НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

На правах рукопису
УДК 004.934.5

До захисту допущено
В. о. завідувача кафедри ММСА
О.Л.Тимощук
«___» _____ 2020 р.

Магістерська дисертація

на здобуття ступеня магістра за спеціальністю 122 Комп'ютерні науки
на тему: «Система виділення людського голосу з зашумленого аудіозапису з використанням глибокого навчання»

Виконав:
студент ІІ курсу, групи КА-83 мн
Харченко Дмитро Олександрович _____

Науковий керівник:
Дідковська Марина Віталіївна,
доцент кафедри ММСА ІПСА
КПІ ім. Ігоря Сікорського,
канд.техн.наук, доц. _____

Рецензент:
Вунтесмері Юрій Володимирович,
доцент кафедри електронної інженерії
КПІ ім. Ігоря Сікорського,
канд.техн.наук, доц. _____

Засвідчую, що у цій магістерській дисертації
немає запозичень з праць інших авторів
без відповідних посилань

Студент _____

Київ
2020

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Рівень вищої освіти — другий (магістерський)
Спеціальність (спеціалізація) — 122 «Комп'ютерні науки» («Системи штучного інтелекту»)

ЗАТВЕРДЖУЮ
В. о. завідувача кафедри ММСА
О. Л. Тимощук
«___» _____ 2020 р.

ЗАВДАННЯ
на магістерську дисертацію студенту Харченку Дмитру Олександровичу

1. Тема дисертації: «Система виділення людського голосу з зашумленого аудіозапису з використанням глибокого навчання», науковий керівник дисертації Дідковська Марина Віталіївна, к.т.н., доцент, затверджені наказом по університету від «07» квітня 2020 року № 959-с

2. Термін подання студентом дисертації: 13 травня 2020 р.

3. Об'єкт дослідження: виділення людського голосу

4. Предмет дослідження: глибоке навчання для обробки шуму в аудіо

5. Перелік завдань, які потрібно розробити:

1) дослідити існуючі методи зменшення шумів на звукозаписах;
2) провести порівняльний аналіз досліджених методів;
3) спроектувати систему виділення людського голосу;
4) реалізувати один з модулів спроектованої системи;
5) розробити стартап-проект виведення на ринок результатів дослідження.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:

1). Архітектури досліджених нейронних мереж
2). Блок-схема спроектованої системи
3). Графіки навчання нейронних мереж
4). Архітектура запропонованої модифікації
5). Принципи роботи згорткових нейронних мереж
6). Принципи роботи рекурентних нейронних мереж
7). Принципи роботи генеративно-змагальних нейронних мереж

7. Орієнтовний перелік публікацій:

(1) Харченко Д. Пришвидшена генеративно змагальна мережа для відділення голосу від шуму на звукозаписі. *Наука онлайн: Міжнародний електронний науковий журнал*. 2020. №5. URL: https://nauka-online.com/ua/publications/tehnicheskie-nauki/2020/5/prishvidshena-generativno-zmagalna-merezha-dlya-viddilennya-golosu-vid-shumu-na-zvukozapisi/

8. Дата видачі завдання: 10 березня 2020 року

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації
1	Концептуальний вступ дисертації. Формулювання об'єкта, предмета, цілі, завдань, новизни, практичної значущості результатів	10.03.2020 — 15.03.2020
2	Огляд технічної літератури за темою	16.03.2020 — 20.03.2020
3	Другий розділ. Аналіз можливих методів оптимації моделі	21.03.2020 — 01.04.2020
4	Вибір методів аналізу	02.04.2020 — 07.04.2020
5	Тренування нейронних мереж	08.04.2020 — 02.05.2020
6	Аналіз отриманих результатів	03.05.2020 — 05.05.2020
7	Оформлення звіту, формулювання висновків	05.05.2020 — 10.05.2020

Студент Д.О. Харченко

Науковий керівник дисертації М.В. Дідковська

РЕФЕРАТ

Магістерська дисертація: 103 с., 36 рис, 25 табл і 52 джерела.

ПОКРАЩЕННЯ МОВЛЕННЯ, ГЛИБОКЕ НАВЧАННЯ, ГЕНЕРАТИВНІ ЗМАГАЛЬНІ МЕРЕЖІ, РЕКУРЕНТНІ МЕРЕЖІ, ЗГОРТКОВІ МЕРЕЖІ, ЗВУКОЗАПИС, PYTHON, ОБРОБКА СИГНАЛІВ

Мета даної роботи – дослідити системи на базі глибокого навчання, що виділяють людський голос з поміж фонових шумів. Основною ціллю роботи стали дослідження можливості використання глибокого навчання для задачі фільтрації шуму на звукозаписі.

Об’єктом дослідження даної магістерської дипломної роботи є виділення людського голосу. Предмет дослідження – глибоке навчання для обробки шуму в аудіо.

Актуальність дослідження полягає у невирішеності питання видалення шумів із запису мови, окрім недавньої RTX Voice, що має суттєві обмеження в технічному (Nvidia) та програмному забезпеченні (Windows).

Були досліджені 3 провідні архітектури глибокого навчання, що спроектовані для задачі покращення мовлення. Перша - SEGAN заснована на генеративних змагальних мережах, наступна - WaveNet модифікований для прибирання шуму та остання - EHNet, що використовує рекурентні та згорткові мережі. Проведено порівняльний аналіз досліджених архітектур та запропоновано та реалізовано модифікацію SEGAN, що пришвидшує швидкість роботи та навчання. Також запропонована архітектура системи, що побудована на базі запропонованої модифікації.

ABSTRACT

Master’s thesis: 103 pp., 36 fig., 25 tab., 52 sources.

VOICE BIOMETRY, PERSONAL AUTHENTIFICATION, LIMITATED COMPUTING RESOURCES, FAST BIOMETRY, MFCC, CLASSIFIERS, PYTHON

The purpose of this work is to investigate systems based on deep learning that distinguish the human voice from the background noise. The main goal of the work was to study the possibility of using deep learning for the problem of noise filtering on sound records.

The object of study of this master's thesis is the selection of the human voice. The subject of research -deep learning for denoising in audio.

The relevance of the study lies in the unresolved issue of removing noise from speech recording, except for the recent RTX Voice, which has significant limitations in hardware (Nvidia) and software (Windows).

Three leading deep learning architectures designed for the task of improving speech were studied. The first is SEGAN based on generative competition networks, the next is WaveNet modified for noise removal and the last is EHNet, which uses recurrent and convolutional networks. A comparative analysis of the studied architectures was performed and a modification of SEGAN was proposed and implemented, which accelerates the speed of work and learning. Also there was proposed the architecture of the system, built on the proposed modification.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

GAN – Generative adversarial network

МН – Машинне Навчання

НМ – Нейронні Мережі

ШІ – Штучний Інтелект

ШНМ – Штучні Нейронні Мережі

MLP – Multilayer Perceptron

NLP – Natural Language Processing

ВСТУП

У постійно зростаючому глобалізованому світі спілкування є важливою частиною повсякденного життя. Мова - найкращий комунікативний метод, який забезпечує найбільше контексту для повідомлення. Однак є багато сценаріїв, коли мова не може бути використана, і ми змушені вдаватися до тексту чи інших носіїв інформації. В цій роботі розглядається один із аспектів, які заважають доступності мови: фоновий шум. Область дослідження полягає у поліпшенням мови і спрямована на усунення фонового шуму, щоб внести чіткість і зрозумілість у зразок мови.

Вдосконалення мови - важливий предмет, який має кілька можливих застосувань у різних галузях. Основним додатком було б зниження шуму при розмовах через мобільні телефони та подібні технології. Одним із неочевидних застосувань може стати використання на суперечливих записах із громадськими діячами, що було забруднене галасливим фоном. У світі, де правильна інформація є запорукою істини, а дезінформація зараз поширюється як пожежа, ясність має велике значення. Прикладом цього може бути аудіокліп світового лідера, що вимовляє щось у переповненій кімнаті, яку підбирає телефон небіжчика. У цьому аудіокліпі мова не зрозуміла через оточення, і розмова, що може дискрридитувати одного з найпотужніших лідерів у світі, стає непевною. За допомогою технології вдосконалення мови можна уникнути непорозумінь.

Іншим способом застосування є використання покращення мовлення в як безпосередньо шумового фільтру в режимі онлайн. Тобто використання його як прошарок між звуко записуючим пристроєм та додатком що використовується для онлайн розмови.

Ще одним звичайним застосуванням удосконалення мовлення було б використання його як інструменту для розпізнавання мовлення, оскільки шумні файли вже не вважалися б нежиттєздатними як дані. Це було б навіть

ігровим полем для незалежних творців, які можуть записувати свої подкасти, інтерв'ю, аудіокниги тощо будь-де без потреби у професійному обладнанні.

Таким чином, **ціллю роботи** є розробка архітектури системи фільтрації шумів в реальному часі, опис принципів роботи цієї системи та практична реалізація одного з її модулів.

Об’єктом дослідження є виділення людського голосу та системи обробки звуку.

Предметом дослідження є глибоке навчання у задачах обробки звуку.

Робота складається з 4-х розділів. В першому розділі розглядається сучасний стан речей у галузі фільтрації шуму на звукозаписі з мовою. В другому розділі наведено теоретичні основи що необхідні для розуміння принципів роботи модуля системи що розробляється. Третій розділ присвячено розробці архітектури системи, практичній реалізації одного з її модулів, а саме нейронної мережі що безпосередньо виконує фільтрацію. Також в третьому розділі описано результати експерименту з навчання та тестування нейронних мереж. В останньому, четвертому, розділі розглядається цільова система як стартап проект.

РОЗДІЛ 1 ОГЛЯД ОБЛАСТІ

1.1 Виділення мови

В англомовній літературі часто зустрічається поняття speech enhancement. Воно означає “покращення розбірливості мови”. Загалом під покращенням розбірливості мається на увазі прибирання фонових шумів, проте іноді можна зустріти інше, наприклад трансляція шопоту в звичайну мову[9]. В даній дисертації розглядається саме прибирання шумів, та використовується поняття “виділення мови” для такого звуження.

Методи фільтрації шуму можна початково розділити на дві групи:

- активні
- пасивні

Перші використовують окремий пристрій звукозапису для отримання інформації про шум задля його видалення. Такі системи зазвичай використовують при професійному звукозаписі. Пасивні ж використовують лише оригінальний звукозапис що ускладнює задачу, проте не вимагає від системи додаткового пристрою, що дає змогу використовувати їх пересічному користувачу, проте значно ускладнює саму задачу фільтрації. В даній роботі розглядається саме пасивна фільтрація, оскільки в сфері активної фільтрації вже досягнуто значних успіхів.

Значним фактором для методів вирішення задачі пасивної фільтрації є тип шуму. Більшість математичних методів фільтрації мають припущення щодо форми та ступеня шуму. Оскільки в загальних умовах це припущення не вірно, то кожний метод використовується у вузько направлених задачах. В загальному випадку, використовують поріг за силою звуку, щоб прибрати звук під час пауз. В даній роботі розглядається саме фільтрація шуму в загальному випадку.

1.2 Загальне становище області

Тоді як обробка сигналів та методи базовані на підпросторах історично були домінуючими в теорії покращення мовлення, галузь сьогодні переходить на більш новітні підходи. В інформатиці існує сфера, яка була популяризована останнім часом, усім відоме машинне навчання (ML). Воно вже було використано для надзвичайно різних галузей промисловості та досягло успіху майже в усіх. Причиною цього є поліпшення обчислювальної потужності за допомогою GPU(графічих прискорювачів), оскільки NVIDIA та AMD зробили наступний крок на шляху до ШІ. Настільки революційні алгоритми побудовані на фундаментах статистичної теорії, що практично були обмежені часом обчислення в минулому. Однак, як і у всіх дослідженнях, люди завжди будуть прагнути вдосконалення. Для гігантів галузі, таких як Google, це означає створення нових комбінацій та нових ітеративних алгоритмів, що можуть переглядати практичне рішення раніше важких проблем, таких як AlphaGo та ін. [1]. Незважаючи на це, не всі дослідження потребують гігантських корпорацій оскільки існують інновації, спричинені саме дефіцитом даних.

1.3 Нові моделі

Сучасні нововведення, що отримали широке розповсюдження, розроблені на основні поняття звичної теорії глибокого навчання. Значна частина нових архітектур - це природне розширення попередніх ідей, просто доведених до наступного рівня. Одні поєднують дві успішні архітектури в одну, а інші просто запозичили поняття в інших областях і застосовували їх з великим успіхом у нових сферах.

Основними розробками в даній галузі є наступні архітектури:

- WaveNet
- SEGAN

Обидві архітектури в основі мають ідею вилучення “суті” з даних для видалення спотворень, в нашому випадку шумів.

1.2.1 WaveNet

Серед певних нововведень, що були впроваджені, виділяється нова архітектура WaveNet, генеративна модель для необробленого аудіо. Це поступова авторегресивна модель, яка стала провідною для генерації мовлення. Модель заснована на Pixel CNN архітектурі, що закріпила своє значення в генеруванні зображень. Ця архітектура настільки вдала, що її модифікації дали значні результати в інших сферах.

Говорячи про галузь покращення мовлення, недавно розвинена архітектура під назвою WaveNet, що базується на оригінальній архітектурі, та спрямована на зменшення шуму в забруднених аудіофайлах для покращення мови.

1.2.2 Генеративно змагальні мережі

Прикладом алгоритму, який може бути конкурентом або альтернативою для WaveNet є нова концепція генеративної змагальної мережі (GAN). Ця нова концепція була надзвичайно вдалою для створення зображення та тегування зображень. Було б цікаво спробувати адаптувати метод з візуального до аудіо формату, оскільки отримати аналітичне представлення даних цілком досяжна задача.

GAN були представлені на сцені машинного навчання як повноцінна архітектура у 2016 році, проте перший його приклад можна знайтиу 2014 р. [2]. З цього моменту з'явилася велика кількість варіацій, що охоплює широкий діапазон тем.

Головною особливістю даної архітектури є побудова двух мереж що змагаються. Одна виконує роль генератора нових “фейкових” данных, коли інша намагається відрізнити згенеровані дані від справжніх.

Окрім задачі генерування, ця архутектура також отримала розповсюдження в задачах навчання без учителя.

Серед великого різноманіття, нашу увагу привертає SEGAN - Speech Enhancement GAN. Вона використовує згорткові мережі задля зменшення шумів з аудіо що містить мову. Запропонована Паскалем, Бонафортом, та Серра [5].

Повний список GAN можна знайти на github GAN-Zoo від Hindupur [6], якщо читач зацікавлений досліджувати більше.

1.4 Актуальність видалення шумів з запису з мовленням

На сьогодні штучні нейронні мережі відіграють ключову роль як інструмент в машинному навчанні та штучному інтелекті. Адже вони мають чимало безумовних переваг:

- не обмежені конкретною областю застосування, їх використовують в багатьох різних областях науки. Це дає можливість менше заглиблюватись у конкретну галузь, ставши спеціалістом з штучних нейронних мереж. Тобто дає можливість навчати більш універсальних розробників, що здатні виходити за вузькі рамки однієї галузі;
- потребують меншої обробки даних у порівнянні з математичними

методами;

- часто показують кращі результати за математичні моделі;
- на відміну від математичної моделі, мають властивість паралелізму. Тобто швидкість їх розробки можна збільшити додавши ресурсів.

Хоча перші нейронні мережі винайдені ще у середині минулого століття, значної популярності вони набули лише з 2012 року, що здебільшого зумовлено підтримкою виробників графічних прискорювачів.

Головним чинником такого росту є вдала реалізація алгоритмів навчання на графічних прискорювачах(GPU). GPU, в якості основного завдання, виконують паралельне обчислення зображень. Саме тому вони здатні помітно прискорити навчання нейронної мережі, оскільки його можливо вдало розпаралелити.

Говорячи про задачу видалення шумів з мови, в цьому році виробник GPU у тестовому режимі показав технологію RTX Voice[3]. Ця технологія використовує графічний прискорювач користувача, для очищення аудіо потоку під час онлайн розмови. На жаль технологія є приватною, та невідомо що саме вони використовують, проте найбільш ймовірно, що в основі знаходиться нейронна мережа. Оскільки ця приватна технологія, можна сказати, є першою вдалою спробою фільтрації для звичайного користувача (раніше фільтри використовували для професійних цілей, а в інших випадках використовували ковзкий поріг відсіву).

Значний успіх даної технології доводить перспективність використання штучних нейронних мереж для такої задачі. Також, оскільки виробник використовує цю технологію як одну з головних переваг нових пристроїв, можна зробити висновок, що у користувачів існує потреба в подібній технології.

Попри існування відкритих архітектур для даної задачі вони не відповідають вимогам для системи в реальному часі, оскільки одні архітектури недостатньо швидкі, інші мають значно гіршу якість результатів. Водночас, при існуванні вдалої реалізації цільової системи, її використання обмежено

однією операційною системою та одним виробником графічних прискорювачів. Таким чином існування такої системи підігріває інтерес для користувачів інших операційних систем чи апаратних складових.

1.5 Постановка задачі

Отже, з вище сказаного, робимо висновок, що на сьогодні не створена відкрита та кросс-платформенна система для фільтрації запису мови від шуму, задовільна як за якістю роботи так і за швидкістю рівня «в реальному часі».

Завданням даної роботи є:

1. Вдосконалення існуючих підходів задля вирішення проблеми відповідності вимогам;
2. Проектування архітектури системи що проводить фільтрування в реальному часі;
3. Реалізувати модуль цієї системи, що відповідає за безпосереднє фільтрування.

1.6 Висновки до розділу

Протягом останніх років, починаючи з 2012 року, машинне навчання, зокрема штучні нейронні мережі, набули великої популярності. Здебільшого це відбулося завдяки появі нового обладнання, а саме потужних графічних прискорювачів – GPU, та його пристосування для задач машинного навчання. Іншим вагомим фактором розвитку, є попит користувачів на більш якісні технології, особливо в проблемах, що важко вирішуються математичними методами.

Серед них, вже добре розвинуті напрями такі як машинний Зір та машинна обробка природної мови. В обох напрямках шум та його фільтрація відіграють значну роль у якості, проте окремо ця проблема як напрям розвинена досить слабко.

Використання машинного навчання є перспективним напрямом дослідження, оскільки щороку з’являється нові моделі, та покращуються старі, водночас зі збільшенням обчислювальної можливості технічного забезпечення пересічного користувача, що уможливлює створення відкритих систем для широкого користування.

В результаті сказаного вище, в якості дослідження було обрано виділення мови як метод фільтрації шумів за допомогою глибокого навчання.

Актуальність цього дослідження доводить нещодавня розробка комерційного продукту від впливового виробника GPU, що вирішує ту саму проблему, вірогідно подібними методами. Осільки їх рішення суттєво обмежене платформою та вони показали що існує значний попит на подібну технологію. Іншим фактором є також те, що наукова спільнота потребує відкритих рішень, оскільки такі рішення можна покращити або перенести в інші напрями.

РОЗДІЛ 2 ОГЛЯД МАТЕМАТИЧНИХ МЕТОДІВ ВИДІЛЕННЯ ЛЮДСЬКОГО ГОЛОСУ

2.1 Теоретичне підґрунтя

Лін, Тегмарк та Ролник [11] стверджують, що зі збільшенням зацікавленості людей темою машинного навчання та великих даних, останні розробки переходять від створених вручну, ретельних та повністю зрозумілих аналітичних алгоритмів до архітектурного підходу з алгоритмом глибокого навчання. Вони зазначають, що їх можна зрозуміти лише на евристичному рівні, де ми емпірично знаємо, що певні протоколи тренувань приведуть до відмінних результатів.

У своїй роботі вони намагаються пояснити ефективність глибокого навчання у фізиці, але досі не можна дійти до остаточного висновку щодо того, чому певні нейронні мережі працюють краще, ніж інші. Водночас, бути сліпим до основних факторів ніколи не було ознакою хорошої науки, оскільки це може призвести до втрати розуміння щодо напряму розвитку дослідження. Тим не менш, цей архаїчний пошук розуміння не може конкурувати з тим, що глибоке навчання часто може працювати від початку і до кінця, при цьому не вимагає часто обмежуючого чинника вилучення клінічних особливостей, підкріплених фундаментом наукової теорії.

Одні можуть повірити, що таке дослідження ніщо інше, як просто кидання дротиків на дошку і що випадкові спроби та помилки можуть досягти покращених показників. Це, очевидно, помилкове судження, оскільки все ще потрібне розуміння різних основ. Для цієї дисертації необхідні знання з основ теорії вдосконалення мови, загальної теорія глибокого навчання та більш поглиблені знання про генеративні моделі.

Основні складові трьох моделей, що розглядаються:

a) SEGAN

- Конволюційні нейронні мережі

- Обхідні з’єднання
- Генеративні змагальні мережі

б) WaveNet Denoising

- Загальна архітектура WaveNet
- Конволюційні нейронні мережі

в) EHNet

- Згорткові нейронні мережі
- Виділення спектрограми
- Рекурентні мережі з довгою короткочасною пам’яттю
- Повнозв’язні мережі

Для того щоб розглядати архітектури нейронних мереж для виділення голосу, спочатку необхідно розібратися з поняттям «виділення голосу», основними поняттями теорії обробки сигналів та безпосередньо теорії нейронних мереж.

2.2 Вдосконалення мови

Термін "вдосконалення мови" описує більш вузьке поняття ніж зменшення шуму, оскільки зменшення шуму може бути більш загальним в залежності від проблеми яка вирішується. В даній роботі будуть використані різні назви цього поняття, такі як «виділення мови» чи «покращення мови». Мета, що стоїть за вдосконаленням мови, полягає у визначенні, де знаходиться шум в аудіо, і видаленні його, що покращує ясність, зрозумілість та мелодійність. Алгоритми, що використовується для вдосконалення мовлення різняться. З історичної точки зору, обробка сигналу була найбільш поширеним методом для видалення шуму. Однак, останнім часом сфера глибокого навчання отримала деякі нові технологій.

Основна проблема вдосконалення мови - це дилема рівня шуму і

паритету мовлення [12]. З тим, як працює обробка сигналів, особливо при використанні лінійних фільтрів або методів що використовують підпростори, зменшення рівня шуму призводить до спотворення мовлення, в той час як посилення мовлення приводить до більшої кількості артефактів. Так само, як і декомпозиція на зсув та дисперсію [13], яку ми можемо побачити в машинному навчанні, це чутливе налаштування та оптимізація для досягнення найкращих результатів.

У той час як більшість старих моделей використовують спектрограмний аналіз та вилучення ознак, нові алгоритми глибокого навчання намагаються усунути ці етапи на користь нейронних мереж від початку до кінця(end-to-end). Хоча розглянуті далі моделі вважаються end-to-end, це іноді піддають сумніву, оскільки вони все ж таки потребують певної первинної обробки даних.

2.2.1 Оцінка виділення мови

Чіткість, розбірливість і приємність - все це дуже суб'єктивно для людини. Тому оцінити математично, наскільки добре працює результат, не так просто. Однак, це не зупинило людей від спроб, оскільки є ще деякі об'єктивні оцінки, які можна використовувати.

Для нашого об'єктивного оцінювання ми застосували перцептуальну оцінку якості мови (PESQ) [14] та короткочасної об'єктивної розбірливості(STOI) [15]. Обидва методи оцінки співвідносяться з розумінням мовлення та намагаються моделювати оцінки людського сприйняття. Ху та Лойзу [16] зауважують, що PESQ є найбільш складною оцінкою для обчислення і рекомендований ITU-T, є одним із трьох департаментів Міжнародного союзу електрозв'язку. Він дає оцінку від -0,5 до 4,5, коли більш високі показники вказують на вищу якість.

STOI - це новіший метод оцінки, який має високу кореляцію з

розбірливістю як звичайних зашумлених, так і часово-частотних зважених зашумлених записів. Він дає значення від 0 до 1, де більш високі бали вказують на більш високу якість.

Теорія, що стоїть за обома моделями оцінювання, досить обширна і оскільки ця робота в основному зосереджена на глибокому навчанні в рамках вдосконалення мовлення, ми закликаємо зацікавленого читача вивчити джерела самостійно.

Хоча всі об'єктивні методи оцінки можуть бути лінійною комбінацією один з одного все ще є інтерес бачити їх окремо.

Також, оскільки система призначена для пересічного користуваача, використаємо найпростіший метод оцінки - людську оцінку. Оцінювання проводиться за шкалою від 1 до 5, де:

- 5 - Відмінно: Чиста мова без спотворень та помітних шумів;
- 4 - Дуже добре: мовлення з природним звучанням, але є і деякіпомітні фонові шуми;
- 3 - Гаразд: або фоновий шум занадто сильний, або якість мови знизилася (без фонового шуму);
- 2 - Дуже погано: мова важко зрозуміла, та домінуючий фоновий шум;
- 1 - Жахливо: не схоже на людську мову.

Це схоже на шкалу оцінювання, яку використовують Паскаль, Бонафонт та Серра у своїй роботі[5].

2.3 Обробка сигналів

У той час як новіші моделі в основному з’являються у сфері глибокого та машинного навчання, з історичної точки зору ми все ще маємо фундаментальну теорію обробки сигналів, яка була найсучаснішою донедавна.

