Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження нейронних мереж в задачі тривимірної реконструкції
(тема)

Виконав:
студент 2 курсу, групи СШІм-19-1
Божченко О.В.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник доц. Магдаліна І.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Божченко Олексію Васильовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження нейронних мереж в задачі тривимірної реконструкції

затверджена наказом університету від 31 жовтня 20 20 р. № _____
2. Термін подання студентом роботи до екзаменаційної комісії 15 грудня 20 20 р.
3. Вихідні дані до роботи Науково-технічні публікації, дані відомих наукових проектів щодо розробки систем логістики, дані статей, результати експериментальних досліджень.

4. Перелік питань, що потрібно опрацювати в роботі 1. Аналіз предметної галузі та постановка задачі, 2. Проектування системи, 3. Програмна реалізація системи

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)_____
Рисунок 1 – Модель штучного нейрона, Рисунок 2 – Приклади функцій активації, Рисунок 3 – Проста одношарова штучна нейронна мережа, Рисунок 4 – Перший шар згорткової нейронної мережі з пулінгом, Рисунок 5 – Метод обробки країв початкового зображення, Рисунок 6 – Пулінг із зменшенням розмірності, Рисунок 7 – Порівняння реконструкцій об'єктів різними методами, Рисунок 8 – Ілюстрація знакової функції відстані, Рисунок 9 – Архітектура DISN, Рисунок 10 – Модуль вилучення локальних і глобальних ознак, Рисунок 11 – Архітектура субмережі для оцінки параметрів камери, Рисунок 12 – Порівняння стандартної та нечіткої растеризації, Рисунок 13 – Ймовірнісна карта трикутника, Рисунок 14 – Огляд архітектури Soft Rasterizer, Рисунок 16 – Двомірна та тривимірна симетрія об'єктів, Рисунок 17 – Архітектура SymmetryNet, Рисунок 18 – Порівняння реконструкцій SymmetryNet та інших методів, Рисунок 19 – Архітектура запропонованої моделі, Рисунок 20 – Схематична структура автоасоціатора, Рисунок 21 – Криві помилок на навчальних та тестових даних, Рисунок 22 – Проблема перенавчання мережі, Рисунок 23 – Результат

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Аналіз предметної галузі	доц. Магдаліна І.В..		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на дипломну роботу	01.11.2020	виконано
2	Аналіз предметної галузі і постановка завдання	10.11.2020 – 14.11.2020	виконано
3	Дослідження методів та технологій	15.11.2020 – 17.11.2020	виконано
4	Аналіз існуючих реалізацій	18.11.2020 – 19.11.2020	виконано
5	Розробка методу рішення поставленої проблеми	20.11.2020 – 22.11.2020	виконано
6	Створення імітаційної моделі	23.11.2020 – 30.11.2020	виконано
7	Тестування і опрацювання імітаційної моделі	01.12.2020 – 02.12.2020	виконано
8	Оформлення пояснювальної записки	03.12.2020 – 10.12.2020	виконано
9	Оформлення графічних матеріалів	11.12.2020 – 12.12.2020	виконано
10	Попередній захист	14.12.2020	виконано
11	Захист перед ЕК	16.12.2020	

Дата видачі завдання 01 листопада 20 20 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Записка пояснювальна: 88 с., 23 рис., 2 дод., 28 джерел.

АВТОЕНКОДЕР, ГЛИБОКЕ НАВЧАННЯ, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, ТРИВИМІРНА РЕКОНСТРУКЦІЯ, KERAS, PYTHON, PYTORCH, SUPERVISED LEARNING

У данній атестаційній роботі розглядається рішення задачі тривимірної реконструкції об’єкту з одного зображення за допомогою технологій штучного інтелекту, а саме згорткових та автоасоціативних нейронних мереж. Суть задачі полягає в відтворенні тривимірної моделі об’єкту, зображенного на рисунку, маючи лише одне таке його зображення. Для вирішення поставленої задачі було проаналізовано математичну модель задачі, розглянуто існуючи методи для тривимірної реконструкції із застосуванням нейронних мереж, проаналізовано їх переваги і недоліки, а також проаналізовано відповідну літературу та електронні ресурси.

Крім того, в цій роботі описується процес імітаційного моделювання, який включає в себе створення, навчання і використання таких нейроних мереж. Розглядається запропонована архітектура мережі, її переваги та недоліки та надається теоритичне обгрунтування вибору цієї архітектури. Проводиться порівняльний аналіз точності моделей нейроних мереж що використовуться для вирішення поставленої задачі.

Результатом атестаційної роботи є розроблена демоверсія інтелектуальної системи тривимірної реконструкції об’єктів з одного зображення.

РЕФЕРАТ

Записка пояснительная: 88 с., 23 рис., 2 прил., 28 источников.

АВТОЭНКОДЕР, ГЛУБОКОЕ ОБУЧЕНИЕ, СВЕРТОЧНЫЕ НЕЙРОННЫЕ СЕТИ, ТРЕХМЕРНАЯ РЕКОНСТРУКЦИЯ, PYTHON, KERAS, PYTORCH, SUPERVISED LEARNING

В данной аттестационной работе рассматривается решение задачи трехмерной реконструкции объекта с одного изображения с помощью технологий искусственного интеллекта, а именно сверточной и автоассоциативной нейронных сетей. Суть задачи состоит в воспроизведении трехмерной модели объекта, изображенного на рисунке, имея лишь одно такое его изображение. Для решения поставленной задачи была проанализирована математическая модель задачи, рассмотрены существующие методы для трехмерной реконструкции с применением нейронных сетей, проанализированы их преимущества и недостатки, а также проанализирована соответствующая литература и электронные ресурсы.

Кроме того, в этой работе описывается процесс имитационного моделирования, который включает в себя создание, обучение и использование нейронных сетей. Рассматривается предложенная архитектура сети, ее преимущества и недостатки и предоставляется теоретическое обоснование выбора этой архитектуры. Проводится сравнительный анализ точности разных архитектур нейронных сетей, применяющихся для решения рассматриваемой задачи.

Результатом аттестационной работы является разработанная демоверсия интеллектуальной системы трехмерной реконструкции объектов с одного изображения.

ABSTRACT

Explanatory note: 88 pages, 23 figures, 2 appendixes, 28 sources.

3D-RECONSTRUCTION, AUTOENCODER, CONVOLUTIONAL NEURAL NETWORKS, DEEPLEARNING, KERAS, PYTHON, PYTORCH, SUPERVISED LEARNING

This thesis focuses on the solution of the single view three-dimensional reconstruction problem using artificial intelligence technologies, namely convolutional and auto-associative neural networks. The essence of the problem is to reproduce a three-dimensional model of the object depicted in the figure, having only one such image. To solve this problem, the mathematical model of the problem was analyzed, the existing methods for three-dimensional reconstruction using neural networks were considered, their advantages and disadvantages were analyzed, and the relevant literature and electronic resources were taken into consideration.

In addition, this paper describes the process of simulation, which includes the creation, training and use of such neural networks. The proposed network architecture, its advantages and disadvantages are considered and the theoretical substantiation of the choice of this architecture are given. A comparative analysis of the accuracy of neural network models used for this task is also given.

The result of the work is a developed demo version of the intelligent system of three-dimensional objects reconstruction from a single image.

ЗМІСТ

ВСТУП

Разом із розвитком інформаційних технологій у широкому сенсі, в останні роки особливою проблемою є збір та обробка великої кількості інформації з метою покращення процесів, що відбуваються у складних інформаційних системах. Ця інформація може бути представлена практично у будь-якому вигляді, в залежності від специфіки системи, що розглядається. Одним із найбільш поширених видів такої інформації є графіка. Виникнення, збереження та обробка графічних даних є притаманною до цілої низки сфер діяльності: комп’ютерна графіка, медіа, аналіз медичних зображень, аналіз графічного контенту в соціальних мережах та інтернеті, самокеровані автомобілі, графіка в комп'ютерних іграх і кіноіндустрії, оборонна діяльність – графічні дані можна зустріти чи не в кожній промисловості. Виникнення таких даних породжує і нові проблеми їх обробки – навіть за останні 10 років з’явились абсолютно нові задачі, які потребували новітніх методів їх вирішення. Не дивно, що ціла область, що займається обробкою графічних даних, комп’ютерний зір, розвивається шаленими темпами. Задачі розпізнавання, семантичної сегментації зображень, фільтрації, детектування об’єктів, відновлення зображень – це лише найбільш популярні проблеми комп’ютерного зору. Як не дивно, та певна частина з них вирішувалася ще багато десятків років тому. Але, звичайно, виникають й відносно нові проблеми комп’ютерного зору.

Так, однією з них є проблема тривимірної реконструкції об’єктів з зображень. Об'ємна відбудова або 3D-реконструкція у широкому сенсі – це процес охоплення форми і зовнішнього вигляду реальних об'єктів, за якого вхідними даними є декілька зображень об’єкту. Дослідження 3D реконструкції завжди було важко досяжною метою. За допомогою об'ємної відбудови можна визначити 3D-контури будь-якого об'єкта, а також дізнатися

тривимірні координати будь-якої точки контуру. Об'ємна відбудова об'єктів є загально науковою проблемою і основною технологією широкого спектра областей, таких як автоматизоване проектування, комп'ютерна графіка, комп'ютерна анімація, медична візуалізація, віртуальна реальність, цифрові медіа і т. д. Суттю задачі є перетворення двовимірного зображення об’єкту на його тривимірну, тобто об’ємну, модель.

До недавнього часу ця задача вирішувалася за допомогою методів, які зараз вже прийнято називати традиційними методами комп’ютерного зору. Але, звичайно, в них була й ціла низька недоліків, таких як низька якість отриманої відбудови, висока вартість у сенсі часу відбудови, дуже висока залежність від домену зображення, модель якого будується, а у деяких випадках необхідність наявності декількох зображень даного об’єкту. Проте, в останні роки, водночас із швидким розвитком і використанням нейронних мереж, виникають багато нових, власно кажучи, нейромережевих методів тривимірної реконструкції об’єктів. В таких методах відбудова об’єкту формується як оптимізаційна задача, що вирішується на просторі так званої навчальної вибірки, у якій представлені парі тривимірних відбудов об’єктів та їх двовимірних представлень у виді звичайних зображень. Але, не зважаючи на високу якість відбудови такими методами, навіть в них є свої недоліки, такі як висока тривалість конкретної відбудови та необхідність в наявності цілої низки зображень об’єкту під різними кутами, що не завжди є можливим у областях, де отримання навіть одного такого зображення є дуже коштовним, а іноді й зовсім неможливим, як, наприклад, відбудова медичних зображень. Таким чином, беручи до уваги усе, що зазначено вище, актуальною для дослідження є тема нейромережевої тривимірної відбудови об’єктів з єдиного зображення.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Поняття проблеми тривимірної реконструкції

Під завданням тривимірної реконструкції зображень мається на увазі процедура відтворення поверхонь і форм просторових об'єктів на основі однієї або декількох фотографій, отриманих з різних ракурсів. Дане завдання є одним з фундаментальних завдань комп'ютерного зору нарівні з задачами класифікації, сегментації або детектування об’єктів. Задача тривимірної реконструкції викликає великий інтерес в науковому співтоваристві, адже вона має безліч практичних застосувань в таких областях, як архітектура, робототехніка, медицина, комп'ютерна графіка, візуалізація та анімація. Наприклад, інформація про ураження пацієнтів може бути представлена в тривимірному вигляді на комп'ютері, що пропонує новий і точний підхід в діагностиці, що може мати життєво важливе клінічне значення. Цифрові моделі рельєфу місцевості можуть бути відновлені за допомогою таких методів, як повітряна лазерна альтиметрія або радар із синтезованою апертурою. Однак, як правило, така реконструкція пов'язана з великими обчислювальними витратами і цілою низкою припущень щодо структури об’єкту, ракурсу його зображення або розташування камери, умов освітлення.

Довгий час для вирішення завдання тривимірної реконструкції зображень застосовувалися класичні методи без використання машинного навчання, які базуються на використанні стереоскопічних даних, тобто потребують використання стереоскопічної камери, апріорних геометричних властивостях об’єктів, перспективних спотвореннях об’єктів та характерних властивостях їх освітлення за затінення [1], [2].

Незважаючи на те, що дані методи дозволяють домогтися довільно непоганої якості реконструкцій, як для методів, що використовують

детерміновані алгоритми відбудови, у більшості з них є хоча б одне з наступних істотних обмежень: вимога значного числа зображень з різних кутів огляду, вимога добре відкаліброваних камер, необхідність виконання припущення про те, що зображення з різних ракурсів можуть бути успішно зіставлені між собою, а також ряду припущень про властивості поверхонь об'єктів на зображеннях. Так, розглядаючи ці недоліки більш детально, можна сказати, що використання багатьох зображень об’єкту з різних ракурсів не завжди є можливим, адже у деяких сферах використання подібних тривимірних реконструкцій, як, наприклад, аналіз медичних зображень, збір таких знімків може бути неможливим або ж коштувати велику кількість грошей та часу. Припущення ж щодо певних властивостей поверхонь об’єктів та їх структури також зовсім не часто виконуються у реальному житті. Описані обмеження істотно ускладнюють або унеможливлюють використання кожного з класичних підходів в певному ряді практичних сценаріїв, до того ж, навіть їх відносно високої якості реконструкції може бути недостатньо у сучасних задачах для того, щоб використовувати їх із користю.

Зовсім недавно, завдяки останнім досягненням і триваючим активним дослідженням в області глибинного навчання і появи таких наборів даних тривимірних моделей, як ShapeNet [3], до задачі тривимірної реконструкції зображень стало можливо підійти з використанням глибоких згорткових нейронних мереж та машинного навчання. На даний момент, вже існує кілька надихаючих наукових робіт про класифікацію [4] і реконструкцію [5], [6] просторових об'єктів, однак, навчання нейронних мереж в просторі тривимірних даних як і раніше залишається вкрай складним завданням в зв'язку великою розмірністю даних, що розглядаються.

Взагалі, методи тривимірної відбудови прийнято розподіляти на дві основні категорії: активні і пасивні методи реконструкції. Активні методи,

тобто методи даних діапазону, враховуючи карту глибини, реконструюють тривимірний профіль за допомогою чисельних методів і будують об'єкт на основі моделі. Ці методи активно взаємодіють з відновлюваних об'єктом, механічно або радіометричного. Простим прикладом механічного підходу є застосування глибиноміра (або щупа) для вимірювання відстані до обертового об'єкта, встановленого на оборотний столик. Більш часто застосовуються, радіометоди випускають випромінювання на об'єкт, а потім заміряють його відображену частина. Прикладами можуть служити рухомі джерела освітлення, застосування різнобарвного освітлення, лазерні далекоміри і лідар, мікрохвильові та ультразвукові датчики та інші методи тривимірного сканування. Пасивні ж методи 3D-реконструкції не впливають на відновлюваний об'єкт, а лише використовують датчик, щоб виміряти випромінювання відбите або випромінене поверхнею об'єкта, щоб отримати його тривимірну форму. Зазвичай, в якості сенсора застосовують фоторецептори камер, чутливі до видимого діапазону. На вхід алгоритму обробки подається або набір з декількох зображень (один, два або більше), або відеопотік. У такому випадку говорять про реконструкцію, засновану на зображеннях, результатом роботи якої є 3D-зображення.

На даний момент більшість одержуваних таким чином 3D-фотографій і моделей мають низьку якість, з високою кількістю окремо розташованих шумових фрагментів моделі. Багато в чому якість вихідної моделі залежить від якості серії вхідних зображень (рівня шуму, оптичних деформацій, відблисків), а також від складності самої вихідної моделі (наявність пустот і порожнин, опуклості, поверхонь що відбивають світло і т.д.). При реконструкції з використанням серії космічних знімків, велику роль відіграють різного роду атмосферні ефекти.

1.2 Штучні нейронні мережі

Математична модель нейронної мережі є концептуальною моделлю біологічної нейронної мережі і складається з пов'язаних між собою у різний спосіб шарів штучних нейронів, які організовують загальну активну систему і функціонально чинять вплив на роботу один одного. У більшості архітектур моделей штучних нейромереж активність нейрона визначається перетворенням зовнішнього сумарного сигналу інших нейронів на даний нейрон.

З моменту свого становлення штучних нейронних мереж розвивалися достатньо уособлено від звичайних методів, нерідко цілком змінюючи уявлення про предмет і проблематику теорії машинного навчання і розпізнавання образів, залишаючи значний вплив на теоретичний і методологічний метод цих областей знання. Через деякий час після розвитку базових моделей штучних нейронних мереж, відбувся значний поділ науки про нейромережі на види топологій архітектури мереж і методи навчання мереж. У більшій кількості архітектур штучних нейронних мереж функції активації нейронів є зафіксованими, а ваги є параметрами мережі. Деякі вхідні сигнали нейронів є зовнішніми входами сукупної мережі, а деякі виходи нейронів – виходами сукупної мережі.

Задача штучної нейромережі полягає в трансформації вектора сигналів у вихідний вектор сигналів, що здійснюється через операції над входом параметрами і топологією мережі. Штучний нейрон характеризується своїм поточним станом.

Тут можна провести аналогію з нервовими клітинами головного мозку людини, які можуть бути пошкоджені або неправильно правцювати [7]. Він має групу вагів – односпрямованих вхідних зв'язків, з'єднаних з виходами інших нейронів, присутня така частина, як аксон – вихідний зв'язок штучного

нейрона, з якого сигнал (високий чи низький за значенням) надходить на зв'язки наступних нейронів. Загальний вигляд штучного нейрона наведено на рисунку 1.1. Штучний нейрон десь схожий за властивостями на нейрон, що дані звичайному нейрону. Тут безліч вхідних сигналів, позначених x_1, x_2, ... , x_N, надходить на штучний нейрон. Ці вхідні сигнали, в сукупності позначаються вектором X, приходять до зв'язків біологічного нейрона. Кожен зв'язок характеризується величиною його вагою.

