Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Перетворення стилю 3D текстур за допомогою згорткових нейронних мереж
(тема)

Виконав:
студент 2 курсу, групи КІТм-21-2
Кузьменко В.О.
(прізвище, ініціали)
Спеціальність 123 Комп'ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп'ютерні інтелектуальні технології
Керівник проф. Безсонов О.О.
(посада, ініціали, прізвище)

Допускається до захисту _____
(підпис)

Зав. кафедри _____ О.Г. Руденко
(підпис)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем
Рівень вищої освіти другий (магістерський)
Спеціальність 123 Комп'ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп'ютерні інтелектуальні технології

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____202_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові
___Кузьменку Вячеславу Олександровичу_____

1. Тема роботи (проекту)_Перетворення стилю 3D текстур за допомогою згорткових нейронних мереж_____
_____
затверджена наказом університету від "___"_____202_р. № _____
2. Термін подання студентом роботи до екзаменаційної комісії ___ _____202_р.
3. Вихідні дані до роботи (проекту)_____
__Нейронні мережі_____
__3D текстури_____

4. Перелік питань, що потрібно опрацювати в роботі _____
__Аналіз поняття текстури та штучної нейронної мережі_____
__Згорткові нейронні мережі_____
Алгоритм перетворення стилю 3D тестур за допомогою згорткових нейронних
__мереж_____

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням кафедри)

___11_слайдів_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявністю консультантів згідно до наказу, зазначеному у п.1)

Найменування розділу	Консультант посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

1 КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	7.11.2022	
2	Аналіз предметної області	8.11.2022-15.11.2022	
3	Аналіз джерел з проблемної галузі	15.11.2022-26.11.2022	
4	Розробка алгоритму перетворення стию 3D текстур	26.11.2022-13.12.2022	
5	Оформлення пояснювальної записки	13.12.2022-18.12.2022	
6	Оформлення графічного матеріалу	18.12.2022-19.12.2022	
7	Перевірка виконаного проекту керівником	19.12.2022	
8	Захист роботи	22.12.2022	

Дата видачі завдання _7_ _листопада____2022 р.

Студент _____

(підпис)

Керівник роботи проф. каф КІТС О.О. Безсонов

(підпис) (посада, ініціали,прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 66., 21 рис., 1 табл., 20 джерел.

КОМП’ЮТЕРНЕ БАЧЕННЯ, ТЕКСТУРА, МЕРЕЖА, МОДЕЛЬ, АЛГОРИТМ, НАВЧАННЯ, ГЕНЕРАЦІЯ

Метою кваліфікаційної роботи є вивчення методів комп’ютерного зору та методів машинного навчання для розробки моделі, що виконує перетворення 3D текстур.

Результатом роботи стала розроблена модель алгоритму перетворення стилю 3D текстур на основі згорткових нейронних мереж.

ABSTRACT

Explanatory note of the qualification work: 66 pages, 21 figures, 1 table, 20 sources.

COMPUTER VISION, TEXTURE, NETWORK, MODEL, ALGORITHM, LEARNING, GENERATION

The purpose of the master's thesis is to study computer vision methods and machine learning methods for developing a model that performs 3D texture transformation.

The result of the work was a developed model of the algorithm for transforming the style of 3D textures based on convolutional neural networks.

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем

АНОТАЦІЯ

КВАЛІФІКАЦІЙНОЇ РОБОТИ

рівень вищої освіти другий (магістерський)

Перетворення стилю 3D текстур за допомогою згорткових нейронних мереж
(тема)

Виконав:
студент 2 курсу, групи КІТм-21-2
_____Кузьменко В.О._____
(прізвище, ініціали)
Спеціальність 123 Комп'ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп'ютерні інтелектуальні технології
Керівник _____проф. Безсонов О.О._____
(посада, ініціали, прізвище)

2022 р.

АНОТАЦІЯ

Кузьменко В.О.. Перетворення стилю 3D текстур за допомогою згорткових нейронних мереж. – Магістерська кваліфікаційна робота.

У магістерській кваліфікаційній роботі вирішено актуальну задачу перетворення стилю 3D текстур з використанням згорткових нейронних мереж.

Метою кваліфікаційної роботи є вивчення методів комп'ютерного зору та методів машинного навчання для розробки моделі, що виконує перетворення 3D текстур.

Об'єктом дослідження цієї роботи є згорткова нейронна мережа.

Предмет дослідження методи комп'ютерного зору.

У першому розділі представлені основні поняття текстури та штучних нейронних мереж. Були розглянуті карти текстур, колірні дані RGB, алгоритм растеризації, афінне відображення текстури. В процесі виконання роботи було виведено значення штучних нейронних мереж: штучна нейронна мережа — це взаємозв'язана мережа вузлів, уподібнена до безкрайої мережі нейронів у головному мозку.

У другому розділі – розглянуто поняття згорткової нейронної мережі — це клас глибинних штучних нейронних мереж прямого поширення, який успішно застосовувався до аналізу візуальних зображень. Було визначено види шарів таких мереж, розглянуто архітектуру прямого поширення та модель згорткового та арегувального шару, а також термін регуляризації та виведено області застосування згорткових нейронних мереж.

В розділі третьому розділі кваліфікаційної роботи було виведено алгоритм перетворення стилю 3D текстур за допомогою згорткових нейронних мереж. Виведено модель згорткової нейронної мережі та описано

процес перетворення стилю текстур, а також результати трансформації 3D текстур.

За результатами роботи зроблено висновки та було розглянуто методи комп’ютерного зору та методів машинного навчання для розробки моделі, що виконує перетворення 3D текстур.

КОМП’ЮТЕРНЕ БАЧЕННЯ, ТЕКСТУРА, МЕРЕЖА, МОДЕЛЬ, АЛГОРИТМ, НАВЧАННЯ, ГЕНЕРАЦІЯ

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API — Інтерфейс прикладного програмування

GPU — Graphics processing unit (графічний процесор)

RGB — Кольоровий простір (red,green,blue)

RGBA — Адаптивна колірна модель

MNIST — Об'ємна база даних зразків рукописного написання цифр

VGG — Згорткова нейронна мережа(Visual Geometry Group)

БШП — Багатошаровий перцептрон

ЗНМ — Згорткова нейронна мережа

РНМ — Рекурентна нейронна мережа

ШНМ — Штучна нейронна мережа

ВСТУП

Сьогодні комп’ютерне бачення, розпізнавання образів, машинне навчання, а також прогнозування та аналіз даних є одними з найважливіших напрямків досліджень і розробок у сучасній прикладній математиці та кібернетиці. Прискорення темпів розвитку технологій інформаційного суспільства, розвиток концепцій «розумного дому» та «розумного міста», розвиток Інтернету та систем штучного інтелекту визначають особливе місце для цих сфер сучасного світу. У багатьох задачах прикладного програмування використовуються методи збору даних, кластеризації та класифікації та статистичного висновку. Технології розпізнавання образів активно впроваджуються в повсякденне життя. Раніше завдання розпізнавання, які вважалися складними, тепер вирішуються за допомогою звичайних мобільних пристроїв і смартфонів.

Загальними методами та підходами до вирішення завдань виявлення, розпізнавання та класифікації є:

1) порівняння з вибіркою - класифікація за найближчим середнім, за відстанню до найближчого сусіда. До групи порівняння зі зразком можна віднести і структурні методи розпізнавання.

2) зіставлення з шаблоном - метод розпізнавання, який використовує маленьке зображення або шаблон для пошуку областей у збільшеному зображенні.

3) нейронні мережі - клас методів глибокого навчання, що використовуються для автоматичного вивчення властивостей об’єкта та його подальшої ідентифікації. Відмінною рисою цих методів від інших є здатність до навчання.

Метою роботи є вивчення методів комп’ютерного зору та методів машинного навчання для розробки моделі, що виконує перетворення 3D текстур.

Для досягнення поставленої мети в роботі були поставлені та вирішені такі завдання:

1) провести огляд методів розпізнавання об’єктів, комп’ютерного зору та машинного навчання;

2) обрати методи дослідження та існуючі алгоритми виявлення та класифікації об’єктів, аналіз існуючих методів опису та визначення текстур і текстонів зображень;

3) представити алгоритм за допомогою якого буде відбуватися перетворення 3D текстур.

1 АНАЛІЗ ПОНЯТТЯ ТЕКСТУРИ ТА ШТУЧНОЇ НЕЙРОННОЇ МЕРЕЖІ

1.1 Загальні властивості текстур

Відображення текстури — це метод відображення текстури на створеній комп’ютером графіці. Текстура тут може бути високочастотною деталлю, текстурою поверхні або кольором.

Оригінальна техніка була впроваджена Едвіном Кетмуллом у 1974 році. Відображення текстур спочатку називалося дифузним відображенням, методом, який просто відображав пікселі з текстури на 3D-поверхню («обгортаючи» зображення навколо об’єкта). В останні десятиліття поява багатопрохідного рендерингу, мультитекстурування, mipmaps і більш складних відображень, таких як відображення висоти, відображення рельєфу, відображення нормалей, відображення зміщення, відображення, дзеркальне відображення, відображення оклюзії та багато інших варіацій техніки (керовані системою матеріалів) зробили можливим імітацію майже фотореалізму в режимі реального часу шляхом значного зменшення кількості багатокутників і розрахунків освітлення, необхідних для створення реалістичної та функціональної 3D-сцени.

1.1.1 Текстурні карти

Карта текстури — це зображення, нанесене на поверхню фігури або багатокутника. Це може бути растрове зображення або процедурна текстура. Вони можуть зберігатися у звичайних форматах файлів зображень, на які

посилаються формати 3d-моделей або визначення матеріалів, і об’єднуються в пакети ресурсів.

Вони можуть мати 1-3 виміри, хоча 2 виміри найчастіше зустрічаються для видимих поверхонь. Для використання з сучасним апаратним забезпеченням дані карти текстури можуть зберігатися в розрізненому або мозаичному порядку для покращення когерентності кешу. API візуалізації зазвичай керують ресурсами карти текстури (які можуть бути розташовані в пам’яті пристрою) як буферами або поверхнями, і можуть дозволяти «рендеринг до текстури» для додаткових ефектів, таких як постобробка або відображення середовища.

Зазвичай вони містять колірні дані RGB (збережені як прямі кольори, стислі формати або індексовані кольори), а іноді й додатковий канал для альфа-змішування (RGBA). Можна використовувати альфа-канал (який може бути зручним для зберігання у форматах, аналізованих апаратним забезпеченням) для інших цілей, наприклад для відображення.

Кілька текстурних карт (або каналів) можна комбінувати для контролю дзеркальності, нормалей, зміщення або підповерхневого розсіювання, наприклад для візуалізації шкіри.

Кілька зображень текстури можна об’єднати в атласи текстур або масив текстур, щоб зменшити зміни стану для сучасного обладнання (їх можна вважати сучасною еволюцією графіки мозаїчних карт). Сучасне обладнання часто підтримує текстури кубічної карти з кількома гранями для відображення середовища.

Комп’ютер сприймає будь яке зображення як масив чисел (рис. 1.1). Матриця праворуч містить числа від 0 до 255, кожне з яких відповідає яскравості пікселя на лівому зображенні. Обидва накладаються на середнє зображення.

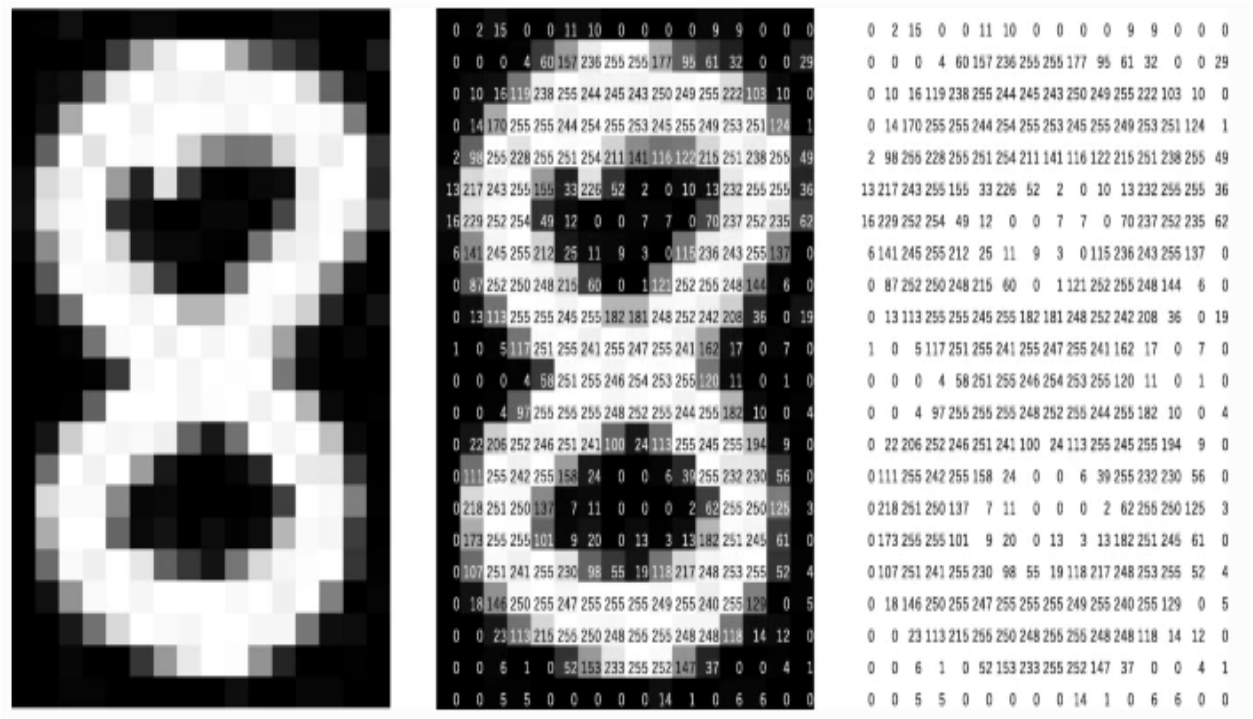

Рисунок 1.1 — Зображення в представленні масиву чисел

1.1.2 Алгоритми растеризації

У реалізації програмного та апаратного забезпечення розвинулися різні техніки. Кожен пропонує різні компроміси в точності, універсальності та продуктивності.

Деякі апаратні системи, напр. Sega Saturn і NV1 напряму перетинають текстурні координати, інтерполюючи спроектовану позицію в просторі екрана через простір текстур і розміщуючи текселі в буфер кадру (у випадку NV1 була використана квадратична інтерполяція, що дозволяло рендеринг). Sega надала інструменти для випікання відповідних плиток текстури для кожного квадрата з моделей, нанесених на UV-карту.