У цій магістерській дисертації ми розглядаємо два поняття, які часто використовуються в теорії обробки сигналів: фільтр Вінера, який ми будемо використовувати як орієнтир і віконне перетворення Фур'є як загальний спосіб отримання спектрограми.

2.3.1 Фільтр Вінера

Фільтр Вінера - один з основних фільтрів в обробці сигналів. Рівняння, яке визначає кінцеву тривалість імпульсного відгуку(FIR) фільтра Вінера розглянуте Васегі [17] як:

$$\hat{x}(m) = \sum_{k=0}^{P-1} \omega_k y(m-k) = w^T y \,, \tag{2.1}$$

де $w = [w_0, w_1, \dots, w_{P-1}]$ – вектор коефіцієнтів фільтра Вінера, m дорівнює інедексу дискретного часу і $y = [y(m), y(m-1), \dots, y(m-P-1)]$ є вхідним сигналом фільтра.

Весь процес зображено на рисунку 2.1 де вхідний стаціонарний сигнал y фільтрується вінерівським вектором для отримання наближення до потрібного сигналу x (m).

Функція помилки визначається як:

$$e(m) = x(m) - \hat{x}(m) \tag{2.2}$$

Об'єктивним критерієм в теорії Вінера є метод найменших квадратів і розраховується як:

$$[E[e^2(m)] = E[(x(m) - w^T y)^2] =$$
$$= E[x^2(m)] - 2w^T E[yx(m)] + w^T E[yy^T]w =$$

$$= r_{xx}(0) - 2w^T r_{xy} + w^T R_{yy} w \ , \quad (2.3)$$

де $R_{yy} = E[y(m), y^T(m)]$ – автокореляційна матриця вхідного сигналу y і $r_{xy} = \mathrm{E}\,[\mathrm{x}\,(\mathrm{m})\,\mathrm{y}\,(\mathrm{m})]$ є коваріаційною матрицею вхідних та цільових сигналів. Щоб мінімізувати середньо квадратичну помилку, ми отримуємо фільтр Вінера як:

$$w = R_{yy}^{-1} r_{yx} \ , \quad (2.4)$$

що означає, що матриця автокореляції та коваріаційна матриця - це необхідні два компоненти для обчислення нашого оптимального фільтру Wiener.

2.4 Віконне перетворення Фур'є

Звуковий файл має багато властивостей, які можна використовувати як дані. Більшість моделей ML рухаються до повного кінцевого навчання без вилучення будь-яких ознак, все ж є деякі методи, які можуть зробити дані що краще відповідають моделі. Оскільки аудіо часто є одновимірними даними, це може стати обмеженням для вивчення відносин між точками даних. Віконна трансформація Фур'є або STFT – це спосіб витягнути більше інформації з аудіо хвилі.

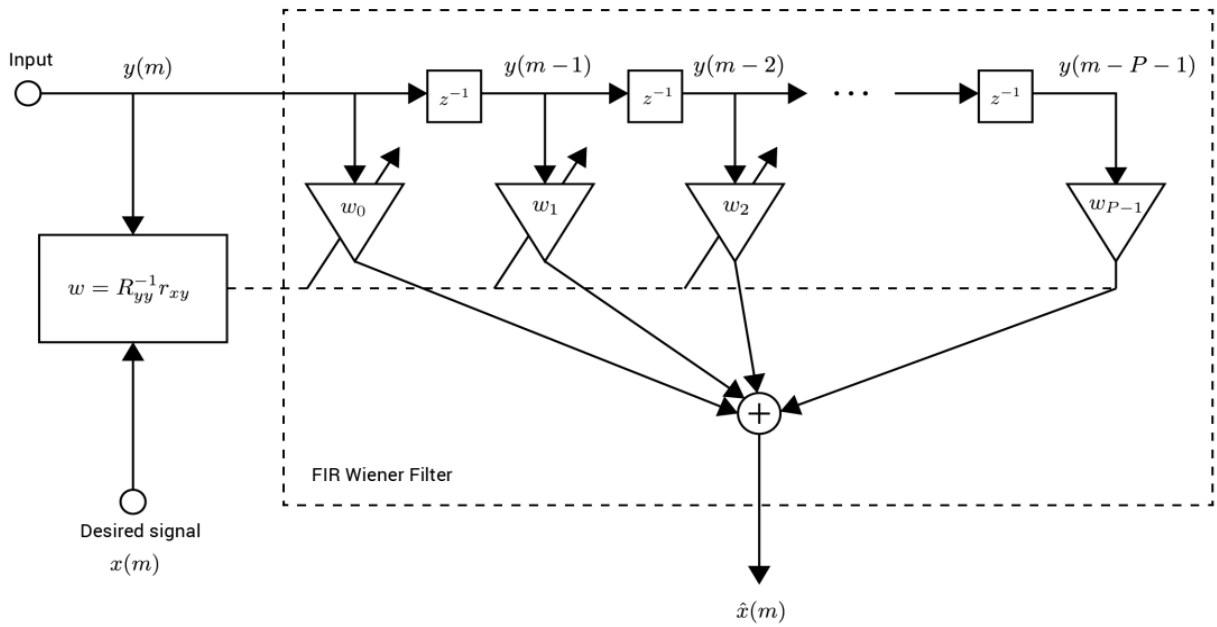

Рисунок 2.1 – Структура фільтра Вінера. Вхідний сигнал зі статичним шумому (m) фільтрується фільтром Вінера і виробляє x (m). Зображення адаптоване від Васегі [17].

Нехай x (n) – сигнал, визначений для всіх n, і $X_n\left(e^{j\omega k}\right)$ – віконне перетворення Фур'є з x (n), оцінене в момент n і частотою ω_k. Тоді ми можемо визначити STFT як:

$$X_n\left(e^{j\omega k}\right) = \sum_{m=-\infty}^{\infty} \omega(n-m)x\,(m)e^{-j\omega_k m}, \qquad (2.5)$$

де $\omega(n)$ – функція вікна (фільтр низьких частот). З цього часто роблять спектрограму що являє собою двовимірний графік за часом та частотою. Модель, розглянута в цій роботі, EHNet, використовує ці властивості, аби застосувати нейронну мережу на 2D простір, де вихідні дані є 1D.

2.5 Машинне навчання

Визначення машинного навчання змінюється в залежності від джерела. Викладачі зі Стенфорда описують це так “... науку про те, як змусити

комп'ютери діяти, не будучи виключно програмованими". Фаггелла [18] склав список цитат, де є представники галузі, такі як NVIDIA, які замість цього вирішили використовувати складніше означення "Машинне навчання у своїй основі – це практика використання алгоритмів для аналізу даних, навчання з ними, а потім знаходження рішення або передбачення чогось у світі" . Люди не з галузі, такі як McKinsey & Co., використовують інше визначення, схоже на цитату зі Стенфорда: "Машинне навчання засноване на алгоритмах, які можуть навчатися на даних, не покладаючись на правила програмування ".

Правда полягає в тому, що всі ці визначення можуть бути правильними, машинне навчання - це все перелічене і водночас, щось посередині. Це широкий термін, саме тому дослідники роблять уточнення щодо того, з чим вони працюють.

Машинне навчання, як було вже сказано раніше, має багато застосувань у різних областях. Вдосконалення мови, ми можемо вважати проблемою видалення шумів та наслідувати Goodfellow, Bengio та Courville [19] які визначають його так: "… алгоритм задається вводом пошкодженого зразка $\tilde{x} \in R_n$ який отриманий шляхом невідомого пошкодження із чистого зразра $x \in R_H$. Система що навчається повинна передбачити умовний розподіл ймовірностей $\mathrm{p}(x \mid \tilde{x})$".

Спосіб навчання може бути різноманітним, а алгоритм – часто визначається наявними даними. Однак ці підходи завжди знаходяться в межах трьох категорій: навчання з вчителем, напівавтоматичне навчання та навчання без учителя [20].

2.5.1 Навчання з вчителем

Розповсюдженим типом набору даних, з яким найчастіше доводиться працювати, є набір даних де є вхідні вектори x з відповідними присутніми

цільовими значеннями *y*. Метою є створення моделі машинного навчання, яка може правильно передбачити *y* з обчисленнями, заданими вхідними векторами *x*. Під час навчання алгоритм буде коригувати ваги системи відповідно до того, як вона наближається ближче до правильних прогнозів. Такий тип навчання може вирішувати або проблеми класифікації, або проблеми регресії залежно від набору даних.

Проблеми класифікації характеризуються відображенням вхідних векторів *x* на одне конкретне значення з скінченного набору дискретних категорій, наприклад, вхідні дані про атрибути пісні, відображені в лейбі компанії.

Проблеми регресії характеризуються відображенням вхідних векторів *x* до однієї або декількох безперервних змінних, наприклад вхідні дані: зріст футбольного гравеця, вага, відсоток жиру тіла та м’язів для передбачення швидкості бігу гравця, яка задається в м / с.

2.5.2 Навчання без учителя

У разі відсутності ручної розмітки даних, доводиться використовувати навчання без учителя. Оскільки немає відповідних цільових значень для відображення вхідних векторів, то натомість метою може бути кластеризація даних у групи подібних атрибутів або оцінити розподіл даних. Одним із таких прикладів є метод головних компонент, де багатовимірні дані проектуються на менший вимір для зменшення розмірності набору даних. Більшість генеративних моделей не мають навчання без учителя, однак деякі з них можна розглядати як напівавтоматичне навчання.

2.5.3 Напівавтоматичне навчання

Поєднуючи невеликий набір розмічених даних із більшим набором даних без міток, можна використовувати напівавтоматичне навчання. Напівавтоматичне навчання – це суміш двох концепцій: навчання без та з учителем, яке проявило себе успішним у підвищенні ефективності виконання певних завдань. Перевагами напівавтоматичного навчання є те, що застосовується мінімальна обчислювальна потужність, яка необхідна для навчання з вчителем зводиться до мінімуму та частина яка відповідає за навчання без учителя підтримується реальними розміченними даними, які дають певні вказівки щодо навчання.

У даній магістерській дисертаціє здебільшого буде досліджуватись напівавтоматичне навчання та навчання без учителя.

2.6 Штучні нейронні мережі

Нейронна мережа - це спроба імітувати структурне з’єднання один з одним у мозку людини. Нейрони сполучаються один з одним за допомогою ваг (синапсів), а вихід визначається тим, які нейрони актиквувались, а які ні.

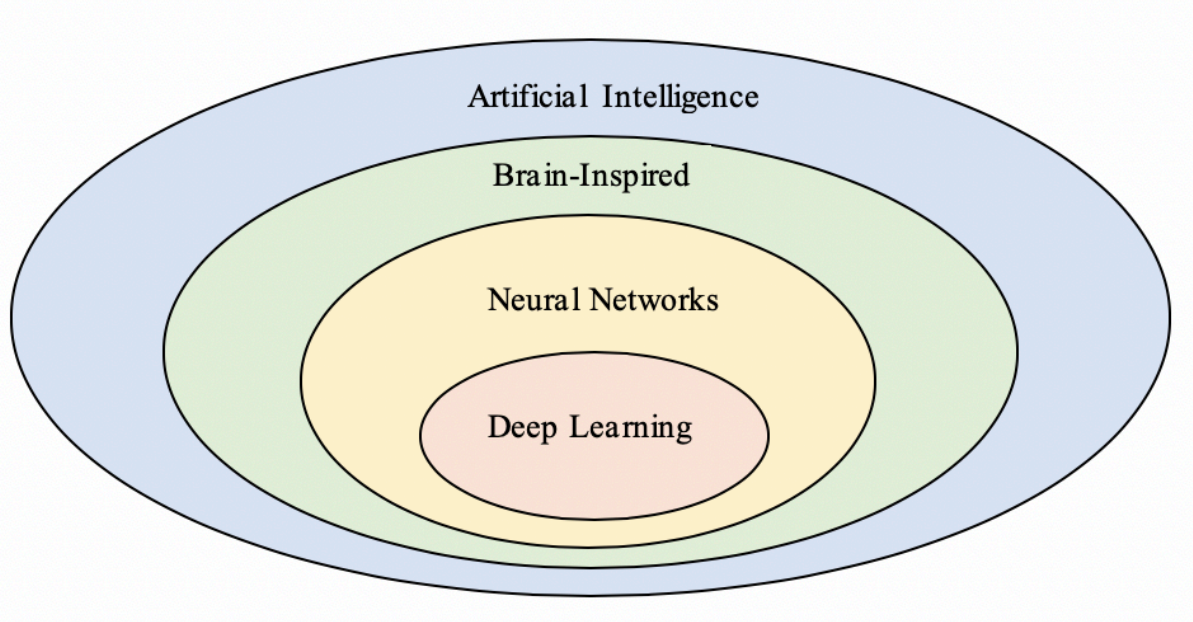

Рисунок 2.2 –Глибоке навчання як підрозділ штучний нейронних мереж.

Існує багато видів штучних нейронних мереж, однак вони розбиваються на різні категорії залежно від потоку даних через мережу. Нейронна мережа прямого поширення, відрізняється прямим потоком даних, хоча також існують циклічні та двонаправлені нейронні мережі. Ми дослідимо всіх три і з'ясуємо яку роль вони відіграють роль в розглянутих тут моделях.

2.6.1 Повнозв'язні мережі

Найпростіша нейронна мережа - це перцептрон, яку Бішоп [21] описує як двокласову модель з трьома компонентами: вхідний шар, вихідний шар і з'єднання між двома шарами. Математично ми можемо описати вихід як $y_i = f_i(\omega_i^T x)$, де y_i – і-й елемент вектора у роміром n, x - вхідний вектор розміру з розміром m, ω_i – відповідно вектор ваг, а f - нелінійна функція активації, задана таким чином:

$$f_i(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0 \end{cases} \tag{2.6}$$

Вибір відображення значень функції активації f розроблений шляхом представлення два класи. Розглянемо наш цільовий вектор t розміром m, що містить два різних класи. Перший клас представлений як 1, тоді як другий клас представлений -1. Прикладом такого відображення може бути $t = [1, -1, 1, 1]$, де ми маємо друге значення як другий клас, а решта – перший клас. Мета алгоритму - знайти правильні ваги *w* такі, що вони можуть правильно передбачити *y*. Ми можемо це зробити за допомогою ітераційного оновленняя ваги відповідно до продуктивності виходу. Якщо вихід є правильним ми зберігаємо ваги, а якщо вихід неправильний, ми змінюємо їх трохи і пробуймо ще раз. Процес оновлення перетворюється в задачу оптимізацією і існує багато різних підходів для її вирішення. Більшість реалізацій включає в себе функцію помилок, яку слід мінімізувати, тобто зниження значення помилки повинно вказувати на кращу продуктивність мережі. Від функції активації, ми хочемо, щоб кожен зразок x_iі мав $f(x_i) > 0$, якщо це належить першому класу і f (x_i) <0, якщо це належить другому класу. Оскільки, t ∈ {−1, + 1}, ми можемо звузити задачу до $t_i f(x_i) > 0$ для всіх x_i. Тому ми можемо написати своє функція помилки як

$$E_p(\omega) = -\sum_{n\epsilon M} t_n \omega^T x_i, \tag{2.7}$$

де M - сукупність усіх неправильно класифікованих зразків. Ця функція називається критерієм перцептона. Оскільки функція помилки не може бути диференційованою, нам доведеться використовувати стохастичний градієнтний спуск (SGD). Цей підхід ітераційно оновлює ваги до мініму $E_p(\omega)$ відповідно до:

$$\Delta\omega = -\eta \nabla E(\omega), \tag{2.8}$$

де $0 < \eta \leq 1$ – ступінь навчання системи. Теорема конвергенції гарантує, що якщо існує гіперплощина, то це може розділити два класи,

алгоритм буде збігатись в межах невеликої кількісті кроків. Незважаючи на всі будівельні блоки на місці є ще одна відсутня деталь. Якщо всі значення вхідного вектора дорівнюють нулю, ваги не матимуть значення, оскільки вихід завжди буде нульовим. Цю проблему можна вирішити додавши зміщення до вхідного вектора. Замість старої довжини *x* яка була n, тепер довжина n + 1 де останній елемент x є theta != 0. Зсув еквівалентний перетину у лінійнуй регресії. Перцептрон був створений Франком Розенблатом і його дослідження алгоритма перцептрона було опубліковано в 1962 році в книзі “Принципи нейродинаміки: перцептрон та теорія мозкових механізмів”.

Узагальнений алгорит перцептрона буде виглядати наступним чином:

1. Ініціалізувати всі ваги випадковим чином.
2. Обчислити вихідний вектор $y = f(w^T x)$
3. Оновити ваги відповідно до методу стохастичного градієнтного спуску $\Delta\omega = -\eta\nabla E(\omega)$,
4. Повторіть кроки 2-3 до зближення або після n кількості повторень.

Перцептрон є дуже важливою віхою в галузі теорії розпізнавання образів у машинному навчанні. Однак Марвін Мінський зазначав, що існували деякі великі обмеження в архітектурі. Він підняв той факт, що відома проблема XOR (рисунок. 2.3) не могла вирішуватись оригінальним алгоритмом перцептрона.

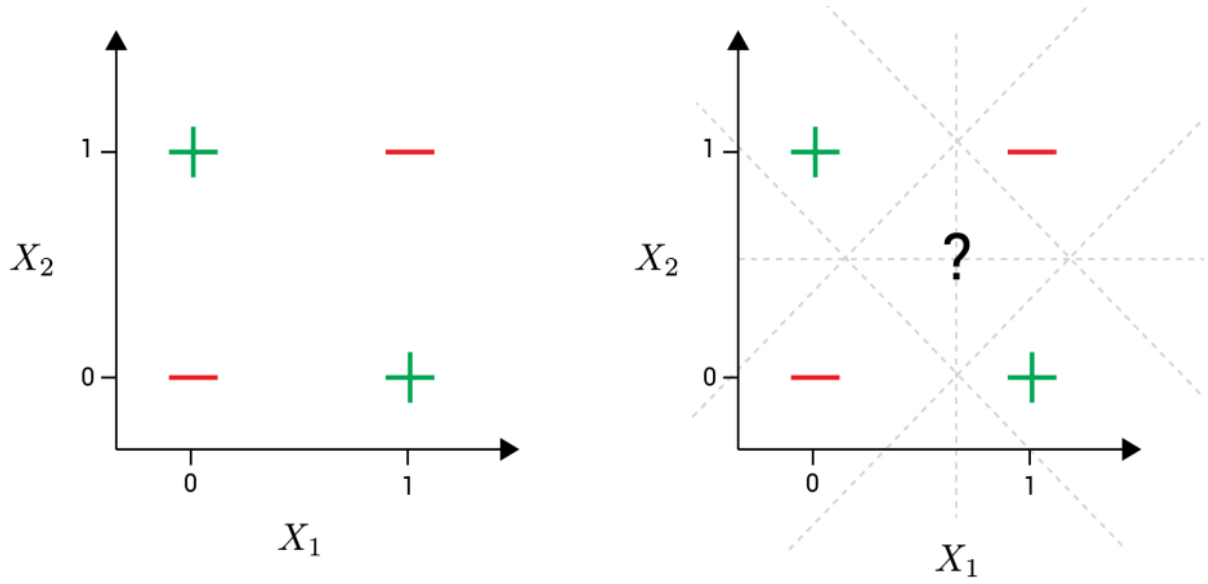

Рисунок 2.3 – Проблема XOR . У нас двокласна проблема де дані не є лінійно відокремленими. Зображення адаптовано з Баттіні [22].

Примітка: функція помилки також відома як цільова функція, значення значення або функція втрати. В майбутньому ми позначимо це як $\mathcal{L}$.

Усі розв'язки, отримані з алгоритму персептрона, є лінійними комбінонації одновимірної гіперплощини. Питання виникає, коли двокласова проблема не є лінійно відокремлюваною, як це можна побачити в XOR (рисунок 2.4). Це спричинило сильне зниження рівня довіри до дослідження нейронної мережеві та зупинило розвиток нейронних мереж поки не було знайдено рішення через кілька років.

Люди зрозуміли, що додавши шар до алгоритму перцептрона і перетворивши його на багатошаровий перцептрон, з'явився нова величина. Цей новий шар називається прихованим шаром і відіграє головну роль у галузі штучних нейронних мереж. При додаванні цього нового шару стало недостатньо алгоритму перцептрона для навчання і його треба модифікувати.

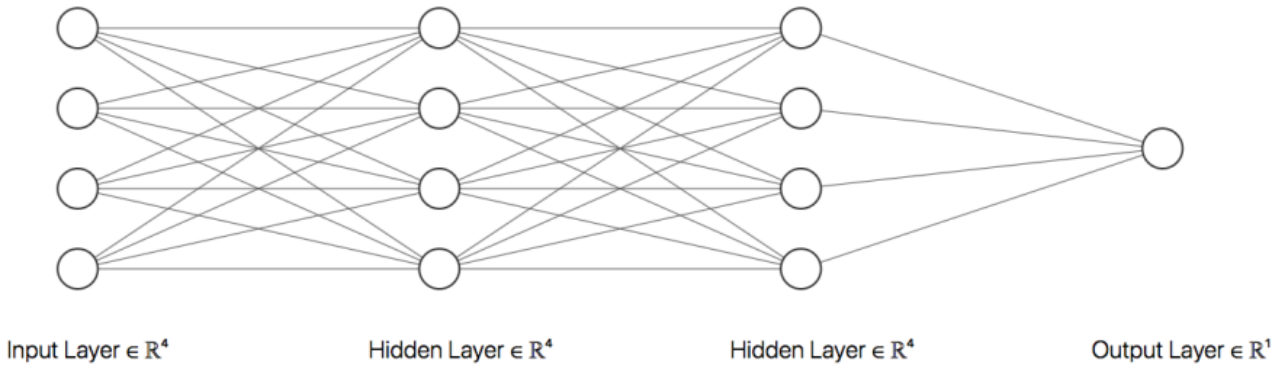

Рисунок 2.4 – Повнозв’язна нейронна мережа

Метод зворотнього поширення помилки є найважливішою компонентою глибокого навчання, оскільки це "навчальна" частина мережі. Коли збільшується кількість шарів, у нас виникає помилка залежно від попередніх шарів, і треба використовувати ланцюгове правило для отримання розв’язку. Однак, коли архітектура стає більше складною, ми також матимемо різний вид зворотнього поширення помилки. Згорнуті шари дадуть один вид поширення помилок із двонаправлених рекурентних нейронних мереж, матимуть інший підхід у розрахунках.

Існує багато різних способів комбінування шарів і вузлів для глибокого навчання. Попередній приклад MLP вважається повнозв’язною нейронною мережею. Для цього характерно те, що кожен вузол в одному шарі з'єднаний з кожним вузлом сусіднього шару.

MLP чудово підходять для проблем класифікації та регресії які мають табличні набори даних. Однак вона не працює так добре, коли дані мають інший вид, такий як дані пікселів із зображення. Однією з причин це тому, що сусідні пікселі на зображенні більше залежать один від одного порівняно з пікселями, які знаходяться далі. Тому не так вигідно з'єднувати вузли, які знаходяться далі.

Також в даних про зображення немає окремих ознак, які безпосередньо впливають на цільове значення порівняно з табличними даними, які часто мають чіткі властивості. Наприклад, вірогідність автомобільної аварії буде залежати від тиску в шинах, розміру шини, розміру автомобіля, людини за кермом тощо. Найбільш важливі ознаки будуть вилучені під час процедури

попередньої обробки даних. Тим часом зображення може бути end to end, , оскільки дані пікселів можуть подаватися безпосередньо в модель без попереднього вилучення ознак. Хоча вилучення ознак, безумовно, все ще можна використовувати як попередню обробку, це не завжди потрібено, у порівняні з деякими інших типами наборів даних.

Найбільшим недоліком пов'язаних нейронних мереж є те що є те, що вони не підходить для масштабування наборів даних, таких як повні зображення. Наприклад, CS231n [23] має таку структуру: бере RGB (3 кольорові канали) зображення, яке складається з 32 пікселі в ширину і 32 пікселі у висоту. Отже, розмір вхідного зображення 32x32x3 і вузол у першому прихованому шарі матиме вагу 3072. Це вже величезна кількість ваг для такого маленького малюнка. Як тільки зображення масштабується, ваги зростають експоненційно і стають некерованими дуже швидко. Найкращим підходом з використанням нейронних мереж до обробки та аналізу зображень є спеціально розроблені нейронні мережі для таких даних.

2.6.2 Згорткові нейронні мережі

Операція згортки вже багато років займає важливе значення в області розпізнавання зображень. Основна ідея - використовувати фільтр або ядро для проходження по зображенню та створення карти ознак на виході. Розглянемо на прикладі зображення 4x4, ми можемо використовувати фільтр 3x3 для проходження по зображенню, як проілюстровано на рисунку 2.6, і створити карту 2x2.