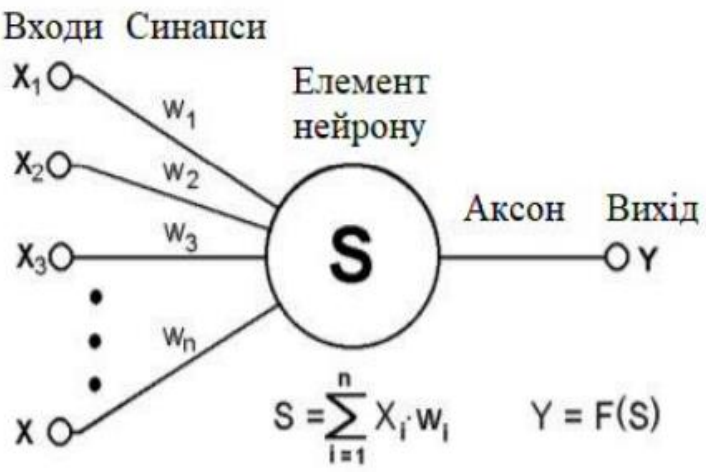

Рисунок 1.1 – Модель штучного нейрона

Для реалізації нелінійності при активації нейрона, його активність, крім різних видів суматорів і систем ваг на входах, визначається функцією одного аргументу – функцією активації. Нейрон в цілому реалізує скалярну функцію від векторного аргументу, а вихідний сигнал нейрона визначається видом функції активації і може бути дійсним або, у деяких випадках, цілим. Функція активації 15 застосовується до зваженої суми постсинаптичних сигналів на вході нейрона.

Тобто, активність нейрона повністю визначається його лише параметрами – вагами і його функцією активації. Є величезна кількість

складних трансформацій, що використовуються при розробці нейронних мереж. Вибір тієї або іншої активаційної функції залежить від умов завдання і топології мережі. Деякі з розглянутих нижче функцій застосовуються тільки в застарілих системах або в навчанні, але вважаються класичними і згадуються щоразу при вивченні штучних нейронних мереж, наведених на рисунку 1.2. Порогова функція Хевісайда [8] є найпростішою кусочно-лінійною передавальною функцією. Ця функція використовувалася в класичному персептроні, в даний час використовується в основному з метою навчання теорії нейронних мереж.

При використанні нескладної кусочнолінійної функції сигнал на виході нейрона лінійно пов'язаний зі зваженою сумою сигналів на його вході. На даний час лінійна функція майже не використовується на практиці. Сигмоїдальна функція активації є монотонно зростаючою повністю диференційованою S-образною нелінійною функцією. Вона впроваджує посилення слабких сигналів і запобігає насиченню сильних сигналів. Є однією з найбільш поширених передавальних функцій, часто використовується в нейронних мережах і сьогодні.

Введення сигмоїдальних функцій було зумовлено недостатньою гнучкістю класифікаторів на основі порогових передавальних функцій і дозволило перейти від жорсткої однорозрядної логіки до більш гнучкої поведінки і адаптивної параметризації нейронних мереж. Прикладом сигмоїдальной функції є логістична передавальна функція [10]. Функція гіперболічного тангенса відрізняється від логістичної кривої тим, що її область значень лежить в інтервалі (-1; 1), що в деяких випадках може спростити завдання навчання нейромереж. У неглибоких нейромережах використовуються нелінійні функції активації.

Часто зустрічаються різновиди сигмоїдальних і тангенціальних функцій є нелінійними, але на практиці при навчанні глибоких нейромереж такі

функції можуть привести до проблем із загасанням або збільшенням градієнтів. Функція ReLU є випрямленою лінійною функцією і на даний момент вважається набагато більш простим і ефективним з точки зору обчислювальної складності варіантом функції активації. Похідна цієї функції дорівнює або 0, або 1, від чого її застосування запобігає розростанню і загасанню градієнтів, і призводить до зменшення ваг, що позитивно позначається на обчислювальної здатності нейромережі.

Активаційна функція ReLU є одним з останніх успіхів в області методів налаштувань глибоких нейронних мереж. Сьогодні існує ціла родина різних модифікацій ReLU, які вирішують проблеми надійності цієї функції при проходженні через нейрон великих градієнтів: Leaky ReLU, Parametric ReLU, Randomized ReLU. Приклади активаційних функцій наведено на рисунку 1.2.

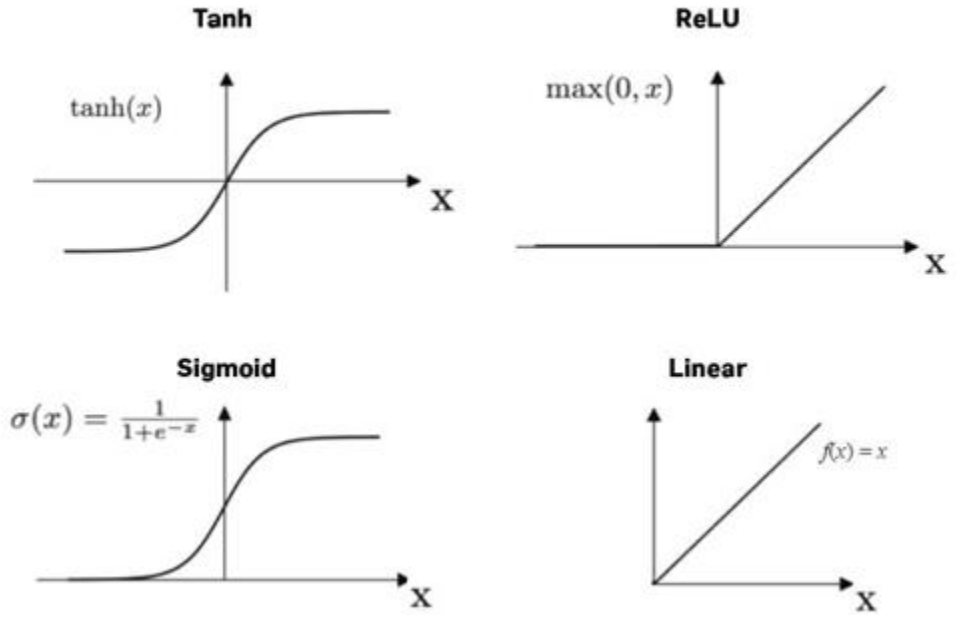

Рисунок 1.2 – Приклади функцій активації

Можливість простого розпізнавання доступна одному нейрону, проте з'єднання тих же нейронів в мережі є необхідною для отримання найкращого результату. Одношаровою називають нейронну мережу, яка складається з сукупності нейронів, утворення яких формує новій шар мережі. Ліві вершини схеми зображеного нейрону служать для розподілу сигналів, що подаються на вхід. За цими вершинами не закріплено жодних обчислень, тому вони не вважаються шаром і позначені колами, щоб відрізняти їх від обчислювальних нейронів, що позначаються квадратами. Існує сполучення окремою вагою кожен елементу з множини входів X з кожним штучним нейроном. За нейроном закріплена робота подачі зваженої суми входів в мережу. З метою спільності штучні та біологічні мережі мають відсутні з'єднання. Існують з'єднання між виходами і входами елементів в самому шарі. Ваги представлені елементами матриці *W*.

Матриця має n рядків і m стовпців, де n – число входів, а m – число нейронів. Наприклад, *w12* – це вага, що зв'язує перший вхід з другим нейроном. Таким чином, обчислення вихідного вектора *Y*, зводиться до матричному множенню *Y = XW*. Приклад простої одношарової нейронної мережі наведено на рисунку 1.3.

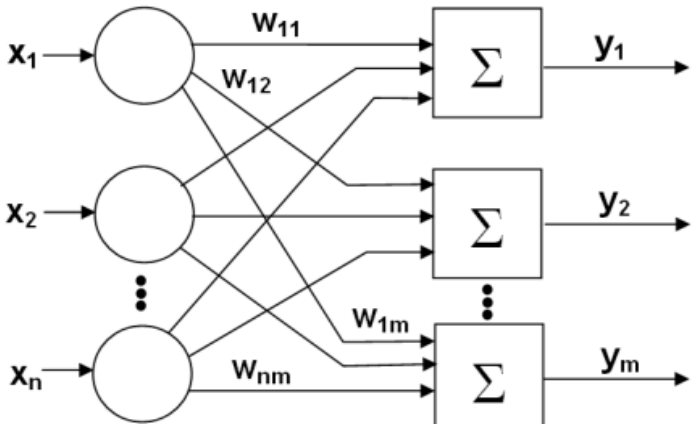

Рисунок 1.3 – Проста одношарова штучна нейронна мережа

1.2.1 Багатошарові нейронні мережі

Відмінності обчислювальних процесів в нейронних мережах часто обумовлені способом взаємозв'язків нейронів. За сукупністю критеріїв на сьогоднішній день багатошарові архітектури можна розділити на статичні і динамічні. Кожен з класів архітектур нейронних мереж може включати безліч підкласів, реалізуючи різні підходи, нижче будуть наведені основні з них. До статичних архітектур відносять мережі прямого поширення, в яких реалізована однонаправлений зв'язок між шарами, відсутні динамічні елементи і зворотній зв'язок, а вихід навченої нейромережі однозначно визначається входом і не залежить від попередніх станів мережі.

Статичні штучні нейронні мережі прямого поширення:

– персептрон;
– нейронна мережа Кохонена;
– когнітрон та неокогнітрон;
– сучасна згорткова нейронна мережа.

На противагу статичним архітектурам, існують динамічні архітектури штучних нейромереж, що реалізують рекурентну структуру з використанням зворотніх зв'язків, завдяки чому стан мережі в кожний момент часу залежить від попереднього стану. Рекурентні нейромережі як правило базуються на багатошаровому персептроні.

Елементарний персептрон організовується на основі сенсорних даних на вході та асоціативних елементів, і реагуючих елементів на виході. Набір S-блоків, пов'язаний з ним утворює асоціацію, і асоціативний активізується після того як досягнуто певне число сигналів від сенсорних елементів. A-елемент передає зважений сигнал на R-елемент обрахунку суми, і в залежності від того, чи перевищує зважена сума деякий поріг, видає результат роботи перцептрону.

Багатошаровий персептрон організовується з додатковими прихованими шарами А-елементів, розташованими між S-елементами і R-елементами. Принципова складність завдань, що вирішуються багатошаровим персептроном, є найвищою для класу персептронів. Навчання елементарного і багатошарового персептрона полягає в зміні вагових коефіцієнт зв’язків A-R.

Персептрон здатний працювати в режимі розпізнавання або узагальнення. Персептрон В одношаровому персептроны вхідні елементи безпосередньо пов'язані з вихідними за допомогою системи ваг, зв'язки S-A організовані за принципом однозначної відповідності. Одношаровий персептрон є окремим випадком класичного елементарного персептрону, найпростішою мережею прямого поширення – лінійним класифікатором, і має безліч принципових обмежень, таких як неможливість реалізації функції XOR. Когніторон був розроблений на основі будови біологічної зорової кори, має ієрархічну принципово багатошарову архітектуру.

Нейрони між шарами когнітрону пов'язані тільки локально, і кожен шар реалізує різні 20 рівні узагальнення: вхідні шари сприймають прості образи, такі як лінії, великі однорідні ділянки, їх орієнтацію і локалізацію в просторі вхідних даних, в той час як глибокі шари сприймають складніші абстрактні структури, незалежні від локалізації та інших простих ознак образу.

Когнітрон організовується з ієрархічно пов'язаних збудливих і гальмуючих шарів. Співвідношення збуджуючих і гальмуючих сигналів на вході нейрона визначає його стан збудження. Існують спрощені моделі когнітрону, що будуються з одновимірних шарів, але спочатку когнітрон конструювався як каскад двовимірних шарів [11]. Пресинаптичний простір сигналів визначає виходи попереднього шару, постсинаптичний простір – входи наступного шару або площини. Нейрон когнітрону сприймає не весь

постсинаптичний простір сигналів, а тільки його частину, реалізується принцип локальної зв'язності.

Область пресинаптичного простору сигналів, що утворюють постсинаптичний простір сигналів, що впливають на стан даного нейрона, називається його локальним рецептивних полем. Рецептивні поля близьких один до одного постсинаптичних нейронів, звані зонами конкуренції, перекриваються, тому активність даного пресинаптичного нейрона позначається на все більше поширення області постсинаптичних нейронів наступних шарів ієрархії. Розміри зон конкуренції обумовлюють кількість сприймання в просторі області ознак.

Неокогнітрон є прямим розвитком ідеї, що лежать в основі когнітрону і точніше моделює структуру зорової кори головного мозку і є класифікатором, здатним до кращого розпізнавання об’єктів. Кожен шар неокогнітрона складається з площини простих S-нейронів і площини складних С-нейронів, що також організують локальну зв'язність [12]. Локальне рецептивне поле на площині S-нейронів наступного шару формується пресинаптичними сигналами площини С-нейронів попереднього шару.

Локальні ознаки способу сприймаються S-нейронами, а спотворення локальних ознак компенсуються С-нейронами. В результаті 21 цього процесу кожен шар після вхідного має своїм входом все більш узагальнену картину, утворену С-нейронами попередніх шарів [13].

З кожним рівнем глибини первинні прості ознаки визначаються у більш складних з’єднаннях. Площину S-нейронів можна розглядати як один нейрон, ваги якого визначають ядро згортки, що застосовуються до попереднього шару у всіх можливих позиціях. Всі С-нейрони реагують на образ, відповідний ядру згортки, в їх рецептивному полі, тому він визначається інваріантно до його локалізації.

Сучасні глибокі згорткові нейронні мережі засновані на ідеях, що лежать в основі неокогнітрона, і сьогодні застосовуються для вирішення широкого кола завдань: від промислових, корпоративних та дослідницьких до повсякденно побутових, які вирішуються мобільними пристроями.

1.3 Згорткова нейронна мережа

Згорткові або конволюційні нейронні мережі (КНН) складаються з одного або декількох згорткових шарів спочатку, а потім один або декілька повністю з'єднаних шарів, як у стандартній багатошаровій нейронній мережі. Архітектура КНН призначена для використання двовимірної структури вхідного зображення (або іншого двовимірного входу, такого як мовний сигнал, наприклад). Це досягається за допомогою локальних зв'язків та зв'язних ваг, за якими йде певна форма об'єднання, що призводить до виникнення інваріантних особливостей (або ознак). Ще однією перевагою КНН є те, що їх простіше тренувати та вони мають набагато менше параметрів, ніж повністю зв'язані мереж з однаковою кількістю прихованих блоків [14]. На рисунку 1.4 показано повний шар КНН, який складається зі згорткових та підвибіркових підшарів.

Вхід до згорткового шару – це зображення $h \cdot w \cdot c$, де h – висота зображення, w ширина, а c – кількість каналів. Наприклад, RGB-зображення має канали = 3. Згортковий шар має k фільтрів (або ядер) розміром $h \cdot w \cdot c$, де h та w менше, ніж розмір зображення, а c може бути таким самим, як кількість каналів с або менше, і може змінюватися для кожного ядра. Розмір фільтрів дає початкову локально зв'язану структуру, кожен з них згортається з зображенням, для створення k мап ознак розміром $h - m + 1$ або $w - n + 1$.

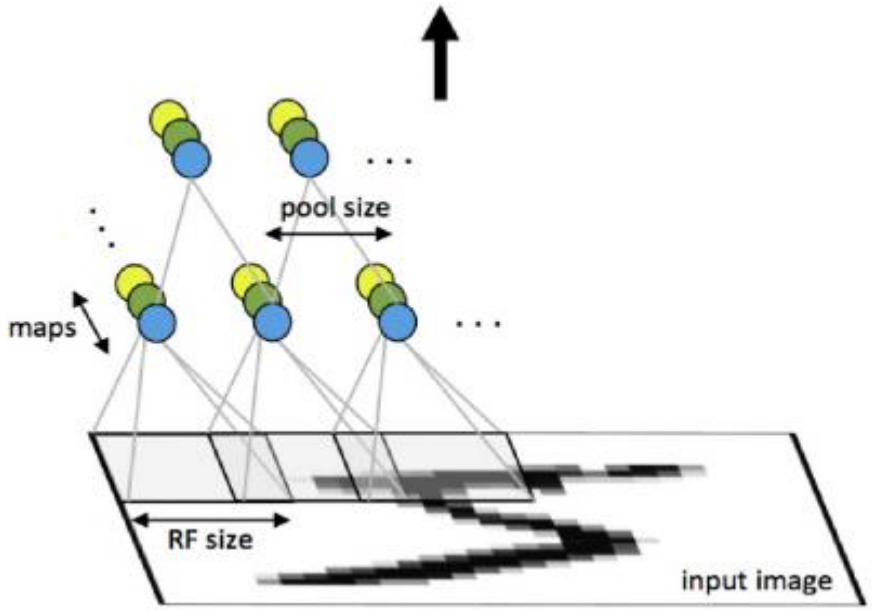

Рисунок 1.4 – Перший шар згорткової нейронної мережі з пулінгом

Кожна така мапа потім підвибірковується, зазвичай, з середнім або максимальним об'єднанням (пулінгом) по *p*p* сусідніх областей, де змінюється від 2 для малих зображень і, зазвичай, не більше 5 для більших входів. Як до, так і після шару підсемплінгу для мапи об'єкта застосовується сигмоідозалежна нелінійність.