Це має перевагу в тому, що карти текстур читаються простим лінійним способом. Пряме відображення текстури також іноді може давати більш природні результати, ніж афінне відображення текстури, якщо примітиви

вирівнюються з помітними напрямками текстури (наприклад, дорожня розмітка або шари цегли). Це забезпечує обмежену форму виправлення перспективи. Однак спотворення перспективи все ще видно для примітивів поблизу камери (наприклад, порт Saturn Sega Rally демонстрував артефакти здавлювання текстури, оскільки сусідні багатокутники були майже обрізані без UV-координат).

Ця техніка не використовується в сучасному обладнанні, оскільки UV-координати виявилися більш універсальними для моделювання та більш узгодженими для відсікання.

1.1.3 Зворотне відображення текстури

У більшості підходів використовується інверсне відображення текстури, яке перетинає примітиви візуалізації в просторі екрана, одночасно інтерполюючи координати текстури для вибірки. Ця інтерполяція може бути афінною або перспективно правильною. Однією з переваг є те, що кожен вихідний піксель гарантується лише один раз; зазвичай вихідні дані карти текстури зберігаються в меншій бітовій або стиснутій формі, тоді як кадровий буфер використовує вищу бітову глибину. Інша — більша універсальність для UV-картографії. Кеш текстур стає важливим для буферизації читань, оскільки шаблон доступу до пам’яті в просторі текстур є більш складним.

Афінне відображення текстури лінійно інтерполює координати текстури по всій поверхні, тому це найшвидша форма відображення текстури. Деяке програмне та апаратне забезпечення (наприклад, оригінальна PlayStation) проектує вершини в 3D-просторі на екран під час візуалізації та лінійно інтерполює координати текстури в екранному просторі між ними («зворотне відображення текстури»). Це можна зробити за допомогою

збільшення UV-координат фіксованої точки або за допомогою алгоритму збільшення помилок, подібного до алгоритму ліній Брезенхема.

На відміну від перпендикулярних багатокутників, це призводить до помітних спотворень під час трансформації перспективи, особливо як примітиви поблизу камери (рис 1.2).

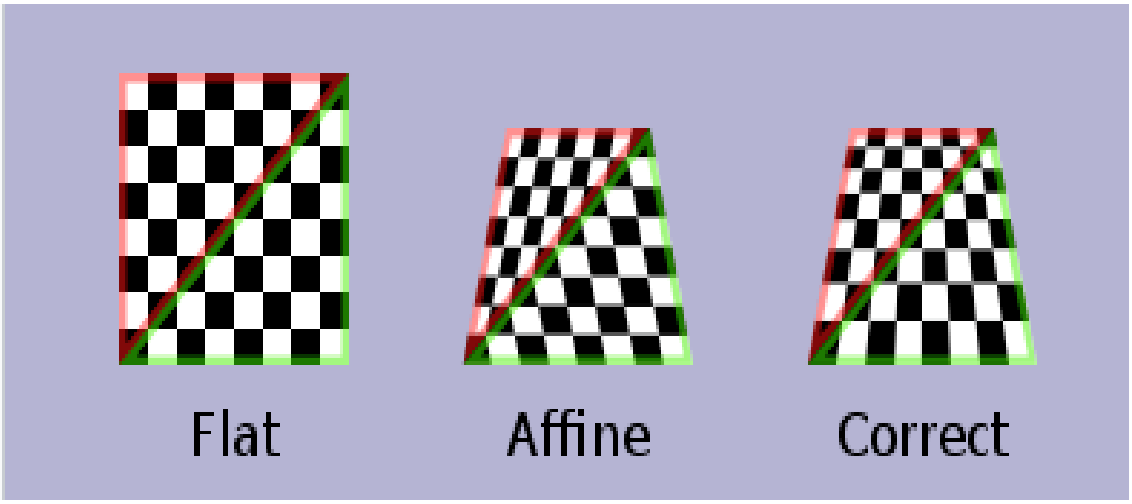

Рисунок 1.2 — Дефекти афінного відображення текстури

Правильне текстурування перспективи враховує положення вершин у 3D-просторі, а не просто інтерполяцію координат у 2D-просторі екрана (рис. 1.3) витрачає більше ресурсів [11].

Рисунок 1.3 — Механізм рендеру віртуального простору в грі Doom

Щоб виконати перспективну корекцію координат текстури u та v, де z є компонентом глибини з точки зору глядача, ми можемо скористатися тим фактом, що значення $\frac{1}{z}$, $\frac{1}{z}$ і $\frac{u}{z}$ є лінійними в просторі екрана по всій текстурованій поверхні. Навпаки, оригінальні z, u та v перед поділом не є лінійними по всій поверхні екрана. Таким чином, ми можемо лінійно інтерполювати ці зворотні величини по всій поверхні, обчислюючи виправлені значення для кожного пікселя, щоб отримати правильне відображення текстури в перспективі.

Для цього ми спочатку обчислюємо зворотні величини в кожній вершині нашої геометрії (3 точки для трикутника). Потім ми лінійно інтерполюємо ці зворотні величини між n вершинами (наприклад, використовуючи барицентричні координати), що призводить до інтерпольованих значень по всій поверхні.

Щоб повернутися до u, v простору, ми спочатку обчислюємо:

$$u_{correct} = \frac{1}{zReciprocal} = \frac{1}{\frac{1}{z}}, \quad (1.1)$$

де $u_{correct}$ - корекція,

zReciprocal - значення взаємної координати.

Ця корекція робить так, що в частинах багатокутника, які знаходяться ближче до глядача, різниця від пікселя до пікселя між координатами текстури менша (розтягування текстури ширше), а в частинах, розташованих далі, ця різниця більша (стиснення текстури).

Афінне відображення текстури безпосередньо інтерполює координату текстури між двома кінцевими точками u_1 та u_0:

$$□_□ = (1 - □)□_0 + □□_1, при\ 0 \leq □ \leq 1, \quad (1.2)$$

де $□_0$, $□_1$ - кінцеві точки,

□ - коефіцієнт.

Правильне відображення перспективи інтерполюється після ділення на глибину, а потім використовує свою інтерпольовану зворотну величину для відновлення правильної координати:

$$□_□ = \frac{(1 - □)\frac{□0}{□0}}{(1 - □)\frac{1}{□0}} \quad (1.3)$$

де $□_0$, $□_□$ - координати,

□ - коефіцієнт.

Апаратне забезпечення тривимірної графіки зазвичай підтримує правильне текстурування перспективи.

Розвинулися різні техніки для рендерингу геометрії, нанесеної на текстуру, у зображення з різними компромісами між якістю та точністю, які можна застосувати як до програмного, так і до апаратного забезпечення.

Класичне програмне забезпечення для відображення текстури зазвичай робило лише просте відображення з максимум одним світловим ефектом (зазвичай застосовуваним через таблицю пошуку), і правильність перспективи була приблизно в 16 разів дорожчою.

Двигун Doom обмежив світ вертикальними стінами та горизонтальними підлогою/стелею з камерою, яка могла обертатися лише навколо вертикальної осі. Це означало, що стіни матимуть постійну координату глибини вздовж вертикальної лінії, а підлоги/стелі матимуть постійну глибину вздовж горизонтальної лінії. Швидке афінне відображення можна використовувати в цьому напрямку, оскільки це було б правильним. Деякі пізніші рендерери цієї епохи симулювали невеликий крок камери зі

зсувом, що створювало видимість більшої свободи під час використання тієї самої техніки рендерингу.

Деякі движки змогли відтворити карти висот із накладенням текстури (наприклад, Voxel Space від Nova Logic і Outcast) за допомогою інкрементальних алгоритмів, подібних до Bresenham, створюючи вигляд ландшафту з накладенням текстури без використання традиційних геометричних примітивів [13].

Програмні рендерери зазвичай віддають перевагу поділенню екрана, оскільки це має менше накладних витрат. Крім того, вони намагаються виконати лінійну інтерполяцію вздовж лінії пікселів, щоб спростити налаштування (порівняно з двовимірною афінною інтерполяцією) і, таким чином, знову ж таки накладні витрати (також афінне відображення текстур не вписується в низьку кількість регістрів процесора x86). ; 68000 або будь-який RISC набагато більше підходить).

Для Quake був застосований інший підхід, який обчислював правильні координати перспективи лише один раз на кожні 16 пікселів рядка сканування та лінійно інтерполював між ними, фактично працюючи зі швидкістю лінійної інтерполяції, оскільки правильні обчислення перспективи виконуються паралельно на співпроцесорі [14].

Ще один метод полягав у наближенні перспективи за допомогою швидшого обчислення, такого як поліном. Ще один метод використовує значення 1/z двох останніх намальованих пікселів для лінійної екстраполяції наступного значення. Потім ділення виконується, починаючи з цих значень, так що потрібно розділити лише невеликий залишок [15], але обсяг бухгалтерського обліку робить цей метод надто повільним у більшості систем.

Нарешті, двигун Build розширив трюк постійної відстані, який використовувався для Doom, шляхом знаходження лінії постійної відстані для довільних багатокутників і візуалізації вздовж неї.

Обладнання для відображення текстур спочатку було розроблено для моделювання (наприклад, як реалізовано в генераторах зображень Evans і Sutherland ESIG), професійних графічних робочих станцій, таких як Silicon Graphics, машин для трансляції цифрових відеоефектів, таких як Ampex ADO, а пізніше з'явилося в шафах Arcade, використовувалися для ігрових консолей та відеокарт для ПК у середині 1990-х років. У моделюванні польоту відображення текстур забезпечувало важливі сигнали руху.

API (GPU) забезпечують спеціалізовані фіксовані функціональні блоки, які називаються семплерами текстур або блоками відображення текстури, для виконання відображення текстури, як правило, з трилінійною фільтрацією або кращою анізотропною фільтрацією з кількома натисканнями та обладнанням для декодування певних форматів, таких як DXTn. Станом на 2016 рік апаратне забезпечення відображення текстур є поширеним, оскільки більшість SOC містять відповідний GPU.

Деякі апаратні засоби поєднують відображення текстури з визначенням прихованої поверхні у відкладеному рендерингу на основі мозаїки або рендерингу розгортки; такі системи отримують лише видимі пікселі за рахунок використання більшого робочого простору для трансформованих вершин. Більшість систем зупинилися на підході Z-буферизації, який усе ще може зменшити робоче навантаження на відображення текстур за допомогою сортування спереду назад (рис .1.4).

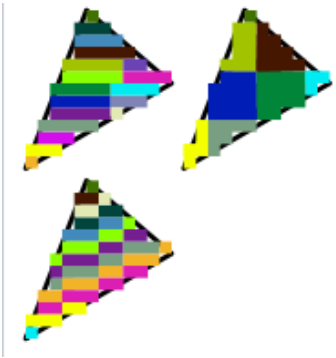

Рисунок 1.4 — Методи поділу екранного простору

1.2 Структура нейронних мереж

Термін «штучна нейронна мережа» походить від біологічної нейронної мережі, яка структуру людського мозку. Подібно до людського мозку з взаємопов'язаними нейронами, штучні нейронні мережі також мають нейрони, пов'язані між собою на різних рівнях мережі. Ці нейрони називають вузлами.

Дендрити біологічної нейронної мережі являють собою вхідні дані штучної нейронної мережі, ядро — вузол, синапс — вагу, а аксон — вихід (рис. 1.5).

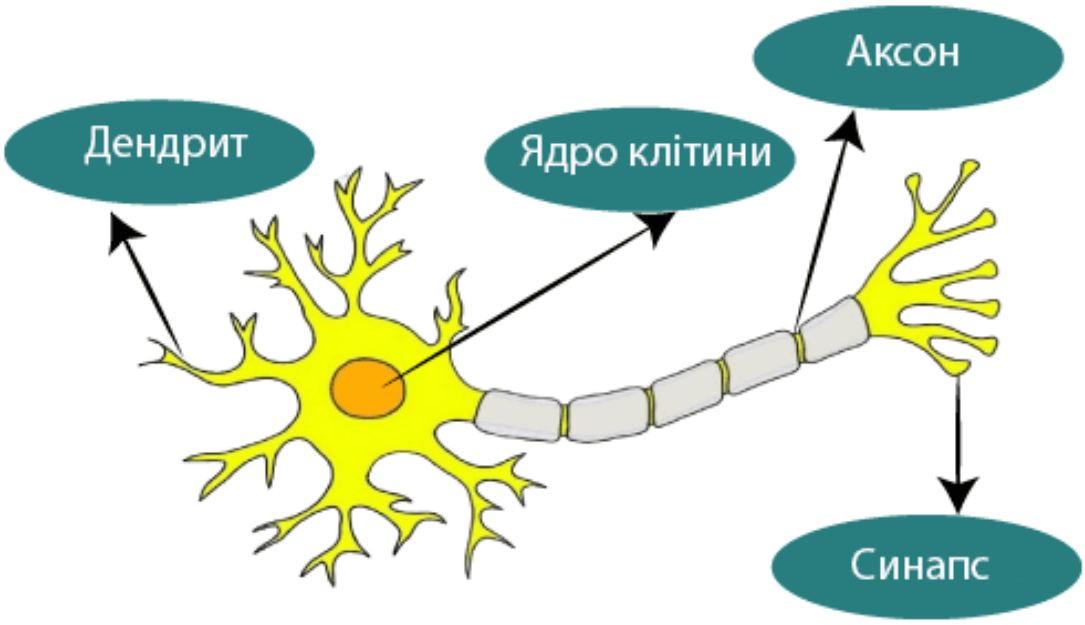

Рисунок 1.5 — Схема біологічної нейронної мережі.

Зв’язок між біологічною нейронною мережею та штучною нейронною мережею показано на таблиці 1.1.

Таблиця 1.1 — Порівняння біологічною та штучною нейронною мережою

Біологічна нейронна мережа	Штучна нейронна мережа
дендрит	вхідні дані
ядро клітини	вузли
синапс	ваги
аксон	вихід

Штучні нейронні мережі у сфері штучного інтелекту намагаються імітувати мережу нейронів для створення людського мозку (рис. 1.6), що дозволяє комп’ютерам розуміти речі та приймати рішення, як люди. Штучні нейронні мережі розроблені шляхом програмування комп’ютерів, щоб поводитися так само просто, як взаємопов’язані клітини мозку.

