НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

«На правах рукопису»
УДК 004.043

«До захисту допущено»

Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
“___”_____2018 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 121 «Інженерія програмного забезпечення»

на тему: Система розпізнавання звуків живої природи за допомогою нейронної мережі

Виконав: студент другого курсу, групи ІТ-74мп
(шифр групи)

Поляков Анатолій Сергійович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник доцент, к.т.н., доц. Корнага Я.І. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

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

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

Київ – 2018 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

Рівень вищої освіти – другий (магістерський)

Спеціальність 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ І.Р. Пархомей
(підпис)

«___»_____2018 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Полякову Анатолію Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема дисертації «Система розпізнавання звуків живої природи за допомогою нейронної мережі», _____
науковий керівник дисертації Корнага Ярослав Ігорович, к.т.н., доц._____
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «___»_____ 2018 р. №_____

2. Термін подання студентом дисертації _____

3. Об'єкт дослідження – процес автоматизованого управління обробки звуків.

4. Предмет дослідження – моделі та методи інформаційних технологій для обробки звуків.

5. Перелік завдань, які потрібно розробити – аналіз проблеми та існуючих рішень; аналіз і реалізація методу; розробка додатку; дослідження ефективності розробленого додатку.

6. Орієнтовний перелік ілюстративного матеріалу – три плакати та три креслення

7. Орієнтовний перелік публікацій – одна публікація

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз предметної області	13.09.2018 р.	
2	Постановка задачі	15.09.2018 р.	
3	Аналіз інформаційного забезпечення	20.09.2018 р.	
5	Аналіз алгоритмічного забезпечення	25.09.2018 р.	
6	Розробка алгоритмічного забезпечення	15.10.2018 р.	
7	Розробка програмного забезпечення	01.11.2018 р.	
8	Маркетинговий аналіз стартап-проекту	10.11.2018 р.	
9	Висновки	15.11.2018 р.	

Студент _____ А.С. Поляков__
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Я.І. Корнага___
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

У роботі розглянуто проблему в області автоматизованого розпізнавання звуків живої природи, показано основні особливості існуючих рішень та додатків, їх переваги та недоліки.

При розпізнаванні різних звуків, їх кількості та висоти, на запис попадають звуки які розпізнавати не потрібно, якщо виконувати всі ці операції без автоматизованої системи потрібно використовувати експертів даної сфері. Вони можуть виконати всі задачі, проте на це піде дуже багато часу та сил які можна було зберегти для інших задач.

Визначено завдання для системи розпізнавання звуків живої природи за допомогою нейронної мережі, відібрано нейронну мережу та спосіб її навчання, який найбільш підходять для даної задачі. Описано структуру нейронної мережі та проведено експерименти по навчанню та її роботі.

Ця система надає вам автоматизоване розпізнавання звуків та їх виділення без використання важких для навчання додатків та програм на персональні комп'ютери. Дозволяє зменшити витрати часу та сил для розпізнавання звуків живої природи.

Ключові слова: нейронна мережа, спектрограма, навчання, функція, алгоритм.

Розмір пояснювальної записки – 86 аркушів, містить 7 ілюстрацій, 26 таблиць, 6 додатків.

ABSTRACT

The paper considers the problem in the field of automated sound recognition of living nature, shows the main features of existing solutions and applications, their advantages and disadvantages.

When recognizing different sounds, their number and height, recording does not get the sounds that need to be recognized, if you perform all these operations without an automated system, you need to use experts in the field. They can accomplish all the tasks, but this will take a lot of time and effort that could be saved for other tasks.

The tasks for the for the system of recognition of sounds of living nature with the help of a neural network are determined, and the neural network and the method of training chosen which are most suitable for this task are chosen. The structure of the neural network is described and experiments on training and work.

This system provides you with automated sound recognition and selection, without the use of hard-to-learn apps and applications on personal computers. Allows you to reduce the time and effort for the recognition of living nature sounds.

Keywords: neural network, spectrograph, training, function, algorithm.

Explanatory note size – 86 pages, contains 7 illustrations, 26 tables, 6 applications.

Пояснювальна записка
до магістерської дисертації

на тему: Система розпізнавання звуків живої природи за допомогою нейронної мережі

Київ – 2018 року

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – База даних

СУБД – Система управління базами даних

HTML – HyperText Markup Language

ПЗ – Програмне забезпечення

API – Application programming interface

SQL – Structured Query Language

UML – Unified Modeling Language

MAP – Mean Average Precision

AUROC – Area Under the Receiver Operating Characteristic Curve

RoI – Агрегування областей інтересу

ВСТУП

Важливою проблемою екології в усьому світі, яка полягає в вивченні взаємодій між організмами та їх середовищем, є контроль за тваринами, особливо з постійною загрозою зміни клімату.

Використання акустики для моніторингу та класифікації тварин у їхніх природних умовах останнім часом викликало великий інтерес. Класифікація видів тварин на основі записаних звукових даних є, наприклад, корисною при моніторингу поведінки розмноження, біорізноманіття та динаміки населення. Тварини є особливо корисним екологічним показником, оскільки вони швидко реагують на зміни у їх середовищі.

Якщо експерти будуть класифікувати та слідкувати за тваринами вручну, це принесе досить малі результати так як обсяг інформації дуже великий. Тому для рішення цієї проблеми люди почали використовувати штучні нейронні мережі. За допомогою навченої нейронної мережі великі обсяги інформації можна обробити дуже швидко. Но для початку нейронну мережу треба навчити, тому як без достатньої кількості матеріалу для навчання результати можуть бути зовсім не вірні. З цього і випливають сучасні питання цієї сфери – яку нейронну мережу використовувати та як подавати інформацію для нейронної мережі.

Розділ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

1.1. Об’єкт та предмет дослідження.

Область інтересу до екології - це моніторинг тварин, щоб краще зрозуміти їх поведінку, біорізноманіття та динаміку населення. Акустично активні тварини можуть бути автоматично класифіковані за їх звуками, тварина є особливо корисним екологічним показником, оскільки швидко реагує на зміни у своєму середовищі. При ідентифікації різноманітних тварин власноруч у людей зайнятих у сфері екології не вистачить часу та можливостей, щоб аналізувати ефективно треба використовувати нейронні мережі спеціально навчені для цього. У них є багато можливостей для ідентифікації. Якщо нейрона мережа не правильно ідентифікує тварину, то метод роботи нейронної мережи не є ідеальним, якщо оброблювати інформацію в великих обсягах це буде помітно експертам. Тому нам потрібна нейронна мережа яка зможе впоратися за даною задачею.

1.1.1 Огляд існуючих рішень

Змагання по ідентифікації видів, такі як BirdCLEF, the Neural Information Processing Scaled for Bioacoustics (NIPS4B) 2013 р., Та версія програми Challenges for Classification of Birds, та the Machine Learning for Signal Processing (MLSP) 2013, були проведені з метою створення та оцінки таких автоматичні класифікатори з записів пташиних пісень, взятих з поля. Доведено, що багатообіцяюча класифікаційна техніка - це згорткові нейронні мережі. У 9-му щорічному конкурсі MLSP автори роблять висновок зі словами: "згорткові нейронні мережі досягають відмінних результатів без широкої технічної характеристики, тому подальші дослідження таких методів є обґрунтованими", а переможцями BirdCLEF 2016 було використано збіжну нейронну мережу (ЗНМ), яка була навчена на розширених даних спектрограм, обчислені з аудіо файлів пісні птаху.

1.1.2. Змагання по ідентифікації

Кілька видів змагань по класифікації птахів з тісно пов'язаними, але мали різні описи завдань, які були проведені протягом останніх кількох років. Інтерес та участь у цих змаганнях були високими, що вказує на те, що це важливі проблеми та що їх необхідно вирішити.

Змагання як правило, передбачають, які види присутні в наборі записів із прихованими мітками, які називаються тестовим набором, а також для подання прогнозованого виду для кожної точки тестування для оцінки з нанесенням ілюстрації на землю.

Опис завдання може відрізнятися від прогнозування лише наявності або відсутності птахів у записі для прогнозування всіх активно співаючих видів птахів. Тобто проблеми мають різні ступені складності. Ми розглянемо деяких змагань.

1.2. MLSP 2013

The IEEE International Workshop on Machine Learning for Signal Processing (MLSP) оголосила про змагання категорії ідентифікації видів птахів у 2013 році. Змагання було визначити всі акустично активні види птахів у кожному аудіо запису тестового набору з 19 різними видами птахів. Тобто завдання було розглядаються як одноразові проблеми з кількома мітками.

Набір даних складався з 645 десятисекундних аудіозаписів, які були розділені на тренувальний набір (50%) та тестовий набір (50%). Мітки птахів для кожної запису в наборі тренувань були оприлюднені, але етикетки для кожного запису в тестовому наборі були збережені в секреті.

У програмі брали участь 79 команд, а 8 з 10 команд топ-класу надали двосторінкове резюме своїх методів, що свідчить про сильний інтерес до проблеми. Команда-переможець використовувала random forest (RF) класифікатор, де функції здобули з вводу з використанням відповідності шаблону. Шаблони обчислювали за допомогою спеціальної методики сегментації часової частоти, де кожний кожен сегмент зберігався як шаблон і

обчислювався лише з 81 аудіозаписів, які були помічені одним класом звуку. Потім розраховували спектрограму кожного запису. Функції були витягнуті для кожної спектрограми шляхом обчислення нормалізованої крос-кореляційної карти між спектрограммою та кожним шаблоном, подібність між шаблоном і спектрограмою була потім оцінена на максимальному значенні нормалізованої кросової кореляційної карти з використанням методу відповідності шаблону, що означає, що кожна спектрограма дає вектор функції з тією ж довжиною, що і число шаблонів, які було витягнуто з 81 записів мічених сигналів. Метод також використовував функції, доступні в якості базової лінії у виклику, такі як гістограма сегментів, які були додані до вектору функцій і використовувались як вхідні дані для класифікатора.

Багато хто з команд розробляли специфічні завдання; проте, одна команда, яка посіла четверте місце, використовувала необроблені дані спектрограми для навчання згорткової нейронної мережі. Про це заявив Бріггс. Що подальше дослідження використання згортаних нейронних мереж у цій проблемній області має підставу.

1.2.1. NIPS4B 2013

У Neural Information Processing Scaled for Bioacoustics (NIPS4B) опис завдання був подібний до опису завдання MLSP 2013. Учасникам було запропоновано ідентифікувати всіх активно співаючих птахів у кожному з тестових файлів. Однак число можливих видів становило 87 замість 19, а записи могли змінюватися в довжину (від 0,5 до 5,5 с)

Переможець рішення цього виклику використовував подібний підхід. Головна відмінність полягає в тому, що для кожного аудіофайлу витягуються додаткові функції. На додаток до функцій, отриманих шляхом оцінки відповідності шаблону до максимального значення нормалізованої крос-кореляційної карти, векторний компонент додатково доповнюється статистикою файлів та сегментів.

1.2.2. BirdCLEF 2016

Завдання BirdCLEF використовує дуже великий набір даних із записом птахів. Дані взято з бази даних пташиних пісень xeno-canto , яка є веб-сервісом, де ентузіасти птиці можуть завантажувати та ділитися записами птахів.

Набір даних BirdCLEF є підмножиною бази даних Xeno-Canto і складається з 999 різних видів. Набір даних складається приблизно з 33 200 записів, які були нормалізовані до 44-бітових 16-бітних монофонічних (правого каналу) аудіофайлів.

Завдання полягало у визначенні видів птахів, присутніх у кожному записуванні. Учасникам було запропоновано надати список найуспішніших птахів для записів прихованого тестового набору. Набір даних поділений на 1/3 даних тесту та 2/3 навчальних даних, а показник, який використовується для оцінки ефективності класифікації тестів, що поставляються командами-учасниками, є середньою точністю середньої точності (MAP) над усіма записами в тестовому наборі .

Найкращий метод використав згорткову нейронну мережу, де вхід до мережі був сегментами спектрограми, обчисленого з звукових файлів. Звукові файли попередньо обробляються шляхом вилучення двох звукових класів із кожного аудіофайлу: шум та сигнал (пташиний спів), які поділяються на однаково довгі звукові сегменти приблизно на 3 секунди. Сегменти сигналу являють собою фактичний пташиний спів, кожен з яких має пов'язані з ним види птахів. Потім проби, показані нейронній мережі, завантажуються та розширюються випадковим чином. Кожен сегмент сигналу додатково поєднується з іншим сегментом сигналу одного класу, вибраним випадковим чином, а також з трьома випадковими вибірками сегментів шуму. Зразки потім додатково збільшуються випадковим зрушенням у часовій області та невеликим випадковим зсувом (5%) у частотній області.

1.3. Постановка задачі

Метою роботи є Ідентифікація тварин може здійснюватися вручну експертами домену; однак, з зростаючим обсягом даних, це швидко стає нудним і трудомістким процесом. Тому необхідні автоматичні інструменти, які можуть допомогти у цьому процесі.

Проблеми, які розглядаються, можна розділити на дві основні частини: (i) класифікацію сигналів і (ii) вилучення ознак. Ці завдання описані більш детально, і кожне завдання сформульовано як задачу, яку робота спрямована вирішити.

Для досягнення мети необхідно вирішити наступні задачі:

1. аналіз нейронних мереж, методів роботи з інформацією;

2. розробка нейронної мережі на потрібній мові програмування, з використанням потрібної інформації трансформованої для можливості нейронної мережі навчатися за використанням цієї інформації ;

3. реалізація нейронної мережі яка ідентифікує живу природу по голосу.

1.3.1. Переваги системи ідентифікації звуків живої природи за допомогою нейронної мережі

Переваги системи ідентифікації звуків живої природи за допомогою нейронної мережі котрі можуть виділити її:

Швидкість видавання правильного результату. Власноруч експерти для ідентифікації тварин втрачають купу часу. За допомогою системи ідентифікації звуків живої природи за допомогою нейронної мережі цей час можна зберегти на інші потрібні речі.

Збирання статистики кількості ідентифікованих тварин, для того щоб можна було слідкувати за екологічною популяцією.

Використання нейронної мережі для якої є вже величезна база інформації для навчання. За допомогою цього шанс негативного результату дуже не великий.

Висновки до розділу

Було проведено аналіз вже існуючих систем по ідентифікації тварин за допомогою нейронної мережі. Ми розглянули три програми, в яких у програмі MLSP 2013 вирішальним рішенням був random forest, навчені ймовірності, отримані шляхом підбору шаблонів специфічних для видових спектрограм. Переможець рішення NIPS4B 2013 використовував ці результати як вихідну точку, але ввів додатковий набір функцій, статистично виведених з аудіофайлів. Ласек також використовував подібний метод, щоб виграти BirdCLEF 2015. Однак під час виклику BirdCLEF 2016 було показано, що метод переможців був з використанням згорткової нейронної мережі, де вхід до мережі був сегментами спектрограми, обчисленого з звукових файлів. Звукові файли попередньо оброблявся шляхом вилучення двох звукових класів із кожного аудіофайлу. Ці сигнали були приблизно однаково довгими. Після цієї перемоги було сказано що згорткові нейронні мережі, навчені спектральним даними, обчисленими із звукозаписів, можуть перевершити інші сучасні системи.

Розділ 2. АНАЛІЗ НЕЙРОННИХ МЕРЕЖ, МЕТОДІВ РОБОТИ ІЗ СИГНАЛАМИ ТА ВИБІР ІНСТРУМЕНТІВ РОЗРОБКИ

2.1. Нейронні мережі

2.1.1. Нейронна мережа

Математична модель, яка намагається імітувати структуру та функціональні можливості біологічних нейронних мереж. Основним будівельним блоком кожної штучної нейронної мережі є штучний нейрон, тобто проста математична модель (функція). Така модель має три прості набори правил: множення, підсумовування та активація. На вході штучного нейрона входи зважуються, що означає, що кожне вхідне значення множиться з індивідуальним вагою. У середній частині штучного нейрона - сумарна функція, яка об'єднує всі зважені входи та зміщення. На виході з штучного нейрона сума попередньо зважених входів і упереджень відбувається через функцію активації, яка також називається передавальною функцією.

Хоча робочі принципи та простий набір правил штучного нейрону виглядають як нічого особливого, повний потенціал та потужність розрахунку цих моделей виникають, коли ми починаємо з'єднувати їх у штучних нейронних мережах. Ці штучні нейронні мережі використовують простий факт, що складність може вирости з декількох основних і простих правил.

Для повного збору переваг математичної складності, які можуть бути досягнуті через взаємозв'язок окремих штучних нейронів, а не просто перетворення системи в складні та некеровані, ми взагалі не взаємодіємо з цими штучними нейронами випадковим чином. У минулому дослідники виробили кілька "стандартизованих" топографій штучних нейронних мереж. Ці попередньо визначені топографії допоможуть нам простіше, швидше і ефективніше вирішити проблеми. Різні типи штучних топографічних нейронних мереж підходять для вирішення різних типів завдань. Визначивши тип заданої задачі, необхідно визначити топологію штучної нейронної мережі,

яку ми збираємося використовувати, а потім точно налаштувати її. Нам потрібно точно настроїти саму топологію та її параметри.

