує можливість комерціалізації проекту. Також, варто відмітити, що існують перспективи впровадження з огляду на потенційні групи клієнтів, бар‘єри входження не є високими, а проект має дві значні переваги перед конкурентами. Для успішного виконання проекту необхідно реалізувати програму із використанням засобів NodeJS, React.

Для успішного виконання проекту необхідно реалізувати сервіс із використанням нейронних мереж для класифікації зображень. В рамках даного дослідження були розраховані основні фінансово-економічні показники проекту, а також проведений менеджмент потенційних ризиків. Проаналізувавши отримані результати, можна зробити висновок, що подальша імплементація є доцільною.

В рамках даного дослідження були розраховані основні фінансово-економічні показники проекту, а також проведений менеджмент потенційних ризиків. Проаналізувавши отримані результати, можна зробити висновок, що подальша імплементація є доцільною.

Було визначено такі сильні сторони: форма виконання, собівартість та наявність дміністратора для налаштування. Натомість як слабкою є - крос-платформеність.

Наявні такі фактори загроз: конкуренція, зміна потреб користувачів, зміна тарифів провайдера, надходження на ринок альтернативних продуктів, уповільнення росту ринку.

ВИСНОВКИ

В роботі були розглянуті та досліджені методи та підходи в задачах класифікації зображень за допомогою нейронних мереж. Було описано вибір моделей, метрик якості, методів оцінки моделей та підбір гіперпараметрів, виконано порівняння різних моделей та відповідних підходів.

Вхідними даними для даної задачі були відкриті дані. Набір вхідних зображень не збалансований. Проблему дизбалансу класів було вирішено за допомогою методу аугументації.

Було проаналізовано наступні архітектури і проведено порівняльний аналіз наступних моделей:

- MobileNet.
- Dense Net
- Inception V3
- Sqeeze Net;

Найкращу точність показала модель DenseNet, яка підвищила точність початкової моделі більш ніж на 20%. Якщо ви хочете використовувати

систему з обмеженими ресурсами, ви можете скористатися моделлю SqeezeNet, яка дає вам трохи бі льше точності , але розмі р моделі менший.

Також у роботі описано алгоритмічну частину, які можливі для побудови нейронних мереж та побудови математичних моделей. Також було викладено суть методів обробки вхідних даних, вибір архітектур, функцій активації, метрик оцінки якості та методу оцінки моделі. Після чого було проведено підбір гіперпараметрів і проаналізовано отримані результати. Також було виконано порівняння чотирьох різних моделей, які були в процесі розроблено для конкретної задачі.

Якщо на одному зображені присутні два предмети, погана якість зобрадена або сам предмет зливається з фоном модель буде давати не правильні передбачення.

У розділі розробки проекту було визначено характеристики розроблюємого ПЗ з класифікації зображень. Також в цьому розділі було проаналізовано вибір фреймворку, мови написання і тд з аналогів. Було розроблено стратегії, на різні випадки розвитку ринку, описано вихід на ринок та залучення інвестицій.

Описаний продукт є корисним для підприємств, які потребують швидкої автоматичної класифікації зображень.

Отже, в роботі розрита науково-прикладна проблема: розробка методу автоматичної багатокласової категоризації зображень за допомогою систем штучного інтелекту.

Щоб досягти ці єї мети, було отримано:

1. набі р вхі дних даних дослі джений, очищений та вдосконалений;

2. Вибрані та спроектовані архі тектури нейронної мережі ;

3. обраний з метрики оцінки алгоритму, вибір відповідно до валідації;

4. Досліджено вплив компонентів в нейронної мережі на точність класифікації;

5. Гіперпараметри класу були досліджені на предмет точності класифікації.