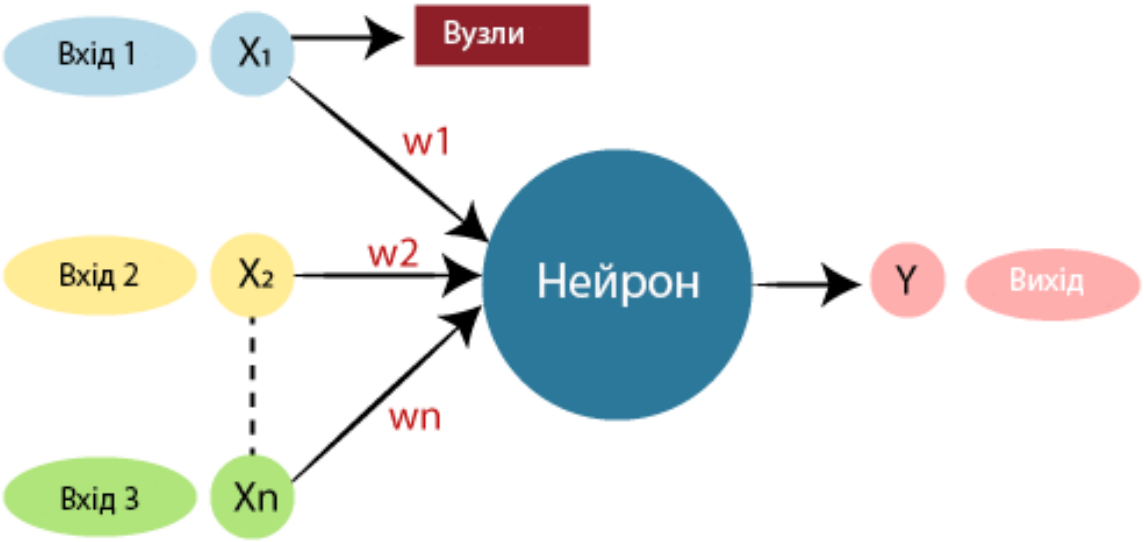

Рисунок 1.6 — Вигляд типової штучної нейронної мережі

У мозку людини приблизно 1000 мільярдів нейронів. Кожен нейрон має асоціацію в діапазоні від 1000 до 100 000 точок. У людському мозку дані зберігаються розподіленим способом, і ми можемо отримувати частини цих даних із нашої пам’яті паралельно, коли це необхідно. Можна сказати, що мозок людини складається з неймовірно дивовижних паралельних процесорів.

Ми можемо зрозуміти штучні нейронні мережі на такому прикладі. Розглянемо приклад цифрового логічного вентиля, який приймає вхідні дані та створює вихідні дані. Ворота АБО, яка приймає два входи. Якщо один або обидва входи "увімкнено", ми отримуємо "ввімкнено" на виході. Якщо обидва входи "закриті", то ми отримуємо "закриті" на виході. Тут вихід залежить від вхідних даних. Наш мозок виконує різні завдання. Співвідношення виходу і входу постійно змінюється завдяки «навчаючим» нейронам нашого мозку.

1.3 Архітектура штучної нейронної мережі

Щоб зрозуміти концепцію архітектури штучної нейронної мережі, ми повинні зрозуміти, з чого складається нейронна мережа. Щоб визначити нейронну мережу, яка складається з великої кількості штучних нейронів, які називаються одиницями, розташованими в послідовності шарів. Давайте розглянемо різні типи шарів, доступних у штучній нейронній мережі.

ШНМ в основному складається з трьох рівнів (рис. 1.7):

1. Вхідний шар — к випливає з назви, він приймає вхідні дані в кількох різних форматах, наданих програмістом.
2. Прихований шар — є проміжним між вхідним і вихідним шарами. Він виконує всі обчислення, щоб знайти приховані особливості та закономірності.
3. Вихідний рівень — вхідні дані проходять серію перетворень за допомогою прихованого шару, що в кінцевому підсумку призводить до вихідних даних, які передаються за допомогою цього шару.

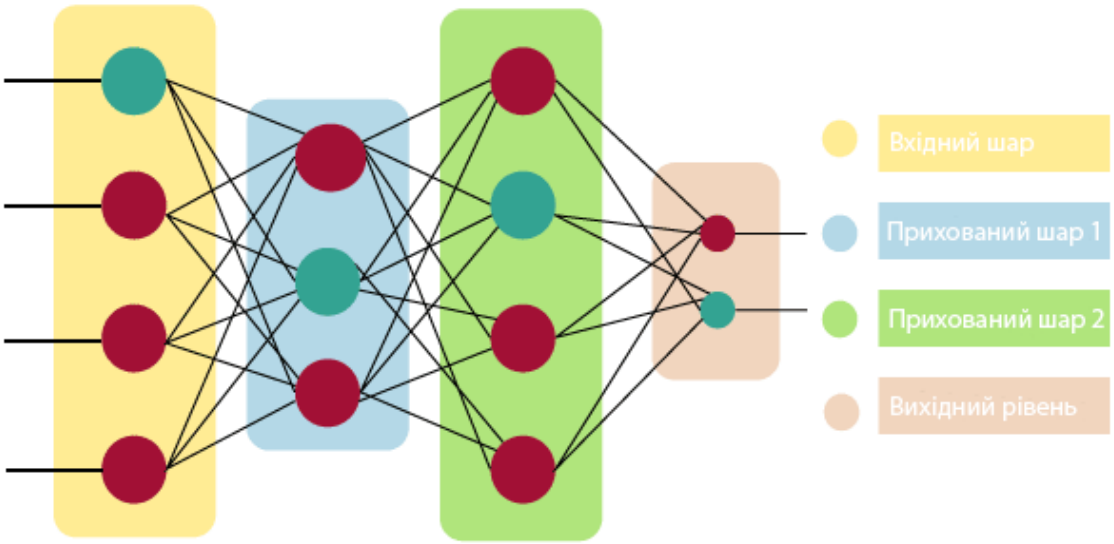

Рисунок 1.7 - Рівні штучної нейронної

Штучна нейронна мережа приймає вхідні дані та обчислює зважену суму вхідних даних і включає зміщення (1.4). Це обчислення представлено у вигляді функції передачі.

$$\sum_{=0} \quad \square\square * \square\square + \square, \tag{1.4}$$

де $\square\square$ – вхідні дані;

$\square\square$ – вихідні дані;

$\square$ – зміщення.

Він визначає, що зважена сума передається як вхідні дані для функції активації для отримання результату. Функції активації визначають, чи повинен вузол запускатися чи ні. На вихідний шар потрапляють лише звільнені. Доступні відмінні функції активації, які можна застосувати до типу завдань, які ми виконуємо.

Переваги штучної нейронної мережі:

1. Штучні нейронні мережі мають числове значення, яке може виконувати більше одного завдання одночасно.
2. Зберігання даних по всій мережі — дані, які використовуються в традиційному програмуванні, зберігаються у всій мережі, а не в базі даних. Зникнення кількох даних в одному місці не заважає мережі працювати.
3. Здатність працювати з неповними знаннями – після навчання ШНМ інформація може давати вихід навіть з неадекватними даними. Втрата продуктивності тут залежить від значущості відсутніх даних.
4. Має розподіл пам'яті – для можливості адаптації ШМН важливо визначити приклади та заохочувати мережу відповідно до бажаного результату, демонструючи ці приклади мережі. Послідовність мережі прямо

пропорційна вибраним екземплярам, і якщо подія не може відображатися в мережі в усіх своїх аспектах, це може спричинити помилковий вихід.

5. Відмовостійкість — вимагання однієї чи кількох комірок ШНМ не забороняє їй генерувати вихідні дані, і ця функція робить мережу відмовостійкою.

Недоліки штучної нейронної мережі:

1. Забезпечення належної структури мережі – немає особливих вказівок щодо визначення структури штучних нейронних мереж. Відповідна структура мережі досягається шляхом досвіду, проб і помилок.

2. Нерозпізнана поведінка мережі – це найважливіше питання ANN. Коли ANN створює рішення для тестування, воно не дає зрозуміти, чому і як. Це знижує довіру до мережі.

3. Апаратна залежність – штучні нейронні мережі потребують процесорів з паралельною обчислювальною потужністю відповідно до їх структури. Тому реалізація обладнання є залежною.

4. Складність показу проблеми в мережі – ШНМ можуть працювати з числовими даними. Проблеми повинні бути перетворені в числові значення перед введенням у ШНМ. Механізм представлення, який тут потрібно вирішити, безпосередньо впливатиме на продуктивність мережі. Це залежить від здібностей користувача.

Штучну нейронну мережу найкраще можна представити як зважений орієнтований граф, де штучні нейрони утворюють вузли (рис. 1.8). Зв’язок між виходами нейронів і входами нейронів можна розглядати як спрямовані ребра з вагами. Штучна нейронна мережа отримує вхідний сигнал від зовнішнього джерела у вигляді шаблону та зображення у вигляді вектора. Потім ці входи математично призначаються за допомогою нотацій x(n) для кожного n числа входів.

Після цього кожен вхід множиться на відповідні вагові коефіцієнти (ці вагові коефіцієнти є деталями, які використовуються штучними нейронними мережами для вирішення певної проблеми). Загалом, ці ваги зазвичай

представляють силу взаємозв'язку між нейронами всередині штучної нейронної мережі. Усі зважені вхідні дані підсумовуються всередині обчислювального блоку.

Якщо зважена сума дорівнює нулю, тоді додається зсув, щоб зробити вихід ненульовим або щось інше для масштабування відповіді системи. Зміщення має той самий вхід, а вага дорівнює 1. Тут загальна сума зважених вхідних даних може бути в діапазоні від 0 до позитивної нескінченності. Тут, щоб зберегти відгук у межах бажаного значення, певне максимальне значення порівнюється, а загальна кількість зважених вхідних даних передається через функцію активації.

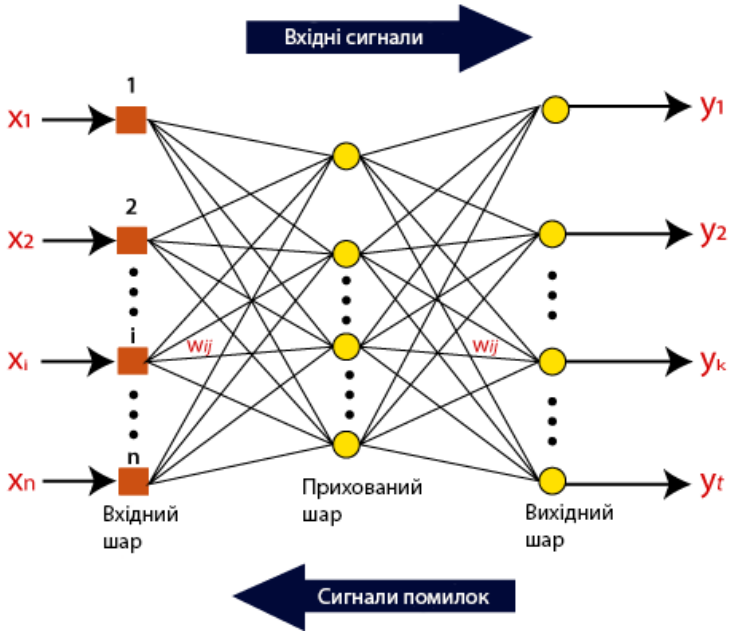

Рисунок 1.8 - Представлення штучної нейронної мережі

Функція активації відноситься до набору функцій передачі, які використовуються для досягнення бажаного результату. Існує інший тип функції активації, але переважно лінійні або нелінійні набори функцій.

2 ЗГОРТКОВА НЕЙРОННА МЕРЕЖА

Останніми роками глибокі нейронні мережі знову набули величезної популярності. З наявністю набагато потужніших обчислювальних ресурсів, таких як кластери центральних процесорів і графічних процесорів (блоки обробки графіки), дослідження тепер здатні створювати мережеві структури з навіть більшою кількістю і ширших рівнів, ніж раніше (рис. 2.1).

Рисунок 2.1 — Класифікація нейронних мереж

Зростання глибинних нейронних мереж від 8-рівневої AlexNet до 19-рівневої VGG, 22-рівневої GoogleNet, а потім 152-рівневої ResNet (рис. 2.1), демонструє чітке узагальнення ідеї, що більш глибокі мережі працюють краще, і це має було досить переконливо, що тенденція еволюції штучних нейронних мереж є в напрямку ще глибших або складніших структур (рис.2.2).

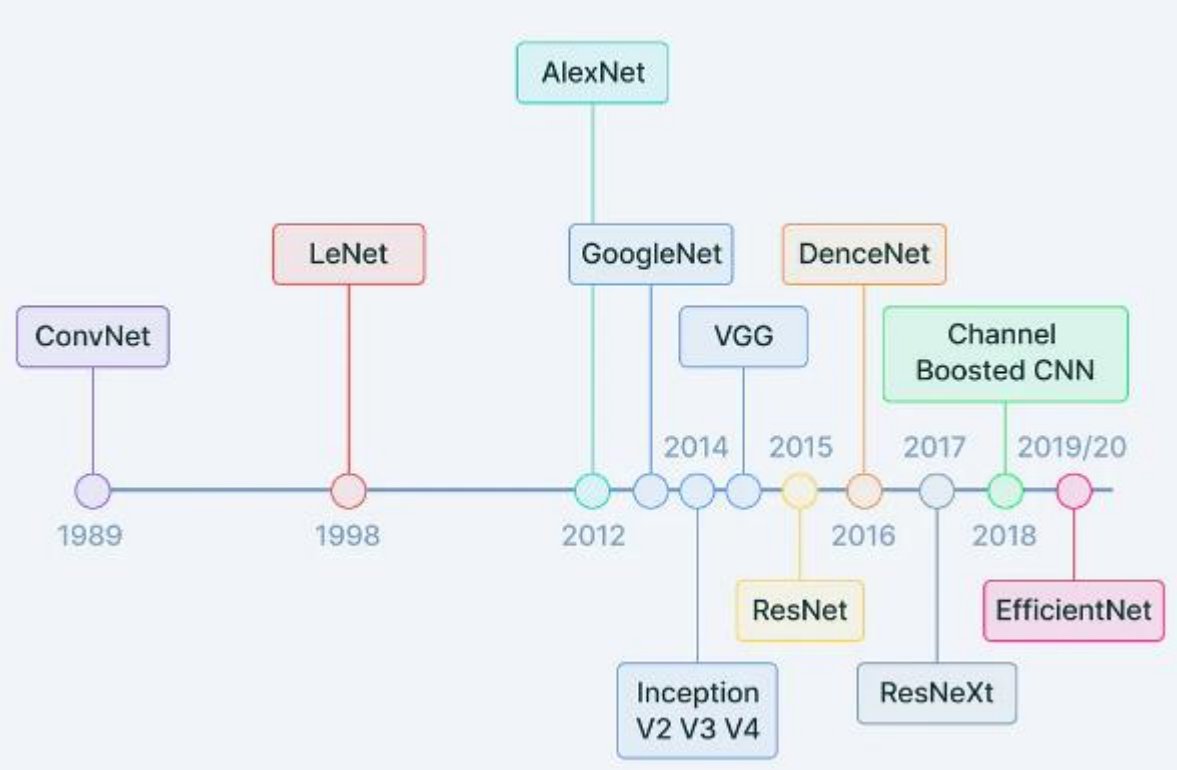

Рисунок 2.2 — Розвиток згорткових нейронних мереж

Однак ця тенденція повсюдно базується на припущенні достатньо великого розміру вибірки (тобто Big Data), і питання надмірного або недостатнього навчання можна розумно ігнорувати. У той час як у численних додатках реального світу кількість зразків у наборі даних може бути відносно обмеженою, і проблеми виникнуть із узагальненим методом укладання простого шару в спробі покращити продуктивність. Цей виклик «малих даних» вимагатиме зовсім іншого мислення та підходу, ніж існуючі «великі

дані». Зокрема, у випадку «невеликих даних» (наприклад, вибірка < 5000), межі все ще розмиті щодо того, чи чим глибше ми йдемо, тим краще ми працюємо, чи на якій глибині чи ширині ми досягаємо максимальної точності. Ця проблема невеликого розміру вибірки становить особливий інтерес, коли нейронні мережі застосовуються до медичних зображень, включаючи МРТ, КТ, ультразвукові та гістопатологічні цифрові зображення, які часто мають обмежений розмір вибірки, обмежений доступністю популяції пацієнтів і маркуванням експертів. Типові стратегії, які використовуються для навчання Big Data, можуть бути непридатними для цих програм.