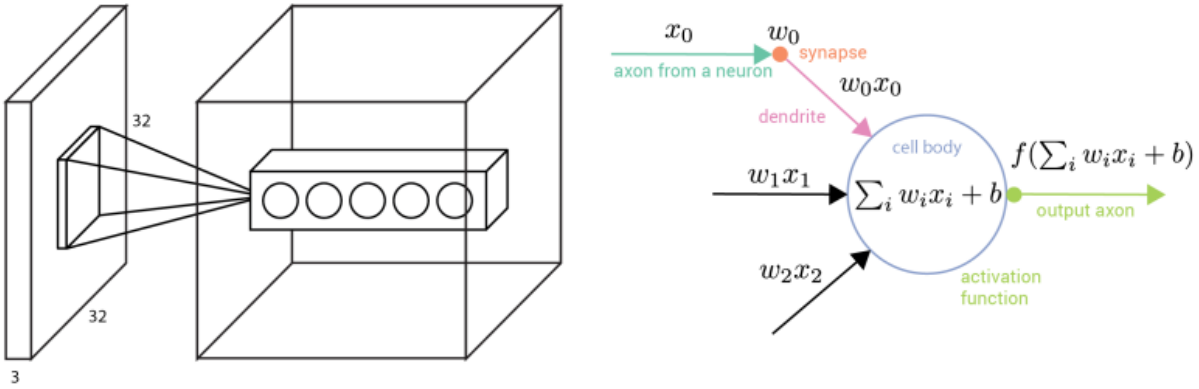

Рисунок 2.5 – Згорткова нейронна мережа.

Математично це можна розглядати так**:**

$$h_{i,j} = \sum_{k=1}^{m} \sum_{l=1}^{m} \omega_{k,l} x_{i+k-1,j+1} \quad (2{,}9)$$

де h - карта об’єднаних ознак, w - ядро згортки, x - вхідне зображення і m - ширина і висота ядра.

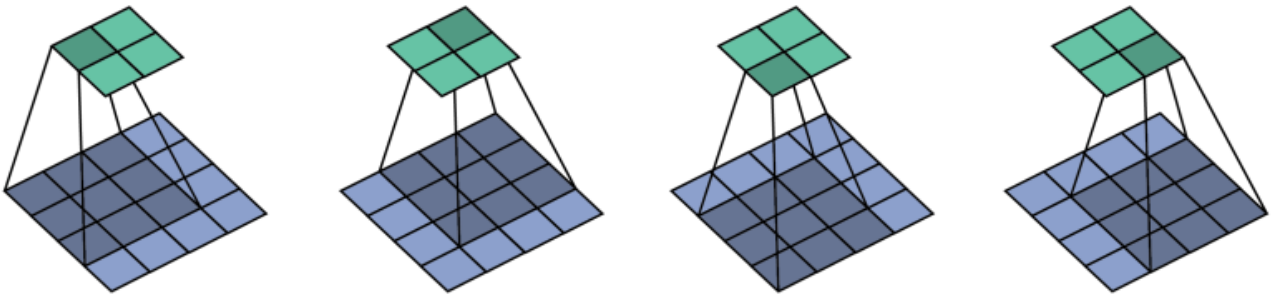

Рисунок 2.6 – Операція згортки . Фільтр розміром 3 згортається над зображенням 4x4 і створює карту розмірів 2x2, коли вона ковзає.

Перевага використання згорток полягає в тому, що вузли у шарі повині бути з’єднані з конкретною областю вузлів у сусідньому шарі. Це можна показати, продовживши приклад CS231n [23], рознлянутий у попередньому пункті та застосувати операцію згортки. Якщо у нас є фільтр розміром 5x5 який накладається на зображенні 32x32x3, кожен вузол має бути з’єднаний з 5x5x3 = 75 вузлами у наступному шарі. Це сильно зменшує кількість обчислень на відміну від повнозв’язної нейронної мережі. Для згорткових шарів існує три гіперпараметри, які нас цікавлять.

1. Згорткова нейронна мережа (CNN) часто має багато фільтрів, і

результати кожної картки ознак будуть складені в наступний шар. Різні фільтри додають вклад до активації вузлів, оскільки кожен фільтр може шукати різні речі на зображенні. Кількість фільтрів визначає глибину наступного шару.

2. Фільтри також можуть рухатися по-різному, залежно від того, що потребується. Кількість кроків, з яким рухається кожна згортка, є гіперпараметром та називається кроком. Крок часто позначає значення кількості кроків праворуч за згорткою, тобто по осі x. Є деякі випадки, коли можна скористатися віссю y, хоча це використовують не часто. Крок n означає, що фільтр рухається на n кроків, а основна CNN, показана на рисунку 2.6, має крок 1, проте є багато випадків, коли використовують крок 2 або навіть 3. Збільшення кроку зменшує розміри виводу, оскільки карта ознак матиме менше елементів.
3. Останній гіперпараметр - це техніка, яка може керувати просторовий розмір вихідної карти ознак. Ця процедура передбачає додавання нулів по краях вхідної карти ознак. Оскільки згортки створюють карту ознак менших розмірів в порівнянні з вхідним, нульовим доповненням можна зберегти розмір у всій мережі. Це передбачає розширення вхідного простору нулями або навколо межі зображення або іноді дробово між ними.

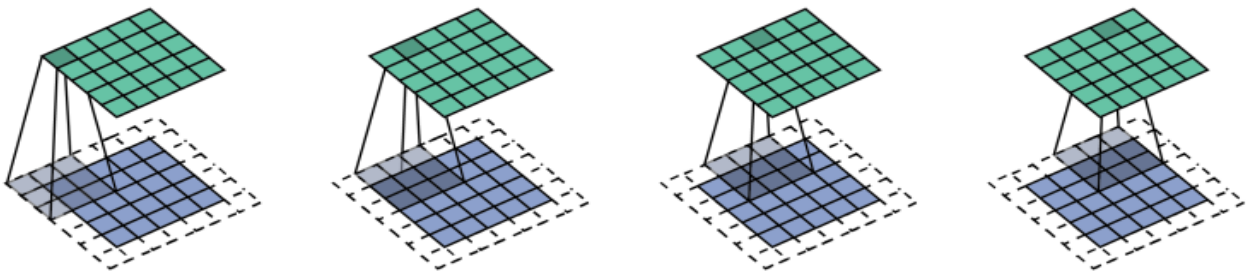

Рисунок 2.7 – Транспонована згортка: Фільтр розміром 3, що проходить зображення 5x5 з одиничною рамкою з нульовим доповненням, щоб отримати карту розміру 4

Хоча CNN розроблені спеціально для зображень, ефективність

алгоритму запровадила спроби застосування CNN на наборах даних які можуть бути представлені аналогічно. Мовлення та аудіо зокрема мають подібні локальні залежності та звукові хвилі можуть бути представлені алогічним чином, як значення пікселів. Як наслідок цього, CNN також показали значні вдосконалення в області розпізнавання мовлення та покращення мовлення.

Є в математиці термін, який називається деконволюцією що по суті є функцією зворотньою до згортки, наприклад, $F_{deconvolve}\big(G_{convolve}(H)\big) = H$. Однак, у глибокому навчанні цей термін був неправомірно використаний. Шар деконволюції визначається як нанесення дрібно крокових згорток для створення карти ознак у тій же частині, що і згортковий шар. Він не повертає операції згортки попереднього шару, а лише змінює на протилежний розміри карти об’єктів, тобто покращує зображення. Останнім часом ми намагалися відійти від цього терміна [cnn-indepth] і замість цього використали транспоновані згортки як правильну деномінацію.

Деконволюцію можна легко зрозуміти, подивившись рисунок 2.7. Ми доповнюємо вхідні дані нулями, щоб фільтр міг ковзати по більшому вхідному простіру, таким чином створюючи більший простір ознак. Термін дрібно кроково стосуються доповнення поміж вхідних даних (мал. 2.8).

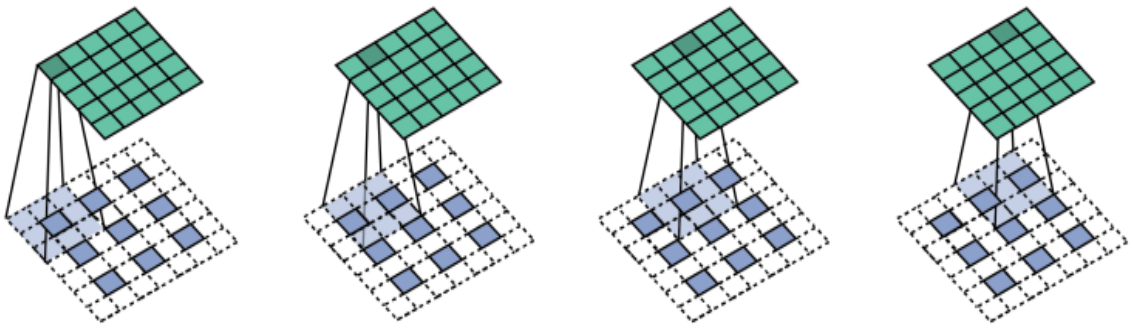

Рисунок 2.8 – Транспонована згортка : фільтр розміром 3, що перегортається 3x3 зображення з частково ступінчастою нульовою накладкою кроку 2 на скласти карту розмірності 5.

2.6.3 Функції активації

Є кілька різних функцій активації, які корисні для глибокого навчання. Метод зворотнього розповсюдження помилки матиме різні результати на виході в залежності від функції активації. Деякі з функцій, що відповідають моделям, що переглядаються в даній роботі наведені нижче.

Розглянемо найбільш поширені:

1. Softmax (Нормалізована експоненціальна функція)
2. Сигмоїд
3. ReLU
4. LeakyReLU
5. PReLU

Нормалізована експоненціальна функція є найпоширенішою функцією для останнього шар нейронної мережі, оскільки він нормалізує виходи в розподіл ймовірностей, що враховує K ймовірностей (заданий K вхідні точки). Він визначається як:

$$softmax(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}, \tag{2.10}$$

де $I \in R_k$

Нормалізована експоненціальна функція використовується головним чином у завданнях класифікації, оскільки ми можемо знайти максимум функції softmax як найбільш вірогідний вихідний клас для вхідних даних.

Сигмоїдна функція, мабуть, є найпоширенішою функцією активації, яка були використані в ML. Вона визначається як:

$$f(x) = \frac{1}{1+e^{x}}, \tag{2.11}$$

де x - вхід. З форми графіка (рисунок 2.9) бачимо градієнт гладкий для великих абсолютних значень, в той час як дуже крутий для низьких абсолютних значень. Якщо дані не є відповідними, ми можемо мати випадок вибуху чи зникнення градієнтів. Сигмоїдна функція часто позначається як σ (x).

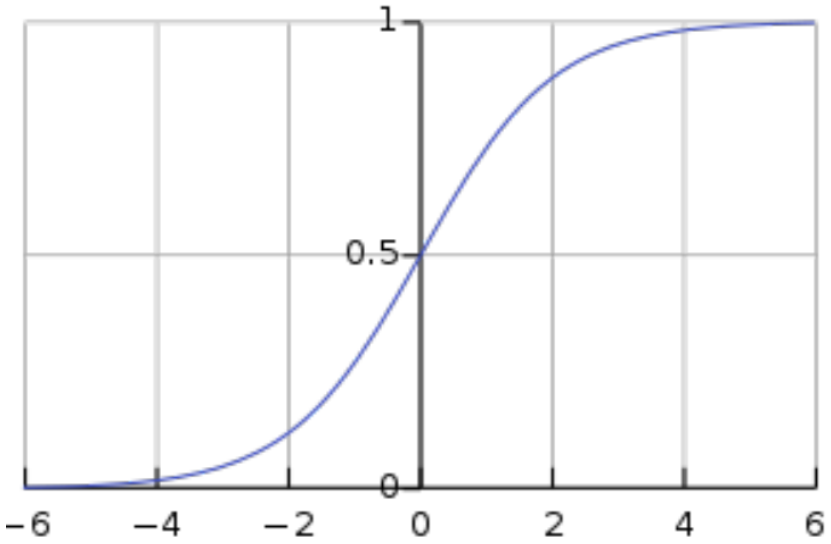

Рисунок 2.9 - Сигмоїдна функція.

Дуже популярною функцією активізації є випрямляч (ReLU) і він покав себе надзвичайно ефективно у багатьох моделях незважаючи на свою простоту. Вихід ReLU $f(a) \rightarrow output$ визначається як:

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}, \tag{2.12}$$

де x – вихідне значення ReLU гарантує, що градієнти не будуть вибухати, оскільки вони є постійни для значення $x \geq 0$. Однак, з ReLU виникає питання, коли вузли можуть помирати. Оскільки значення менше 0 не впливатимуть на нейронну мережу, може виникнути ситуація, коли вузол застряг у негативному просторі, що фактично означає, що вузол перестає сприяти, тобто вмирає.

Для боротьби з проблемою ReLU ми можемо використовувати LeakyReLU. Замість того, щоб повністю знищуючи вхід негативного значення, ми визначаємо активаційну функцію як:

$$f(x) = \begin{cases} x, & x \geq 0 \\ \alpha x, & x < 0 \end{cases}, \tag{2.13}$$

де у нас є додатковий параметр 0 <α <1, щоб переконатися, що негативні значення все ще сприяють, хоча і в незначній мірі. Стандартне значення α встановлюється 0,01.

Часто важко визначити, яким має бути параметр α. Один має використовувати еміпіричні докази попередніх моделей, щоб знайти, яке значенняє найбільш є підходящим, але це, як правило, ненадійно. Так само, як і багато речей у глибокому навчанні, ми можемо позбутися цієї відповідальності, відпустивши мережу вивчає цей параметр. He et al [24] представив параметрично випрямлені лінійні одиниці (формула 2.12)

Оскільки тепер ми можемо дізнатися цей параметр, для одношарової CNN зворотне розповсюдження визначається як:

$$\frac{\partial \mathcal{L}}{\partial a_i} = \sum_{y_i} \frac{\partial \mathcal{L}}{\partial f(y_i)} \frac{\partial f(y_i)}{\partial a_i}, \tag{2.14}$$

де $\frac{\partial \mathcal{L}}{\partial f(y_i)}$ – градієнт, що поширюється з більш глибокого шару. Береться сума по всій карті особливостей над усією картою фінкцій, тобто $i \in \{1, \dots, n^2\}$, де n – висота/ширина карти функцій. Градієнт функції активізації:

$$\frac{\partial f(y_i)}{\partial a_i} = f(x) = \begin{cases} 0, & y_i \geq 0 \\ y_i, & y_i < 0 \end{cases}, \tag{2.15}$$

Якщо у нас є кольорові канали, ми просто підсумовуємо ці карти зображень якдобре. Він та ін. [24] зазначають, що додавання PReLU є незначнимнаслідком складності в часі як вперед, так і назадрозповсюдження в CNN.

2.6.4 Рекурентні нейронні мережі

EHNet використовує двонаправлені LSTM(Long short-time memory), і тому важливо зрозуміти концепцію цієї архітектури.

І CNN, і MLP вважаються новими мережами, де потік архітектури йде в одному напрямку. Однак, дозволяючи циклічні зв'язки між вузлами ми досягаємо рекурентних нейромереж (RNN). Тоді як мережа прямого розповсюдження має виходи, які залежать лише від нашого поточного спостереження x_t, RNN мають свій прихований стан h_t залежний від попереднього прихованого стану h_{t-1}. Незважаючи на схожість з прихованими моделями Маркова, внутрішні стани RNN працюють поза припущеннями Маркова, що означає, що вони володіють здатністю брати до уваги віддалені зв'язки. Це означає що RNN матиме внутрішній стан, де послідовність входів впливає на майбутні результати. Для загального RNN (рисунок 2.10) Паскану, Міколов і Бенджо [26] визначають прихований стан h_t для часу t як:

$$h_t = f(h_{t-1}, x_t, \theta), \tag{2.16}$$

де x_t - вхід, f нелінійна перетворююча функція і θ - параметри. Для оригінальної RNN f визначається як:

$$f = W_{hh}\sigma(h_{t-1}) + W_{xh}x_t + b, \tag{2.17}$$

який має матрицю рекурентних вагів W_{hh}, зміщення b та матрицю вагів входів W_{xh} визначені як набір параметрів θ.

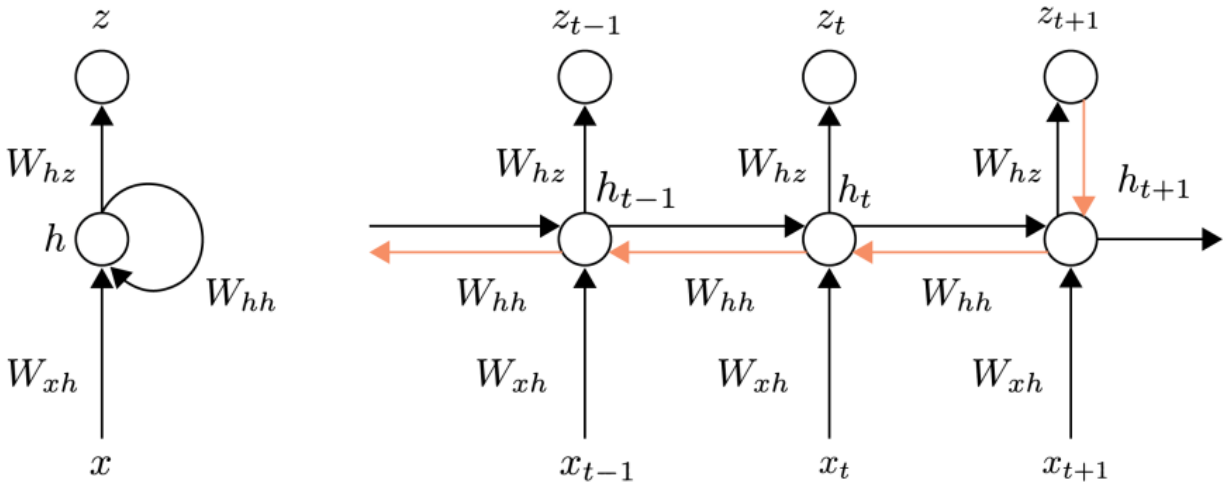

Рисунок 2.10 – Рекурентні нейронні мережі . Зліва рекурсивна модель, а справа відповідне розширена модель RNN у часовій послідовності. Зображення адаптоване від Chen[25].

Нехай функцією активації буде softmax:

$$z_t = softmax(W_{hz}h_t + b_z), \tag{2.18}$$

Ваги W_{hh} W_{xh} та W_{hz} тимчасово спільні у всій мережі. Існує кілька різних способів побудови RNN, хоча всі дотримуються однакової загальної структури оригінального графу обчислень (рисунок 2.10). Частина глибокого навчання для RNN - це велика кількість прихованих одиниць у прихованому стані. Однак ми спочатку розглянемо лише основні найпростіші RNN з цієї теорії.

Існує великий недолік у оригінальних RNN, коли послідовні дані дуже великі. Ми можемо спостерігати це, вивчаючи зворотне розповсюдження. Нехай $\mathcal{L}$ позначає цільову функцію, а $\mathcal{L}_t$ позначає вихід у часовий крок t. З цього ми можемо отримати:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{1 \le t \le T} \frac{\partial \mathcal{L}_t}{\partial \theta}, \tag{2.19}$$

Тепер, якщо ми розширюємо це для кожного кроку часу, ми отримуємо:

$$\frac{\partial \mathcal{L}_t}{\partial \theta} = \sum_{1 \le k \le t} \frac{\partial \mathcal{L}_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial^+ h_k}{\partial \theta}, \tag{2.20}$$

Середній член суми є винуватцем зникнення або вибуху градієнту, через те, що він розраховується як добуток попереднього входу, тобто:

$$\frac{\partial h_t}{\partial h_k} = \prod_{t \ge 1 > k} \frac{\partial h_i}{\partial h_{i-1}}, \tag{2.21}$$

тому якщо входи менші за 1, ми отримаємо еволюцію градієнтів, які стають все менше і менше, поки вони не зникнуть або навпаки, коли входи більші за 1. На щастя, існує спосіб боротьби з цим використовуючи комірки довготривалої пам’яті.

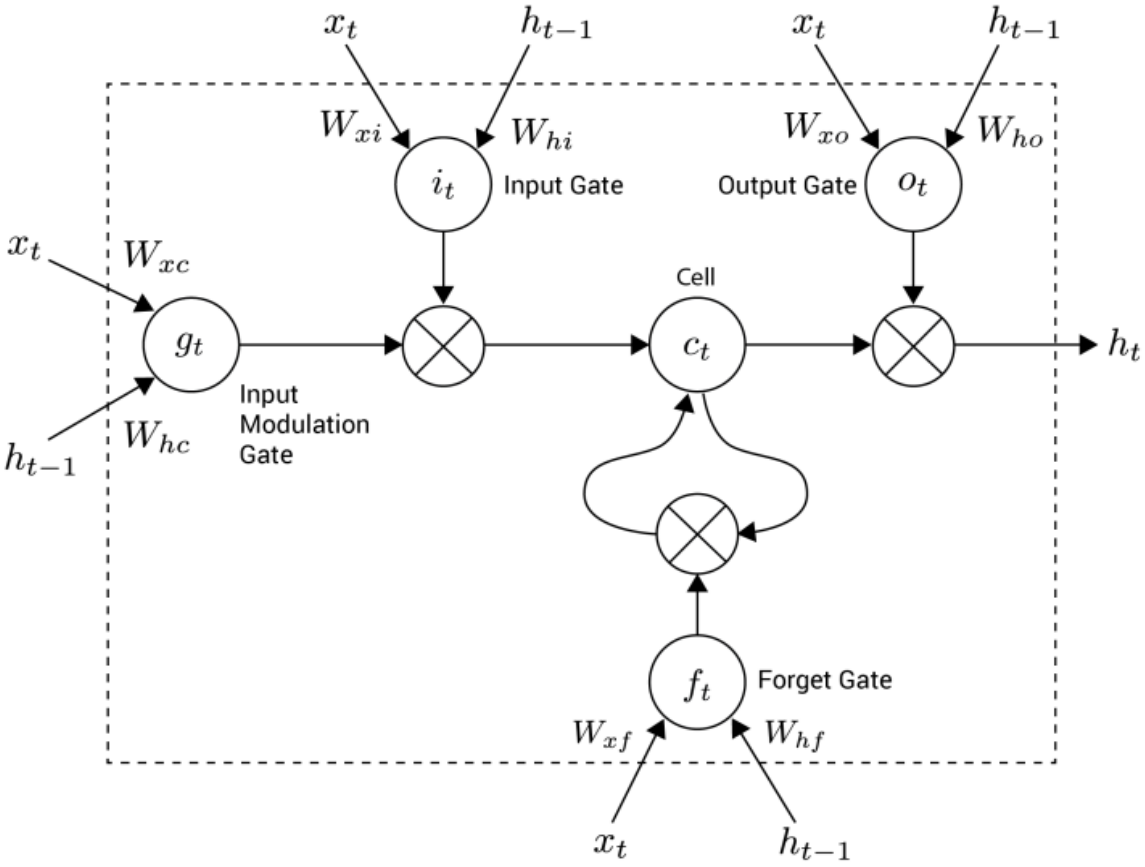

Рисунок 2.11 – Довга короткострокова пам'ять : За допомогою вентильних блоків градієнт може плавно протікати по системі.

Спосіб боротьби з градієнтами, що вибухають або зникають, - це додавання вентильних блоків до мережі. Визначення цих вентилів:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i), \quad (2.22)$$

$$g_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c), \quad (2.23)$$

$$c_t = f_t \odot c_{t_1} + i_t \odot g_t, \quad (2.24)$$

$$o_t = \sigma(W_{x0}x_t + W_{h0}h_{t-1} + b_0), \quad (2.25)$$

$$h_t = o_t \odot \tanh(c_t), z_t = \text{softma}x(W_{hz}h_t + b_z), \quad (2.26)$$

де f_t вказує вентиль забуття, i_t вхідний вентиль, o_t вихідний вентиль і g_t вхідний модуляційний вентиль. Вентилі забуття є головним героєм у вирішенні зникаючих градієнтів, оскільки це змушує проблематичну складову добутку мати елементи близькі до одного. Виведення цього доказу досить довгий процесс, це залишається як вправа для читача (або переконатися в цьому з пояснення Байєра [27]).

Ця специфічна архітектура доданих вентилів відома як довга короткочасна пам’ять (LSTM).

