**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА СИСТЕМНОГО ПРОГРАМУВАННЯ І
СПЕЦІАЛІЗОВАНИХ КОМП'ЮТЕРНИХ СИСТЕМ

«На правах рукопису»
УДК 004.852

«До захисту допущено»

Завідувач кафедри СПСКС

_____ В.П.Тарасенко_
(підпис) (ініціали, прізвище)

"___"_____2018р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 123 Комп'ютерна інженерія
Комп'ютерні системи та компоненти

на тему: « СИСТЕМА НЕЙРОМЕРЕЖЕВОГО РОЗПІЗНАВАННЯ ДЕТАЛЕЙ ОДЯГУ НА ЗОБРАЖЕННІ»

Виконав: студент ІІ курсу, групи КВ-71мп

Гудіков Владислав Олегович _____
(підпис)

Науковий керівник к.т.н., доцент Сапсай Т.Г. _____
(підпис)

Рецензент_____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2018 року

Національний технічний університет України
“Київський політехнічний інститут
імені Ігоря Сікорського”

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп‘ютерних систем
Рівень вищої освіти – другий (магістерський)
Спеціальність 123 Комп'ютерна інженерія
Комп’ютерні системи та компоненти

ЗАТВЕРДЖУЮ

Завідувач кафедри СПСКС

_____В.П.Тарасенко

“___” _____ 2018 р.

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

Гудікову Владиславу Олеговичу

1. Тема дисертації: «Система нейромережевого розпізнавання деталей одягу на зображенні»,

науковий керівник дисертації: доцент кафедри СПіСКС, к.т.н., доцент Сапсай Т.Г.

затверджені наказом по університету від «30» жовтня 2018 р. №4030-с

2. Термін подання студентом дисертації: «7» грудня 2018 р.

3. Об’єкт дослідження: система організації нейромережевого розпізнавання на зображенні об‘єкта на основі створення моделі нейронної мережі.

4. Предмет дослідження: методи навчання та організації моделей загорткових нейронних мереж.

5. Перелік завдань, які потрібно розробити:

- проведення систематизації існуючих методів вирішення задач комп‘ютерного бачення;
- розглянути і проаналізувати архітектури та принципи організації нейронних мереж;
- дослідити методи навчання нейронних мереж, а також способи оптимізації параметрів згорткових мереж;

- розробити систему нейромемережевого розпізнавання деталей одягу на зображенні;
- експериментальне дослідження характеристик розробленої системи.

6. Орієнтовний перелік ілюстративного матеріалу:
   - структура базової згорткової нейронної мережі;
   - похибки класифікації зображень звичайної нейронної мережі та мережі з залишковим навчанням;
   - похибки класифікації зображень;
   - структура розробленої системи розпізнавання об‘єктів;
   - результати експеременталних досліджень.
7. Орієнтовний перелік публікацій:
   - «Використання залишкового навчання у згорткових нейронних мережах», XI наукова конференція магістрантів та аспірантів “Прикладна математика та комп’ютинг” ПМК-2018-2;
   - «Оптимізація аналізу даних з використанням згорткових нейронних мереж та залишкового навчання», V Міжнародна науково-технічна Internet-конференція «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами»
8. Дата видачі завдання: “4” вересня 2017 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Приміт ка
1.	Вивчення літератури за тематикою проекту	14.12.2017	
2.	Аналіз існуючих рішень	17.01.2018	
3.	Підготовка матеріалів першого розділу магістерської дисертації	17.03.2018	
4.	Підготовка матеріалів другого розділу магістерської дисертації	23.05.2018	
5.	Підготовка матеріалів третього розділу магістерської дисертації	16.10.2018	
6.	Підготовка графічної частини дипломного проекту	28.10.2018	
7.	Оформлення документації дипломного проекту	03.11.2018	
8.	Попередній розгляд магістерської дисертації на засіданні кафедри	26.11.2018	

Студент _____ Гудіков В.О.

Науковий керівник дисертації _____ Сапсай Т.Г.

РЕФЕРАТ

Актуальність теми. Технології у 21 столітті стрімко розвиваються. Це зумовлено потужністю обчислювальної техніки, яка використовується у наші дні. Теорія машинного навчання почала свій розвиток ще у 60-ті роки минулого сторіччя, проте за відсутності відповідної обчислювальної техніки подальший розвиток цієї галузі був неможливий, хоча більшість принципів для розробки структур сучасних штучних інтелекті запозичені саме з тих часів. Так протягом останніх 15 років галузь машинного навчання є однією із провідних у комп'ютерних науках.. Використання нейромережевого підходу при вирішенні задач з комп'ютерного бачення є унікальним. Застосовуючи основні принципи побудови згорткової нейронної мережі, що є фундаментом для вирішення задач комп'ютерного бачення, можна використовувати будь-які необхідні тренувальні множини для розпізнавання об'єктів або рухів, відновлення сцен та зображень.

Об'єктом дослідження система організації нейромережевого розпізнавання деталей на зображенні об‘єкта на основі створення моделі нейронної мережі.

Предметом дослідження є методи навчання нейронних мереж та способи реалізації архітектор згорткових нейронних мереж.

Методи дослідження — методи машинного навчання з попереднім використанням тренувальної множини.

Мета і задачі дослідження: створити систему нейромережевого розпізнавання деталей одягу на зображені об‘єкта на основі згорткової нейронної мережі з використанням залишкового блоку.

Для досягнення цієї мети вирішити наступні задачі: розглянути і проаналізувати архітектури та принципи організації нейронних мереж та методів вирішення задач комп‘ютерного бачення, дослідити методи навчання нейронних мереж, а також способи оптимізації параметрів

згорткових мереж, запропонувати структуру системи розпізнавання об‘єктів, яка забезпечить зменшення похибки при класифікації деталей одягу. Провести експериментальне дослідження характеристик системи.

Наукова новизна одержаних результатів полягає в наступному:

1. Обґрунтовано використання методів залишкового навчання разом із основними принципами роботи фільтру Калмана та прихованими марківськими процесами для запобігання проблеми деградації нейронної загорткової мережі, що є результатом згасання градієнта.
2. Запропоновано нову множину гіперпараметрів згорткової нейронної мережі: кількість шарів нейронної мережі, значення коефіцієнта темпу навчання, що визначає крок антиградієнта на кожній ітерації, використання стохастичного градієнткого спуску та функції активації ReLu.
3. На цій основі запропонована структура системи нейромережевого розпізнавання деталей на зображенні об‘єкта з використанням загорткової нейронної мережі з залишковим навчання.

Практична цінність одержаних результатів полягає в тому, що розроблена система розпізнавання деталей одягу на основі згорткової нейронної мережа, яка може використовуватись не тільки для класифікації атрибутів одягу, а також для інших задач класифікацій об’єктів на зображенні.

Апробація роботи. Основні положення і результати роботи представлені та обговорені на:

- XI конференція молодих вчених ПМК-2018-2 «Прикладна математика та комп’ютинг»;

- V Міжнародна науково-технічна Internet-конференція «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами»

Публікації. За темою досліджень опубліковано 2 дві наукові праці - тези доповідей на конференціях.

Структура та обсяг роботи. Магістерська дисертація складається з вступу, трьох розділів, висновків та додатків.

У *вступі* зроблено оцінку сучасного стану розвитку машинного навчання, визначено основні проблеми при навчанні нейронної мережі, обґрунтовано актуальність напряму, сформульовано мету і задачі досліджень

У *першому розділі* наведено теоретичні відомості щодо заданої тематики, визначено основні методи вирішення задач комп'ютерного бачення, методи навчання нейронних мереж.

У *другому розділі* наведено види нейронних мереж та основні елементи, з яких складаються такі мережі, та проаналізовані параметри моделей згорткових нейронних мереж, що використовуються при навчанні мереж.

У *третьому розділі* розроблена система розпізнавання деталей одягу на основі навчання первинної згорткової нейронної мережі з залишковим блоком, що дозволяє зменшити похибку при класифікації об'єктів.

У *висновках* представлені результати проведеної роботи

Ключові слова: згорткова нейронна мережа, залишкове навчання, темп навчання, розпізнавання об'єкта, навчання мережі, комп'ютерне бачення.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ, ПОЗНАЧЕНЬ, ТЕРМІНІВ

ANN (artificial neural network) – штучна нейронна мережа

AlexNet – вид згорткової нейронної мережі

Back-propagation – метод зворотного поширення похибки

Batch Normalization – метод пакетної нормалізації

CNN (convolutional neural network) – згорткова нейронна мережа

Caffe – програмна бібліотека для глибинного навчання

Continuous Bag Of Words – метод природного розпізнавання мови

Dropout – метод випадкового видалення ваг

Fast R-CNN (Fast Region-based Convolutional Network) – вид згорткової нейронної мережі

ImageNet – база даних зображень для навчання нейронної мережі

Learning Rate – темп навчання

Long Short-Term Memory – блок рекурентної нейронної нейронної мережі

Max Pooling – агрегування даних

ModelZoo – форум про машинне навчання

OpenCV – програмна бібліотека для візуалізації даних

PCA (principal component analysis) - метод головних компонент

RPN (region proposal network) – мережа попонування регіонів

SVM (support vector machine) - метод опорних векторів

Stochastic Gradient Descent – стохастичний градієнтний спуск

Transfer Learning – метод передачі навчання

ВСТУП

Інформаційні технології в двадцять першому столітті розвиваються надзвичайно стрімко. Це зумовлено новими теоретичними та практичними досягненнями в галузі обчислювальної техніки. Так ще у 60-ті роки минулого століття були сформульовані основні положення теорії машинного навчання, проте за відсутності відповідних можливостей технічної бази подальша практична реалізація цієї теорії буда дещо сповільнена, хоча більшість принципів для розробки сучасних систем штучного інтелекту запозичено саме з тих часів. Протягом останніх 15 років машинне навчання стало одним із провідних напрямів розвитку комп'ютерних наук, адже положення цієї теорії лягли в основу найперспективніших напрямів розвитку IT-технологій та сучасної кібернетики.

З часом машинне навчання розподілили на три провідні напрями: статистичне навчання, обробка природної мови та комп'ютерне бачення. Для вирішення задач, пов'язаних з розпізнаванням об'єктів та зображень існують різні способи, але найкращі результати отримані при використанні загорткових нейронних мереж глибинного навчання.

Застосовуючи основні принципи побудови згорткової нейронної мережі, як основу для реалізації задач комп'ютерного бачення, можна використовувати будь-які тренувальні множини для розпізнавання об'єктів або рухів, відновлення сцен та зображень. Основними задачами при розробці нейронної мережі є нормалізація даних та підбір параметрів і гіперпараметрів мережі. Ефективність навчання нейронної мережі визначається підходом до архітектурних рішень, методів запобігання від перенавчання та загасання градієнта зворотного поширення похибки.

В даній роботі запропоновано згорткову нейронну мережу розпізнавання деталей одягу на зображенні, з використанням нових підходів комп'ютерного бачення, таких як архітектура Inception та залишкове навчання. Розроблена нейронна мережа при комбінації

запропонованих підходів дозволяє оптимізувати вибір ваг при навчанні, запобігти згасанню градієнта для нейронних мереж з великою кількість прихованих шарів, зменшити час обчислення за рахунок оптимізаціє розмірності карт ознак. Створена нейронна мережа при комбінації двох запропонованих підходів: базової глибинної нейронної мережі та блоку залишкового навчання. Дана мережа дає кращі показники у порівнянні з нейронними мережами, що використовують їх окремо, а також класичними згортковими нейронними мережами. В роботі велику увагу приділено стохастичному градієнтному спуску, що дозволило ефективно визначити глобальний мінімум функції похибки.

1. МЕТОДИ ВИРІШЕННЯ ЗАДАЧ КОМП’ЮТЕРНОГО БАЧЕННЯ

1.1. Нейронна мережа

Штучна нейронна мережа (ANN) – це парадигма обробки інформації, яка запозичила ідею біологічної нервової системи. Ключовим елементом цієї парадигми є нова структура системи обробки інформації. Вона складається з великої кількості взаємопов'язаних процесорних елементів (нейронів), які працюють в унісон для вирішення конкретних проблем. ANN, як і люди, навчаються на прикладі. Через процес навчання вони налаштовуються на вирішення певної задачі, наприклад, розпізнавання образів або класифікація даних. Навчання в біологічних системах передбачає коригування синаптичних зв'язків, які існують між нейронами. Це також стосується і ANN.

Симуляція нейронної мережі здається нещодавнім досягненням, але ця область комп’ютерних наук бере свій початок ще до появи перших комп'ютерів. Багато прогресивних досягнень у цій сфері завдячують використанню примітивних комп'ютерних емуляцій. Так, перший штучний нейрон був створений в 1943 році нейрофізіологом Уорреном МакКуллохом та математиком Уолтером Пітцем, але технології, доступні для того часу, не дозволяли їм зробити занадто багато.

Після тривалого періоду інтересу до цієї сфери, ця область зазнала занепаду. Це відбулося після публікації Марвіном Лі Мінським та Сеймуром Папертом в 1969 році книги «Персептрон», в якій вони підсумували загальне розчарування серед дослідників щодо нейронних мереж, і яку більшість вчених того часу прийняли без подальшого аналізу. В цей період, коли фінансування та професійна підтримка були мінімальними, важливі досягнення були зроблені відносно невеликою кількістю дослідників, серед них, наприклад, Дж. Хопфилд. Ці піонери зуміли створити переконливу технологію, яка подолала обмеження,

визначені Мінським і Папертом, але загалом дослідження нейронних мереж сповільнилися до того часу, коли комп'ютерні технології досягли великих обчислювальних потужностей. Сьогодні можемо спостерігати відродження інтересу до ANN, що тягне за собою і відповідне збільшення фінансування цієї сфери.

Нейронні мережі мають чудову здатність аналізувати складні або неточні дані та можуть використовуватися для вилучення патернів та виявлення тенденцій, які занадто складні, щоб їх помітили люди або нескладні комп'ютерні алгоритми. Підготовлену нейронну мережу можна розглядати як «експерта» у категорії інформації, для аналізу якої вона була створена. Цей «експерт» може враховувати нові цікаві ситуації в сфері, до якої він був підготовлений, та може бути використаний для надання прогнозів та відповідей на запитання типу «що, якщо».

Інші переваги нейронних мереж:

1. адаптивне навчання: вміння навчитися виконувати завдання на основі первісних даних, наданих для початкового досвіду.

2. самоорганізація: ANN може створити власну структуру інформації, яку вона отримує під час навчання.

3. операції в режимі реального часу: розрахунки ANN можуть виконуватися паралельно одна з одною і обробляються спеціальними апаратними пристроями.

4. толерантність до помилок через кодування інформації та складання резервних копій. Звісно, часткове знищення мережі призводить до певної деградації продуктивності, однак деякі можливості мережі можуть зберігатися навіть при значному пошкодженні мережі[1].

Нейронні мережі використовують принципово інший підхід до вирішення завдань, ніж звичайні комп'ютери. У звичайних комп'ютерах використовується алгоритмічний підхід, тобто комп'ютер виконує набір поетапних інструкцій для вирішення певної проблеми. Відомо, що комп'ютер не може розв’язати завдання, якщо не будуть визначені

конкретні кроки, які він повинен виконати. Це зводить коло можливостей звичайних комп'ютерів до вирішення задач, які ми вже розуміємо і вміємо розв’язувати, але вони були б набагато корисніші, якщо б могли робити те, що ми точно не вміємо.

Принцип обробки інформації у нейронній мережі є наближеним до алгоритму роботи мозку людини. Мережа складається з великої кількості взаємопов'язаних елементів (нейронів), що працюють паралельно над вирішенням конкретного завдання. Так само, як і мозок людини, нейронні мережі не можуть бути запрограмовані для вирішення всіх можливих проблем, але вони вміють адаптивно навчатися. Це навчання відбувається на певних прикладах, однак відбирати ці приклади слід дуже обережно, оскільки помилка в цій сфері може призвести до погіршення роботи всієї мережі. Недоліком нейронної мережі є також і те, що вона самостійно «приймає рішення» щодо того чи іншого завдання, а отже результати такого рішення є не завжди передбачуваними. В цьому нейронна мережа принципово відрізняється від звичайних комп'ютерів, які використовують когнітивний підхід до вирішення проблем – комп’ютер діє в межах невеликих однозначних поетапних інструкцій, перетворених на мовну програму високого рівня, а потім переведених на машинний код, зрозумілий комп'ютеру. Ці машини є цілком передбачуваними, а потенційні помилки можуть бути пов'язані лише з несправністю програмного чи апаратного забезпечення.

Зазначимо, що нейромережі та звичайні алгоритмічні комп'ютери не конкурують, але доповнюють одне одного. Є завдання, які більш підходять для алгоритмічного підходу, наприклад, арифметичні операції, а є задачі, в яких більш ефективними є нейронні мережі, наприклад, розпізнавання образів. А велика кількість завдань вимагає використання комбінації цих двох підходів (як правило, алгоритмічний комп'ютер виконує нагляд за нейронною мережею) для максимально ефективної роботи.

Багато чого ще невідомо щодо того, як мозок готує інформацію для обробки, а для пояснення цього існує багато теорій. У людському мозку типовий нейрон збирає сигнали від інших через безліч тонких структур – дендритів. Нейрон посилає спайки електричної активності через аксон, який розгалужується на тисячі гілок, в кінці кожної з яких знаходиться структура, що називається синапсом. Синапс реагує на електричну активність аксона, внаслідок чого відбувається гальмування або збудження активності у зв'язаних ним нейронах. Коли нейрон отримує збуджувальний вхід, він посилає відповідний імпульс електричної активності на його аксон. Навчання відбувається шляхом зміни ефективності синапсів, так що вплив одного нейрона на інший змінюється. На рис. 1.1 зображено елементи нейрону.

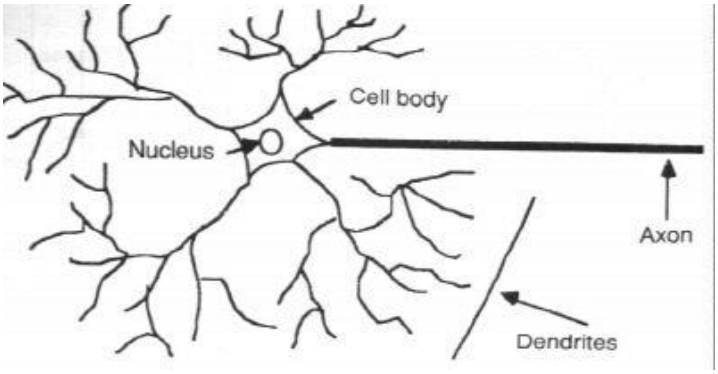

Рисунок 1.1 – Елементи нейрону

Для створення нейронних мереж, спочатку необхідно вивести суттєві ознаки нейронів та їх зв'язку між собою. Наступним кроком – є імітування цих ознак за допомогою комп'ютерів. Однак, оскільки наше знання нейронів є неповним, а обчислювальні потужності – обмеженими, то створені моделі є певними ідеалізаціями реальних мереж нейронів.

Штучний нейрон – це пристрій з багатьма входами та одним виходом. Нейрон має два режими роботи: режим тренувань і режим використання. У тренувальному режимі нейрон навчається відповідати так (чи ні) для певних шаблонів вводу. У режимі використання, коли у

вхідному шаблоні викладено вихідний шаблон, пов'язаний з ним висновок стає поточним виходом. На рис 1.2 зображена логічна модель нейрону.

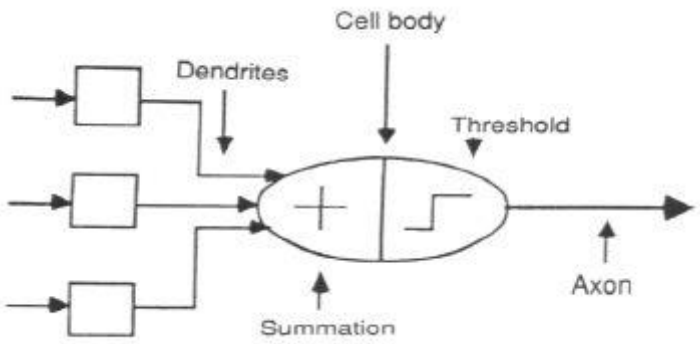

Рисунок 1.2 – Модель нейрона у машинному навчанні

Важливою концепцією, яка багато в чому пояснює високі адаптивні властивості нейронних мереж, є правило стрільби, яке для нейронних мереж можна зрозуміти наступним чином:

«Візьміть збірку навчальних патернів для вузла, деякі з яких змушують нейрон спрацювати (перший набір моделей), та інші, які не дозволяють йому це робити (другий набір). У випадку використання шаблонів, що не знаходяться в тренувальній множині, нейрон «спрацьовує», якщо шаблони мають більше елементів, спільних з «найближчим» шаблоном у першому наборі, ніж з «найближчим» шаблоном у другому наборі. Якщо схожих ознак однакова кількість, то шаблон залишається в невизначеному стані». Правило стрільби визначає, чи буде нейрон використовуватись для певного шаблону введення. Це стосується будь-яких шаблонів введення, а не тільки тих, на яких був натренований вузол. Прикладом використання простого правила стрільби для нейронних мереж можна назвати техніку Хеммінга.