2.1 Структура згорткової нейронної мережі

ЗНМ — це тип моделі глибокого навчання для обробки даних, які мають шаблон сітки, наприклад зображень, який натхненний організацією зорової кори тварин [13] і призначений для автоматичного й адаптивного вивчення просторових ієрархій ознак із низьких - до шаблонів високого рівня. ЗНМ — це математична конструкція, яка зазвичай складається з трьох типів шарів (або будівельних блоків): шари згортки, об’єднання та повністю зв’язані шари. Перші два, шари згортки та об’єднання, виконують виділення ознак, тоді як третій, повністю пов’язаний рівень, відображає витягнуті функції в кінцевий результат, наприклад класифікацію. Рівень згортки відіграє ключову роль у ЗНМ, який складається зі стеку математичних операцій, таких як згортка, спеціалізований тип лінійної операції. У цифрових зображеннях значення пікселів зберігаються у двовимірній (2D) сітці, тобто масиві чисел, і невелика сітка параметрів, яка називається ядром, оптимізованим екстрактором ознак, застосовується до кожної позиції

зображення , що робить ЗНМ дуже ефективними для обробки зображень, оскільки особливість може з’являтися в будь-якому місці зображення. Оскільки один шар передає свій вихід на наступний рівень, витягнуті функції можуть ієрархічно та поступово ставати складнішими. Процес оптимізації таких параметрів, як ядра, називається навчанням, яке виконується таким чином, щоб мінімізувати різницю між виходами та основними мітками істинності за допомогою алгоритму оптимізації, що називається зворотним поширенням і градієнтним спуском, серед інших.

Згорткова нейронна мережа складається з одного або кількох згорткових шарів (часто з кроком підвибірки), за якими слідує один або більше повністю пов’язаних шарів, як у стандартній багатошаровій нейронній мережі. Архітектура ЗНМ розроблена для використання переваг 2D-структури вхідного зображення (або іншого 2D-введення, наприклад мовного сигналу). Це досягається за допомогою локальних зв’язків і прив’язаних вагових коефіцієнтів, що супроводжуються певною формою об’єднання, що призводить до інваріантних функцій перекладу. Ще одна перевага ЗНМ полягає в тому, що їх легше навчити і вони мають набагато менше параметрів, ніж повністю підключені мережі з такою ж кількістю прихованих блоків. У цій статті ми обговоримо архітектуру ЗНМ і алгоритм зворотного поширення для обчислення градієнта щодо параметрів моделі з метою використання оптимізації на основі градієнта. Перегляньте відповідні навчальні посібники щодо згортання та об’єднання, щоб отримати докладнішу інформацію про ці конкретні операції.

У останніх дослідженнях машинного навчання використовуються ручні методи вилучення ознак, такі як аналіз текстури, а потім звичайні класифікатори машинного навчання, такі як опорні векторні машини [15, 16]. Існує кілька відмінностей між такими методами та ЗНМ. По-перше, ЗНМ не вимагає ручного вилучення функцій. По-друге, архітектури ЗНМ не обов’язково вимагають сегментації пухлин або органів фахівцями. По-третє, ЗНМ набагато більше потребує даних через мільйони параметрів, які можна

оцінити, і, отже, є дорожчим з точки зору обчислень, що призводить до потреби графічних процесорів (GPU) для навчання моделі.

ЗНМ складається з ряду шарів згортки та підвибірки (рис. 2.1), за якими необов’язково йдуть повністю зв’язані шари. Вхідними даними для згорткового шару є зображення розміром m x m x rm x m x r, де mm — висота та ширина зображення, а rr — кількість каналів, напр. зображення RGB має r=3r=3. Згортковий шар матиме kk фільтрів (або ядер) розміром n x n x qn x n x q, де nn менше розміру зображення, а qq може дорівнювати кількості каналів rr або бути меншим і може відрізнятися для кожного ядра. Розмір фільтрів створює локально зв’язану структуру, кожна з яких згортається разом із зображенням для створення kk карт властивостей розміром m−n 1m−n 1. Кожна карта потім зазвичай підвибирається із середнім або максимальним об’єднанням за p x pp x p суміжних областей, де p коливається між 2 для малих зображень і зазвичай не перевищує 5 для великих вхідних даних. До або після шару підвибірки до кожної карти ознак застосовуються адитивне зміщення та сигмоподібна нелінійність. На малюнку 2.3 нижче показано перший рівень у ЗНМ, що складається із згорткових і підрівнів підвибірки. Одиниці одного кольору мають рівну вагу.

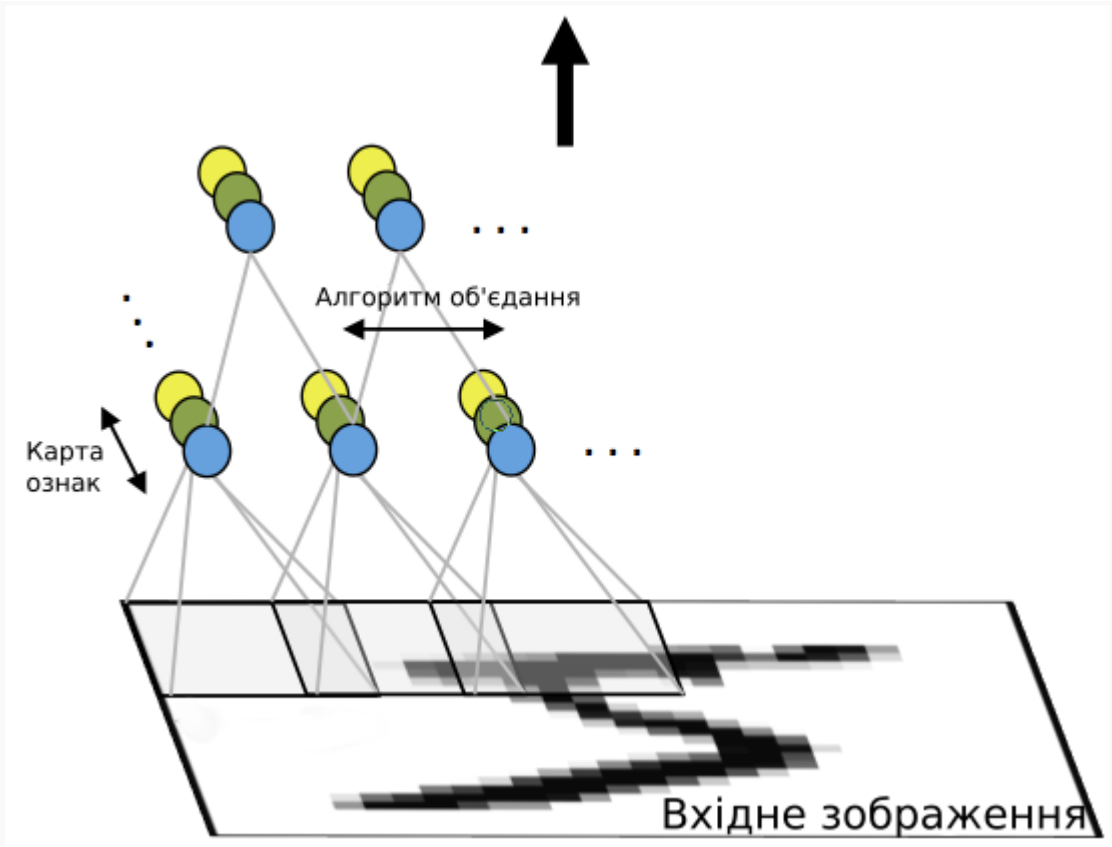

Рисунок 2.3 —Перший рівень згорткової нейронної мережі з об’єднанням.

Після згорткових шарів може бути будь-яка кількість повністю зв'язаних шарів. Щільно з’єднані шари ідентичні шарам стандартної багатошарової нейронної мережі.

2.2 Згортковий шар мережі

Згортковий рівень є фундаментальним компонентом архітектури ЗНМ, який виконує вилучення ознак, яке зазвичай складається з комбінації лінійних і нелінійних операцій, тобто операції згортки та функції активації.

Згортка — це спеціалізований тип лінійної операції, яка використовується для виділення ознак, де невеликий масив чисел, який називається ядром, застосовується до вхідних даних, які є масивом чисел,

який називається тензором. Поелементний добуток між кожним елементом ядра та вхідним тензором обчислюється в кожному місці тензора та підсумовується, щоб отримати вихідне значення у відповідній позиції вихідного тензора, що називається картою ознак. Ця процедура повторюється із застосуванням кількох ядер для формування довільної кількості карт ознак, які представляють різні характеристики вхідних тензорів.

Таким чином, різні ядра можна розглядати як різні екстрактори функцій. Двома ключовими гіпер параметрами, які визначають операцію згортки, є розмір і кількість ядер. Перше зазвичай становить 3 × 3, але іноді 5 × 5 або 7 × 7. Останнє є довільним і визначає глибину вихідних карт ознак.

На рисунку 2.4 показано приклад операції згортки з розміром ядра 3 × 3, без заповнення та кроком 1. Ядро застосовується до вхідного тензора, а поелементний добуток між кожним елементом ядра та вхідними даними тензор обчислюється в кожному місці та підсумовується для отримання вихідного значення у відповідній позиції вихідного тензора, що називається картою ознак.

Операція згортки, описана вище, не дозволяє центру кожного ядра перекривати крайній елемент вхідного тензора та зменшує висоту та ширину вихідної карти ознак порівняно з вхідним тензором. Заповнення, як правило, нульове заповнення, є технікою для вирішення цієї проблеми, коли рядки та стовпці нулів додаються з кожного боку вхідного тензора, щоб центр ядра відповідав крайньому зовнішньому елементу та залишався в одній площині. розмірності за допомогою операції згортання.

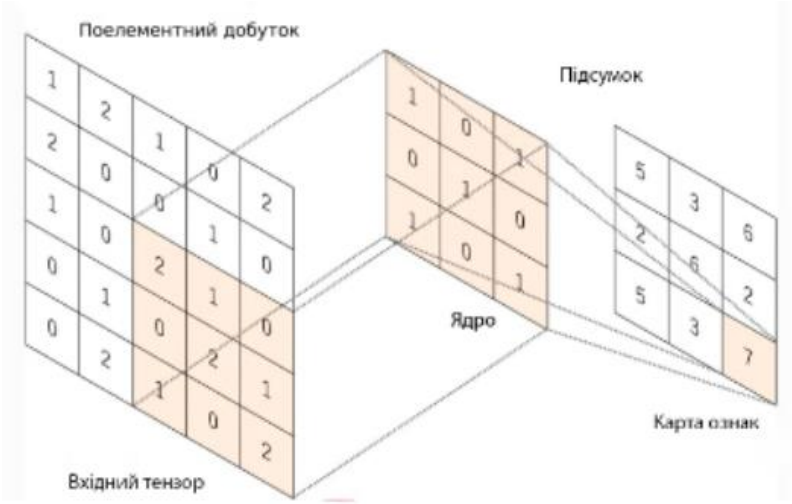

Рисунок 2.4 — Операція згортки

Сучасні архітектури ЗНМ зазвичай використовують нульове доповнення, щоб зберегти розміри в площині, щоб застосувати більше шарів. Без нульового доповнення кожна послідовна карта функцій ставала б меншою після операції згортки.

2.3 Шар об’єднання

Шар об’єднання забезпечує типову операцію зменшення дискретизації, яка зменшує площинну розмірність карт функцій, щоб запровадити інваріантність трансляції до невеликих зсувів і спотворень, а також зменшити кількість наступних параметрів, які можна вивчати. Слід зазначити, що в жодному з шарів об’єднання немає параметрів, які можна вивчати, тоді як розмір фільтра, крок і відступ є гіперпараметрами в операціях об’єднання, подібних до операцій згортки.

Найпопулярнішою формою операції об’єднання є максимальне об’єднання (рис. 2.5), яке витягує патчі з вхідних карт об’єктів, виводить максимальне значення в кожному патчі та відкидає всі інші значення (рис. 6).

На практиці зазвичай використовується максимальне об’єднання з фільтром розміру 2×2 із кроком 2. Це зменшує дискретизацію площинного розміру карт об’єктів у 2 рази. На відміну від висоти та ширини, розмірність глибини карт об’єктів залишається незмінною.