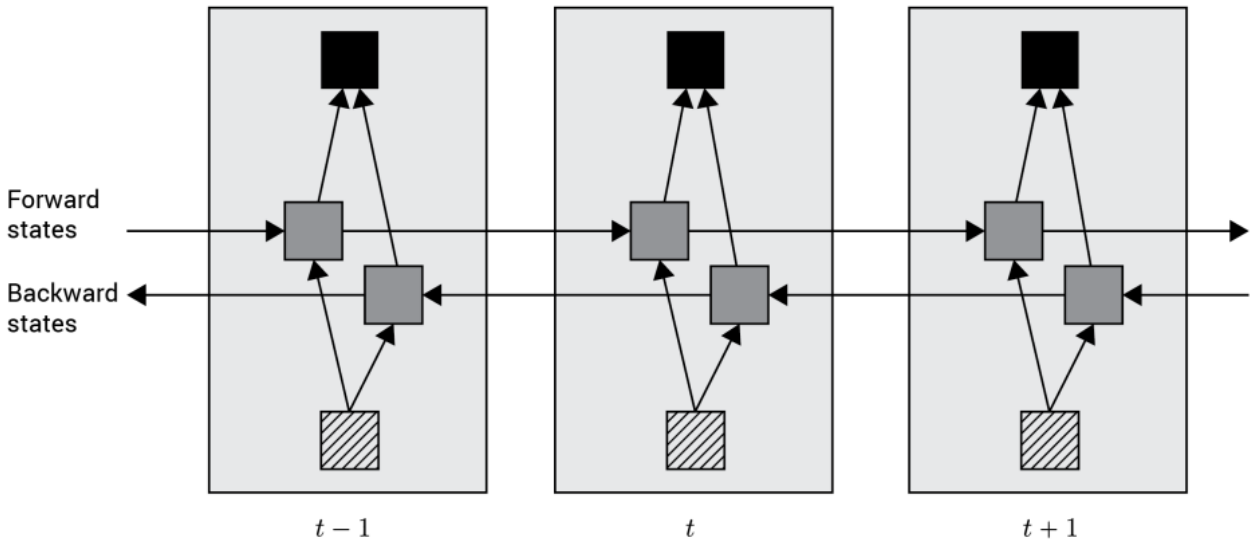

Рисунок 2.12 – Двонаправлені рекурентні нейронні мережі. Додавши другий шар нейронів, що знаходяться в іншому напрямку, ми можемо захопити всі вхідні дані за всі часові кроки. Зображення від Шустераі Паліваль [28].

На жаль, для оригінальних RNN та LSTM вони не можуть отримати доступ до майбутніх зразків за архітектурою. Тому, коли проблеми, де такі зразки доступні та цікаві для обробки, ми повинні використовувати двунаправлені RNN (BRNN). BRNN - це просто два RNN, складені один на

одного в прихованому шарі, але потік даних протилежний нижній. Розроблені Шустером і Палівалом [28] наприкінці 90-х рр. їм вдалося зафіксувати всі кроки часу одразу. На рисунку 2.12 ми можемо побачити потік даних для цього типу RNN.

2.6.5 ResNet

По мірі того, як шари стають все глибшими в нейронних мережах, ми отримуємо проблему деградації точності тренувань. Це зазначається у праці He et al. [29]. Вони заявляють, що це деградація точності, несподівано, не обумовлена перенавчанням.

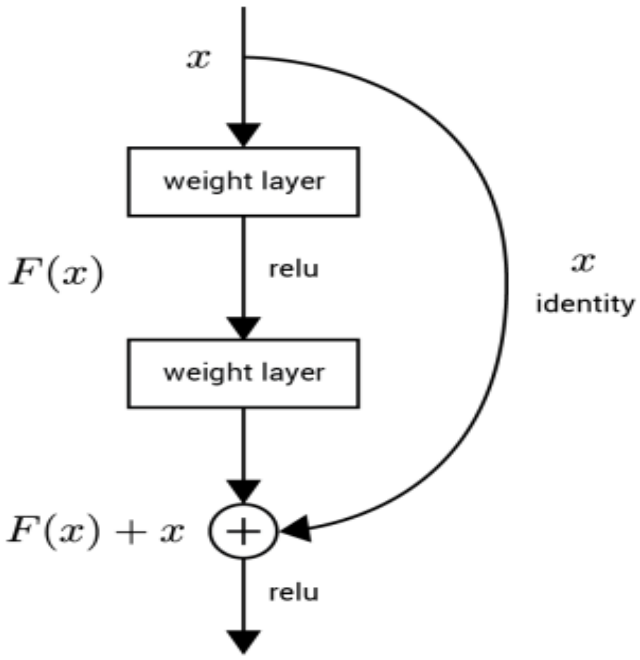

Рисунок 2.13 – Невеликий залишковий блок із прямим відображенням x. Зображення адаптовано з He et al. [29].

Вони показують, як глибший шар створює проблеми з дизайном на прикладі. Нехай неглибока мережа буде нашою орієнтиром і додамо шари прямого відображення для формування більш глибокої мережі. Інтуїтивно, помилка навчання повинна бути такою ж, оскільки додані шари не змінюють

будь-якого з розрахунків. Однак, як виявляється, ми не отримуємо порівняного або кращого рішення в більш глибокій мережі в порівнянні з початковою мережею.

З цього експерименту автори вбачають кількість шарів як головну причину у зниженні точності тренувань. Тому вони запропонували ResNets, глибока залишкова систему навчання.

Нехай $\mathcal{F}(x)$ - основне відображення, яке зазвичай відображає декілька шарів. В залишковому навчанні, ми натомість цим шарам наближається до залишкової функції $\mathcal{F}(x) \coloneqq \mathcal{H}(\mathrm{x}) - x$. Оригінальне відображення потім перетворюють на $\mathcal{F}(x) + x$. Вони досягають цього нового відображення додаючи з'єднання для швидкого доступу / обхідного з'єднання. Для ResNets ці обхідні з'єднання визначаються як:

$$y = \mathcal{F}(x, \{W_i\}) + W_s x, \tag{2.27}$$

де x і y - вихід і входи в залишковому блоці, коли F - це залишкове відображення, що навчається. На малюнку 2.13 маємо залишок $\mathcal{F} = W_2 f(W_1 x)$, де f позначає ReLU. W_s часто є тотожнім відображенням, оскільки це лише впливовий член у рівнянні, якщо F і x різних розмірів.

Дивовижна факт про ResNet полягає в тому, що концепція може бути застосована до всіх нейронних мереж з великими просторами зв'язку і добре сприйнялась спільнотою. Зовсім недавно Bonnier et al. [30] продемонстрував, як ResNet адаптація їх моделі глибокого підпису покращила продуктивність їх мережі, що надалі утверджує важливість фреймворку.

Ми побачимо, як це реалізовано і в SEGAN, і в WaveNet denoising.

2.6.6 Нормалізація батчів

У процесі глибокого навчання виникає більше проблем з навчанням, зі збільшенням кількості шарів. Оскільки для входу кожного шару змінюється розподіл після кожної зміни параметрів попереднього шару, що призводить до внутрішнього коваріаційного зсуву. Це явище може бути нівельоване задопомогою нормалізації даних. Іоффе та Сегеді[31] ввів новий спосіб нормалізації там, де нормалізуються кожен тренувальний міні-батч замість того, щоб нормалізувати весь набір даних. Перевага полягає у підвищенні продуктивності і тим самим дозволяє навчанню бути більш гнучким щодо ініціалізації параметрів та вибору швидкості навчання. Однак цей метод можна було ще вдосконалити, оскільки є ще невеликі проблеми з нормалізацією на міні-батчі.

Коли ми нормалізуємося на міні-батчі, ми все ще маємо проблему залежності коли вхідний зразок x сильно залежить від кількох інших зразків з того ж міні-батчу. Щоб виправити це, Salimans та ін. [32]вводять віртуальну нормалізацію батчу, де спочатку ми маємо опірний набір прикладів, витягнутих із даних. Коли ми пізніше робимо нормалізацію, ми не нормалізуємо вхідний зразок x на зразках з того ж міні-батчу, а замість цього на поєднанні опорного батчу та вхідного зразка. Проблема тут полягає в тому, що віртуальна нормалізація потребує два міні-батчі для кожного прямого проходження, що може вважатися обчислювально витратно.

SEGAN та WaveNet, обидва використовують або звичайну або віртуальну нормалізацію.

2.7Генеративні моделі

Генеративні моделі за своєю суттю - це максимум напівавтоматичне навчання, оскільки значна частина даних часто не маркується. Графік на рисунку 2.14 - це спосіб, яким Гудфеллоу [33] описує таксономію генеративних моделей. Розділ відбувається, коли є невідомий розподіл (щільність) або явна щільність. Як ми бачимо, навіть якщо GANsi WaveNet в одній генеративній родині, вони все ще дуже різні відносно внутрішніх властивостей моделей.

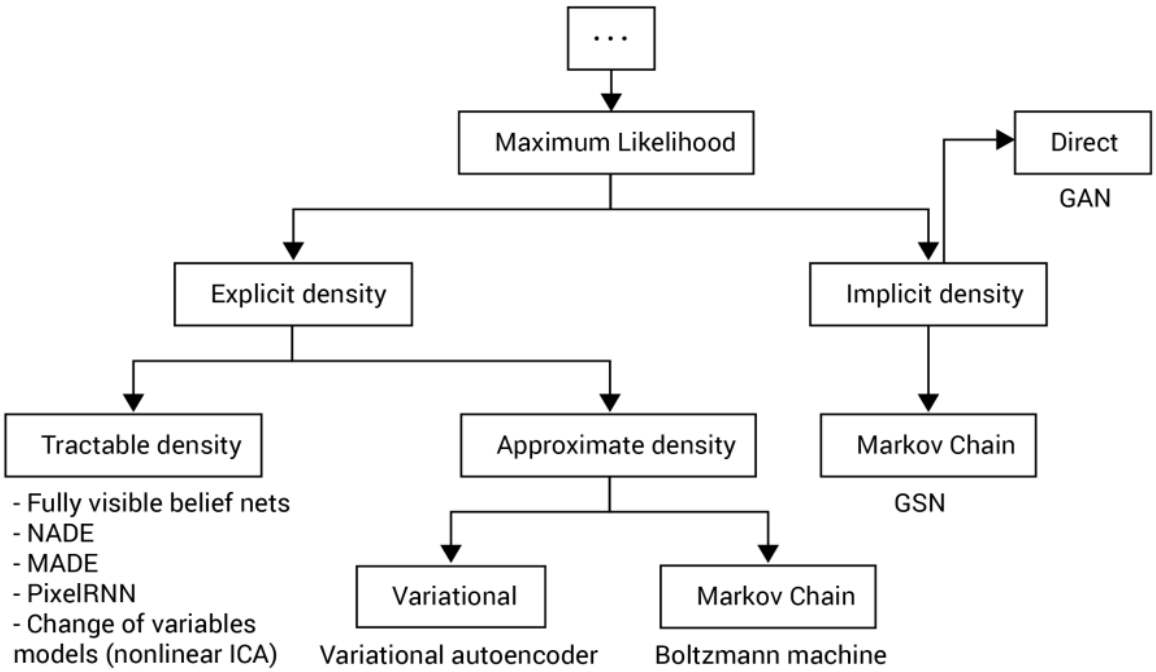

Рисунок 2.14 - Таксономія генеративних моделей. Зображення адаптовано від Goodfellow [33].

2.7.1 Gated PixelCNN

В оригінальній статті про PixelRNN та PixelCNN від Oord, Kalchbrenner і Kavukcuoglu [34] вони вводять ідею моделювання розподілу пікселів за

допомогою або двовимірних LSTM, або CNN. Хоча архітектура RNN давала набагато кращі результати, CNN виступили набагато швидше у навчанні. На сучасному ринку ми зіткнутися з такими роздільними здатностями, як 4K і навіть 8K, що призводять до невпевності, у можливості навчання для PixelRNN. З цієї причини вони прагнули покращити архітектуру PixelCNN, оскільки обчислювальна швидкість стала суттєво більш значущою.

Почнемо з визначення спільної ймовірності над зображенням x як:

$$p(x) = \prod_{i=1}^{n^2} p(x_i | x_1, \dots, x_{i-1}), \quad (2.28)$$

де n - розмір зображення. Порядок пікселів у растровому форматі сканування, тобто ми проходимо рядок за рядком і вивчаємо кожен піксель окремо. Це означає, що піксель буде залежати від кожного пікселя вище та зліва від нього. Вони справляються з цією умовністю, застосовуючи маску на дані пікселів, переконуючись, що пікселі не видно праворуч і нижче поточного пікселя.

RGB-кольори також залежать один від одного, коли B залежить від (R, G), а G залежить від R. Це досягається шляхом поділу карти об’єктів на трійки на кожному шарі. Вхідне зображення NxNx3 та вихідне прогнозування NxNx3x256 - це розповсюджені розміри для PixelCNN.

У новій статті Oord et al. [35] вони стверджують, що одна основна перевага архітектури LSTM - це їхні пропускні блоки, що обробляють складні взаємодії, і що "повторювані з'єднання дозволяють кожному шару в мережі мати доступу до всіх сусідів попередніх пікселів, в той час як область сусідів, доступних PixelCNN, зростає лінійно з глибиною згорткового стека". PixelCNNs може виправити другу частину, просто додаючи більше шарів, але першу частину неможливо уникнути з оригінальною архітектурою. Вони виправляють це шляхом введення Gated PixelCNN, замінивши ReLU міжз шарами згортки пропускною функцією активації:

$$y = \tanh\left(W_{k,f} * x\right) \odot \sigma\left(W_{k,g} * x\right), \tag{2.29}$$

де k - кількість шару, а $*$ - оператор згортки.

Недоліком оригінальної архітектури PixelCNN є сліпа пляма, що виникає у зоровому полі через наше маскування.

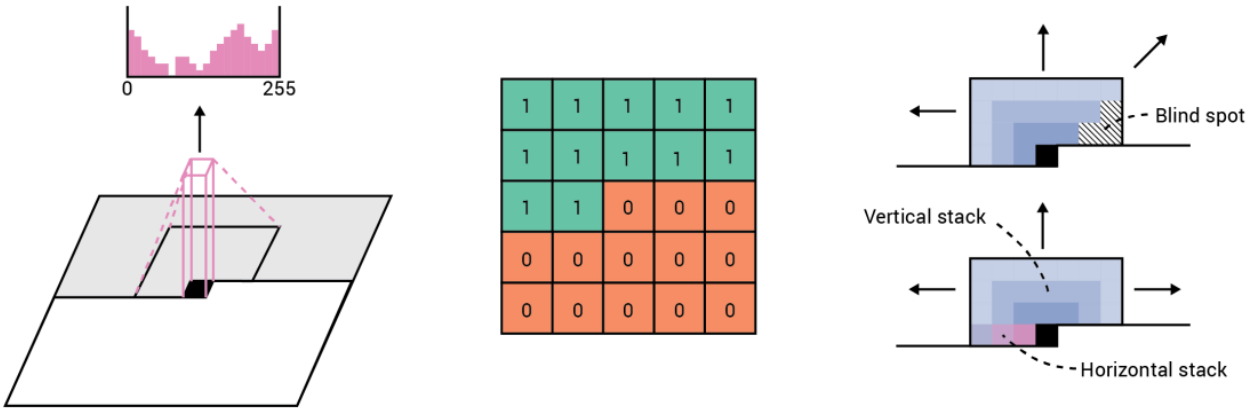

Рисунок 2.15 – Архітектура PixelCNN із згортковим маскуванням .Зліва направо: (1) Оригінальний PixelCNN. (2) Приклад того, як можна використовувати маску на фільтрі 5x5. (3) Вгору : Оригінальний PixelCNN має сліпе місце у полі зору. Знизу :Сліпу пляму видаляють двома згортковими стеками. Зображення адаптовано від van den Oord та ін. [35].

Розглянемо маску, яку зображено на рисунку 2.15, коли фільтр досягає самих правих частин зображення, ми зрештою пропустимо деякі пікселі, які ніколи не будуть враховані. Це призводить до того, що пікселі, передбачені за допомогою PixelCNN, не будуть залежати від усіх попередніх пікселів, як це потрібно (2.24). Ілюстрація (рисунок 2.15) показує, як це усувається за допомогою поєднання двох згорток, де одна обробляє всі рядки вище поточного пікселя, а інша частина обробляє всі пікселі в одному рядку до поточного пікселя. Потім, поєднавши два виходи, ми можемо вирішити проблему сліпої плями. Вони позначають ці дві згортки як вертикальну та горизонтальну. Горизонтальна згортка буде мати властивості маски, тоді як вертикальна не буде. Кожен шар горизонтальної згортки буде приймати вихід попереднього шару, а також вертикальної. Однак горизонтальна згортка не може бути з'єднаною з вертикальною у зворотному порядку, оскільки це

порушило б умовний розподіл.

2.7.2 WaveNet

Протягом 2016 року вплив CNN змусив людей переосмислити авторегресивні генеративні моделі. Як згадувалося раніше,CNN в основному розглядалися для розпізнавання зображень та були сфокусовані на комп'ютерному зорі. Однак, хоча обробка сигналу має аналогічні атрибути з теорія, що стоїть за перетворенням Фур'є і згортками, вона також має дуже подібну структуру даних, оскільки форми хвиль, можна моделювати так само як піксельні зображення. Маючи на увазі цю ідею, ван ден Оорд та ін. вирішили переробити їх архітектуру PixelCNN, для застосуванню до звукових хвиль і досягли великих успіхів. Вони назвали цей новий підхід WaveNet [36].

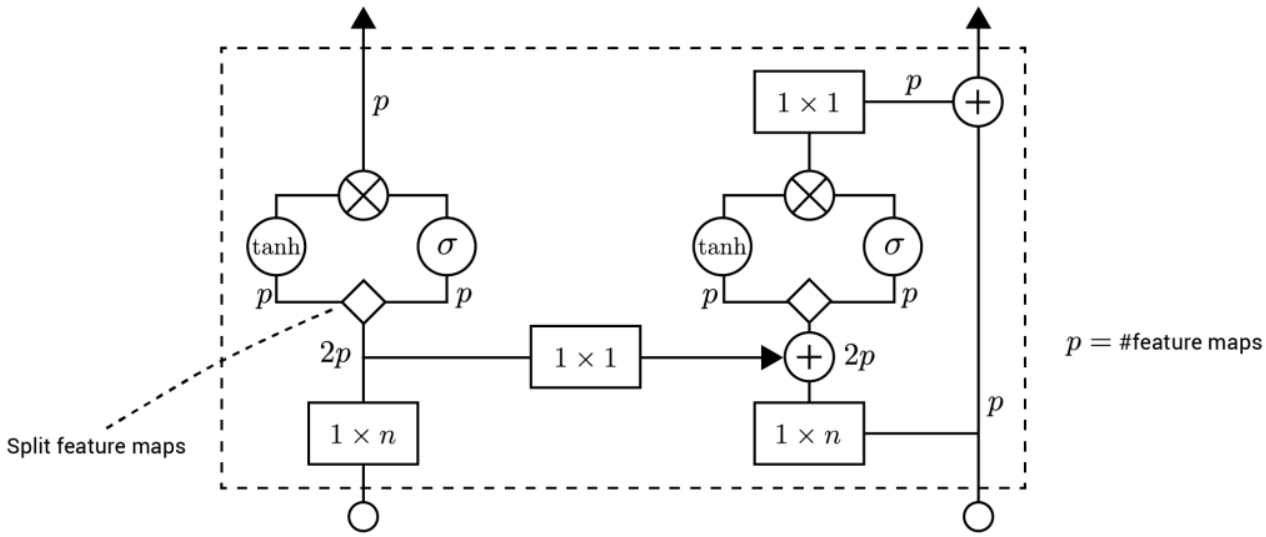

Рисунок 2.16 – Один шар в архітектурі Gated PixelCNN. Зображення від Ордом, Калчбреннером та Кавуккуоглу[34].

Аналогічно PixelCNN для пікселів зображення вони визначають умовну ймовірність хвилі x = $\{x_1, \dots, x_T\}$ як добуток умовних ймовірностей:

$$p(x) = \prod_{t=1}^{T} p(x_t | x_1, \dots, x_{t-1}), \tag{2.30}$$

Еквівалентний попередньому, зразок звуку x_t залежить від усіх часових кроків та модель може не порушувати порядок в якому моделюються дані. Ідея для реалізації цього полягає у введенні причинно-наслідкової згортки, аналогічно з масками в PixelCNN. Оскільки аудіо лише одновимірне, вони зазначають, що процедуру маскування можна впровадити шляхом зміщення виходу згортки на кілька кроків. Дані спостерігається послідовно, після прогнозування кожного зразка він буде поданий назад у систему, щоб передбачити наступний. Причинно-наслідкова структура згортки зображжена на рисунку 2.17.

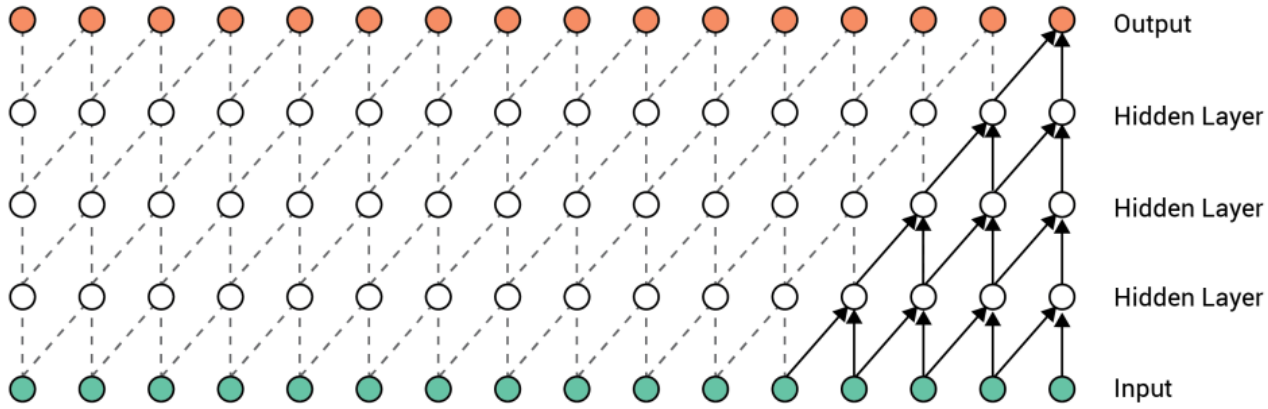

Рисунок 2.17 – Стек причинно-наслідкових згорткових шарів . Вхід буде лише залежати від попередніх записів. Зображення адаптовано з Van Den Ord та ін. [36].

Проблема причинних згорток полягає в тому, що вона вимагає або великої кількості шарів або більші фільтри, щоб мати необхідний розміру рецептивного поля, що може обмежити перевагу у швидкості тренування CNN. Van Den Ord та ін. [36] вводить розширені причинно-наслідкові згортки для їх архітектури, щоб залишити перевагу великого рецептивного поля, не збільшуючи обчислювальної вартості. Вони описують розширені згортки як згортки, де фільтр застосований до області, більша за її довжину, пропускаючи вхідні значення з певним кроком. Ми можемо бачити, як це еквівалентно розширенню оригінального фільтру нулями та схоже на субдескритизацію або крокову згортку. Різниця полягає в тому, що ми підтримуємо розмір вхідних даних у виході. Візуалізацію концепції можна побачити на рисунку 2.18.

У той час як вентильні функції активації (рівняння 2.25) ті ж, що і для PixelCNN, WaveNet має ще кілька доповнень до мережі, наприклад, softmax, модифікований для неочищеного аудіо та концепцію залишкових та обхідних з'єднання з ResNet [29].

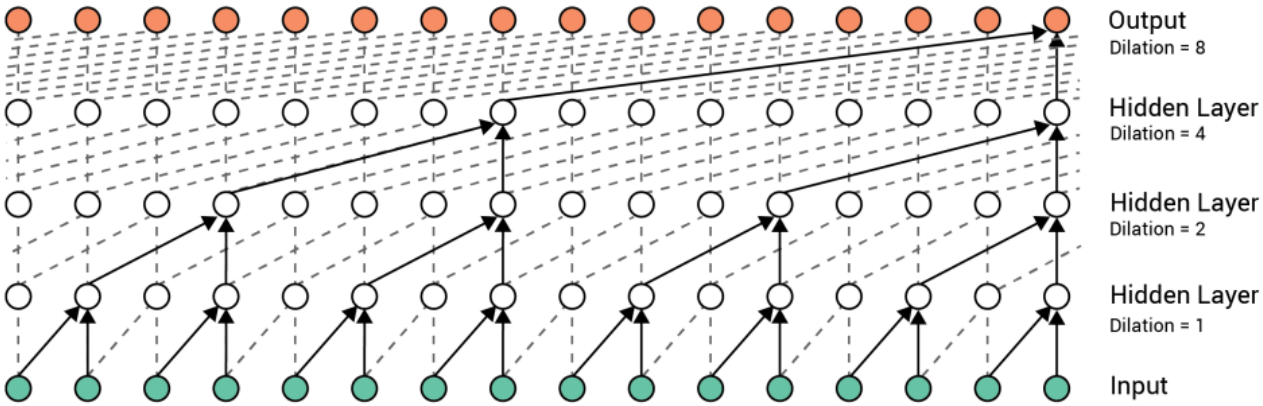

Рисунок 2.18 – Розширені причинно-наслідкові згортки. Ми збільшуємо рецептивне поле при збереженні обчислювальних переваг CNN. Зображення від van den Oord та ін. [36].

У статті про PixelCNN van den Oord та ін. показали, як розподіл softmax краще працює при умовних розподілаху порівнянні зі змішаними моделями. З іншого боку, через те, що необроблений звукозапис зазвичай зберігають в 16-бітових цілих значеннях на кожний часовий крок, кожен softmax шар має на виході 65536 ймовірностей на кожен крок часу для відображення всіх можливих значень. Щоб вирішити цю проблему, вони вирішили застосувати μ-закон нелінійного перетворення даних згідно з mu-законом, а потім привести його до більш розумних 256 значень:

$$f(x_t) = sign(x_t)\frac{ln(1+\mu|x_t|)}{ln(1+\mu)}, \tag{2.31}$$

Залишковий блок моделі можна побачити на рисункуу 2.19 і цей блок багато разів накладається у моделі.