Важливою сферою застосування нейронних мереж є розпізнавання образів. Воно може бути реалізовано за допомогою нейронної мережі, яка була відповідним чином натренована (рис. 1.3). Під час тренувань мережа навчається поєднувати результати з шаблонами введення. Коли мережа використовується, вона визначає вхідний шаблон і намагається вивести пов'язаний вихідний шаблон. У цьому випадку мережа формує висновок,

який відповідає навчальному вхідному шаблону, що якнайменше відрізняється від заданого шаблону.

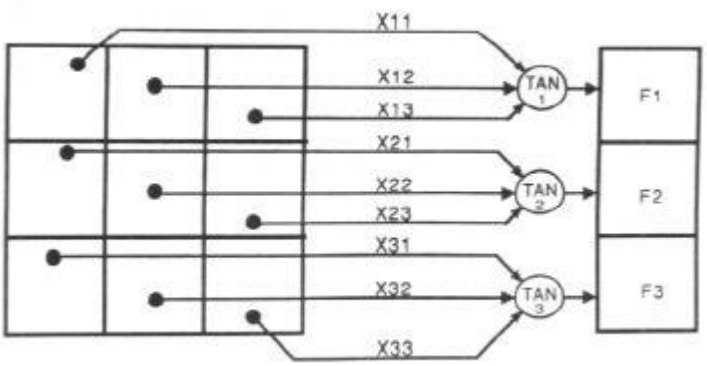

Рисунок 1.3 – Приклад нейронної мережі

Модель нейрону, що розглядалася вище, не робить нічого нового у порівнянні із звичайними комп'ютерами. Більш вдосконалена модель нейрона (рис. 1.4) – це модель Мак Куллоха та Пітца (MCP). Відмінність даної моделі від попередньої полягає в тому, що входи «зважуються», тобто ефект, який кожен вхід має при прийнятті рішення, залежить від ваги конкретного входу. Ці зважені входи потім об'єднуються, і якщо вони перевищують попередньо встановлене порогове значення, нейрон активується. У будь-якому іншому випадку нейрон не спрацьовує.

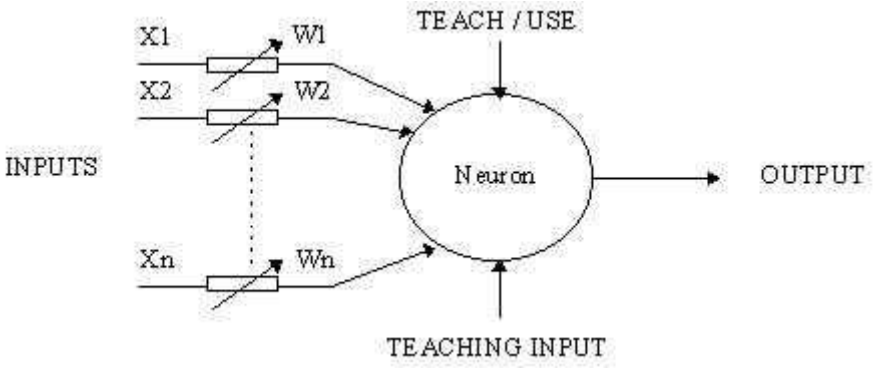

Рисунок 1.4 – MCP нейрон

Додавання ваг вхідних і порогових значень робить цей нейрон дуже гнучким і потужним. Нейрон MCP має здатність адаптуватися до певної ситуації, змінюючи ваги та/або порогове значення. Існують різні алгоритми, які змушують нейрон «адаптуватися». Найбільш часто використовуються правило дельти та зворотне поширення помилки. Перше

використовується в мережах прямого зв'язку, а останнє – в мережах зворотного зв'язку.

Мережі прямого зв'язку (рисунок 1.5) дозволяють сигналам проходити тільки в одну сторону – від входу до виходу. В таких мережах немає зворотного зв'язку (циклів), тобто вихід будь-якого шару не впливає на той самий шар. Мережі прямого поширення, як правило, є мережами, які пов'язують входи з виходами. Вони широко використовуються у розпізнаванні образів. Цей тип організації також називається «знизу вгору» або «зверху вниз».

Зворотні мережі можуть мати сигнали, що проходять в обох напрямках, шляхом введення петлі в мережі. Мережі зворотного зв'язку є дуже потужними і можуть бути надзвичайно складними. Мережі зворотного зв'язку є динамічними. Їхній «стан» постійно змінюється до досягнення точки рівноваги, де вони і залишаються, доки не відбудеться введення змін, що викликає необхідність пошуку нової точки рівноваги. Архітектури зворотного зв'язку також називають «інтерактивними» або «періодичними», хоча останній термін часто використовується для позначення зворотного зв'язку в одношарових організаціях.

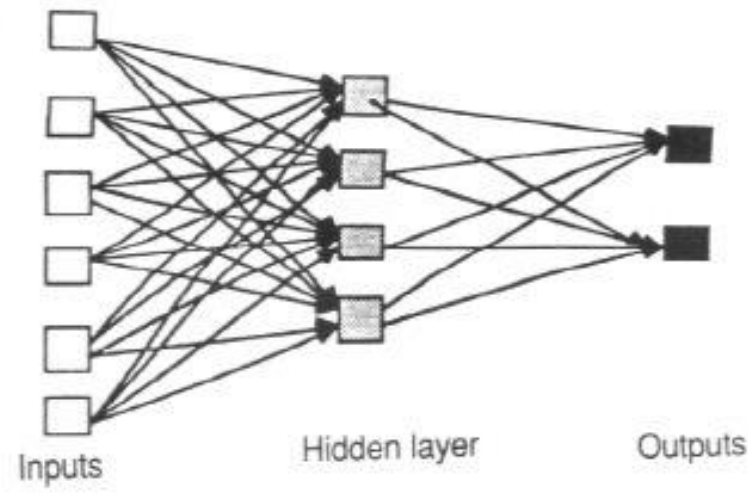

Рисунок 1.5 – Модель повноз’вязаної нейронної мережі

Найпоширенішим типом штучної нейронної мережі є мережа, що складається з трьох груп або шарів одиниць, де вхідний шар є підключеним до прихованого шару, який, в свою чергу, підключається до вихідного шару (див. рис. 1.5).

Основними ознаками такої повнозв’язної нейронної мережі є:

• Призначення вхідного шару полягає у отриманні вхідної інформації, яка надсилається в мережу.

• Діяльність кожного прихованого шару визначається діями значень вхідного шару та ваг на зв'язках між входом і прихованими шарами.

• Поведінка вихідного шару залежить від активності прихованих шарів та ваг між прихованими та вихідними шарами.

Цей тип мережі цікавий тим, що приховані шари можуть вільно створювати власні уявлення про вхід. Вага, що утворюється внаслідок скалярного добутку вхідних та прихованих шарів, визначає, коли вона має активуватись, і шляхом зміни цих ваг, прихована вага може обрати ознаку, яку вона представляє.

Ми також розрізняємо одношарові та багаторівневі архітектури. Одношарова організація, в якій всі підрозділи пов'язані один з одним, являє собою найбільш загальний випадок і має більше потенційної обчислювальної потужності, ніж ієрархічно структуровані багатошарові організації. У багатошарових мережах одиниці часто нумеруються за шаром, замість того, щоб слідувати глобальній нумерації.

Як ми вже зазначали, у 60-ті роки вийшла найвпливовіша робота Марвіна Лі Мінського та Сеймура Паперта на тему нейронних мереж під заголовком «Персептрон» – термін, який ввів Френк Розенблатт. Персептрон (рисунок 1.6) є моделлю MCP (нейрон з ваговими входами) з деякою додатковою фіксованою попередньою обробкою. Елементи, позначені як A1, A2, Aj, Ap, називаються одиницями асоціації і їх завдання полягає в тому, щоб видобути специфічні, локалізовані ознаки з вхідних зображень. Персептрони імітують основну ідею візуальної системи тварин.

Вони здебільшого використовувались при розпізнаванні образів, хоча їх можливості набагато більші.

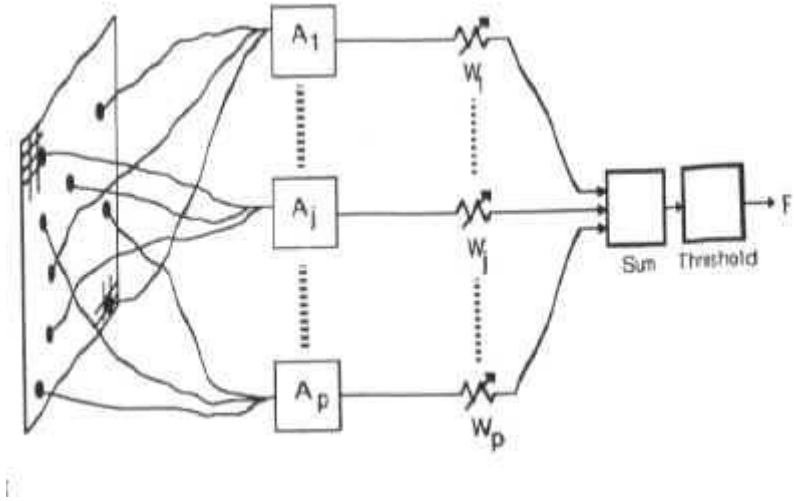

Рисунок 1.6 – Модель персептрону

У своїй роботі Мінський та Паперт описали обмеження одношарового персептрону. Так, було математично доведено, що одношарові персептони не могли виконувати деякі основні операції розпізнавання образів, такі як визначення співвідношення форм. Як вже зазначалося, вплив книги був величезним, і після її публікації багато дослідників нейронних мереж втратили свій інтерес до даної сфери.

1.2. Методи навчання нейронних мереж

В цьому підрозділі розглянуто основні методи навчання нейронних мереж, такі як: зворотне поширення помилки (Back-propagation); стохастичний градієнтний спуск (Stochastic Gradient Descent); адаптація швидкості навчання (Learning Rate Decay); випадкове видалення ваг (Dropout); дискретизація на основі вибірки (Max Pooling); пакетна нормалізація (Batch Normalization); довга короткотермінова пам'ять (Long Short-Term Memory); безперервний пакет словникової моделі (Continuous Bag Of Words) та передавання навчання (Transfer Learning).

- Першим з методів навчання нейронних мереж є зворотне поширення помилки (Back-propagation) – це метод для обчислення часткових похідних (або градієнтів) функції, яка має форму функціональної композиції (як у нейронних мережах). При вирішенні задач оптимізації, використовуючи градієнтні методи (градієнтний спуск є лише одним з них), обчислюється градієнт функції на кожній ітерації.

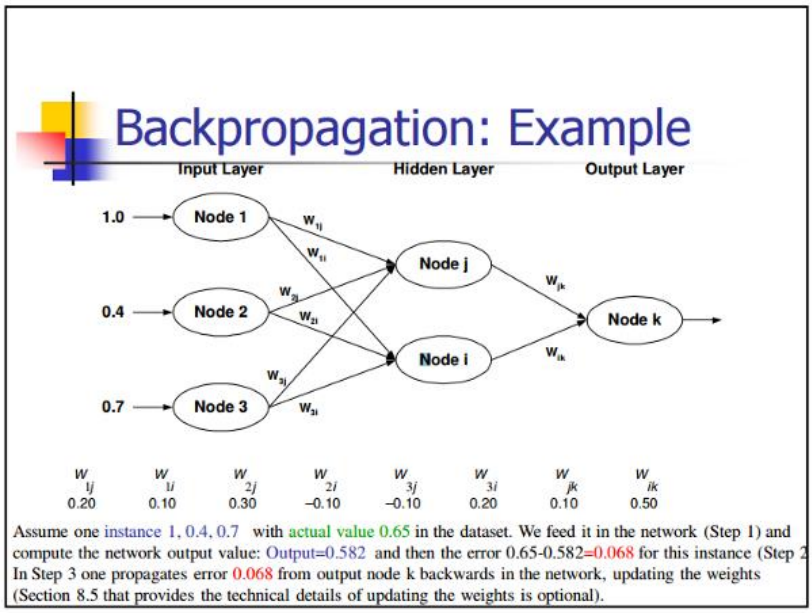

Рисунок 1.7 –Метод зворотного поширення помилки

Для нейронних мереж цільова функція має форму композиції. Є два загальні способи визначення функції: аналітична диференціація та приблизна диференціація. Аналітична диференціація – це обчислення похідних, використовуючи правило ланцюга (основне обчислення). Приблизна диференціація – провадиться з використанням кінцевої різниці. Оскільки кількість оцінок функцій складає - O(N), де N - кількість параметрів, то даний метод є дорогим та довгим за часом обчислення, порівняно з аналітичною диференціацією. Однак кінцева різниця зазвичай використовується для перевірки ваг, що були розраховані під час зворотного поширення помилки.

- Другий метод навчання нейронних мереж – стохастичний градієнтний спуск (Stochastic Gradient Descent).

Аби отримати базове розуміння стохастичного градієнтного спуску необхідно уявити річку, яка бере свій початок з вершини гори. Мета градієнтного спуску – досягнення найнижчої точки передгір'я.

Якщо структура гори є такою, що річка безперешкодно протікає до пункту призначення (тобто до найнижчої точки передгір'я), то це є ідеальним варіантом, який необхідно отримати. В машинному навчанні це означає, що необхідно знайти глобальний мінімум (або оптимальне рішення), починаючи від початкової точки (вершини пагорба). Однак існують варіанти, коли природа місцевості має кілька можливих шляхів, що можуть змусити річку «потрапити в пастку» та утворити застій. В умовах машинного навчання такі ями називають локальними мінімумами, які не є бажаними. Існує безліч способів уникнути таких мінімумів.

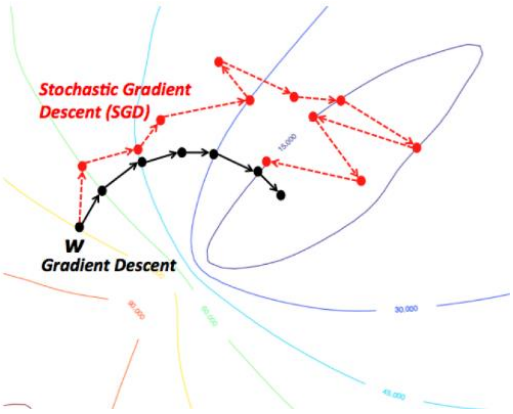

Рисунок 1.8 –Метод стохастичного градієнтного спуску

Залежно від характеру місцевості (або функціонування в умовах машинного навчання) градієнтний спуск схильний до закріплення в локальному мінімумі. Але, коли існує особливий вид гірського ландшафту, сформований як чаша (в рамках машинного навчання це називається опуклою функцією), алгоритм завжди гарантовано знаходить оптимальний шлях до глобального мінімуму. Ці види спеціальних площин (опуклих функцій) завжди є найкращим випадком при оптимізації в машинному

навчанні. Крім того, в залежності від того, де на вершині гори існує початок спуску (тобто початкові значення функції), можливо досягти глобального мінімуму іншим шляхом. Так само, залежно від швидкості руху річки (наприклад, швидкість навчання або розмір кроку для алгоритму градієнтного спуску), можливо досягти кінцевого пункту призначення іншим способом. Завдяки адаптації цих двох критеріїв, знаходиться декілька шляхів до глобального мінумуму.

- Наступним методом навчання нейронних мереж, що ми розглянемо, є – метод адаптації швидкості навчання (Learning Rate Decay).

Адаптація швидкості навчання при використанні стохастичного градієнтного спуску може підвищити продуктивність і скоротити час навчання. Іноді це називають адаптивним темпом навчання.

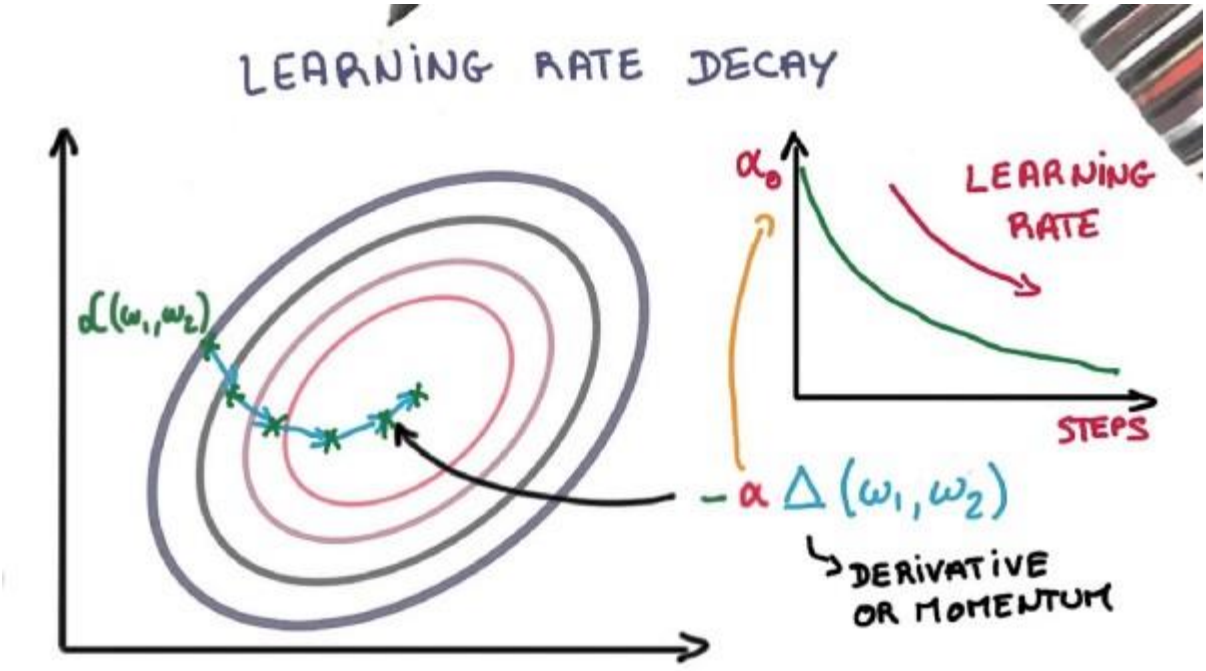

Рисунок 1.9 – Метод адаптації швидкості навчання

Найпростішою адаптацією швидкості навчання нейронних мереж під час тренувань – це застосування методів, які зменшують час навчання. Для цього необхідно додати коефіцієнт, що буде впливати на значення кроку градієнта. Якщо даний коефіцієнт буде достатньо великим на початку

навчання, швидкість пошуку глобального мінімуму збільшиться, що у свою чергу вплине на швидкість вивчення правильних ваг на ранній стадії та більш точного налаштування.

Найбільш популярними та легкими у використанні є два наступних метода адаптації швидкості навчання:

- зміна швидкості навчання шляхом зміни кількості ітерацій;
- зміна швидкості навчання через зменшення кроку градієнту.

- Важливим для розгляду є метод випадкового видалення ваг (Dropout).

Глибинні нейронні мережі з великою кількістю параметрів є дуже потужними системами машинного навчання. Однак серйозною проблемою в таких мережах є перенавчання. Це пов’язано, зокрема, із повільними темпами навчання самої системи, оскільки, аби визначити, що нейронна мережа правильно вирішує свою задачу, необхідно пройти етап тестування. Dropout є методом вирішення цієї проблеми.

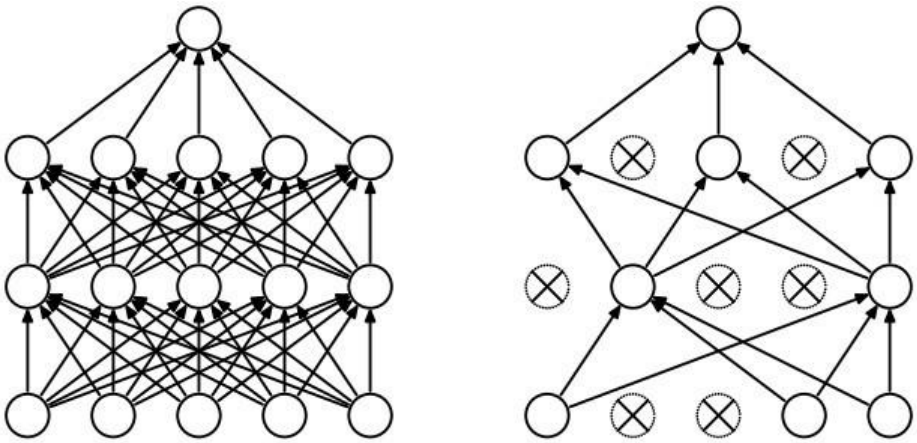

Рисунок 1.10 – Метод випадкового видалення ваг

Ключова ідея цього методу полягає в тому, щоб під час тренування випадково видалити вхідне значення (та відповідні зв'язки) з нейронної мережі (див. рисунок 1.10). Це запобігає надмірному навчанню, під час якого вилучаються зразки з експоненціального числа різних «розріджених» мереж. Проводячи тестування, легко наблизити ефект усереднення

прогнозів усіх цих зменшених мереж, шляхом простого використання однієї нерозвиненої мережи, яка має меншу кількість ваг. Випадкове видалення ваг значно зменшує перенавчання та дає значні покращення результатів, порівняно з іншими методами регуляризації. Метод Dropout дозволяє покращувати ефективність мереж, які використовуються у сфері комп'ютерного бачення, розпізнавання мовлення, класифікації документів та обчислювальної біології.