Вхідними даними для ЗНМ у загальному випадку є RGB зображення фіксованого розміру N × N пікселів. Єдиним предобробним кроком, який необхідно виконати, є нормалізація зображення шляхом віднімання середнього значення RGB обчисленого на тренувальному наборі даних від кожного пікселя.

Зображення пропускається через набір згорткових шарів, де застосовуються фільтри із малим рецептивним полем 3 × 3 (найменший розмір для отримання поняття про верх-низ, ліво-право, центр). Згортковий шар являє собою набір карт (карти характеристик), у кожної карти є синаптичне ядро (скануюче ядро або фільтр). Кількість карт визначається

вимогами до задачі та архітектурою нейромережевої моделі. При збільшенні кількості карт збільшиться якість розпізнавання, але при цьому збільшується і обчислювальна складність. Пропонується пропустити зображення через набір згорткових шарів, де застосовуються фільтри із малим рецептивним полем 3 × 3 (найменший розмір для отримання поняття про верх-низ, ліво-право, центр). Крок конволюції зафіксовано у 1 піксель; просторовий відступ (padding) вхідних даних є таким, що просторова роздільна здатність зберігається після згортання, наприклад відступ складає 1 піксель для згорткових шарів розміром 3 × 3. Розмір карт можна визначити за формулою 1.1:

$$(w, h) = (mW - kW + 1, mH - kH + 1), \qquad (1.1)$$

де *(w, h)* – розмір згорткової карти, який обраховується;

mW – ширина попередньої карти;

mH – висота попередньої карти;

kW – ширина ядра (фільтру).

Ядро являє собою фільтр або вікно, яке ковзає по всій області попередньої кати і знаходить певні характеристики об'єкта. Якщо розміри ядро малі, то неможливо буде визначити характеристики. Якщо розміри ядра занадто великі, то збільшується кількість зв'язків між нейронами і відповідно складність обчислень. Також розмір ядра вибирається таким, щоб розмір карт згорткового шару був парним, що дозволяє не втрачати інформацію при зменшенні розмірності у шарі пуллінгу. Ядро є системою розділяємих вагів або синапсів.

У звичайній багатошаровій нейромережі дуже багато зв’язків між нейронами, тобто синапсів, що уповільнює процес детектування. У згортковій мережі – навпаки, загальні ваги дозволяють зменшити кількість зв'язків і

дозволити знаходити одну й ту ж характеристику по всій області зображення. Початково значення кожної карти згорткового шару рівні 0. Значення вагів ядер задаються випадково в області від 0,1 до 0,9. Ядро ковзає по попередній карті і виконує операцію згортування, яка часто використовується для обробки зображень. Оператор згортки вдає із себе робочу одиницю згорткового шару конволюційної нейронної мережі. Сам шар складається з набору таких ядер, які налаштовують свої ваги під час навчання і кожен з них навчається на певний тип патернів вихідного розподілу.

У згортковій мережі – навпаки, загальні ваги дозволяють зменшити кількість зв'язків і дозволити знаходити одну й ту ж характеристику по всій області зображення.

При цьому залежно від методу обробки країв початкової матриці результат може бути менше початкового зображення (valid), такого ж розміру (same) або більшого розміру (full), як показано на рисунку 1.5.

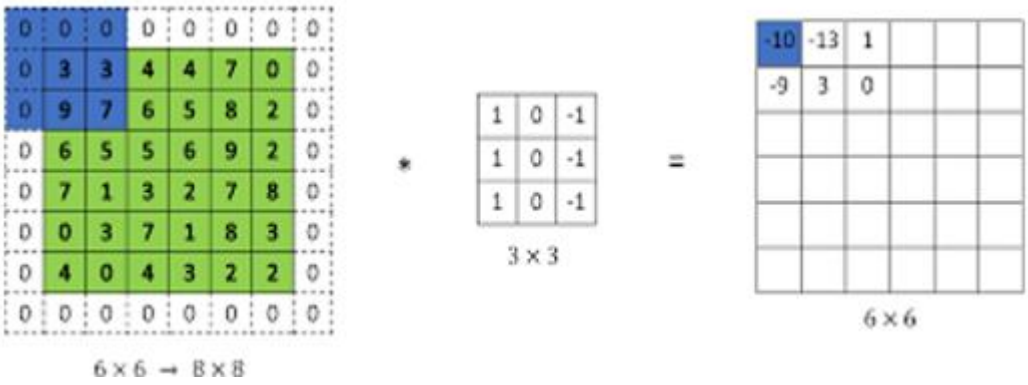

Рисунок 1.5 – Метод обробки країв початкового зображення

В спрощеному вигляді цей шар можна описати формулою 1.2:

$$x^l = f(x^{l-1} * k^l + b^l), \tag{1.2}$$

де x^l – вихід шару l;

$f(x)$ – функція активації;

b^l – коефіцієнт зсуву шару l;

* – операція згортки входу x із ядром k.

При цьому за рахунок крайових ефектів розмір початкових матриць зменшується:

$$x_j^l = f(\sum_i x^{l-1} * k^l + b^l), \quad (1.3)$$

де x^l_i – вихід шару l;

$f(x)$ – функція активації;

b^l – коефіцієнт зсуву шару l;

* – операція згортки входу x із ядром k.

Мета пулінгового шару – зменшення розмірності карт попереднього шару. Якщо на попередній операціях згортки уже виявлені деякі характеристики, то для подальшої обробки настільки детальне відображення уже не потрібне, і воно ущільнюється до менш детального. Фільтрація вже непотрібних деталей допомагає запобігти перенавчанню. В процесі сканування фільтрами шару пулінгу карти попереднього шару, скануюючий фільтр не перетинається на відміну від згорткового шару.

Просторовий пулінг (pooling) виконується п'ятьма шарами макспулінгу (maxpooling), які слідують за деякими згортковими шарами (не за всіма згортковими шарами слідують шари макспулінгу). Макспуллінг виконується з вікном розміром 2 × 2 пікселя та кроком 2 пікселя. Шар пулінгу зменшує розмірність простору незалежно в кожному зрізі по глибині вхідних даних. На рисунку 1.6 зображено вхідні дані розміру [224 x 224 x 64] агреговані із фільтром розміру 2, відступом 2 до вихідних даних розміром [112 x 112 x 64]. При цьому глибина масиву зберігається. Це

означає, що із чотирьох чисел (квадрату 2 х 2) вибирається одне максимальне. За блоком згорткових шарів (які мають різну глибину та архітектури) слідує три повністю з'єднаних (fully connected) шари: перших два мають 4096 канали кожен, третій є відповідальним за класифікацію і має 1000 каналів (по одному каналу на кожен клас). Останнім шаром є шар софтмакс (softmax) класифікатору. Всі приховані шари активуються нелінійною трансформацією ReLU, яка може бути визначена за наступною формулою 1.4.

$$A(x) = max(0, x) \tag{1.4}$$

де x – вхідний вектор i-го екземпляру.

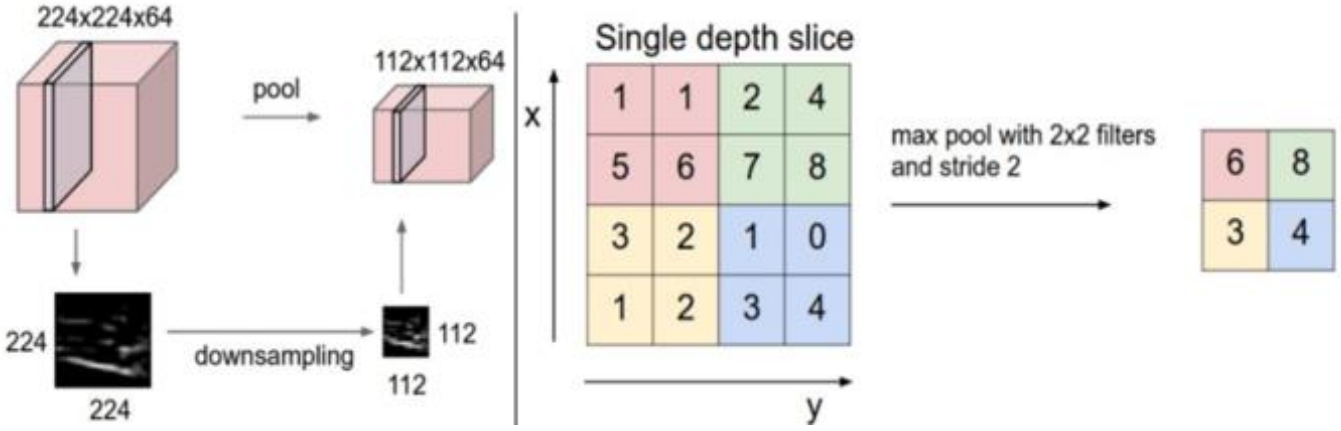

Рисунок 1.6 – Пулінг із зменшенням розмірності з 224 х 224 до 112 х 112

В даній мережі не використовується шари локальної нормалізації (LRN): оскільки така нормалізація не покращує продуктивність роботи згорткової нейромережевої моделі на виконанні наших завдань, але призводить до збільшення об'єму пам'яті, яка використовується та часу обчислень. Тобто нейрони кожної карти попереднього шару з'єднані із одним

нейроном прихованого шару. Фільтрація вже непотрібних деталей допомагає запобігти перенавчанню. В процесі сканування фільтрами шару пулінгу карти попереднього шару, скануюючий фільтр не перетинається на відміну від згорткового шару.

Таким чином число нейронів прихованого шару дорівнює числу карт попереднього шару, але зв'язки можуть бути необов'язково такими, наприклад, що тільки частина нейронів якоїсь із карт попереднього шару пов'язана із першим нейроном прихованого шару, а частина, що залишилася із другим, або всі нейрони першої карти пов'язані із нейронами першого та другого прихованого шару.

1.4 Огляд існуючих методів тривимірної реконструкції

За останні роки були запропонована безліч методів тривимірної реконструкції об’єктів, де методи, що базуються на навчанні глибоких нейронних мереж досягли особливо багатообіцяючих результатів. Для представлення тривимірних моделей, багато з них використовували так зване воксельне представлення чи хмари точок через відносну легкість представлення їх до нейронної мережі. Тим не менш, такі представлення часто є обмеженими у сенсі роздільної здатності. Приклади тривимірної реконструкції позначено на рисунку 1.7. Деякі з недавніх методів використовують явні представлення поверхні об’єктів у нейронній мережі, проте, вони використовують припущення про фіксовано топологію даних, що обмежує гнучкість таких систем. Нажаль, такі дистанційні функції впроваджують лише приблизні оцінки для порівняння схожості об’ємних форм [15].

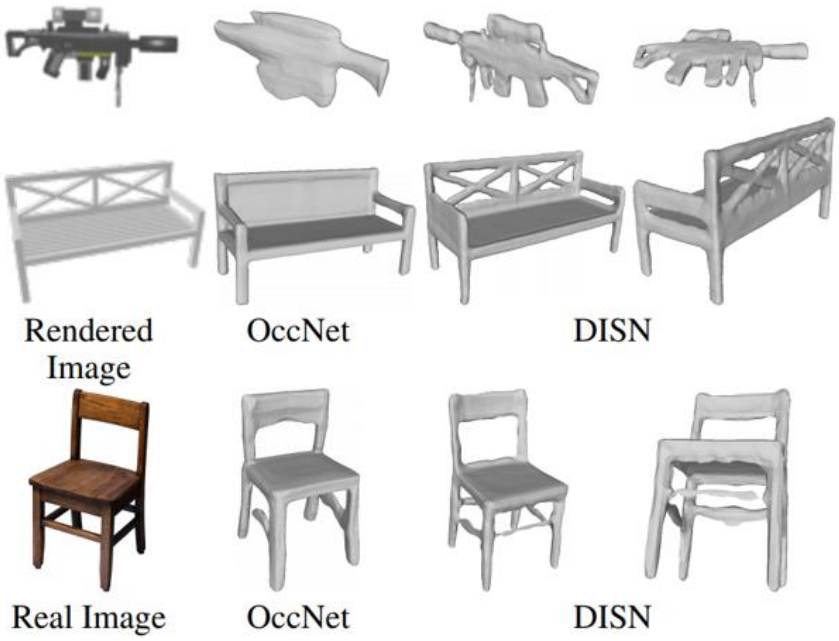

Рисунок 1.7 – Порівняння реконструкцій об’єктів різними методами

В той час як багато методів тривимірної реконструкції використовують стисненні представлення тривимірної форми, отримані з двовимірного зображення, вони схильні ігнорувати такі структурні явища як порожнини у об’єктах або тонкі деталі форми, що змиваються при використанні вище названих дистанційних функцій.

Такі маленькі деталі займають дуже мале місце в тривимірному представленні об’єкту і ігнорування їх не завдає великої шкоди для чисельних значень функції втрат у порівнянні з дійсною формою об’єкту, але можуть бути сильно помітними для людського ока, що призводить до, у дійсності, неякісного результату реконструкції. В останній час було проведені активні дослідження методів тривимірної реконструкції, що базуються на навчанні, які використовували як представлення вокселі, дерева, хмари точок та примітивні поверхні. Зовсім недавно, Сінх та інші запропонував генерування тривимірної моделі через прості геометричні фігури. Танг та

інші використав остови тривимірних моделей для реконструкції поверхні, однак, цей метод потребує організації додаткового набору даних із примітивними формами. Гроє презентував AtlasNet для генерації тривимірних поверхонь використовуючи набір параметризованих елементів поверхонь. Венг та інші запропонував графову нейронну мережу Pix2Mesh, де генерував модель об’єкту через перетворення базового мешу. Танг та інші використав остови тривимірних моделей для реконструкції поверхні, однак, цей метод потребує організації додаткового набору даних із примітивними формами.

1.4.1 Глибока неявна поверхнева нейронна мережа

У методі реконструкції, що використовує глибокі неявні поверхневі нейронні мережі (DISN) для вирішення проблеми відновлення тривимірної форми з складною структурою поверхні, використовується модуль локального вилучення ознак. А саме, оцінюються параметри точки погляду вхідного зображення. Ця інформація використовується для проекції кожної точки вхідного зображення на тривимірну модель.

Після вилучення таких місцевих ознак зображення, вони використовуються разом з іншими його картами представлення ознак для того щоб оцінити значення знакової дистанційної функції для тривимірних координат моделі. Такий модуль дозволяє нейронній мережі навчитися залежності між точками вхідного зображення та тривимірними координатами моделі, що значно покращує якість реконструкції для фігур складної поверхні.

Такий підхід був першим, що зміг досягти високої якості реконструкції на складних прикладах. Ціль метода формується так: маючи двовимірне зображення об’єкту, що необхідно реконструювати, треба відновити його

тривимірну форму як і загально, так і малі її деталі. Для моделювання використовується знакова дистанційна функція (SDF). Як зображено на рисунку 1.8, SDF – безперервна функція, яка виконує відображення даної тривимірної точки $p = (x, y, z) \in \mathbb{R}^3$ на область дійсних значень $s \in \mathbb{R} : s = SDF(p)$.

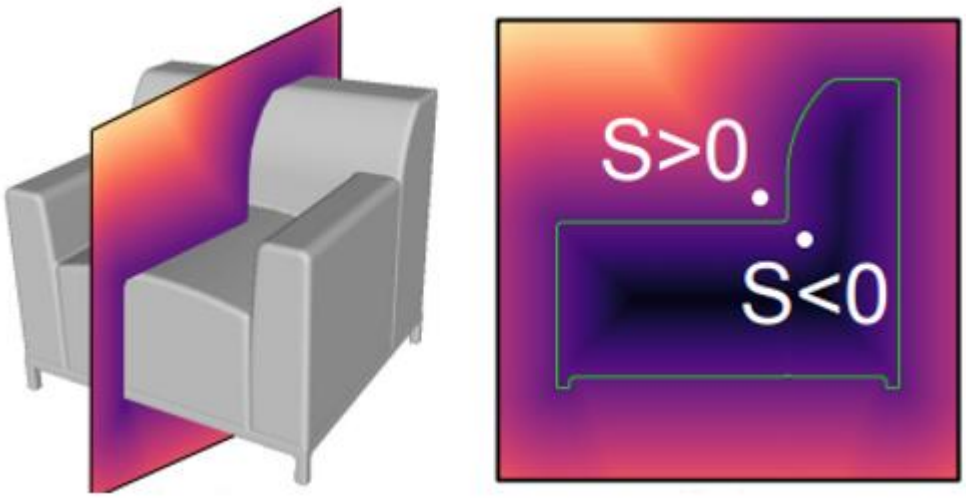

Рисунок 1.8 – Ілюстрація знакової функції відстані

Абсолютне значення s відповідає відстані точки до поверхні, а знак визначає лежить точка на поверхні, чи позанею. Ізо-поверхня $S_0 = \{\boldsymbol{p} | SDF(\boldsymbol{p}) = 0\}$ неявним чином репрезентує тривимірну поверхню. У цьому методі для відновлення тривимірної структури об'єкта використовується глибока штучна нейронна мережа прямого поширення, що надає можливість оцінити знакову дистанційну функцію з наданого зображення. DISN приймає єдине зображення як вхід і обчислює значення SDF для кожної точки простору.

На відміну від традиційних тривимірних згорткових нейронних мереж, які генерують тривимірну об'ємну сітку фіксованого розміру, глибока неявна

поверхнева нейронна мережа формує безперервне поле довільного розміру. Цього вдається досягти за рахунок модулю вилучення локальних ознак зображення. Огляд архітектури DISN запроваджено на рисунку 1.9.