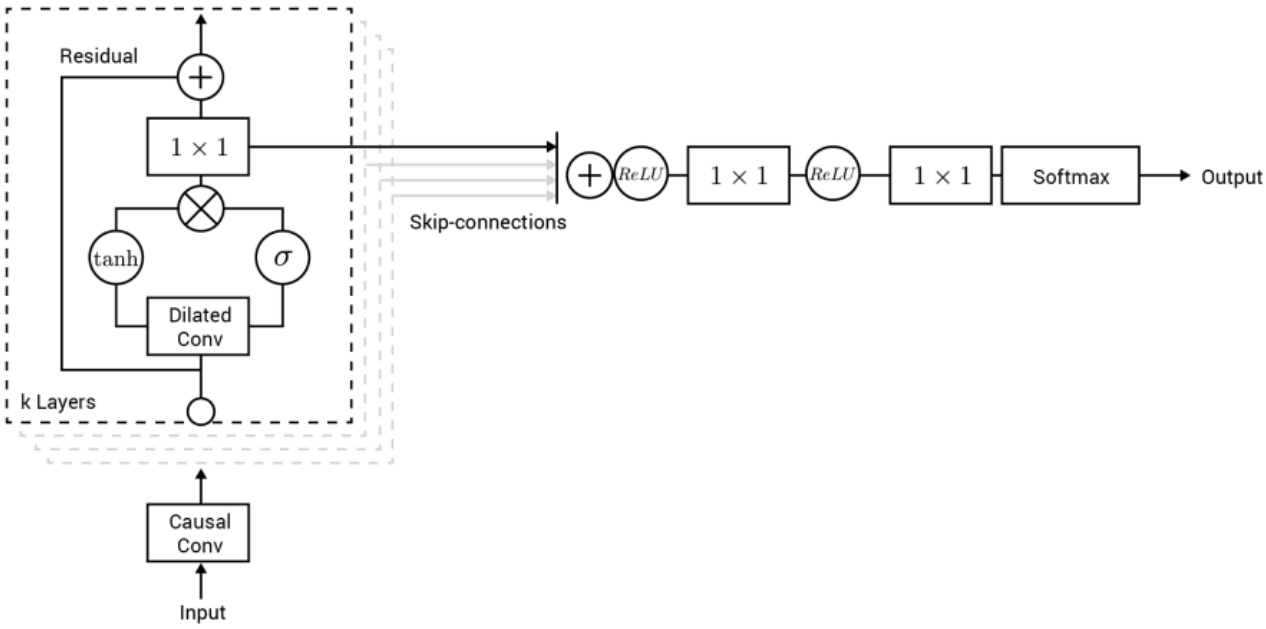

Рисунок 2.19 – Залишкові та пропускні з'єднання : залишковий блок та як він пов'язаний з усією архітектурою за допомогою пропускних з'єднань. Укладання показано пунктирними контурами. Зображення адаптовано з ван ден Oord та ін. [36].

2.7.3 Генеративні змагальні мережі

Перший зразок генеративно змагальної мережі був помічений у статті, опублікований Goodfellow et. в 2014 році. Вони представили концепцію нової генеративної моделі для боротьби з труднощами, які могли виникнути для раніше поширених моделей, коли мова йшла про розрахунки, наприклад, MLE. Як і в реальному світі, основна ідея GAN - цещо конкуренція породжує успіх.

Замість загальної ідеї підготовки єдиної мережі, GAN запровадили концепцію навчання двох мереж одночасно: одна - класифікатор інша - генератор. Класифікатор - це наша суддя, коли генератор - його противник. Генератор буде створювати підроблені дані, які намагатимуться обдурити класифікатор. Goodfellow та ін. al [2014] наводять аналогію фальсифікатора, який намагається обманути поліцію. Вони називають цей підхід змагальними

мережами.

Дивлячись на їх статтю, припустимо, у нас є два MLP, як відповідні змагальні мережі. Дискримінатор спробує вивчити розподіл генератора p_g над даними x. Вони визначають розподіл змінної шумового входу $p_z(z)$ і представляють відображення в просторі даних як $G\left(z;\ \theta_g\right)$, де G - MLP для генератора з параметрами θ_g. MLP дискримінатора потім визначається як $D(x;\ \theta_d)$ і виводиться скалярне значення, що позначає ймовірність того, що x походить від реальних даних. Для D ми максимізуємо ймовірність призначення правильного мітки щодо того, звідки зразок, а для G ми мінімізуємо вірогідність того, що D позначає дані генератора як підроблені, які ми визначаємо як $\log\left(1 - D\left(G(z)\right)\right)$. Функція втрат (яку вони позначають як функцію V) всієї змагальної мережі стає комбінацією двох цілей:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[log(D(x))] + \mathbb{E}_{z\sim p_z(z)}\big[log(1 - -log(D\left(G(z)\right)))\big] \quad (2.32)$$

Goodfellow [37] зауважує, що ця постановка minmax також може розглядатися як гра з нульовою суммою. Хоча функції втрат в загальному випадку є задачею оптимізації, де ми шукаємо локальний мінімум, для GAN замість цього шукають рівновагу Неша з теорії ігор. За сценарієм, де D і G грають свою роль ідеально, наша рівновага Неша матиме G (z) отримані з того ж розподілу, що і тренувальні дані, і D (x) = 0.5 для всіх x.

Хоча функція втрат може змінюватись для різних структур GAN, це поняття теорії ігор як і раніше є найважливішою ідеєю концепції. Найпоширенішими мережами, які використовуються в якості заміни MLP, є часто CNN та LSTM.

Хоча GANs на перший погляд здаються простими в ідеї, їх навчання набагато складніше, ніж для більшості стандартних архітектур глибокого навчання. Балансування двох мереж одночасно стає все більш важким,

водночас складність системи зростає і виникають деякі загальні проблеми:

- Дискримінатор занадто швидко отримує перевагу і змушує генератора перестати вчитись далі, оскільки кожен підроблений зразок маркується правильно та як наслідок його градієнти зменшуються.
- Гра коливається між переможцями та програвшими постійно і ‘мережі ніколи не сходяться.
- Генератор застряє з вибірками вибірок одного типу(режими), це називається колапсом режиму.

Є ще кілька труднощів для GAN, які важко врахувати, але три згадані - найбільш поширені. Оскільки концепція змагальних мереж все ще відносно нова порівняно з іншими методами глибокого навчання, оптимальної стратегії, як керувати цим балансуванням, ще не встановлено. Як вже згадувалось в попередніх розділах глибоке навчання не має жодних математичних доказів чому щось краще, з певними параметрами і реалізацією. Як сказано, є лише емпіричні дані, які можна використовувати як певні вказівки щодо майбутніх підходів. Тим не менше, поза межами звичних методів, таких як рання зупинка навчання, є деякі ідеї, які можуть покращити продуктивність GAN, якщо вони не сходяться:

- Ми оновлюємо дискримінатор частіше, ніж генератор або навпаки, щоб уникнути відрив жодної мережі. Деякі [37] проти цього методу та припускають, що одночасне оновлення є найбільш вигідним підходом.
- Ми змушуємо генератор генерувати підроблені зразки за межами їх поточного режиму одразу, щоб уникнути колапсу режиму.
- Змінюємо функцію втрат, щоб включити більш точний та детальне представлення даних.

SEGAN бореться з проблемами, використовуючи умовний GAN, що використовує теорію методу найменших квадратів (LSGAN). Ми розглянемо ці два поняття далі.

Функція втрат регулярних GAN виглядає як функція сигмоїдної перехресної ентропії (2.2) і виявилася проблематичною під час навчання на конкретних даних. X. Мао та ін. [38] стверджує, що проблеми виникають через те, що оцінка підроблених зразків, що неправильно класифікуються, не враховує відстань до межі класифікації. Це означає, що зразок, далекий від реальних даних, не створить суттєвої помилки для дискримінатора, навіть якщо відстань може бути дуже великою. У свою чергу, це ще один випадок згасаючих градієнтів і суворо обмежує навчання змагальної мережі.

Вони пропонують GAN з використанням методу найменших квадратів (LSGAN), щоб змусити генератор створювати зразки, ближчі до реальних даних, штрафуючи відстань так само. Функція втрат перетворюється на:

$$\min_D V_{LSGAN}(D) = \frac{1}{2}\mathbb{E}_{x\sim p_{data}(x)}[(D(x)-b)^2] + \\ + \frac{1}{2}\mathbb{E}_{z\sim p_z(z)}[(D(G(z))-a)^2] \quad (2.33)$$

$$\min_G V_{LSGAN}(G) = \frac{1}{2}\mathbb{E}_{z\sim p_z(z)}[(D(G(z)-c)^2] \quad (2.34)$$

де параметри a і b - мітки для двох класів – підроблені і реальні дані, коли c - це значення, яким G сподівається переконати D. Відповідно до цієї тези, ми хочемо, щоб G генерував зразки, які є максимально реальними, що означає, що c = b.

Проблема з GAN - це обробка перетворення «один до багатьох». Подібно до Мірза та ін. [39], оригінальні GAN мають відображення «один до одного» від входу до виходу, що обмежує використання в реальних умовах. Зображення часто можна описати кількома тегами, оскільки існують синоніми або споріднені слова в більшості мов, і це стає нерозумно врахувати їх у випадку один до одного. Рішення цього, яке вони запропоновано це умовний GAN, який додає залежність змагальної мережі до нової змінної шляхом додавання додаткового вхідного шару (як показано на малюнку 2.20).

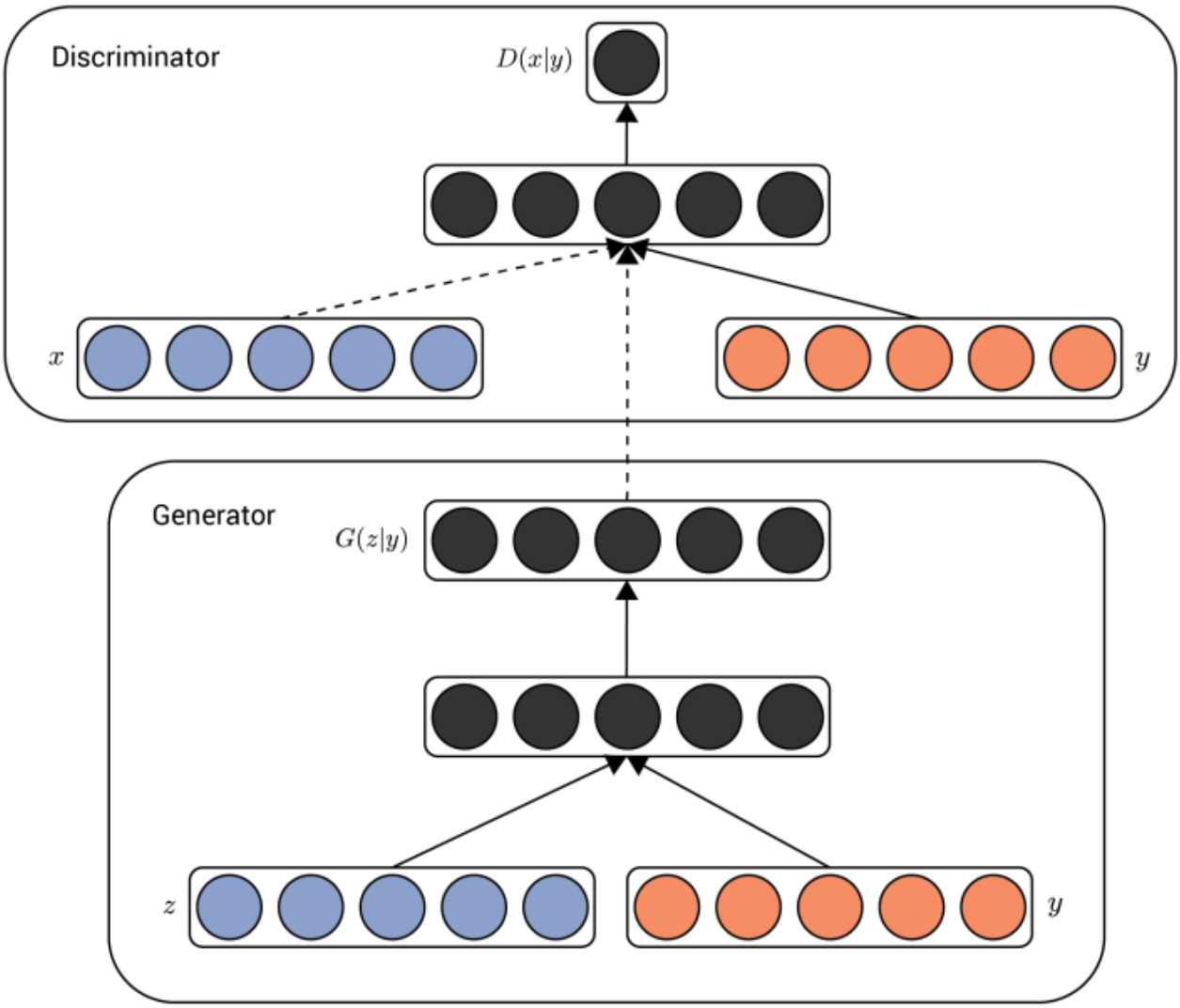

Рисунок 2.20 – Умовний GAN : Просто додаючи другий вхідний шару до мережі отримуємо умовний вихід D (x | y) і G (z | y).Зображення від Mirza та ін. [39]

Нехай у нас буде нове спостереження q від якого залежить змагальна мережа. Якщо обидва D і G мають на вході цей додатковий атрибут, ми отримуємо нову функцію втрат:

$$\min_G \max_D V(D,G) = \frac{1}{2}\mathbb{E}_{x,q}[logD(x,q)] + \mathbb{E}_{z,q}\left[\log\left(1 - logD\left(G(z,q)\right)\right)\right] \quad (2.35)$$

Ізола та ін. [40] перевіряють важливість обумовлення дискримінатора досліджуючи функцію втрат, де D не отримує q:

$$\min_G \max_D V(D,G) = \mathbb{E}_x[logD(x)] + \mathbb{E}_{z,q}\left[\log\left(1 - logD\left(G(z,q)\right)\right)\right] \quad (2.36)$$

Вони роблять висновок, що колапс режиму G набагато помітніше коли D сліпий до умовного внеску, що підтверджує що (2.36) це найкращий підхід до функції умовних втрат.

2.8 Огляд моделей

В цій дисертації ми аналізуємо результативність трьох різних глибоких мережах, досліджуємо модифікацію одної з цих мереж та використовує простий фільтр Вінера як орієнтир. Ці мережі були обрані тому, що вони мали чіткі відмінності та схожість зв'язки, з тим, як рухається галузь. SEGAN використовує GAN які в моді для розпізнавання зображень, WaveNet вже показав його можливості як найсучаснішого для створення генерування мови та позначає цікавий спосіб розширити рамки його застосування і нарешті, EHNet - це поєднання двох перевірених мереж(CNN та LSTM), які вважаються найсучаснішими у багатьох галузях.

2.8.1 SEGAN

Генеративна змагальна мережа для вдосконалення мови (SEGAN) розроблений Паскуалом, Бонафонте та Серра [5] і застосовує GAN архітектуру для покращення мовлення.

У SEGAN генераторна мережа G працює як остаточний інструмент для покращення мови. Мережа обумовлена додатковим входом, який є прихованим вектором, що представляє випадковий вектор шуму z. Вони

мають двійкове кодування з 1 і 0 для міток класу «справжній» і «підробка». Функція втрат - це модифікована версія умовного GAN, що замінює часто використовувані перехресні ентропії на функцію найменших квадратів у G:

$$\min_D V_{LSGAN}(D) = \frac{1}{2}\,\mathbb{E}_{x,\tilde{x}\sim p_{data}(x,\tilde{x})}[(D(x,\tilde{x})-1)^2] + \\ + \frac{1}{2}\,\mathbb{E}_{z\sim p_z(z),\tilde{x}\sim p_{data}(\tilde{x})}[(D(G(z,\tilde{x}),\tilde{x}))^2] \quad (2.37)$$

$$\min_D V_{LSGAN}(D) = \frac{1}{2}\mathbb{E}_{z\sim p_z(z),\tilde{x}\sim p_{data}(\tilde{x})}[(D(G(z,\tilde{x}),\tilde{x})-1)^2] + \\ + \lambda|G(z,\tilde{x})-x|_1 \quad (2.38)$$

Зверніть увагу на штрафну норму L1, додану до нашої функції втрат для G. Це зроблено тому що, як було доведено, воно є ефективним при маніпулюванні зображення [40].

Генератор G повністю згортковий без будь-яких повнозв’язних шарів що заощаджує обчислювальний час та підсилюють кореляцію між близькими за часом входами. Існує етап кодування мережі де вхідний сигнал $\tilde{x}$ спочатку проектується та стискається до значної меншого представлення, яке можна з'єднати з прихованим вектором z. Процес проходить через ряд згорток з кроком, які мають PReLU як функції активації. Вони називають цей проектований вектор перед конкатенацією як вектор суті c. Комбінований вектор потім розшифровується симетрично з оберненими згортками та PReLU.

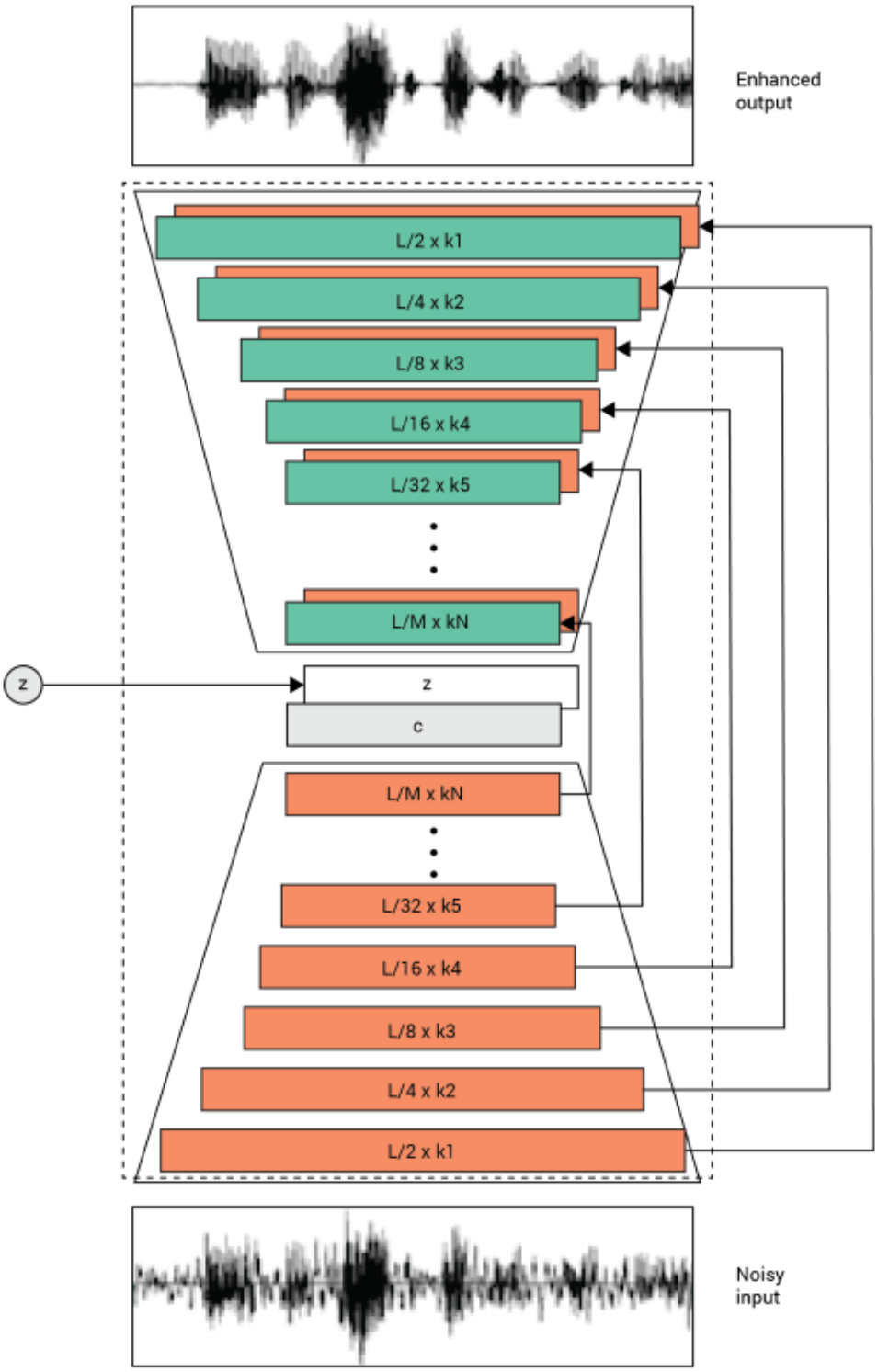

Рисунок 2.21 - Генератор G в SEGAN : Шумний вхід обробляється через стадію кодера згорткового шару, де він закінчується вектором суті c. Цей вектор з'єднаний з шумовим вектором із нормального розподілу N(0, σ) і подається в мережу декодування, яка є дзеркалом стадії кодування.

Так само, як WaveNet, вони також вводять пропускні з'єднання для кращої продуктивності. У цьому випадку кожен шар кодування підключений

до відповідного рівня декодування. Вони пояснюють це тим, що згортка - це вузьке місце, яке може втратити цінну інформацію, а пропуск з'єднань - це підхід для збереження цих важливих деталей низького рівня.

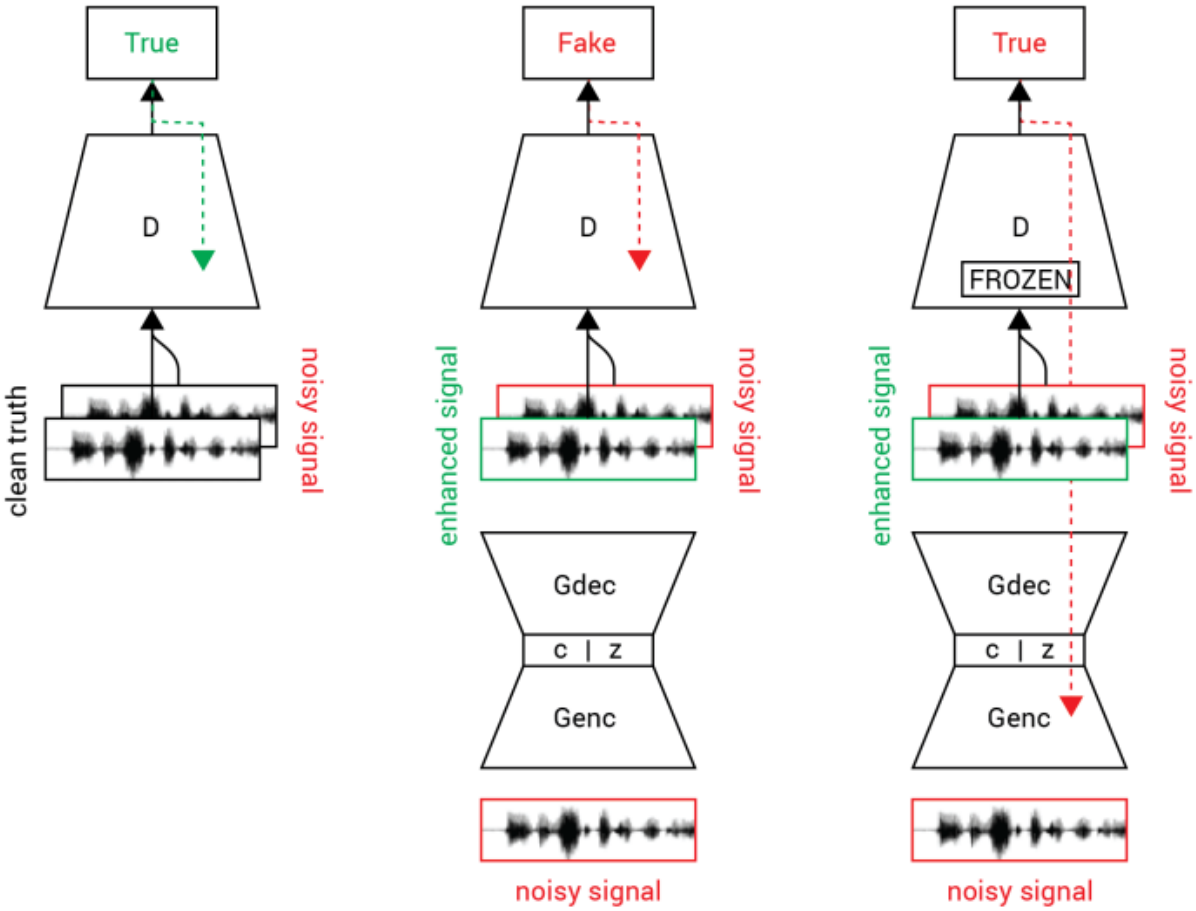

Рисунок 2.22 – 3 випадки SEGAN. Зліва направо : (1) Дискримінатору подається зразки з реальних даних і він позначає їх як реальні. (2) Дискримінатору подається підробки з генератора та він маркує їх як підроблені. (3) Дискримінатору подається підроблені зразки з генератора і маркує їх як справжні. Пунктирними лініями зображено градієнтний потік. Зображення адаптовано від С. Паскуаля [35].