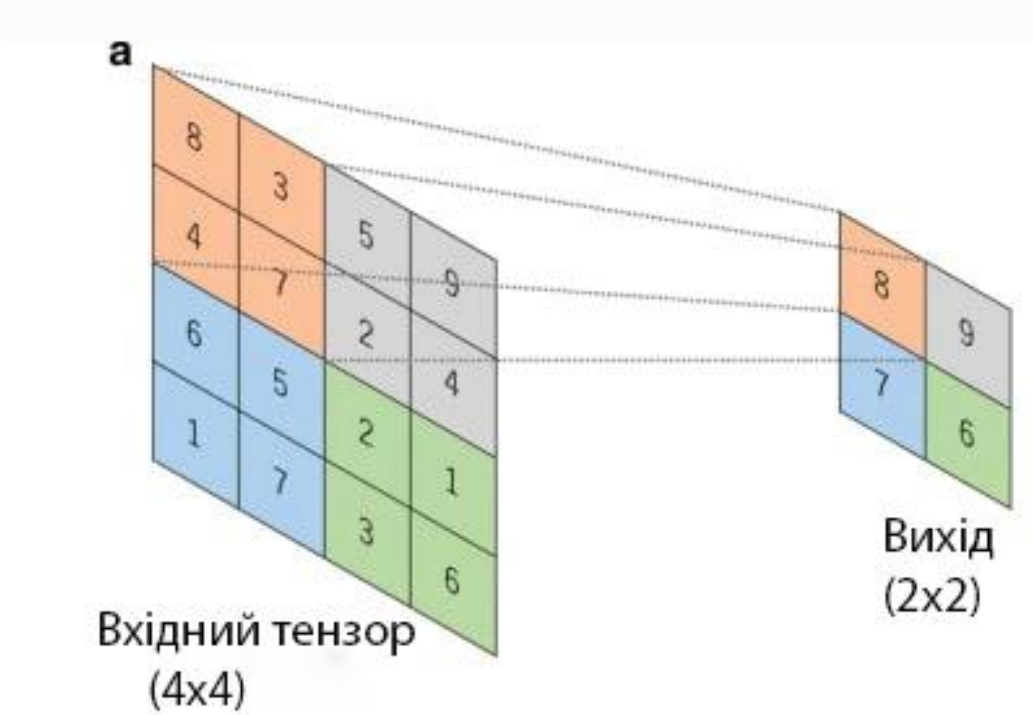

Рисунок 2.5 — Алгоритм максимального об’єднання

2.4 Навчання мережі

Навчання мережі — це процес пошуку ядер у шарах згортки та вагових коефіцієнтів у повністю зв’язаних шарах, що мінімізує відмінності між вихідними прогнозами та заданими базовими мітками істинності в навчальному наборі даних. Алгоритм зворотного поширення — це метод, який зазвичай використовується для навчання нейронних мереж, де функція втрат і алгоритм оптимізації градієнтного спуску відіграють важливу роль. Ефективність моделі під певними ядрам і вагами обчислюється функцією

втрат шляхом прямого поширення на тренувальному наборі даних, а параметри, які можна вивчати, а саме ядра та ваги, оновлюються відповідно до значення втрат за допомогою алгоритму оптимізації, що називається зворотним поширенням і градієнтним спуском, серед іншого.

Функція втрат, яку також називають функцією вартості, вимірює сумісність між вихідними прогнозами мережі через пряме поширення та заданими наземними мітками істинності. Зазвичай використовуваною функцією втрат для багатокласової класифікації є крос-ентропія, тоді як середня квадратична помилка зазвичай застосовується для регресії до неперервних значень. Вид функції втрат є одним із гіперпараметрів і потребує визначення відповідно до поставлених завдань.

Градієнтний спуск — це алгоритм оптимізації, який ітеративно оновлює параметри, які можна вивчати, щоб мінімізувати втрати, що вимірює відстань між вихідним прогнозом і базовою міткою істинності (рис. 2.4). Градієнт функції втрат визначає напрямок, у якому функція має найкрутіший темп зростання, і всі параметри оновлюються в негативному напрямку градієнта з розміром кроку, визначеним на основі швидкості навчання.

Градієнтний спуск зазвичай використовується як алгоритм оптимізації, який ітеративно оновлює параметри, які можна вивчати, тобто ядра та ваги, мережі, щоб мінімізувати втрати. Градієнт функції втрат дає нам напрямок, у якому функція має найкрутіший темп зростання, і кожен параметр, який можна вивчати, оновлюється в негативному напрямку градієнта з довільним розміром кроку, визначеним на основі гіпер параметра, що називається швидкістю навчання (рис. 2.6).

Слід зазначити, що на практиці швидкість навчання є одним із найважливіших гіпер параметрів, який необхідно встановити перед початком навчання. На практиці з таких причин, як обмеження пам’яті, градієнти функції втрат щодо параметрів обчислюються за допомогою підмножини навчального набору даних, що називається міні-пакетом, і

застосовуються до оновлень параметрів. Цей метод називається міні-пакетним градієнтним спуском, який також часто називають стохастичним градієнтним спуском, і міні-пакетний розмір також є гіпер параметром. Крім того, було запропоновано та широко використовується багато вдосконалень алгоритму градієнтного спуску, наприклад стохастичний градієнтний спуск з імпульсом.

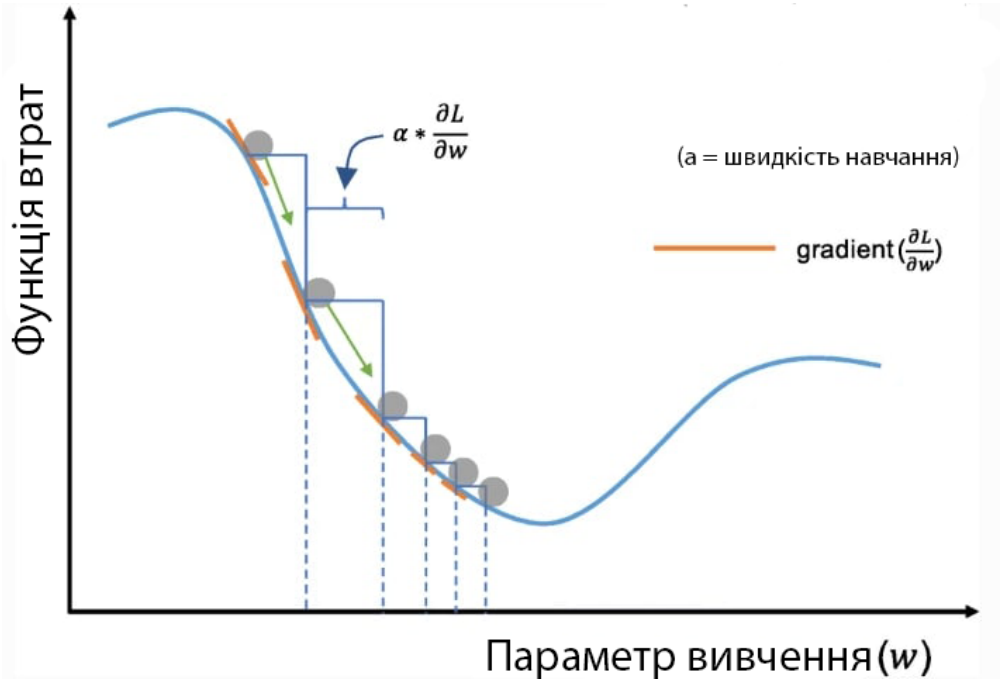

Рисунок 2.6 — Обчислення градієнтного спуску

Градієнт, математично, є частковою похідною від втрати щодо кожного параметра, що вивчається, і одне оновлення параметра формулюється таким чином:

$$□ = □ - □ * \frac{□□}{□□} \quad , \qquad (2.1)$$

де w – параметр, який можна вивчати,

α – швидкість навчання,

L – функція втрат.

Дані та базові мітки істинності є найважливішими компонентами в дослідженнях із застосуванням глибокого навчання або інших методів машинного навчання. Ретельний збір даних і базових міток істинності, за допомогою яких можна навчати та тестувати модель, є обов'язковим для успішного проекту глибокого навчання, але отримання високоякісних мічених даних може бути дорогим і трудомістким. Хоча може існувати декілька наборів даних медичних зображень, відкритих для громадськості, у цих випадках слід приділяти особливу увагу якості наземних міток істинності.

Доступні дані зазвичай поділяються на три набори: навчальний, перевіряльний і тестовий набір (рис. 2.7), хоча існують деякі варіанти, наприклад перехресна перевірка.

Рисунок 2.7 — Алгоритм перевірки даних

Навчальний набір використовується для навчання мережі, де значення втрат обчислюються за допомогою прямого розповсюдження, а параметри, що вивчаються, оновлюються за допомогою зворотного розповсюдження. Набір перевірки використовується для оцінки моделі під час процесу навчання, точного налаштування гіпер параметрів і виконання вибору моделі. В ідеалі тестовий набір використовується лише один раз у самому кінці проекту, щоб оцінити продуктивність остаточної моделі, яка була налаштована та обрана в процесі навчання за допомогою наборів для навчання та перевірки. Потрібні окремі набори для перевірки та тестування, оскільки навчання моделі завжди передбачає точне налаштування її гіпер параметрів і вибір моделі. Оскільки цей процес виконується на основі продуктивності набору перевірки, деяка інформація про цей набір перевірки просочується в саму модель, тобто відбувається переобладнання набору перевірки, навіть якщо модель ніколи не навчається безпосередньо на ньому для параметрів, які можна вивчати. З цієї причини гарантується, що модель з точно налаштованими гіпер параметрами в наборі перевірки буде добре працювати на цьому ж наборі перевірки. Таким чином, абсолютно невидимий набір даних, тобто окремий тестовий набір, необхідний для відповідної оцінки продуктивності моделі, оскільки ми дбаємо про продуктивність моделі на даних, які ніколи раніше не бачили, тобто можливість узагальнення.

Варто зазначити, що термін «валідація» по-різному використовується в галузі медицини та машинного навчання [26]. Як описано вище, у машинному навчанні термін «перевірка» зазвичай відноситься до етапу тонкого налаштування та вибору моделей під час процесу навчання. З іншого боку, у медицині «валідація» зазвичай означає процес перевірки ефективності моделі прогнозування, що аналогічно терміну «тест» у машинному навчанні.

2.5 Навчальна вибірка

Одним із можливих рішень проблеми невеликого розміру вибірки є використання попередньо навчених мереж, також відомих як трансферне навчання. Ці підходи набули популярності в багатьох галузях, щоб впоратися з відсутністю значних вибірок у наборі даних. Ідея полягає в тому, щоб ініціалізувати нейронну мережу за допомогою вагових коефіцієнтів, навчених у відповідних доменах, і точно налаштувати модель за допомогою даних у домені. Цей підхід забезпечує розумний початковий стан і може прискорити навчання, коли два домени близькі. Існують певні високоефективні рішення в області медичної візуалізації щодо використання попередньо навченої мережі. Однак у полях із використанням медичних зображень, де формат вхідних пікселів (наприклад, дані, не пов’язані з RGB, такі як зображення ультразвуку/МРТ у градаціях сірого або RGB + Depth), може повністю відрізнятися від звичайних фотографічних зображень, де використовуються попередньо навчені мережі з інших доменів зображення неявно висувають гіпотезу про те, що оптимальна мережева структура є універсальною (мережі з можливістю передачі), а не керованою (або меншою мірою) даними. Навіть якщо формат пікселя ідентичний, необхідно ретельно перевірити, чи є попередньо навчена мережа повним і надійним рішенням у вищезазначених областях, оскільки в деяких наборах даних підвищення продуктивності за допомогою набору попереднього навчання, якщо не суттєве, може вказувати на значна подібність між набором для попереднього навчання та цільовим набором, що вказує на те, що в багатьох випадках попереднє навчання може працювати, лише якщо набори були суттєво подібними. Також може виникнути «Негативний перехід» у випадках, коли існує великий розрив домену між набором даних попереднього навчання та набором даних під час навчання, що призводить до зниження продуктивності під час навчання після тонкого налаштування порівняно з навчання з нуля.

Отже, все ще існує деяка неоднозначність у безпомилковій природі методології попередньо навченої мережі.

Крім того, оскільки ми стикаємося з потребою застосовувати нейронні мережі до все менших і менших розмірів вибірки, проблема полягає в тому, що ми не знаємо, чи підмножина більшої вибірки даних може мати таку ж оптимальну мережу, як і весь набір даних. Наприклад, з меншим розміром навчальних даних дуже глибока мережа (наприклад, ResNet) може погано узагальнюватися, оскільки вона може перевищувати невеликі навчальні дані, вимагаючи застосування іншої схеми навчання (наприклад, додавання регулярізації) або іншої стратегії, спеціалізованої на невеликі набори даних.

Тут ми прагнемо відповісти на вищезазначені питання, коли мова йде про навчання згорткової нейронної мережі з малими розмірами вибірки. По-перше, ми вичерпно повторили список мереж із верхньою межею складності, навчили їх і вивчили їх продуктивність, прагнучи зрозуміти залежність продуктивності мережі для кожного набору даних від структурних гіпер параметрів, а саме природи шарів (наприклад, згортка, повний зв’язок, максимальне об’єднання… тощо), кількість шарів і розмір шарів. Далі ми дослідили гіпотезу «природа проти розміру»: чи оптимальна структура нейронної мережі в ідентичному піксельному форматі (RGB) в основному визначається «природою» даних (наприклад, фотографічні, каліграфічні, мікроскопічні зображення) або «розмір» даних. Іншими словами, ми хотіли б вивчити, чи будуть набори даних із різних джерел даних або модальності зображень мати дуже схожі (або різні) оптимально ефективні мережі порівняно з впливом іншого розміру вибірки. Якщо ефект «розміру даних» більший, ніж ефект «природи даних», то це означає, що складність структури оптимальної мережі зростатиме, оскільки розмір даних стає більшим. У цьому випадку ми можемо сліпо використовувати попередньо навчені мережі, навчені на різних модальностях зображення, і завжди обираємо збільшення складності використовуваної мережі, якщо формат пікселів ідентичний.

Навпаки, якщо ефект природи даних є більшим, тоді використання попередньо навченої мережі з різних джерел зображень може бути неприйнятним. Крім того, він підтримуватиме стратегію, згідно з якою, коли йдеться про проблему великих даних, ми можемо логічно використовувати лише їхню частину для оптимізації структури мережі. Завдяки дослідженню цієї гіпотези ми прагнемо підтвердити або скасувати здатність попередньо навченої мережі як універсальної водонепроникної процедури в області менших розмірів вибірки (медичні зображення, розпізнавання обличчя тощо), проти протилежного підходу, який вимагатиме пошук найкращої мережі, спеціально розробленої для наборів даних меншого розміру, які ми встановили як проблему в різних сферах. Нарешті, ми розрахуємо найкращу продуктивність, яку може досягти згортка нейронної мережі після оптимізації структури мережі, і порівняємо її із середньою продуктивністю випадково вибраних мереж. Ця евристика слугує насамперед для визначення факторів, які впливають на оптимальність архітектури мережі, а також для демонстрації проблеми, з якою стикаються такі мережі, коли їм надаються менші навчальні набори. Це не призначено для заміни кращої евристики оптимізації структури мережі. Однак досі не було проведено комплексних досліджень, які намагалися б проаналізувати цю гіпотезу «природа проти розміру».

Щоб перевірити це, ми використали три набори даних зображень абсолютно різного характеру, включаючи MNIST (каліграфічні), CIFAR10 (фотографічні) і дані зображення Mitosis (мікроскопічні). Ми навчили та протестували кожну структуру з наступною оптимізацією розмірів шару (ширини шару), використовуючи невеликі підмножини (менше 5000 зразків) цих наборів даних, і дослідили різницю в продуктивності в різних мережевих структурах, як описано в наступних розділах.