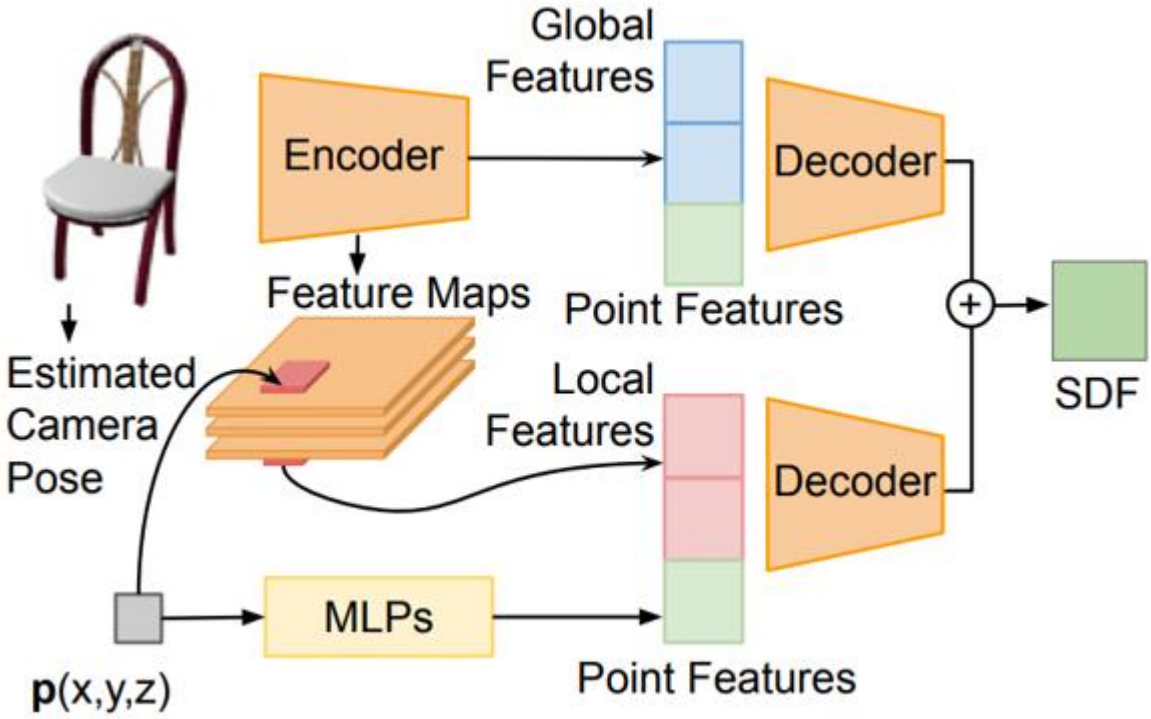

Рисунок 1.9 – Архітектура DISN

Для даного зображення, DISN складається з двох частин: оцінки ексцентричних параметрів камери(camera pose estimation) та прогнозування SDF. Маючи предсказані параметри камери, кожна тривимірна точки форми проектується на двовимірне зображення і таким чином збирає локальні ознаки для цієї частини зображення. Після цього, глибока неявна мережа декодує дану просторову точку у значення дистанційної знакової функції використовуючи як і локальні, так і глобальні ознаки зображення, зняті з згорткових шарів мережі. Архітектуру модуля вилучення локальних ознак зображено на рисунку 1.10.

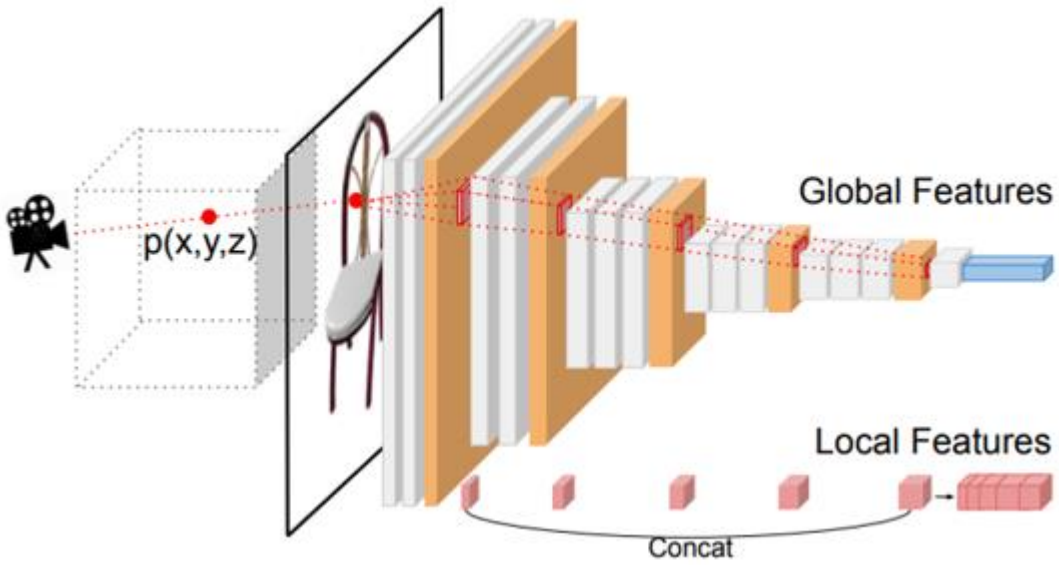

Рисунок 1.10 – Модуль вилучення локальних і глобальних ознак

Таким чином, при надходженні нового зображення, першою ціллю є проведення оцінки ексцентричних параметрів камери. Але пряма регресія на параметри камери за допомогою згорткових нейронних мереж часто працює досить погано. Для вирішення цієї проблеми, Інсафутдінов та Досовіцький представили ансамблевий підхід для оцінки параметрів камери за допомогою використання декількох шаблонних поз-кандидатів. Однак, такий підхід вимагає тренування занадто великої кількості параметрів. Однак, робота Жоу показує, що шестивимірне представлення повороту $b = (b_x, b_y)$, де $b \in \mathbb{R}^6$, $b_x \in \mathbb{R}^3$. Для даного b, матриця повороту $R = (R_x, R_y, R_z)^T \in \mathbb{R}^{3*3}$ обчислюється як

$$\begin{aligned} R_x &= N b_x, \\ R_z &= N(R_x \times b_y), \\ R_y &= R_z \times R_x, \end{aligned} \tag{1.9}$$

де $R_x, R_y, R_z \in \mathbb{R}^3$;

N – операція нормалізації.

Замість того, щоб вираховувати значення функції втрат напряму, оцінка пози камери використовується для трансформації даної просторової точки в систему координат камери. Далі необхідно вирахувати значення функції втрат L_{cam} як середню квадратичну похибку між трансформованою точкою простору та реальною точкою на моделі:

$$L_{cam} = \frac{\sum_{\mathrm{p}w \in PC_w} \|\mathrm{p}_G - (\mathrm{R}\mathrm{p}_w + t)\|_2^2}{\sum_{\mathrm{p}w \in PC_w} 1}, \tag{1.10}$$

де $PC_w \in \mathbb{R}^{N \times 3}$ – точка відносно реальної системи координат;

N – кількість точок в PC_w.

Таким чином, два декодери приймають на вхід глобальні та локальні ознаки, вилучені з вхідного зображення, та роблять оцінку значення дистанційної знакової функції.

Використовуючи лише глобальні ознаки, нейронна мережа успішно може змоделювати загальну форму об’єкта, але не здатна змоделювати більш складні деталі, як показано на рисунку 1.11.

Модуль локального вилучення ознак допомагає відновити тривимірну структуру об’єкта більш точно за рахунок оцінки залишкових значень дистанційної знакової функції.

Знакова дистанційна функція втрат використовується як регресійна, а не як класифікаційна метрика, за якої можна було б оцінювати лише відносне положення точки до поверхні об’єкту. Так стратегія дозволяє вилучати елементи поверхонь, які відносяться до різних ізо-значень.

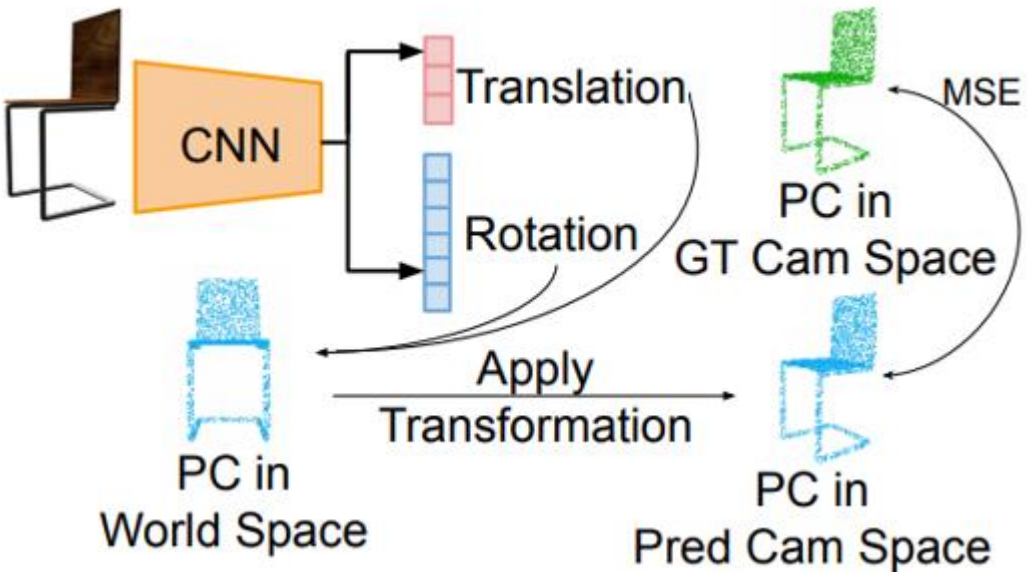

Рисунок 1.11 – Архітектура субмережі для оцінки параметрів камери

Для того, щоб нейронна мережа сконцентрувалася на відновленні малих деталей біля та всередині деякої поверхні S_0, запропоновано використовувати зважену функцію втрат, представлену як:

$$L_{SDF} = \sum_{\mathbf{p}} m|f(I,p) - SDF^I(\mathbf{p})|, \tag{1.11}$$

де m визначається як

$$m = \begin{cases} m_1, & \text{якщо } SDF^I(\mathbf{p}) < \delta \\ m_2, & \text{інакше} \end{cases}, \tag{1.12}$$

де δ – певне граничне значення;

m – параметри моделі.

1.4.2 Мережа нечіткої растеризації

Ще одним методом оцінки залежності між двовимірним зображенням та його тривимірною моделлю є мінімізація функцій втрат, що базується на двовимірних ключових точках або контурах вхідного зображення або апріорних знаннях про форму та природу об’єкта. Однак, такі підходи є або дуже чуттєвими до домену, з якого приходить зображення об’єкту, або можуть надати дуже слабкі результати через розрідженість двовимірних ознак. З іншого боку, зворотна задача реконструкції до тривимірного об’єкту з плоского зображення полягає у відношенні кожного пікселю вхідного зображення до вершини тривимірної моделі.

Проте, процес рендерингу зображення є недиференційованим за своєю природою. Наприклад, стандартні методи рендерингу мешу включає в себе дискретні вибіркові операції, що називаються растеризацією, що заперечує можливість обчислення градієнтів для відновлення вершин мешу. Оскільки функція рендерингу є нелінійною та й взагалі досить складною, в деяких методах були спроби апроксимувати градієнти за допомогою вручну підібраних близьких функцій, що дозволяло обійти проблему недиферінційованості процесу. І хоча такий підхід показав багатообіцяючі результати в задачі тривимірної реконструкції, непослідовність між прямим та зворотним проходами може призвести до неконтрольованості процесу оптимізації .

У даній роботі було запропоновано дійсно цілком диференційований варіант процесу рендеренгу, який був би здатен відновити меш на прямому проході мережі. Такий фреймворк бере до уваги різні тривимірні ознаки як геометрія об’єкту, нормалі до поверхі, кольори, умови освітлення тощо. Це дозволяє використовувати цю модель як частину інших нейромережевих систем, що вирішують задачу аналізу тривимірних даних без додаткових

налаштувань. Суть підходу полягає у новому погляді на процес рендеренгу як на нечіткий ймовірнісний процес. На відміну від стандартного процесу растеризації, пропонується надати усім полігонам-трикутникам ймовірнісний внесок при оцінці кожного рендерованого пікселю таким чином, щоб можна було змоделювати ймовірнісні карти.

У той час як традиційні методи рендеренгу склеюють фрагменти моделі дискретно, пропонується використовувати диференційовану агрегуючу функцію яка б поєднувала у собі і звичайні карти кольору та ознак, так і ймовірнісні полігональні карти. Такий новітній механізм дозволяє градієнтам проходити до усіх вершин полігонів, навіть перетинених. Такий підхід називається нечіткою растеризацією так як він зм'якшує дискретні растеризації з метою їх диференційованості. Завдяки послідовним процесам прямого та зворотного поширення помилки, Soft Rasterizer [15] може запроваджувати високоякісні проходи градієнтів, що можна б було використовувати у цілій низці задач, пов'язаних з обробкою тривимірної форми.

Що стосується відновлення тривимірної моделі об'єкту, такий підхід надає можливість генерувати якісні моделі з об'єктів, що перетинаються, завдяки використанню ймовірнісних мап та запроваджують гладку поверхню помилки, що дозволяє уникати локальних мінімумів, за рахунок гладкого рендеренгу. Порівняння нечіткої та дискретної растеризації наведено на рисунку 1.12. Проблема формується як моделювання дискретної бінарної маски в нечіткій повністю диференційованій манері. Для досягнення цього, пропонується використовувати два головних компоненти – карту ймовірностей $\{D_i\}$, яка моделює ймовірність знаходження кожного пікселю всередині даного полігону, та агрегаційну функцію A, яка суміщає мапи кольору, що базуються на $\{D_i\}$ та відносні глибинні карти між полігонами.

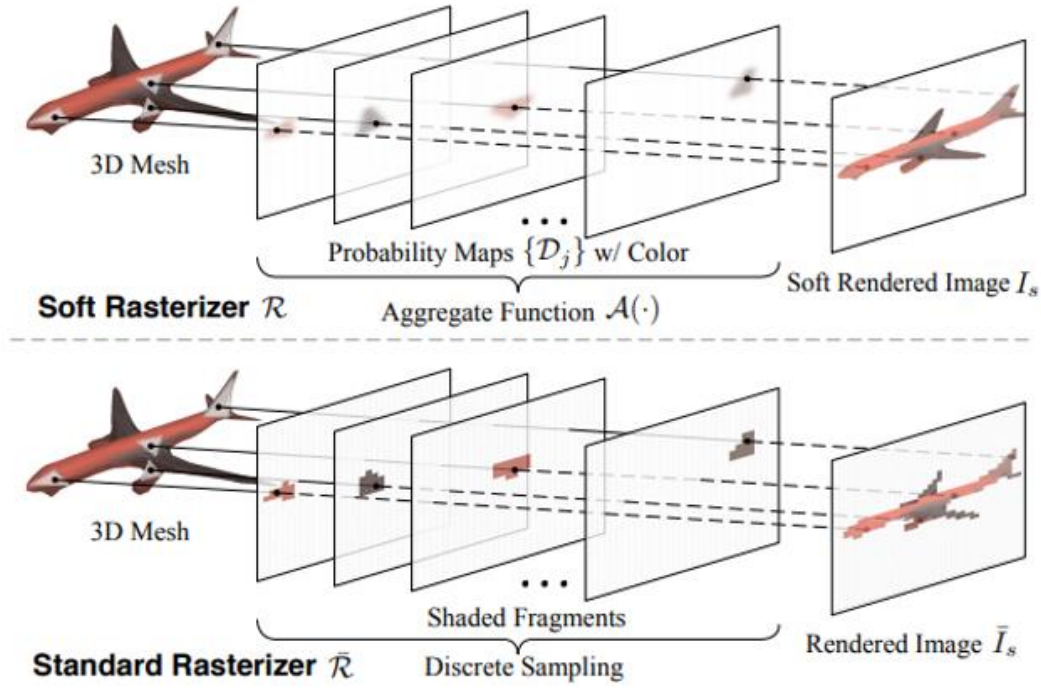

Рисунок 1.12 – Порівняння стандартної та нечіткої растеризації

Вплив кожного трикутника f_i на зображення через ймовірнісну карту D_i. Для оцінки ймовірності D_i в точці p_i, функція має враховувати як відносну позицію точки, так і відстань між D_i та p_i. Для цього, значення D_i в точці p_i обчислюється як:

$$D_j^i = sigmoid\left(\delta_j^i \frac{\mathrm{d}^2(\mathrm{i},j)}{\sigma}\right), \tag{1.13}$$

де σ – позитивний скаляр, що визначає чіткість розподілу ймовірностей;

δ_j^i – знаковий індикатор такий, що $a = \{+1,\ \text{якщо}\ p_i\ \in f_j;\ -1, \text{інакше}\}$.

Значення σ за замовчуванням встановлюється рівним 1×10^{-4}. $d(i,j)$ є найближчою відстанню між p_i та границями f_i. Відстань d за замовчуванням обрана як Евклідова відстань. Сигмоїдальна функція використовується як інтуїтивна безперервна апроксимація бінарної маски. До того ж, знаковий

індикатор зіставляє пікселі всередині та ззовні контуру f_i. Менше значення σ призводить до більш чіткого розподілу ймовірностей, в той час як більше значення призводить до розмивання границь цього розподілу. Це схематично зображено на рисунку 1.13.