- Розглянемо метод навчання нейронних мереж Max Pooling. Агрегування - це метод дискретизації на основі вибірки. При реалізації методу береться зразок вхідного представлення (матриця зображень, прихованого шару тощо), та зменшується її розмірність, що дозволяє робити припущення щодо функцій, які містяться в субрегіонах.

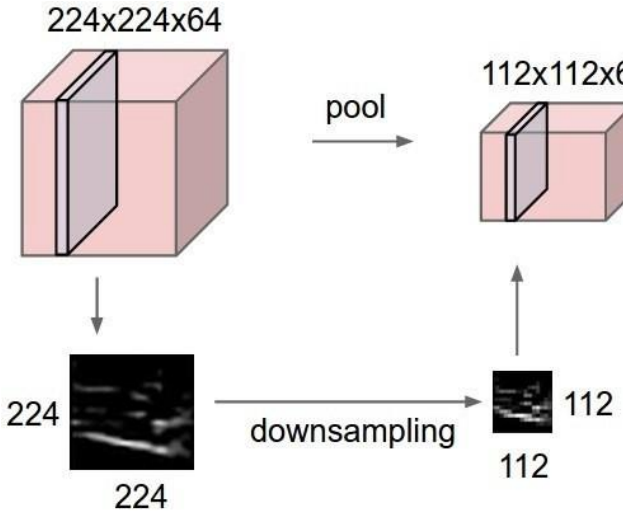

Рисунок 1.11 – Приклад застосування шару агрегування

Це зроблено, для подолання проблеми перенавчання, шляхом надання абстрактної форми об'єкту, що розглядається. Крім того, це зменшує обчислювальні витрати, через зменшення кількості параметрів для навчання. Агрегування здійснюється шляхом застосування максимального фільтра до складових субрегіонів початкової матриці.

- Цікавим для розгляду є такий метод навчання нейронних мереж, як метод пакетної нормалізації (Batch Normalization).

Природно, що нейронні мережі, включаючи глибинні мережі, вимагають ретельного налаштування початкових ваг та параметрів навчання. Пакетна нормалізація допомагає трохи спростити це завдання.

Основні проблеми ваг:

- незалежно від того, обиралися початкові ваги випадково чи емпіричним шляхом, вони знаходяться далеко від тих значень, що необхідно отримати. Впродовж початкових ітерацій, відбувається багато змін значень ваг для ознак, які необхідно активувати;
- глибинна нейронна мережа сама по собі є неправильною, оскільки, навіть невелике збурення в початкових шарах, призводить до значних змін ваг у шарах, що знаходяться глибше[2].

Під час зворотного розповсюдження похибки вищевказані проблеми викликають відхилення від градієнтів, тобто градієнти, щоб згенерувати правильні результати, повинні компенсувати ці відхилення. Це призводить до необхідності додаткових ітерацій для правильного навчання.

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Рисунок 1.12 – Метод пакетної нормалізації

Пакетна нормалізація регулює цей градієнт для уникнення відхилення. За допомогою пакетної нормалізації, активація непотрібних ваг зменшується, що призводить до прискорення навчання.

- Розглянемо також метод довгої короткотермінової пам'яті (Long Short-Term Memory).

Мережа *Long Short-Term Memory (*LSTM) має наступні три аспекти, які відрізняють його від звичайного нейрона в рекурентній нейронній мережі:

1) мережа забезпечує можливість контролю вводу вхідного значення нейрона;
2) мережа вирішує, коли необхідно запам’ятати, що було обчислено на попередньому кроці;
3) мережа забезпечує можливість формування вихідного сигналу на наступній ітерації.

Ефективність мережі LSTM полягає в тому, що вона забезпечує вирішення цих трьох аспектів на основі поточного вводу.

Вхідний сигнал x_t (див. рис. 1.13) у поточний час вирішує всі вищевказані 3 аспекти. Вхідний елемент приймає рішення щодо пункту 1. Елемент забування приймає рішення щодо пункту 2, а вихідний елемент приймає рішення за пунктом 3. Такий принцип було запозичено з принципу роботи нашого мозку, який може працювати з раптовими контекстними перемикачами на основі вхідних даних.

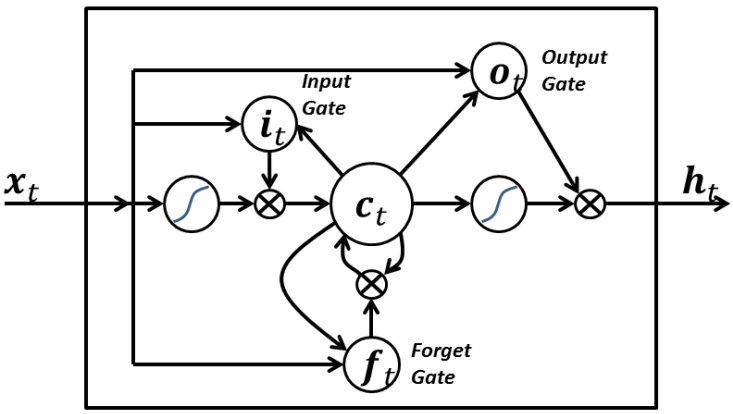

Рисунок 1.13 – Метод довгої короткотермінової пам'яті

- Надзвичайно важливим є також метод Skip-gram.

Skip-gram – це метод вивчення контекстного розташування слів. Мета цього методу полягає в тому, щоб через подібність векторів певних слів

визначити семантичну або синтаксичну схожість між відповідними словами. Основна ідея методу Skip-gram полягає в наступному: два терміни подібні, якщо вони мають подібний контекст (див. рисунок 1.14).

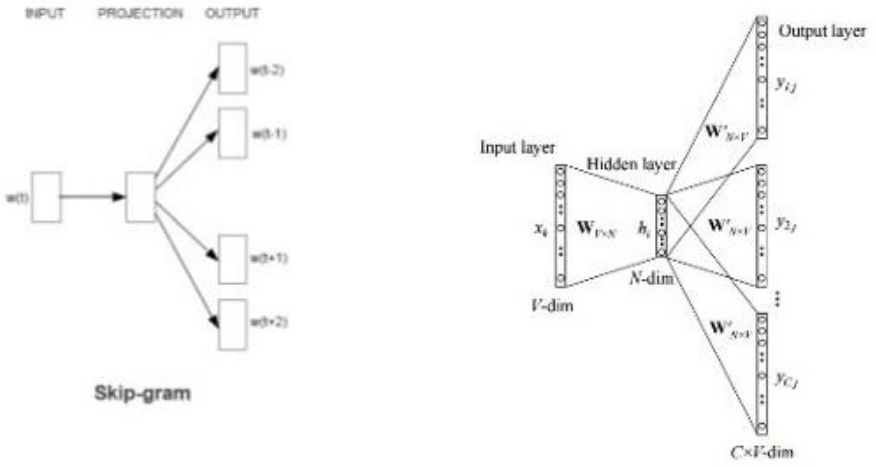

Рисунок 1.14 – Метод Skip-gram для вивчення контекстного розташування слів

Наприклад, візьмемо речення «кішки – ссавці». Якщо замість слова «кішки», використати термін «собаки» – речення однаково буде мати сенс. Отже, у цьому прикладі «собаки» та «кішки» можуть поділяти той самий контекст (тобто «ссавці»).

Виходячи з вищезгаданої гіпотези, можливо розглянути контекстне вікно (вікно, що містить K послідовних термінів). Потім необхідно пропустити одне з цих слів та спробувати навчити нейронну мережу спрогнозувати цей термін, за умови, що вона отримує всі терміни, крім власне того, що був пропущений,. Якщо два слова кілька разів поділяють подібні контексти у великому корпусі, то вектори цих термінів матимуть схожі значення.

- Близьким, але не тотожним до вищенаведеного методу є безперервний пакет словникової моделі (Continuous Bag Of Words).

Відповідно до цього методу, кожне слово в документі, у процесі обробки природної мови, представляється як вектор чисел. Це призводить

до того, що слова, які з'являються в подібному контексті, близькі один до одного за значеннями. Мета методу Continuous Bag Of Words полягає в тому, щоб мати змогу використовувати контекст, що оточує конкретне слово, для визначення конкретного слова, яке міститься в даному контексті.

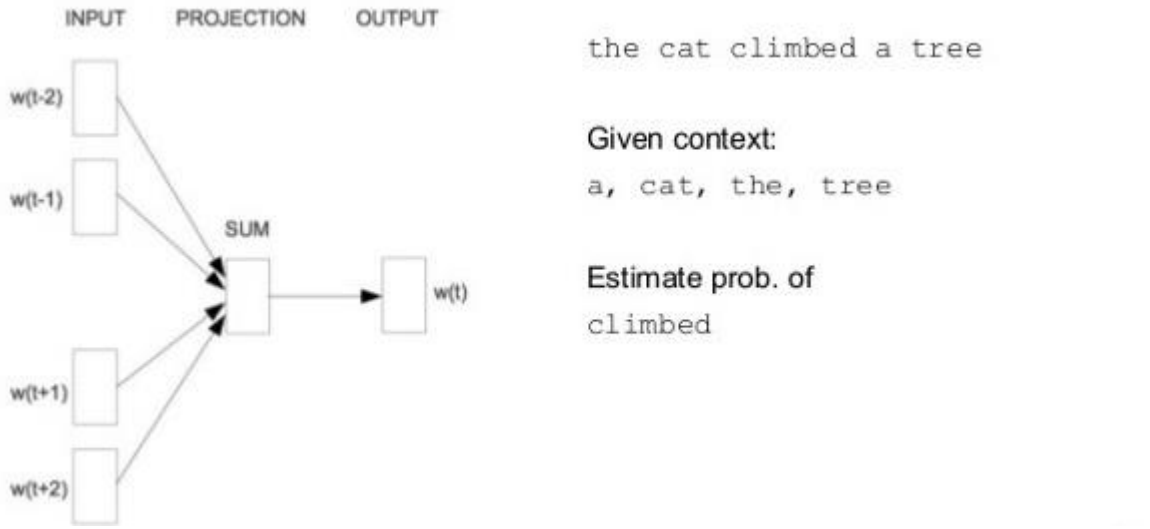

Рисунок 1.15 – Метод безперервного пакету словникової моделі

Під час обробки значної кількості речень у великому текстовому корпусі, кожен раз, коли зустрічається певне слово, до уваги також приймаються слова, що його оточують. Потім до нейронної мережі вводяться контекстні слова і прогнозується поява слова в центрі певного контексту. Серед тисяч таких контекстних слів, центральне слово – один екземпляр набору даних для нейронної мережі. Під час тренування нейронної мережі, закодований вивід прихованого шару являє собою представлення для певного слова. Тож під час тренування великої кількості речень, слова в подібному контексті отримують схожі вектори.

- Останнім методом навчання нейронних мереж, який ми розглянемо в межах цього дослідження, є метод передачі навчання (Transfer Learning). Розглянемо використання цього методу.

Нехай зображення проходить через згорткову нейронну мережу. Скажімо, існує зображення, на якому застосовується згортка, і, як наслідок, отримується комбінації пікселів у вигляді виходів. Припустимо, що вони є

краями. Наступна згортка дозволить на виході сформувати набір значень комбінацій країв (див. рисунок 1.16).

Останній шар нейронної мережі, як правило, є дуже специфічним. Наприклад, при використанні ImageNet, у мережах останній шар може шукати дітей, собак, літаки тощо. Поверх цього шару існує підмережа, яка шукає очі, вуха, рот або колеса.

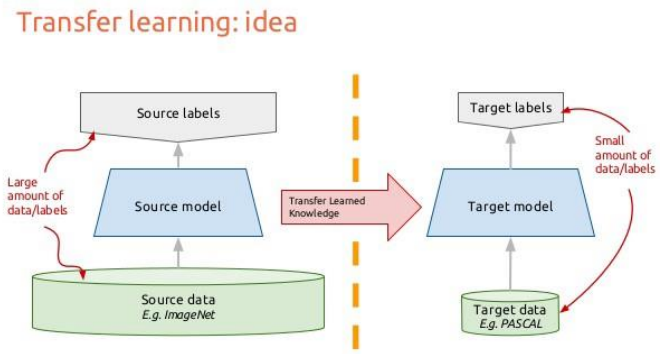

Рисунок 1.16 – Метод передачі навчання

Кожен шар в глибинній CNN поступово збирає вищі та більш високі уявлення про особливості. Останні пари шарів, як правило, спеціалізуються на визначені конкретних класів.

Метод передачі навчання полягає в тому, що у CNN, яка була натренована на одному наборі даних, видаляються ваги з останнього шару для того, щоб знову тренувати цей шар на іншому наборі даних. Це дозволяє використовувати одну й ту саму згорткову нейронну мережу для класифікації різних об’єктів. Існує декілька підходів передачі навчання в залежності від схожості навчальних даних та величини тренувальної множини. Внаслідок використання цього методу, час тренування значно зменшується, тому передача навчання є корисним інструментом, коли даних недостатньо, або тренування потребує надто багато ресурсів.

1.3. Типи задач області комп’ютерного бачення

Комп'ютерне бачення є одним з найбільш популярних напрямків в рамках глибинного навчання, його можна використовувати у багатьох

академічних дисциплінах, таких як комп'ютерні науки, математика (інформаційний пошук, машинне навчання), інжиніринг (робототехніка, мова, НЛП, обробка зображень), фізика (оптика), біологія (неврологія) і психологія (когнітивна наука). Оскільки комп'ютерне бачення являє собою розпізнавання візуальних середовищ та їх контекстів, багато вчених вважають, що даний напрям є основним у розвитку глобального штучного інтелекту.

Існує п’ять основних задач, які визначають комп’ютерне бачення як напрям машинного навчання:

- розпізнавання облич: Snapchat і Facebook використовують алгоритми виявлення облич для застосування фільтрів і розпізнавання облич на зображеннях;
- пошук зображень: Google Images здійснює пошук відповідних зображень. Алгоритми аналізують вміст в запиті зображення і забезпечують надання відповіді на запит на основі відповідності його змісту вхідному зображенню;
- ігри та управління: Microsoft Kinect – великий комерційний продукт в ігровій індустрії, який використовує стерео бачення;
- спостереження: камери відеоспостереження в громадських місцях використовуються для виявлення підозрілої поведінки.
- біометрія: використовується в біометричної ідентифікації шляхом ідентифікації відбитків пальців, аналізу обличчя тощо;
- розумні автомобілі: візуальний аналіз залишається основним джерелом інформації для виявлення дорожніх знаків, сигналів та інших обставин.

В якості основних перспективних напрямків розвитку комп’ютерного бачення, що сьогодні привертають найбільшу увагу дослідників, можна назвати наступні сфери: класифікація зображень, визначення об'єктів, відстеження об'єктів, семантична сегментація та сегментація екземпляру. Розглянемо ці напрями більш детально:

1.3.1. Класифікація зображень.

Класифікація зображень передбачає наявність певного вхідного набору зображень, який вже був попередньо класифікований. В межах комп’ютерного бачення необхідно врахувати ці категорії для нового набору тестових зображень, зробити певні прогнози щодо того, які зображення увійдуть у нові класифікації, та виміряти точність прогнозів, що були зроблені. Існують різні проблеми, пов'язані з цим завданням, включаючи зміну масштабу, освітлення та кута огляду, внутрішньокласові варіації зображення, деформацію зображення, оклюзії тощо.

Для вирішення завдань класифікації у сфері комп’ютерного бачення використовується наступний підхід. Замість того, щоб визначати кожну окрему категорію, комп'ютер забезпечується великою кількістю прикладів кожного класу зображення, а потім виконується алгоритм навчання, який при обробці цих прикладів дізнається про візуальний зовнішній вигляд кожного класу. Іншими словами, спочатку накопичується навчальна множина завантажених зображень, а потім комп'ютер обробляє ці дані. З огляду на цей факт, повна класифікація зображень може бути формалізована наступним чином:

- Вхідними даними є готова тренувальна множина, яка складається з N зображень, які відповідають певному значенню K.
- Навчальний набір використовується для підготовки класифікатора, щоб дізнатися, як кожен з класів виглядає.
- Проводиться оцінка якості класифікатора. Використовується тестувальна множина, і визначається, як часто класифікатор помилявся при визначенні класу об’єкту на зображенні.

Найбільш популярною архітектурою, що використовується для класифікації зображень, є згорткова нейронна мережа (CNN). CNN, як правило, починається з вхідного «сканеру», який не призначений для

розбору всіх даних тренувальної множини за один раз. Наприклад, для введення зображення 100 х 100 пікселів, не потрібно, щоб вхідний вектор мав розмірність 10000. Замість цього, створюється сканування вхідного шару скажімо 10 х 10, який репрезентує перші 10 х 10 пікселів зображення. Після проходження першої частини зображення, використовуються наступні 10 х 10 пікселів зображення, така техніка називається «ковзне вікно».

Ці вхідні дані, замість звичайних шарів, подаються за допомогою згорткових шарів. Кожен вузол стосується лише сусідніх вузлів. Такі згорткові шари, як правило, зменшуються, коли вони стають більш глибокими, оскільки вхідні дані легко ділити. Окрім цього, в згорткових нейронних мережах часто присутній агрегувальний шар. Агрегування є способом відфільтрувати деталі: загальноприйнята методика агрегації – взяти 2 х 2 пікселі та передати піксель з найбільшою величиною певного атрибута.

В даний час, більшість методів класифікації зображень проходять навчання в ImageNet – набір даних з приблизно 1,2 мільйона навчальних зображень з високою роздільною здатністю. Тестові зображення представляються без первинної анотації (сегментації або міток) і алгоритм повинен самостійно виробляти мітки, вказуючи, які об'єкти присутні у зображеннях. Деякі з найкращих існуючих методів комп'ютерного бачення були випробувані на цьому наборі даних за допомогою провідних груп з вивчення комп'ютерного бачення в Оксфорді, INRIA та XRCE. Як правило, системи комп'ютерного зору використовують складні багатоступеневі алгоритми, а їх ранні етапи, зазвичай, налаштовуються шляхом оптимізації кількох параметрів.

1.3.2. Визначення об'єктів.

Завдання визначення об'єктів у зображеннях, зазвичай, передбачає виведення обмежувальних ярликів та міток для окремих об'єктів, що

відрізняється від завдання класифікації/локалізації. Так, класифікація та локалізація застосовується до багатьох об'єктів, в той час, як визначення передбачає пошук простого домінуючого об'єкта. Існує лише два типи визначення об'єктів: об'єкт, що обмежений границями, та об'єкт, що границями не обмежений. Наприклад, при виявленні автомобілів потрібно виявити всі автомобілі в певному зображенні, що міститься в обмежувальних границях.

Якщо використовувати техніку ковзного вікна, подібно до того, як класифікується та локалізується зображення, потрібно застосувати CNN до різних частин зображення. Через те, що CNN класифікує кожну частину як об'єкт або фон, необхідність застосування CNN до величезної кількості розташувань і масштабів, що сильно збільшує час навчання нейронної мережі.

Для того, щоб впоратися з цією проблемою, дослідники нейронної мережі запропонували скористатися регіонами, замість того, щоб знаходити області зображень, які можуть містити об'єкти. Це значно збільшує швидкість навчання нейронної мережі. Перша нейронна мережа, яка була створена за таким принципом, називається R-CNN (регіонально-орієнтована згорткова нейронна мережа). У R-CNN спочатку виконується сканування вхідного зображення на предмет можливих об'єктів за допомогою алгоритму під назвою Selective Search, що генерує близько 2000 регіонів. Після цього, запускається CNN на кожному з цих регіонів. Нарешті, береться вихід кожного CNN і подається на вхід SVM, щоб класифікувати регіон і лінійну регресію, щоб зменшити границі об'єкта.

По суті, проблема визначення об’єкту перетворюється на проблему класифікації зображень. Тим не менш, є деякі складнощі, наприклад, тренування відбувається дуже повільно та потребує великої кількості пам’яті.