2.6 Застосування згорткових нейронних мереж

Як ми обговорювали спочатку, ЗНМ є основою комп’ютерного зору, де значима інформація витягується з візуальних даних для виконання певних дій. За останнє десятиліття галузь розвинулася, і впровадження технологій на основі штучного інтелекту/ML стало значним зростанням. Комп’ютерний зір набирає обертів завдяки вичерпним можливостям застосування в різних галузях. Нижче наведено деякі з найважливіших сфер застосування комп’ютерного зору:

1. Відеоспостереження — багато стартапів працюють над інтелектуальними системами спостереження, які забезпечуватимуть сповіщення в режимі реального часу у разі крадіжки зі зломом, пожежі, насильства чи будь-якої іншої надзвичайної ситуації. Розумні камери відеоспостереження були розгорнуті в місцях, схильних до аварій або сигналів світлофора. За допомогою технології виявлення номерних знаків і OCR ДАІ може ідентифікувати громадян, які порушують правила дорожнього руху або затримуються за ДТП. Прилеглі поліцейські відділки та лікарні також можуть бути повідомлені про смертельні аварії.

2. Охорона здоров’я — Тривають дослідження з використання інструментів комп’ютерного зору в радіології для визначення проблемних областей під час сканування МРТ або рентгенівського знімка для огляду лікарями чи радіологами. Це може допомогти скоротити додатковий час для аналізу кожного звіту пацієнта з нуля. Для цього використовуються різні методи, такі як класифікація, семантична сегментація, виявлення медичних об’єктів тощо. Були також проведені деякі експерименти для створення повного звіту на основі рентгенівського сканування, що має на меті значно зменшити робоче навантаження на радіологів, особливо в часи пандемії, коли лабораторії переповнені зразками пацієнтів, а робоча сила обмежена.

3. Оцифрування посвідчень особи — розробляються інструменти для введення відсканованих зображень посвідчень особи, таких як картки Adhaar, ідентифікаційні картки виборця, картки PAN тощо, і отримання з них відповідної інформації. Ви можете використовувати це для легкої перевірки інформації, наданої користувачем. Ці інструменти можуть зіграти значну роль в автоматизації процесу адаптації нових клієнтів/клієнтів.

4. Підробка підпису — у банках звіряння поточного підпису особи з підписом у банківських документах було дуже ручним завданням. Представник банку зіставив підписи та визначив, чи збігається шаблон, який був схильний до помилок. Комп’ютерний зір значно полегшив це завдання, виконавши перевірку схожості зображень і визначивши, чи були підписи підробленими чи ні.

5. Регулювання вмісту в соціальних медіа — соціальні медіа – це платформа, на якій є користувачі різних вікових груп. Часто розповсюджуваний вміст, наприклад зображення чи відео, може бути неприйнятним для дітей через відвертий вміст. Раніше такий вміст позначався, якщо про нього повідомляли користувачі, але тепер використовуються методи комп’ютерного зору, щоб визначити, які зображення містять відвертий вміст або екстремальне насильство. Вони або видаляються, або приховуються з попередженням на власний розсуд.

3 АЛГОРИТМ ПЕРЕТВОРЕННЯ СТИЛЮ 3D ТЕКСТУР ЗА ДОПОМОГОЮ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

Мета візуальної трансформації текстури полягає в тому, щоб зробити висновок про процес генерації з прикладу текстури, який потім дозволяє створювати будь-яку кількість нових зразків цієї текстури. Критерієм оцінки якості синтезованої текстури зазвичай є перевірка людиною, і текстури успішно синтезовані, якщо людина-спостерігач не може відрізнити оригінальну текстуру від синтезованої.

Перенесення стилю з одного зображення на інше можна вважати проблемою передачі текстури. Метою передачі текстури є синтезувати текстуру з вихідного зображення, одночасно обмежуючи синтез текстури, щоб зберегти семантичний вміст цільового зображення. Для синтезу текстури існує велика кількість потужних непараметричних алгоритмів, які можуть синтезувати фотореалістичні природні текстури шляхом повторної вибірки пікселів даної вихідної текстури [7]. Більшість попередніх алгоритмів передачі текстури покладаються на ці непараметричні методи синтезу текстури, одночасно використовуючи різні способи збереження структури цільового зображення. Наприклад, Ефрос і Фріман вводять карту відповідності, яка включає такі характеристики цільового зображення, як інтенсивність зображення, щоб обмежити процедуру синтезу текстури [8]. Герцман та ін. використовуйте аналогії зображень, щоб перенести текстуру з уже стилізованого зображення на цільове зображення. Ашихмін зосереджується на передачі високочастотної інформації про текстуру зі збереженням грубого масштабу цільового зображення [1]. Лі та ін. покращити цей алгоритм шляхом додаткового інформування передачі текстури інформацією про орієнтацію країв.

Хоча ці алгоритми досягають чудових результатів, усі вони страждають від однакового фундаментального обмеження: вони

використовують лише низькорівневі характеристики цільового зображення для передачі текстури. Однак в ідеалі алгоритм передачі стилю повинен мати можливість витягувати семантичний вміст зображення з цільового зображення (наприклад, об’єктів і загального пейзажу), а потім інформувати процедуру передачі текстури для відтворення семантичного вмісту цільового зображення. у стилі вихідного зображення. Тому фундаментальною передумовою є пошук репрезентацій зображення, які незалежно моделюють варіації семантичного змісту зображення та стилю, в якому

Загалом існує два основні підходи до пошуку процесу генерації текстури. Перший підхід полягає у створенні нової текстури шляхом повторної дискретизації або пікселів [5], або цілих ділянок вихідної текстури. Ці непараметричні методи передискретизації та їх численні розширення та вдосконалення здатні створювати високоякісні природні текстури дуже ефективно. Вони не визначають реальну модель природних текстур, а радше дають механістичну процедуру того, як можна рандомізувати вихідну текстуру без зміни її перцептивних властивостей.

На відміну від цього, другий підхід до трансформації текстур полягає в явному визначенні параметричної моделі текстури. Модель, як правило, складається з набору статистичних вимірювань, які проводяться для просторового масштабу зображення. У моделі текстура однозначно визначається результатом цих вимірювань, і кожне зображення, яке дає той самий результат, має сприйматися як одна текстура. Тому нові зразки текстури можна генерувати шляхом пошуку зображення, яке дає ті самі результати вимірювання, що й оригінальна текстура. Концептуально цю ідею вперше запропонував Юлеш [13], який припустив, що візуальну текстуру можна однозначно описати об’єднаними гістограмами N-го порядку її пікселів. Пізніше моделі текстур були натхненні властивостями лінійного відгуку ранньої зорової системи ссавців, які нагадують властивості орієнтованих смугових фільтрів (Gabor) [10]. Ці моделі текстур базуються на статистичних вимірюваннях, зроблених на відгуках фільтра, а не

безпосередньо на пікселях зображення. Поки що найкращою параметричною моделлю для перетворення текстури є, ймовірно, та, яку запропонували Портілла та Сімончеллі, яка базується на наборі ретельно створених вручну зведених статистичних даних, обчислених на відповідях лінійного банку фільтрів під назвою Steerable Pyramid . Незважаючи на те, що їх модель Hoiver показує дуже хорошу продуктивність у синтезі широкого діапазону текстур, вона все ще не може охопити повний обсяг природних текстур.

У цій роботі я пропоную нову модель параметричної текстури для вирішення цієї проблеми. Замість опису текстур на основі моделі ранньої візуальної системи [10] я використовую згорткову нейронну мережу – функціональну модель для всього вентрального потоку – як основу нашої моделі текстури. Я поєдную концептуальну структуру просторової сумарної статистики щодо відгуків ознак із простором бідних ознак згорткової нейронної мережі, яка була навчена розпізнаванню об’єктів. Таким чином я отримую текстурну модель, яка параметризована просторово інваріантними представленнями, побудованими на ієрархічній архітектурі обробки згорткової нейронної мережі.

3.1 Модель згорткової нейронної мережі

VGG означає Visual Geometry Group — це стандартна глибока архітектура згорткової нейронної мережі із кількома рівнями. «Глибокий» відноситься до кількості шарів з VGG-16 або VGG-19, що складаються з 16 і 19 згорткових шарів (рис. 3.1). Архітектура VGG є основою новаторських моделей розпізнавання об’єктів. Розроблена як глибока нейронна мережа, VGGNet також перевершує базові показники для багатьох завдань і наборів даних за межами ImageNet. Крім того, зараз це одна з найпопулярніших архітектур розпізнавання зображень.

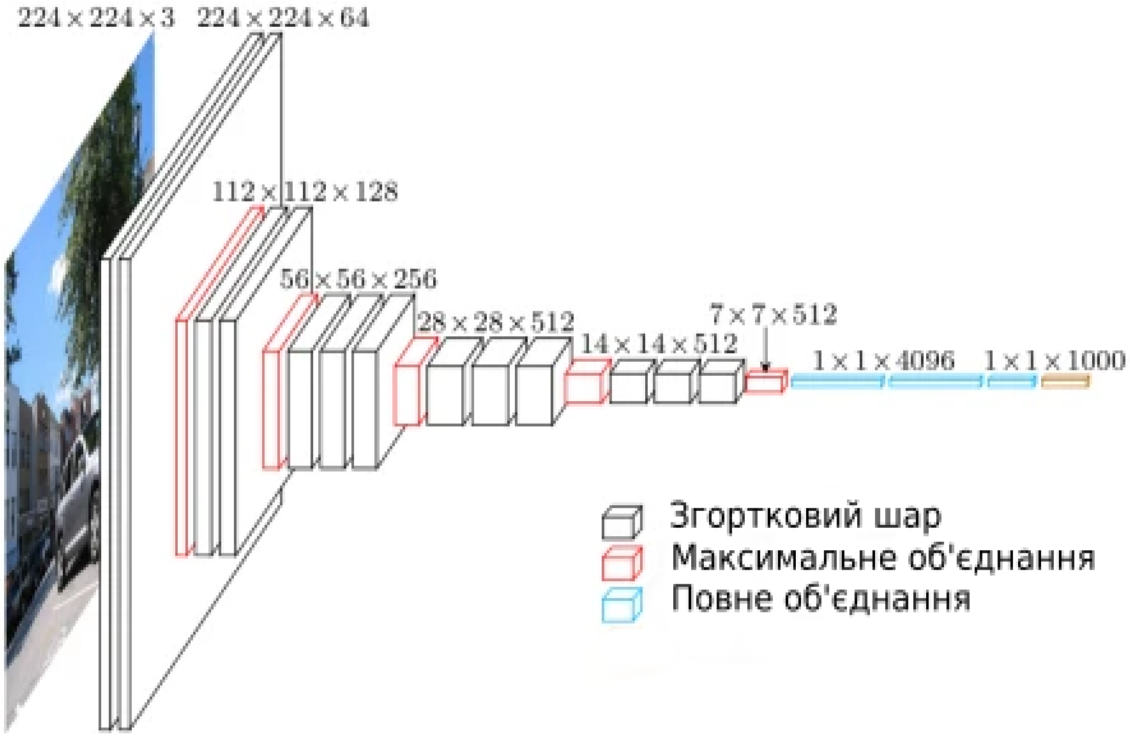

Рисунок 3.1 — Розподіл шарів в нейронній мережі

Модель VGG, або VGGNet, яка підтримує 16 рівнів, також називається VGG16, яка є моделлю згорткової нейронної мережі, запропонованою А. Зіссерманом і К. Симоньян з Оксфордського університету. Ці дослідники опублікували свою модель у дослідницькій статті під назвою «Дуже глибокі згорткові мережі для розпізнавання великомасштабних зображень». Модель VGG16 досягає майже 92,7% точності топ-5 тестів ImageNet. ImageNet — це набір даних, що складається з понад 14 мільйонів зображень, що належать до майже 1000 класів. Крім того, це була одна з найпопулярніших моделей, представлених на ILSVRC-2014. Він замінює великі фільтри розміром ядра кількома фільтрами розміром ядра 3 × 3 один за одним, таким чином значно покращуючи AlexNet. Модель VGG16 навчалася за допомогою графічних процесорів Nvidia Titan Black протягом кількох тижнів. Як згадувалося вище, VGGNet-16 підтримує 16 шарів і може класифікувати зображення за 1000

категоріями об’єктів, включаючи клавіатуру, тварин, олівець, мишу тощо. Крім того, модель має вхідний розмір зображення 224 на 224.

Концепція моделі VGG19 така ж, як і VGG16, за винятком того, що вона підтримує 19 рівнів. «16» і «19» означають кількість вагових шарів у моделі (згорткових шарів). Це означає, що VGG19 має три більше згорткових шарів, ніж VGG16.

Я використовую мережу VGG-19 (рис 3.2), згорткову нейронну мережу, навчену розпізнаванню об’єктів. Було використано простір функцій, наданий 16 згортковими та 5 об’єднаними рівнями мережі VGG-19.

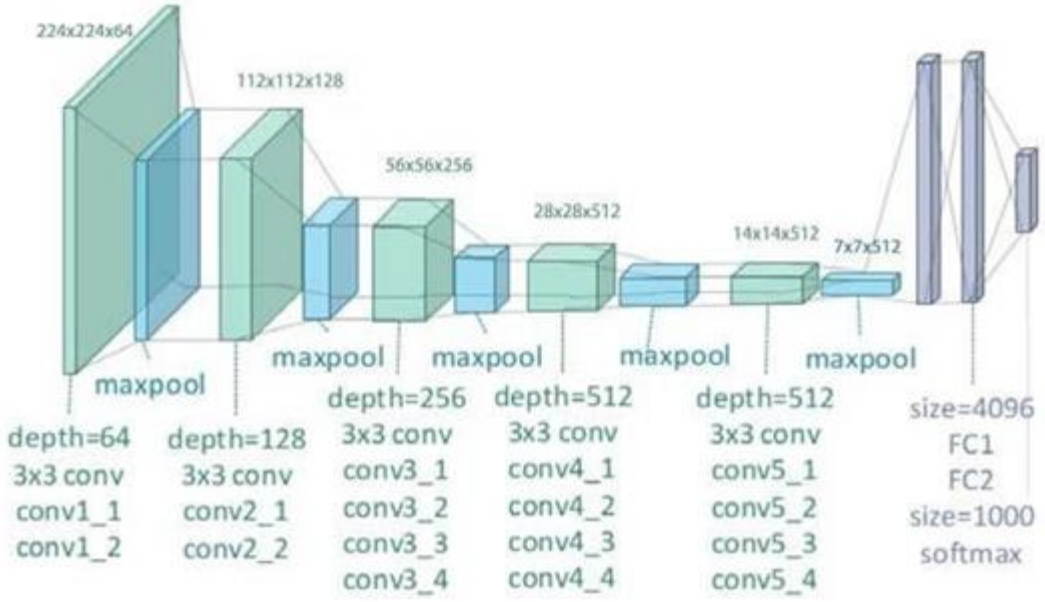

Рисунок 3.2 — Модель мережі VGG-19

VGGNets базуються на найважливіших характеристиках згорткових нейронних мереж (ЗНМ). На рисунку 3.3 показано базову концепцію роботи ЗНМ.

Мережа VGG побудована з дуже маленькими згортковими фільтрами. VGG-16 складається з 13 згорткових шарів і трьох повністю зв'язаних шарів. VGGNet приймає вхідне зображення розміром 224×224. Для конкурсу

ImageNet творці моделі вирізали центральну ділянку розміром 224 × 224 на кожному зображенні, щоб підтримувати вхідний розмір зображення.