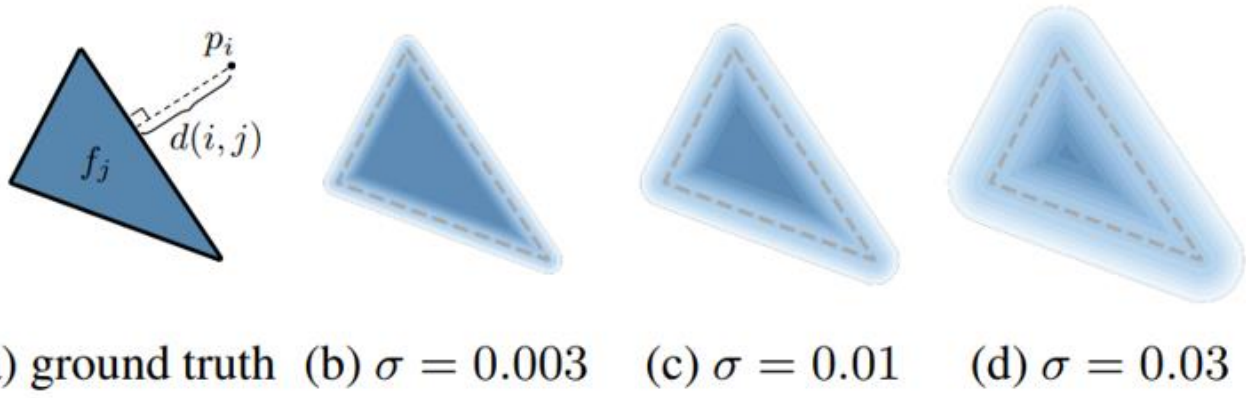

Рисунок 1.13 – Ймовірнісна карта трикутника в залежності від значення σ

З наближенням значення σ до 0, результуюча карта ймовірності наближується до істинної форми трикутника, що дозволяє наблизити розрахунок до традиційного процесу растеризації. Для кожного трикутника f_i визначається карта кольору C_i в пікселі p_i через інтерполяцію кольору вершини в барицентричних координатах. Барицентричні координати нормалізуються до діапазону [0, 1] для точки p_i, що лежить за межами f_i.

Далі пропонується використовувати агрегаційну функцію A для склеювання кольорових мап $\{C_j\}$ для отримання зрендерованого виходу I, що засновується на $\{D_j\}$, та відносних глибинних картах $\{z_j\}$. Агрегаційна функція A_S визначається як:

$$I^i = A_s(\{C_j\}) = \sum_j \omega_j^i C_j^i + \omega_b^i C_b, \tag{1.14}$$

де C_b – фоновий колір.

Ваги $\{\omega_j\}$ задовольняють $\sum_j \omega_j^i + \omega_b^i = 1$ і визначаються як:

$$\omega_j^i = \frac{D_j^i \exp(z_j^i/\gamma)}{\sum_k D_k^i \exp(z_k^i/\gamma) + \exp(\varepsilon/\gamma)}, \tag{1.15}$$

де z_j^i – нормалізована зворотна глибину тривимірної точки на f_i чиєю двовимірною проекцією є p_i.

Як можна помітити, ω_j є функцією двох змінних – D_j та z_j. Таким чином, ω_j присвоює більше значенням ближчим трикутникам, які мають більше значення z_j. Структура системи для реконструкції об'єкту з одного погляду показана на рисунку 1.14.

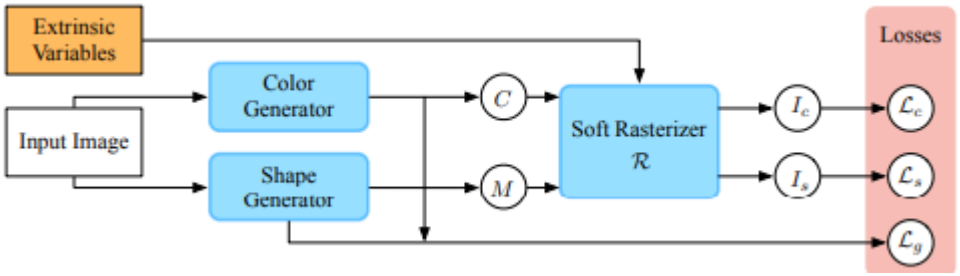

Рисунок 1.14 – Огляд архітектури Soft Rasterizer

Як можна побачити з рисунку, сама нейронна мереже оптимізується на трьох різних функціях втрат: $\mathcal{L}_s$ – функції втрат силуету, $\mathcal{L}_c$ – функції втрати кольору та функції втрати геометричної форми $\mathcal{L}_g$. Функція втрати силуету визначається як

$$\mathcal{L}_s = 1 - \frac{\|\hat{I}_S \otimes I_S\|_1}{\|\hat{I}_S \oplus I_S - \hat{I}_S \otimes I_S\|_1}, \tag{1.16}$$

де $\hat{I}_S$, I_S – прогнозована та реальна реконструкції.

Функція втрат кольору визначається як:

$$\mathcal{L}_c = \left\| \hat{I}_c - I_c \right\|_1. \quad (1.17)$$

Для досягнення високої візуальної якості, додається геометрична функція втрат. Фінальна функція втрат – зважена сумах усіх лоссів, зазначених вище:

$$\mathcal{L} = \mathcal{L}_s + \lambda \mathcal{L}_c + \mu \mathcal{L}_g. \quad (1.18)$$

Відстань d за замовчуванням обрана як Евклідова відстань. Сигмоїдальна функція використовується як інтуїтивна безперервна апроксимація бінарної маски. До того ж, знаковий індикатор зіставляє пікселі всередині та ззовні контуру f_i. Менше значення σ призводить до більш чіткого розподілу ймовірностей, в той час як більше значення призводить до розмивання границь цього розподілу. Відстань d за замовчуванням обрана як Евклідова відстань. Сигмоїдальна функція використовується як інтуїтивна безперервна апроксимація бінарної маски. До того ж, знаковий індикатор зіставляє пікселі всередині та ззовні контуру f_i. Менше значення σ призводить до більш чіткого розподілу ймовірностей, в той час як більше значення призводить до розмивання границь цього розподілу. Результат реконструкції методом нечіткої растеризації зображено на рисунку 1.15.

Таким чином, повністю диференційований процес рендеренгу, запропонований у цьому методі надає можливість створювати якісні тривимірні реконструкцій об’єктів.

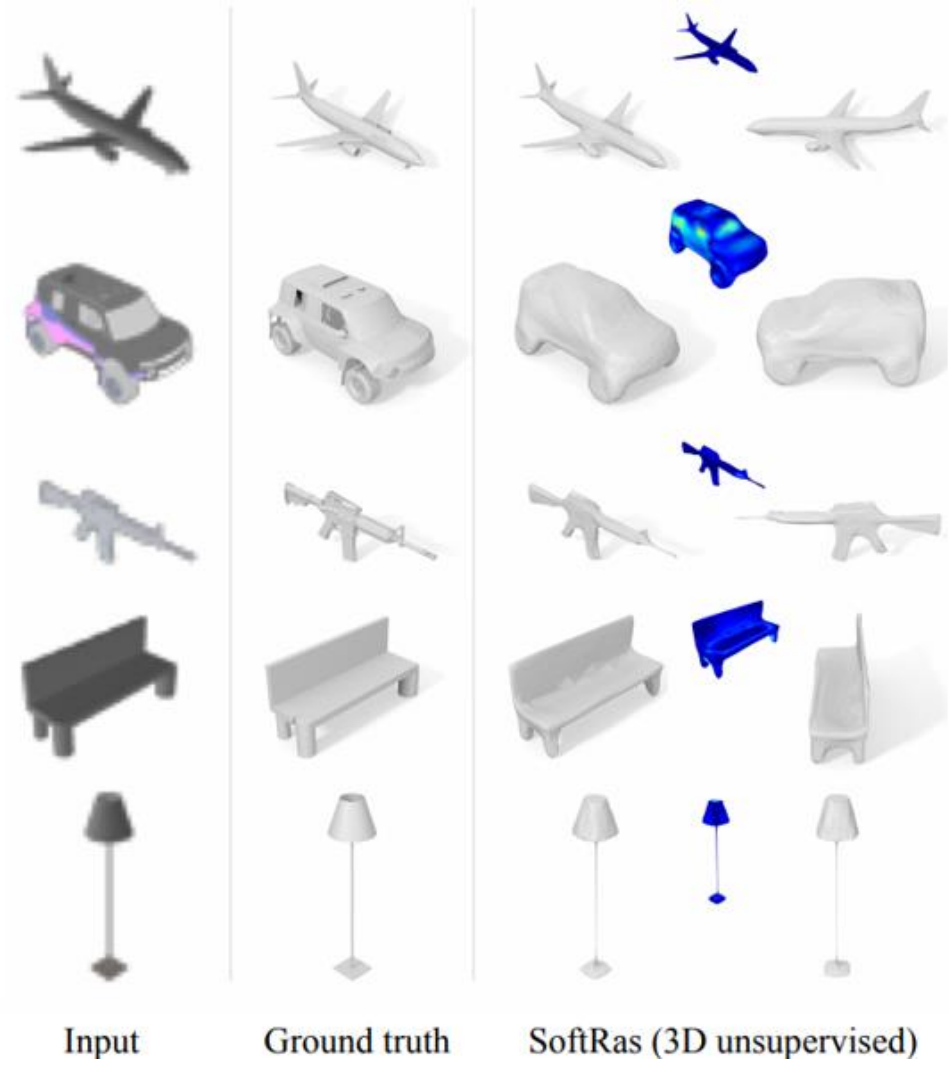

Рисунок 1.15 – Результат реконструкції Soft Rasterizer

1.4.3 Навчання реконструкції через симетрію відбиття

На відміну від методів, які використовують декілька зображень для тривимірної відбудови, методи реконструкції, зазначені вище, не можуть повністю експлуатувати геометричні обмеження між вхідним двовимірним зображення та результуючою тривимірною моделлю. Отже, формулювання задачі поставлено неправильно і веде до неточності відновлення фігури та обмеження можливості узагальнення на інші задачі. Для вирішення цієї

проблеми було запропоновано використовувати структурну властивість більшості об’єктів, створених людиною, а саме симетрію їх відображення, що впроваджує геометричній зв’язок між зображенням та його глибиною і дає можливість повністю реконструювати тривимірну модель зображення. Для досягнення цієї мети була запропонована штучна нейронна мережа SymmetryNet [17], що комбінує сильні сторони розпізнавання, заснованого на навчанні та геометричних методів реконструкції.

Спочатку SymmetryNet оцінює параметри уявної дзеркальної площини об’єкту, а потім відновлює глибину зображення об’єкту через стереопсис його відображення. Мережа складається з остової мережі для вилучення ознак та повністю диференційованого модуля для оцінки втрат реконструкції. Такий фреймворк надає можливість природним чином використовувати інформацію з відображення пікселів на єдиному зображенні. Натхненні успіхом згорткових нейронних мереж в задачах класифікації та детектування, була застосована велика кількість способів тривимірного представлення таких як вокселі, карти глибини, хмари точок та знакові дистанційні функції. І хоча ці способи показують багатообіцяючі результати на деяких наборах даних, реконструкція об’єкту з одного зображення сама по собі є нерозв'язною задачею.

Не маючи апріорної інформації про геометричні обмеження об’єкту, відновлені форми за визначенням не можуть бути достатньо точними навіть на тренувальних об’єктах, не кажучи вже про об’єкти, які мережа раніше не бачила під час навчання. Для того, щоб обійти цю властивість задачі, і використовується властивість симетрії об’єктів. Для більшого розуміння, на рисунку 1.16 відображено приклади симетричності в природних та рукотворних об’єктах.

Рисунок 1.16 – Двомірна та тривимірна симетрія об'єктів

Традиційні методи реконструкції, що використовують декілька зображень об’єктів, формують функцію втрат через фотометричні ознаки зображень та після цього намагаються відновити карту глибини зображення. Деякі методи напряму вивчають залежність між ділянками зображення з різних зображень. На відміну від цих методів, SymmetryNet використовує окремий модуль зіставлення ознак, що базується на симетричних властивостях об’єктів, що робить його потужним методом, який можна використовувати для реконструкції з єдиного зображення. Для того, щоб оцінити параметри площини симетрії, виконується оцінка параметрів камери і в той же час вирішується задача оцінки глибина єдиного вхідного зображення. Щоб досягти цієї мети була SymmetryNet має в собі пайплайн, що комбінує сильні сторони розпізнавання, заснованого на навчанні та геометричних методів реконструкції.

Нехай набір точок в однорідній системі координат позначається як $\mathbb{O} \subset \mathbb{R}^4$. Якщо $\mathbb{O}$ відповідає симетрії об’єкту w.r.t. трансформації $M \in \mathbb{R}^{4\times4}$, тоді справедливо буде наступне твердження:

$$\forall \in \mathbb{O}: MX \in \mathbb{O}, \tag{1.19}$$

де MX – відповідна точка з X відносно симетрії.

Наприклад, якщо об'єкт має симетричне відображення відносно площини YZ, то його трансформація записується як $M_x = diag(-1,1,1,0)$. Маючи дві тривимірні точки $X, X' \in \mathbb{O}$ в однорідних координатах, що відповідають симетричній трансформації $X' = MX$, їх двомірні проекції $\mathbf{x}, \mathbf{x}'$ мають задовольняти наступним умовам:

$$\mathrm{x} = KR_tX/d, \tag{1.20}$$

де $\mathrm{x} = [x, y, 1, 1\ /\ d]^T$ двомірні координати точки;

d – глибина в системі камери;

$K \in \mathbb{R}^{4\times4}$ – внутрішні параметри камери;

$R_t = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$ – матриця ексцентричних параметрів камери.

Тоді можна вивести наступні обмеження для двомірних проекцій x, x':

$$\mathrm{x}' \propto KR_tMR_t{}^{-1}K^{-1}\mathrm{x} = C\mathrm{x}\,. \tag{1.21}$$

Дійсно, ці обмеження є дуже цінними для тренування нейронної мережі, так як тепер є геометрично коректний спосіб оцінити наскільки правильною є прогнозована в точці (x, y) глибина d, порівнявши дільниці зображення в точках (x, y) та (x', y'). Якщо оцінка d була зроблена правильно, порівнювані дільниці зображень мають бути однаковими так як $\mathbb{F}(X) = \mathbb{F}(X')$.

Ця властивість називається фотоконсистенцією. Рівняння з формули 1.12 надає генералізований спосіб репрезентації будь-якого виду симетрії за допомогою отриманої матриці C.

Для симетрії відображення, більш інтуїтивною параметризацією є використання рівняння площини симетрії в системі координат камери. Нехай $\tilde{x} \in \mathbb{R}^3$ – координата точки на площині симетрії в просторі камери. Рівняння площини симетрії може бути записано як:

$$w^T \tilde{x} + 1 = 0, \tag{1.22}$$

де $w \in \mathbb{R}^3$ – параметризація площини симетрії.

Тоді відношення між C та w можна записати як:

$$C = K\left(I - \frac{2}{\|w\|_2^2}\begin{bmatrix} w \\ 0 \end{bmatrix}[w^T \ 1]\right)K^{-1}. \tag{1.23}$$

Ціль детектування симетрії відображення полягає в відновленні параметрів w з зображень. У випадку симетрії відображення, не можна визначити $\|w\|_2$ не знаючи апріорних значень розміру об'єкту. Через це $\|w\|_2$ не відновлюється, а приймає деяке константне значення. Для використання цієї системи у реальних ситуаціях, розміри об'єктів можна вивести, знаючи реальний розмір об'єкту або знаючи відстань від камери до об'єкту. SymmetryNet виконує два завдання одночасно з наскрізною системою глибокого навчання: виявлення симетрії та оцінка глибини. Він приймає одне зображення як вхідні та вихідні параметри площини симетрії та карт глибини. Зображення архітектури усього пайплайну представлено на рисунку 1.17.

Функцією модуля вилучення ознак є створення початкового тривимірного тензору для функції втрат об'єму $V(x, y, d)$. Потім, d рівномірно дискретизується так, щоб зробити начальний об'єм однорідним до операції тривимірної згортки.

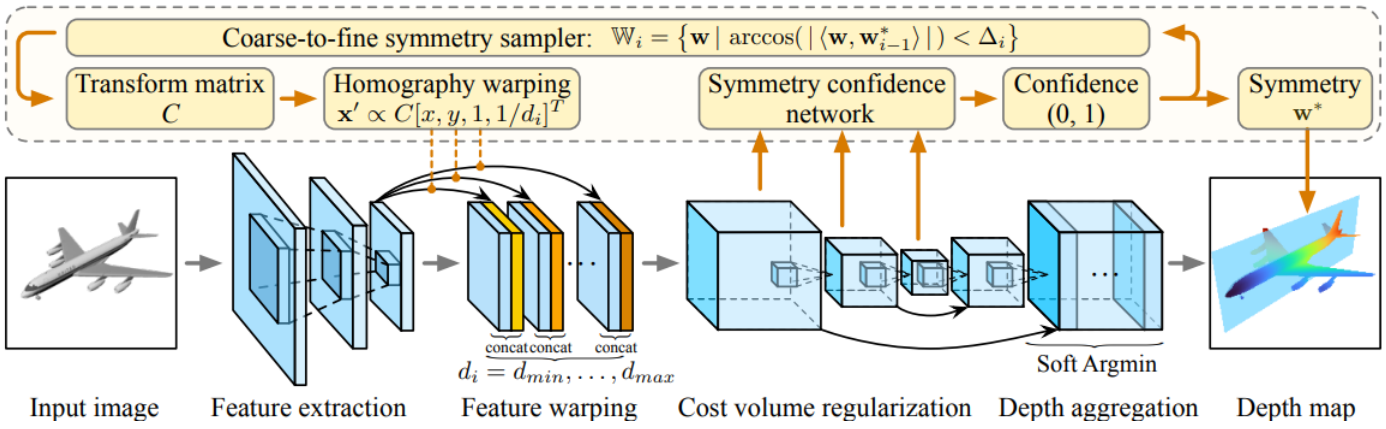

Рисунок 1.17 – Архітектура SymmetryNet

Таким чином, тензор V визначається як конкатенація ознак, вилучених з оригінального зображення та тривимірної моделі:

$$V(x, y, d) = [F(x, y), F(x', y')]. \quad (1.24)$$

Оцінка глибини вираховується з вектору ймовірностей P як карта глибини $\widehat{\mathrm{D}}$ і вираховується за формулою:

$$\frac{1}{|D|} \sum_{d \in D} dP(x, y, d). \quad (1.25)$$

Загальна функція втрат мережі вираховується як сума $L = L_{dpt} + L_{cls}$, де L_{dpt} та L_{cls} визначаються як:

$$L_{dpt} = \frac{1}{n} \sum_{x,y} \left| \widehat{D}(x, y) - \frac{\|\hat{w}\|_2}{\|w\|_2} D(x, y) \right|, \quad (1.26)$$

Результати реконструкцій, отриманих за допомогою SymmetryNet зображено на рисунку 1.18.