Точна топологія штучної нейронної мережі не означає, що ми можемо почати використовувати нашу штучну нейронну мережу, це лише попередня умова. Перш ніж ми зможемо використовувати нашу штучну нейронну мережу, ми повинні навчити її вирішенню типу заданої проблеми. Подібно до того, як біологічні нейронні мережі можуть вивчати свою поведінку / відповіді на основі вхідних даних, які вони отримують від свого середовища, штучні нейронні мережі можуть робити те ж саме. Існує три основні навчальні парадигми: контрольоване навчання, безконтрольне навчання та навчання підкріплення. Ми вибираємо навчальну парадигму подібно, як ми обрали топографію штучної нейронної мережі - на основі проблеми, яку ми намагаємося вирішити. Незважаючи на те, що навчальні парадигми відрізняються за своїми принципами, у них все є спільним; на основі "навчальних даних" та "правил навчання" (вибрана вартість функції) штучна нейронна мережа намагається досягти правильної вихідної відповіді відповідно до вхідних сигналів.

Вибравши топологію штучної нейронної мережі, точне налаштування топології і, коли штучна нейронна мережа вивчить правильну поведінку, ми можемо почати використовувати її для вирішення заданої проблеми. Штучні нейронні мережі існують вже протягом деякого часу, і ми можемо знайти їх у таких сферах, як контроль процесу, хімія, ігри, радіолокаційні системи, автомобільна промисловість, космічна промисловість, астрономія, генетика, банківська справа, виявлення шахрайства тощо. таких проблем, як наближення функції, регресійний аналіз, прогнозування часових рядів, класифікація, розпізнавання образів, прийняття рішень, обробка даних, фільтрація, кластеризація тощо, називаючи кілька.

2.1.2. штучний нейрон

Штучний нейрон є основним будівельним блоком кожної штучної нейронної мережі. Його конструкція та функціональність походять від спостереження біологічного нейрона, який є основним будівельним блоком біологічних нейронних мереж (систем), який включає головний мозок, спинний мозок та периферичні ганглії. Подібності в дизайні та функціональні можливості можна побачити в тому випадку, коли ліва частина фігури являє собою біологічний нейрон із його сомою, дендритами та аксоном, а правою частиною малюнка являє собою штучний нейрон з його входами, вагами, передавальною функцією, зміщенням і виходи

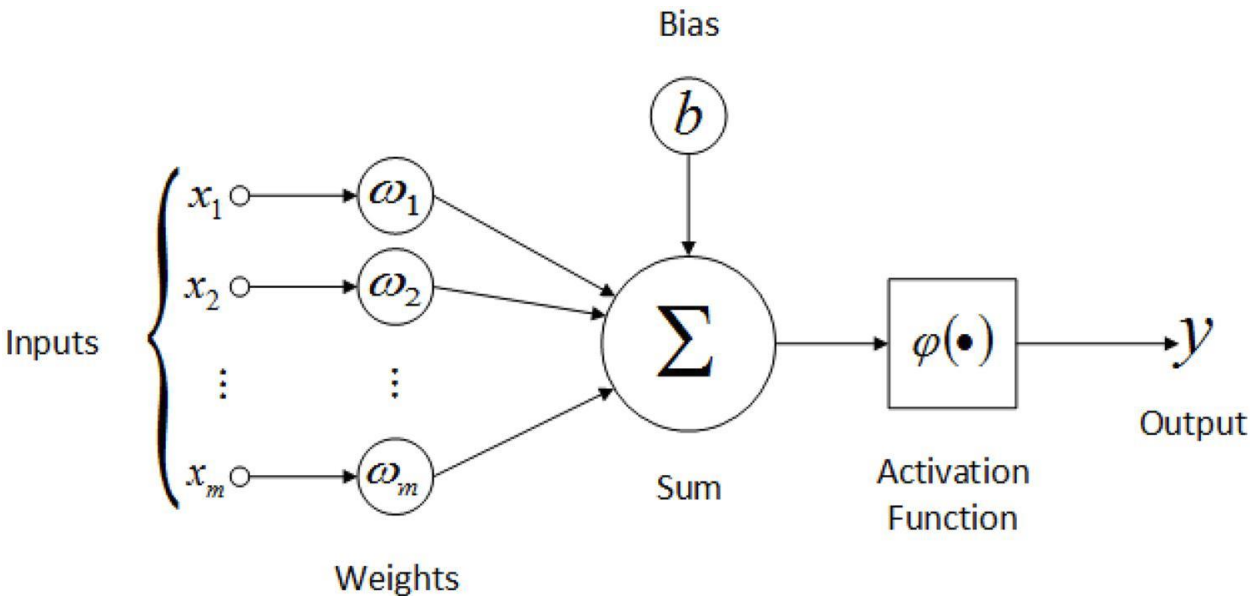


Рис. 2.1. Складові частини штучного нейрону

У випадку біологічного нейрона інформація потрапляє в нейрон через дендрит, то сома обробляє інформацію і передає її через аксон. У випадку штучного нейрона інформація надходить у тіло штучного нейрона за допомогою вхідних даних, що зважуються (кожен вхід може бути індивідуально помножений з вагою). Тіло штучного нейрона потім підсумовує зважені входи, зміщення та "обробляє" суму з передавальною функцією. В кінці штучний нейрон передає оброблену інформацію за допомогою виводу.

Перевага простоти моделі штучного нейрона видно в її математичному описі нижче:

$$y(k) = F\left(\sum_{i=0}^{m} w_i(k) \cdot x_i(k) + b \right),$$

Де:

- $x_i(k)$ вводить значення в дискретний час k, де і йде від 0 до m,
- $w_i(k)$ це значення ваги в дискретному часі k де і йде від 0 до m,
- b - упередження,
- F - функція передачі,
- $y_i(k)$ - вихідне значення в дискретному режимі часу k.

Як видно з моделі штучного нейрона та її рівняння, основною невідомими змінною нашої моделі є його передавальна функція. Функція перенесення визначає властивості штучного нейрона і може бути будь-якою математичною функцією. Ми вибираємо його на основі проблеми, яку потрібно вирішити штучному нейрону (штучній нейронній мережі), і в більшості випадків ми вибираємо його з наступного набору функцій: функція кроку, лінійна функція та нелінійна (сигмоїдна) функція.

Крокова функція - це двійкова функція, яка має лише два можливі вихідні значення (наприклад, нуль та одне). Це означає, що якщо значення вхідного значення відповідає певному порогу, вихідне значення призводить до одного значення, а якщо конкретний поріг не відповідає, то призводить до різного вихідного значення. Ситуацію можна описати за допомогою рівняння

$$y = \begin{cases} 1 & if \;\; w_i x_i \geq threshold \\ 0 & if \;\; w_i x_i < threshold \end{cases},$$

Коли цей тип передавальної функції використовується в штучному нейроні, ми називаємо цей штучний нейронний персептрон. Персептрон використовується для розв'язання задач класифікації, і тому він найчастіше можна знайти в останньому шарі штучних нейронних мереж. У випадку

лінійної передавальної функції, штучний нейрон робить просте лінійне перетворення над сумою зважених входів та зміщення. Такий штучний нейрон на відміну від персептрон найчастіше використовується в вхідному шарі штучних нейронних мереж. Коли ми використовуємо нелінійну функцію, найчастіше використовується сигмоїдна функція. Сигмоїдна функція має легко обчислений похідний, який може бути важливим при підрахунку вагових оновлень у штучній нейронній мережі.

2.1.3. Штучні нейронні мережі

Якщо об'єднати два або більше штучних нейронів, то ми отримаємо штучну нейронну мережу. Один штучний нейрон має нульові шанси у вирішенні реальних проблем, в той самий час як штучні нейронні мережі можуть вирішити реальні проблеми. Фактично штучні нейронні мережі здатні вирішувати складні реальні проблеми шляхом обробки інформації в їх основних будівельних блоках (штучних нейронах) в нелінійному, розподіленому, паралельному та місцевому вигляді

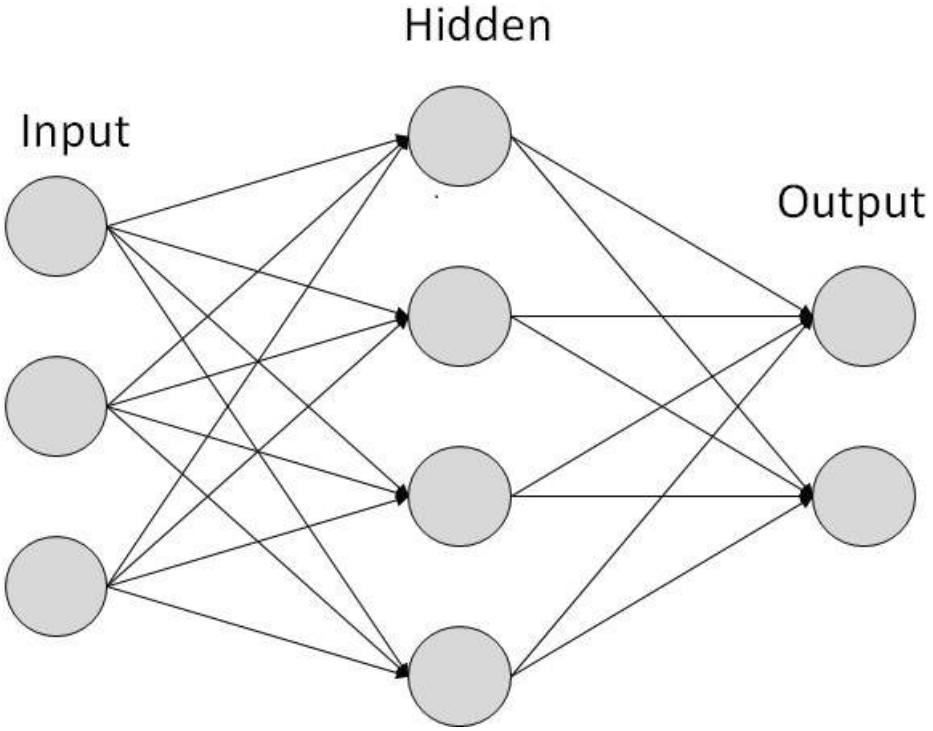


Рис. 2.2. Штучна нейрона мережа

Те, як окремі штучні нейрони взаємопов'язані, називаються топологією, архітектурою або графіком штучної нейронної мережі. Той факт, що взаємозв'язок може бути здійснено багатьма способами, призводить до численних можливих топологій, які поділяються на два основних класи. На рисунку 4 показані ці дві топології; ліва частина фігури являє собою просту податкову топологію (ациклічний граф), в якій інформація надходить від входів до виходів лише в одному напрямку, а права частина малюнка являє собою просту рекурентну топологію (напівкремний граф), де частина інформації надходить не тільки в одне напрямок від введення до виходу, але також у зворотному напрямку. При спостереженні слід сказати, що для полегшення обробки та математичного опису штучної нейронної мережі ми об'єднуємо окремі нейрони у шарах. ми можемо бачити вхідний, прихований і вихідний рівень.

Коли ми обираємо і будуємо топологію нашої штучної нейронної мережі, ми лише закінчили половину завдання, перш ніж ми зможемо використовувати цю штучну нейронну мережу для вирішення заданої проблеми. Подібно до того, як біологічні нейронні мережі мають вивчати відповідні відповіді на дані вхідних даних з навколишнього середовища, штучні нейронні мережі мають робити те ж саме. Таким чином, наступним кроком є вивчення належної реакції штучної нейронної мережі, і це може бути досягнуто завдяки навчанню (контрольоване, непідконтрольне або підкріплене навчання). Незалежно від того, який метод ми використовуємо, завдання навчання полягає в тому, щоб встановити значення ваги та упередження на основі даних навчання, щоб мінімізувати вибрану функцію вартості.

2.2. Згорткова нейронна мережа

Згорткова нейронна мережа (ЗНМ) - це нейронна мережа, яка була піонером ле-кун, а пізніше була популяризована Крижевським з впровадженням алекснет. Типова ЗНМ складається з згорткових шарів та

об'єднаних шарів, за якими іде повністю зв'язана нейронна мережа. Згорткові шари та об'єднані шари повинні вивчати, як витягувати відповідні, інваріантні локальні спотворення, функції з введення, і повністю пов'язана нейронна мережа повинна навчитися класифікувати ці функції. Функціями, здобутими згортковими шарами, можуть бути, наприклад, краї об'єктів на зображенні.

2.2.1. Згортковий шар

Згортковий шар приймає зображення як вхід, і створює набір функціональних карт як вихідний. Вхідне зображення може містити кілька каналів (наприклад, RGB), що означає, що згортковий шар вивчає відображення 3D-гучності на інший 3D-гучність. Цей шар складається з декількох ядер згортки, кожен з яких складається з регульованих ваг. Ваги регулюються під час оптимізації, використовуючи стохастичний градієнт. Карти функцій створюються шляхом виконання дискретної згортки між кожним ядром у згортковому шарі та обсягом вводу, що дає одну карту об'єктів для кожного ядра.

Виконання згортки можна розглядати як просунення вікна вздовж вхідного зображення та обчислення точного продукту між ядром і значеннями сусіднього патча або так званого сприйнятного поля на зображенні, яке видно через вікно. Вікно має той самий розмір, що і ядро, і важливо зрозуміти, що ядро має глибину, що простягається через всі канали вхідного зображення.

Формально, нехай w_k^l та b_k^l позначають ваги та терміни зсуву для l-го шару k-го згорткового ядра, і хай $x^l \in \mathrm{R}^{\mathrm{WH}}$ - це 1-й канал або шар вхідного зображення форми (W, H, D). Тоді ми можемо визначити значення функції, $c_{i,j}^k$ у позиції (i, j) отриманої k-й функції карту як:

$$c_{i,j}^k = f(\sum_{l=1}^{D} w_k^l \cdot x_{i,j}^l + b_k^l),$$

де D - глибина вхідного сигналу (кількість каналів), $x_{i,j}^{l}$ -сприйняткове поле, орієнтоване навколо (i, j) l-го шару; f - випрямлена лінійна одиниця (ReLU), яка визначається як: $f(x) = \max(0, x)$

Зверніть увагу, що результуюча кількість карток функцій така ж, як кількість використовуваних ядер, а також, що ваги та упередженість використовуються повторно для обчислення точка продукту для кожного рецептивного поля для кожного ядра та шару, що називається ваговим. Ідея полягає в тому, що якщо ядро корисно для пошуку локальних функцій у положенні (x1, y1) на зображенні, то воно також має бути корисним в іншому положенні (x2, y2).

Таким чином, конфігурація згорткового шару визначається кількістю використаних ядер, шириною та висотою ядра, а також розміром кроку, який визначає відстань між сприймаючими полями. Наприклад, розмір кроку 1 × 2 призведе до слайдів вікна вздовж зображення із розміром кроку по одному пікселю по горизонталі та двома пікселями, що вертикально зроблять ширину отриманої карти функції такою самою, як і вхідного зображення, але висота карта особливостей половини висоти вхідного зображення

Таким чином, вихідний розмір згорткового шару визначається числом використовуваних ядер та розміром кроку. Кількість ядра визначає глибину обсягу виводу, а розмір кроку визначає ширину та висоту вихідного об'єму. Наприклад, якщо вхідний об'єм згорткового шару є зображенням форми (32, 32, 3), кількість використовуваних ядер становить 16, а розмір кроку становить 1 × 2, тоді об'єм виводу буде мати форму (32, 16, 16).

2.2.2. Шар об’єднання

Карти функцій, обчислені згортковим шаром, запускаються через шар згрупування, який призначений для об'єднання семантично подібних функцій і тим самим зменшує розмір функціональних карт. Використовуваний шар згруповання - макс-об'єднання, яке просто обчислює максимальне значення

локальних сприйнятливих полів у кожному з карток властивостей. Розміри полів визначаються величиною кроку, а розмір полів визначається розміром ядра панелі об'єднання. Знову ж таки, нехай $x_{i,j}^{l}$ - сприйняткове поле l-го вхідного шару, центрованого навколо (i, j), тоді значення l-го об'єднаного шару $p_{i,j}^{l}$ можна визначити як: $p_{i,j}^{l} = \max(x_{i,j}^{l})$

де max приймає максимальне значення рецептивного поля.

Наприклад, якщо розмір кроку становить 2 × 2, то ширина та висота об'єднаних карт функцій зменшуються вдвічі, тому що ми просуваємося по 2×2 кварталів кожної карті об'єктів з розміром кроку 2 пікселя горизонтально та 2 пікселями вертикально. Тобто, якщо вхід максового пулу шару з ядром розміром 2 × 2 і шаром розміром 2 × 2 має форму (32, 32, 4), то вихідний буде мати форму (16, 16, 4). Припускається, що вхідний сигнал дорівнює нулю для обробки краю корпусів.

2.2.3. Вхід ЗНМ

Підсумовуючи: заключний внесок згорткової нейронної мережі є логарифмічна спектрограма додаткових точок даних. Розширені точки даних обчислюються з вихідних звукових файлів, застосовуючи наступні кроки:

(i) нормалізувати звукові файли,

(ii) окремі сегменти шуму та сигналу,

(iii) об'єднують сегменти одного класу та шуму для формування унікальної точки даних,

(iv) обчислення логарифмічної спектрограми аудіо сегмента,

(v) знизити 4 низьких і 24 найвищих частотних смуг, а також

(vi) зміна часу та зміна частоти спектрограми.

Результуюча зміщенна логарифмічна спектрограма потім перерозподіляється на 256 × 512 пікселів, щоб мати рівномірну силу двох, що дає вхідний вектор форми (256, 512, 1), який використовується як вхід для нейронної мережі згортки.