Безпосереднім нащадком R-CNN є Fast R-CNN, що покращує швидкість виявлення за допомогою двох нововведень:

1) вилучення ознак перед тим як пропонувати регіони, таким чином, запускається лише один CNN на всьому зображенні;

2) заміна SVM шаром softmax, що розширює існуючу нейронну мережу для прогнозування, а не створює нову модель.

Fast R-CNN продемонстрував значно кращі показники швидкості, оскільки він тренує лише одну CNN для всього зображення. Проте, для створення регіонів алгоритм вибіркового пошуку все ще займає багато часу.

Таким чином, Fast R-CNN, на даний момент, є канонічною моделлю для виявлення об'єктів на базі глибинного навчання. Він замінює алгоритм повільного вибіркового пошуку швидкою нейронною мережею. Це відбувається завдяки пропонуванню регіонів (RPN) для прогнозування пропозицій на основі ознак зображення. RPN використовується для визначення того, «де шукати», щоб зменшити обчислювальні вимоги загального процесу визначення об’єкту. RPN швидко та ефективно сканує кожне місцеположення, щоб оцінити, чи потрібна подальша обробка в певному регіоні. Це відбувається шляхом виводу k пропозицій, кожна з яких складається з двох балів, що представляють ймовірність розташування об'єкту в кожному місці. Після отримання пропозицій щодо регіону, інформація подається безпосередньо у Fast R-CNN[3].

Загалом, зараз R-CNN досягла набагато кращої швидкості навчання та більшої точності. Варто зазначити, що, хоча нові мережі досягли більш високої швидкості виявлення об’єктів, небагато моделей змогли покращити точність визначення об’єктів у порівнянні з Fast R-CNN. Іншими словами, Fast R-CNN, можливо, не найпростіший або найшвидший спосіб виявлення об'єктів, але він все ще є одним з найточніших.

1.3.3. Відстеження об'єктів.

Відстеження об'єктів означає процес нагляду за певним об'єктом, який представляє інтерес, або за декількома такими об'єктами у певній

сцені. Даний напрям традиційно має застосування у відео, де спостереження проводиться після виявлення початкового об'єкту. Тепер воно має вирішальне значення для автономних систем керування, зокрема, автоперевезення автомобілів від таких компаній, як Убер або Тесла.

Методи відстеження об'єктів відповідно до моделі спостереження можна розділити на дві категорії: генеративний метод та дискримінаційний метод. Генеративний метод використовує генеративну модель для опису очевидних характеристик і мінімізує помилку реконструкції для пошуку об'єкта (метод головних компонент PCA). Дискримінаційний метод може бути використаний для розмежування об'єкта і фону, його продуктивність є більш надійною, і він поступово стає основним методом відстеження. Дискримінаційний метод також називається «відстеження за визначенням» і глибинне навчання належить до цієї категорії. Щоб досягти відстеження шляхом виявлення, ідентифікуються всі об'єкти-кандидати для всіх кадрів та використовується глибинне навчання, щоб розпізнати серед кандидатів об'єкт, який нам потрібен. Тут можна відзначити дві базові мережеві моделі: стек автокодувальників (SAE) та згорткова нейронна мережа (CNN).

Найпопулярнішою глибинною мережею для відстеження завдань за допомогою SAE є Track Deer Learning, що пропонує попереднє навчання в автономному режимі та оптимізацію мережі в реальному часі. Розглянемо цей процес більш детально.

Попереднє тренування автоматичного кодувальника проводиться без нагляду та застосовується штабельний зворотній зв'язок. Для цього використовуються великі природні набори даних для зображень, щоб отримати загальне уявлення про об'єкт. Автокодером можна отримати більш надійну можливість експресії функцій, що здійснюється шляхом додавання шуму на вхідних зображеннях та відтворення оригінальних зображень.

При об'єднанні класифікатора з кодуючою частиною, попередньо підготовленої мережі, отримуємо класифікаційну мережу. Потім, використовуючи позитивні та негативні зразки з початкового кадру, налаштовується мережа, яка може дискримінувати поточний об'єкт і фон. DLT використовує частковий фільтр, як модель руху, для створення патчів-кандидатів поточного кадру. Класифікаційна мережа виводить оцінки ймовірності для цих патчів, що означає впевненість у їхніх класифікаціях, а потім вибирає найвищий з цих патчів як об'єкт. Під час оновлення моделі DLT використовується спосіб обмеження порогу.

Завдяки своїй перевазі в класифікації зображень та виявленні об'єктів, CNN стала основою для комп'ютерного бачення та візуального відстеження. Загалом, великомасштабна CNN може бути навчена, як класифікатор, так і трекер. Два представлені алгоритми відстеження на основі CNN – це повністю згортковий мережевий трекер (FCNT) і багатодоменний CNN (MD Net).

FCNT успішно аналізує та використовує можливості функціональних карт моделі VGG, що попередньо підготовлені ImageNet, і призводить до таких спостережень:

- карти ознак CNN можуть бути використані для локалізації та відстеження об’єктів;
- багато карт ознак CNN є шумними або такими, що не відповідають меті розпізнавання певного об'єкта з його фону;
- верхні шари фіксують семантичні поняття на категоріях об'єктів, тоді як нижні шари кодують більше дискримінаційні ознаки для захоплення варіацій між класами.

Через ці спостереження, FCNT розробляє мережу вибору ознак для вибору найвідповідніших карт функцій у версії conv4-3 та conv5-3 мереж VGG. Потім, щоб уникнути надмірного налаштування на шумних картах ознак, він також створює два додаткові канали (називаються SNet та GNet) для обраних карт ознак з двох шарів окремо. GNet фіксує інформацію про

категорію об'єкта, а SNet розпізнає об'єкт з фону з подібним виглядом. Обидві мережі ініціалізуються за допомогою заданого обмежувального поля в першому кадрі, щоб отримати теплові карти об'єкта, а для нових кадрів це поле відсікається та розповсюджується область інтересу (ROI), яка орієнтована на розташування об'єкта в останньому кадрі. Нарешті, через SNet та GNet, класифікатор отримує дві теплові карти для прогнозування, а трекер вирішує, яку саме теплову карту буде використано для генерації кінцевого результату відстеження, залежно від того, чи існує шум.

На відміну від ідеї FCNT, MD Net використовує всі послідовності відео для відстеження рухів у них. Наведені вище мережі використовують невідповідні дані зображення, щоб зменшити потребу в тренуванні даних відстеження, і ця ідея має певні відхилення. Об'єктом одного класу в цьому відео може бути фон в іншому відео, тому MD Net пропонує ідею мультидомену, щоб самостійно відрізняти об'єкт і фон в кожній області. І домен вказує на набір відео, які містять один і той самий об'єкт.

Як показано нижче, MD Net ділиться на дві частини: загальні шари і специфічні для домену К шаблонів. Кожна гілка містить бінарний класифікаційний шар із функцією активації softmax, яка використовується для того, щоб відрізняти об'єкт і фон в кожному домені, а також спільні шари, що діляться з усіма доменами, для забезпечення загального уявлення.

В останні роки завдання візуального відстеження викликає велику увагу дослідників. Існує багато напрямків, які були досліджені на сьогоднішній день: застосування мережевих моделей, таких як Recurrent Neural Network та Deep Belief Net; проектування структури мережі для адаптації до обробки відео та end-to-end навчання; оптимізація, та навіть поєднання процесу, структури та параметрів глибинного навчання з традиційними методами комп'ютерного бачення або підходами в інших областях, таких як мовна обробка та розпізнавання мовлення.

1.3.4. Семантична сегментація.

Семантична сегментація (Central to Computer Vision) – це процес сегментації, який розподіляє цілі зображення на піксельні групи, котрі потім можуть бути помічені та класифіковані. Зокрема, Central to Computer Vision намагається семантично зрозуміти роль кожного пікселя в зображенні (наприклад, це автомобіль, мотоцикл чи інше). Наприклад, на рисунку 1.17, окрім класифікації людини, дороги, автомобілів, дерев тощо, також необхідно окреслити межі кожного об'єкта. Тому, на відміну від класифікації, нам потрібні щільні піксельні прогнози з наших моделей.

Рисунок 1.17. – Приклад семантичної сегментації

Як і в інших задачах, пов'язаних із комп'ютерним баченням, CNN мережі мали величезний успіх у сегментації. Одним із популярних початкових підходів була класифікація патчів за допомогою ковзного вікна, де кожен піксель був розділений на класи, використовуючи патч зображень навколо нього. Це, однак, є дуже неефективним для обчислень, оскільки тут не використовуються спільні функції між перекриваючими патчами.

Вирішують цю недосконалість – повністю згорткові мережі UC Berkeley (FCN), які популяризували кінцеві CNN-архітектури для щільних прогнозів без будь-яких повністю підключених шарів. Це дозволило

генерувати карти сегментації для зображень будь-якого розміру. Також вони є набагато більш швидкими, порівняно з підходом класифікації патчів. Майже всі підходи семантичної сегментації, що з’являлися пізніше, також прийняли цю парадигму.

Проте залишається ще одна проблема – згортки при вихідній роздільній здатності зображення є дуже дорогими. Для боротьби з цим, FCN використовує downsampling та upsampling всередині мережі. Нижній шаблонний шар називається смугастою згорткою, тоді як шар верхнього зразка – перекладеною згорткою.

Незважаючи на зростаючі та спадні шари, FCN надає грубо сегментаційні карти через втрату інформації під час об'єднання. SegNet – це більш ефективна структура за критерієм використання пам’яті, ніж FCN, яка використовує повнозв’язні мережі та структуру кодера-декодера. У SegNet сполучення ярликів/пропусків вводяться з карт ознак високої роздільної здатності, щоб поліпшити неточний вміст зразків.

1.3.5. Сегментація екземпляру.

Сегментація екземпляру дозволяє сегментувати різні екземпляри класів, наприклад, маркування п’яти автомобілів з п’ятьма різними кольорами. Сегментація екземпляру вирішує набагато більш складні завдання, ніж класифікація.

До цих пір нами було продемонстровано, як використовуються карти ознак CNN для ефективного пошуку різних об'єктів у зображенні з обмежувальними полями. Сегментація екземпляру має на меті розширення таких методів, щоб знайти точні пікселі кожного об'єкта, а не обмежуватися його границями. Ця проблема сегментації екземпляра досліджується, зокрема, в Facebook AI, де використовується архітектура, відома, як Mask R-CNN.

Принцип роботи Mask R-CNN є простим і пов’язаний із роботою Fast R-CNN і Faster R-CNN. Оскільки Fast R-CNN добре показав себе в питаннях виявлення об'єктів, то було запропоновано покращити алгоритм його роботи, щоб також провести сегментацію на рівні пікселів.

Mask R-CNN робить це шляхом додання відповідної гілки до Fast R-CNN, що виводить бінарну маску, яка і визначає чи є даний піксель частиною об'єкта. Дана гілка – це повнозв’язна згорткова нейронна мережа на основі CNN. З використанням карти ознак CNN як входу, мережа виводить матрицю з одиницями у всіх місцях, де піксель належить об'єкту, та нулями в інших місцях (бінарна маска).

Mask R-CNN також вирішує проблему невірного розподілення регіонів вихідного зображення, обраних за допомогою RoIPool (регіони інтересів). Це відбувається за допомогою методу, відомого як RoIAlign (область інтересів вирівнювання). По суті, RoIAlign використовує білінійну інтерполяцію, щоб уникнути помилки при округленні, що викликає неточності у виявленні та сегментації. Після створення потрібних масок, Mask R-CNN поєднує їх з класифікаціями та обмежувальними коробками від Fast R-CNN, щоб генерувати точні сегментації.

1.4. Висновки до першого розділу

Показано, що на даний момент в рамках глибинного навчання нейронних мереж є багато перспективних напрямків, одним із основних серед яких є комп'ютерне бачення, що представляє собою розпізнавання візуальних середовищ та їх контекстів.

Проведений детальний огляд методів вирішення задач комп‘ютерного бачення показав, що для створення системи розпізнавання деталей одягу доцільно обрати нейромережевий метод.

Проаналізовано принципи організації згорткових нейронних мереж та основні методи навчання таких мереж. Розглянуті принципи організації

навчання нейронних мереж показали, що на сьогодні залишаються невирішеними питання ефективного навчання нейронних структур для підвищення точності класифікації об‘єктів.

2. ГЛИБИННІ НЕЙРОННІ МЕРЕЖІ

2.1. Види глибинних нейронних мереж

Глибинні нейронні мережі набули великої популярності через свою здатність вирішувати складні завдання. Існує сім основних видів глибинних нейронних мереж: нейронна мережа прямого поширення, мережа радіальних базисних функцій, самоорганізаційна карта Кохонена, рекурсивна нейронна мережа, рекурентна нейронна мережа, згорткова нейронна мережа, модулярна нейронна мережа.

2.1.1. Нейронна мережа прямого поширення

Дана нейронна мережа є однією з найпростіших форм ANN, де вхідні дані передаються в одному напрямку. Дані проходять через вхідні шари та передаються на шари виходу. Ця нейронна мережа може мати прихованих шарів. Нейронна мережа прямого поширення похибки, зазвичай використовує функції активації для поширення похибки.

На рисунку 2.1 зображено нейронну мережу прямого поширення, яка складається з одного шару. Тут обчислюється сума добутків вхідних значень і ваг, що подаються на вихід. Вихідне значення враховується, коли воно перевищує певне значення, тобто порогову величину (зазвичай 0), після чого нейрон вважається активним (зазвичай 1), а якщо він не спрацьовує, то приймає неактивне значення (зазвичай -1).

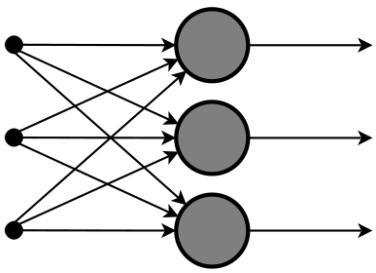

Рисунок 2.1 - Одношарова модель мережі прямого поширення

Нейронні мережі прямого поширення знаходять своє застосування в задачах комп'ютерного бачення та розпізнаванні мовлення, де класифікація цільових класів ускладнюється. Такі типи нейронних мереж добре справляються із зашумленими даними.

2.1.2. Мережа радіальних базисних функцій

Мережа радіальних базисних функцій (radial basis function network RBF) має два шари, перший – вхідні ознаки об'єднуються з функцією радіальних базисів у внутрішньому шарі, другий – вихідний шар цих функцій. При цьому враховуються відстані від точки до центру. Так, на рисунку 2.2 зображено графік, на якому показана відстань, яка розраховується від центру до точки в площині, подібно до радіуса окружності. Тут можна використовувати методи розрахунку відстані в евклідовому просторі. Модель залежить від максимального охоплення або радіуса окружності при класифікації точок у різні категорії. Якщо точка знаходиться в радіусі або навколо нього, то ймовірність того, що нова точка буде належати цьому класові, є високою.

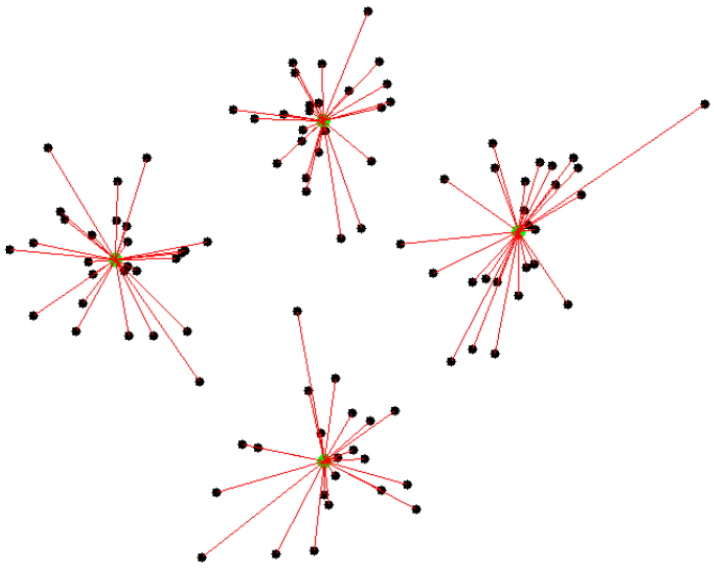

Рисунок 2.2 - Розрахунок відстані вхідних параметрів до спільного центру

2.1.3. Самоорганізаційна карта Кохонена

Метою карт Кохонена є введення до дискретної карти, що складається з нейронів, векторів довільної розмірності. На такій карті потрібно навчитись створювати власну організацію навчальних даних, яка складається з одного або двох вимірів. Під час тренування карта розташування нейрона залишається постійною, але ваги відрізняються в залежності від контексту. Цей процес самоорганізації має різні етапи. На першій фазі кожне значення нейрона ініціалізується з невеликою вагою та вхідним вектором. На другому етапі – нейрон, найближчий до точки, є «виграшним нейроном», а нейрони, пов'язані з таким нейроном, також рухаються до точки, як показано на рис. 2.3. Відстань між точкою та нейронами розраховується за евклідовою відстанню. Нейрон, що знаходиться на найменшій відстані перемагає. Через ітерації всі точки кластеризовані, і кожний нейрон представляє собою певний вид кластеру. Це є основою організації роботи нейронної мережі Кохонена.

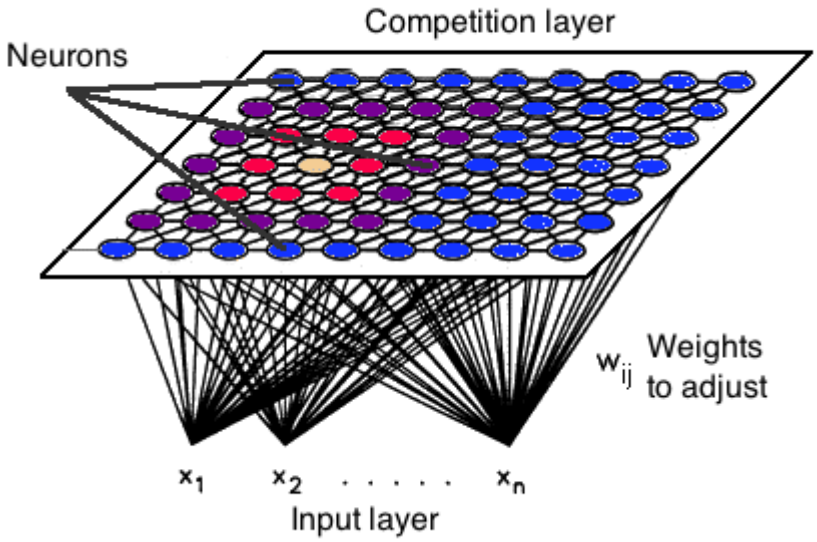

Рисунок 2.3 - Самоорганізаційна карта Кохонена

2.1.4. Рекурентні нейронні мережі

Рекурентна нейронна мережа працює за принципом збереження вихідного сигналу шару та його передачі назад на вхід, щоб допомогти спрогнозувати новий результат шару.

В цій мережі перший шар схожий на нейронну мережу прямого поширення з добутком суми ваг і функцій. Процес навчання рекурентної нейронної мережі починається після першого обчислення ваг, отже, до здійснення наступної ітерації кожен нейрон буде пам'ятати деяку інформацію, яку він зберіг на попередній ітерації, та грати роль комірки пам'яті при виконанні обчислень. У цьому процесі необхідно дозволити нейронній мережі працювати над прямим поширенням похибки та пам'ятати, інформацію, яка потрібна для подальшого використання. Якщо тренування мережі завершилося невірно, необхідно змінити темп навчання.

2.1.5. Згорткова нейронна мережа

Згорткові нейронні мережі подібні до нейронних мереж прямого поширення, де нейрони мають навчальну вагу та упередження. Ця мережа використовується в процесі обробки сигналів та зображень, наприклад, в сфері комп’ютерного бачення.

На рис. 2.4 представлена згорткова нейронна мережа. В даній нейронній мережі вхідний сигнал надається частково, як фільтр. Це допомагає мережі запам'ятовувати зображення частинами та проводити необхідні розрахунки. Ці обчислення включають перетворення зображення зі шкали RGB або HSI в сірий масштаб. Після цього, зміни в значенні пікселів допомагають виявити краї зображень, які можна буде класифікувати та розділити на різні категорії.