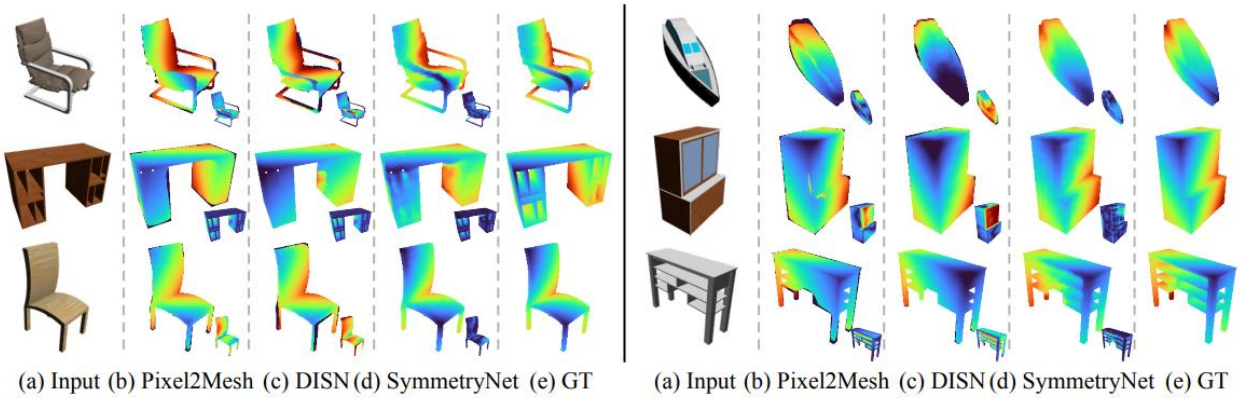

Рисунок 1.18 – Порівняння реконструкцій SymmetryNet та інших методів

1.5 Постановка задачі

Таким чином, проаналізувавши існуючі методи тривимірної реконструкції, оцінив їх сильні стороні, такі як здатність працювати в умовах перетину об’єкту реконструкції, стійкість до змін оточення, освітлення, можливість використання малої кількості навчальних прикладів та недоліки, такі як неточність побудови геометрії вихідного зображення, низький рівень деталізації тощо., завдання дослідження зводиться до наступного:

– обрати та обґрунтувати архітектуру нейронної мережі для створення тривимірної реконструкції;

– спроектувати алгоритм навчання отриманої нейронної мережі;

– провести імітаційне моделювання та порівняльний аналіз розробленої системи глибинного навчання з існуючими на даний момент методами.

2 ПРОЕКТУВАННЯ СИСТЕМИ

2.1 Архітектура запропонованої моделі

Проаналізувавши існуючі методи тривимірної реконструкції з одного зображення та оцінивши застосовані підходи, їх сильні та слабкі сторони, пропонується розробити новітню архітектуру нейронної мережі для вирішення задачі тривимірної відбудови. У цьому розділі презентується архітектура, відмінна від розглянутих вище, яка в цілому складається з блоку енкодера, що формуватиме стиснене представлення вхідного зображення, декодеру, який виконуватиме задачу початкового відновлення тривимірної структури об’єкту зображення, та ще одного додаткового блоку, метою якого поліпшення візуальної структури тривимірної моделі, отриманої після прямого проходу блоку декодування. Необхідність створення нової архітектури для вирішення даної задачі випливає з деяких серйозних недоліків методів, оглянутих раніше, таких як проблема реконструкції об’єктів складної структури, необхідність наявності великої кількості тренувальних даних, необхідність наявності зображень об’єктів з різних ракурсів, довгий час відтворення однієї реконструкції та нестійкість до змін умов розташування та освітлення об’єкта. Запропонована модель приймає на вхід єдине зображення об’єкта та реконструює його як тривимірну воксельну сітку. Спочатку, енкодер створює набір мап ознак вхідного зображення, після цього декодер приймає на вхід створені мапи та використовує їх для створення першого наближення тривимірної реконструкції об’єкту. В решті решт, спеціально створений блок нейронної мережі покращує грубе наближення, отримане з декодера, та таким чином відтворює фінальну версію реконструкції об’єкта. Огляд архітектури запропонованої моделі представлено на рисунку 2.1.

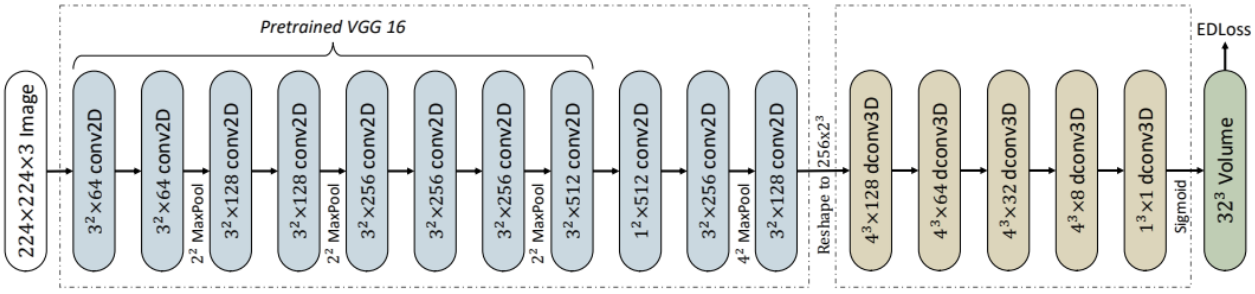

Рисунок 2.1 – Архітектура запропонованої моделі

2.1.1 Автокодувальники

Автокодувальник – це нейронна мережа, яка вчиться копіювати свій вхідний сигнал на свій вихідний сигнал. Він має внутрішній (прихований) шар, який описується прихованим вектором стану нейронної мережі, що використовується для представлення вхідних даних, і складається з двох основних частин: енкодера, який створює відображення вхідного сигналу у вигляді прихованого вектору, і декодера, який використовує це приховане відображення сигналу для реконструкції оригінального вхідного сигналу.

Виконання завдання відновлення відображення можна було б досягти лише просто продублювавши вхідний сигнал, і саме тому автоенкодери зазвичай обмежуються таким чином, що змушують їх лише приблизно реконструювати вхідні дані, зберігаючи лише найрелевантніші аспекти даних у вихідній реконструкції [18].

Ідея автокодерів популярна в галузі нейронних мереж протягом десятиліть, і перші інтелектуальні системи, що використовують ці властивості автокодувальників, датуються 80-ми роками минулого сторіччя. Найбільш традиційним їх застосуванням було зменшення розмірності вхідних даних (dimensionality reduction) або вивчення нових признаків, але нещодавно концепція автоенкодера стала більш широко застосовуватися для

тренування генеративних моделей штучних нейронних мереж. Деякі з найпотужніших штучних нейронних мереж в 2010-х роках використовували розріджені автоенкодери, розміщені всередині глибоких нейронних мереж.

Основний принцип роботи і навчання мережі автокодувальника – отримати на вихідному шарі відгук, найбільш близький до вхідного. Для того щоб рішення не виявилося встановленням відношення ідентичності, на проміжний шар автокодувальника накладають обмеження: проміжний шар повинен бути або меншої розмірності за вхідний та вихідний шари, або штучно обмежується кількість одночасно активних нейронів проміжного шару – це називається розрідженою активацією. Такі обмеження змушують нашу модель шукати узагальнення і залежність в даних що надходять на вхідний шар мережі та виконувати їх стиснення. Таким чином, штучна нейронна мережа автоматично навчається виділяти з вхідних даних загальні ознаки, які кодуються в значеннях ваг штучної нейронної мережі. Так, при навчанні мережі на наборі різних вхідних зображень, нейронна мережа може самостійно навчитися високорівневі ознаки, такі як дуги і лінії. Структура автокодувальника у широкому сенсі показана на рисунку 2.2.

Найчастіше автокодувальники застосовують як каскадний елемент для навчання глибоких мереж. Автокодувальники також можна застосовувати для попереднього, абстрактного навчання глибокої мережі при навчанні без вчителя. Для цього шари тренують поступово і по черзі. До кожного нового шару на час навчання підключається ще один вихідний шар, що архітектурно наближує мережу до автоенкодеру, після чого на вхід мережі подається один пакет даних. Параметри усіх шарів такої мережі оптимізуються за допомогою методу зворотного поширення помилки. Після цього один шар автоенкодеру вимикається і додається новий такий, що відповідає наступному ще не натренованому шару мережі. На вхід мережі ще раз надходить той же набір даних, натреновані перші шари мережі не змінюються і працюють в якості

константної трансформації вхідних чергового шару автоенкодера, що навчається. Так навчання відбувається для всіх шарів мережі окрім останніх. Останні шари можна тренувати також за допомогою методу зворотного поширення помилки в режимі навчання з учителем.

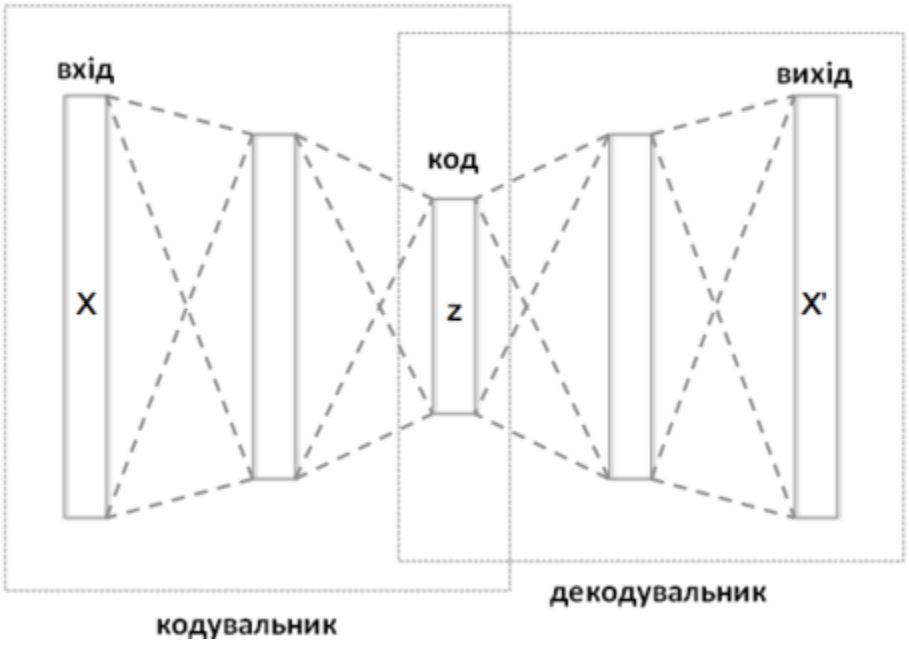

Рисунок 2.2 – Схематична структура автоенкодеру

За архітектурної точки зору найбільш простою варіацією автокодувальника є повнозв'язна нейронна мережа прямого поширення сигналу, що, насправді, є подібною до багатошарового перцептрону із шаром входу, шаром виходу та одним або декількома прихованими шарами, які їх поєднують. Різницею між автоенкодером та багатошаровим персептроном є те, що в автоенкодеру вихідний шар має ту ж саму кількість вузлів, як і шар входу, а ще й те, що замість тренування прогнозуванню цільового значення змінної *Y* для заданих входів *X*, автокодувальники тренують реконструкції їхніх власних входів *X*. Автоенкодер за визначенням завжди складається з

двох головних блоків, а саме енкодеру та декодеру, які можна визначити як функціонал $\varphi: X \to F$ і та $\psi: F \to X$, так, що:

$$\arg\min \|X - (\varphi \circ \psi)X^2\|. \quad (2.1)$$

У випадку, коли кількість прихованих шарів дорівнює одному, автоенкодер бере деякий вхід $x \in \mathbb{R}^d$ і виконує його відображення на $z \in \mathbb{R}^p$:

$$z = \sigma_1(Wx + b). \quad (2.2)$$

В цій формулі σ є активаційною функцією, як, наприклад, сигмоїдальна функція або ReLU. Після цього z відображується на реконструкцію x' тієї ж розмірності, що й x:

$$x' = \sigma_2(W_z' + b'). \quad (2.3)$$

Автоенкодери також навчають мінімізувати помилки реконструкції, наприклад, середньоквадратичну похибку:

$$L(x, x') = \|x - x'\|^2. \quad (2.4)$$

Якщо деякий простір ознак F має меншу розмірність, ніж вхідний простір X, то вектор ознак $\varphi(x)$ може бути розглянуто як стисле представлення входу x. А якщо ж приховані шари мають більшу розмірність за вхідний шар мережі, автоенкодер, в теорії, може навчитися ідентичної функції, і стати марним. Проте, на практиці виявилося, що автокодувальники все одно можуть навчитися вилучати корисні ознаки і за таких обставин.

В нашій задачі енкодер необхідний для створення стисненого відображення вхідного зображення об’єкту, яке пізніше буду декодовано декодером. На цьому етапі входом мережі є зображення формату RGB, яке представлено як тензор розміром $224 \times 224 \times 3$. З нього створюється відображення у вигляді тензору розміром $512 \times 28 \times 28$. Ця трансформація виникає під час проходження вхідного тензору через перші 9 згорткових шарів мережі разом із відповідною нормалізацією пакету та функціями активації ReLU, що насправді є мережею VGG-16, натренованою на наборі даних ImageNet. Після цієї процедури вилучення ознак йде ще три згорткових шари із нормалізаціями, експоненційними функціями активації для створення фінального подання зображення, яке пізніше буду подано на декодуючий блок нейронної мережі. Розміри фільтрів останніх трьох шарів рівні 3^2, 3^2 та 1^2 відповідно. Розмір вихідного тензору дорівнює $512 \times 512 \times 256$, і пізніше трансформується до тензору форми 256×2^3. Після другого шару додаткової згортки є шар максимальної підвибірки розміру 3^2.

Декодер відповідає за трансформацію двомірних мап ознак у тривимірний об’ємний тензор [24]. В декодері присутні п’ять тривимірних транспонованих операцій згортки. А саме, перші 4 згортки мають розмір ядра рівний 4^3 із кроком 2 та розширенням рівним 1. За кожним шаром згортки йде шар пакетної нормалізації та активаційна функція ReLU, окрім останнього шару, який має сигмоїдальну функцію активації. Транспоновані згорткові шари мають кількість каналів рівну 128, 64, 32, 8 та 1 відповідно. Виходом декодера є вокселізована форма розміром 32^3. Ця трансформація виникає під час проходження вхідного тензору через перші 9 згорткових шарів мережі разом із відповідною нормалізацією пакету та функціями активації ReLU, що насправді є мережею VGG-16, натренованою на наборі даних ImageNet. Після цієї процедури вилучення ознак йде ще три згорткових шари із нормалізаціями, експоненційними функціями активації для створення

фінального подання зображення, яке пізніше буду подано на декодуючий блок нейронної мережі.

2.1.2 Модуль контекстно-незалежного комбінування

З різних точок зору ми можемо бачити різні видимі частини об'єкта. Відновні якості видимих частин є набагато вище, ніж у невидимих частин. Тому пропонується розробити контекстно-орієнтований модуль злиття, що адаптивно підбиратиме якісну реконструкцію для кожної частини (наприклад, ніжки столу) з різних грубих реконструкцій.

Обрані приблизні реконструкції використовуються для того, щоб створити єдину високоякісну реконструкцію об’єкта. З огляду на грубі варіанти реконструкцій і відповідного їм контексту, контекстно-відомий модуль синтезу генерує оціночну карту для кожного грубого об'єму, а потім зливає їх в один об’єм шляхом зваженого підсумовування всіх грубих об’ємів відповідно до їх бальних карт [19]. Просторова інформація вокселів зберігається у контексті модуль синтезу, і, отже, можна використовувати інформацію з декількох переглядів, щоб краще відновити структуру об'єкта. Зокрема, модуль контенкстно-незалежної генерації створює деякий контекст c_r r-го грубого наближення через конкатенацію виходів останніх двох шарів декодеру. Мережа оцінки контексту складається з п'яти наборів тривимірних згорткових шарів, кожен з яких має розмір ядра 3^3 і заповнення 1, після чого йде пакетна нормалізація та активація ReLU. Кількість вихідних каналів згорткових шарів становить 9, 16, 8, 4 та 1 відповідно. Отриманий бал m_r для контексту c_rнормується за всіма отриманими оцінками. Як функцію нормалізації було обрано softmax. Таким чином, оцінка $s_r^{(i,j,k)}$ для позиції (i,j,k) r-го вокселю може бути обчислена як

$$s_r^{(i,j,k)} = \frac{exp\left(m_r^{(i,j,k)}\right)}{\sum_{p=1}^{n} exp\left(m_p^{(i,j,k)}\right)}, \quad (2.5)$$

де n – кількість зображень.