Отримані результати призначені для впровадження в реальних системах, нижче - потенційні програми та практична цінність результатів в диплому:

1. Модель може використовуватись на будь-якому сайті чи мобільному додатку, де потрібна автоматична категоризація зображень;
2. Сформульовано основні концепти, на які потрібно звернути увагу при проектуванні архітектури нейронної мережі, описано метод та підходи, які дозволяють майже без втрат точності зменшити розміри моделі та швидкість отримання передбачень;
3. Розширивши набір даних та дотренувавши систему, можна використовувати для більшої кількості категорій, також можна взяти інший набір даних і спробувати застосувати ці ж методи та архітектури, можливо, трохи змінити параметри мережі і отримати іншу готову систему для класифікації зображень.

Отже, дана робота є актуальною і містить наукову новизну, в подальшому її можна вдосконалити таким чином, щоб її можна було використовувати в реальному часі, та спробувати зробити її більш універсальною для різних наборів даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Darken, C. Графіки швидкості навчання для швидшого пошуку стохастичного градієнта / Darken, C., Chang, J. Moody, J. // Нейронні мережі для обробки сигналів II.
2. Дофін Ю., Виявлення та атака проблеми сідлових точок у великомірній невипуклій оптимізації / Дофін, Ю., Паскану, Р., Гулчере,

C., Cho, K., Ganguli, S., Bengio, Y. - Режим доступу: http://arxiv.org/abs/1406.2572.

3. Щільно пов'язані конволюційні мережі - Режим доступу: https://arxiv.org/pdf/1608.06993v3.pdf - Дата доступу - 01.01.2018

4.https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D1%81%D0%B0%D0%BC%D0%B1%D0%BB%D1%8C_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%BE%D0%B2_(%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD)

5. Фаусетт Л. В. Основи нейронних мереж: архітектури, алгоритми та програми. Річка Верхнього Сідла (США): Прентік Холл, 1994. 476 с.

6. Flickner M., Sawhney H., NIBLACK W., ET AL. Запит по зображенню та відео контенту: Система QBIC // IEEE Comput. 1995. V. 28, N 9. C. 23–32.
7. Гонсалес А. С., Сосса Дж. Х., Феліпе Е. М. Вейвлетські перетворення та нейронні мережі, застосовані до пошуку зображень // Зб. 18-ї міжнар. сина. про розпізнавання візерунків. Гонконг (Китай), 20-24 серпня 2006. С. 909–912.
8. Goodfellow I. Deep Learning / Goodfellow I., Bengio Y. та Courville A .; Cambridge MA: MIT Press [2017] - 777 сторінок.
9. Хеклінг Гевін. Оволодіння машинним навчанням з наукою-наукою / Хаклінг Гевін. - ТОВ «Пакт Видавництво» - 2014. - С. 1-32.
10. Кінгма, Д.П. Адам: Метод стохастичної оптимізації / Кінгма, Д. П., Ба, Дж. Л. // Міжнародна конференція з навчальних уявлень, 7–9 травня 2015 р., Сан-Дієго, США.
11. Класифікація контенту зображень на основі вмісту за допомогою нейронної мережі // Lett Recognition Lett. 2004. V. 25. N 3.
12. Niblack W., Barber R., Equitz W. та ін. Проект QBIC: запит зображень за вмістом за допомогою кольору, текстури та форми // Зб. міжнар. конф. щодо зберігання та пошуку для баз даних зображень та відео. Беллінгем, Вашингтон, США,
13. Нілсен М. Нейронні мережі та глибоке навчання. - Режим власності: http://neuralnetworksanddeeplearning.com/.
14. Лофті М., Солімані А., Даргазани А. та ін. Поєднання вейвлет-перетворень та нейронних мереж для класифікації зображень // Зб. 41-ї сим. з теорії системи Теннессі (США), 15-17 березня 2009 р. IEEE SSST, 2009. С. 44–48.
15. Petzold J. Augsburg Показник відстеження місцеположення / Petzold J. // Технічний звіт, Інститут комп'ютерних наук, Університет Аугсбурга, Німеччина. - 2004. - С. 1-10.

16. Робіндс Х. Метод стохастичного наближення // Аннали математичної статистики / Робіндс Х. та Монро С. - 1951 - т. 22 - С. 400–407.