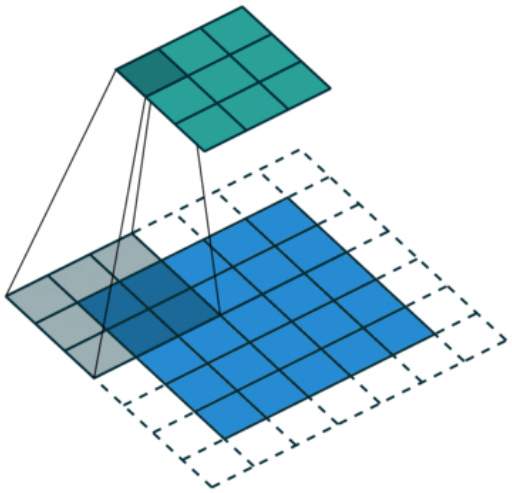

Рисунок 2.4 - Накладання фільтру згортки на зображення

2.1.6. Модулярні нейронні мережі

Модулярні нейронні мережі складаються із множини різних мереж, що працюють незалежно для досягнення конкретного результату. Кожна нейронна мережа має безліч входів, які, порівняно з іншими мережами, є унікальними. Під час виконання завдань ці мережі не взаємодіють одна з одною та не обмінюються жодними сигналами. Перевага модульної нейронної мережі полягає в тому, що вона розбиває великий обчислювальний процес на менші компоненти, це полегшує складність операцій, які необхідно виконати під час навчання. Дане розподілення завдань допомагає зменшити кількість з'єднань та запобігає взаємодії окремих нейронних мереж між собою, що значно підвищує швидкість обчислень, проте час обробки буде залежати від кількості нейронів та їх участі в обчисленні результатів. На рис. 2.5 зображено приклад модулярної нейронної мережі.

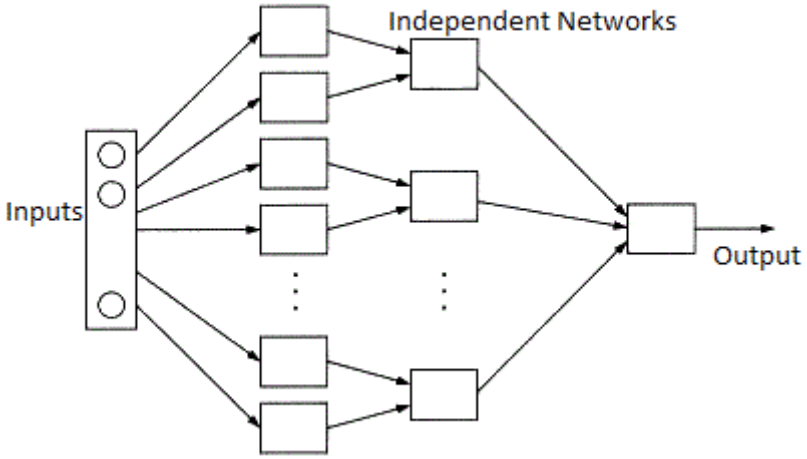

Рисунок 2.5 - Модулярна нейронна мережа

2.2. Основні елементи згорткової нейронної мережі

Більшість алгоритмів машинного навчання включають в себе гіперпараметри – змінні, значення яких встановлюється перед оптимізацією параметрів моделі. Встановлення значень гіперпараметрів можна розглядати як вибір моделі, яку необхідно використати для певного набору даних. Гіперпараметри часто встановлюються вручну або обираються за допомогою деякого алгоритму пошуку оптимальних параметрів. Нейромережі можуть мати багато гіперпараметрів, зокрема ті, які визначають структуру самої мережі та процедуру її навчання.

2.2.1. Гіперпараметри стохастичного градієнта

Під час навчання нейронної мережі результуюча модель буде залежати не тільки від обраної структури, але і від методу, який використовується для встановлення параметрів мережі. Сам спосіб навчання може мати багато гіперпараметрів. Стохастичний градієнтний спуск оновлює параметри мережі з використанням градієнтного спуску на підмножині навчальних даних, що періодично перемішується або

вважається нескінченною. Нехай t < T – певна ітерація, тоді параметри θ оновлюватимуться на даній ітерації, як показано в рівнянні 1:

$$\theta^{(t)} \leftarrow \theta^{(t-1)} - \epsilon_t \frac{1}{B} \sum_{t'=Bt+1}^{B(t+1)} \frac{\partial L(z_{t'}, \theta)}{\partial \theta} \quad , \qquad (1)$$

де $z_{t`}$ – це приклад тренувальної множини. Гіперпараметрами в даному випадку є: функція втрат L, темп навчання на ітерації t – ϵ_t, розмір пакету прикладів B, та кількість ітерацій T.

Темп навчання

Темп навчання ϵ_t визначає як «швидко» градієнт оновлення прямує до глобального мінімуму. Якщо темп навчання дуже низький, модель буде сходитися надто повільно; якщо ж темп навчання занадто високий – модель буде неправильно натренована. Коли ϵ_t змінюється з ітераціями, необхідно визначити початкову швидкість навчання ϵ_0 та спосіб обчислення ϵ_t. Для нейронних мереж із стандартизованими вхідними даними або даними, що мають значення в діапазоні [0,1], ϵ_0 зазвичай встановлюється в межах 1 та 10^{-6}, а найкращим значенням для першої спроби є 0,01. Проте, зважаючи на важливість ϵ_0, даний параметр потребує точного налаштування [4].

Темп навчання ϵ_t зазвичай зменшується зі зміною ітерації. Якщо використати загальновідоме евристичне припущення, спочатку потрібно знайти гідне початкове значення темпу навчання, після чого його треба точно налаштувати. Одним з підходів є ініціалізація $\epsilon_t=\epsilon_0$, де $t<\tau$ (τ - новий гіперпараметр), після чого нове значення темпу навчання розраховується за формулою $\epsilon_t=\epsilon_0/t\alpha$ (α - ще один гіперпараметр). Необхідно використовувати менші значення α, коли функція, на якій шукається глобальний мінімум, не є опуклою, і використовується певний метод

градієнтного усереднення (наприклад, імпульс). Також загальноприйнятою практикою є адаптивне встановлення значення τ в залежності від ітерації.

Функція втрат (Loss function)

Функція втрат порівнює вже відомі вихідні дані для тренувального прикладу та значення, яке було отримане під час навчання. В загальному випадку функція втрат – квадрат віддалі між двома точками, яка визначається за рівнянням 2:

$$L = \frac{1}{2}\sum_i (y_i - z_i)^2 \quad , \tag{2}$$

де y_i – і-те значення вихідного шару мережі, а z_i - це і-те значення відповіді, що визначені в тренувальній множині. Коефіцієнт 1/2 використовується для того, щоб зробити градієнт більш простим. Коли вихідні значення нейронної мережі розглядаються як розподіл імовірності (наприклад, використовується вихідний шар softmax), загальним і теоретично доцільним є використання функції втрат перехресної ентропії, яка виглядає як рівняння 3:

$$L = -\sum_i y_i \log(z_i) \tag{3}$$

Розмір пакетної вибірки

Теоретично вибір B переважно визначається вручну: коли значення B – більше, оновлення ваг можна обчислювати більш ефективно завдяки використанню паралельних архітектур; коли значення B – менше, можна зробити більше оновлень ваг. Через те, що значення B не повинно впливати на ефективність валідаційного та тестового етапів, B можна оптимізувати окремо від інших гіперпараметрів.

Число тренувальних ітерацій

Найпоширеніший спосіб визначення T використовує принцип ранньої зупинки. Рання зупинка просто припиняє тренування після того, як продуктивність навчання на етапі валідації перестає збільшуватись (тобто похибка починає неухильно зростати, а не зменшуватись). Це може бути потужним засобом запобігання перенавчання, оскільки налаштуванням інших гіперпараметрів можна знехтувати.

Імпульс

Найпоширенішою методикою є «гладке» оновлення градієнта, використовуючи інтеграційний фільтр разом з параметром β, як показано у функції 4:

$$\bar{g} \leftarrow (1-\beta)\bar{g} + \beta\frac{\partial L(z_t,\theta)}{\partial\theta}, \quad (4)$$

де $\bar{g}$ може бути використаний замість «справжнього» оновлення градієнта при застосуванні градієнтного спуску. Деякі математичні підходи можуть забезпечити більш швидкий спуск градієнта при використанні відповідного імпульсу, однак для випадкового градієнтного спуску, стандартне градієнтне оновлення (β = 1) з гармонійно зменшеним темпом навчання є оптимальними.

2.2.2. Гіперпараметри моделі

Сама структура нейронної мережі містить в собі численні гіперпараметри, включаючи розмір і нелінійність кожного шару. Чисельні властивості ваг також певним чином обмежені, а їх ініціалізація може сильно впливати на показники продуктивності. Попередня обробка

вхідних даних також може бути важливою для забезпечення спуску градієнта. Багато гіперпараметрів можуть варіюватися в різних шарах.

Кількість прихованих шарів

Великі приховані шари дозволяють нейронній мережі добре натренуватись, тому навіть при використанні регуляризації дуже важливо створювати приховані шари необхідної розмірності. Використання однакової розмірності для всіх прихованих шарів, як правило покращує роботу. Крім того, використання першого прихованого шару, який більше, ніж вхідний шар – є вдалою практикою. Також при використанні безконтрольного попереднього навчання, шари повинні бути набагато більшими, ніж при чисто контрольованій оптимізації.

Відсікання вагових компонент

Щоб зменшити надмірне перенавчання, іноді до критерію тренування (функція втрат) додають регуляризацію ваг мережі. Під час спроб утримати значення ваг мережі θ близькими до нуля (типовий випадок), регуляризація L2 додає $\lambda_2\Sigma_i\theta_i^2$, а L1 додає $\lambda_1\Sigma_i|\theta_i|$, де λ_2 та λ_1 - множники лагранжа, які визначають, наскільки важливою є дана регуляризація.

Нелінійність

Найбільш вживаними нелінійностями є: сигмоїда $1/(1 + e^{-a})$, яка плавно переводить вхідні дані у діапазоні [0,1]; формула, що еквівалентна сигмоїді з діапазоном [-1,1]; випрямляч (ReLU) max(0, a), або порогова функція. Сигмоїда на вихідному рівні може спричинити проблеми, у випадку використання глибинними мережами з наглядачем.

Ініціалізація ваг

Упередження, як правило, ініціалізуються зі значенням 0, але ваги повинні бути ініціалізовані ретельно; їхня ініціалізація може мати великий вплив на локальний мінімум, виявлений алгоритмом навчання. Якщо ініціалізувати випадковим чином, ваги часто ініціалізуються в діапазоні [-r, r]. Для обмежених машин Больцмана слід використовувати нормальний розподіл з малим стандартним відхиленням (0,1 або 0,01). Попереднє навчання без наглядача може, по суті, розглядатися як складна ініціалізація, що в більшості випадків не шкодить загальному процесу навчання. Загальні методи ініціалізації без нагляду включають в себе обмежені машини Больцмана (DBM) або автокодери.

Випадкове початкове значення та усереднення моделі

Багато процесів, що беруть участь у навчанні нейронної мережі, передбачають використання генератора випадкових чисел (наприклад, випадкова вибірка даних навчання, ініціалізація ваг тощо). Як наслідок, значення, передані генератору випадкових чисел, можуть мати незначний вплив на результати. Проте інші випадкові значення можуть спричинити утворення моделі, що принципово відрізняється від попередньої (навіть, якщо вона і працює). Це призводить до тренування низки моделей з кількома випадковими наборами значень, які використовують усереднену модель (за допомогою пакетування, байєсівських методів) для підвищення продуктивності.

Попередня обробка вхідних даних

Вхідні дані можуть сильно впливати на продуктивність мережі. Загальновживаними є: поелементна стандартизація (віднімання середнього і ділення за стандартним відхиленням), аналіз основних компонентів, уніфікованість (перетворення кожного значення об'єкта до його приблизного нормалізованого рангу або квантиля) і нелінійність (такі як логарифм або квадратний корінь).

2.2.3. Дослідження гіперпараметрів

Кількість вказаних вище гіперпараметрів вказує на те, що при створенні нейронної мережі існує велика кількість варіантів однієї мережі, навчання якої буде визначатися параметрами, що були обрані дослідниками. Тому, для забезпечення відтворюваності результатів дослідження, при встановлені гіперпараметрів слід використовувати принциповий підхід або, принаймні, чітко вказувати обрані гіперпараметри в описі моделі. Якщо дослідник бере участь у пошуку гіперапараметрів, а їхні значення не зазначені чітко, результати дослідження важко відтворити.

Вибір гіперпараметру можна розглядати як задачу оптимізації (яка не обов'язково опукла в будь-якій окремій змінній) та проблему узагальнення (оскільки можливе перенавчання). Вибір стає особливо складним через обчислювальні витрати: кожний гіперпараметр повинен бути використаний для навчання нової моделі, а тренування моделей, як правило, займає тривалий час. Проте було доведено, що для деяких гіперпараметрів оптимальне значення можна отримати за допомогою більш швидких методів (наприклад, випадкового встановлювання ваг).

У більшості випадків для кожного з гіперпараметрів необхідно встановити діапазон значень, які потрібно випробувати. Можливий варіант, коли за наслідками випробувань найкраще значення опиниться на границі даного діапазону. Це буде свідчити про те, що найбільш

оптимальні значення гіперпараметру будуть знаходиться за межами встановленого діапазону. Також варто враховувати, що «найкраще» значення не обов'язково знаходиться посередині інтервалу, оскільки функції не завжди є опуклими. Необхідно також обрати «шкалу» інтервалу. Найчастіше для визначення інтервалу використовують логарифмічний діапазон, оскільки співвідношення різних значень переважно є кращим показником очікуваних змін.

Координатний спуск

Можна застосувати ідею координатного спуску до оптимізації гіперпараметрів – залишити без змін усі гіперпараметри, крім одного, і коригувати цей гіперпараметр, щоб звести до мінімуму похибку. Метод використовується через те що більшість гіперпараметрів нейронної мережі мало впливають на процес навчання, тому варто зосередитись на одному параметрі, та оптимізувати модель за допомогою підбору цього параметру.

Пошук сіткою

Пошук сіткою передбачає налаштування гіперпараметрів із використанням заданого діапазону значень. Це передбачає поелементний добуток усіх інтервалів, тому час обчислення експоненційно зростає із кількістю параметрів. Розглянемо стандартну структуру класифікації - у існує набір даних, який необхідно розділити на дані для тренувань (деформація) та дані для валідації. Необхідно вирішити проблему оптимізації (яка, як правило, може бути чимось на кшталт мінімізації помилки навчання плюс термін регуляризації), що є функцією параметрів моделі, скажімо w, зразка тренувань та деяких гіперпараметрів α та β.

Зверніть увагу, що ця функція перевірки валідації дуже дорога для оцінки - для кожного значення α та β, щоб знайти значення цієї функції,

потрібно вирішити проблему оптимізації P. Крім того, ця функція може бути невипуклою, гладкою і т. д., тому що принципово неможливо знайти глобальний мінімум цієї функції.

Тому доцільним є використання пошуку сіткою: обрати множину значень α - (α1, α2, ...), множину значень β - (β1, β2, ...) і для кожної пари значень, оцінити функцію позибки. Потім обрати пару, яка дає мінімальне значення функції похибки.

Пари (α1, β1), (α1, β2), ..., (α2, β1), (α2, β2), ... при нанесенні в просторі виглядають сіткою, звідки й пішла назва методу.

Випадковий пошук

Простою альтернативою пошуку сіткою є вибір гіперпараметрів випадковим чином. Це аналогічно тривіально паралелізується і може спрацювати набагато краще, ніж пошук сіткою в режимі реального часу, що може значно скоротити час пошуку, порівняно з пошуком сіткою. До того ж, лише деякі гіперпараметри мають велике значення. Єдина реальна відмінність між пошуком сіткою та випадковим пошуком знаходиться на першому кроці - випадкова пошукова система вибирає точку випадковим чином з конфігураційного простору.

2.2.4. Шари згортки

Згортка є досить простою операцією, основою якої виступає ядро – невелика матриця ваг, що «ковзає» над вхідними даними, виконує поелементне множення досліджуваної частини вхідного сигналу, а потім підсумовує усі значення у вигляді одного вихідного пікселя.

Ядро повторює цей процес для кожного місця, де воно проходить, перетворюючи одну 2D матрицю ознак на іншу. Вихідні ознаки є, по суті, зваженими сумами (з вагами, що є значенням самого ядра) вхідних ознак,

розташованих приблизно в одному місці вхідного пікселя на вхідному шарі. Незалежно від того, чи знаходиться ознака входу в межах «приблизно того ж місця», чи визначається безпосередньо, а також чи знаходиться вона в області ядра, що створює вихідний сигнал, чи ні – розмір ядра безпосередньо свідчить про те, скільки вхідних ознак поєднуються для створення нової ознаки виходу.

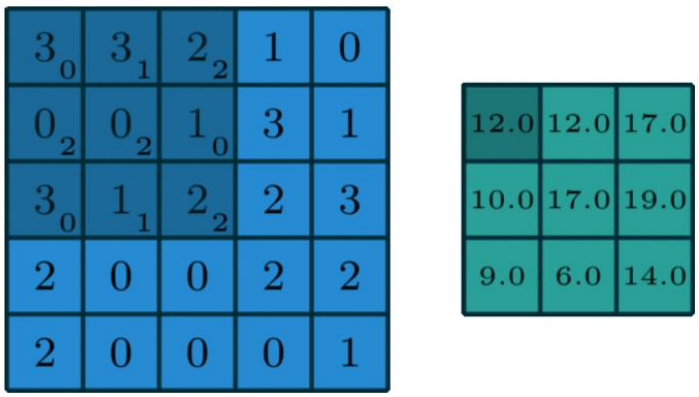

Рисунок 2.6 – Приклад згортки та створення нового ядра

Згортковий шар досить різко контрастує з повністю підключеним шаром. У наведеному вище прикладі 5×5=25 вхідних ознак і 3×3=9 вихідних ознак. У випадку стандартного повнозв’язного шару матриця мала би 25×9=225 параметрів, причому кожна ознака виходу була би зваженою сумою кожної окремої ознаки введення. Згортки дозволяють зробити це перетворення лише за 9 параметрами, з кожною ознакою виходу, а, отже, замість того, щоб «дивитися» на кожну ознаку входу, достатньо тільки «поглянути» на ознаки виходу, що знаходяться приблизно в тому самому місці [5].

На рис. 2.7 видно, що під час процесу ковзання краї, зазвичай, «відсікаються», перетворюючи матрицю ознак 5×5 на 3×3. Пікселі на краю ніколи не знаходяться в центрі ядра, оскільки ядро не виходить за межі краю. Це не дуже зручно тому, що часто необхідно, щоб розмір вихідного сигналу дорівнював вхідному.

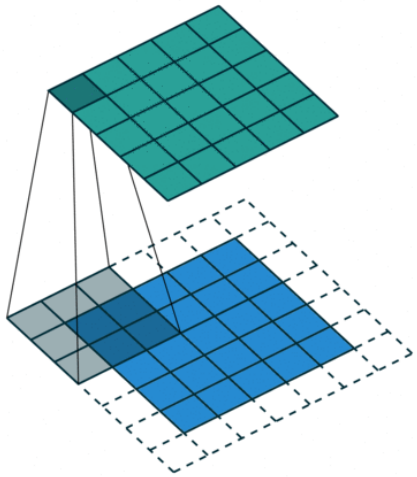

Рисунок 2.7 – Використання падингу під час згортки

Падинг виконує дуже корисну функцію – додавання до країв додаткових «фальшивих» пікселів (зазвичай, значеннями 0, який визначається терміном «нульовий падинг»). Таким чином, ядро при переміщенні може дозволити оригінальним пікселям краю знаходитись у центрі, одночасно, поширюючись на фальшиві пікселі за межею, що дозволяє виводити вихідний сигнал такого ж розміру, як і вхідний.

Страйдинг

Часто при запуску згорткового шару необхідно отримати вихід з меншим розміром, ніж вхід. Це звичайна ситуація у згорткових нейронних мережах, де розмір просторових розмірностей зменшується при збільшенні кількості каналів. Одним із способів досягнення цього є використання шару агрегування, ще одним – страйдинг (див. рис. 2.8).

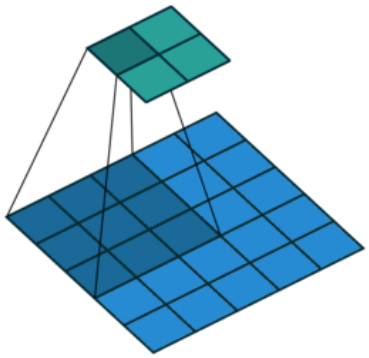

Рисунок 2.8 – Використання страйдингу у згортковій мережі

Ідея страйду полягає в тому, щоб пропустити деякі розташування ядра під час згортки. Страйд розміром 1 означає, що нове розташування ядра обирається із кроком в один піксель, тобто, звичайна робота згорткового шару. При страйді розміром 2 – нове розташування ядра обирається із кроком в два пікселі, пропускаючи всі інші розташування в процесі згортки, зменшуючи їх приблизно вдвічі. Відповідно, страйд розміром 3 передбачає пропуск кожного третього розташування.