Нарешті, фінальний воксель v^f визначається як сума добутків грубих об'ємів та їх оцінок:

$$v^f = \sum_{r=1}^{n} s_r v_r^c. \quad (2.6)$$

2.1.3 Модуль структурного корегування

Блок структурного корегування можна розглядати як залишкову мережу, що має на меті виправити неправильно відновлені частини тривимірної моделі. Ідея полягає в об'єднанні тривимірного автоенкодера із зв'язками U-Net. За допомогою цих зв'язків можна зберегти структуру об'єму, отриману з блоку контекстного поєднання. Зокрема, енкодер має три шари тривимірної згортки, кожен з яких має 4^3 фільтрів з паддінгом 2, пакетну нормалізацію та функцію активації ReLU та шаром макспулінгу з розміром ядра рівним 2^3. Після енкодеру йде два повнозв'язних шари розмірами 2048 та 8192 відповідно.

Декодер складається з трьох шарів транспонованої згортки, в кожному є 4^3 фільтрів з паддінгом 2 та шагом в 1. Окрім останніх шарів транспонованої згортки, після яких йде сигмоїдальна активація, інші шари мають шари пакетної нормалізації та функцію активації ReLU. Структура блоку корегування зображено на рисунку 2.3.

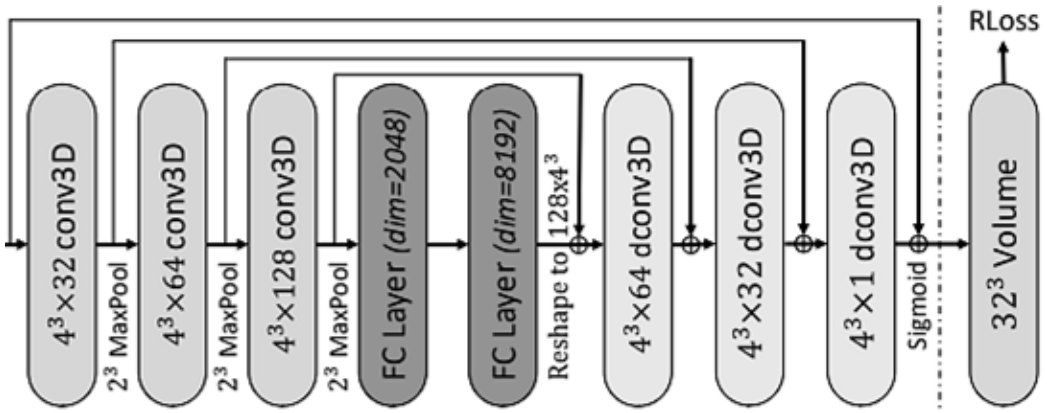

Рисунок 2.3 – Блок структурного корегування

2.2 Показники навчання нейронної мережі

Епохою називається прямий і зворотний прохід по всім тренувальним прикладам. Довжина серії – кількість тренувальних семплів для однієї епохи прямого і зворотного циклів мережі. Кількість епох відповідає кількості здійснених проходів: кожен прохід використовує одні й ті ж навчальні приклади. Одна епоха дорівнює виконанню прямого проходу та зворотного проходу поширення помилки.

Так, якщо є усього, наприклад, 100 прикладів, batch = 50, нам буде потрібно зробити дві ітерації, щоб завершити одну епоху. Кажучи більш формально, тренування нейронної мережі – багатопараметрична оптимізаційна задача для деякої опуклої функції. Процес тренування мережі можна описати за допомогою кривої, на якій ординатна вісь відображає кількість епох або ітерацій, яку пройшла мережа, а вісь абсцис відображає значення помилки, що отримано при класифікації вхідних даних.

На рисунку 2.4 зображено два графіки: графік значення функції втрат на тренувальному наборі та графік значення середньої функції втрат на валідаційному наборі.

Коли більшу кількість даних буде використано для тренування, тоді менше похибок матиме архітектура мережі, що відповідатимуть навчальним даним. У деякий момент часу значення лосс-функції тренувального та валідаційного наборів повинно бути хоча б однаковою за магнитудою. Побудувавши криву навчального набору, можна оцінити, чи здатна архітектурна модель мережі відповідати даним для потрібної класифікації помилки. Значення функції втрат не повинно бути значно нижчим, ніж значення помилки навчального набору.

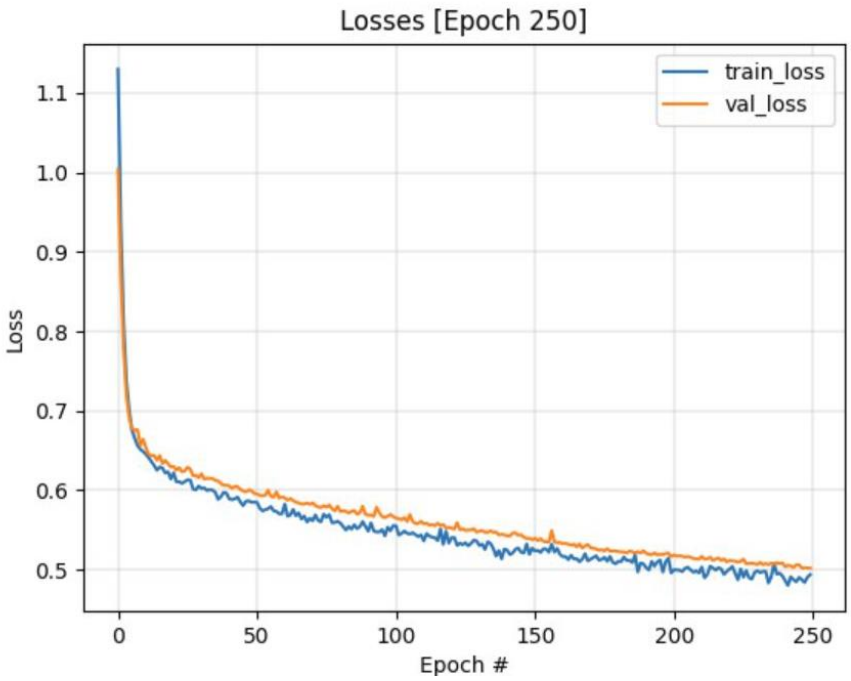

Рисунок 2.4 – Криві помилок на навчальних та тестових даних

Якщо помилка на навчальному наборі занадто висока, то збільшення даних вже не допоможе адже це означатиме, що модель занадто слабка для того щоб змоделювати навіть дані, які вона бачила. Тобто, таким чином стане ясно, що сама архітектура моделі, оптимізатор або передобробка даних потребують корегування. У випадку, коли на графіку значення похибки на

тренувальному наборі значно перевищує значення похибки на валідаційному наборі, можна зменшити роздільну здатність вхідних даних, або додати ще навчальних зразків даних, що, треба сказати, не завжди можливо[20].

Ці контрольні значення допомагають обчисленню та відображенню гіперпараметрів. Їх можна подати у вигляді графіків гіперпараметрів на горизонтальній осі та значення якості на вертикальній осі для подальшої оптимізації гіперпараметрів моделі. Регресійна метрика як втрата є типовими показниками якості, або для нашої задачі це класифікаційна функція втрат крос-ентропія та перетин над об'єднанням. У ситуації, за якої кількість епох для тренування розглядається як головний гіперпараметр, показники діють як індикатор занадто довгого часу навчання та зниження його ефективності. Побудувавши графік значень помилки на навчальному наборі, а також значень функції втрат на валідаційному, можна також дізнатися, чи є загроза перенавчанню мережі. Приклад зображено на рисунку 2.5.

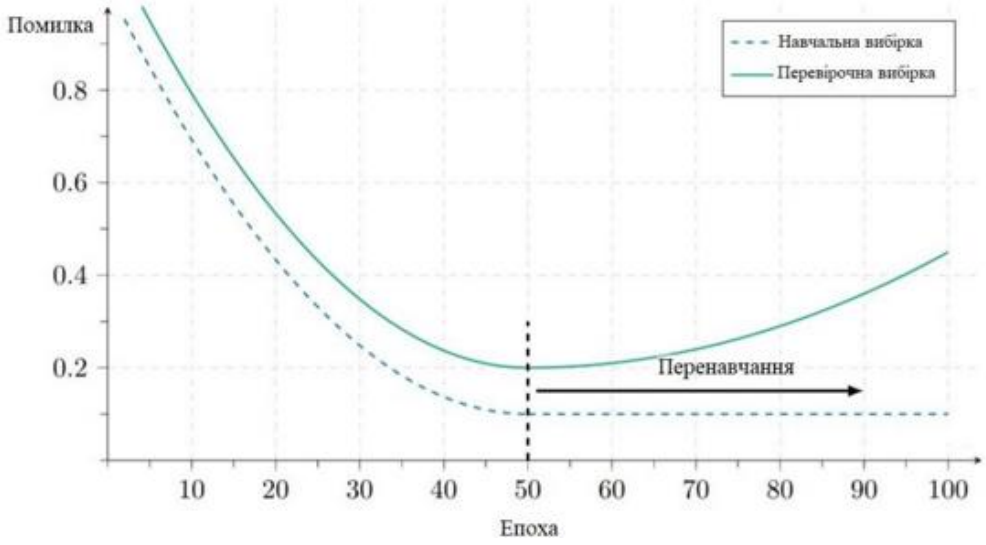

Рисунок 2.5 – Проблема перенавчання мережі

Розглянемо рисунок 2.5 детальніше. На ньому зображено класичну ситуацію перенавчання або оверфітингу нейронної мережі. У певний момент нейронна мережа якісно реконструюватиме лише дані з навчального набору і втратить здатність до генералізації. В теорії, чим довше буде продовжуватися ефективне тренування мережі, тим більше вона запам’ятає тренувальний набор даних. Поки естіматор продовжує покращувати показник якості на тренувальному наборі, він може, навпроти, погіршувати точність реконструкції при застосуванні тестового набору. За випадку, коли значення якості покращується при достатній кількості епох можна спробувати змінити значення початкових ваг, чи застосувати нормалізацію-стандартизацію моделі та вхідних даних, додавши регулярізацію. Функція втрати запропонованої мережі визначається як усереднена бінарна-ентропія між вокселем реконструйованої моделі та дійсної моделі та формально може бути визначена наступним чином:

$$\ell = \frac{1}{N}\sum_{i=1}^{N}[gt_i log(p_i) + (1 - gt_i) log(1 - p_i)], \qquad (2.7)$$

де N – кількість вокселів в реальній моделі.

2.2.1 Оцінка якості реконструкції

Для оцінки якості проведеної реконструкції запропонованим методом була виконана бінарізація ймовірностей для переходу від раціональних до бінарних значень. Границею бінарізації, шляхом перебору, було обрано значення 0.3. Як метод оцінки використовувалася міра відношення перетину та об'єднання або IoU. Формально формулу для оцінки якості можна записати як

$$IoU = \frac{\sum_{i,j,k} I(p_{(i,j,k)} > t) I(gt_{(i,j,k)})}{\sum_{i,j,k} I[I(p_{(i,j,k)} > t) + I(gt_{(i,j,k)})]}, \quad (2.8)$$

де $p_{(i,j,k)}$ – ймовірність вокселю в точці (i, j, k);

$gt_{(i,j,k)}$ – фактичне значення в точці (i, j, k);

I – індикаторна функція;

t – граничне значення бінарізації.

Вище значення IoU відповідає більшій якості реконструкції.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Огляд бібліотек глибинного навчання

На цей час у імплементації нейронних мереж важливу, як правило, роль мають спеціалізовані фреймворк та пакети, які містять у собі вже повністю реалізовані алгоритми та базові структури для полегшення процесу розробки. Найпопулярнішими з них є такі бібліотеки, як Theano, PyTorch та Torch та ін. Ці програмні пакети реалізують прикладний програмний інтерфейс для виконання низькорівневих затратних розрахунків, тому процес вибору певного пакету, в першу чергу, має засновуватися на суперечностях про продуктивність та зручність розробки.

Треба помітити, що, наприклад, Torch містить інтерфейс для високорівневої мови програмування Lua, що не є дуже популярною серед програмістів, а це істотно погіршує зручність і темп розробки. Відповідно до досліджень, проведених з метою ранжування фреймворків, і Theano, і Torch показують приблизно однакову продуктивність для одного и того ж датасету.

Крім незручності застосування пакету Torch, згаданої вище, мається ще одна, що також пов'язана зі зручністю розробки. Попередні два інструменти, згаданих вище, здатні працювати з більш високорівневою бібліотекою Keras [25], яка може надавати загальні, для PyTorch і Theano, вже передозначені алгоритми та архітектури. У ході порівняння документації бібліотек та якості навчальних додатків було прийнято рішення використовувати для програмної реалізації нейронної мережі бібліотеку PyTorch.

PyTorch – програмний продукт, а саме бібліотека, написана на мові Python, яка реалізує високорівневе API для роботи зі штучними нейронними

мережами. Цей пакет створювався заради зменшення труднощів при побудові нейронних мереж, які пов'язані суто з синтаксичними рисами деяких пакетів.

Theano [26] і його високорівневі надбудови утворюють свого роду сімейство бібліотек, які мають аналогічні характеристики. Оскільки Theano насправді є не зовсім бібліотекою машинного навчання, а швидше за бібліотекою оптимізації низького рівня для обчислювальних графів, існує безліч інших бібліотек, в основі яких лежить Theano.

Theano концентрується на ідеї обчислювальних графів: вона знає, як взяти обчислювальну структуру графа і перетворити його в дуже ефективний код, який використовує бібліотеки SciPy-NumPy [27], нативні бібліотеки, такі як BLAS і чистий C++ код, щоб працювати якомога швидше на центральних або графічних процесорах.

Theano пропонує оптимізацію швидкості і стабільності, оскільки вона внутрішньо реорганізовує та оптимізує обчислення. Однією з її «фішок» є автоматична диференціація: потрібно тільки реалізувати пряму частину моделі, і Theano автоматично визначить, як обчислити градієнти в різних точках, дозволяючи користувачам виконувати градієнтний спуск для навчання моделі.

Цей фреймворк, насправді, не є бібліотекою машинного навчання, оскільки він не надає користувачеві попередньо вишикувані моделі, які можуть навчатися на наборах даних. Вона є математичної бібліотекою, яка надає інструменти для створення призначених для користувача моделей машинного навчання. Використання Theano спрощує реалізацію методу зворотного поширення помилки для згортальних мереж і рекурентних мереж в цілому, оскільки вона моделює НН у вигляді обчислювального графа. З цієї причини вона надає функції «if else» або «switch», що дозволяють використовувати умовний потік управління на графі. Інші обгортки Theano включають Pylearn2, Bloaks і Keras. Алгоритми Pylearn2 можуть бути

записані з використанням виразів Theano, а Theano оптимізує і стабілізує вираження. Вона включає в себе все необхідне для мереж багатошарових персептронів, обмежених машин Больцмана і згорткових нейронних мереж.

Caffe – фреймворк глибокого навчання, розроблений Berkeley Vision and Learning Center (BVLC) в основному для завдань розпізнавання об'єктів. Caffe бере початок від реалізації нейронної мережі AlexNet і написаний в основному на C ++; він спочатку призначався для CHH та забезпечував легку реалізацію нейронних мереж прямого поширення. Він загальновизнано хороший для тонкої настройки існуючих моделей, хоча вже не такий гарний для рекурентних мереж. Він вимагає установки деяких програмних пакетів, таких як SciPy-NumPy, BLAS, Intel MKL, OpenBLAS. У Caffe є певний рівень абстракції, який вище, ніж Cuda-Convnet, і нижче, ніж інші, такі як Pylearn і Torch. Caffe включає так званий «зоопарк моделей», що надає безліч попередньо навчених моделей.

3.1.1 Огляд інструментальних засобів

Torch в значній мірі використовується дослідними лабораторіями Facebook AI і Google DeepMind. Замість того, щоб слідувати тренду використання Python, Torch написана на C і Lua. Lua - це скриптова мова високого рівня, призначена для вбудованих пристроїв. Torch дозволяє будувати довільні графи нейронних мереж і розпаралелювати їх на ЦПУ ефективним способом. Torch має велику екосистему пакетів, керованих спільнотою, в сферах машинного навчання, комп'ютерного зору, обробки сигналів, паралельних обчислень, обробки зображень і відео. Torch тяжіє до класу «Тензор» з великим подібністю до масивів Numpy, які виконують ту ж роль, що і тензори в Theano. Це дозволяє визначити глибоку мережу послідовно, як стек шарів, аналогічно Theano і Caffe, і відправляти типи і

обчислення на ДП. Зрештою, у Torch менше готової вбудованої функціональності, ніж у Caffe, тому потрібно більше працювати над написанням коду, але з іншого боку це забезпечує більшу гнучкість. Він має багато модульних частин, які можна комбінувати, а також пропонує безліч попередньо навчених моделей.