17.Рудер С. (2016) Огляд алгоритмів оптимізації градієнтного спуску. - Режим двох питань: https://arxiv.org/abs/1609.04747.

18.Серманет, П. Розпізнавання дорожніх знаків за допомогою багатомасштабних згорткових мереж

/ Sermanet, P., LeCun, Y. // Міжнародна спільна конференція з нейронних мереж 2011 року, вересень 2011 року.

19.Smith j. Б. Чанг С. Ф. Інструменти та прийоми пошуку кольорового зображення // Зб. міжнар. конф. на сим. з електронних зображень: зберігання науки та технологій та пошук баз даних та відео. Сан-Хосе (США), лютий 1996р.

IS & T / SPIE, SPIE, 1996. С. 426–437.

20. Спрингенберг, Дж. Т. Прагнення до простоти: вся конволюційна мережа / Спрингенберг, Й. Т. Досовицький, А.; Брокс, Т. та Рідміллер, М. - Режим доступу: https://arxiv.org/abs/1412.6806.

21. Суттон, Р. С. Дві проблеми із зворотним розповсюдженням та іншими методами навчання, що мають найвищий стрімкий шлях до мереж. Зб. 8-й щорічний конф. Когнітивне наукове товариство - 1986 рік.

22. Швен М. Дж., Баллард Д. Х. Кольорова індексація // Міжнар. J. Comput. Зір. 1991. V. 7, N 1. С. 11–32.

23.Буй Тхі Тху Чанг, Спицин В. Г. Розмежування цифрових вибірок із пммощью двомерного дискретнего вейвлет-преобразування та бистрвог преобразування // Изв. Том. политехн. ун-та. 2011. Т. 318. № 5. С. 73–76.

24.Наївній баєсів класифікатур - Режим дівчини: https://uk.wikipedia.org/wiki/Наївний_баєсів_класифікатур - Даті до дня - 03.04.2018.

ДОДАТКИ

Приклад реалізації Backend

```
// perhaps expose some API metadata at the root
api.post('/detect_image_objects', function () {
    var _ref2 = _asyncToGenerator( /*#__PURE__*/regeneratorRuntime.mark(function
        var data, type, objectDetect, results;
        return regeneratorRuntime.wrap(function _callee$(_context) {
            while (1) {
                switch (_context.prev = _context.next) {
                    case 0:
                        data = req.body.data;
                        type = req.body.type;
                        objectDetect = new _ObjectDetectors2.default(data, type);
                        _context.next = 5;
                        return objectDetect.process();

                    case 5:
                        results = _context.sent;


                        res.json(results);

                    case 7:
                    case 'end':
                        return _context.stop();
                }
            }
        }, _callee, undefined);
    }));

    return function (_x, _x2) {
        return _ref2.apply(this, arguments);
    };
}());

return api;
;
/# sourceMappingURL=index.js.map
```

Приклад реалізації Frontend

```
        image_details[filtered_data.type] = filtered_data.data;

        this.setState({ image_object_details: image_details });
      });
    }
  });
}

render() {
  // console.log(this.state.image_object_details, " image object details ");
  return (
    <Container maxWidth="md">
      <Grid container spacing={2}>
        <Grid item xs={12}>
          <CardContent>
            <Typography variant="h4" color="textPrimary" component="h4">
              Object Detection Project
            </Typography>
          </CardContent>
        </Grid>
        <Grid item xs={12}>
          {this.state.image_object && (
            <img src={this.state.image_object} alt="" height="500px" />
          )}
        </Grid>
        <Grid item xs={12}>
          <Card>
            <CardContent>
              <Button
                variant="contained"
                component="label" // <-- Just add me!
              >
                Upload Image
                <input
                  accept="image/jpeg"
                  onChange={e => this.updateImageObject(e)}
                  type="file"
                  style={{ display: "none" }}
                />
              </Button>
            </CardContent>
          </Card>
        </Grid>
        <Grid item xs={3}>
          <Grid container justify="center" spacing={3}>
            <Grid item>
              {this.state.image_object && (
                <Button
```