Сьогодні для зменшення розмірів внутрішніх шарів, більш сучасні мережі, такі як архітектура ResNet, повністю відмовляються від шарів агрегування на користь згорткових шарів з використанням страйду.

Багатоканальна версія згортки

Більшість зображень на вході мають 3 канали, та з просуванням вглиб мережі це число тільки збільшується. Зручно розглядати канали, як альтернативну версію певного зображення, яка підкреслює деякі аспекти цього зображення та уникає інших. На рис. 2.9 зображено три варіанти одного й того самого зображення, розкладеного на одноканальні зображення по каналу кольору.

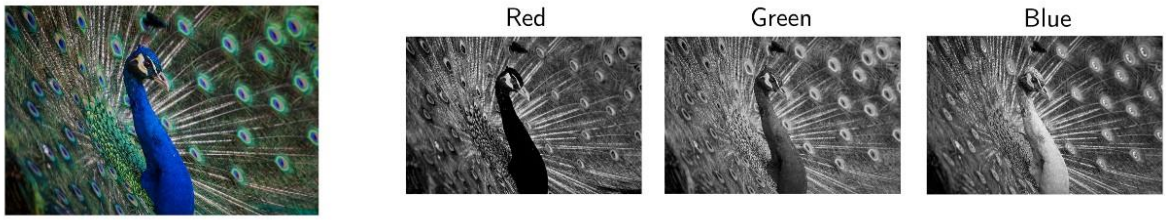

Рисунок 2.9 – Розкладання зображення на альтернативні одноканальні зображення

Більшість часу нейронні мережі мають справу із зображеннями RGB з трьома каналами кольорів. Важливо врахувати при цьому, що у випадку з одним каналом, термін фільтр та ядро є взаємозамінними, тоді, як насправді, вони є досить різними категоріями. Фактично, кожен фільтр – це сукупність ядер, де для кожного вхідного каналу шару існує одне унікальне ядро. Один фільтр у згортковому шарі обробляє один і тільки один вхідний канал. При цьому певне ядро фільтра «ковзає» над відповідними вхідними каналами, створюючи їх оброблену версію. Деякі ядра можуть мати більшу вагу, ніж інші, що дозволяє їм надавати перевагу певним каналам вводу (наприклад, фільтр може мати ядро червоного каналу з більшою вагою, отже, більше відповідати відмінностям в ознаках червоного каналу).

Потім кожна оброблена версія одноканального вхідного сигналу сумується з іншими для формування одного спільного каналу. Ядра кожного фільтру створюють одну версію кожного каналу, а фільтр, у цілому, виробляє один загальний вихідний канал.

Такі перетворення використовуються для будь-якої кількості фільтрів, кожен з яких за допомогою описаного вище процесу обробляє вхід з власним набором ядер і скалярним зміщенням, створюючи власний вихідний канал. Потім вони об'єднуються всі разом, щоб отримати загальний результат, причому кількість вихідних каналів – це кількість фільтрів, що були використані. Зазвичай, застосовується нелінійність,

перш ніж передавати це як вхід в інший шар згортки, який потім повторює цей процес.

Згортки залишаються лінійними перетвореннями

Розглянемо такий випадок: нехай існує вхідний сигнал 4 × 4, і необхідно перетворити його на сітку 2 × 2. Якщо використати мережу прямого поширення, вхідний сигнал 4 × 4 трансформується у вектор довжиною 16, що пройде через повнозв'язний шар із вхідними даними довжиною 16 та вихідним довжиною 4. Можна візуалізувати матрицю ваг W для шару (рис. 2.10)

$$\begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} & w_{1,4} & w_{1,5} & w_{1,6} & w_{1,7} & w_{1,8} & w_{1,9} & w_{1,10} & w_{1,11} & w_{1,12} & w_{1,13} & w_{1,14} & w_{1,15} & w_{1,16} \\ w_{2,1} & w_{2,2} & w_{2,3} & w_{2,4} & w_{2,5} & w_{2,6} & w_{2,7} & w_{2,8} & w_{2,9} & w_{2,10} & w_{2,11} & w_{2,12} & w_{2,13} & w_{2,14} & w_{2,15} & w_{2,16} \\ w_{3,1} & w_{3,2} & w_{3,3} & w_{3,4} & w_{3,5} & w_{3,6} & w_{3,7} & w_{3,8} & w_{3,9} & w_{3,10} & w_{3,11} & w_{3,12} & w_{3,13} & w_{3,14} & w_{3,15} & w_{3,16} \\ w_{4,1} & w_{4,2} & w_{4,3} & w_{4,4} & w_{4,5} & w_{4,6} & w_{4,7} & w_{4,8} & w_{4,9} & w_{4,10} & w_{4,11} & w_{4,12} & w_{4,13} & w_{4,14} & w_{4,15} & w_{4,16} \end{bmatrix}$$

Рисунок 2.10 – Матричне представлення ваг повнозв'язного шару

Хоча операція з ядром згортки може на перший погляд виглядати дещо дивно, вона все ще є лінійним перетворенням з еквівалентною матрицею. Якщо необхідно застосувати ядро K розміром 3 на зміненому вхідному сигналі 4 × 4 для отримання вихідного сигналу 2 × 2, еквівалентна матриця перетворення буде виглядати [6]:

$$\begin{bmatrix} k_{1,1} & k_{1,2} & k_{1,3} & 0 & k_{2,1} & k_{2,2} & k_{2,3} & 0 & k_{3,1} & k_{3,2} & k_{3,3} & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{1,1} & k_{1,2} & k_{1,3} & 0 & k_{2,1} & k_{2,2} & k_{2,3} & 0 & k_{3,1} & k_{3,2} & k_{3,3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & k_{1,1} & k_{1,2} & k_{1,3} & 0 & k_{2,1} & k_{2,2} & k_{2,3} & 0 & k_{3,1} & k_{3,2} & k_{3,3} & 0 \\ 0 & 0 & 0 & 0 & 0 & k_{1,1} & k_{1,2} & k_{1,3} & 0 & k_{2,1} & k_{2,2} & k_{2,3} & 0 & k_{3,1} & k_{3,2} & k_{3,3} \end{bmatrix}$$

Рисунок 2.11 – Матричне представлення ваг згорткового шару

З матриці видно, що використовуються лише 9 параметрів. Згортка, як і раніше, є лінійним перетворенням, але, в той самий час, це також різний вид перетворення. Для матриці з 64 елементами існує всього 9 параметрів, які повторно використовуються кілька разів. Кожен вихідний вузол бачить лише вибране число входів (ті, що знаходяться всередині

ядра), і немає взаємодії з будь-якими іншими входами тому, що ваги для них встановлено на 0.

Локальність

• Ядра об'єднують пікселі лише з невеликої локальної області, щоб сформувати вихід, тобто вихідні ознаки характеризують лише вхідні ознаки з невеликої локальної області.

• При цьому ядра застосовуються глобально по всьому зображенню для створення матриці виходів.

Отже, при зворотному розповсюдженні, яке повністю відрізняється від вузлів класифікації мережі, ядра мають задачу вивчення ваг для створення функцій лише з набору локальних входів. Також, оскільки саме ядро застосовується по всьому зображенню, функції ядра, що вивчаються, повинні бути достатньо загальними, щоб надходити з будь-якої частини зображення.

Пікселі завжди відображаються у послідовному порядку та впливають один на одного, наприклад, якщо всі пікселі, що знаходяться поруч, червоні, то, швидше за все, шуканий піксель також є червоним. Якщо ж в процесі такого порівняння зустрічається відхилення, це є цікавою аномалією, яку можна перетворити на функцію.

Ця ідея, насправді, заснована на основних методах ідентифікації ознак в комп'ютерному баченні, які були створені раніше. Наприклад, для виявлення краю можна використовувати фільтр виявлення краю Собеля – ядро з фіксованими параметрами, що працює так само, як і стандартна одноканальна згортка.

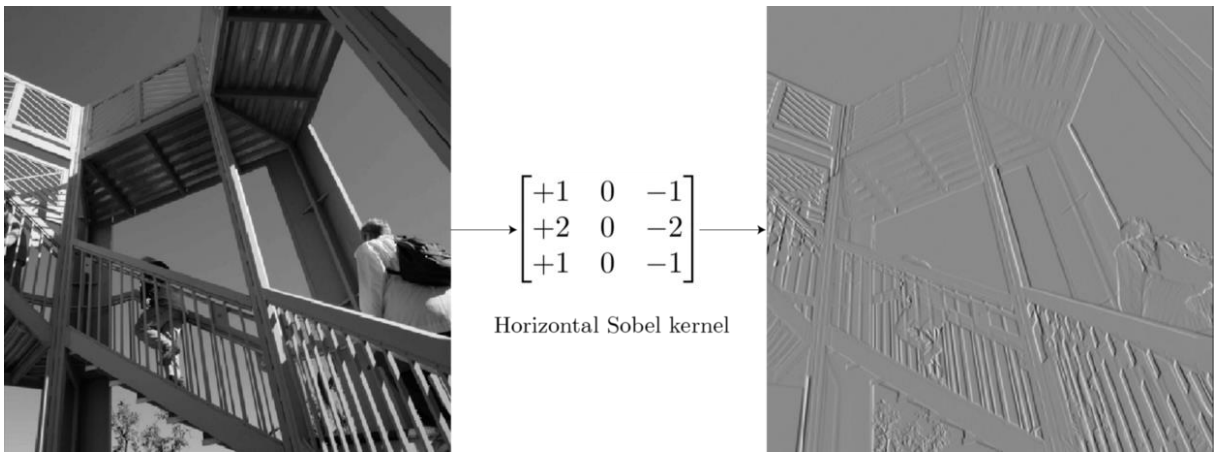

Рисунок 2.12 – фільтр виявлення краю Собеля

Для сітки, яка не містить країв (наприклад, фонове небо), більшість пікселів мають однакове значення, тому загальний вихід ядра в цьому місці становить 0. Для сітки з вертикальним краєм існує різниця між пікселями ліворуч і праворуч від краю, отже, ядро вираховує це як відмінність від нуля, активізуючи та виявляючи краї. Ядро працює лише із сітками 3 × 3, одночасно виявляючи аномалії в локальному масштабі.

2.2.5. Повнозв’язна нейронна мережа-класифікатор

Повнозв’язний шар нейронів є повністю зв’язаним з усіма активаціями попереднього шару, як це робиться в звичайних нейронних мережах. Отже, їх активацію можна обчислити за допомогою матричного множення, а потім зміщення.

Варто зазначити, що єдиною відмінністю між повнозв’язними шарами та шарами згортки є те, що нейрони в шарі згорток з'єднані лише з локальним регіоном на вході. Однак нейрони в обох шарах все ще обчислюють поелементний добуток, тому їх функціональна форма однакова. Таким чином, виявляється, що можна конвертувати повнозв’язні шари у шари згорток та навпаки:

- Для будь-якого згорткового шару є такий самий повнозв'язний шар, який реалізує таку ж передавальну функцію. Матриця ваг буде однакової розмірності, і, за

винятком деяких блоків (через локальне підключення), де ваги у багатьох блоках рівні (через розподіл параметрів), буде містити нулі.

- І, навпаки, будь-який повнозв’язний шар може бути перетворений у шар згортки. Наприклад, повнозв’язний шар розміром K=4096, який сприймає вхідний сигнал розміром 7×7×512, можна еквівалентно виразити як згортковий шар з кількістю фільтрів F=7, кількістю падингів P=0, S=1, K=4096. Іншими словами, встановлюється розмір фільтра як точний розмір вхідного шару, отже, його вихід буде просто 1×1×4096, оскільки, тільки один фільтр дає ідентичний результат, як повнозв'язний шар.

З цих двох перетворень найбільш корисною є практика перетворення шару FC у шар CONV. Розглянемо архітектуру ConvNet, яка приймає зображення 224x224x3, а потім використовує серію згорткових шарів і шарів агрегування, щоб зменшити зображення до об'єму активації розміром 7x7x512 (в архітектурі згорткової мережі це робиться за допомогою п’яти об'єднаних шарів, які кожного разу зменшують пропускну здатність в два рази, внаслідок чого набувається остаточний просторовий розмір 224/2/2/2/2/2 = 7). Після цього згорткова мережа використовує два повнозв’язні шари розміром 4096 і, нарешті, останній повнозв’язний шар з 1000 нейронами, які обчислюють результати класу.

Нижче вказаний алгоритм, за допомогою якого можна перетворити кожний з цих повнозв’яних шарів у відповідні версії згорткових шарів:

- Замінити перший повнозв'язний шар, який має розмірність [7x7x512], конфігураційним шаром, який використовує розмір фільтра F = 7, що дає об'єм виводу [1x1x4096].

- Другий повнозв’язний шар також замінити на згортковий шар, який використовує розмір фільтра F = 1, що дає на виході розмірність [1x1x4096].
- Аналогічним чином замінити останній повнозв’язний шар на згортковий шар, який використовує розмір фільтра F = 1, що дає кінцевий результат [1x1x1000].

Кожне з цих перетворень могло б практично маніпулювати (наприклад, перетворювати) вагові матриці W у кожному повнозв’язному шарі на згортковий шар. Виявляється, що це перетворення дуже ефективно дозволяє проходити по оригінальній згортковій мережі за багатьма просторовими позиціями у збільшеному зображенні.

Наприклад, якщо зображення 224x224 утворює сигнал розміром [7x7x512] і необхідно зменшення на 32, тоді пересилання зображення розміром 384x384 через перетворену архітектуру дасть еквівалентний сигнал у розмірі [12x12x512], оскільки 384/32 = 12. Після проходження наступних трьох згорткових шарів, отримуємо остаточний обсяг розміру [6x6x1000], оскільки (12 - 7) / 1 + 1 = 6. Таким чином, замість одного вектору класу розміром [1x1x1000], буде отримано весь макет класів 6x6 на зображенні 384x384 [7].

Природно, що проходження через перетворену згорткову мережу набагато ефективніше, ніж використання оригінальної згорткової мережі у всіх цих місцях послідовно. Такий підхід часто використовується на практиці для покращення продуктивності, коли, наприклад, необхідно змінювати розмір зображення, щоб зробити його більш великим, де найбільш раціональним є застосування перетвореної згорткової мережи для оцінки класів на багатьох просторових позиціях.

2.2.6. Функція активації Softmax

Функція Softmax приймає N-мірний вектор дійсних чисел і перетворює його в вектор значень в діапазоні (0,1), який визначається за формулою:

$$p_i = \frac{e^{a_i}}{\sum_{k=1}^{N} e_k^a} \qquad (6)$$

Як видно з назви, функція softmax є спрощеною версією функції max. Замість того, щоб вибирати одне максимальне значення, функція пропоційно розподіляє значення, щоб в сумі отримати одиницю.

Ця властивість функції softmax, яка виводить розподіл імовірності, робить її придатною для імовірнісного тлумачення у завданнях класифікації.

Щоб зробити функцію softmax чисельно стабільною, необхідно нормалізувати значення вектора шляхом множення чисельника і знаменника на константу C.

$$\begin{aligned} p_i &= \frac{e^{a_i}}{\sum_{k=1}^{N} e^{a_k}} \\ &= \frac{Ce^{a_i}}{C\sum_{k=1}^{N} e^{a_k}} \\ &= \frac{e^{a_i+\log(C)}}{\sum_{k=1}^{N} e^{a_k+\log(C)}} \end{aligned} \qquad (7)$$

Можна обрати довільне значення для log(C), хоча, зазвичай, обирається log (C) = -max(a), оскільки, він зміщує всі елементи вектору в діапазон від від’ємних значень до нуля.

Через бажану властивість функції softmax, що визначає розподіл імовірності, дана функція використовується як останній шар в нейронних мережах. Для цього потрібно обчислити похідну або градієнт і повернути його назад до попереднього шару під час зворотного розповсюдження похибки.

$$\frac{\partial p_i}{\partial a_j} = \frac{\partial \frac{e^{a_i}}{\sum_{k=1}^{N} e^{a_k}}}{\partial a_j} \tag{8}$$

За основними правилами знаходження похідної, необхідно знайти похідну функції (8):

$$f(x) = \frac{g(x)}{h(x)} \tag{9}$$

Отримуємо похідну (10)

$$f'(x) = \frac{g\prime(x)h(x) - h\prime(x)g(x)}{h(x)^2} \tag{10}$$

В даному випадку $g(x) = e^{a_i}$ та $h(x) = \sum_{k=1}^{N} e^{a_k}$. В $h(x)$, $\frac{\partial}{\partial e^{a_j}}$ завжди буде e^{a_j}. Проте для g(x), $\frac{\partial}{\partial e^{a_j}}$ буде e^{a_j} тільки якщо i = j, в іншому випадку 0.

Якщо i = j

$$\begin{aligned} \frac{\partial \frac{e^{a_i}}{\sum_{k=1}^{N} e^{a_k}}}{\partial a_j} &= \frac{0 - e^{a_j} e^{a_i}}{\left(\sum_{k=1}^{N} e^{a_k}\right)^2} \\ &= \frac{-e^{a_j}}{\sum_{k=1}^{N} e^{a_k}} \times \frac{e^{a_i}}{\sum_{k=1}^{N} e^{a_k}} \\ &= -p_j . p_i \end{aligned} \tag{10}$$

Якщо i≠j

$$\begin{aligned} \frac{\partial \frac{e^{a_i}}{\sum_{k=1}^{N} e^{a_k}}}{\partial a_j} &= \frac{e^{a_i} \sum_{k=1}^{N} e^{a_k} - e^{a_j} e^{a_i}}{\left(\sum_{k=1}^{N} e^{a_k}\right)^2} \\ &= \frac{e^{a_i}\left(\sum_{k=1}^{N} e^{a_k} - e^{a_j}\right)}{\left(\sum_{k=1}^{N} e^{a_k}\right)^2} \\ &= \frac{e^{a_j}}{\sum_{k=1}^{N} e^{a_k}} \times \frac{\left(\sum_{k=1}^{N} e^{a_k} - e^{a_j}\right)}{\sum_{k=1}^{N} e^{a_k}} \\ &= p_i(1 - p_j) \end{aligned} \tag{11}$$

В загальному випадку похідна буде дорівнювати (12)

$$\frac{\partial p_i}{\partial a_j} = \begin{cases} p_i(1-p_j) & if \quad i=j \\ -p_j . p_i & if \quad i \neq j \end{cases} \tag{12}$$

З загального виду градієнту можна зробити висновок, що похідну функції Softmax дуже просто обчислити, тому її використання є виправданим з огляду на швидкість обчислень.

2.3. Висновки до другого розділу

Розглянуто основні види глибинних нейронних мереж.

На основі проведеного порівняльного аналізу архітектур нейронних мереж показано, що саме згорткові нейронні мережі знайшли широке застосування при обробці зображень. Описані основні елементи згорткової мережі та особливості вибору параметрів для навчання нейронних мереж.

Обґрунтовано використання функції активації Softmax, за допомогою якої відбувається визначення ймовірності того, до якого класу належить об’єкт. Показано, що перевагою її використання є також простота обчислення похідної даної функції, яка необхідна при зворотньому поширенні похибки. Дана функція використовується як останній шар в нейронних мережах і дозволяє підвищити ефективність класифікації об‘єктів на зображенні.

3. РОЗРОБКА СИСТЕМИ НЕЙРОМЕРЕЖЕВОГО РОЗПІЗНАВАННЯ ДЕТАЛЕЙ ОДЯГУ НА ЗОБРАЖЕННІ НА ОСНОВІ ЗГОРТКОВОЇ НЕЙРОННОЇ МЕРЕЖІ

3.1. Нормалізація даних тренувальної множини

Перед тим, як починати навчання нейронної мережі, необхідно визначитись із класами, які будуть використовуватися під час класифікації об’єктів. Для цього необхідно створити структуру класів. У даній роботі створюється нейронна мережа для класифікації атрибутів одягу. Дане завдання потребує класифікації великої кількості різних об’єктів на одному зображенні. Для цього необхідно створити декілька нейронних мереж, які, окрім останнього шару, матимуть однакову архітектуру. Для оптимізації буде використовуватись одна й та сама нейронна мережа за допомогою техніки передачі навчання. Для цього необхідно одного разу навчити нейронну мережу, а навчені ваги повторно використовувати з певними модифікаціями. В даному випадку лише ваги, що використовуються класифікатором, будуть перенавчатись на нових класах, які необхідно навчитись ідентифікувати нейронній мережі.

Атрибути одягу відносяться до сфери моди, та містять великий та складний обсяг інформації. Щоб описати когнітивний процес одягу, необхідно побудувати дерево ієрархічних атрибутів, що і буде структурованою ціллю класифікації, яка може бути широко застосована до пошуку образів одягу, навігації по тегам, рекомендаціям щодо поєднання елементів одягу тощо. На рис. 3.1 зображено ієрархічну модель класів, які необхідно визначити на етапі класифікації нейронній мережі. З зображення видно, що нобхідно створити сім нейронних мереж для класифікації множини ознак одягу: висока горловина, комір, лацкан, довжина рукаву, довжина тілесної частини, довжина спідниці, довжина брюк.