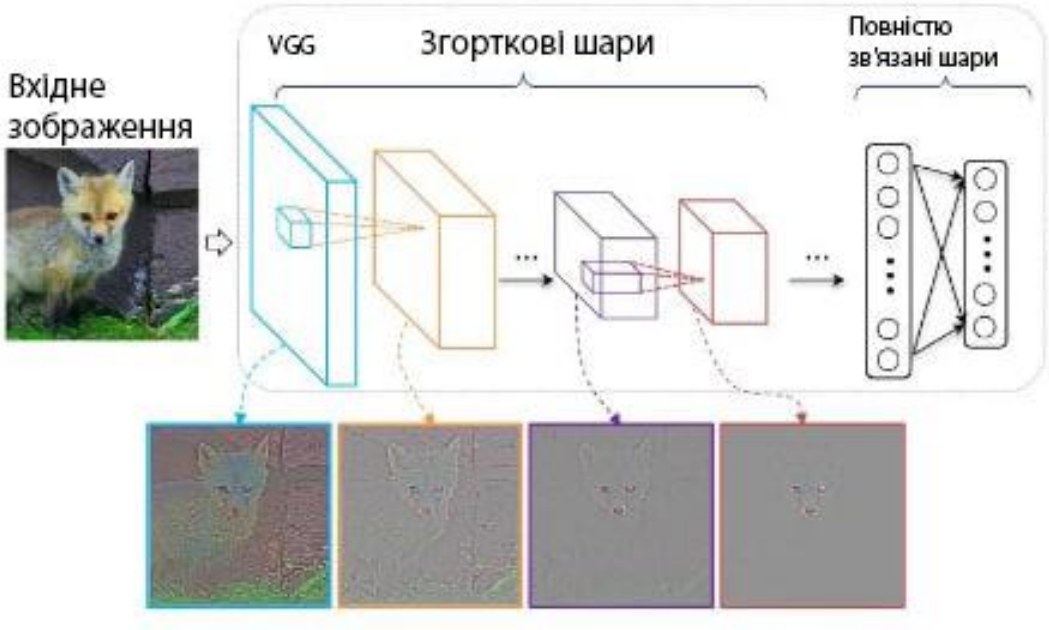

Рисунок 3.3 — Принцип дії нейронної мережі VGG19

Згорткові шари: згорткові шари VGG використовують мінімальне сприйнятливе поле, тобто 3×3, найменший можливий розмір, який усе ще захоплює вгору/вниз і вліво/вправо. Крім того, існують також згорткові фільтри 1×1, які діють як лінійне перетворення вхідних даних.

Далі йде блок ReLU, який є величезною інновацією від AlexNet, яка скорочує час навчання. ReLU означає функцію активації випрямленої лінійної установки; це кусково-лінійна функція, яка буде виводити вхідні дані, якщо вони позитивні; інакше вихід дорівнює нулю. Крок згортки фіксується на 1 пікселі, щоб зберегти просторову роздільну здатність після згортки (крок — це кількість піксельних зрушень над вхідною матрицею). Приховані шари: усі приховані шари в мережі VGG використовують ReLU. VGG зазвичай не використовує нормалізацію локальної реакції, оскільки це збільшує споживання пам’яті та час навчання. Крім того, це не покращує загальну точність. Повністю підключені рівні: VGGNet має три повністю

підключені рівні. З трьох рівнів перші два мають по 4096 каналів кожен, а третій має 1000 каналів, по 1 для кожного класу.

Архітектура мережі базується на двох основних обчисленнях:

1. Лінійно випрямлена згортка з фільтрами розміру *3k*, де *k* — кількість вхідних карт ознак. Крок і відступ згортки дорівнює одиниці, так що вихідна карта об’єктів має ті самі просторові розміри, що й вхідні карти об’єктів.

2. Максимальне об’єднання в регіонах, що не перекриваються, що зменшує вибірку карт функцій у два рази.

Ці два обчислення застосовуються по черзі. Кілька згорткових шарів супроводжується шаром максимального об’єднання. Після кожного з перших трьох шарів об’єднання кількість карт функцій подвоюється. Разом із просторовою дискретизацією це перетворення призводить до зменшення загальної кількості відповідей ознак у два рази. На рисунку 3.4 наведено схематичний огляд архітектури мережі та кількості карт функцій у кожному шарі. Оскільки я використовую лише згорткові шари, вхідні зображення можуть бути як завгодно великими. Перший згортковий шар має той самий розмір, що й зображення, а для наступних шарів співвідношення між розмірами карти функцій залишається фіксованим. Як правило, кожен рівень у мережі визначає нелінійний банк фільтрів, складність якого зростає разом із положенням рівня в мережі.

Навчена згорточна мережа є загальнодоступною, і її можливість використання для нових програм підтримується caffe-framework [12]. Для генерації текстури я виявив, що заміна операції максимального об’єднання на об’єднання середнього покращує градієнтний потік і отримує трохи чіткіші результати, тому зображення, наведені нижче, створено з об’єднанням середнього. Нарешті, з практичних міркувань я перемасштабував iights у мережі таким чином, щоб середня активація кожного фільтра для зображень і позицій дорівнювала одиниці. Таке перемасштабування завжди можна виконати без зміни виходу нейронної мережі, якщо мережа повністю лінійна по частинах 1.

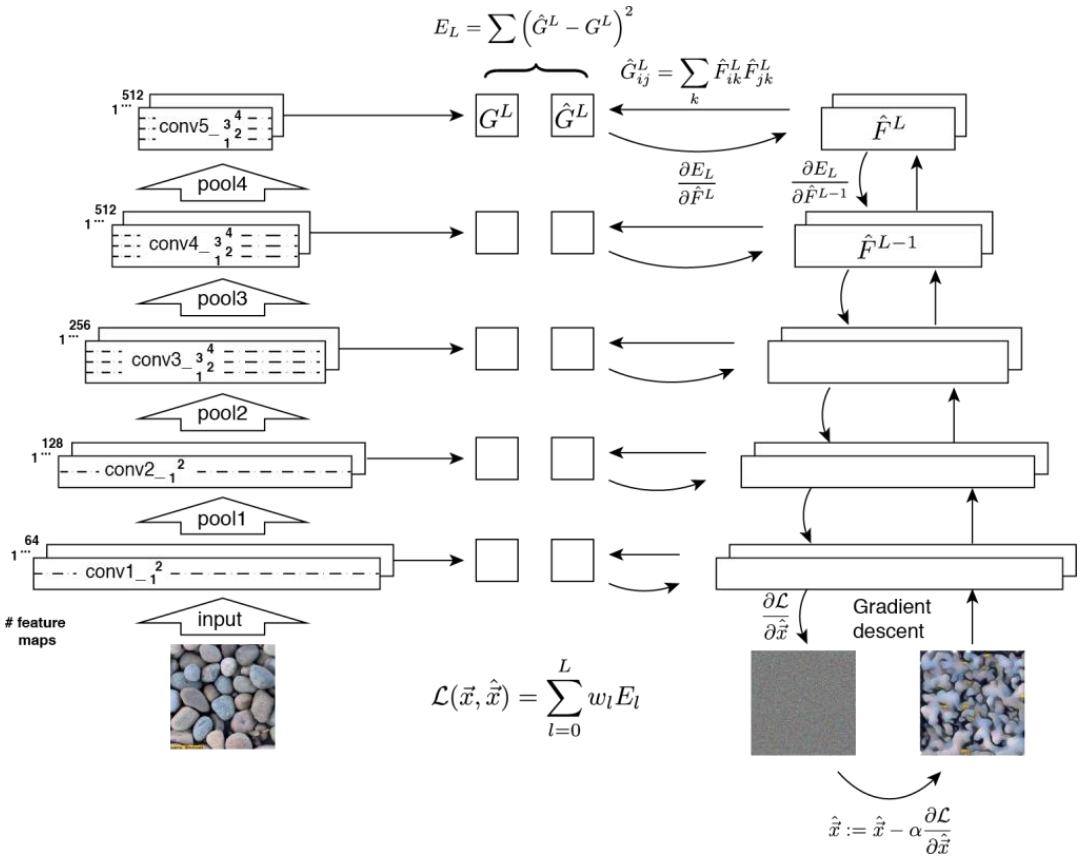

Рисунок 3.4 — Метод трансформації

3.2 Модель текстури

Модель текстури, яку я описую нижче, багато в чому відповідає дусі запропонованої Портілла та Сімончеллі. Щоб створити текстуру з заданого вихідного зображення, я спочатку однорідно витягую з цього зображення елементи різних розмірів. Далі я обчислюю просторову зведену статистику щодо відповідей функцій, щоб отримати стаціонарний опис вихідного зображення (рис. 3.2). Нарешті я знаходжу нове зображення з таким самим стаціонарним описом, виконуючи градієнтний спуск на випадковому зображенні, яке було ініціалізовано білим шумом (рис. 3.2).

Основна відмінність від роботи Портілли та Сімончеллі полягає в тому, що замість використання лінійного банку фільтрів і набору ретельно

підібраних зведених статистичних даних я використовую простір функцій, наданий високопродуктивною глибокою нейронною мережею, і лише одну просторову зведену статистику: кореляції між характеристики відповідей на кожному рівні мережі.

Щоб охарактеризувати задану векторизовану текстуру $\vec{\square}$ у нашій моделі, і спочатку пропускає $\vec{\square}$ через згортову нейронну мережу та обчислює активації для кожного шару l у мережі. Оскільки кожен рівень у мережі можна розуміти як нелінійний банк фільтрів, його активації у відповідь на зображення утворюють набір відфільтрованих зображень (так звані карти функцій). Шар з Nl різними фільтрами має Nl карт властивостей розміром Ml, кожна з яких векторизована. Ці карти функцій можна зберігати в матриці Fl ∈ RNl×Ml, де

l — активація j-го фільтра в позиції k у шарі l. Текстури за визначенням стаціонарні,

тому модель текстури повинна бути агностиком щодо просторової інформації. Підсумкова статистика, яка відкидає просторову інформацію в картах об’єктів, дається за допомогою кореляції між відповідями різних об’єктів. Ці кореляції ознак, з точністю до константи пропорційності, задані

Матриця Грама Gl ∈ RNl×Nl, де Gl є скалярним добутком карти ознак:

$$\square\square = \sum \quad \square\square * \square\square \qquad (3.1)$$

Де $\square\square$ - скалярний добуток карти ознак I,

$\square\square$, $\square\square$ - шар j та i.

Набір матриць Грама G1, G2, ..., GL з деяких шарів 1, . . . , L у мережі у відповідь на задану текстуру забезпечує стаціонарний опис текстури, який повністю визначає текстуру в нашій моделі (рис. 3.1).

3.3 Генерація текстури

Щоб створити нову текстуру на основі заданого зображення, я використовую градієнтний спад із зображення білого шуму, щоб знайти інше зображення, яке відповідає представленню матриці Грама вихідного зображення. Ця оптимізація виконується шляхом мінімізації середньоквадратичної відстані між елементами матриці Грама вихідного зображення та матриці Грама генерованого зображення.

Нехай $\underline{\square}$ і $\widehat{\square}$ – вихідне зображення та зображення, яке було згенеровано, а $\square^{\square}$ і $\widehat{\square^{\square}}$ – їхні відповідні представлення матриці Грама в шарі l. Градієнти l і, таким чином, градієнт $L(\underline{\square}, \widehat{\square})$ щодо пікселів $\widehat{\square}$ можуть бути легко обчислені за допомогою стандартної помилки зворотного поширення [18]. Градієнт $\partial L \; \partial \underline{\square}$ можна використовувати як вхідні дані для певної чисельної стратегії оптимізації. У нашій роботі я використовую L-BFGS, який здається розумним вибором для задачі оптимізації великої розмірності. Вся процедура в основному базується на стандартному проході вперед-назад, який використовується для навчання згорткової мережі. Таким чином, незважаючи на велику складність моделі, генерація текстури може бути виконана в прийнятний час за допомогою графічних процесорів і оптимізованих за продуктивністю наборів інструментів для навчання глибоких нейронних мереж [12].

3.4 Результати перетворення текстур

З чотирьох різних вихідних зображень нашою моделлю створені текстури (рис.3.5).

Рисунок 3.5 - Створені стимули.

Кожен рядок зображень було створено з використанням зростаючої кількості шарів у моделі текстури, щоб обмежити спуск градієнта (мітки на малюнку вказують на самий верхній включений шар). Іншими словами, для термінів втрат вище певного шару встановлено wl = 0, тоді як для членів втрат нижче та включаючи цей шар я встановлюю wl = 1. Наприклад, зображення в першому рядку ('conv1 1') генерується лише з представлення текстури першого рівня ('conv1 1') мережі VGG. Зображення у другому рядку («pool1») були створені шляхом спільного зіставлення представлень текстури поверх шару «conv1 1», «conv1 2» і «pool1». Таким чином я отримую текстури, які показують, яку структуру природних текстур охоплюють певні етапи обчислювальної обробки моделі текстури.

Перші три стовпці показують зображення, створені з природних текстур. Обмеження всіх шарів до шару «pool4» створює складні природні текстури, які майже не відрізняються від оригінальної текстури (рис. 3.6, п’ятий стовпчик). Навпаки, якщо обмежити лише кореляцію ознак на шарі loist, текстури містять мало структури та недалекі від спектрально узгодженого шуму.

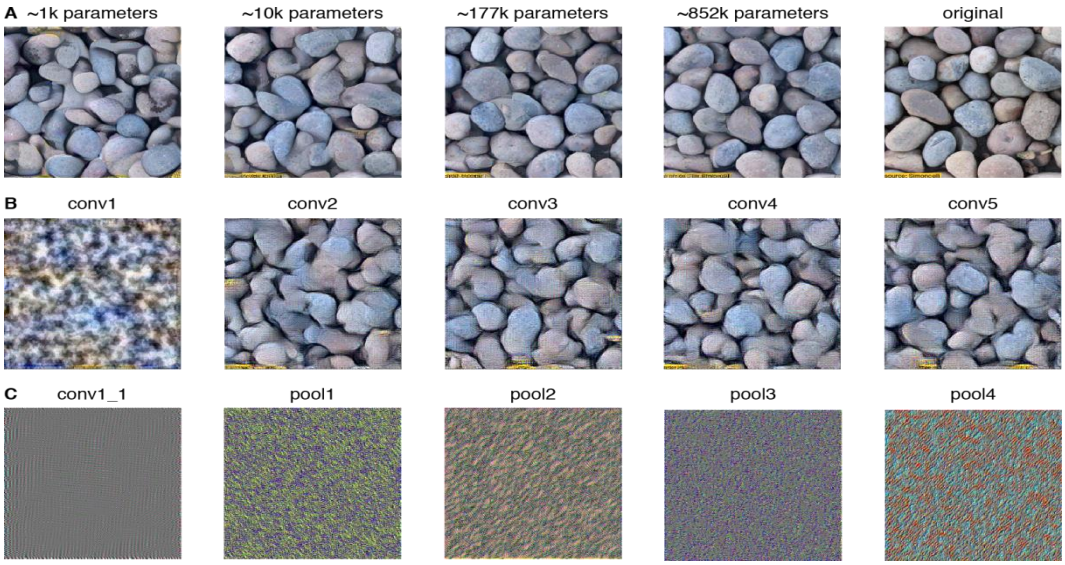

Рисунок 3.6 — Кількість параметрів у моделі текстури.