2.2.4. Архітектура

Архітектура моделі згорткової нейронної мережі, використовує нормальний вхідний шар, а потім п'ять аналогічних будівельних блоків з різними конфігураціями, де кожен будівельний блок складається з партійної нормалізації, згортки та максимуму пулу шару. І, нарешті, мережа має щільний шар і soft max шар з 40% відсіву на кожному. Випадання використовується для введення шуму до моделі та запобігання перенавчання. Випадання 40% означає, що для кожної епохи існує 40% шанс, що нейрон дезактивується. Вся архітектура мережі зведена в Таблиці. Всі згорткові шари використовують виправлені лінійні одиниці (ReLUs) як функції активації. Два щільні шари на кінці використовують ReLU і активацію softmax відповідно.

Таблиця 2.1. Архітектура мережі використовується в базовому сценарії. Перший стовпець містить тип шару, другий стовпець містить конфігурацію шару, а третій стовпчик містить вихідну форму шару (рядки, стовпці, канали).

Layer (type)	Configuration	Output Shape
InputLayer		(256, 512, 1)
BatchNormalization		(256, 512, 1)
Convolution2D	64 5x5 kernels, 1x2 stride	(256, 256, 64)
MaxPooling2D	2x2 kernel, 2x2 stride	(128, 128, 64)
BatchNormalization		(128, 128, 64)
Convolution2D	64 5x5 kernels, 1x1 stride	(128, 128, 64)
MaxPooling2D	2x2 kernel, 2x2 stride	(64, 64, 64)
BatchNormalization		(64, 64, 64)
Convolution2D	128 5x5 kernels, 1x1 stride	(64, 64, 128)
MaxPooling2D	2x2 kernel, 2x2 stride	(32, 32, 128)
BatchNormalization		(32, 32, 128)
Convolution2D	256 5x5 kernels, 1x1 stride	(32, 32, 256)
MaxPooling2D	2x2 kernel, 2x2 stride	(16, 16, 256)
BatchNormalization		(16, 16, 256)
Convolution2D	256 3x3 kernels, 1x1 stride	(16, 16, 256)
MaxPooling2D	2x2 kernel, 2x2 stride	(8, 8, 256)
BatchNormalization		(8, 8, 256)
Flatten		(16384)
Dropout	dropout 0.4	(16384)
Dense		(1024)
Dropout	dropout 0.4	(1024)
Dense		(999)
Total Params	19,523,883	

2.3. Глибока залишкова нейронна мережа

Глибокі залишкові нейронні мережі були побудованні, і вони використовують "ярлики" для покращення швидкості конвергенції та точності

класифікації дуже глибоких ЗНМ. Глибинні мережі, як правило, утрудняють тренування через зникаючі або вибухові градієнти, які в основному були пом'якшені за допомогою нормалізованої ініціалізації ваги, активації ReLU та проміжних нормалізаційних шарів. Причиною ярликів є те, що, дозволяючи легше поширювати сигнал через мережу, проблема вибуху або зникнення градієнтів може бути зменшена, що, в свою чергу, робить глибоку мережу простішою для тренувань.

Залишкова мережа виробляється з менших будівельних блоків, що називаються залишковими одиницями, які в основному складаються з згортанних і нормалізуючих шарів, де вхід пристрою додатково зливається з виходом останнього складеного шару, щоб імітувати ярлик.

2.3.1. Залишковий блок

Розглянемо два складені згорткові шари в ЗНМ, і припустимо, що вони вивчають відображення $H(x)$, де x є входом до першого шару, а $H(x)$ - вивід другого шару. Гіпотеза полягає в тому, що легше вивчити залишкову функцію $F(x) = H(x) - x$ ніж сам відображення $H(x)$. Тоді відображення $H(x)$ стає $F(x) + x$.

Причина цього полягає в припущенні, що відображення $H(x)$, яке слід вивчати зі складеного шару, зазвичай ближче до ототожнення, ніж від нульового відображення, що означає, що легше вивчити частину відображення, яке це різниця між відображенням тотожності та оптимальним відображенням, ніж сама оптимальне відображення. У крайньому випадку, якщо оптимальне відображення є тотожним відображенням, то легше натиснути $F(x)$ на нуль, ніж дізнатись фактичне відображення.

Вона реалізується в мережі за допомогою ярлика, де вхідний сигнал x зі складеного шару підключається до виводу складеного шару, використовуючи просту аддитивну одиницю. Цей будівельний блок - це те, що ми називаємо залишковим блоком.

У найпростішому випадку, коли розміри виходу залишкової функції та вводу однакові, залишковий блок можна визначити як:

$$y_l = x_l + F(x_l, W_l)$$
$$x_{l+1} = \sigma(y_l) \quad ,$$

де xl та xl + 1 є входом і виходом l-го залишкового одиниці, F - параметризована або вивчена залишкова функція, Wl - параметри складеного шару, а σ - функція активації ReLU.

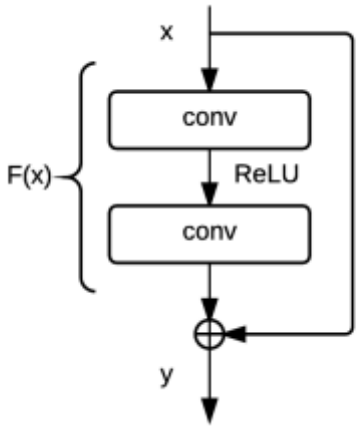


Рис.2.4. Спрощена залишкова одиниця, яка складається з двох складені згорткових шарів, активованих за допомогою ReLU. Вхід першого згорткового шару аддітивно зливається з виходом останнього згорткового шару, який ми називаємо ярликом.

Загалом, залишкові одиниці можуть бути визначені як:

$$y_l = h(x_l) + F(x_l, W_l)$$
$$x_{l+1} = f(y_t) \quad ,$$

де x_l та x_{l+1} є входом і виходом l-ї залишкової одиниці, f - функція активації, F - вивчена залишкова функція, h - це відображення, яке застосовується до даних, що протікають через ярлик, у рівнянні над ним буде тотожним відображенням, а W_l - параметри, пов'язані з l-ю залишковою

одиницею. Якщо розміри $F(x_l, W_l)$ та xl не є однаковими, тоді h потрібно буде налаштувати розмірність x_l за допомогою, наприклад, підвиборки.

Початковий залишковий блок покращується так званою "повною попередньою активацією", що є зміною порядку розташованих шарів блоку. Реалізація полягала в тому, що якщо f є функцією тотожності, то рівняння може бути вставлено у формулу, що надає мережею три чудових властивості: (i) однакове залишкове властивість, яке видно в кожному блоці, також стає присутнім між будь-якими двома одиницями в мережі, (ii) обчислення стають більш ефективними, і (iii) зворотні градієнти, що розповсюджуються, навряд чи зникають навіть з довільно малими вагами.

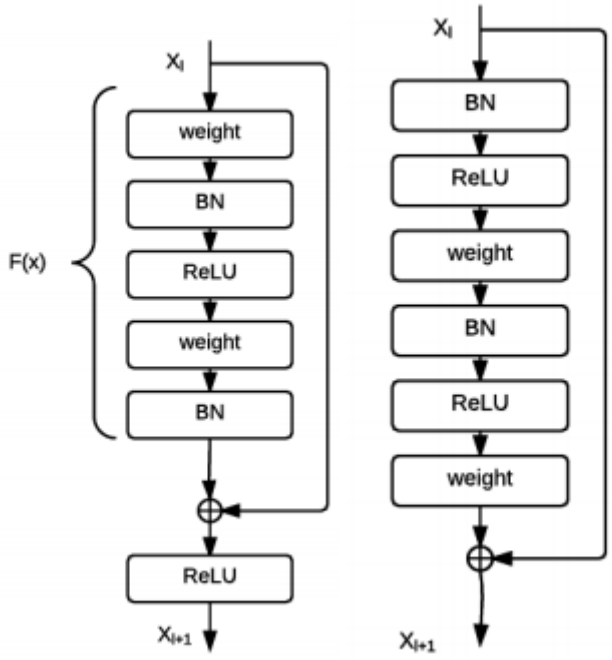


Рис. 2.5. На цьому рисунку показана архітектура вихідного залишкового (ліворуч) та вдосконаленого залишкового блоку, що використовує "повну попередню активацію" (праворуч). Зверніть увагу, що активація більше не виконується після залишкового блоку, а всередині його, отже, "попередньої активації". Це означає, що функція f у рівнянні, яка застосовується до даних, що проходять через ярлик, стає ідентичною функцією.

Зміна порядку початкового залишкового блоку в порівнянні з покращеним може бути видно на рисунку 3.2, де ліва зображення показує оригінальний залишковий блок, а правильне зображення показує поліпшений залишковий блок. Тут BN означає нормалізацію пакету, ReLU означає виправлену лінійну одиницю, яка є типом функції активації, а вагами – згортаними шарами.

Архітектура 18-шарової глибинної залишкової нейронної мережі складається з початкового обертального та максового пулу шару, за яким виділяються вісім основних блоків (з різними конфігураціями), а наприкінці результат об'єднуються, сплюснуто і використовується як вхід до кінцевого щільного шару який має один нейрон для кожного етикетки класу, активований функцією softmax. Шари застосовуються до входу в тому ж порядку, як вказано вище.

Базовий блок складається з двох конвеєрних шарів, кожен з яких передує нормалізації партії та активації ReLU, а вхід блоку дозволяється безпосередньо надходити до виводу складеного шару за допомогою ярлика, реалізованого за допомогою аддитивного шару злиття.

2.4.Класифікація сигналів

Існує кілька способів визначення проблеми класифікації видів птахів. Перш за все, ми повинні встановити, що ми хочемо класифікувати: чи ми зацікавлені у відсутності або присутності птиці у записах? Ми зацікавлені в тому, скільки людей є? Чи потрібно класифікувати фактичні види присутніх птахів? Або це час і початок кожної окремої пісні, яку ми шукаємо?

Враховуючи лише наявність або відсутність птиці в запису, вона ставить проблему з назвою одиночний екземпляр двійковій мітки (ОЕДМ). Тобто для кожного аудіозапису існує два можливих класу, або птах присутній (співає) у запису, чи ні. Це корисно, наприклад, у системі, яка збирає дані у полі. Якщо система може розпізнати, коли птах присутній у вибірці даних, вона може прийняти обгрунтоване рішення про те, чи зберігати цей зразок на диску.

Якщо нас цікавить фактичний вид співаючого птаха, кількість можливих класів збільшується до числа спостережуваних видів птахів, що робить його одиночним екземпляром однієї мітки (OEOM). Існує ще один екземпляр для класифікації, а саме запис, але у всьому світі існує безліч різних видів птахів, що означає, що звукові класи більше не є двозначними. Це корисно, якщо ми хочемо розібратися з даними, зібраними в полі, і відповісти на такі питання, як: Які види птахів присутні? Наскільки різноманітним є популяція птахів?

якщо ми зацікавлені у всіх записах пісні у записі, ми потребуємо можливості класифікувати кілька видів птахів на запис, роблячи мульти-мітки з одним екземпляром (MMZOE) проблемою. Яке є точним формулюванням проблеми, але це також важко вирішити.

ми розглядаємо лише найбільш видимих співочих птахів у кожному записуванні, тобто ми розглядаємо його як одноразову проблему з однією міткою. Однак використані дані міститимуть записи, в яких одночасно співають кілька птахів, що означає, що в цих випадках фоновий вигляд буде розглядатися як шум.

Формально, нехай $X = \{\bar{x}_1,\ldots,\bar{x}_n\}$ - це набір записів пісні пісні або навчальних зразків, де кожен $\bar{x}_i \in X$ асоціюється з основним видом $y_j \in Y$. Нехай $Y = \{y_1,\ldots, y_n\}$ - сукупність всіх видів міток і нехай $f : X \to Y$ - функція, яка відображає навчальни*й* зразок основного виду істини землі. Тоді проблема класифікації OEOM полягає в тому, щоб знайти параметр w, такий, що параметризована функція $f_w : X \to Y$ є гарною оцінкою f. Наскільки ефективною є оцінка f_w, можна визначити шляхом введення функції втрат, де втрата $(f(\bar{x}), f_w(\bar{x}))$ визначається як мала, якщо оцінка $f_w(\bar{x})$ близька до ієрархії землі $f(\bar{x})$, а більший - інакше. Тобто втрата прагне до нуля, якщо

передбачення близьке до істинності землі. Тоді проблема оптимізації може бути визначена як:

$$w = \arg\min_{w} \sum_{\bar{x} \in X} loss(f(\bar{x}), f_w(\bar{x})) .$$

Тобто, ми хочемо знайти параметр w так, щоб загальна втрата для моделі f_w при оцінці на навчальних даних X була настільки мала, наскільки це можливо. Однак пам'ятайте, що оптимізація базується на втраті навчальних даних. Це не гарантує, що модель добре працює над даними, які вона не бачила раніше. Якщо це буде, ми скажемо, що модель добре узагальнює. Для того, щоб оцінити, наскільки добре модель узагальнює дані, зазвичай дані поділяються на дані тренувань та дані тесту, а потім оптимізуються або навчаються на першому, але оцінюються лише на останньому.

f_w буде моделюватися за допомогою згорткової нейронної мережі (ЗНМ), як у виграшному рішенні завдання BirdCLEF 2016 щодо класифікації птахів. Проте, крім традиційного ЗНМ, ми пропонуємо використовувати залишкове навчання з ідентифікаційними відображеннями в рамках архітектури ЗНМ. Ці модифікації в архітектурі ЗНМ показали багатообіцяючі результати та були частиною виграшного рішення завдання класифікації ILSVRC 2015 року. Модифікації повинні дозволяти тренувати набагато глибші мережі, а глибші мережі зазвичай пов'язані з кращими результатами класифікації.

2.5.Виділення признаків

Ще однією проблемою під час навчання нейронних мереж є визначення того, які функції слід використовувати як вхідні дані в мережу. Зокрема, як витягнути абстрактні функції, які містять стільки інформації про вихідний матеріал, скільки можливо, але з меншими розмірами, щоб забезпечити ефективне навчання.

Дизайн методу вилучення ознак можна розглядати як компроміс між сепарабельністю та скороченням. Ми хочемо, щоб підрядник або екстрактор функції $\Phi(x)$ зменшив розміри точок даних x, не жертвуючи сепарабельністю, тобто хочемо, щоб $\Phi(x) \neq \Phi(x')$ якщо $f(x) \neq f(x')$, де f - справжній класифікатор. Якщо ця властивість тримається, то ми кажемо, що Φ відокремлює f.

Спренгель використовує спектральні обчислення як підрядника, так і вхідний шум, фільтрувані шматки спектрограм до ЗНМ. Ці шматки розширюються чотирма способами: переміщенням часу, переміщенням висоти, додавання шуму та об'єднання аудіофайлів одного класу. В цій дисертації пропонується альтернативне представлення функцій, коли Мель-частотні кепстральні коефіцієнти (МЧЦК) доповнюються множинною шириною частоти-дельта-збільшення даних, щоб визначити, чи може вона підвищити точність класифікації у домені пташиних пісень.

Я пропоную використовувати нову архітектуру - згорткову нейронну мережу. Ми також досліджуємо використання мета-даних, крім аудіо-даних, як способу підвищення точності класифікації. Класифікатор повинен мати змогу ідентифікувати найбільш помітну співочу птицю в акустичному звуковому полі. Акустичний звуковий майданчик - це композиція геофонії (наприклад, вітер, дерева та дощ), біофонія (наприклад, пташки, жаби та комахи) та антропофонія (наприклад, літаки, автомобілі та поїзди).

З огляду на цю мету існують три основні питання дослідження, на які спрямована на відповідь: (i) Чи можна використовувати глибокі залишкові нейронні мережі для класифікації видів птахів на основі записів акустичних даних і наскільки добре вони виконують? (ii) Чи можна використовувати кілька збільшень частоти дельта-частоти для покращення точності класифікації у цій проблемній області? І (iii) чи можуть додаткові метадані записів використовувати для підвищення точності класифікації?

2.5.1. Ініціалізація

Ваги глибоких нейронних мереж можуть бути ініціалізовані шляхом об'єднання зі стандартного розподілу з нульовим середнім та невеликим фіксованим стандартним відхиленням. Однак, якщо вага в мережі починається занадто мала, то сигнал, який поширюється через шари мережі, може скоротитися занадто швидко, щоб бути корисним, або, навпаки, якщо ваги починаються занадто великими, сигнал може зростати надто швидко, як він проходить через мережу, що робить його великим, щоб бути корисним. Тому може стати важко зблизити дуже глибокі мережі. Іншою причиною проблем конвергенції може бути те, що градієнти стають нестабільними при поганій ініціалізації, в результаті чого градієнти зникають.

Якщо градієнти стають малими, то швидкість конвергенції до оптимального також стає невеликою, а час тренування для мережі збільшується. Тобто початкові ваги мережі впливають на те, як сигнал поширюється вперед у мережі, і як градієнти поширюються назад у мережі, і в ідеалі обидва з них повинні бути стабільними.