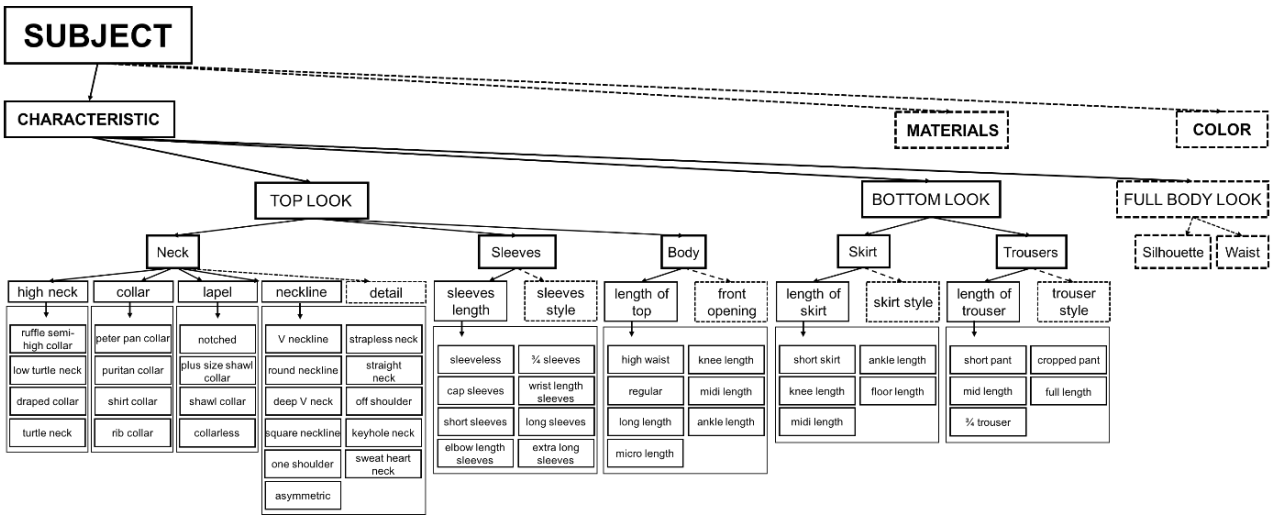

Рисунок 3.1 – Ієрархія класів атрибутів одягу

В даному наборі даних, на якому будуть проводитись тренування, існує ряд важливих характеристик, що дозволять точно класифікувати зображення та уникнути подальших нормалізацій:

а) взаємне виключення: значення атрибутів під конкретним класом атрибутів є взаємовиключними. Наприклад, висока горловина на певному зображенні не може одночасно бути також напіввисокою. Слід зауважити наступне: з огляду на складність проблеми, щоб гарантувати взаємне виключення атрибуту, залишено деякі конкретні зображення, в яких модель носить кілька накладених предметів одягу, і, таким чином, генерує кілька різних атрибутів у одному вимірі;

б) незалежність: атрибути різних класів можуть співіснувати в одному зображенні, незалежно один від одного. Наприклад, висока горловина та комір сорочки можуть співіснувати в одному зображенні.

Структура даних і анотацій тренувальної множини існують за наступною структурою:

- зображення
- анотації
- README.md

a) Папка «Зображення» містить зображення у форматі JPEG. Наприклад, зображення може бути названо як «0000001.jpg».

b) Папка «Анотації» містить характеристики міток атрибутів у форматі

CSV.

c) README.md: містить докладну інформацію до вищезазначених даних.

Структура файлу CSV зображено на табл. 3.1.

Таблиця 3.1 – Структура анотацій

ImageName	AttrKey	AttrValues
0000001.jpg	sleeve_length_labels	nnnnnnnym
0000001.jpg	skirt_length_labels	nynnnn
0000001.jpg	neck_design_labels	nnnyn
0000001.jpg	coat_length_labels	nnynnnnn

ImageName: ім'я зображення, яке відповідає конкретному файлу зображення у папці «Зображення».

AttrKey: ключ атрибута, наприклад, довжина рукава (sleeve_length_labels), довжина брюк (pant_length_labels) тощо.

AttrValues: значення атрибутів, що відповідають атрибутивному розміру в AttrKey. Наприклад, в класі довжини рукава міститься дев’ять значень, а саме: невидимий, без рукавів, короткий рукав, середня довжина, рукав 3/4, рукав до зап'ястя, довгий рукав та наддовгий рукав, якому відповідає анотація «nnnnnnmyn» на малюнку вище. Анотації містять усього дев'ять букв, причому кожну букву можна трактувати наступним чином:

«y» - означає «так», «повинен бути»;

«m» - означає «можливо»;

«n» - означає «ні», «не повинно бути».

Для кожного розміру атрибута у певному зображенні може бути одна і лише одна «y», інші можуть бути «m» або «n».

Всі AttrKeys та відповідні AttrValues вказані в README.md.

3.2. Залишкове навчання

Деградація точності визначення глобального мінімуму функції похибки вказує на те, що не всі системи можуть бути оптимізовані за швидкістю пошуку такого мінімуму. Нейронна мережа навчається за допомогою обчислення значення функції похибки на кожній ітерації та коригування значень ваг кожного шару. Проблема полягає в тому, що при збільшенні кількості шарів, збільшується кількість ітерацій знаходження похідної функції похибки відносно значень кожного шару. Разом із тим, із просуванням в глибину нейронної мережі, значення, яке коригує ваги, постійно зменшується.

Ідея залишкового навчання запозичена із чисельних методів, а саме з методів апроксимації функції. Саме поняття «залишок» – це похибка при визначенні апроксимованої функції.

Нехай необхідно знайти таке значення x, при якому значення функції:

$$f(x) = y \tag{13}$$

Замінивши x наближеним значенням x_0, отримаємо залишок:

$$\Delta = y - f(x_0) \tag{14}$$

Похибка в такому випадку дорівнює:

$$\varepsilon = x - x_0 \tag{15}$$

Так, коли точне значення x невідомо, обчислення похибки неможливе, але можна визначити залишок. Для апроксимації функції $f(x_0)$, що є розв'язком функції $f(x)$:

$$f(x) = g(x) \tag{16}$$

залишком є функція:

$$\Delta = g(x) - f(x_0) \tag{17}$$

У більшості випадків, чим менше залишок, тим апроксимована функція ближче до реального значення, тобто:

$$\left|\frac{f_0(x)-f(x)}{f(x)}\right| << 1 \tag{18}$$

Ефективність використання залишків полягає в тому, що за умови відсутності точних значень x, існує можливість використати залишок як показник відхилення апроксимації від точного значення функції. Простіше знайти різницю функції апроксимації та вхідної функції, що наближається до 0, ніж шукати таке значення x_0, при якому результат апроксимованої функції наближається до значення початкової функції. Такий підхід тісно корелює з ідеєю навчання нейронної мережі, оскільки функція похибки в загальному вигляді є різницею між реальним та обчисленим значеннями функції[8].

Проведені дослідження показують, що неможливо підвищити точність класифікації об’єктів при звичайному збільшенні кількості шарів . Впровадження залишкового навчання забезпечує збільшення кількості шарів нейронної мережі без додавання нових шарів згорток. Це зумовлено використанням вхідних значень попередніх шарів, що мають вже визначену карту ознак. На рис. 3.2 наведено архітектуру залишкового блоку, який використовується у згортковій нейронній мережі для підвищення точності.

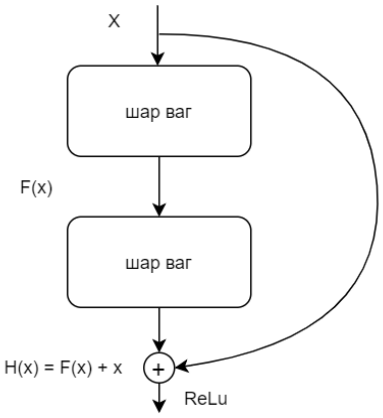

Рисунок 3.2 - Структура залишкового блока

Нехай *H(x)* функція, що визначається після проходження вхідного сигналу *x* через множину сполучених шарів, умовно дорівнює модифікованому сигналу *x*. Існує гіпотеза, що стек нелінійних шарів може асимптотично наближатись до значення складних функцій [4]. Можливо припустити, що такий стек асимптотично наближається до залишкової функції $F(x) = H(x) - x$. Отже, початкову функцію *H(x)* можна записати як $H(x) = F(x) + x$. Значення, що надходять від шарів згорток, використовуються як аргументи функції активації, яка забезпечує нелінійні перетворення скалярного добутку вхідних значень та значень ваг. В згорткових нейронних мережах функція ReLu використовується для того, щоб не враховувати ваги, які не застосовуються у визначенні тієї чи іншої ознаки. Функція ReLu у контексті нейронної мережі є передавальною функцією, що визначається наступним чином:

$$f(x) = \begin{cases} x, \text{якщо } x > 0 \\ 0, \text{якщо } x \leq 0 \end{cases} \quad (19)$$

З функції (19) видно, що коли значення x > 0, то результатом функції буде значення аргументу *x*, в іншому випадку − 0. Отже, в матриці ознак з'являється значна кількість нулів, як наслідок – певна множина ваг у шарі не бере участі у навчанні нейронної мережі. При зворотному

поширенні похибки ваги жодним чином не змінюються, а час пошуку глобального мінімуму функції похибки – збільшується. Для цього, перед тим, як використовувати функцію активації на вихідних значеннях, необхідно додати вхідні значення *x* як шум, що допоможе зберегти якомога більше ваг у збудженому стані та не дозволить втратити ознаку, яку потрібно ідентифікувати.

Блок залишкового навчання можна описати двома функціями y_n та x_{n+1}:

$$y_n = h(x_n) + F(x_n, W_n) \quad (20)$$

$$x_{n+1} = f(y_n), \quad (21)$$

де x_n – вхідні значення до n-го залишкового блоку;

W_n– множина ваг в n-ому шарі;

$F(x_n, W_n)$ – скалярний добуток, наприклад, двох згорткових шарів;

$f(y_n)$ – функція активації ReLu, що застосовується після поелементного додавання вхідного сигналу *x* та вихідного сигналу $F(x_n, W_n)$;

$h(x_n)$ – функція ідентифікації $h(x_n) = x_n$.

Оскільки ReLu є функцією ідентифікації виду $x_{n+1} = y_n$, то рівняння (20) можна записати як:

$$x_{n+1} = x_n + F(x_n, W_n) \quad (22)$$

Якщо записати рівняння (22) рекурсивно, отримаємо рівняння:

$$x_{n+2} = x_{n+1} + F(x_{n+1}, W_{n+1}) = x_n + F(x_n, W_n) + F(x_{n+1}, W_{n+1}) \quad (23)$$

В загальному вигляді маємо:

$$x_N = x_n + \sum_{i=n}^{N-1} F(x_i, W_i), \tag{24}$$

де N – номер шару.

Вхідні параметри x_N на будь-якому рівні можна представити як суму вхідних значень x_n та залишкової функції у вигляді $\sum_{i=n}^{N-1} F(x_i, W_i)$. Отже, можна отримати лінійну залежність між будь-яким значенням x_N та x_n. Це дозволяє визначити необхідні показники при зворотному поширенні похибки:

$$\frac{\partial E}{\partial x_n} = \frac{\partial E}{\partial x_N}\frac{\partial x_N}{\partial x_n} = \frac{\partial E}{\partial x_N}\left(1 + \frac{\partial}{\partial x_n}\sum_{i=n}^{N-1} F(x_i, W_i)\right) = \frac{\partial E}{\partial x_N} + \\ + \frac{\partial E}{\partial x_N}\frac{\partial}{\partial x_n}\sum_{i=n}^{N-1} F(x_i W_i), \tag{25}$$

де E – функція похибки. З рівняння (25) видно, що градієнт функції похибки складається з двох адитивних елементів: перший $\frac{\partial E}{\partial x_N}$, що поширює похибку на вхідний сигнал x_i, не розповсюджуючи її на ваги; та другий $\frac{\partial E}{\partial x_N}\frac{\partial}{\partial x_n}\sum_{i=n}^{N-1} F(x_i W_i)$, який розповсюджує похибку на ваги. Саме перший елемент показує, що похибка розповсюджується на всі вхідні сигнали x_i, незалежно від розташування n-го шару нейронної мережі. Також з рівняння (25) видно, що згасання градієнта $\frac{\partial}{\partial x_n}$ в такому випадку є малоймовірним тому, що загалом $\frac{\partial}{\partial x_n}\sum_{i=n}^{N-1} F(x_i W_i)$ не може дорівнювати мінус 1 для всіх вхідних значень x_i. Отже, градієнт шару не зникає, навіть, коли ваги відносно малі.

Додавання залишкового блоку у структуру нейронної мережі призводить до більш коректної зміни ваг під час зворотного розповсюдження похибки, як наслідок – вектор градієнту матиме найбільш оптимальний напрям для знаходження глобального мінімуму функції похибки [9].

3.3. Визначення оптимальних параметрів та гіперпараметрів нейронної мережі

Визначення параметрів та гіперпараметрів згорткової нейронної мережі є важливою задачею під створення навчальної моделі. В даному випадку необхідно зосередитись на двох основних гіперпараметрах нейронної мережі: кількість шарів та темп навчання.

Під час навчання глибинних згорткових нейронних мереж основною проблемою є проблема згасаючого градієнта. Згасаючий градієнт - виникає при тренуванні певних штучних нейронних мереж із методами пошуку фінкції похибки за допомогою градієнтих методів (наприклад, зворотнє розповсюдження похибки). Зокрема, ця проблема ускладнює вивчення та налаштування параметрів попередніх шарів у мережі. Ця проблема посилюється, коли збільшується кількість шарів в архітектурі.

Це не є проблемою архітектури нейронних мереж - це проблема градієнтних методів навчання, викликана деякими функціями активації. Розповсюджені функції активації, такі як гіперболічний тангенс, мають градієнти в проміжках (-1, 1) або [0, 1), а зворотне поширення похибки розраховується за правилом ланцюга. В результаті відбуваються множення n невеликих чисел під час розрахунків градієнтів для шарів, що розташовані ближче до входу нейронної мережі, а це означає, що градієнт експоненційно спадає з n, і передні шари тренуються дуже повільно.

Методи на основі градієнта вивчають значення параметрів, розуміючи, як невелика зміна значення параметра впливатиме на висновок мережі. Якщо зміна значення параметра викликає дуже невелику зміну вихідної мережі - мережа просто не може ефективно вивчити параметр, що є проблемою.

Це саме те, що відбувається в проблемі згасаючого градієнта - градієнти вихідної мережі для параметрів у ранніх шарах стають

надзвичайно малими. Це чудовий спосіб сказати, що навіть велика зміна значення параметрів для ранніх шарів не впливає на результат.

Проблема згасаючого градієнта залежить від вибору функції активації. Багато звичайних функцій активації (наприклад, сигмоїда або тангенс) стискає вхідні значення в дуже малий діапазон. Наприклад, сигмоїда відображає дійсні числа на малому діапазоні [0,1]. Внаслідок цього існують великі області вхідних значень, які віднесені до надзвичайно малого діапазону. У цих регіонах вхідного простору навіть велика зміна вхідного сигналу призведе до невеликої зміни вихідного сигналу - отже, градієнт невеликий.

Ситуація стає набагато гіршою, коли необхідно скласти кілька шарів таких нелінійностей на вершині один одного. Наприклад, перший шар буде впорядкувати велику область вхідних значень до меншого діапазону вихідних значень, яка буде нанесена на ще менший регіон на другому шарі, який буде перетворений на ще менший регіон третього рівня і так далі. В результаті навіть велика зміна параметрів першого шару не сильно змінює вихід.

Можна уникнути цієї проблеми, використовуючи активаційні функції, які не мають цієї властивості "розбити" вхідний простір у невелику область. Популярним вибором є використання залишкового навчання. Використання залишкового навчання дуже доцільне під час виникнення даної проблеми. Так після прогону на валідаційній множині даних декількох нейронних мереж складено залежність між значеннями більш точного глобального мінімуму та кількістю шарів у нейронній мережі. На рис.3.3 зображено графіки залежності для звичайних нейронних мереж з 20 та 56 шарами, а також для залишкових мереж з 20, 56 та 101 шарами.

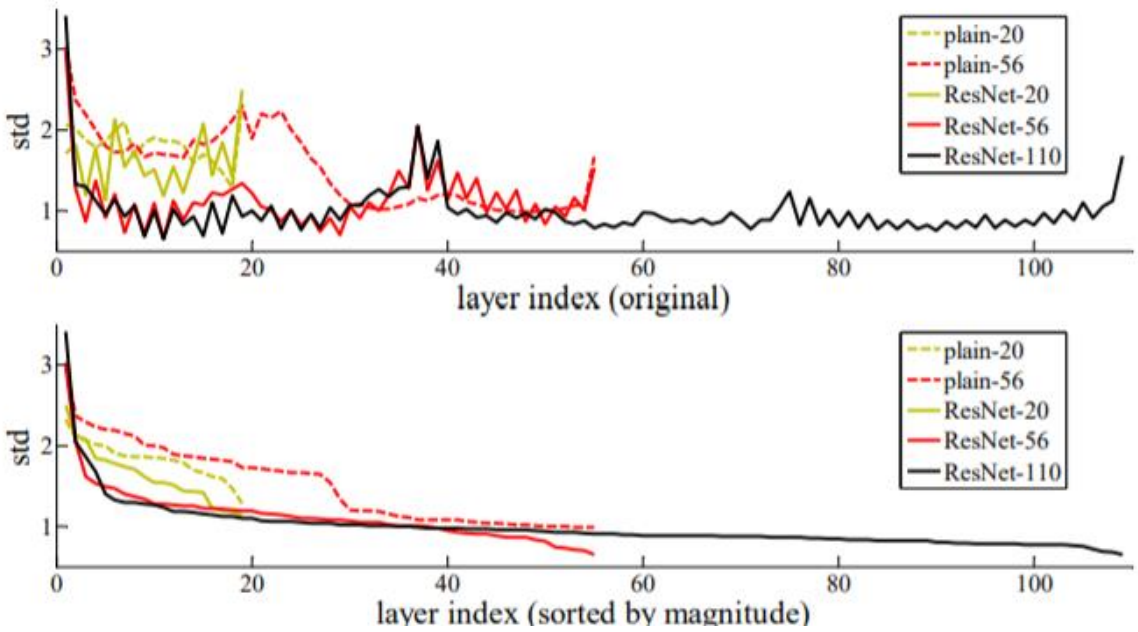

Рисунок 3.3 – Залежність кількості шарів та значення похибки класифікації

З графіку видно, що використання 56 шарів є найоптимальнішим за часом навчання. Тому в рамках даної роботи використовується згорткова нейронна мережа з кількістю шарів 50.

Мережі глибинного навчання, як правило, навчаються оптимізатором стохастичного градієнтного спуску. В даній роботі також використовується стохастичний градієнтний спуск. Даний метод дозволяє встановити швидкість навчання. Цей параметр повідомляє оптимізатору, наскільки далеко потрібно перемістити ваги у напрямку градієнта. Якщо рівень навчання є низьким, то тренування є більш надійними, але оптимізація займе багато часу, тому що кроки до мінімальної функції втрат невеликі. Якщо рівень навчання високий, навчання може не сходитися і навіть не розходитися. Зміни ваги можуть бути настільки великими, що оптимізатор перевищує мінімум і погіршує втрати.

Підготовка повинна починатися з відносно великої швидкості навчання, оскільки на початку випадкові ваги далеко не оптимальні, а

потім навчання може зменшуватися під час тренування, щоб дати більш точне оновлення ваг.

Існує кілька способів вибору гарної відправної точки для вибору темпу навчання. Найкращий підхід полягає в тому, щоб спробувати декілька різних значень і подивитися, який з них дає найкращу мінімізацію втрат, не зменшуючи швидкості тренувань. Можна почати з великого значення, наприклад, 0.1, а потім спробувати експоненціально нижчі значення: 0.01, 0.001 та ін. Коли навчання починається з великим темпом, втрата не поліпшується і, можливо, навіть зростає, поки запускаються перші кілька ітерацій тренування. Під час тренування з меншою швидкістю навчання в деяких випадках значення функції втрат починає зменшуватися в перших кількох ітераціях. Ця швидкість навчання є максимальною, яку можна використати - будь-яке більш високе значення не дозволяє навчанню коректно завершитись. Навіть ця величина занадто висока: вона не буде достатньою для навчання протягом декількох ітерацій, тому що з часом мережа буде вимагати більш точного оновлення ваги. Отже, найбільш вдалий темп навчання, щоб розпочати тренування нейронної мережі, буде, напевно, на 1-2 порядки нижчим. З низьким темпом навчання втрата мінімізується повільно, зі збільшенням темпу навчання сходження до глобального мінімуму функції втрат прискорюється, проте ймовірність пропустити точку глобального мінімуму також збільшується. Потрібно вибрати точку на графіку з найшвидшим зменшенням втрат. У цій роботі функція втрат швидко зменшується, коли темп навчання знаходиться в діапазоні [0.001, 0.01] [10].