3.1.2 Використання графічних прискорювачів

Для тренування нейронних мереж з використанням усіх сильних сторін графічних процесорів використовують пакети, що спеціалізуються на паралельному процесу роботі мережі. Так, для моделі, яка класифікує зображення з датасету CIFAR-10, застосування графічних прискорювачів дало можливість прискорити тренування більш чим на 700%, якщо порівнювати з ЦПУ. Більш того, це ще не межа прискоренню, і процес тренування можна зробити навіть швидше, використовувавши бібліотеку cuDNN від NVIDIA.

На відміну від більш простої альтернативи cuDNN – пакету CUDA від NVIDIA, який надає можливість виконання на GPU усіляких обчислень, cuDNN була розроблена для навчання глибоких нейронних мереж, і в особливості саме таких як згорткові. Вона має оптимізовані під графічні прискорювачі імплементації згорткових і рекурентних мереж, різних функцій активації (ReLU, сигмоїдальна, тангенс та ін.), алгоритму оптимізації і т.д. cuDNN надає можливість тренувати мережі на графічному прискорювачі в кілька разів швидше, ніж просто використовуючи інтерфейс CUDA. Також, що приємно, бібліотека є безкоштовною і її можна завантажити на сайті видавництва.

Подивимося на метод, за допомогою якого бібліотекам із підтримкою тренування на графічних прискорювачах вдається уникнути складності та

повноцінно відтворити навчання глибокої мережі. Практично процес можна розділити на такі етапи [26]:

– графічний прискорювач приймає графічні дані, що є відображенням стану нейронної мережі і містять декілька текстур, що представляють декілька змінних нейронної мережі, більш того, такі фільтри містять текстуру з двомірною адресацією пам’яті, тобто кожна зі згаданих текстур є змінною нейронної мережі;

– на графічному прискорювачі виконують здійснення прямого проходу в штучній нейронній мережі та здійснення зворотнього проходу, причому виконання включає в себе виконання згорткових і рекурентних операцій;

– виконують декілька програм для зміни ваг на мапах ознак в згортковій нейронній мережі через зміну графічних даних базуючись на результатах зворотнього проходу;

– ще раз виконують необхідну кількість програм для здійснення прямих проходів, зворотніх проходів і даних для нової епохи, поки згорткова нейронна мережа повністю не натренується.

Експеремент проведено на графічному прискорювачі NVIDIA Tesla M80 та процесорі марки Intel моделі i7-6060U. Tesla M80 –графічний прискорювач або відеокарта об’ємом 12 Гб та тактовою частотою у 3176 МГц, в основі якого лежить графічний чіп GM708 архітектури Maxwell, який має 684 ядра CUDA і шину пам'яті 256 біт. Оптимізатор PyTorch використовує процесорні інструкції SSE4.X, AVX.

3.2 Програмна реалізація моделі

Після налаштування програмного середовища, можна переходити до етапу програмної реалізації моделі. У цій роботі будемо використовувати втрату двійкової бінарної-ентропії як функцію втрати навчання і

оцінюватимемо мережу на тестовому наборі даних, використовуючи міру IoU. Більша частина коду складається з побудови відповідного класу DataLoader, який надає нам зразки випадкових зображень відповідних їм моделей. Для навчання мережі дуже важливо отримати збалансований набір відносно різних категорій об’єктів. Таким чином, на кожній ітерації ми надаємо однакову кількість об’єктів кожної категорій. Код завантаження даних є змістовним, але в кінцевому рахунку простим: завантажити зображення з однієї з обраних категорій разом із відповідною йому тривимірною моделлю.

Сама архітектура мережі, визначена в скрипті тренування, являє собою згорткову автоенкодерну нейронну мережу, що складається з блок енкодеру та декодеру, кожна з яких містить 9 згорткових шари з розмірами ядра 7, 5 і 5, а також шаром об'єднання. Пройшовши через згорткові шари, ми дозволимо мережі побудувати одновимірний дескриптор кожного входу, вирівнюючи його і передаючи їх через лінійний шар з 512 входами. Це дозволяє мережі дізнатися змістовні дескриптори для кожного входу і робить вихід симетричним (впорядкування вхідних даних не має відношення до нашої мети).

Найважливішим кроком всієї операції є наступний: ми обчислюємо значення функції середньої повоксельної бінарної ентропії для отриманих об’ємів. В загалі-то, для тренування мережі ми могли б використовувати й іншу функцію втрат, яка б оцінювала різницю між увімкненими вокселями реальної моделі та вокселями, що були прогнозовані. Тим не менш, я отримав кращі результати (більш швидку конвергенцію), використовуючи двійкову поперечну втрату ентропії. Тому до мережі додаємо ще один лінійний шар з двома вихідними функціями (рівною кількістю, різним числом), щоб отримати логіти.

У коді програми є три основні функції: функція тренування, функція тестування та функція прогнозування. У функції тренування ми підживлюємо мережу відповідними зображеннями об’єктів, що необхідно реконструювати.

Функція тестування служить для вимірювання точності мережі на тестовому наборі даних. Ми проводимо тест після кожної тренувальної епохи, щоб спостерігати за ходом тренувань і запобігати перенавчанню. Прогнозуюча функція, отримана за допомогою зображень. Можна скористатися методом predict після завершення навчання, встановивши глобальну змінну do_learn на False. У якості вхідного зображення було використано RGB-зображення розміром 224×224 пікселі із розміром батча в 64 елементи. Виходом мережі є квадратна воксельна сітка розміром $32 \times 32 \times 32$ вокселі. Мережа, як зазначено вище, було розроблена на фреймворці PyTorch та оптимізувалася оптимізатором Adam з параметрами $\beta_1 = 0.9$ та $\beta_2 = 0.999$. Початкова норма навчання дорівнює 0.001 та зменшувалася в 2 разі кожні 150 епох. Спочатку мереже тренується без модулю контекстно-незалежного комбінування упродовж 250 епох, після цього параметри мережі заморожувалися і архітектура тренувалася повністю ще 150 епох. Вихідний код програми наведено у додатку А.

3.3 Використані набори даних

В роботі були використані два найбільш повних набори даних з представленими тривимірними моделями об’єктів та їх зображеннями. ShapeNet – це велике, багате інформацією сховище тривимірних даних моделей об’єктів. Він містить моделі, що охоплюють безліч семантичних категорій. На відміну від попередніх сховищ тривимірних моделей, він надає великі набори анотацій для кожної моделі та зв'язки між моделями у сховищі та іншими мультимедійними даними поза сховищем. Як і ImageNet, ShapeNet

надає представлення даних, що містяться в ієрархічній категоризації відповідно до стандарту WordNet. На відміну від інших сховищ моделей, ShapeNet також надає багатий набір анотацій для моделей.

Анотації включають геометричні атрибути, такі як види орієнтації, ключові точки, симетрії фігур (площина відбиття, інші симетрії обертання) та масштаб об'єкта в одиницях реального світу. Ці атрибути забезпечують цінні ресурси для обробки, розуміння та візуалізації тривимірних фігур таким чином, щоб знати про семантику форми. На даний момент зібрано приблизно 3 мільйони фігур з онлайн-сховищ 3D-моделей та класифіковані з них 300 тисяч відповідно до таксономії WordNet. В наборі даних Pix3D зображення отримано двома способами. Одним з них є сканування зображення меблів IKEA з Інтернету та вирівнювати їх з моделями САПР, наведеними у наборі даних IKEA. Інший – безпосередньо сканувати тривимірні форми та фотографувати об'єкти. Набір даних IKEA містить 219 високоякісні 3D-моделі меблів IKEA, але їх лише 759 зображення для 90 фігур. Тому було вирішено зберегти тривимірні фігури із набору даних IKEA, але розширити набір 2D-зображень за допомогою онлайн-пошукових систем зображень та краудсорсингу.

Для кожної тривимірної фігури спочатку шукається відповідне їй зображення через Google, Bing та Baidu, використовуючи назву моделі як ключове слово. Так було отримано 104 220 зображень для 219 фігур. Потім було використано Amazon Mechanical Turk (AMT) для того щоб видалити недоречні. Для кожного зображення три AMT працівники позначали, чи відповідає це зображення 3D-формі або ні. Ці атрибути забезпечують цінні ресурси для обробки, розуміння та візуалізації тривимірних фігур таким чином, щоб знати про семантику форми. На даний момент зібрано приблизно 3 мільйони фігур з онлайн-сховищ 3D-моделей та класифіковані з них 300 тисяч відповідно до таксономії WordNet.

3.4 Огляд отриманих результатів

Модель було натреновано на наборах даних ShapeNet та Pix3D і протестовано на валідаційній частині набору даних ShapeNet та реальних прикладах. Як результат було досягнуто середнього значення IoU в 0.756 на 13 найбільш категоріях широко представлених категоріях об’єктів. Результати реконструкції зображено на рисунку 2.6.

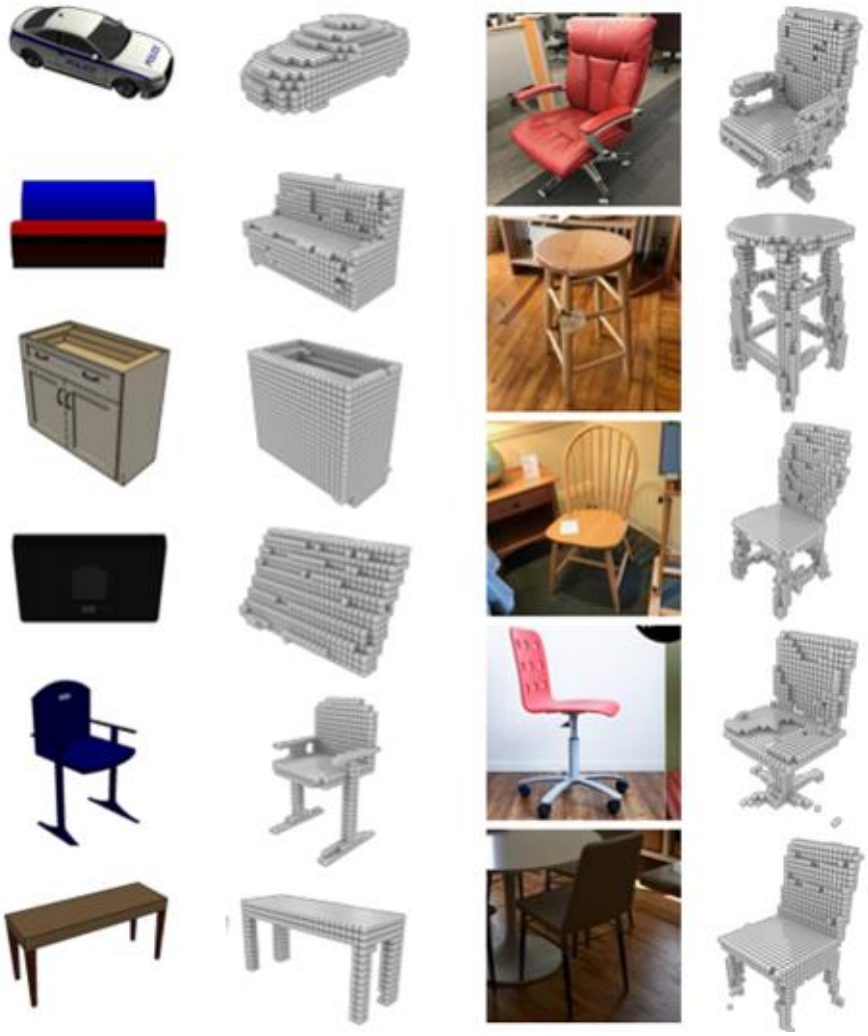

Рисунок 2.6 – Результат реконструкції зображень ShapeNet (зліва) та реальних зображень (зправа)

ВИСНОВКИ

В ході виконання атестаційної роботи було проаналізована важлива проблема комп'ютерного зору, а саме тривимірна реконструкція об'єктів з одного зображення.

Були розглянуті і проаналізовані існуючи методи, виявлені іх недоліки та сильні сторони. Так, загальною проблемою для усіх методів, що базуються на навчанні з вчителем, є відстуність навчального набору даних, який був би достатньо великим і водночас репрезентативним у сенсі варіативності ракурсів об'єктів, оклюзій, оточення та освітлення. Також, проаналізованим методом притаманні такі хибні риси, як низька якість реконструкції об'єктів складної структури, наприклад, тонкі деталі.

Проте, деякі методи, такі як реконструкція з прикладів, не використовують навчання, а лише намагаються змінити шаблон таким чином, щоб ліпше пасувати вхідному зображенню, вирішуючи при цьому задачу класифікації, та на жаль, вони програють у деталізації, хочу й досить якісно відновлюють опуклі об'єкти.

Методи ж, що базуються на глибокому навчанні, показують набагато якісніщі реконструкцій, та вирішують проблему невеликої кількості тренувальних об'єктів синтетичними даними.

Також у роботі було запропоновано вдосконалену архітектуру нейронної мережі, що базується на тривимірних згортках та архітектурі, що є комбінацією блоків енкодингу-декодингу та блоку фіналізації реконструкції для подальшої корекції отриманих результатів.

Отримані результати є задовільними у сенсі використаної метрики оцінки якості, проте, через відносну низьку ємність відображення, що пояснюється наявністю обчислювальних потужностей, може скластися враження низької деталізації отриманої реконструкції.

Проте, можна сказати, що у майбутньому така архітектура може розвиватися й надалі. Таким чином, результатом роботи є розроблена архітектура, що виконує реконструкцію об'єкта з єдиного його зображення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. A. Kundu, Y. Li, and J. M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018.

2. V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In Proceedings of the 26th annual conference on Computer graphics and interactive techniques.

3. Shichen Liu1, Tianye Li1, Weikai Chen, Hao Li. Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning, Oct. 2019 URL: https://arxiv.org/abs/ 1904.01786 (дата звернення 27.05.2019)

4. A. Kar, C. Hane, and J. Malik: Learning a multi-view stereo machine. 2012. 331–363 p.

5. Zhao J., Mathieu M. et al., Stacked what-where auto-encoders Jun. 2015Hardt M., Ma T. Identity matters in deep learning Nov. 2016. URL: https: //arxiv.org/abs/1611.04231 (дата звернення 26.05.2019).

6. Yichao Z., Yi Ma. Learning to Detect 3D Reflection Symmetry for Single-View Reconstruction. Cambridge, MA: MIT Press. 2019. ISBN 978-0262581110.

7. Huang G., Liu Z., Weinberger K. Q. Densely connected convolutional networks, Aug. 2016

8. P. J. Bentley and S. Kumar. The ways to grow designs: A comparison of embryogenies for an evolutionary design problem. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-1999), pages 35–43, San Francisco, 1999. Kaufmann.

9. Bergstra J, Bengio Y. Random search for hyper-parameter optimization // Journal of Machine Learning Research. Feb. 2012. vol. 13. P. 281–305.

10. Dropout: a simple way to prevent neural networks from overfitting. / N.

Srivastava, G. E. Hinton et al. // Journal of Machine Learning Research. 2014. vol. 15. no. 1. P. 1929–1958.

11. On an eigenvalue inequality involving the Hadamard product / Hiai, Fumio; Lin, Minghua. February 2017.

12. Ioffe S., Szegedy C., Batch normalization: Accelerating deep network training by reducing internal covariate shift. Feb. 2015.

13. J. Clune, K. Stanley, R. Pennock, and C. Ofria. On the performance of indirect encoding across the continuum of regularity. Evolutionary Computation, IEEE Transactions on, 15(3): 346–367, 2011.

14. G. Cybenko. Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals, and Systems, 2(4):303–314, 1989.

15. Howard A. G. Some improvements on deep convolutional neural network based image classification Dec. 2013.

16. J. Drchal, J. Koutnık, and M. Snorek. HyperNEAT controlled robots learn to drive on roads in simulated environment. In Proceedings of the IEEE Congress on

17. Chetlur S., Woolley C. cuDNN: Efficient primitives for deep learning, Oct. 2014 URL: https://arxiv.org/abs/1410.0759 (дата звернення 27.05.2019)

18. F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In European Conference on Computer Vision, pages 561–578. Springer, 2016.

19. Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh: Realtime multi-person 2d pose estimation using part affinity fields, Oct. 2018 URL: https://arxiv.org/abs/1812.08008 (дата звернення 27.10.2020)

20. X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman: Pix3d: Dataset and methods for single-image 3d shape modeling, 2016

21. Weiyue Wang et al.: DISN: Deep Implicit Surface Network for High-quality Single-view 3D Reconstruction, May. 2019 URL: https://arxiv.org/abs/1905.1071 (дата звернення 27.10.2020)

22. Yichao Zhou, Shichen Liu: Learning to Detect 3D Reflection Symmetry for Single-View Reconstruction, Oct. 2020 URL: https://arxiv.org/abs/2006.1004 (дата звернення 27.10.2020)

23. M. Wang, L. Wang, and Y. Fang. 3DensiNet: A robust neural network architecture towards 3D volumetric object prediction from 2D image.

24. E. Mjolsness, D. H. Sharp, and J. Reinitz. A connectionist model of development. Journal of Theoretical Biology, 152:429–453, 1991.

25. S. Risi and K. O. Stanley. Indirectly encoding neural plasticity as a pattern of local rules. In S. Doncieux, B. Girard, A. Guillot, J. Hallam, J.-A. Meyer, and J.-B. Mouret, editors, From Animals to Animats 11, volume 6226 of Lecture Notes in Computer Science, pages 533–543. Springer Berlin / Heidelberg, 2010.

26. E. R. Kandel, J. H. Schwartz, and T. M. Jessell, editors. Principles of Neural Science. Elsevier, Amsterdam, third edition, 1991.

27. Chetlur S., Woolley C. cuDNN: Efficient primitives for deep learning, Oct. 2014 URL: https://arxiv.org/abs/1410.0759 (дата звернення 27.05.2019)