Вони реалізують дискримінатор D аналогічно кодеру генератор, проте єтри ключові відмінності, які вони зазначають:

- Вхідний шар складається з двох вхідних каналів 16384 значень.
- Він використовує LeakyReLU з α = 0,3 і має віртуальну батч нормалізацію.
- Останній шар активації має одновимірний шар згортки з фільтром розміром 1. Причина цього полягає у зменшенні

кількість параметрів у повнозв'язаному шарі від 8×1024 до 8, зберігаючи при цьому навчання 1024 каналів.

Рисунок 2.22, взятий з їх статті, показує архітектуру в повному обсязі.

2.8.2 WaveNet Denoising

Хоча оригінальна архітектура WaveNet [36] орієнтована на генерування мовних зразків з природнім звучанням, Rethage, Pons,і Serra [44] запропонували модифікацію моделі у додаток, який може покращувати мову в аудіофайлах. Вони починають з формулювання проблеми мовлення як знаходження мовного сигналу s_t у змішаному сигналі m_t, визначеному як сума мовного сигналу s_t і фонового шуму b_t :

$$m_t = s_t + b_t \tag{2.39}$$

Основна відмінність архітектури – це видалення причинно – наслідкових згорток, як майбутні вибірки, може допомогти прогнозувати здатність мережі-робота. Вони стверджують, що ми можемо змусити його працювати в реальному часі застосовуючи невелику затримку у часі для доступу до майбутніх даних. Тому причинний характер видаляється з позначення WaveNet модель та передбачувані зразки більше не надходять у мережу. Для обліку майбутніх зразків на вході вони зміщують асиметричне доповнення з оригінальної моделі WaveNet в симетричне доповнення на кожному шарі розширеної згортки, це можна побачити на рисунко 2.23. Це призводить до того, що модель має доступ до такої ж кількості минулих зразків, як і майбутніх, що, у свою чергу, збільшує кількість контексту навколо поточного зразка.

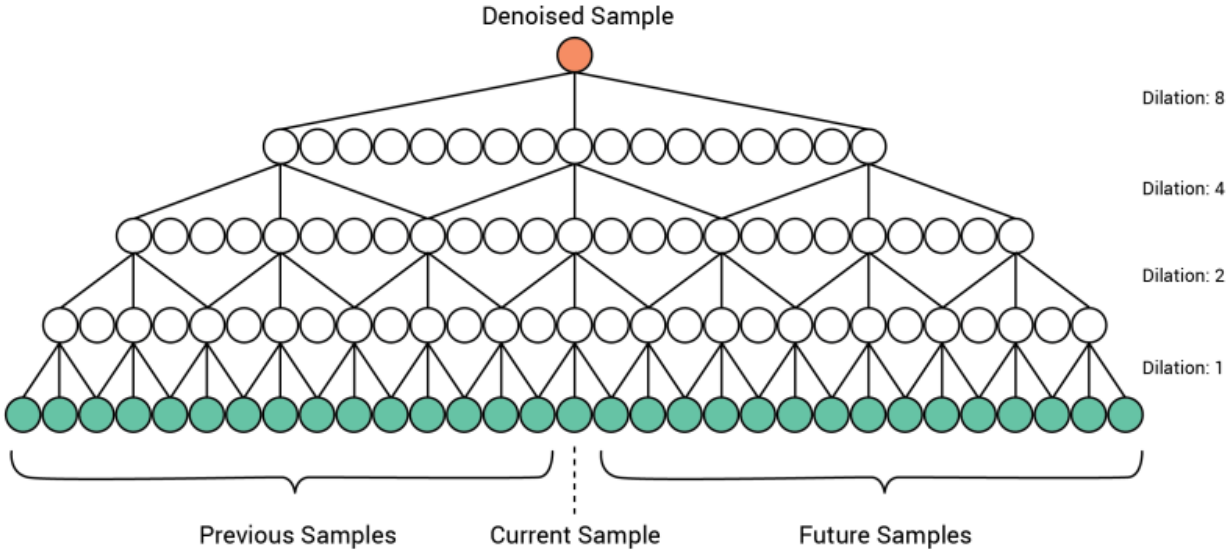

Рисунок 2.23 – Розширені згортки без причинності . Асиметричний характер WaveNet змінено на симетричну форму для подвоєння контекстних зразків навколо поточного. Зображення адаптоване з Rethage, Pons і Serra [44].

Рівняння 2.39 призначене для дискретних виходів softmax і спроектоване щоб не робити жодних припущень щодо розподілу даних. Однак для Rethage, Pons і Serra [44] передбачення дійсної величини за гауссовим розподілом працює набагато краще. Причина цьому потенційно багатомодальний вихід, отриманий з рівняння 2.27 може ввести артефакти, які будуть уникнені у випадку дійсних значень. Вони також стикалися з великою дисперсією та непропорційно високим фоновим шумом через квантування за μ-законом. Отже, вони вирішили ігнорувати попередню обробку і мають модель прогнозування які натомість має дійсні значення у прогнозі.

Не подаючи попередньо згенеровані зразки назад у модель, ми вже не можемо вважати це авторегресивною моделлю. це натомість інша модель, яка намагається мінімізувати втрати функція регресії, яку вони визначають як:

$$\mathcal{L}(\widehat{s_t}) = |s_t - \widehat{s_t}| + \left|b_t - \widehat{b_t}\right| \tag{2.40}$$

де $\widehat{s_t}$ - очищена мова і $\widehat{b_t} = m_t - \widehat{s_t}$. Це втрата L1 з невеликим коригуванням, у вигляді другого доданку.

У своїй роботі вони використовують умовність, коли модель має бінарно закодований скаляр як зсув у кожній операції згортки. Для набору з 28 ораторів, які використовуються в оригінальній статті, скаляр визначається як значення між 1-28. Якщо диктор невідомий, у нас замість цього нуль. На етапі використання моделі встановлюється також нуль.

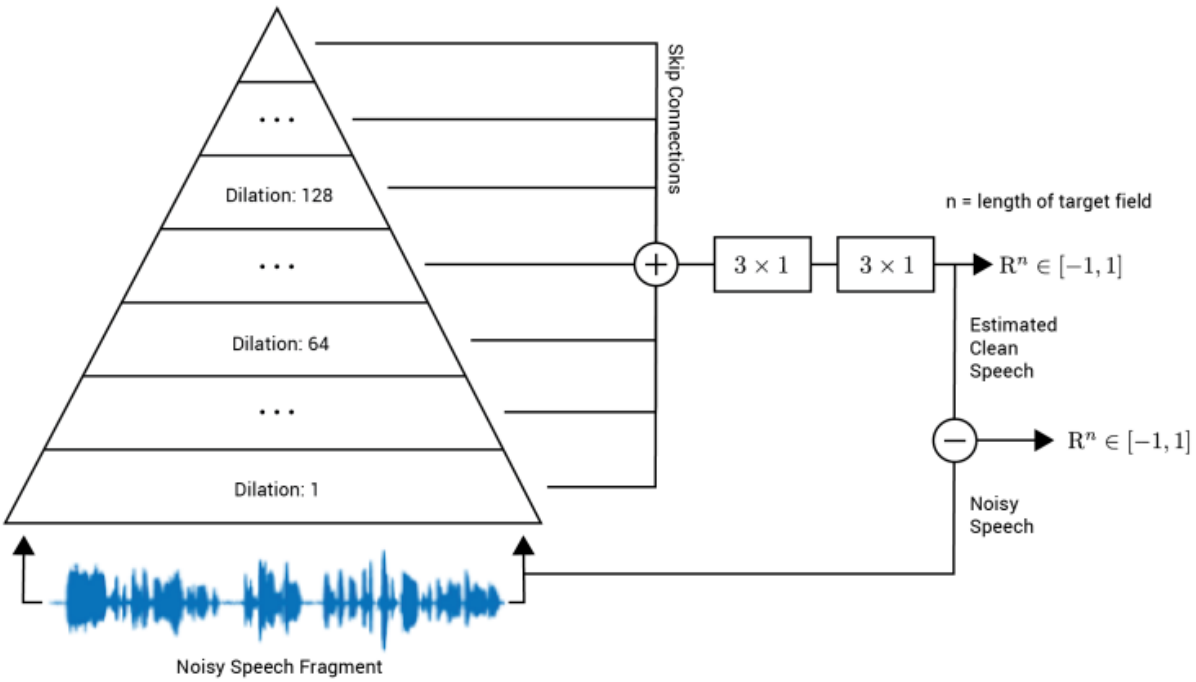

Рисунок 2.24 – Архітектура WaveNet . Симетрична розширена згорткова мережа з пропускними з'єднаннями що виводиться до двох 3x1 фільтрів для прогнозування цільового поля. Шумний фон отримується шляхом віднімання шумної мови від оціночного чистого мовлення. Зображення адаптовано з Rethage, Pons і Serra [44].

Модель має 3 стеки з 30 залишкових шарів, які мають коефіцієнт розширення, який подвоюється для кожного шару від 1 до 512. На першому кроці у мережі лінійно проектується 1-канальний вхід на 128 каналів за згорткою 3x1. Це відповідає кількість фільтрів у кожному залишковому шарі.

Пропускні з'єднання розроблені як 128 згорткових фільтрів розміром 1x1, а вихід - це підсумовування цих з'єднань через ReLU. Потім є два заключні шари згортки 3x1 які не розширені, і містять 2048 та 256 фільтрів відповідно. Вони також використовують ReLU між цими шарами. Нарешті вихідний шар

використовує фільтр 1x1 для лінійного проектування карти ознак в єдиний вихідний сигнал.

Ілюстрація на малюнку 2.24 дає огляд всієї мережі.

2.8.3 EHNet

Ще одна end-to-end модель була представлена Zhao et al. [45] і поєднує в собі CNN з LSTM. EHNet суто керується даними та не дає жодних припущень щодо шуму. Вони визначили проблему як мінімізацію наступної функції, щоб знайти найкращий параметр моделі θ:

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^{n}|g_\theta(x_i) - y_i|^2 \tag{2.41}$$

де $x \epsilon \mathbb{R}_+^{d\times t}$ – спектрограма шуму і $y \epsilon \mathbb{R}_+^{d\times t}$ є спектрограми чистого сигналу. d - розмірність кожного кадру, який вони називають числом частотних вузлів в спектрограмі і t -час, а точніше - довжина спектрограми.

Модель виконана в три чіткі етапи, як видно на рисунку 2.25 виключаючи крок попередньої обробки.

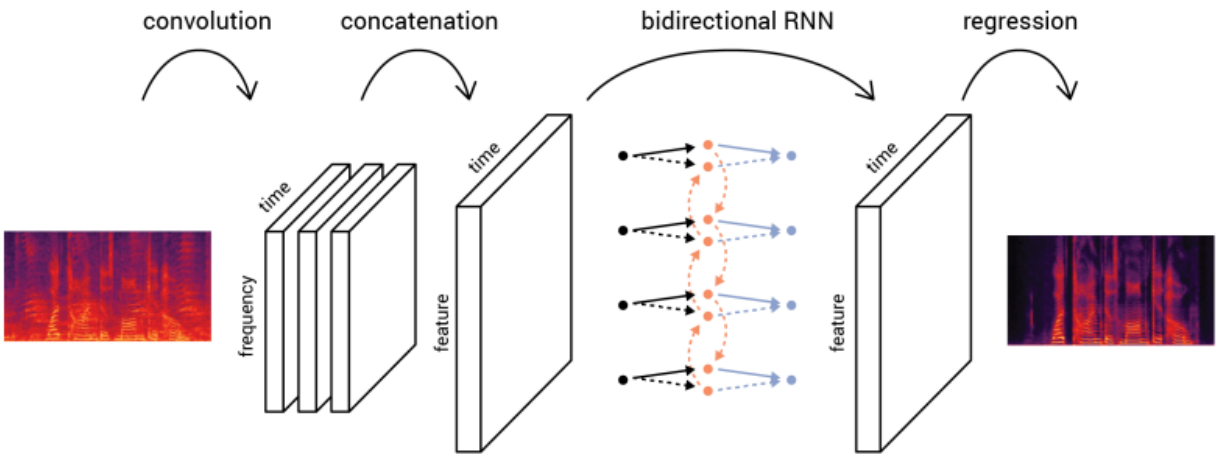

Рисунок 2.25 – EHNet . Архітектура моделі від початку до кінця в три кроки. (1) Вхід обробляється згортковими шарами і з'єднується з двовимірною картою ознак на виході. (2) Карта ознак перетворюється через двонаправлену RNN вздовж часового виміру. (3) Прогноз виконується повнозв'язним шаром і виводить спектрограму кадр за кадром.

EHNet має крок попередньої обробки, який застосовує віконне перетворення Фур'є для отримання спектрограми з кожного аудіофайлу.

Дані спектрограми обробляються згортковою мережею. Вони визначають свої згорткові фільтри як b × w, тобто не симетричного розміру. Для того щоб зберегти розміри вхідної спектрограми після згортки, вони мають нульове доповнення розміром $d \times \frac{w}{2}$, де w має бути непарним. Вони також пропонують зменшити час обчислення, оскільки сусідні частотні вузли дуже схожі, додавши крок згортки $\frac{b}{2}$. Кожен шар активується за допомогою ReLU.

RNN, які реалізовані в EHNet, є двонаправленими, щоб охопити довгострокові відносини в обох напрямках. Перед поданням карти ознак, їх необхідно перетворити на щось прийнятне. Вони вертикально з'єднують кожну карту характеристик уздовж виміру ознак та утворюють складену 2D карту ознак. У математичному позначенні маємо k карт ознак $h_{z_j}(x)\ \epsilon\ \mathbb{R}_+^{p\times t}$, які перетворюються на

$$H(x) = \left[h_{z_1}(x); \ldots; h_{z_k}(x)\right] \epsilon\ \mathbb{R}^{pk\times t} \tag{2.42}$$

Ця нова карта ознак подається в двосторонній LSTM.

Останній компонент – це повністю пов'язаний шар, який приймає вихід від LSTM, $\widehat{H}(x) \epsilon \mathbb{R}^{q \times t}$, і застосовує лінійну регресію для прогнозування $\hat{y}$. Для кожного t ми маємо:

$$\hat{y}_t = max\{0, W\widehat{H} + b_w\}, W \epsilon \mathbb{R}^{q \times t}, b_W \epsilon \mathbb{R}^d \quad (2.43)$$

2.8.4 Стійкість

Стійкість у глибокому навчанні є мірою того, наскільки добре працює мережа на нових даних як навчальних так і тестових. Для перевірки стійкості у граничному випадку, можна було б спробувати залучити змагальні атаки, які можуть перешкоджати мережі приймати рішення. Дослідження [46] [47] в цій темі були проведені для багатьох різних мереж та підходів.

У цій тезі концепція надійності буде використовуватися лише як метод оцінки моделей замість повного на дослідження, тобто ми розглядаємо ефективність моделі при порівняльному аналізі на двох різних наборах даних. Якщо набір даних також не впливає на модель суттєво, ми стверджуємо, що це надійно і навпаки.

2.9 Модифікована модель

В якості базової моделі було обрано модель SEGAN. Паскуаль та ін. [9] використовують модифіковану SEGAN для перетворення шопоту в мову. В якості модифікації вони пропонують використання іншого підходу до регуляризації, що дозволило їм отримати голосніший аналог вхідного звуку.

Іншою зміною було впровадження вагів для обхідних з’єднань, тобто при оберненій згортці до результату додається зважений вихід відповідного згорткового шару. Також вони зменшили кількість шарів залишивши розмірність випадкового вектору шляхом збільшення кроку згортки.

Чому саме ці модифікації розглянуто? З огляду на завдання даної роботи, головним чином нас цікавить швидкість без втрат якості. Оскільки ці модифікації показали непогані результати в задачі перетворення шопоту допускається, що з ними втрата якості не буде значною, в той час коли зменшення кількості шарів удвічі має значно покращити швидкість роботи. Іншим цікавим моментом є ефект реверберації, котрий в ідеї можна компенсувати введенням вагів у обхідних з’єднаннях.

В результаті, аналогічно до запропонованої ними модифікації, ми використаємо ваги для обхідних зв’язків, та більший крок згортки для зменшення кількості шарів, проте не змінюємо спосіб регуляризації, оскільки нашою задачею не є підсилення гучності, та для нас має сенс отримати мову того ж рівня потужності що й вхідний запис. Загалом очікується що мережа буде швидше навчатись, при не значних втратах точності завдяки навчанню вагів обхідних зв’язків. В результаті маємо мережу зображену на рисунку 2.26.

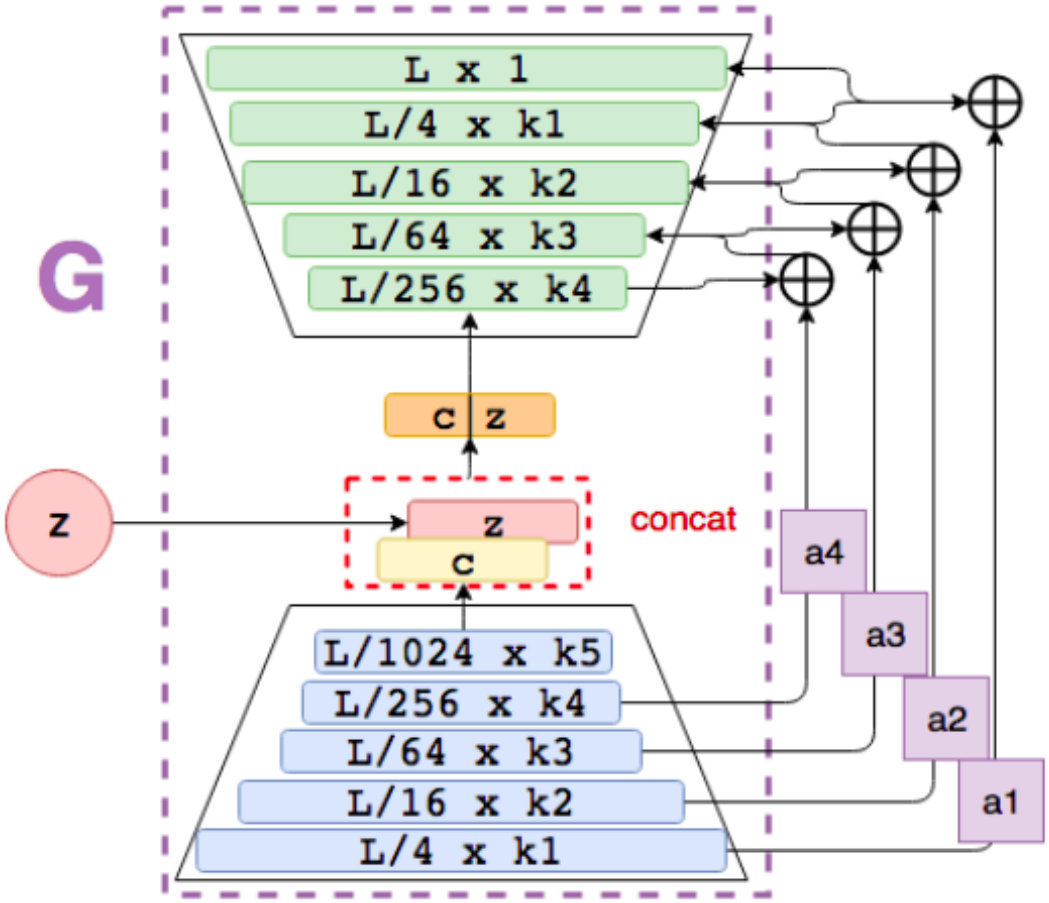

Рисунок 2.26 – Модифікований SEGAN.

2.10 Висновки до розділу

В цьому розділі були наведені теоретичні основи для розуміння концепту та результатів дослідження. Були розглянуті базові штучні нейронні мережі, згорткові нейронні мережі, рекурентні мережі, функції активації, ResNet, тощо. Було наведено основні принципи трьох архітектур що використовуються в роботі та базуються на згорткових і рекурентних нейронних мережах. Була запропонована модифікація однієї з мереж, що за припущенням значно пришвидшить роботу фільтру.

РОЗДІЛ 3 РЕЗУЛЬТАТИ РОБОТИ

3.1 Розробка архітектури системи

Спочатку необхідно окреслити межі системи що розроблюється. Оскільки основним застосуванням є фільтрація в режимі онлайн, аналогічно до RTX Voice, то ця система має виступати прошарком між додатком що використовує користувач та джерелом звукозапису який використовує цей додаток. Тобто наша система має отримувати звуковий потік від певного джерела, яким найчастіше виступає драйвер звукозаписуючого пристрою робити відповідні перетворення та видавати потік обробленого звуку у додаток.

Оскільки різні драйвери та пристрої мають різну робочу частоту, а мережа орієнтована на зразки з первою частотою (так званий «сеплінг»), то перед використанням нейронної мережі необхідно виконати «ресемплінг» який найчастіше вирішується як задача інтерполяції.

Аналогічно «ресемплінг» необхідно проводити після отримання результату мережі, оскільки додаток може очікувати дані іншої форми. Загальна схема зображена на рисунку 3.1.

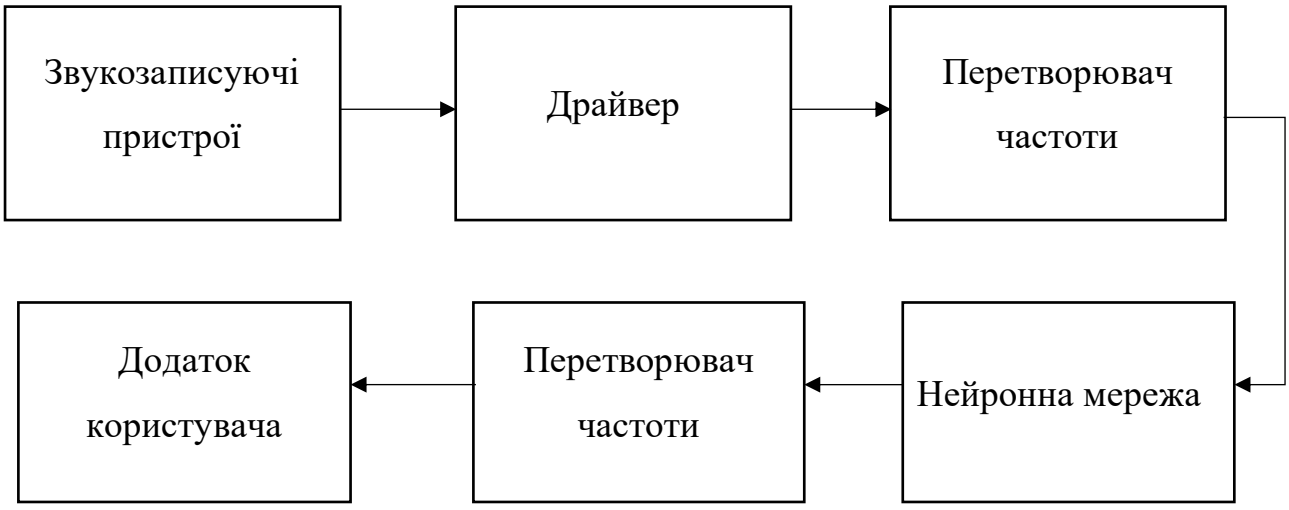

Рисунок 3.1 –Потік звукозапису.

Натомість перетворювача частоти може також виступати інша

трансформація, у випадку коли вхідні дані відрізняються не лише за частотою дискретизації, а й за іншими характеристиками.

Оскільки пристрій та драйвер вже присутні в системі що може використовувати користувач, оскільки він використовує додаток для спілкування, то частинами системи що потребує реалізації є перетворювач та нейронна мережа.

В якості перетворювача можна використовувати вже готову утиліти, наприклад «soc» для Linux орієнтованих систем, що був використаний для підготовки даних в експерименті.

Таким чином частина системи що потребує реалізації, це безпосередньо нейронна мережа та побудова зав’язків між блоками системи, або іншими словами, потоку даних.

Оскільки для формування потоку необхідно більш детально вивчати принцип потому даних від драйвера до додатку (щоб перехопити цей поток), то реалізація цієї частини не розглядається. Тобто в даній роботі розглядається реалізація безпосередньо нейронної мережі.

Також задля вилучення фактору зміни оточення, нейронну мережу реалізовано в рамках Docker контейнера, що робить можливим використання його як оболонки для нашої системи. Також у цей контейнер можна помістити трансорматори частот, в такому випадку схема матиме вигляд зображений на рисунку 3.2. Проте слід зауважити, що використання контейнерів дещо обмежує вибір операційних систем користувачів (також досі не впроваджено можливість використання GPU у Docker для Windows), проте спрощує розробку самого додатку. Оскільки для Windows існує RTX Voice (більшість користувачів Windows мають GPU від Nvidia), то виконання нашої системи в Docker контейнері є доцільним.