Глорот і Бенжіо пропонують, що вони називають нормалізованою ініціалізацією:

$$W_j^{(i)} \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}, \frac{\sqrt{6}}{\sqrt{n_j} + n_{j+1}}\right],$$

де U [-a, a] - рівномірний розподіл в інтервалі (-a, a), n_i - розмір шару і. Однак цей метод ініціалізації передбачає, що функції активації є лінійними, що не відповідає дійсності для ReLU. У роботі використовується метод ініціалізації, який враховує ReLU.

2.5.2. Оптимізація

Метод оптимізації - це стохастичний градієнтний спуск зі швидкістю навчання, η, 0,001, імпульс Нестерова, μ, 0,9 та швидкість розпаду 1e-6. У розділі 1.1.1 ми пояснили, що проблема оптимізації може бути визначена як:

$$w = \arg\min_{w} \sum_{\bar{x} \in X} loss(f(\bar{x}), f_w(\bar{x}))\,,$$

де w - параметри, які мінімізують втрати над тренувальним набором X для параметризованої функції f_w. Градієнтний спуск використовується для оновлення параметрів у напрямку, який найбільше знижує втрати:

$$w_{t+1} = w_t - \eta \Delta Q(w_t) = w_t - \eta \sum_{i=1}^{N} \Delta Q_i(w_t)\,,$$

де $Q_i(w) = loss(f(\bar{x}_i), f_w(\bar{x}_i))$ це втрата моделі тренувального зразка $\bar{x}_i \in X$. Обчислення повного градієнта при кожному оновленні, проте, є дорогим. Тому використовується стохастичний градієнтний спуск, який виконується "on-line" і наближає дійсний градієнт, використовуючи лише один зразок або підмножина зразків. Для простоти, в цьому поясненні використовується лише один зразок. Правило оновлення для стохастичного градієнта гідне тоді можна визначити як: $w_{t+1} = w_t - \eta \Delta Q_i(w_t)$ де ми обчислюємо градієнт функції втрат при навчанні вибірки і та оновлюємо параметри в негативному напрямку цього градієнта, масштабованого за швидкістю навчання η.

Правило оновлення з використанням імпульсу Нестерова можна визначити як:

$$v_{t+1} = \mu v_t - \eta \Delta Q_i(w_t + \mu v_t)\,,$$

$$w_{t+1} = w_t + v_{t+1}\,,$$

де v_t представляє імпульс, який накопичується, якщо напрямок градієнта залишається тим самим через кілька оновлень, а $\mu \in [0, 1]$ - коефіцієнт імпульсу Нестерова. Правило оновлення застосовується один раз для кожного навчального зразка протягом епохи навчання, а зразки тренувань перемішуються.

2.5.3. Функція втрати

Функція втрат - це показник того, наскільки ефективна мережева модель виконує набір позначених точок даних. Найбільш часто використовувані функції втрат є середньою квадратною помилкою та перехресною ентропією, обидві з яких є безперервними, диференційованими і можуть бути ефективно обчислені, що необхідно під час оптимізації для обчислення градієнта.

ми використовуємо функцію втрати ентропії, оскільки вона має ще три чудові властивості: (i) вона завжди позитивна; (ii) втрата прагне до нуля, коли оцінка мережі має тенденцію до бажаного результату, і (iii) сума втрат, яка визначається частковими похідними від функції втрат, пропорційна тому, як неправильна оцінка мережі. Особливо (ii) та (iii) є властивостями, які інтуїтивно очікують від навчальної системи.

Окремо-маркована категоріальна крос-ентропія може бути визначена як:

$$C = -\frac{1}{N}\sum_{n=1}^{N}\left[y_n \log \hat{y}_n + (1-y_n)\log(1-\hat{y}_n \right],$$

де N - загальна кількість навчальних зразків, $y_n = f(\bar{x}_n)$ - це бажаний вихід для точки даних $\bar{x}_n$, а $\hat{y}_n = f_w(\bar{x}_n)$ - очікуваний результат.

2.6. Обґрунтування вибору програмних засобів

2.6.1. Python

Python - це мова програмування, створена наприкінці 1980-х і названана честь Monty Python, який тисячі людей використовують для тестування мікрочипів на Intel, для включення Instagram, для створення відеоігор з бібліотекою PyGame. Він невеликий, дуже нагадує англійську мову та має сотні існуючих сторонніх бібліотек.

Тож які основні причини, чому вибирають Python.

Читабельність

Python дуже нагадує англійську мову, використовуючи такі слова як "не" та "в", щоб зробити це дуже часто, коли ви часто читаєте програму чи скрипт

уголос комусь іншому та не відчуваєте, що ви говорите якусь розмовну мову. Це також допомагає дуже суворі правила пунктуації Python, що означає, що у вас немає фігурних дужок ({}) у всьому коді.

Крім того, Python має набір правил, відомих як PEP 8, які розповідають кожному розробнику Python, як форматувати свій код. Це означає, що ви завжди знаєте, де слід розміщувати нові рядки, і що ще важливіше, що майже будь-який інший сценарій Python, який ви виберете, незалежно від того, написано новим чи досвідченим професіоналом, буде виглядати дуже схожим і таким же простим для читання. Той факт, що мій код Python, що має п'ять або більше років досвіду, виглядає дуже схожим на код, який пише Гвідо ван Россум (творець Python), такий підсилення його.

Бібліотеки

Python існує вже більше 20 років, тому багато коду, написаного на Python, склалися протягом десятиліть, і, будучи мовою програмування з відкритим вихідним кодом, багато з них було випущено для інших користувачів. Ви можете встановити це програмне забезпечення у вашій системі для використання власними проектами. Наприклад, якщо ви хочете використовувати Python для побудови сценаріїв з аргументами командного рядка, встановіть бібліотеку "click", а потім імпортуйте її у свої сценарії та використовуйте її. Існують бібліотеки для практично будь-якого випадку користування, яке ви можете придумати, від маніпулювання зображенням, до наукових розрахунків, до автоматизації сервера.

2.6.2. PHP

PHP це мова сценаріїв на стороні сервера призначений в основному для веброзробки, але також використовується в якості мови програмування загального призначення.

PHP код може бути вбудований в HTML або HTML5 розмітки, або він може бути використаний в поєднанні з системами різних веб-шаблонів, системи управління веб-контенту та веб-фреймворків. PHP-код зазвичай

обробляється у PHP інтерпретатора, реалізованого у вигляді модуля в веб-сервері або в якості Common Gateway Interface виконуваного файлу. Програмне забезпечення веб-сервер об'єднує результати коди інтерпретуються і виконуються PHP, які можуть бути будь-яким типом даних, включаючи зображення, яке генерується вебсторінкою. PHP код також може бути виконаний за допомогою інтерфейсу командного рядка і може бути використаний для реалізації автономних графічних програм. Стандартний PHP інтерпретатор, від Zend Engine, це вільне програмне забезпечення, яке розповсюджується за ліцензією PHP. PHP була широко портирована і може бути розгорнута на більшості веб-серверів майже на кожній операційної системи і платформи, безкоштовно.

Мова PHP не еволюціонували без формальної специфікації або стандарту до 2014 року, в результаті чого канонічний PHP інтерпретатор як дефакто стандартом. З 2014 року робота пішла на створення формальної специфікації PHP.

PHP зберігає цілі числа в платформі залежать від діапазону, або 64-бітове або 32-бітове ціле число еквівалентно типу. Непідписані цілі числа перетворюються в підписани значення в певних ситуаціях; ця поведінка відрізняється від інших мов програмування. Цілі змінні можуть бути задані за допомогою десяткового (позитивні і негативні), восьмеричні, шістнадцяткові виконавчі нотації. Числа з плаваючою точкою також зберігається в діапазоні конкретної платформи.

Вони можуть бути вказані за допомогою позначень з плаваючою точкою. PHP має нативний логічний тип, який схожий на нативні булеві тип в Java і C ++. Використовуючи правила перетворення логічного типу, відмінні від нуля значення інтерпретуються як справжні і нуль як помилкове, як і в Perl і C ++.

Тип нульових даних являє собою змінний, яка не має ніякого значення; NULL, це єдине допустиме значення для цього типу даних. Змінні типу

«ресурс» є посилання на ресурси з зовнішніх джерел. Вони зазвичай створюються з функцій певного розширення, і можуть бути оброблені тільки за допомогою функцій з того ж розширення; приклади включають в себе файл, зображення і ресурси бази даних. Масиви можуть містити елементи будь-якого типу, PHP може працювати, в тому числі ресурсів, об'єктів та інших масивів. Замовлення зберігається в списках значень хеш з двома ключами і значеннями. PHP також підтримує рядки, які можуть бути використані в одинарні лапки, подвійні лапки, nowdoc або синтаксис Heredoc.

Стандартна бібліотека PHP намагається вирішити стандартні проблеми та реалізує ефективні інтерфейси доступу до даних і класи. PHP є універсальною мовою сценаріїв, які особливо підходять для розробки серверних веб додатках, в цьому випадку PHP зазвичай працює на веб-сервері. Будь який PHP код у файлу виконується за допомогою PHP виконання, як правило, для створення динамічного вмісту веб-сторінки або динамічні зображення, які використовуються на веб-сайтах або в інших місцях. Він також може бути використан для командного рядка сценаріїв і на стороні клієнта графічного призначеного для користувача інтерфейсу (GUI) додатків. PHP може бути розгорнуто на більшості веб-серверів, багатьох операційних систем і платформ, а також може використовуватися з багатьма системами керування базами даних (СКБД). Більшість веб-хостинг-провайдерів підтримують PHP для використання їх клієнтами. Він доступний безкоштовно і PHP Group надає повний вихідний код для користувачів, щоб створювати, настроювати і розширювати для своїх власних потреб.

Динамічні веб-сторінка: Приклад сценаріїв на стороні сервера (PHP і MySQL).

PHP діє головним чином в якості фільтра, приймає вхідні дані з файлу або потоку, що містить тексту і / або інструкції по PHP і висновок іншого потоку даних. Найчастіше вихід буде HTML, хоча це може бути JSON, XML або двійкові дані, такі як зображення і аудіо форматів. Так як PHP 4, PHP

аналізатор компілює введення, щоб зробити байти-код для обробки по Zend Engine, даючи поліпшену продуктивність у порівнянні з його інтерпретатором попередника. Спочатку розроблений для створення динамічних веб-сторінок, PHP в даний час зосередженний в основному на стороні сервера сценаріїв, і він схожий на іншихмовах сценаріїв на стороні сервера, які забезпечують динамічний вміст з вебсервера до клієнта, такі як Microsoft, класичний ASP, компанії Sun Microsystems JavaServer Pages, і mod_perl. PHP також привертає розвиток багатьох структур програмного забезпечення, які надають будівельні блоки та дизайн структуру для сприяння швидкої розробки додатків. Деякі з них включають ПРАДО, CakePHP, Symfony, CodeIgniter, Laravel, Yii Framework, Phalcon і Zend Framework, пропонуючи функції, аналогічні іншим веб-фреймворків.

Архітектура ЛАМПА стала популярною в веб-індустрії як спосіб розгортання веб-додатків. PHP зазвичай використовується в цій зв'язці поряд з Linux, Apache і MySQL, хоча може також ставитися до Python, Perl, або будь-якої суміші з трьох. Подібні пакети, WAMP і MAMP, також доступні для Windows, і MacOS. Незважаючи на те, як PHP і Apache надаються як частина бази Macos установки, користувачі цих пакетів шукають більш простий механізм установки, який може бути більш легко містити в актуальному стані.

2.6.3. Java

Java об'єктно-орієнтована мова програмування яка впровадила нову мережеву програму, яка називається аплетом, який змінив спосіб, за яким інтернет-світ замислювався над вмістом. Java також звернувся до деяких найпотрібніших проблем, пов'язаних з Інтернетом: переносимість та безпека.

Аплет - це особлива програма Java, яка призначена для передачі через Інтернет і автоматично виконується веб-браузером, сумісним з Java. Крім того, аплет завантажується за вимогою без подальшої взаємодії з користувачем. Якщо користувач натискає посилання, яке містить аплет, аплет буде автоматично завантажений і запускатися в браузері. Аплети призначені для

невеликих програм. Вони зазвичай використовуються для відображення даних, що надаються сервером, обробки введення користувача або надання звичайних функцій, таких як калькулятор позики, який виконує локально, а не на сервері. По суті, аплет дозволяє деяким функціям переміщатись з сервера до клієнта.

Створення аплету змінило Інтернет-програмування, оскільки воно розширило всесвіт об'єктів, які можуть вільно переміщатися в кіберпросторі. Загалом, існує дві дуже широкі категорії об'єктів, які передаються між сервером і клієнтом: пасивна інформація та динамічні, активні програми. Наприклад, коли ви читаєте електронну пошту, ви переглядаєте пасивні дані. Навіть коли ви завантажуєте програму, код програми залишається лише пасивними даними, поки ви не виконаєте його. Навпаки, аплет - це динамічна програма, що самостійно виконує. Така програма є активним агентом на клієнтському комп'ютері, однак його ініціює сервер.

Як бажано, як динамічні, мережеві програми, вони також представляють серйозні проблеми в галузі безпеки та мобільності. Очевидно, що програму, яка завантажує та виконує автоматично на клієнтському комп'ютері, повинна запобігати нанесенню шкоди. Він також повинен мати можливість працювати в різних середовищах та під різними операційними системами. Як ви побачите, Java вирішив ці проблеми ефективно та елегантно. Давайте подивимося трохи ближче до кожного.

Як ви, напевно, знаєте, що кожен раз, коли ви завантажуєте "звичайну" програму, ви ризикуєте, оскільки завантажений код може містити вірус, троянський кінь або інший шкідливий код. В основі проблеми лежить той факт, що шкідливий код може завдати шкоди, оскільки він отримав несанкціонований доступ до системних ресурсів. Наприклад, програма вірусів може збирати приватну інформацію, таку як номери кредитних карт, залишки на рахунках банків та паролі, шляхом пошуку вмісту локальної файлової системи вашого комп'ютера. Щоб Java дозволила безпечно завантажувати та

виконувати на комп'ютері клієнта аплети, необхідно було запобігти запуску такого нападу аплету.

Java досягла цього захисту, обмеживши аплет у середовище виконання Java, і не дозволяючи йому отримувати доступ до інших частин комп'ютера. (Ви побачите, як це буде зроблено найближчим часом.) Можливість завантажувати аплети з упевненістю, що ніякої шкоди не буде зроблено, і що ніякої безпеки не буде порушено, багато хто вважає єдиним найбільш інноваційним аспектом Java.

Портативність - це найважливіший аспект Інтернету, оскільки до нього є багато різних типів комп'ютерів та операційних систем. Якщо програма Java повинна була запускатися практично на будь-якому комп'ютері, підключеному до Інтернету, то для цього необхідно мати певний спосіб включити цю програму в різні системи. Наприклад, у випадку аплету той самий апплет повинен мати можливість завантажувати та виконувати різноманітність різних процесорів, операційних систем та веб-переглядачів, підключених до Інтернету. Непрактично мати різні версії аплету для різних комп'ютерів. Цей же код повинен працювати на всіх комп'ютерах. Тому було потрібне певне засіб для створення портативного виконуваного коду. Як ви побачите незабаром, той самий механізм, який допомагає забезпечити безпеку, також допомагає створити портативність.

Ключ, який дозволяє Java вирішити тільки описані проблеми безпеки та портативності, полягає в тому, що вихідний код компілятора Java не є виконуваним кодом. Швидше, це байт-код. Bytecode - це високо оптимізований набір інструкцій, призначений для виконання системи Java run time, яка називається віртуальною машиною Java (JVM). По суті, оригінальний JVM був розроблений як інтерпретатор для байт-коду. Це може стати трохи сюрпризом, оскільки багато сучасних мов розробляються для складання виконуваного коду через проблеми продуктивності.

Проте той факт, що програма Java виконується JVM, допомагає вирішити основні проблеми, пов'язані з веб-програмами. Ось чому. Переклад програми Java у байт-код допомагає набагато простіше запускати програму у різноманітних середовищах, оскільки для кожної платформи потрібно застосовувати лише JVM. Після того, як пакет існування для даної системи існує, може працювати будь-яка програма Java. Пам'ятайте, що хоча деталі JVM відрізнятимуться від платформи до платформи, всі розуміють той самий байт-код Java. Якщо програма Java була скомпільована до власного коду, то для кожного типу процесора, підключеного до Інтернету, повинні були існувати різні версії однієї і тієї ж програми. Це, звичайно, не є можливим рішенням. Таким чином, виконання байт-коду JVM - це найпростіший спосіб створити справді портативні програми.

Той факт, що програма Java виконується JVM, також допомагає зробити її безпечною. Оскільки JVM контролюється, він може містити програму та запобігти появі сторонніх ефектів за межами системи. Безпека також посилюється через певні обмеження, які існують на мові Java. Коли програма інтерпретується, вона, як правило, працює повільніше, ніж однакова програма буде запускатися, якщо скомпільований до виконуваного коду. Однак, з Java, різниця між двома не настільки велика. Оскільки байт-код був високо оптимізований, використання байт-коду дозволяє JVM виконувати програми набагато швидше, ніж ви могли очікувати.