Кожен рядок відповідає різному етапу обробки в мережі. Якщо лише обмежити представлення текстури на шарі loist, синтезовані текстури мають невелику структуру, подібно до спектрально узгодженого шуму. Зі збільшенням кількості шарів, на яких я зіставляю представлення текстури, я бачу, що я створюю зображення зі зростаючим ступенем природності (рядки 2–5; мітки ліворуч вказують на найвищий включений шар). Вихідні текстури в перших трьох колонках раніше використовували Портілла та Сімончеллі [20]. Для кращого порівняння я також показую їхні результати (останній рядок). В останньому стовпці показано текстури, згенеровані з нетекстурного зображення, щоб краще зрозуміти, як модель текстури представляє інформацію про зображення.

Виявлено, що статистична структура природних зображень зіставляється у зростаючому масштабі, оскільки збільшується кількість шарів, які я використовую для створення текстури. Я не включив жодних шарів над шаром «pool4», оскільки це не покращило якість синтезованих текстур. Для порівняння я використовував вихідні текстури, а також показано результати їх моделі текстур (рис. 3.3, останній рядок).

Для кращого уявлення про те, як працює трансформація текстури, а також показані текстури, згенеровані з нетекстурного зображення, взятого з набору перевірки ImageNet (рис. 3.3, останній стовпець). Наш алгоритм створює текстуровану версію зображення, яка зберігає локальну просторову інформацію, але відкидає глобальне просторове розміщення зображення. Розмір областей, у яких зберігається просторова інформація, збільшується разом із кількістю шарів, які використовуються для створення текстури. Цю властивість можна пояснити збільшенням розмірів рецептивного поля одиниць над шарами глибокої згорткової нейронної мережі.

При використанні зведеної статистики з усіх рівнів згорткової нейронної мережі кількість параметрів моделі дуже велика. Для кожного шару з Nl картами функцій я підбираю Nl (Nl + 1)/2 параметрів, тому, якщо я використовую всі шари до «pool4» включно, наша модель матиме 852 тис. параметрів (рис. 3.4, четвертий стовпець). Я вважаю, що ця модель текстури сильно параметризована. Насправді, якщо використовувати лише один рівень на кожній шкалі в мережі (тобто «conv1 1» і «pool1-4»), модель містить 177 тисяч параметрів, майже не втрачаючи якості (рис. 3.4, третій стовпець). Я можу додатково зменшити кількість параметрів, виконавши PCA вектора ознак на різних рівнях мережі, а потім побудувавши матрицю Грама лише для перших k головних компонентів. Використовуючи перші 64 основні компоненти для шарів «conv1 1» і «pool1-4», я можу додатково зменшити модель до 10 тисяч параметрів (рис. 3.4, другий стовпець). Цікаво, що обмеження лише середніх значень карти функцій у шарах «conv1 1» і «pool1-4» (1024 параметри) вже створює цікаві текстури (рис. 3.4, перший стовпець).

Ці спеціальні методи зменшення параметрів показують, що представлення текстури можна сильно стиснути з невеликим впливом на якість сприйняття синтезованих текстур. Пошук мінімального набору параметрів, який відтворює якість повної моделі, є цікавою темою поточних досліджень. Більшу кількість природних текстур, синтезованих за допомогою моделі параметрів 177k, можна знайти в Додатковому матеріалі, а також на нашому ibsite3. Там також можна спостерігати деякі збої моделі у випадку дуже правильних штучних конструкцій (наприклад, цегляних стін).

Загалом я вважаю, що глибока архітектура мережі VGG з малими згортковими фільтрами, здається, особливо погано підходить для цілей генерації текстур. При виконанні такого ж експерименту з еталонною мережею caffe, яка дуже схожа на AlexNet [15], якість згенерованих текстур знижується двома способами. По-перше, статистична структура вихідної текстури не повністю узгоджується навіть при використанні всіх обмежень (рис. 3.4, «conv5»). По-друге, спостерігається штучна сітка, яка накладає згенеровані текстури (рис. 3.4). Я вважаю, що штучна сітка походить від більших розмірів рецептивного поля та кроків у довідковій мережі caffe.

У той час як результати довідкової мережі caffe показують, що архітектура мережі важлива, вивчені простори функцій однаково важливі для створення текстур. Під час синтезу текстури з мережею з архітектурою VGG, але випадковими світлами, генерація текстури не вдається (рис. 3.3), що підкреслює важливість використання навченої мережі.

Щоб краще зрозуміти особливості нашої текстури в контексті початкового завдання мережі з розпізнавання об’єктів, я оцінив, наскільки погана ідентичність об’єкта може бути лінійно декодована з особливостей текстури на різних рівнях мережі. Для кожного шару я обчислив представлення матриці Грама кожного зображення в наборі для навчання ImageNet і навчив лінійний класифікатор soft-max для передбачення ідентичності об’єкта. Оскільки я не зацікавлений в оптимізації ефективності передбачення, я не використовував жодного доповнення даних і тренувався

та тестувався лише на 224 224 центральних кадрах зображень. Я обчислив точність цих лінійних класифікаторів на перевірочному наборі ImageNet і порівняв їх із продуктивністю оригінальної мережі VGG-19, також оціненою на 224 224 центральних кадрах зображень перевірки.

Аналіз показує, що наше представлення текстури постійно роз’єднує інформацію про ідентифікацію об’єкта (рис. 3.4). Ідентифікація об'єкта може декодуватися все більш погано через шари. По суті, лінійне декодування з фіналу шар об'єднання працює майже так само погано, як вихідна мережа, що свідчить про те, що наше представлення текстури зберігає майже всю інформацію високого рівня. На перший погляд це може здатися дивним, оскільки подання текстури не обов’язково зберігає глобальну структуру об’єктів у текстурних зображеннях (рис. 3.3, останній стовпець). Я вважаю, що ця послідовність шар об'єднання працює майже так само погано, як вихідна мережа, що свідчить про те, що наше представлення текстури зберігає майже всю інформацію високого рівня. На перший погляд це може здатися дивним, оскільки подання текстури не обов’язково зберігає глобальну структуру об’єктів у текстурних зображеннях (рис. 3.3, останній стовпець).

Я вважаю, що ця «невідповідність» насправді очікувана і може дати розуміння того, як ЗНМ кодує ідентичність об’єкта. Згорткові представлення в мережі еквівалентні до зсуву, а завдання мережі (розпізнавання об’єктів) не залежить від просторової інформації, тому я очікую, що інформацію про об’єкт можна зчитувати незалежно від просторової інформації в картах функцій. Я показую, що це справді так: лінійний класифікатор на матриці Грама шару «pool5» наближається до продуктивності повної мережі (87,7% проти 88,6% точності топ-5, рис. 3.4).

ВИСНОВКИ

У представленій кваліфікаційній роботі було розглянуто методи комп’ютерного зору та методів машинного навчання для розробки моделі, що виконує перетворення 3D текстур.

Для досягнення поставленої мети в роботі були поставлені та вирішені такі завдання:

1) проведено огляд методів комп’ютерного зору та машинного навчання;

2) проаналізовано існуючі методи опису та визначення текстур і текстонів зображень, обґрунтовано використання згорткових нейронних мереж;

3) представлено архітектуру згорткової нейронної мережі перетворення 3D зображеннь.

Згорткові мережі є одними з найуспішніших алгоритмів розпізнавання зображень, доступних сьогодні.

Моделі, що реалізують відповідну архітектуру, займають перші місця в конкурсах алгоритмів розпізнавання, таких як ImageNet: згорткові мережі розпізнають рукописні цифри вибірки MNIST з помилкою 0,23%, обличчя людини – з помилкою 2,4%; результат конкурсу ImageNet для переможця 2014 року−Згорточна мережа Google GoogLeNet становить 0,06656% хибного розпізнавання, що є найкращим результатом на даний момент і порівняно з людською помилкою у зразку ImageNet.

Серед недоліків можна виділити труднощі в обробці дрібних об'єктів і нездатність впоратися з такими спотвореннями, як фільтр розмивання або сильний шум (такі спотворення присутні в навколишньому світі, наприклад, при погляді через товсте скло).

У той же час згорткові мережі відносно легко справляються з проблемами високоточного розпізнавання, які викликають труднощі у

людини, наприклад, розпізнавання певних моделей автомобілів або порід собак, і іншими завданнями, що вимагають виділення вузькоспеціальних ознак.

Однією з головних особливостей згорткових мереж є те, що така модель не має інформації про те, як саме локалізований шуканий об'єкт - його конкретне розташування та орієнтація в просторі.

У той же час, при вирішенні прикладних завдань управління та обробки інформації знання параметрів локалізації є необхідною умовою– залежно від розташування або положення об'єкта система обробки інформації може по-різному класифікувати зображення відповідно до поставлених перед нею завдань.

З іншого боку, без знання параметрів локалізації згорточна мережа вразлива до узагальненого класу проблем цілісності, тобто модель, навчена деякими локальними особливостями зображеного об’єкта, позитивно класифікуватиме штучні зображення, де відповідні елементи розташовані в хаотичному порядку. порядок - помилка, яку легко уникає людина.

Операція згортки забезпечує інваріантність по відношенню до трансляції локальних ознак (зрушення по осях x і y).

Архітектура згорткових мереж не забезпечує стійкості до інших афінних перетворень, таких як обертання, дзеркальне відображення та масштабування.

Для вирішення цієї проблеми зазвичай використовують евристичні методи (вирівнювання зображення по лінії горизонту, використання просторових пірамід і різних відображених копій оригіналу).

Перетворення, які не є афінними до плоского зображення, такі як обертання об’єкта в 3D-площині, так само не можуть бути оброблені за допомогою згортки. Для об’єктів, представлених у різних орієнтаціях, модель має вивчати різні набори ознак, що не перекриваються.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Bhadeshia H.K.D.H. (1999). Нейронні мережі в матеріалознавстві. ISIJ International 39 (10): 966–979. doi:10.2355/isijinternational.39.966.

2. М., Бішоп, Крістофер (1995). Нейронні мережі для розпізнавання образів. Clarendon Press. ISBN 0198538499. OCLC 33101074.

3. Цибенко, Г.В. (2006). Апроксимація суперпозицією сигмоїдальної функції. У ван Шуппен, Ян Х. Математика управління, сигналів і систем. Springer International. с. 303–314. PDF (англ.)

4. Дьюдні, А. К. (1997). Так, у нас немає нейтронів: захоплююча екскурсія перипетіями поганої науки. Нью-Йорк: Wiley. ISBN 9780471108061. OCLC 35558945. (англ.)

5. Дуда, Річард О.; Гарт, Пітер Елліот; Лелека, Девід Г. (2001). Класифікація патернів (вид. 2). Wiley. ISBN 0471056693. OCLC 41347061. (англ.)

6. Егмонт-Петерсен, М.; де Ріддер, Д.; Гендельс, Г. (2002). Обробка зображень за допомогою нейронних мереж – огляд. Розпізнавання образів 35 (10): 2279–2301. doi:10.1016/S0031-3203(01)00178-9. (англ.)

7. Герні, Кевін (1997). Знайомство з нейронними мережами. UCL Press. ISBN 1857286731. OCLC 37875698. (англ.)

8. Хайкін, Саймон С. (1999). Нейронні мережі: комплексна основа. Прентіс Холл. ISBN 0132733501. OCLC 38908586. (англ.)

9. Фальман, С.; Lebiere, C (1991). Каскадно-кореляційна навчальна архітектура. створена для Національного наукового фонду, номер контракту EET-8716324, і Агентства передових оборонних досліджень (DOD), наказ ARPA № 4976 згідно з контрактом F33615-87-C-1499. (англ.)

10. Герц, Дж.; Палмер, Річард Г.; Крог, Андерс С. (1991). Введення в теорію нейронних обчислень. Аддісон-Уеслі. ISBN 0201515601. OCLC 21522159.</ref> (англ.)

11. Лоуренс, Жанетт (1994). Вступ до нейронних мереж: дизайн, теорія та застосування. Каліфорнійське наукове програмне забезпечення. ISBN 1883157005. OCLC 32179420. (англ.)

12. Теорія інформації, логічний висновок та алгоритми навчання. Cambridge University Press. ISBN 9780521642989. OCLC 52377690. (англ.)

13. Маккей, Девід, Дж. К. (2003). Теорія інформації, логічний висновок і алгоритми навчання. Cambridge University Press. ISBN 9780521642989. (англ.)

14. Мастерс, Тімоті (1994). Обробка сигналів і зображень за допомогою нейронних мереж: джерело C++. Дж. Вайлі. ISBN 0471049638. OCLC 29877717. (англ.)

15. Ріплі, Браян Д. (2007). Розпізнавання образів і нейронні мережі. Cambridge University Press. ISBN 978-0-521-71770-0. (англ.)

16. Зігельманн, Х.Т.; Зонтаг, Едуардо Д. (1994). Аналогові обчислення через нейронні мережі. Теоретична інформатика 131 (2): 331–360. doi:10.1016/0304-3975(94)90178-3. (англ.)

17. 1944-, Сміт, Мюррей, (1993). Нейронні мережі для статистичного моделювання. Ван Нострандт Рейнхольд. ISBN 0442013108. OCLC 27145760.</ref> Smith, Murray (1993) Neural Networks for Statistical Modeling, Van Nostrand Reinhold, ISBN 0-442-01310-8 (англ.)

18. Вассерман, Філіп Д. (1993). Передові методи в нейронних обчисленнях. Ван Нострандт Рейнхольд. ISBN 0442004613. OCLC 27429729. (англ.)

19. Крузе, Рудольф,; Боргельт, Крістіан; Клавонн, Ф.; Мовес, Крістіан; Штайнбрехер, Матіас; Хелд, Паскаль (2013). Обчислювальний інтелект : методологічний вступ. Спрингер. ISBN 9781447150121. OCLC 837524179. (англ.)

20. Боргельт, Крістіан (2003). Neuro-Fuzzy-Systeme : von den Grundlagen künstlicher Neuronaler Netze zur Kopplung mit Fuzzy-Systemen. Переглянути напр. ISBN 9783528252656. OCLC 76538146. (нім.)