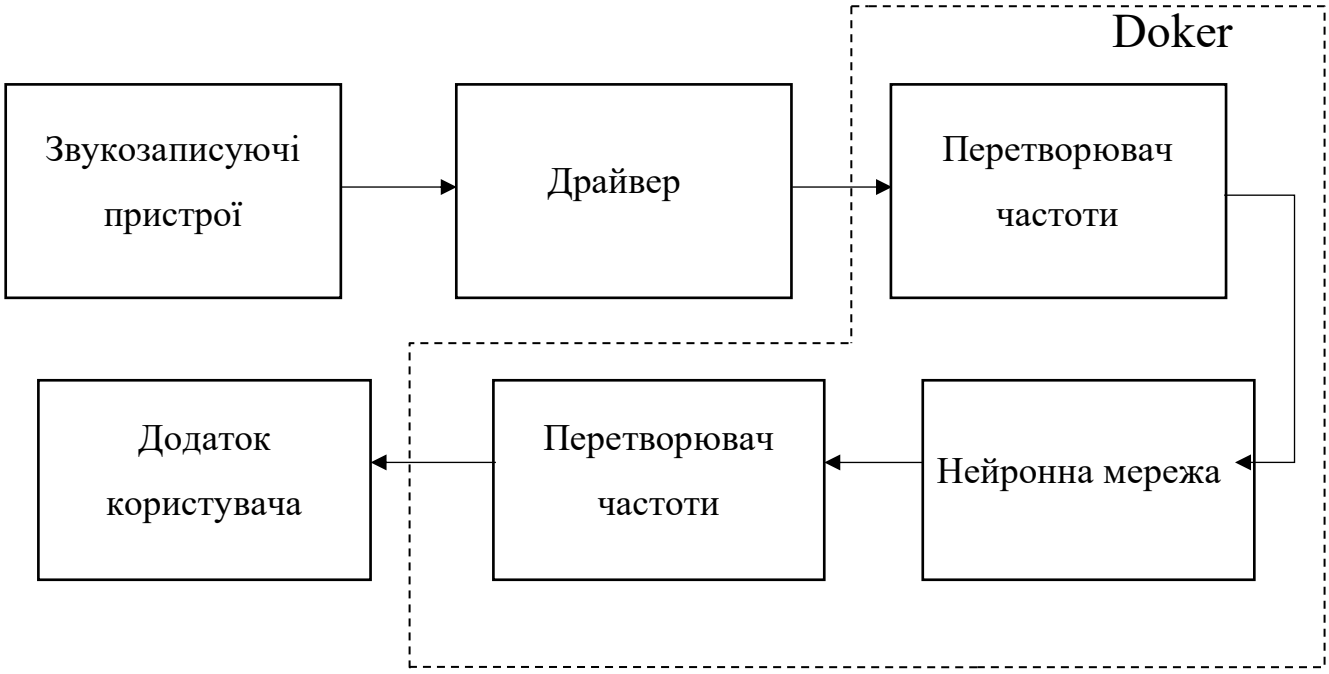

Рисунок 3.2 – Потік звукозапису з використанням Doker контейнеру.

Таким чином, в даній роботі пропонується архітектура зображена на рисунку 3.2, та залишається розглянути вибір нейронної мережі.

Для того щоб обрати яку саме мережу слід використовувати, необхідно провести експеримент та провести порівняльний аналіз його результатів.

3.2 Аналіз експерименту

3.2.1 Формування датасету

Дані, використані в цій дисертації, були взяті з записів опублікованих Едінбургським університетом [41]. Її опис викладено в роботі Valentini-Botinho et al. [42]. Були використано записи англійської мови через наявність даних, та те що вони були використані в роботах авторів розглянутих архітектур.

У своїй роботі вони заявляють, що це набір чистих і зашумлених паралельних записів мовлення, призначена для підготовки та тестування моделей покращення мови, які працюють на 48 кГц. Невелика попередня

обробка робиться для зменшення частоти до 16 кГц для більш швидких обчислень.

Набір даних складається з двох частин, і перша частина включає 28 мовців з одної області акцентів в Англії, що розділяє 50/50 між чоловічим та жіночим голосом. Друга частина має 56 ораторів (аналогічний розподіл) з різних акцентних регіонів Шотландії та Сполучених Штатів. Кожен оратор має близько 400 речень.

За даними Valentini-Botinho та ін. [42] вони створили два штучні шуми і мали вісім записів реального шуму, взятих з Demand database[43]. Штучний шум являв собою мовленнєвий шум і стукіт. Перший генерувався "фільтруванням білого шуму з фільтром, частотна характеристика якого відповідала рівню тривалої мови мовця", тоді як другий генерувався шляхом "додавання мови шести спікерів з Voice Bank, які не використовувались ні для навчання, ні для тестування ».

Справжні шуми діставались з перших каналів відповідної версії 48 кГц у Demand database. Ці шуми в деталях були: «побутовий шум (на кухні), офісний шум (у залі), три шуми у громадських місцях (кафетерія, ресторан та станція метро), два шуми в транспорті (автомобіль та метро) і вуличний шум (перехрестя із щільним дорожнім рухом) ”.

У них було 4 різних значення сигнал-шум для кожного шуму (0 db, 5 db, 10 db і 15 db), що означало, що вони мали загалом 40 різних шумних умов. Детальніше про те, як шуми були спеціально побудовані та додані до чистих звукових хвиль, можна прочитати у їх документі, якщо читач зацікавлений дізнатися більше.

У оригінальних статтях для позначення SEGAN та WaveNet використовувався лише набір даних, що складається з 28 різних динаміків. В цій роботі ми розширили тести, а також включили набір даних 56 спікерів. Менший набір даних ораторів все ще використовувався для відтворення

результатів двох робіт.

Для позначення EHNet та WaveNet використовується набір перевірки. На жаль, якщо код відповідає їх роботі, вони використовували тестовий набір як валідаційний. Ми знаємо, що це є порушенням навчального процесу, оскільки він забруднює дані навчання та впливає на кінцеві результати під час оцінювання. Для того щоб вирішити цю проблему ми модифікуємо дані тренувань і розділяємо їх наступним чином:

- Набір даних 28 ораторів: p243 (чоловіки) та p268 (жінки) опущено з навчального набору і замість цього додається до набору перевірок.
- Набір даних 56 спікерів: p234 (жінка), p336 (жінка), p237 (чоловік) та p245 (чоловік) опущено з навчального набору і замість цього додається до набору перевірок.

Ми все ще вважаємо, що набір валідації є частиною навчальних даних, проте модель не чітко навчається на цьому наборі.

3.2.2 Умови експерименту

Для оцінки PESQ використовувався модуль Python, розроблений Jingdong Li [51] на github. STOI взято з MATLAB реалізації CH Taal [52].

Також проводилась людська оцінка на випадкових зразках, що була усереднена за всіма оцінювачами. Записи обрані для людської оцінки зображено в таблиці 3.1.

Таблиця 3.1 – Набір даних для людської оцінки

Речення що читається	.wav-файл з набору
Please call Stella.	p232_001.wav
Now the system is right behind us.	p232_089.wav
We did not compete with any other local farmer.	p232_252.wav
This latter point is hugely important.	p232_155.wav
I was in a position to challenge for this event and didn’t.	p232_410.wav
It is normal.	p257_110.wav
He raised the profile of the European Tour to the sky.	p257_256.wav
Everything will fall into place, it should be fine.	p257_265.wav
But it may take some time to confirm the findings.	p257_422.wav

Через те що глибокі нейронні мережі навчаються досить повільно, в даній роботі не розглядається пошук оптимальних параметрів (на базі GTX 1050Ti навчання однієї мережі триває більше тижня), тому обрані параметри вказані авторами як оптимальні.

Для SEGAN та його модифікації:

- початкове стандартне відхилення вектора шуму z - 0;
- початкова вага для норми L1 встановлена на 100;
- розмір батчу - 100;
- попередня частотна фільтрація на данних з параметром 0.95;
- у всій мережі присутні нейрони зсуву;
- навчання триває 86 епох.

Для WaveNet Denoising:

- глибина розширення - 9;
- постійна швидкість навчання та оптимізатор - AdamOptimizer;

- цільова довжина - 1601;
- розмір батчу - 10;
- 250 епох, з реалізацією завчасної зупинки.

Для EHNet:

- два згорнуті шари з 256 фільтрами. Кроки 1 та 16, а розмір фільтра - 11 та 32 відповідно;
- два шари lstm розміром 1024;
- відхилення – 0.02;
- розмір батчу – 32;
- 200 епох, з реалізацією завчасної зупинки.
- AdamOptimizer з швидкістю навчання від 1e-3 до 1e-6

Фільтр Вінера був використаний для порівняння. В якості реалізації ми використали реалізацію MATLAB [51].

Всі моделі навчались в контейнерах Docker для відтворення необхідного оточення для коду авторів.

3.2.3 Результати експерименту

Для розуміння швидкості навчання мереж далі наведено графіки помилок при навчанні.

Оскільки SEGAN має окремо помилку генератора та оцінювача, при тому що кожна з них має складові, для наглядності зображено по 6 графіків на мережу.

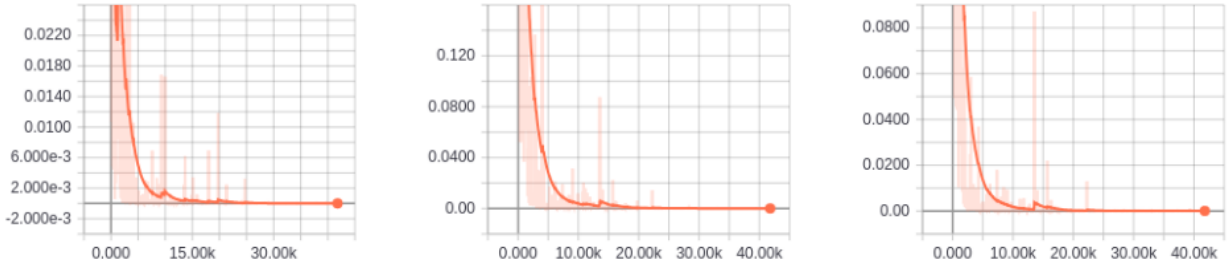

Рисунок 3.3 - помилка SEGAN оцінювача для набору даних 28 ораторів. Зліва праворуч: (1)Помилка фейку. (2) Загальна помилка. (3). Помилка оригіналу.

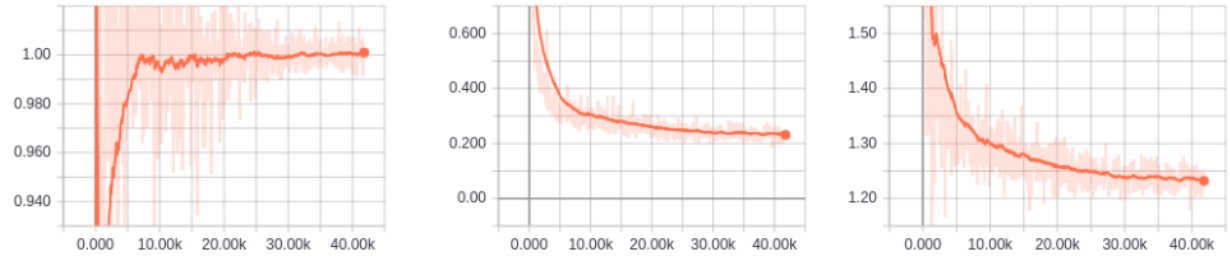

Рисунок 3.4 - помилка SEGAN генератора для набору даних 28 ораторів. Зліва праворуч: (1)Помилка генератора. (2) штраф L1-норми. (3). Загальна помилка.

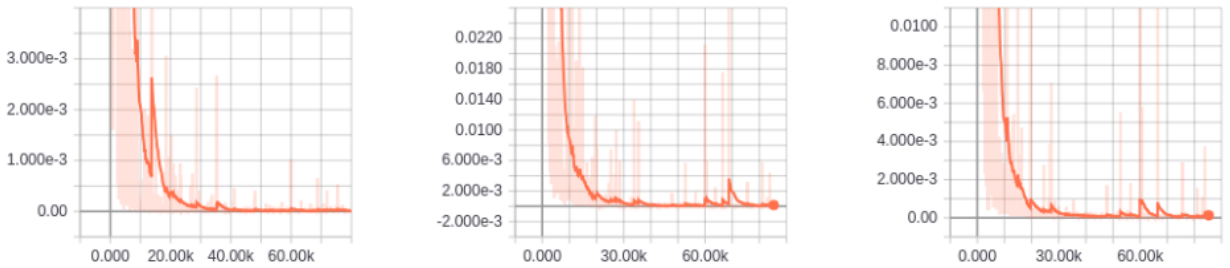

Рисунок 3.5 - помилка SEGAN оцінювача для набору даних 56 ораторів. Зліва праворуч: (1)Помилка фейку. (2) Загальна помилка. (3). Помилка оригіналу.

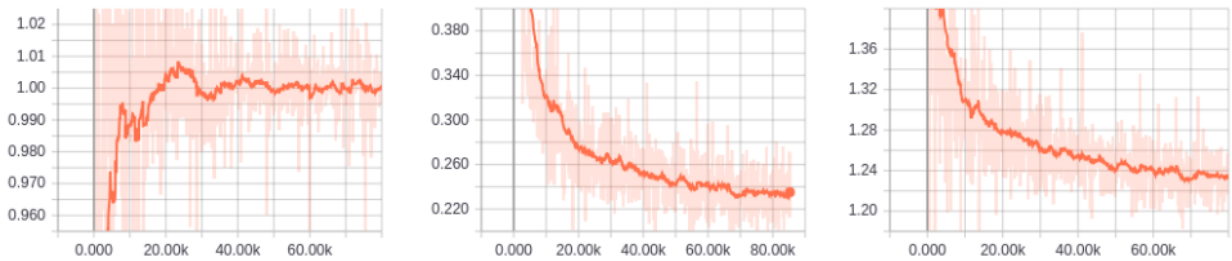

Рисунок 3.6 - помилка SEGAN генератора для набору даних 56 ораторів. Зліва праворуч: (1)Помилка генератора. (2) штраф L1-норми. (3). Загальна помилка.

Для WaveNet зображено звичні графіки тренування, а саме тренувальна та валідаційна помилки.

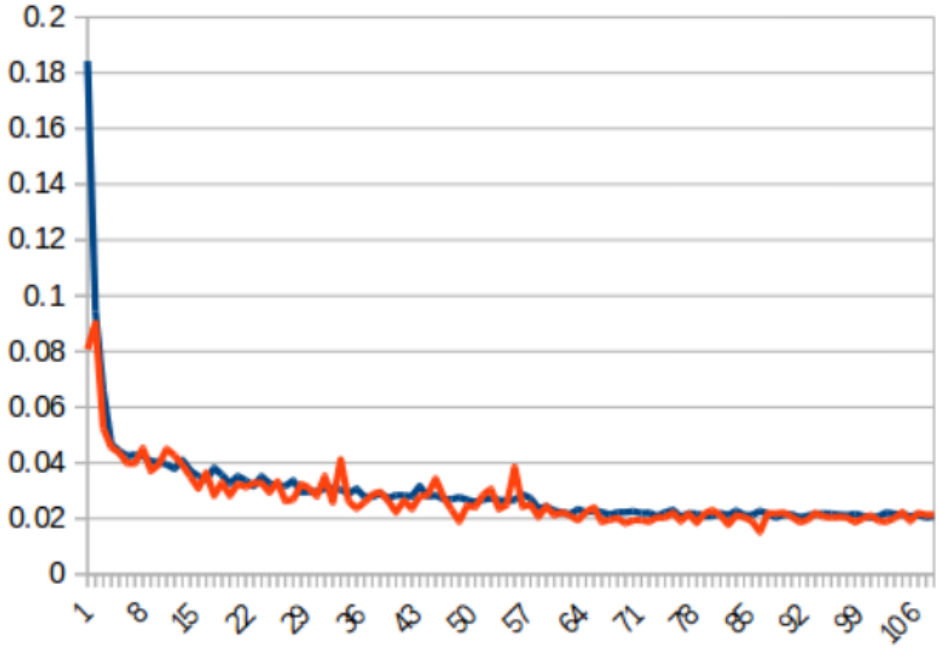

Рисунок 3.7 - Помилка навчання WaveNet для 28 ораторів. Синій позначає тренувальну помилку, а червоний - валідаційну.

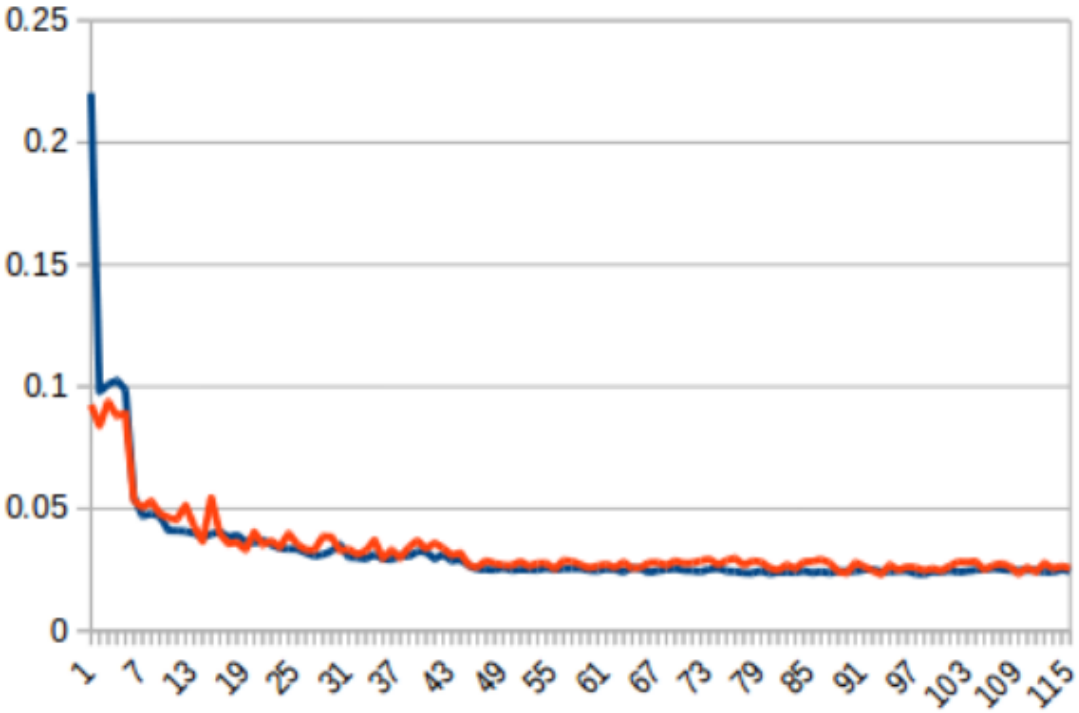

Рисунок 3.8 - Помилка навчання WaveNet для 56 ораторів. Синій позначає тренувальну помилку, а червоний - валідаційну.

Для EHNet наведено графіки аналогічні до WaveNet.

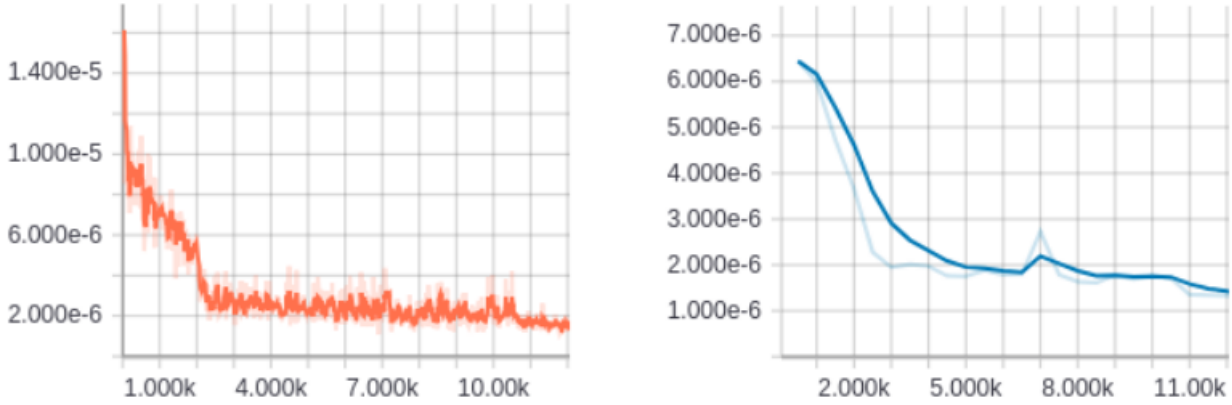

Рисунок 3.9 - Навчання EHNet. Тренувальна помилка (зліва) та валідаційна (зправа) для 28 ораторів.

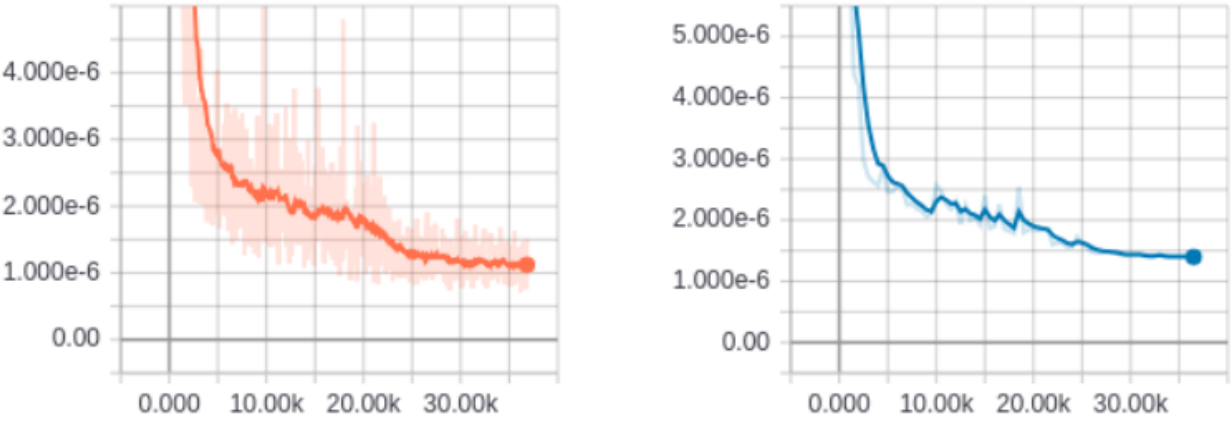

Рисунок 3.10 - Навчання EHNet. Тренувальна помилка (зліва) та валідаційна (зправа) для 56 ораторів.

Оцінка роботи тренованих мереж наведена у таблиці 3.2.

Для WaveNet відсутні метрики STOI через специфіку архітектури, а саме необхідність в майбутніх зразках, що призводить до обрізання запису.

Для того щоб оцінити котру з мереж краще використовувати необхідно порівняти якість та швидкість роботи кожної з розглянутих архітектур.

В результаті бачимо, що найкращі результати показує SEGAN 56, в той час коли його модифікація показує дещо гірші результати. Проте через зменшення кількості шарів модифікація швидше навчається та працює.

Таблиця 3.2. – Оцінка результатів

Модель	PESQ	STOI	Людська оцінка
WaveNet denoising 28	2.91	-	3.01
WaveNet denoising 56	3.04	-	2.93
SEGAN 28	3.22	0.932	3.47
SEGAN 56	3.32	0.935	3.58
Модифікований SEGAN 28	3.10	0.906	3.30
Модифікований SEGAN 56	3.20	0.924	3.47
EHNet 28	2.71	0.886	2.87
EHNet 56	2.86	0.900	2.80
Wiener	3.02	0.921	3.00
Noisy	3.01	0.921	3.07

Іншим фактором порівняння є швидкість роботи та навчання мереж. Проте через нестачу обчислювальних можливостей, час тренування не вдалося виміряти, оскільки воно дуже залежало від навантаження на обладнання та перебоїв електромережі. Тому у порівнянні ми розглядаємо лише час роботи навчених мереж. В якості часу роботи мережі використовується час від подачі даних на вхід мережі до отримання виходу при довжині звукозапису в 2 секунди. Тобто не враховується час завантаження самої мережі, оскільки при довготривалому використанні, цим часом можна знехтувати.

Таблиця 3.3. – Оцінка швидкості роботи

Модель	Час роботи
WaveNet denoising	1.7с
SEGAN	2.9с
Модифікований SEGAN	1.8с
EHNet	5.1с

Якщо порівнювати швидкість навчання, найшвидше працювала WaveNet Denoising, трохи повільніше запропонована модифікація, ще повільніше оригінальний SEGAN і найбільш повільною виявилась EHNet.

Таким чином маємо що лідером у швидкості стала WaveNet denoising, тоді як лідером у якості став оригінальний SEGAN. Проте, оскільки ми шукаємо компроміс для використання у руальному часі, бачимо що запропонована модифікація порівняна у швидкості з лідером, тоді як дає кращі результати за якістю. Враховуючи це, кращим кандидатом на використання у системі в реальному часі є запропонована модифікація.