Незважаючи на те, що Java була розроблена як інтерпретована мова, про Java не існує нічого, що дозволяє автоматично збирати байт-код у власний код, щоб підвищити продуктивність. З цієї причини технологія HotSpot була введена незабаром після першого випуску Java. HotSpot забезпечує компілятор у форматі JIT за байт-код. Коли JIT-компілятор є частиною JVM, вибрані частини байт-коду збираються у виконуваний файл у режимі реального часу за принципом "попит". Важливо зрозуміти, що це непрактично скласти цілу програму Java у виконуваний код відразу, оскільки Java виконує різні

перевірки виконання часу, які можна виконувати лише під час виконання. Замість цього компілятор JIT компілює код, як це потрібно, під час виконання. Крім того, не всі комбінації послідовностей байт-коду - лише ті, на котрі виграють

компіляція Залишковий код просто інтерпретується. Тим не менше, підхід, що належить до часу, все ще дає значний приріст продуктивності. Навіть коли динамічна компіляція застосовується до байт-коду, функції переносимості та безпеки все ще застосовуються, оскільки JVM все ще відповідає за середовище виконання.

Висновки до розділу

Був проведений аналіз що до нейронної мережі в якому ми дізналися - якщо об'єднати два або більше штучних нейронів, то ми отримаємо штучну нейронну мережу. Один штучний нейрон має нульові шанси у вирішенні реальних проблем, в той самий час як штучні нейронні мережі можуть вирішити реальні проблеми. Фактично штучні нейронні мережі здатні вирішувати складні реальні проблеми шляхом обробки інформації в їх основних будівельних блоках (штучних нейронах) в нелінійному, розподіленому, паралельному та місцевому вигляді.

Також ми проаналізували методи класифікації сигналу після чого стало ясно що для класифікації, спочатку потрібно вирішити що нам потрібно класифікувати. З цього випливає аж 3 типи класифікації, обравши потрібний тип класифікації ми потрапляємо нової проблеми яку було проаналізовано – виділення признаків. Також було проаналізовано сильні сторони потрібної нам для розробки нейронної мережі мови програмування.

Розділ 3. РОЗРОБКА АЛГОРИТМІЧНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1.Обробка сигналів

Неактивні аудіодані не підходять як вхідні дані для нейронної мережі, і тому аудіосигнал зазвичай перетворюється на спектральне представлення часу.

3.1.1.Спектрограма

Спектрограмма дискретного звукового сигналу $\bar{x} = x_1,...,x_n$ обчислюється в два або три шага. По-перше, Short-Time Fourier Transform (коротке тимчасове перетворення Фур'є) (STFT) застосовується до аудіо сигналу. STFT розраховується стандартним способом, розділяючи сигнал на різні перекриваючиїся фрейми, а потім вичислить Discrete Time Fourier Transform (DTFT) для кожного кадру, що призводить до матриці з складними значеннями

$$STFT\{\bar{x}\}(m,\omega) \equiv X_m(\omega) = \sum_{n=-\infty}^{\infty} x_n w(n-mR) e^{-j\omega n},$$

де x_n - вхідний сигнал в момент часу n, $w(n)$ - довжина M = 512 вікно хеннінга орієнтовано навколо n, а R = 128 - представляє собою розмір переходу між послідовними кадрами. Я використовував librosa.stft метод бібліотеки librosa для обчислення STFT. По-друге, обчислюється квадратна амплітуда величини STFT, яку ми називаємо ampspectrogram, і, по-третє, натуральний логарифм амплітудної спектрограми обчислюється, яку ми називаємо как logspectrogram.

$$ampspectrogram\{\bar{x}\}(\omega) \equiv \left|\bar{X}(\omega)\right|^2$$

$$Logspectrogram\{\bar{x}\}(\omega) \equiv \log_e\left(\left|\bar{X}(\omega)\right|^2\right),$$

3.1.2.Мель частотних сепстральних коефіцієнтів

Мель Часткові Сепстральні коефіцієнти (МЧЦК) є стандартним вибором як аудіофункцій, які використовуються при розпізнаванні мовлення, і їхній успіх багато в чому залежить від кількості інформації, яку вони містять про аудіосигнал у такій стисненій формі. Мель Частотна Септемна приблизно відповідає тому, як слухняна система людини працює зі звуком. Для обчислення MFCC використовуються наступні етапи:

(i) обчислюють спектрограмму енергетичної величини сигналу,

(ii) об'єднати БПФ в чашки Мел-частоти,

(iii) взяти журнали цих повноважень, і

(iv) виконувати дискретне косінічне перетворення.

MFCC тепер є амплітудами отриманого спектру.

3.1.3. Класифікація видів птахів

метод, який використовується для завдання BirdCLEF буде використаний як базовий рівень.

Попередня обробка

Аудіофайли спочатку переробляються у формат, який можна використовувати для навчання нейронної мережі. Аудіофайли нормалізуються до 16-бітних даних моноканальних хвиль, повторно відібраних з 44100 Гц до 22.050 Гц.

Записи пташиних пісень розділені на дві різні звукові класи: сигнал (пташиний спів) та шум. Розділення дозволяє набути нейронну мережу за найбільш релевантними даними, і це дає нам доступ до класу шумів, який може бути використаний для покращення навчальних зразків. Після того як запис був розділений на сигнальну хвилю і шумову хвилю, кожен розділяються на 3-секундні сегменти, які зберігаються на диску. Сегменти шуму можуть пізніше бути використані для посилення навчальних зразків, що показуються в мережі, які повинні поліпшити узагальнення

Після того як запис був розділен на сигнальну хвилю і шумову хвилю, вони кожен розділяються на 3-секундні сегменти, які зберігаються на диску. Сегменти шуму можуть пізніше бути використані для посилення навчальних зразків, що показуються в мережі, які повинні поліпшити узагальнення. Частина сигналу витягується спочатку обчислюючи маску сигналу $\bar{v}$ для заданої звукової хвилі $\bar{x}$, а потім використовуючи маску для витягання відповідної частини звукової хвилі, де $v_i = 0$ вказує, що x_i не є частиною сигналу, і $v_i = 1$ вказує, що це частина сигналу. Маска походить від бінарного зображення, яке обчислюється шляхом аналізу нормалізованої амплітудної спектрограми $\bar{x}$. Нехай $\tilde{b}$ бінарним зображенням, і нехай $\tilde{s}$ буде нормованою амплітудною спектрограмою $\bar{x}$, де $\tilde{b}$ та $\tilde{s}$ мають однакові розміри. Піксель за індексом (i, j) у бінарному зображенні потім встановлюється в один, якщо $s_j^{(i)}$ у t раз перевищує межу рядка та стовпця $\tilde{s}$ у рядку i та стовпці j

$$b_j^{(i)} = \begin{cases} 1, & if\ s_j^{(i)} > t \times median\left(\bar{s}^{(i)}\right) \wedge s_j^{(i)} > t \times median\left(\bar{s}_j\right) \\ 0, & otherwise \end{cases},$$

Двійкове зображення обробляється додатково, застосовуючи бінарну ерозію, а потім двійкове розширення на зображенні, обидва з використанням розміру ядра від 4 до 4, що згладжує області, позначені як пташиний спів, а маска сигналу $\bar{v}$ походить від Бінарне зображення, встановлюючи v_j до одного, якщо стовпчик $\bar{b}_j$ містить один. Маску також згладжують шляхом виконання ще двох двійкових розбавлення (розмір ядра 4), а потім повторно масштабувати так, що $|\bar{v}| = |\bar{x}|$. Зображення на малюнку були вручну порівняно для кожного етапу з відповідним зображенням, представленим Спренгеля для одного звукового файлу, і метод видає подібні результати.

Алгоритм Обчислення маски сигналу

1: procedure ComputeMask $\left(\bar{r}, t\right)$

2: P xx ← spectrogram $\left(\bar{r}\right)$

3: P xx ← normalize (Pxx)

4: BinaryImage ← medianClipping (Pxx, t)

5: BinaryImage ← erosion (BinaryImage,(4, 4))

6: BinaryImage ← dilation(BinaryImage,(4, 4))

7: mask ← computeM ask(BinaryImage)

return mask

Маска сигналу обчислюється шляхом встановлення порога t = 3, а шум обчислюється шляхом встановлення t = 2,5, а потім інвертує маску в кінці (фліпінг 0c-1c, а 1c-0c). Це може залишити частину хвилі, яка позначена як не сигнал, ані шум (2.5-3). Вважається, що ці частини не сприяють випуску будь-якої відповідної інформації для мережі, і їх просто ігнорують.

3.2. Збільшення даних

Збільшення даних - це спосіб збільшення кількості навчальних зразків у наборі даних шляхом поповнення навчальних даних. Навіть якщо набір даних BirdCLEF є одним з найбільших доступних наборів даних пташиних пісень, кількість навчальних зразків на види птахів є досить невеликою; в середньому близько 30 зразків у звуковому класі. Збільшення даних також може бути використане для посилення штамів навчальних зразків шляхом введення шуму до сигналу, який покликаний запобігти перевизначенню і зробити модель загальною кращою через більшу інваріантність шуму.

3.2.1. Той же клас і додавання шуму

З метою покращення рівня зближення нейронної мережі навчальні зразки збільшуються шляхом додаткового поєднання кожного зразка з іншим зразком одного класу, що дозволяє мережі одночасно переглядати більш релевантні дані. Зразки також додатково поєднуються з трьома випадковими сегментами шуму, щоб зробити мережу більш шумовою інваріантністю, і тому краще узагальнювати. Таким чином, кожен зразок нейронної мережі являє собою комбінацію двох випадкових сегментів сигналу з одного і того ж класу

сигналів $\bar{x}_1$ та $\bar{x}_2$, а також трьох випадкових сегментів шуму $\bar{n}_1$, $\bar{n}_2$ та $\bar{n}_3$ однієї довжини: $\bar{s}_{aug} = \alpha\bar{x}_1 + (1-\alpha)\bar{x}_2 + \beta(\bar{n}_1 + \bar{n}_2 + \bar{n}_3)$

де $\alpha \in [0,1)$ вибирається рівномірно випадковим чином, $\beta = 0.4$ використовується як коефіцієнт демпфірування сегментів шуму. Усі сегменти складають три секунди, а зразки - 22,050 Гц

3.2.2.Зміна часу та висоти

Збільшення зсуву часу здійснюється шляхом розщеплення зразка спектрограму на дві частини вздовж осі часу, а потім покладіть другу частину до першого. Тобто, зміна обертання в часовій області. Підсилення зміщення висоти відбувається аналогічним чином, але в частотній області (вертикально), і зміна становить лише близько 5%. Більші зміни не вважаються корисними. Зміни виконуються на логарифмічній спектрограмі того ж класу та шумопоглинаючих зразків навчання, незадовго до того, як вони будуть показані нейронній мережі.

3.2.3 Підвищення частоти дельта-даних з множинною шириною

Мель-частотні цефстративні коефіцієнти (МЧЦК) - це найчастіше використовувані функції під час навчання аудіо класифікаторів. Хан представляє нову техніку збільшення даних, яка використовується для підвищення корисності таких функцій. Ідея полягає в тому, щоб обчислити дельта-особливості МЧЦК, які містять додаткову інформацію про траєкторії МЧЦК. Це дає мережі не тільки локальну статичну інформацію про сигнал, але також динамічну інформацію про те, як сигнал буде продовжуватися в (найближчому майбутньому). Підраховується збільшення множинної ширини частоти-дельта:

$$d_t = \frac{\sum_{k=1}^{K} k(x_{t+k} - x_{t-k})}{2\sum_{k=1}^{K} k^2},$$

де d_t - дельта-функція, обчислена з кадру t в термінах коефіцієнтів статичного коефіцієнта сепструму x_{t-k} до x_{t+k} . Ширина дельти відноситься до 2K + 1, а Han пропонують ширину Δ3, Δ11 та Δ19 (тобто K = 1, K = 5, K = 9). Вхідний сигнал до ЗНМ є тоді векторним або чотириканальним зображенням (static, Δ3, Δ11, Δ19), де static означає МЧЦК. Отже, якщо | static | = 12 тоді | Δ3 | = | Δ11 | = | Δ19 | = 12, що означає, що ми збільшимо вхідний вектор 12 функцій у вхідний вектор 48 функцій. Корпусні крапки вирішуються шляхом повторного використання першого або останнього коефіцієнта в статичному відповідно.

Я використовував перші 32 МЧЦК за результатами, які показують, що 32 МЧЦК є солодким місцем для ідентифікації звуків жаби та комах. Вони витягуються за допомогою бібліотеки, яка вираховує Мел-спектрограму за допомогою вікна FFT довжиною 2048 пікселів та розміром хопу 512. Оброблені аудіо сегменти складають три секунди з частотою зразків 22,050 Гц, що дає вектор форми (32, 129, 1), який, у свою чергу, змінюється до (32, 128, 1). Дельти потім обчислюються з зміненого розміру, використовуючи librosa.feature.delta, і використовуються для збільшення, як описано в цьому розділі. Таким чином, остаточна форма вектора функції (32, 128, 4), яка використовується як вхід до класифікатора.

3.3. Злиття мета-даних

Таблиця 3.1. Архітектура 18-шарового глибокого залишкового нейрона. Конфігурація базового блоку, наприклад, "64 3x3 ядра, 2x2 кроків" повинна розглядатися як: кількість фільтрів згортаних шарів в базовому блоці становить 64, розмір ядра - 3x3, розмір кроку першого згортаного шару 2x2, а розмір кроку другого згортаного шару 1x1.

Layer (type)	Configuration	Output Shape
InputLayer		(256, 512, 1)
Convolution2D	64 7x7 kernels, 2x2 stride	(128, 256, 64)
MaxPooling2D	3x3 kernel, 2x2 stride	(64, 128, 64)
BasicBlock	64 3x3 kernel, 1x1 stride	(64, 128, 64)
BasicBlock	64 3x3 kernel, 1x1 stride	(64, 128, 64)
BasicBlock	128 3x3 kernel, 2x2 stride	(32, 64, 128)
BasicBlock	128 3x3 kernel, 1x1 stride	(32, 64, 128)
BasicBlock	256 3x3 kernel, 2x2 stride	(16, 32, 256)
BasicBlock	256 3x3 kernel, 1x1 stride	(16, 32, 256)
BasicBlock	512 3x3 kernel, 2x2 stride	(8, 16, 512)
BasicBlock	512 3x3 kernel, 1x1 stride	(8, 16, 512)
AveragePooling2D	8x16 pool size, 1x1 stride	(1, 1, 512)
Flatten		(512)
Dense	He normal, softmax	(999)
Total Params	11,691,751	

Більшість записів у наборі даних містять метадані, пов'язані з ними, деякі з яких, можливо, можуть бути використані для підвищення точності класифікації вивченої моделі f_w . Не всі записи в наборі даних мають повні метадані, а це означає, що важко буде навчити класифікатора, який використовує ці дані. Проте має бути можливим застосувати байєсівський підхід і поєднувати або об'єднати цю інформацію з моделлю, яка була підготовлена тільки до аудіоданих, а отже, для метаданих. Як доказ поняття підняття запису та ймовірність того, що запис належить до певного виду птиці, об'єднується з використанням теореми Байєса для обчислення задня вірогідність виду птахів, що дають запис і висоту. Близько 90% записів мають дані про висоту, пов'язані з ними.

Таблиця 3.2. Архітектура базового блоку.

Layer (type)	Configuration	Output Shape
InputLayer		(n, m, d)
BatchNormalization		(n, m, d)
Activation	ReLU	(n, m, d)
Convolution2D	c kernels, sxs stride	(n/s, m/s, c)
BatchNormalization		(n/s, m/s, c)
Activation	ReLU	(n/s, m/s, c)
Convolution2D	c kernels, 1x1 stride	(n/s, m/s, c)
Merge	[InputLayer, Convolution2D]	(n/s, m/s, c)

Вхідний шар - це просто вивід шару, до якого був підключений основний блок. Це означає, що якщо попередній шар має вихідний вигляд форми (n, m, d), тоді вхідний шар отримує таку форму виходу. Обидва згорткові шари використовують кількість фільтрів, які вказані при побудові (попередня табл), але тільки перший згортковий шар використовує вказаний розмір кроку, другий завжди має розмір кроку 1x1.

3.3.1 Висота

Ми визначаємо події спостереження птиці *j* як B_j Bj, події спостереження за певним піднесенням *e* як *E*, а також події спостереження певної пташиної пісні s як S. Використовуючи теорему Байєса, ми можемо потім висловити задню вірогідність спостереження тварини *j* дає докази e та c як:

$$\Pr(B_j \mid E, S) = \frac{\Pr(ESB_j)}{\Pr(ES)} = \frac{\Pr(B_j)\Pr(S \mid B_j)\Pr(E \mid SB_j)}{\sum_{i=1}^{n} \Pr(B_i)\Pr(S \mid B_i)\Pr(E \mid SB_i)},$$

і задня вірогідність спостереження B_j, що дає свідчення *S* як:

$$\Pr(B_j \mid S) = \frac{\Pr(B_j)\Pr(S \mid B_j)}{\sum_{i=1}^{n} \Pr(B_i)\Pr(S \mid B_i)},$$

Нормалізуючий чинник у рівнянні є постійним і позначаючи його як C, ми отримуємо:

$$\Pr(B_j \mid S) \times C = \Pr(B_j)\Pr(S \mid B_j) \quad ,$$

Зараз ми спостерігаємо, що права частина рівняння з'являється як у

чисельнику, так і в знаменнику першого рівняння (хоча з різними показниками), і, вставивши рівняння останнє у формулу, ми отримуємо:

$$\Pr(B_j \mid E, S) = \frac{\Pr(B_j \mid S)\Pr(E \mid SB_j)}{\sum_{i=1}^{n} \Pr(B_i \mid S)\Pr(E \mid SB_i)},$$

Задня вірогідність $\Pr(B_j \mid S)$ може бути оцінена класифікатором, але слід розрахувати ймовірність $\Pr(E \mid SB_j)$ доказів E, заданих SB_j. Для цього ми припускаємо, що події E та S незалежні, якщо B_j означає, що $\Pr(E \mid SB_j) = \Pr(E \mid B_j)$. Що дозволяє нам переписати останнє рівняння як:

$$\Pr(B_j \mid E, S) = \frac{\Pr(B_j \mid S)\Pr(E \mid B_j)}{\sum_{i=1}^{n} \Pr(B_i \mid S)\Pr(E \mid SB_i)},$$

Оскільки кількість спостережуваних висот для кожного виду досить невелика, нам потрібно наблизити їх розподіл. Розподіл моделюється за щільністю суміші гаусових та рівномірних; наближаючись до уніформи, якщо кількість спостережень невелика, і до гаусових, якщо велика кількість спостережень. Причиною цього є те, що ми не хочемо нульових імовірностей для будь-якого спостережуваного висоти. Гіпотеза полягає в тому, що це дозволяє краще узагальнити модель.