На рис 3.4 можна побачити, що при використанні занадто малого значення темпу навчання зходження до глобального мінімуму може тривати дуже довго, в той самий час при використанні великих значень темпу навчання, можливо пропустити глобальний мінімум. Тому варто використовувати на початку навчання більш високе значення і зменшувати його впродовж усього навчання.

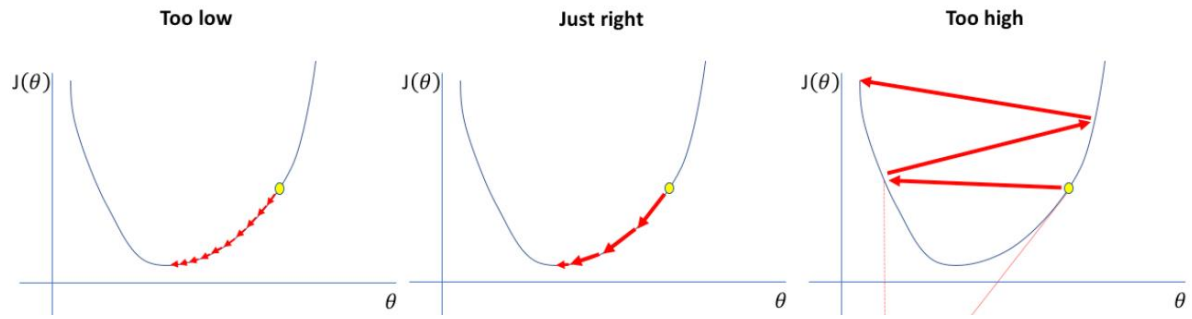

Рисунок 3.4 – Знаходження мінімуму функції з різними темпами навчання

Плато функції втрат нейронної мережі (див. рис. 3.5) є функцією значень параметрів мережі, що кількісно визначають "помилку", пов'язану з використанням певної конфігурації значень параметрів при прогнозуванні на певному наборі даних. Це плато може виглядати зовсім іншим навіть для дуже схожих мережевих архітектур. Нижче наведені зображення, як залишкові підключення в мережі можуть призвести до згладження плато функції втрат.

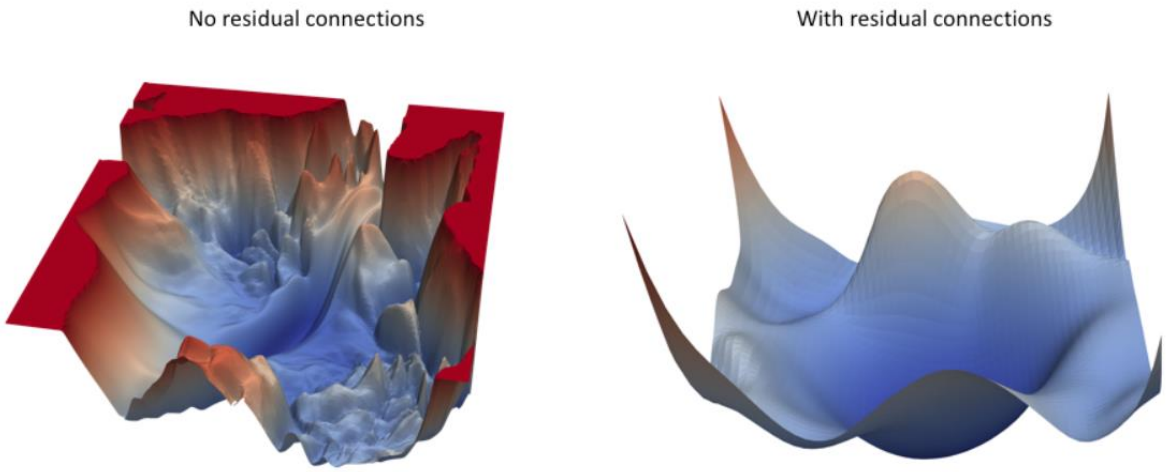

Рисунок 3.5 – Функція похибки без використання залишкових з’єднань (ліворуч) та з використаням залишкових з’єднань (праворуч)

На практиці дуже мало людей навчають згорткову нейронну мережу з нуля (з випадковою ініціалізацією), тому що досить рідко існує набір даних достатнього розміру. Замість цього розповсюдженою практикою є запуск мережі на дуже великому наборі даних (наприклад, ImageNet, що містить 1,2 мільйона зображень з 1000 категоріями), а потім

використовувати натреновану мережу як відправну точку або функцію вилучення ознак для необхідної задачі. Використовуються три основних методи передачі навчання:

• згорткова нейронна мережа для вилучення ознак зображення. Необхідно взяти згорткову мережу, попередньо навчену за допомогою ImageNet, видалити останній повнозв'язний шар класифікатор, а потім використовувати ваги, що залишились для вилучення ознак зображення. В AlexNet таким чином створиться вектор довжиною 4096 для кожного зображення, що містить активацію прихованого шару безпосередньо перед класифікатором. Такі ознаки називаються CNN кодами. Важливо для продуктивності, що на цих кодах можна було застосувати ReLU, якщо на них також була застосована дана функція під час тренування згорткової мережі на ImageNet (як це зазвичай відбувається). Після вилученнях таких векторів для всіх зображень, можна тренувати лінійний класифікатор (наприклад, лінійний SVM або Softmax класифікатор) для нового набору даних.

• точне налаштування згорткової нейронної мережі. Друга стратегія полягає в тому, щоб не тільки замінити та знову натренувати класифікатор на вершині згорткової мережі на новому наборі даних, але також оптимізувати ваги попередньо навченої мережі, продовжуючи використання зворотного поширення похибки. Можна точно налаштувати всі шари згорткової мережі, або можна зберегти деякі попередні шари (через проблеми перенавчання) і лише точно налаштувати частину вищого рівня мережі. Це мотивовано спостереженням, що шари, які знаходяться ближче до входу згорткової мережі містять більше загальних ознак (наприклад, детектори краю або детектори кольорів), які повинні бути корисними для багатьох завдань, але згодом шари згорткової мережі стають поступово більш специфічними для деталей класів що міститься в оригінальному наборі даних. У випадку з прикладом ImageNet, який містить багато собачих порід, значна частина представницької потужності

згорткової мережі може бути присвячена функціям, специфічним для диференціації між породами собак.

• перероблені моделі. Оскільки сучасні згорткові мережі забирають 2-3 тижні для навчання на кількох графічних процесорах за допомогою множини даних ImageNet, загальноприйнятим є те, що люди випускають свої базові версії згорткових мереж на користь тих, хто може використовувати мережі для точного налаштування. Наприклад, у бібліотеки Caffe є форум ModelZoo, де люди діляться вагами мережі.

Вибір типу передачі навчання - це рішення, яке залежить від декількох факторів, але два найбільш важливих - розмір нового набору даних (малий або великий) та його подібність до вихідного набору даних (наприклад, набір подібний до ImageNet з точки зору вмісту зображень і класів, або дуже різні, наприклад зображення мікроскопа). Беручи до уваги, що функції згорктової мережі більш загальні у ранніх шарах і більш специфічні у подальших. Деякі загальні правила для 4 випадків.

1. Новий набір даних невеликий і схожий на оригінальний набір даних. Оскільки дані невеликі, не рекомендується точно налаштовувати згорткову мережу через можливість перенавчання. Оскільки дані схожі з початковими даними, очікується, що ознаки високого рівня абстракції в нейронній мережі будуть також релевантними для даного набору даних. Отже, кращими ідеями може бути підготовка лінійного класифікатора.

2. Новий набір даних великий і схожий на вихідний набір даних. Оскільки існує велика кількість даних, можна мати більшу впевненість у тому, що мережа не перенавчиться, при точному налаштуванні всієї мережі.

3. Новий набір даних невеликий, але дуже відрізняється від оригінального набору даних. Оскільки об’єм даних невеликий, найкраще лише тренувати лінійний класифікатор. Оскільки набори даних дуже відрізняються, можливо, буде не найкращим рішенням тренування лише класифікатора, який містить більше специфічних ознак для початкового

набору даних. Натомість краще, тренувати класифікатор SVM десь раніше в мережі.

4. Новий набір даних великий і дуже відрізняється від оригінального набору даних. Оскільки набір даних дуже великий – це означає, що можна навчити мережу з нуля. Проте на практиці дуже часто все ще вигідно ініціалізувати мережу з вагами з попередньо підготовленої моделі. У цьому випадку буде достатньо даних і впевненості, щодо точного налаштування всієї мережі [11].

Оскільки в даній роботі виконується задача класифікації атрибутів одягу, а дані для всіх мереж будуть дуже схожими використовується 1 випадок, коли необхідно видалити останній шар з вагами класифікатора і тренувати лише його.

3.4. Додавання швидких з’єднань до базової моделі

В ході даної роботи для порівняльної характеристики використовуються дві нейронні мережі:

- звичайна згорткова нейронна мережа з 50 шарами;
- згорткова нейронна мережа з використання залишкового навчання з 50 шарами.

На табл. 3.2 зображено базову структуру для обох мереж. Хоча з точки зору шарової структури мережі ідентичні, головною відмінністю є додавання залишкових блоків. Нейронні мережі складаються зі згорткових шарів різної розмірності. Так в запропонованій структурі перший згортковий шар розмірністю 7×7, шар агрегування 3×3, а надалі використовується комбінація 3 згорткових шарів (1×1, 3×3, 1×1). Дана комбінація є необхідною для застосування залишкового блоку при зміні розмірності вихідного сигналу шару.

Таблиця 3.2 – Структура базової згорткової нейронної мережі

Назва шару	Розмірність матриці на виході	50 шарів
згортка_1	112 × 112	7 × 7 – шар згортки 3 × 3 – шар агрегування
згортка_2	56 × 56	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times3$
згортка_3	28 × 28	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times4$
згортка_4	14 × 14	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times6$
згортка_5	7 × 7	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times3$
повнозв’язний шар	1 × 1	

З використанням вищезазначеної структури мережі, можливе додавання залишкових блоків, які перетворюють звичайну мережу на мережу з залишковим навчанням. Залишкові блоки можуть бути безпосередньо використані, коли вхід і вихід мають однакові розмірності. Коли розмірність наступного шару збільшується можна вирішити дану проблему двома способами:

- залишковий блок все ще виконує поелементне додавання вхідного сигналу з додатковими нулями для півищення розмірності. Даний метод не використовує додаткових параметрів мережі;
- залишковий блок застосовується з використанням згорткового шару з розмірністю 1×1 та страйдом 2 [12].

Для згорткової нейронної мережі з використанням залишкового навчання використовуються блоки які містять 3 згорткових шари. Взагалі немає правил з вибору числа шарів у згортковому блоці, проте точно відомо, що використання одного шару не призводить до покращених значень точності класифікації. На рис. 3.6 зображено модель залишкового блоку, з комбінацією шарів, яка використовується в даній роботі. Таким чином вхідний сигнал передається на 3 шари вперед, а на вхід наступного стеку з 3 шарів передається поелементна сума вхідного та вихідного сигналу згорткового блоку.

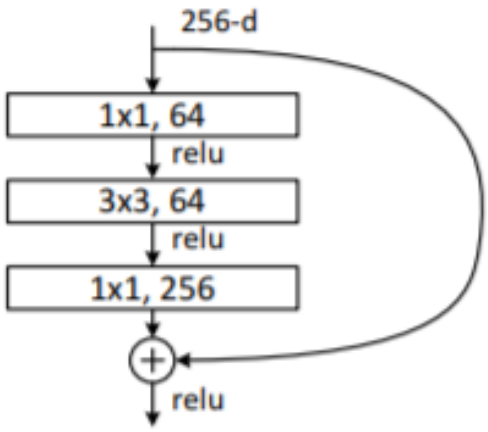

Рисунок 3.6 – Модель згорткового блоку

Для порівняння результатів під час тренування було використано один і той самий набір даних з 55000 різних зображень людей в одязі, на звичайній згортковій нейронній мережі та мережі з залишковим навчання з вищезазначеною структурою. Тренування відбувалося на обчислювальному пристрої з наступними характеристиками:

- 30GB оперативної пам’яті;
- процесор з 8 ядрами та частотою 3.8 ГГц;
- графічна карта Nvidia Quadro P5000 з 16Гб внутрнішньої пам’яті.

Тренування обох мереж без використання методу передачі навчання тривало 20 годин. Тестування нейронних мереж відбувалося на вибірках з 5000 зображень. На табл. 3.3 представлені похибки у 2 категоріях: топ-1 -

клас обрано правильно та топ-5 правильний клас опинився у множині з 5 значень з найвищими показниками ймовірностей.

Таблиця 3.3 – Похибки класифікації зображень звичайної нейронної мережі та мережі з залишковим навчанням

Тип мережі	Топ-1, %	Топ-5, %
Базова нейронна мережа	29.37	10.38
Нейронна мережа з залишковим навчанням	20.86	5.34

З табл. 3.3 видно, що точність класифікації зображень при використанні згорткової нейронної мережі з залишковим навчанням підвищилась на 8 відсотків у порівнянні зі звичайною мережею. Дані нейронні мережі класифікували зображення за довжиною штанів.

Після навчання першої мережі використано метод передачі навчання, для більш швидкого навчання 6 інших мереж для класифікації наступних атрибутів одягу:

- висока горловина
- комір
- лацкан
- довжина рукаву
- довжина тілесної частини
- довжина спідниці

На таблиці 3.4 та 3.5 представлено результати навчання усіх 7 мереж з використанням методу залишкового навчання.

Таблиця 3.4 - Похибки класифікації зображень

Тип атрибуту	Висока горловина	Комір	Лацкан	Довжина рукаву
Топ-1, %	25.38	26.83	25.87	22.41

Топ-5, %	7.53	8.51	6.85	6.21

Таблиця 3.5 - Похибки класифікації зображень

Тип атрибуту	Довжина тілесної частини	Довжина спідниці	Довжина штанів
Топ-1, %	21.71	20.26	20.86
Топ-5, %	5.93	4.81	5.34

З результатів навчання нейронної мережі видно, що точність класифікації атрибутів одягу, які є малими за розмірами – менша, адже для таких атрибутів важче вилучити ознаки необхідні для класифікації.

Результатом навчання нейронної мережі є класифікація атрибутів одягу, тобто ймовірність належності до того чи іншого класу. Проте з використанням бібліотеки OpenCV можливо накласти мапу пікселів, які є визначальними для класифікації атрибутів. Так на рис. 3.6 можна побачити пікселі, які були пріорітетними при вилученні ознак. З рисунку видно, що нейронна мережа досить чітко ідентифікує межі одягу та окремих атрибутів, що і є задачею, яку необхідно було вирішити.

Рисунок 3.6 – Результати роботи нейронної мережі

3.5. Висновки до третього розділу

Розроблено систему нейромережевого розпізнавання деталей на основі згорткових нейронних мереж та реалізовано структуру моделі згорткової мережі з залишковим навчання. Для навчання первинної згорткової мережі виконана нормалізація даних тренувальної множини, створена ієрархія класів деталей одягу, введена структура залишкового блоку. Показано, що ефективність використання залишків полягає в тому, що за умови відсутності точних значень *x*, існує можливість використати залишку як показника відхилення апроксимації від точного значення функції.

Розроблено 7 нейронних мереж для класифікації атрибутів одягу. Це необхідно для вирішення проблеми згасаючого градієнта, яка виникає при тренуванні певних штучних нейронних мереж з використанням методів пошуку фінкції похибки за допомогою градієнтих методів (наприклад, зворотнє розповсюдження похибки). Показано, що ця проблема ускладнює вивчення та налаштування параметрів попередніх шарів у мережі та посилюється при збільшенні кількісті шарів в архітектурі.

Доведено, що для більш точного налаштування системи перед початком тренування згорткової нейронної мережі необхідно обрати гіперпараметри мережі. В даній роботі обрано згорткову мережу з 50 шарами, а темп навчання стохастичного градієнтого спуску ініціалізується зі значення 0.01 та під час навчання зменшується до 0.001. Це призводить до зменшення часу навчання та дозволяє не пропустити точку глобального мінімум функції похибки.

Тестування розробленої системи та її порівняння з базовою зготковою нейронною мережею показало покращення результатів на 8% .

ВИСНОВКИ

У даній роботі було розроблено системи нейромережевого розпізнавання деталей одягу на основі згорткової нейронної мережі з залишковим навчанням. Введення в структуру мережі блоку залишкового навчання забезпечило підвищення точності класифікації деталей одягу при розпізнаванні об‘єктів.

Проведений детальний огляд методів вирішення задач комп‘ютерного бачення показав, що для створення системи розпізнавання деталей одягу доцільно обрати нейромережевий метод.

Проаналізовано принципи організації згорткових нейронних мереж та основні методи навчання таких мереж. Розглянуті принципи організації навчання нейронних мереж показали, що на сьогодні залишаються невирішеними питання ефективного навчання нейронних структур для підвищення точності класифікації об‘єктів.

На основі проведеного порівняльного аналізу архітектур нейронних мереж показано, що саме згорткові нейронні мережі знайшли широке застосування при обробці зображень. Описані основні елементи згорткової мережі та особливості вибору параметрів для навчання нейронних мереж.

Обґрунтовано використання функції активації Softmax, за допомогою якої відбувається визначення ймовірності того, до якого класу належить об’єкт. Дана функція використовується як останній шар в таких нейронних мережах і дозволяє підвищити ефективність класифікації об‘єктів на зображенні.

Розроблено 7 нейронних мереж для класифікації атрибутів одягу. Це необхідно для вирішення проблеми згасаючого градієнта, яка виникає при тренуванні певних штучних нейронних мереж з використанням методів пошуку функції похибки за допомогою градієнтих методів (наприклад, зворотнє розповсюдження похибки). Показано, що ця

проблема ускладнює вивчення та налаштування параметрів попередніх шарів у мережі та посилюється при збільшенні кількісті шарів в архітектурі.

Доведено, що для більш точного налаштування системи перед початком тренування згорткової нейронної мережі необхідно обрати гіперпараметри мережі. В даній роботі обрано згорткову мережу з 50 шарами, а темп навчання стохастичного градієнтого спуску ініціалізується зі значення 0.01 та під час навчання зменшується до 0.001. Це призводить до зменшення часу навчання та дозволяє не пропустити точку глобального мінімум функції похибки.

Важливою частиною даного дослідження є використання методу передачі навчання. Так після навчання однієї згорткової нейронної мережі було створено 6 інших нейронних мереж на основі вже визначених ваг нейронної мережі для розпізнавання різних атрибутів одягу.

Тестування розробленої системи та її порівняння з базовою зготковою нейронною мережею показало покращення результатів на 8%.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Szegedy C. Deep neural networks for object detection / C. Szegedy, A. Toshev, D. Erhan. // NIPS. – 2013.
2. Hoai M. Regularized max pooling for image categorization / Hoai. // BMVC. – 2014.
3. Girshick R. Fast R-CNN / Girshick. // ICCV. – 2015.
4. Krizhevsky A. Imagenet classification with deep convolutional neural networks / A. Krizhevsky, I. Sutskever, G. E. Hinton. // Advances in neural information processing systems. – 2012.
5. Vedaldi A. Convolutional neural networks for matlab / A. Vedaldi, K. Lenc. // Proceedings of the 23rd Annual ACM Conference on Multimedia Conference. – 2015.
6. Dai J. Convolutional feature masking for joint object and stuff segmentation / J. Dai, K. He, J. Sun. // CVPR. – 2015.
7. Szegedy C. Going deeper with convolutions / C. Szegedy, W. Liu, Y. Jia. // CVPR. – 2015.
8. Long J. Fully convolutional networks for semantic segmentation / J. Long, E. Shelhamer, T. Darrell. // CoRR. – 2014.
9. Використання залишкового навчання у згорткових нейронних мережах : матеріали 11-ої міжнар. наук.-техн. конф. ПМК’2018-2, 14-16 лист. 2018, Київ, Україна / НТУУ «КПІ», Ф-т прикладної матем. – Л., 2018.
10. Gunji N. Classifier entry / N. Gunji, T. Higuchi, K. Yasumoto. // ILSVRC. – 2012.
11. Zeiler M. Visualizing and Understanding Convolutional Networks / M. Zeiler, R. Fergus. // ICCV. – 2015.
12. Wei Y. CNN: Single-label to multi-label / Y. Wei, W. Xia, J. Huang. // CoRR. – 2014.