3.3 Висновки до розділу

В даному розділі було описано проведення та результат експерименту. Було проведено порівняння різних архітектур та запропонованої модифікації.

Була запропонована архітектура системи що виконує виділення голосу у вигляді прошарку між додатком користувача та джерелом потоку аудио.

Дослідження показало, що найкращий результат, з точки зору якості досягається використанням мережі SEGAN, проте цю архітектуру не можливо

використовувати в реальному часі, оскільки час обробки більший за час запису. Натомість запропонована модифікація показує себе трохи гірше, але за рахунок значного пришвидшення, саме вона використовується у якості реалізації фільтру.

Окрему увагу в якості подальших досліджень займає оцінка результатів. Як можна було побачити оцінки для початкових записів з шумами мали іноді кращі значення ніж у обробленого запису.

Іншим напрямом подальших досліджень є зміна тренувальних даних, а саме, наприклад, формування набору для української мови.

Також можна зауважити, що результати далекі від приватної технології NVIDIA RTX Voice, проте вирішують її головний недолій, а саме відсутність реалізації для інших апаратних на програмних середовищ.

РОЗДІЛ 4 РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

У даному розділі буде розглянуто ключові особливості розробленої системи як майбутнього стартап-проєкту. Проєкт розглядатиметься як система виділення людського голосу у реальному часі.

4.1 Опис ідеї проєкту

Спочатку проаналізуємо та подамо у вигляді таблиці зміст ідеї стартап-проєкту, можливі напрямки застосування та основні вигоди, які може отримати користувач товару. Ці характеристики стартап-проєкту зображено в таблиці 4.1.

Таблиця 4.1 - Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Програмний додаток для покращення голосу.	1. Застосування як системи що покращує голос в реальному часі при розмові.	Додатковий комфорт при онлайн спілкуванні.
	2. Застосування як системи що покращує голос на готовому звукозаписі.	Можливість створення контенту без використання професійного обладнання.

Аналіз потенційних техніко-економічних переваг ідеї порівняно із пропозиціями конкурентів передбачає:

- визначення переліку техніко-економічних властивостей та характеристик ідеї;

- визначення попереднього кола конкурентів, проектів-конкурентів, товарів-замінників чи товарів-аналогів, що вже існують на ринку;
- збір інформації щодо значень техніко-економічних показників для ідеї власного проекту та проектів-конкурентів.

Відповідно до визначеного вище переліку проводиться порівняльний аналіз показників: гірші значення (W, слабкі); аналогічні (N, нейтральні) значення; кращі значення (S, сильні).

Визначення сильних, слабких та нейтральних характеристик ідеї стартап-проекту “система виділення людського голосу у реальному часі” наведено у Таблиці 4.2.

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

№ п/п	Техніко-економічні характери-тики ідеї	Товари/концепції конкурентів		Характеристика
		Мій проект	Конкурент 1	
1.	Форма виконання	Консольний застосунок	Десктопний застосунок	W
2.	Собівартсть	Низька	Низька	W
3.	Якість результатів	Середня	Висока	W
4.	Наявність інтернету	Ні	Ні	N
5.	Кросплатформеність	Так	Ні	S
6.	Складність використання/ автономність	Ні	Так	S

4.2 Технологічний аудит ідеї проєкту

Визначимо технологічну здійсненність ідеї проєкту за допомогою аналізу таких складових, як технології, за якою буде виготовлено товар згідно ідеї проєкту, існування таких технологій, чи їх необхідно розробити / доробити, доступність таких технологій авторам проекту. Результати даного аналізу зображено в таблиці 4.3.

Таблиця 4.3 – Технологічна здійсненність ідеї проєкту

Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
Програмний додаток для виділення голосу людини.	Технологія обробки звукових файлів soc	Так	Дані технології доступні та використовуються максимально просто
	Технологія навчання нейромережі та її використання Python, TensorFlow .	Так	Дані технології доступні.
	Технологія впровадження GPU для розрахунків від Nvidia - CUDA	Так	Дані технології доступні^проте необхідно підтримувати нові версії.
Обрана технологія реалізації ідеї проекту: технологіяобробки звукових файлів soc, роботи з нейромережами Python TensorFlow, використання GPU - CUDA.			

4.3 Аналіз ринкових можливостей запуску стартап-проєкту

Проведемо аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку. Результати даного аналізу зображено в таблиці 4.4.

Таблиця 4.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	*Показники стану ринку (найменування)*	*Характеристика*
1	Кількість головних гравців, од	2
2	Загальний обсяг продаж, грн/ум.од	1 500 000
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Висока якість очищення, швидкодія
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	65

Таким чином, за попереднім оцінюванням, ринок є привабливим для входження.

Надалі визначимо потенційні групи клієнтів, їх характеристики, та сформуємо орієнтовний перелік вимог до товару для кожної групи. Ці дані зображено в таблиці 4.5.

Таблиця 4.5 – Характеристика потенційних клієнтів стартап-проєкту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
1	Покращення якості звучання у онлайн спілкуванні	Малий бізнес та приватний користувач	Може виникнути потреба в інтеграції до додатку чи нового оточення	Клієнти прагнуть покращення комфорту при спілкуванні.
2	Створення контенту без необхідності у професійному обладнанні	Малий бізнес	Може виникнути потреба у графічному інтерфейсі.	Клієнти прагнуть якісного очищення їх звукозаписів від шумів, без спотворень

Після визначення потенційних груп клієнтів проведемо аналіз ринкового середовища: складемо таблиці факторів, що перешкоджають ринковому впровадженню проєкту (таблиця 4.6), та факторів, що йому сприяють (таблиця 4.7).

Таблиця 4.6 – Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1	Відсутність попиту	Бізнес може не оцінити переваги продукту	Акцентувати увагу на перевагах конкурента та неможливості його використання через обмеження
2	Неякісне фільтрування	Особливості мовлення певних осіб можуть привести до спотворень при фільтрації	Донавчання мережі на нових «особливих» даних..

Таблиця 4.7 – Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1	Кобрендінг	Пропозиція від певної компанії, що спеціалізується на системах покращення аудио, розробити спільний продукт	Виділення частини штату на реалізацію кобрендінгу, кооперація власними розробками між компаніями

Надалі проведемо аналіз пропозиції: визначимо загальні риси конкуренції на ринку. Результати даного аналізу зображені в таблиці 4.8.

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
1. Чиста конкуренція	Гравці ринку не мають явних переваг один над одним	Маркетинг (пояснення ідеї дешевої вартості, підтримки та гарної точності додатку)
2. Регіональна конкуренція	Гравці ринку – інтернаціональні підприємства	Маркетинг (пояснення ідеї дешевої вартості, підтримки та гарної точності додатку)
3. Внутрішньогалузева конкуренція	Гравці ринку знаходяться в одній галузі – розробці ПЗ	Маркетинг (пояснення ідеї дешевої вартості, підтримки та гарної точності додатку)
4. Товарно-видова конкуренція	Усі продукти гравців ринку мають одне призначення	Розробка найбільш інтуїтивного інтерфейсу

Продовження таблиці 4.8

5. Конкурентні переваги нецінові	Продукти відрізняються гнучкістю, функціоналом (незначно) і надійністю.	Маркетинг (пояснення ідеї дешевої вартості, підтримки та гарної точності додатку)
6. Марочна конкуренція	Значна увага приділяється бренду, що розробив продукт	Кобрендінг

Тепер визначимо та обґрунтуємо фактори конкурентоспроможності, які зображені в таблиці 4.9.

Таблиця 4.9 – Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
1	Іноваційність	Продукт представляє реалізацію голосової біометрії – одного з напрямків Штучного Інтелекту
2	Невибагливість до апаратних ресурсів (серверів). А отже дешевизна апаратних ресурсів, потрібних для нашої системи	В продукті використане поєднання класичних методів голосового розпізнавання та методів Машинного навчання
3	Швидкодія	В продукті використане поєднання класичних методів голосового розпізнавання та методів Машинного навчання
4	Точність	Продукт має високу точність аутентифікації, яка може бути порівняна з системами більш висоукого класу
5	Юридична перевага	Відсутня потреба продукту у Big Data, що дає можливість зменшити пов’язаний з ним юридичний ризик
6	Інтеграція	Продукт може бути використаний в будь-якому веб-сайті захищеному протоколом ssl. Продукт не потребує придбання спеціалізованого апаратного забезпечення

Таблиця 4.10 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальни-ки	Клієнти	Товари-замінн
	Динаміка галузі, продуктова лінія, бар’єри проникнен-ня	Наявність товарних знаків, доступ до ресурсів, патенти на продукти	Концентрація постачальни-ків, диференціація витрат	Рівень чутливо-сті до зміни цін, прибутки, зворотній зв’язок	Ціна, лояль ність спожи ва-чів
Виснов-ки:	Конкурен-ція не є інтенсив-ною, адже конкурентів мало (або немає взагалі).	Для входу на ринок необхідно створити товарний знак та написати бета-версію програм-ного продукту.	Постачаль-ники не диктують умови роботи на ринку, бо програмному продукту не потрібно постачання.	Клієнти диктують умови роботи на ринку, бо вони є єдиним джерелом прибутку компанії.	При наявно товарів замінн необхі буде зменш ти програ ного продук створю ПЗ інших технічн систем

За визначеними факторами конкурентоспроможності проведемо аналіз сильних та слабких сторін стартап-проєкту. Результати даного аналізу зображено в таблиці 4.11.

Таблиця 4.11 – Порівняльний аналіз сильних та слабких сторін системи «BioM»

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з BioM*						
			–3	*–2*	*–1*	*0*	*+1*	*+2*	*+3*
1	Іноваційність	15					*		
2	Невибагливість до ресурсів	18		*					
3	Швидкодія	18			*				
4	Якість	10						*	
5	Юридична превага	14			*				

Тепер проведемо SWOT-аналіз на основі виділених загроз і можливостей, та сильних і слабких сторін проєкту. SWOT-матриця зображено в таблиці 4.12.

Таблиця 4.12 – SWOT-аналіз стартап-проєкту

Сильні сторони: іноваційність, невибагливість до обчислювальних ресурсів, швидкодія, точність, юридична перевага	Слабкі сторони: Немає належного досвіду у веденні бізнесу
Можливості: Кобрендінг	Загрози: Відсутність попиту, неточність розпізнавання, порушення прав конфеденційності споживачів (GDRP)

На основі SWOT-аналізу розробимо альтернативи ринкової поведінки для виведення стартап-проєкту на ринок та орієнтований оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок. Дані альтернативи зображено в таблиці 4.13.

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проєкту

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1	Реалізація можливості використання системи не тільки на веб-сайтах, а й телефонних додатках	Середня	18 місяців
2	Створення системи емоційного розпізнавання людини	Висока	22 місяці
3	Розробка MVP	Висока	12 місяців

Серед даних альтернатив було обрано третю альтернативу, адже строки її реалізації найменші та є ймовірність отримання ресурсів.

4.4 Розроблення ринкової стратегії проєкту

Для розроблення ринкової стратегії першим кроком необхідно описати цільові групи потенційних споживачів, які можна побачити в таблиці 4.14.

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1.	Малий бізнес	Готові	2-3 підприємств в рік	Слабка	Середня
2.	Середній бізнес	Середня	1-2 підприємств в рік	Велика	Велика
3.	Великий бізнес	Мала	1 заклад в рік	Велика	Велика
Було обрано цільову групу підприємств групи малого бізнеса.					

Для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку, яку ображено в таблиці 4.15.

Таблиця 4.15 – Визначення базової стратегії розвитку

Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Концентрація на потребах одного цільового сегменту – веб-сайтах.	Створений продукт є іноваційним дешевим початково та дешевим у використанні	Стратегія спеціалізації.

Наступним кроком є вибір стратегії конкурентної поведінки, яку зображено в таблиці 4.16.

Таблиця 4.16 – Визначення базової стратегії конкурентної поведінки

Чи є проєкт «першопрохід-цем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
Так.	Компанія буде шукати нових споживачів	Компанія буде копіювати конкурентів.	Стратегія занять я ніші.

Тепер розробимо стратегію позиціонування, що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проєкт. Її зображено в таблиці 4.17.

Таблиця 4.17 – Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспро-можні позиції власного стартап-проєкту	Вибір асоціацій, які мають сформувати комплексну позицію власного проєкту (три ключових)
Розпізнавання особи за голосом має бути точним. Система є дешевою початково і у використанні Система є швидкою	Проведен-ня крупних оновлень (оптимі-зація розрахун-ків), постійний зворотній зв’язок від клієнтів.	Товар є іноваційним та дешевим при покупці та у використанні порівняно з альтернативами	Іноваційний, дешевий, невибагливий до обчислювальних ресурсів точний, зручний, програма працює в режимі онлайн.

4.5 Розроблення маркетингової програми стартап-проєкту

Сформуємо маркетингову концепцію товару, який отримає споживач. В таблиці 4.18 зображено результати попереднього аналізу конкурентоспроможності товару.

Таблиця 4.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1.	Прийнятна початкова ціна	Прийнятна початкова ціна, в порівнянні з альтернативами	Доступність для компаній з невеликим капіталом.
2.	Прийнятна вартість обслуговування	Невибагливість до апаратних ресурсів клієнта	Можливість встановлення на пристрої з обмеженою потужністю
3.	Іноваційність	Продукт належить до голосової біометрії - напрямку Штучного Інтелекту	Були використані перевірені бібліотеки машинного навчання
4.	Швидкість роботи	Швидкість аутентифікації	Можливість роботи в режимі онлайн
5.	Точне розпізнавання особи.	Точне розпізнавання особи за її голосом.	

Надалі розробимо трирівневу маркетингову модель товару: уточнимо ідею продукту, його фізичні складові, особливості процесу його надання. Дана модель зображена в таблиці 4.19.

Таблиця 4.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
I. Товар за задумом	Програмний продукт – система голосової біометрії, яка надає користувачу додаткову зручну альтернативу звичній нудній аутентифікації по паролю.
II. Товар у реальному виконанні	Властивості / характеристики: 1. Можливість зареєструватися в систему за допомогою голосу 2. Можливість пройти аутентифікацію за допомогою голосу особи 3. Можливість пройти аутентифікацію іншим, альтернативним, спомобом
	Якість: програмний продукт пройшов всі етапі тестування та готовий до використання.
	Файл з розширенням “.ру”, віртуальне середовище.
	Марка: назва організації-розробника «YG», назва товару «BioM».
Рівні товару	Сутність та складові
III. Товар із підкріпленням	Спеціаліст із впровадження встановлює ПЗ.
	Відділ розробки підтримує життєдіяльність ПЗ.
Захист програмного продукту буде організовано за допомогою ноу-хау.	

Тепер визначимо цінові межі, якими необхідно керуватись при встановленні ціни на потенційний товар, яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів. Аналіз проводився експертним методом і його результати зображено в таблиці 4.20.

Таблиця 4.20 – Визначення меж встановлення цін

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
3000-5000 $/рік	5000-6000 $/рік	12000-50000 $/рік	Нижня межа – 2000 $/рік, верхня межа - 3000 $/рік

Надалі визначимо оптимальну систему збуту, в межах якого приймається рішення. Дану систему зображено в таблиці 4.21.

Таблиця 4.21 – Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Клієнт виплачує гроші на рік, тоді до нього приходить спеціаліст із впровадження інформаційних систем і встановлює ПЗ на комп’ютер клієнта.	Встановити програмний продукт на комп’ютери клієнтів.	Один посередник – спеціаліст по впровадженню інформаційних систем.	Канал збуту одного рівня.

Тепер розробимо концепцію маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів. Дану концепцію зображено в таблиці 4.22.

Таблиця 4.22 – Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуютьс я цільові клієнти	Ключові позиції, обрані для позиціонува-ння	Завдання рекламного повідомлення	Концепція рекламно го зверненн я
Клієнт намагається знайти нові методи аутентифікаці ї до контенту в сайтах.	Мережа Інтернет, соціальні мережі, відео-портали.	Дешевизна, невибагливіст ь до апаратних ресурсів, іноваційність ПЗ, швидкодія.	Продемонструва -ти іноваційність, дешевизну початкову та експлуатації, якість.	Показати можливіс ть за невелику ціну зацікавит и користув ачів свого вебсайту/ сервісу.

4.6 Висновки до розділу

В даному розділі було повністю виконано перший етап розроблення стартап-проєкту, а саме, виконано маркетинговий аналіз стартап-проєкту.

За допомогою нього можна сказати, що існує можливість ринкової комерціалізації проєкту, адже на ринку програм систем біометрії наявний попит на системи голосової біометрії, до того ж рентабельність роботи є досить високою.

З огляду на потенційну групу клієнтів, а саме, малий бізнес, що має вебсайт з приватним контентом для своїх клієнтів, та іноваційність технології є великі перспективи впровадження даного програмного забезпечення.

ВИСНОВКИ

У даній магістерській дисертації було описано задачу виділення людського голосу з зашумленого звукозапису. Були розглянуті методи вирішення даної задачі як класичні, наприклад, фільтр Вінера, так і на базі машинного навчання. В якості останніх були розглянуті провідні архітектури, а саме SEGAN, WaveNet denoising та EHNet. Був проведений порівняльний аналіз цих архітектур, а також класичних методів, який показав, що серед цих архітектур найточнішою є SEGAN, а найшвидшою є WaveNet denoising.

Доведена актуальність досліджень у обраній області, а саме відсутністю рішення для платформ відмінних від Windows та GPU від Nvidia.

Був проведений детальний опис принципів роботи кожної з архітектур, а також оглянуті необхідні теоретичні основи задля розуміння їх структури та принципу роботи.

Була запропонована та реалізована модифікація мережі глибокого навчання SEGAN, яка продемонструвала дещо гірші результати але помітно швидшу роботу, що робить можливим її використання у реальному часі.

Була запропонована архітектура системи виділення голосу на базі запропонованої модифікації.

Для подальших досліджень виділено наступні напрями:

1. Покращення оцінки якості результатів обробки;
2. Створення нового набору даних українською мовою, та тестування якості навчання та роботи на них;
3. Покращення якості виділення голосу, оскільки вона все ж помітно гірша за результати приватної технології NVIDIA RTX Voice.

ПЕРЕЛІК ПОСИЛАНЬ

1. D. Silver et al. Mastering the Game of Go without Human Knowledge. 2016.

2. I. Goodfellow et al. Generative Adversarial Networks. 2014.

3. NVIDIA RTX Voice: Setup Guide. URL: https://www.nvidia.com/en-us/geforce/guides/nvidia-rtx-voice-setup-guide/#RTX-Voice (дата звернення: 22.04.2020)

4. L. Yu et al. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. 2016.

5. S. Pascual, A. Bonafonte, and J. SerraÃÄ. SEGAN: Speech Enhancement Generative Adversarial Network. 2017.

6. Avinash Hindupur. The GAN Zoo. 2019. URL: https : / /github.com/hindupuravinash/the-gan-zoo. (дата звернення: 22.04.2020)

7. F. G. Germain, Q. Chen, and V. Koltun. Speech Denoising with Deep Feature Losses. 2018.

8. A. Kumar and D. Florencio. Speech Enhancement In Multiple-Noise Conditions using Deep Neural Networks. 2016.

9. S. Pascual et al. Whispered-to-voiced Alaryngeal Speech Conversion with Generative Adversarial Networks. 2018.

10. W. Han et al. Speech Enhancement Based on Improved Deep Neural Networkswith MMSE Pretreatment Features. 2016.

11. H. W. Lin, M. Tegmark, and D. Rolnik. *Why does deep and cheap learning work so well?* 2017.

12. J. Benesty et al. Speech Enhancement. A Signal Subspace Perspective pp. 2-3. 2014.

13. Christopher M. Bishop. Pattern Recognition and Machine Learning pp. 147-152. 2006.

14. A. W. Rix et al. Perceptual Evaluation of Speech Quality (PESQ) - A New Method for Speech Quality Assessment of Telephone Networks and Codecs, IEEE. 2002.

15. C. H. Taal et al. A Short-Time Objective Intelligibility Measure for Time-Frequency Weighted Noisy Speech. 2010.

16. Y. Hu and P. C. Loizou. Evaluation of Objective Quality Measures for Speech Enhancement, Senior Member, IEEE. 2008.

17. Saeed V. Vaseghi. Advanced Digital Signal Processing and Noise Reduction, Second Edition. 2000.

18. Daniel Faggella. What is Machine Learning. 2019. URL: https: //emerj.com/ai-glossary-terms/what-is-machine- learning. (дата звернення: 24.04.2020)

19. I. Goodfellow, Y. Bengio, and A. Courville. Deep Learning, p. 100. 2017.

20. Christopher M. Bishop. Pattern Recognition and Machine Learning pp. 3-4. 2006.

21. Christopher M. Bishop. Pattern Recognition and Machine Learning pp. 192-196. 2006.

22. Deepak Battini. Solving XOR Problem using neural network - C#. 2019. URL: https://www.tech-quantum.com/solving- xor-problem-using-neural-network-c/. (дата звернення: 24.04.2020)

23. Stanford CS Class CS231n. Convolutional Neural Networks for Visual Recognition. 2019.

24. K. He et al. Delving Deep into Recitifiers: Surpassing Human-Level Performance on ImageNet Classification. 2015.

25. G. Chen. A Gentle Tutorial of Recurrent Neural Network with Error Backpropagation. 2018.

26. R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. 2013.

27. Justin Bayer. Learning Sequence Representation, pp. 11-15. 2015.

28. M. Schuster and K. K. Paliwal. Bidirectional Recurrent Neural Net works. 1997.

29. K. He et al. Deep Residual Learning for Image Recognition. 2015.

30. P. Bonnier et al. Deep Signatures. 2019.

31. S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. 2015.

32. T. Salimans et al. Improved Techniques for Training GANs. 2016.

33. Ian Goodfellow. NIPS 2016 Tutorial: Generative Adversarial Net works. 2017.

34. A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel Recurrent Neural Networks. 2016.

35. A. van den Oord et al. Conditional Image Generation with Pixel-CNN Decoders. 2016.

36. A. van den Oord et al. WaveNet: A Generative Model for Raw Audio. 2016.

37. I. Goodfellow. NIPS 2016 Tutorial: Generative Adversarial Networks. 2016.

38. X. Mao et al. Least Squares Generative Adversarial Networks. 2017.

39. M. Mirza and S. Osindero. Conditional Generative Adversarial Nets. 2014.

40. P. Isola et al. Image-to-Image Translation with Conditional Adversarial Networks. 2018.

41. Cassia Valentini-Botinhao. *Noisy speech database for training speech enhancement algorithms and TTS models*. 2017. URL: https:// datashare.is.ed.ac.uk/handle/10283/2791. (дата звернення: 24.04.2020)

42. C. Valentini-Botinho et al. *Speech Enhancement for a Noise-Robust Text-to-Speech Synthesis System using Deep Recurrent Neural Net- works*. 2016.

43. J. Thiemann, N. Ito, and E. Vincent. *DEMAND: a collection of multi-channel recordings of acoustic noise in diverse environments*. 2013. URL: https ://zenodo.org/record/1227121#.XObEqlUzZhE. (дата звернення: 24.04.2020)

44. D. Rethage, J. Pons, and X. Serra. *A Wavenet for Speech Denoising*. 2018.

45. H. Zhao et al. *Convolutional-Recurrent Neural Networks for Speech Enhancement*. 2018.

46. T. E. Wang et al. *Towards Robust Deep Neural Networks*. 2018.

47. K. Sun, Z. Zhu, and Z. Lin. *Enhancing the Robustness of Deep Neu- ral Networks by Boundary Conditional GAN*. 2019.

48. S. Pascual, A. Bonafonte, and J. Serrà. *Implementation of SEGAN: Speech Enhancement Generative Adversarial Network*. 2019. URL: https://github.com/santi-pdp/segan. (дата звернення: 24.04.2020)

49. D. Rethage, J. Pons, and X. Serra. *Implementation of A Wavenet for Speech Denoising*. 2019. URL: https : / / github . com / drethage/speech-denoising-wavenet. (дата звернення: 24.04.2020)

50. ododoyo. *Partial implementation of Convolutional-Recurrent Neural Networks for Speech Enhancement*. 2019. URL: https://github. com/ododoyo/EHNet. (дата звернення: 27.04.2020)

51. Jingdong Li. *A python package for calculating the PESQ*. 2019. URL: https://github.com/vBaiCai/python-pesq. (дата звернення: 27.04.2020)

52. Cees Taal. *STOI – Short-Time Objective Intelligibility Measure*. 2019. URL: http://www.ceestaal.nl/code/. (дата звернення: 27.04.2020)

53. Oscar Xing Luo. *Human evaluation spreadsheet*. 2019. URL: https://drive.google.com/file/d/1XnguF_eHwKyogMQotkkxXSSae_w3zZp/view?usp=sharing. (дата звернення: 29.04.2020)

54. Esfandiar Zavarehei. *Wiener Filter*. 2019. URL: https : / / mathworks . com / matlabcentral / fileexchange / 7673 - wiener-filter. (дата звернення: i9.04.2020)