Формально, пусть O_j - сукупність спостережуваних висот для j. Ймовірність спостереження e за j визначається як:

$$h(e \mid j) \equiv (1 - \alpha(k_j))\, f(e \mid \mu_j, \sigma_j) + \alpha(k_j)\left(\frac{1}{e_{\max}}\right),$$

Де:

$$f(e \mid \mu, \sigma) = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

це щільність імовірності нормальної розподілу, k_j - потужність O_j, μ_j - середнє значення O_j, σ_j - стандартне відхилення O_j, $e_{\max}$ - максимальна видима висота, а *α (k)* - вагова функція, яка визначається як:

$$\alpha(k) = \begin{cases} 1, & if\ k < 10 \\ 1/k, & otherwise \end{cases},$$

а це означає, що якщо кількість спостережень висоти *k*, менше 10, то густина суміші є рівномірною, а якщо k> 10, то густина суміші схильна до гаусових.

Я припустив, що тварин не можна спостерігати на висоті нижче рівня моря або вище $e_{\max}$, які встановлені на рівні 5000 метрів над рівнем моря. Остаточний мета-класифікатор, що використовує як голос тварини, так і висоту як доказ для оцінки вірогідності того, що належить тварині, тепер можна визначити як:

$$F_w^{(j)}(s,e) = \frac{f_w^{(j)}(s)h(e \mid j)}{\sum_{i=1}^{N} f_w^{(i)}(s)h(e \mid j)},$$

де $f_w^{(i)}(s)$ - це ймовірність того, що *j* задає пісню *s*, як це передбачалося навченим класифікатором, а $h(e \mid j)$ - це ймовірність спостереження *e* за *j*.

3.4. Набір даних

Дані складаються з 33 200 записів і розділених на загальнодоступний навчальний набір із 24 600 записів та приватний або прихований тестовий набір близько 8600 записів. У цьому наборі даних є 999 різних видів. Записи записуються ентузіастами пташиних пісень у полі, і тому можуть містити інші види, що мають різну довжину (секунди до пів години), і містять різну кількість шуму. У деяких записах є люди, які говорять.

Кожен запис у наборі даних має пов'язаний XML-файл з мета-даними, наприклад:

Таблиця 3.3. Імена наборів даних

Dataset	Number of Classes	Description
BCWhole	999	The whole BirdCLEF data set
BCSubset	20	Subset of 20 classes from BCWhole
BCCubeRunBot100	100	100 hardest classes for the baseline
BCResnetBot100	100	100 hardest classes for the resnet

що використовуються, скільки звукових класів у кожному наборі даних, та короткий опис кожного набору даних. BCW - це весь набір даних BCSubset складається з 20 випадкових вибраних класів звуку з BCWhole; BCCubeRunBot100 складається з 100 звукових класів, для яких базова лінія була найгіршою точністю, а BCResnetBot100 складається з 100 звукових класів, для яких залишкова нейронна мережа мав найгіршу точність; коли навчався на BCWhole.

які можуть використовуватися, крім звукових даних, під час виконання класифікації.

Різні набори даних, що використовуються при розробці методів, наведені в таблиці. Кожен набір даних складається з 90% навчальних даних та 10% даних перевірки, як це робив Спренгель, щоб зробити результати порівнянними. Весь набір даних називається BCWhole, але оскільки весь набір даних досить великий, і це займає значний час, щоб тренувати модель на ньому, підмножина її було створено лише з 20 випадково вибраних птахів для використання під час розробки під назвою BCSubset.

3.4.1. Mean Average Precision(MAP)

(MAP) - це показник того, наскільки важливими є прогнози класифікатора щодо міток істинності на землі. Для прогнозування багатьох класів не існує жодного покарання, головне - високі рейтинги іменних ієрогліфів.

Формально, нехай $f_w : X \to Y$ бути навчальною моделлю, яка прогнозує ймовірність точки $\bar{x} \in X$, яка належить до кожного з можливих класів звуку, і нехай $f : X \to Y$ є справжнім класифікатором, а нехай набір наземних істинних міток $G = f(\bar{x})$. Далі, нехай $K = (k_1,...,k_n)$ - це список прогнозованих міток, що класифікуються за прогнозованими ймовірностями $f_w(\bar{x})$, де k_i є прогнозованою назвою виду в рангу i.

Тоді map може бути визначений як:

$$MAP = \frac{\sum_{\bar{x} \in X} AP(\bar{x})}{|X|},$$

де $|X|$ це кількість тестових аудіофайлів, а $AP(\bar{x})$ - середня точність для точки $\bar{x}$ даних, яку обчислюють як:

$$AP(\bar{x}) = \sum_{i=1}^{n} P(i) \times CiR(i),$$

де n - кількість прогнозів, тобто загальна кількість класів звуку, $P(i)$ - точність при розрізі i у рейтинному списку передбачень для $\bar{x}$, а $CiR(i)$ - change-in-recall при відключенні визначена як:

$$CiR(i) = \begin{cases} 1/n, if\ \ k_i \in G \\ 0, otherwise \end{cases},$$

Ця метрика відображає те, що перші повинні бути хорошими прогнозами. Як приклад, давайте скажемо, що існують лише два різних класу звуку, і що мітки істинної мітки є $G = f(\bar{x}) = \{2\}$, навчальний класифікатор передбачає імовірності $f_w(\bar{x}) = (0.7, 0.3)$, що означає, що K = (1, 2). Тоді P (1) = 0, оскільки клас звуку при скороченні 1 не існує в G, P (2) = 1/2, оскільки зараз ми маємо один правильний прогноз з двох у списку по ранжируванні при відключенні 2. Change-in-recall є CiR(1) = 0, і CiR(2) = 1/2. Таким чином, середня точність $P(1)CiR(1) + P(2)CiR(2) = 0{,}25$. MAP - це усереднена середня точність над усіма точками даних у тестовому наборі.

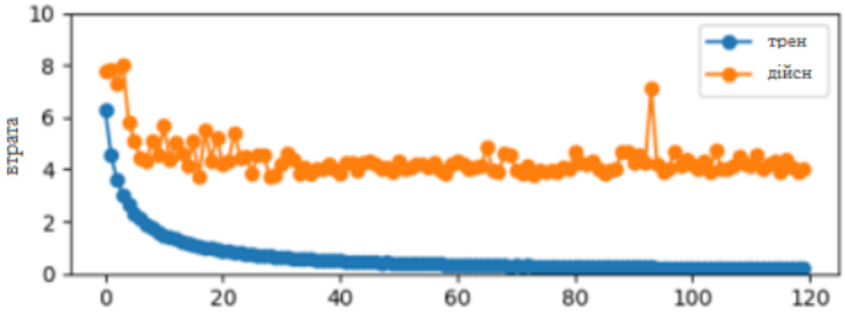
3.4.2. Площа під кривою ROC

The area under the receiver operating characteristic curve (AUROC) - це показник, який вимірює очікування того, що позитивна точка даних, яка рівномірно розподіляється випадковим чином, перевищує точку від'ємних даних, яка рівномірно розподіляється випадковим чином. Цей показник зазвичай використовується в задачах класифікації бінарних програм, але може поширюватися на проблеми з однією міткою, використовуючи підхід де кожен клас розглядається окремо.

Ми дозволимо X позначити набір точок даних і нехай f X^- позначати справжній класифікатор. Розглядаючи певний клас y, ми дозволяємо $\bar{x}^+$ бути позитивним, рівномірно розподіленим випадковим чином з безлічі X^+, де $X^+ = \{\bar{x} | \bar{x} \in X, f(\bar{x}) = y\}$, і нехай $\bar{x}^-$ буде негативним, витягнутим рівномірно випадковим чином з безлічі X^- де $X^- = \{\bar{x} | \bar{x} \in X, f(\bar{x}) \neq y\}$. Тоді метрика AUROC може бути визначена як: $AUROC = \Pr(score(\bar{x}^+) > score(\bar{x}^-))$ де оцінка $(\bar{x})$ представляє ймовірність того, що $\bar{x}$ належить до класу y. Функція оцінки є результатом вивченого класифікатора f_w, який передбачає ймовірність того, що точка даних $\bar{x}$ належить кожному відповідному класу. Очікується, що випадковий класифікатор отримає AUROC оцінку 0,5, а справжній класифікатор, як очікується, отримає рейтинг AUROC 1,0.

3.4.3 Результати навчання

У експерименті було використано 18-шарову залишкову нейронну мережу яка пройшла навчання на 120 епох. історія тренування та результати



якої показано на рисунках. На рисунку показана зміна тренувань та втрат валідації,

Рис.3.2. Показана зміна тренувань та втрат валідації

а також зміна точності навчання та перевірки щодо навчальної епохи

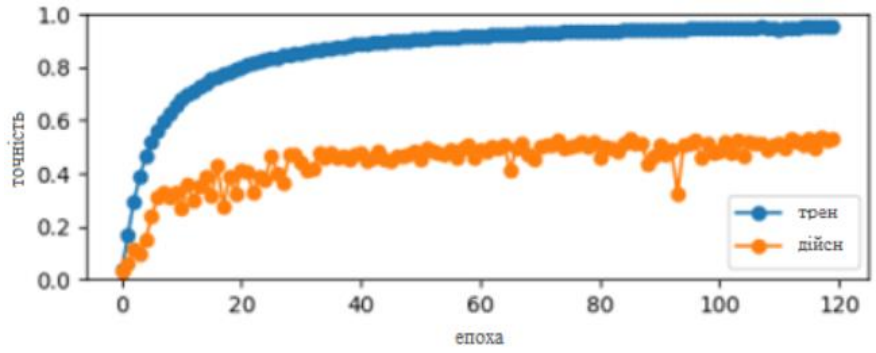


Рис.3.3. Показана зміна точності навчання та перевірки щодо навчальної епохи

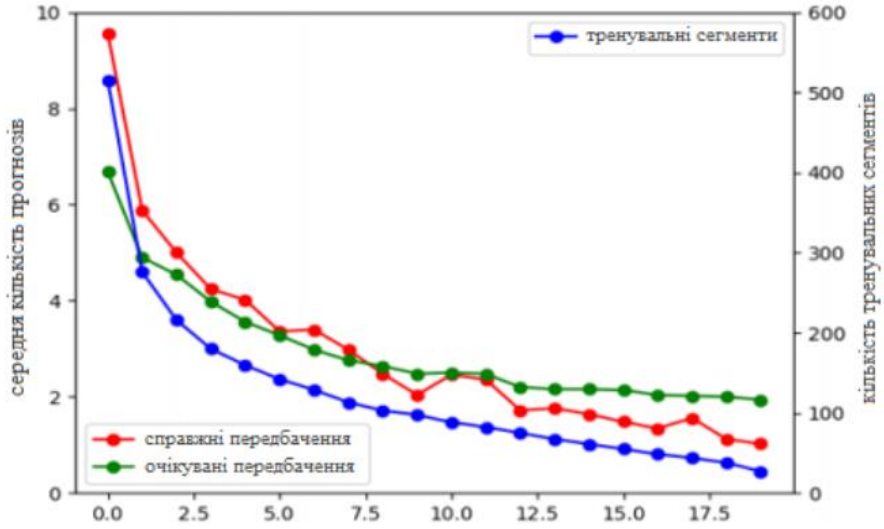


Рис.3.4. Передбачені та тренувальні сегменти

На рисунку показано кількість навчальних сегментів (синіх), зображених на правому осі у відношенні до 5% шматочків класів звуку, класифікованих за кількістю тренувальних сегментів у кожному 5-відсотковому шматочку. Він також показує середню кількість прогнозів, які

отримують шматочки звукових класів (червоний колір) та очікувану середню кількість прогнозів для кожного шматка (зеленого кольору), нанесеного на ліву вісь У.

Висновки до розділу

Було проведено розробку нейронної, та навчання її за допомогою потрібної інформації. Для того щоб навчити нейронну мережу, спочатку треба аудіосигнал перетворити на спектральне представлення часу. Але цього перетворення буде недостатньо для навчання, тому ми використовували методи збільшення набору даних які допомогли нейронній мережі підвищити точність результатів. З метою покращення рівня зближення нейронної мережі ми збільшили навчальні зразки шляхом поєднання кожного зразка з іншим зразком одного класу, що дозволяє мережі одночасно переглядати більш релевантні дані. Зразки також додатково поєднуються з трьома випадковими сегментами шуму, щоб зробити мережу більш шумовою інваріантністю, і тому краще узагальнювати. Було використано 33 200 записів і розділених на загальнодоступний навчальний набір із 24 600 записів та приватний або прихований тестовий набір близько 8600 записів. У цьому наборі даних є 999 різних видів. Записи записуються ентузіастами пташиних пісень у полі, і тому можуть містити інші види, що мають різну довжину (секунди до пів години), і містять різну кількість шуму. Після навчання даної нейронної мережі було показано результати які задовольняли очікування.

Розділ 4. МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЕКТУ

4.1 Опис ідеї проекту

Таблиця 4.1. Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Робочий додаток на мобільний пристрій завдяки якому можна розпізнати тварин та дізнатися інформацію про неї.	Туризм	Туристи можуть дізнаватися про тварин яких вони бачать. Студенти можуть використовують для отримання потрібної інформації. Екологи використовують для обробки інформації та слідкування за популяцією.
	Навчання (комунікація між студентами)	
	Екологія	

Таблиця 4.2. Опис ідеї стартап-проекту

№	Техніко-економічні характеристики ідеї	Продукція конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Дипломний проект	Google Voice	Animal detector			
1	Швидкість роботи	Повільна	Швидка	Середня	Безпосередня робота з базою даних	Простота архітектури	Накопичення інформації в одному місці
2	Зручність використання	Відсутня	Зручно	Зручно	Відсутній зв’язок пристрою з нейронною системою	Розвиток UI має перспективу через наявність API	Розроблений зовнішній вигляд для роботи через веб-сторінку
3	Вимоги до системи	Середні	Мінімальні	Середні	Відсутня оптимізація зі старими системами	Оптимізована робота через веб-додаток	Актуальність програми для нових систем через використання новітніх бібліотек

4	Кросплатформність	Відсутня	Наявна	Відсутня	Відсутня оптимізація для роботи через мобільні пристрої	Налаштована система для роботи через головні інструменти використання користувачами	Присутня оптимізація для роботи через популярні веб-браузери

4.2 Технологічний аудит ідеї проекту

Таблиця 4.3. Технологічна здійсненність ідеї проекту

№	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Робота з накопиченим масивом інформації	Робота через API бази даних MNIST	База даних MNIST та API	Повністю відкрита для роботи з інформацією
2	Робота безпосередньо з інформацією, яку надає користувач	API для роботи з обробки звуку	Розроблені бібліотеки для роботи нейронної мережі	Повністю відкритий програмний код та використання готових бібліотек
3	Робота з накопиченим масивом і інформації та з інформацією безпосередньо користувача системи	Використання всіх зазначених технологій	База даних MNIST та API. Розроблені бібліотеки для роботи нейронної мережі	Повністю відкрита для роботи з інформацією. Повністю відкритий програмний код та використання готових бібліотек
Обрана технологія реалізації ідеї проекту: 3				

Висновок: технологічна реалізація продукту – можлива, вибрана технологія №3, яка може нам допомогти розробити якісний продукт з використанням комбінації технологій, які працюють з великим відкритим масивом даних та обробкою зображень і зберігання результатів

безпосередньо користувачем системи. Техніко-економічні характеристики через на начальному етапі будуть відставати від світових аналогів, які використовуються для іншого призначення та цілей.

4.3 Аналіз ринкових можливостей запуску стартап-проетку

Таблиця 4.4. Попередня характеристика потенційного ринку

№	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн./ум.од	500
3	Динаміка ринку	Стагнація національної економіки. Темпи розвитку світової економіки позитивні.
4	Наявність обмежень для входу	Відсутні. Конкуренти займають свої певні сфери, які висвітлюють функції, які вказані в дисертації стартапу, тільки як додаткові.
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні.
6	Середня норма рентабельності в галузі або по ринку, %	70

Висновок: враховуючи кількість головних гравців по ринку, зростаючу динаміку ринку, невелику кількість конкурентів та середню норму рентабельності можна зробити висновок, що на даний момент, ринок для входження стартап-продукту є привабливим та надає змогу реалізувати продукцію на світовому ринку, оскільки на національному ринку економічний розвиток не надає необхідного результату для вдалої економічної реалізації проекту.

Таблиця 4.5. Характеристика потенційних клієнтів стартап-проекту

№	Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці цільових груп клієнтів	Вимоги споживачів до товару
1	Потреба комунікації між студентами та пересиланням наукового матеріалу один до одного при поганій якості джерела знань	Студенти	Стартап переважно буде виконувати функції пересилання інформації з на електронний пристрій	Зручність у використанні. Швидка робота системи. Спроможність швидко освоїти як користуватися системою. Можливість впроваджувати користувачам свій функціонал у систему для більшого розвитку проекту серед інших цільових аудиторій.
2	Потреба в дослідженні звуків тварини	Екологи	Стартап переважно може мати розвиток у напрямку розпізнавання звуків тварин та моніторингу їх популяції	
3	Потреба в отриманні інформації в іноземній державі	Туристи	Стартап переважно може дати змогу туристам дізнаватися про тварин	

Таблиця 4.6. Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Конкуренти	Наявність конкурентів котрі надають схожі рішення	Зменшення ціни на поставлену послугу; Розробка унікальних характеристик товару; Надання ліцензій на обслуговування
2	Кошти на розробку та підтримку продукту	Закінчення грошей та недостатнє фінансування	Залучення додаткових інвесторів, мотивація роботи на перспективу; Ітеративна розробка продукту задля покрокового виведення продукту на ринок та отримання відповіді користувачів
3	Вихід аналогу	Вихід аналогу даного товару може призвести до знецінення та безідейності даного товару	Вихід товару на ринок в коротші строки з не повною, але достатньою, функціональністю для зацікавлення усіх цільових аудиторій; Проведення рекламної компанії

Таблиця 4.7. Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Новий продукт	Вихід на ринок, Зменшення монополії, Надання нових рішень у сфері	Розробка нової функціональності; Вихід нової продукції на ринок; Надання різноманітних типів ліцензій в залежності від потреб користувача \ замовника.
2	Вихід аналогу	Надати продукт з певними характеристиками та можливостями що відсутні у компаній конкурентів	Аналіз ринку та користувачів задля задоволення їх потреб та надання функціональності у найкоротші строки за ціну, котра є дешевшою ніж у продуктів-замінників.
3	Зворотній зв’язок від користувачів	Можливість отримання необхідної інформації для вдосконалення продукту	Наявність вхідних даних та реакція на них з боку команди розробників задля задоволення потреб та бажань кінцевих користувачів.
4	Грошова винагорода за рекламу	При достатньому попиту на систему можлива комерціалізація продукту на основі реклами задля отримання грошової винагороди для подальшого розвиту продукту та оплати заробітної плати працівникам	Точкова комерціалізація продукту; Введення реклами; Ведення додаткових коштів у проект задля його подальшого розвитку.

Таблиця 4.9. Ступеневий аналіз конкуренції на ринку

№	Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1	Тип конкуренції: монополістична	Товар від кожної компанії на ринку, являється недосконалим замінником товару, реалізованого іншими фірмами; На ринку є умови для входу та виходу; Ціна корелює між суперниками;	Розробка продукту з характеристиками, які покривають сфери вживання що не покривають інші товари-замінники; Кореляція цін у відповідності до товарів замінників; Різни типи ліцензій.
2	Рівень конкурентної боротьби: світовий	Всі продукти замінники розроблялись інтернаціональними командами з різних	Вихід на ринок збуту продукту з клієнто-необхідною функціональністю; Налагодження маркетингу на основних Інтернет ресурсах задля охоплення

		куточків світу, продукти не належать до певної держави, а належать команді розробників	великої кількості потенційних користувачів; Надання бета-версій продукту.

Закінчення таблиці 4.9

3	Галузева ознака: внутрішньогалузева	Даний тип продукту може використовуватися тільки у сфері розробки IT додатків \ продуктів	Надання зручного, інтуїтивно зрозумілого інтерфейсу; Підтримка всім відомих методів взаємодії з середовищем розробки; Наявність документації та он-лайн підтримки.
4	Конкуренція за видами товарів: товарно-видова	Дана конкуренція – конкуренція між товарами одного виду.	Впровадження функціональності яка відсутня у товарів-замінників; Спрощення інтерфейсів; Надання підтримки.
5	Характер конкурентних переваг: цінова та не цінова	Цінові переваги – точкова комерціалізація; Не цінова – надання функціональності, що відсутня у товарах-замінниках.	Надання платних ліцензій лише на критично важливу функціональність для клієнта з певним строком підтримки, що зазначена у відповідній ліцензії; Впровадження унікальної функціональності.
6	За інтенсивністю: марочна	Наявність унікального знаку що відрізняє даний продукт від продуктів-замінників	Впровадження власної назви та власного знаку.

Таблиця 4.10. Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Google	Animal detector	Google Play, Apple Store	Форуми, точки продажу додатку	Продукція компанії Animal detector

Висновки	Прямі конкуренти намагаються сконцентруватися на інших напрямках свої продуктів.	Потенційні конкуренти мають дуже специфічну клієнтуру до якої зазвичай входять корпоративні клієнти. Конкуренція може відбутися тільки продажах на корпоративний рівень клієнтів	Постачальники диктують умови збереження даних, які захищають приватність користувачів. Також постачальники не дають змогу зловживати етичними нормами.	Клієнти можуть диктувати умови на ринку тільки через повідомлення на форумах або в полі відгуків в точках продажу додатку.	Можливість введення стандартизації роботи алгоритмів розпізнавання звуків тварин

Проаналізувавши можливості роботи на ринку з огляду на конкурентну ситуацію можна зробити висновок: оскільки кожний з існуючих продуктів не впливає у великій мірі на поточну ситуацію на ринку в цілому, кожний з існуючих продуктів має свою специфічну сферу використання та свої позитивні та негативні сторони щодо рішення певних типів задач, то робота та вихід на даний ринок є можливою і реалізованою задачею.

Для виходу на ринок продукт повинен мати функціонал що відсутній у продуктів-аналогів, повинен задовольняти потреби користувачів, мати необхідний та достатній функціонал з конфігурування, підтримку зі сторони розробників та можливість розробки спеціального функціоналу за відповідною ліцензією.

Таблиця 4.11. Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспроможності	Обґрунтування
1	Прагматичність	Через запуск стартапу система буде не дуже складно з точки зору архітектури перший час. Через певний період із додаванням функціоналу та оптимізації алгоритмів роботи програмний код буде все складнішим. Такий етап наступить не раніше одного року постійної роботи над проектом.
2	Зручність	Оскільки стартап розробляється на багатьох платформах з різною шириною екранів, то зручність використання системи на різних пристроях буде відігравати не малу роль у спроможності конкурувати з іншими гравцями ринку.

3	Швидкість роботи	Швидкість роботи відіграє велику роль для користувачів, оскільки вони не будуть готові чекати декілька хвилин на виведення результату роботи додатку.

Продовження таблиці 4.11

4	Оптимізація	Якщо додаток буде дуже часто видавати помилки при роботі, то користувачі не будуть вважати додаток надійним.
5	Налаштування під користувача	Різні люди мають різні звички, які вони використовують, наприклад, якщо є люди, які люблять працювати за додатком де є темні кольори, а є такі люди, які люблять світлі кольори. Можливість редагувати зовнішній вигляд додатку надає значну перевагу серед конкурентів.
6	Відкритість вихідного коду	При наявності вихідного коду будь-який продукт має перспективи розвиватися у багатьох напрямках, особливо таких, які можуть бути неочевидні на перший погляд.
7	Приватність	В останні роки приватність людей та інформація щодо них все частіше зловживається шахраями або великими корпораціями, які потребують погодження з умовами доступу до приватної інформації та її обробки..
8	Технічна підтримка	Якщо технічна підтримка компанії буде працювати своєчасно та швидко, то це допоможе зберегти репутацію компанії на відміну від конкурентів, де їй не приділяють увагу.
9	Документація	Будь-який додаток, особливо якщо він має новий функціонал, повинен бути добре роз'яснений своїм користувачам та як його можна використовувати, щоб не мати проблем при подальшій роботі з ним.
10	Ціна	Чим дешевше товар, тим більше шансів що його можуть купити, особливо якщо він працює краще ніж конкурентний товар.

Таблиця 4.12. Порівняльний аналіз сильних та слабких сторін системи стартап-проекту

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з запропонованим						
			-3	-2	-1	0	+1	+2	+3

1	Прагматичність	15	Google Voice	Sound reader	Animal detector				
2	Зручність	8			Animal detector	Sound reader	Google Voice		
3	Швидкість роботи	5					Sound reader	Google Voice	Animal detector
4	Оптимізація	5					Sound reader	Google Voice	Animal detector
5	Налаштування під користувача	10				Animal detector	Sound reader	Google Voice	

Закінчення таблиці 4.12

6	Відкритість вихідного коду	20	Google Voice	Sound reader	Animal detector				
7	Приватність	20	Google Voice	Sound reader	Animal detector				
8	Технічна підтримка	15			Animal detector	Sound reader	Google Voice		
9	Документація	15	Sound reader	Google Voice	Animal detector				
10	Ціна	15		Sound reader r	Animal detector	Google Voice			

Таблиця 4.13. SWOT аналіз стартап-проекту

Сильні сторони (S):	Слабкі сторони (W):
Прагматичність системи через її легкість роботи; Простота у використанні; Наявність відкритого вихідного коду; Збереження приватності інформації користувача.	Неоптимізованість алгоритму; Швидкість роботи системи; Зручність.

Можливості (О):	Загрози (Т):
Зворотній зв’язок з клієнтурою компанії для спроможності розвивати проект в інші напрямки.	Автономна робота алгоритмів на електронному пристрої; Складність роботи алгоритму при невиявлених випадках використання додатку.

Таблиця 4.14. Альтернативи ринкового впровадження стартап-проекту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Безкоштовне надання певного функціоналу у користування споживачам на обмежений термін	Головний ресурс – люди, даний ресурс - наявний	2-3 місяці
2	Реклама	Залучення власних коштів для реклами товару	1-2 місяці
3	Написання статей та опис товару на відомих ресурсах	Головний ресурс – час, даний ресурс - наявний	2-3 тижні
4	Презентація товару на хакатонах й інших ІТ заходах	Ресурс – час та гроші для участі, наявні	1-3 місяці

4.4 Розроблення ринкової стратегії проекту

Таблиця 4.15. Вибір цільових груп потенційних споживачів

№	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	Студенти, які навчаються за денною формою навчання	Присутня	Середній	Майже відсутня	Легка
2	Екологи, які досліджують тварин	Відсутня	Низький	Відсутня	Важка
3	Туристи, які знаходяться в іноземній державі та слідкують за тваринами	Присутня	Середній	Присутня	Середня
Які цільові групи обрано: 1, 3					

Відповідно до проведеного аналізу можна зробити висновок, що підходящою цільовою групою для розповсюдження даного програмного продукту є працівники студенти, викладачі та елементи вищої освіти в цілому Відповідно до стратегії охоплення ринку збуту товару обрано стратегію масового маркетингу, оскільки для масової аудиторії в цілому надається стандартизований продукт з можливістю розширення функціональності за домовленістю (відповідно до ліцензії).

Таблиця 4.16. Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Надання функціональності що відсутня у товарів-замінників, підтримка клієнтів	Проведення реклами, освітлення унікальної функціональності через інтернет ресурси та інші канали, контакт напряму з споживачами; формування лояльності і прихильності споживачів	Зниження ступеню замінності товару; Прихильність клієнтів; Відмітні властивості товару; Відмітні характеристики товару;	Стратегія диференціації

Таблиця 4.17. Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, які?	Стратегія конкурентної поведінки
Ні, оскільки є товари-заміннники, але дані товари замінники не мають деякого необхідного функціоналу	Так, ціль компанії знайти нових споживачів та, частково, забрати існуючих у конкурентів задля задоволення потреб останніх	Компанія частково копіює характеристики товару конкурента, основна ціль компанії розробка нового унікального функціоналу, з підтримкою основного	Стратегія заняття конкурентної ніші

Таблиця 4.18. Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
1	Зручність	Диференціація	Спроможність економити час на вивченні навчального матеріалу	Спроможність легко списувати конспект
2	Відкритість вихідного коду	Диференціація	Перспектива розвитку проекту	Розвиток в науці
3	Документація	Заняття конкурентної ніші	Можливість краще дізнаватися про тварин в іноземній державі	Ідентифікатор

Відповідно до проведеного аналізу можна зробити висновок, що стартап-компанія вибирає як базову стратегію розвитку – стратегію диференціації, як базову стратегію конкурентної поведінки – стратегію заняття конкурентної ніші.

4.5 Розроблення маркетингової програми стартап-проекту

Таблиця 4.19. Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Дізнавання інформації в іноземній державі	Можливість дізнаватися про тварину по звуку	Розпізнавання тварин по звуку
2	Передавання моніторингу популяції	Можливість переводити виділяти кількість звуків	Відсутність аналогів
3	Дослідження щодо тварин	Можливість розвивати науку та вписати своє ім’я в історію	Відкритий вихідний код

Таблиця 4.20. Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
1. Товар за задумом	Система розпізнавання рукописних текстів		
2. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх/Тл/Е/Ор
	Зручність	Нм	Е
	Швидкість роботи	Нм	Тх
	Оптимізація	Нм	Тх
	Ціна	Нм	Е
	Документація	Нм	Тл
	Технічна підтримка	Нм	Тх
	Приватність	Нм	Тх
	Налаштування під користувача	Нм	Ор
3. Товар із підкріпленням	До продажу: наявна повна документація, акції на придбання декількох ліцензій, знижки для певних сегментів на покупку товару		
	Після продажу: додаткова підтримка спеціалістів налаштування, підтримка з боку розробника		
За рахунок чого потенційний товар буде захищено від копіювання: захист інтелектуальної власності, патент			

Таблиця 4.21. Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
500 – 1000 грн./од.	1000 – 2000 грн./од.	6 000 – 15 000 грн./міс.	300 – 700 грн./од.

Таблиця 4.22. Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Студенти будуть купувати продукт групами. Екологи будуть купувати товар на корпоративному рівна. Туристи будуть купувати товар поодинці.	Можливість скачувати додаток в любий час, у будь-якому місці.	2 рівня (посередник + клієнт)	Роздріб

Таблиця 4.23. Концепція маркетингових комунікацій

№	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Студентство	Форуми, приватні зустрічі з компанією розробником	Навчання	Показати можливість користування продукцією для клієнта	Рекламне звернення спрямовано до потенційних клієнтів, де показуються плюси користування системою
2	Екологи		Дослідження	Показати перспективу користування системою	
3	Туристи		Туризм	Показати можливість користування продукцією для клієнта	

Як результат було створено ринкову (маркетингову) програму, що включає в себе визначення ключових переваг концепції потенційного товару,

опис моделі товару, визначення меж встановлення ціни, формування системи збуту та концепцію маркетингових комунікацій.

Висновки по розділу

В четвертому розділі описано стратегії та підходи з розроблення стартап-проекту, визначено наявність попиту, динаміку та рентабельність роботи ринку, як висновок було вказано що існує можливість ринкової комерціалізації проекту. Розглянувши потенційні групи клієнтів, бар’єри входження, стан конкуренції та конкурентоспроможність проекту було встановлено що проект є перспективним. Розглянуто та вибрано альтернативу впровадження стартап-проекту та доведено доцільність подальшої імплементації проекту.

ВИСНОВКИ

Після того як був проведений аналіз у першому розділі, в котрому ми розглянули три програми, в яких у першій програмі було розроблено програму яка використовує random forest, навчені ймовірності, отримані шляхом підбору шаблонів специфічних для видових спектрограм. У другій програмі Переможець рішення використовував ці результати як вихідну точку, але ввів додатковий набір функцій, статистично виведених з аудіо файлів. Однак рішення яке було використано у третій програмі - показало, що метод з використанням згорткової нейронної мережі, де вхід до мережі був сегментами спектрограми, обчисленого з звукових файлів. Звукові файли попередньо оброблявся шляхом вилучення двох звукових класів із кожного аудіо файлу. Ці сигнали були приблизно однаково довгими. Після цієї перемоги було сказано що згорткові нейронні мережі, навчені спектральним даними, обчисленими із звукозаписів, можуть перевершити інші сучасні системи.

У другому розділі було проведено аналіз що до нейронної мережі в якому ми дізналися - якщо об'єднати два або більше штучних нейронів, то ми отримаємо штучну нейронну мережу. Один штучний нейрон має нульові шанси у вирішенні реальних проблем, в той самий час як штучні нейронні мережі можуть вирішити реальні проблеми. Фактично штучні нейронні мережі здатні вирішувати складні реальні проблеми шляхом обробки інформації в їх основних будівельних блоках (штучних нейронах) в нелінійному, розподіленому, паралельному та місцевому вигляді.

Також було проаналізовано методи класифікації сигналу після чого стало ясно що для класифікації, спочатку потрібно вирішити що нам потрібно класифікувати. З цього випливає аж 3 типи класифікації, обравши потрібний тип класифікації ми потрапляємо нової проблеми яку було проаналізовано – виділення признаків. Також було проаналізовано сильні сторони потрібної нам для розробки нейронної мережі мови програмування.

У третьому розділі було проведено розробку нейронної, та навчання її за допомогою потрібної інформації. Для того щоб навчити нейронну мережу, спочатку треба аудіо сигнал перетворити на спектральне представлення часу. Але цього перетворення буде недостатньо для навчання, тому ми використовували методи збільшення набору даних які допомогли нейронній мережі підвищити точність результатів. З метою покращення рівня зближення нейронної мережі ми збільшили навчальні зразки шляхом поєднання кожного зразка з іншим зразком одного класу, що дозволяє мережі одночасно переглядати більш релевантні дані. Зразки також додатково поєднуються з трьома випадковими сегментами шуму, щоб зробити мережу більш шумовою інваріантністю, і тому краще узагальнювати. Було використано 33 200 записів і розділених на загальнодоступний навчальний набір із 24 600 записів та приватний або прихований тестовий набір близько 8600 записів. У цьому наборі даних є 999 різних видів. Записи записуються ентузіастами пташиних пісень у полі, і тому можуть містити інші види, що мають різну довжину (секунди до пів години), і містять різну кількість шуму. У деяких записах є люди, які говорять. Після навчання даної нейронної мережі було показано результати які задовольняли очікування.

Четвертий розділ у якому було розроблено стартап проект визначено наявність попиту, динаміку та рентабельність роботи ринку, як висновок було вказано що існує можливість ринкової комерціалізації проекту. Розглянувши потенційні групи клієнтів, бар’єри входження, стан конкуренції та конкурентоспроможність проекту було встановлено що проект є перспективним. Розглянуто та вибрано альтернативу впровадження стартаппроекту та доведено доцільність подальшої імплементації проекту.

ПЕРЕЛІК ПОСИЛАНЬ

1. Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. Arxiv, 2016.

2. Dan Stowell, Mike Wood, Yannis Stylianou, and Hervé Glotin. Bird detection in audio: a survey and a challenge. 2016.

3. Elias Sprengel, Martin Jaggi, Yannic Kilcher, and Thomas Hofmann. Audio Based Bird Species Identification using Deep Learning Techniques. 2016

4. Jaderick P. Pabico, Anne Muriel V. Gonzales, Mariann Jocel S. Villanueva, and Arlene a. Mendoza. Automatic identification of animal breeds and species using bioacoustics and artificial neural networks. arXiv preprint, pages 1–17, 2015.

5. Peter Jancovic and Munevver Kokuer. Acoustic recognition of multiple bird species based on penalised maximum likelihood. IEEE Signal Processing Letters, 22(10):1–1, 2015.

6. Jason Wimmer, Michael Towsey, Paul Roe, and Ian Williamson. Sampling environmental acoustic recordings to determine bird species richness. Ecological Applications, 23(6):1419–1428, 9 2013.

7. Alexis Joly, Herve Goeau, Herve Glotin, Concetto Spampinato, Pierre Bonnet, Willem-Pier Vellinga, Robert Planque, Andreas Rauber, Robert Fisher, and Henning Müller. LifeCLEF 2014: Multimedia Life Species Identification Challenges. (ii):229–249, 2014.

8. Mario Lasseck. Bird song classification in field recordings: Winning solution for NIPS4B 2013 competition. Proc. of int. symp. Neural Information Scaled . . ., pages 1–6, 2013.

9. Herve Goeau, Herve Glotin, Willem Pier Vellinga, Robert Planque, Andreas Rauber, and Alexis Joly. LifeCLEF bird identification task 2015. CEUR Workshop Proceedings, 1391, 2015.

10. Yann LeCun, Yoshua Bengio, and Hinton Geoffrey. Deep learning. Nature Methods, 13(1):35–35, 2015.

11. Уоссермен Ф. Нейрокомпьютерная техника: Теория и практика. Мир. 1992. 240 с.

12. Agoston M.K. Computer graphics and geometric modeling: Implementation and algorithms. 2005. P. 290-306. DOI: 10.1007/b138805.

13. Amosov O.S. High-speed neurofuzzy algorithms for filtering the mobile object trajectory parameters. 2016. P. 389-392.

14. Jang-Shing Roger Jang. ANFIS: adaptive-network-based fuzzy inference system. 1993. Vol. 23(3). P. 665-685. DOI: 10.1109/21.256541.

15. Ouyang W. Joint Deep Learning for Pedestrian Detection. 2013. P. 2046-2063. DOI: 10.1109/ICCV.2013.257.

16. Enzweiler M. Monocular Pedestrian Detection: Survey and Experiments. 2009. Vol. 31(12). P. 2169-2195. DOI: 10.1109/TPAMI.2008.260.

17. Поляков А.С., Корнага Я.І. Виділення звуку птаха в живій природі за допомогою нейронної мережі. Вчені записки Таврійського Національного Університету Імені В.І. Вернадського. Том 29(68) №5 2018 частина 2. С. 3640.

ДОДАТКИ

ДОДАТОК А

Схема класів фрагментів звуку

ДОДАТОК Б

Схема класів списку звуків

ДОДАТОК В

Схема роботи нейронної мережі

ДОДАТОК Г

База даних

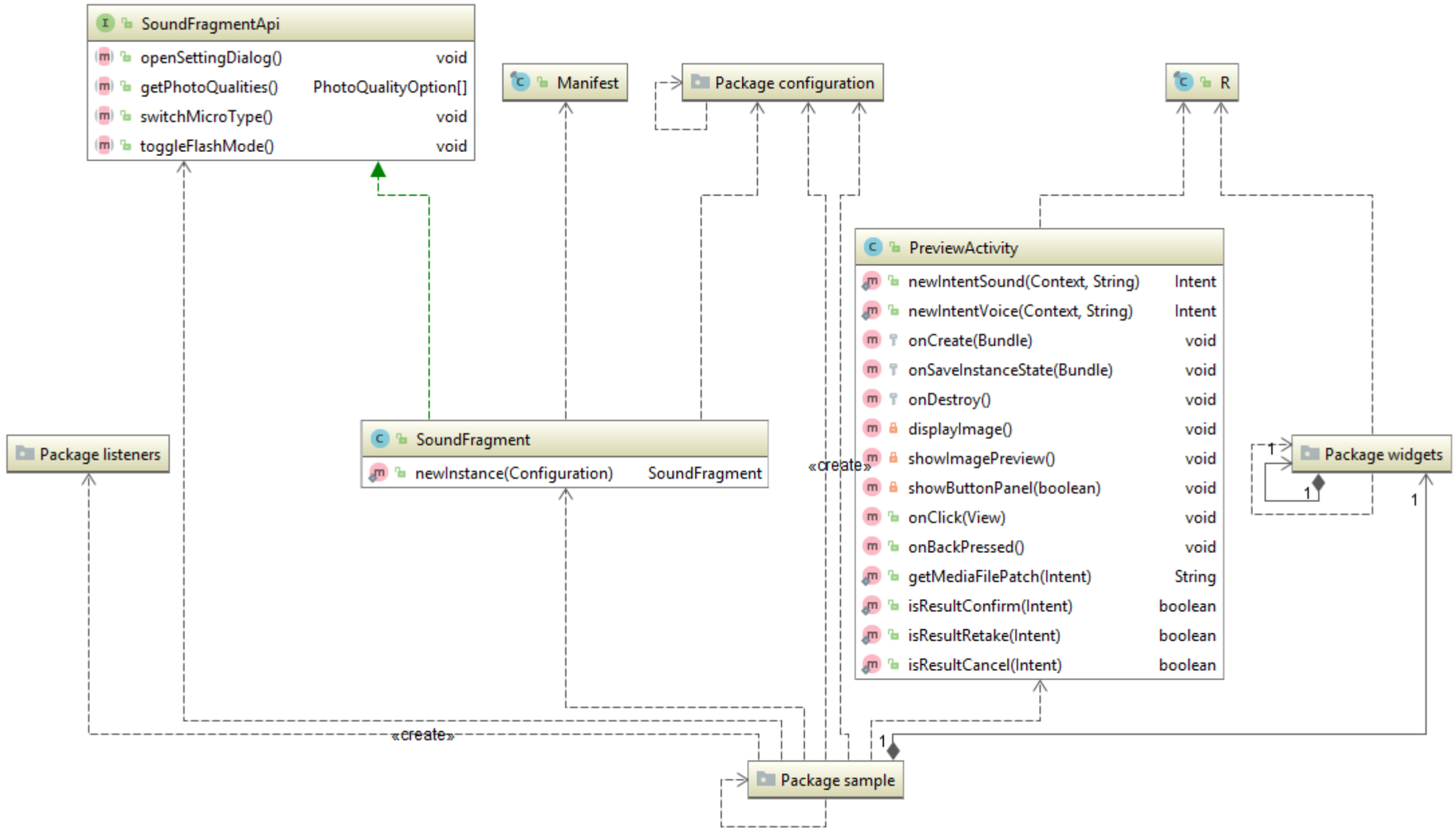
ДОДАТОК Д

Типові спектрограми тварин

ДОДАТОК Е

Зміна бінарного зображення

Схема класів фрагментів звуку

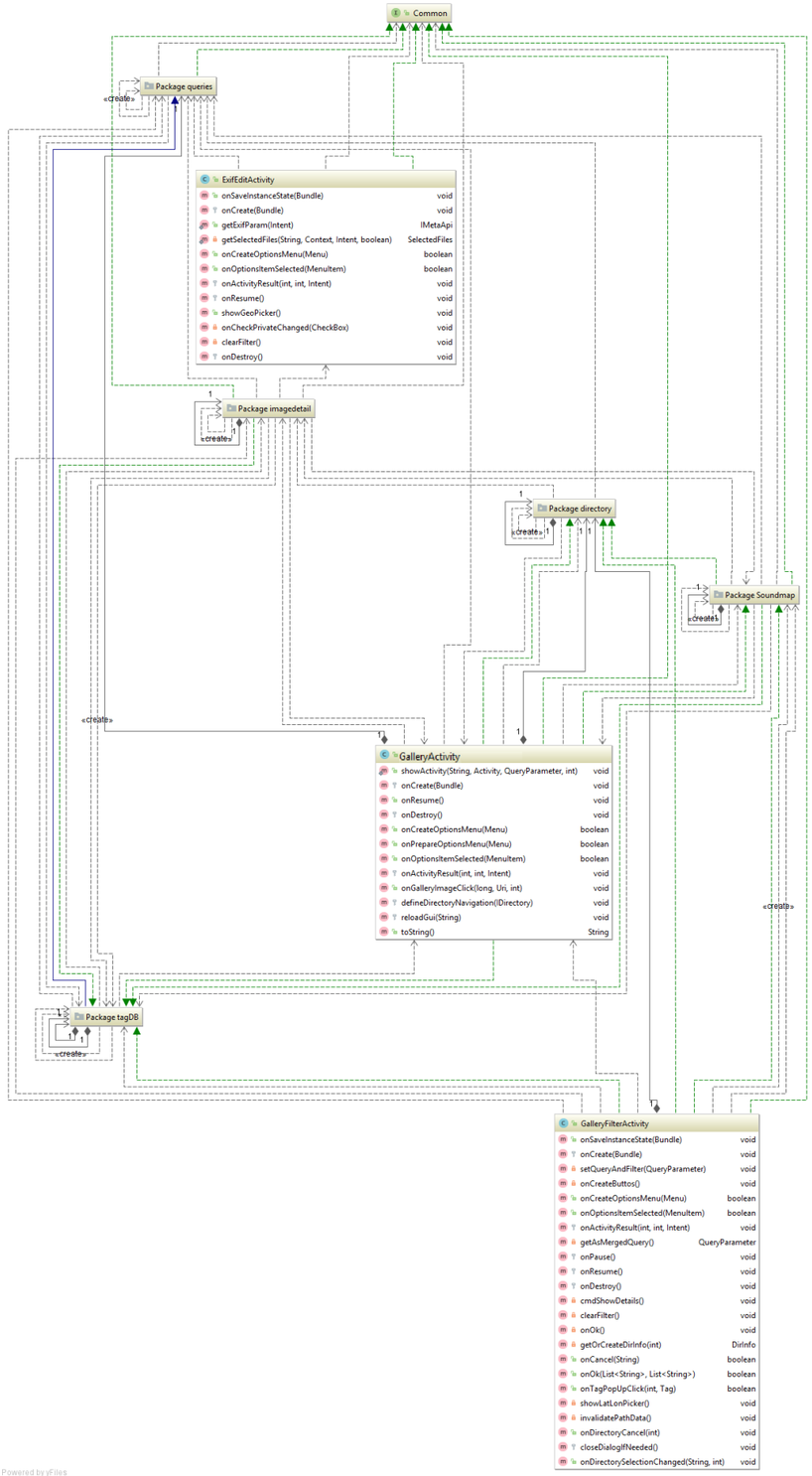


Демонстраційний плакат №__
до магістерської дисертації на тему
„Система розпізнавання звуків живої природи за допомогою нейронної мережі”

Розробив: _____

Прийняв: _____

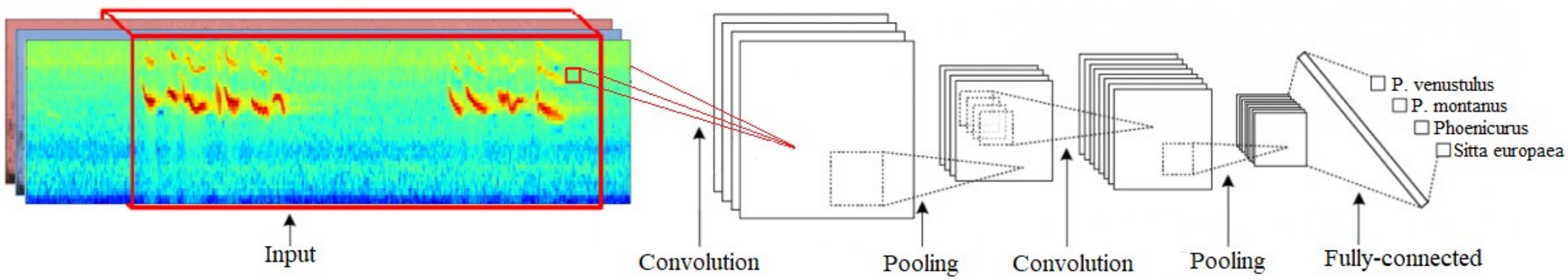
Схема класів списку звуків



Демонстраційний плакат №__
до магістерської дисертації на тему
„Система розпізнавання звуків живої природи за допомогою нейронної мережі”

Розробив: _____
Прийняв: _____

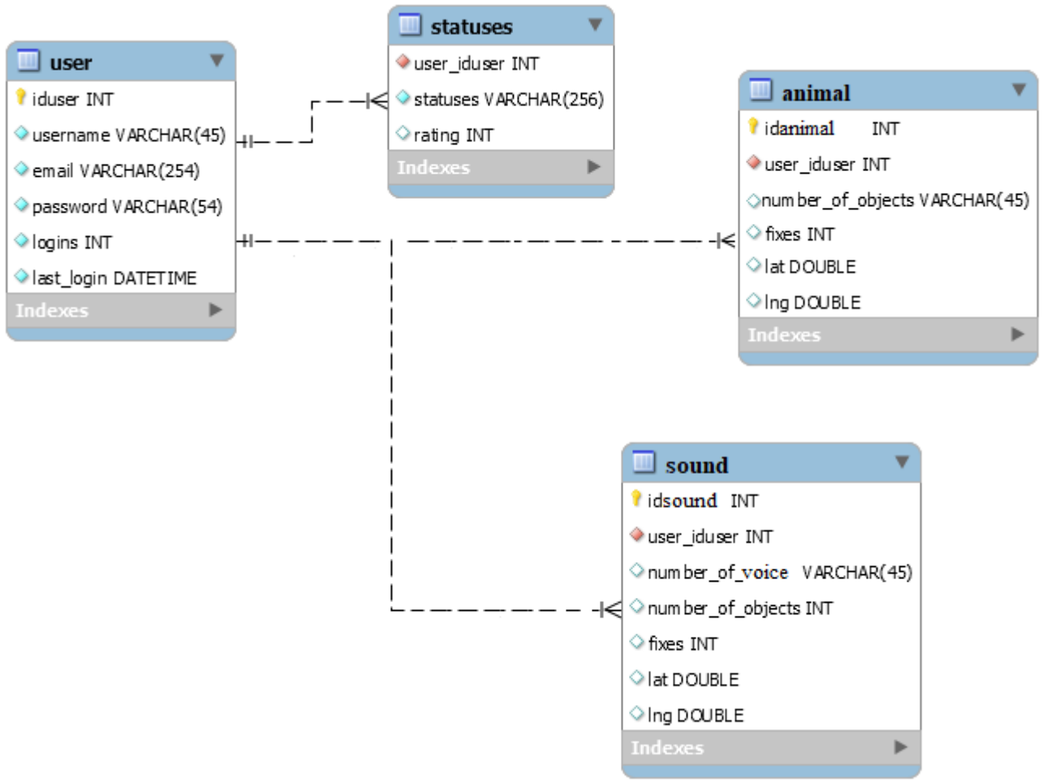
Схема роботи нейронної мережі



Демонстраційний плакат №__
до магістерської дисертації на тему
„Система розпізнавання звуків живої природи за допомогою нейронної мережі”

Розробив: _____
Прийняв: _____

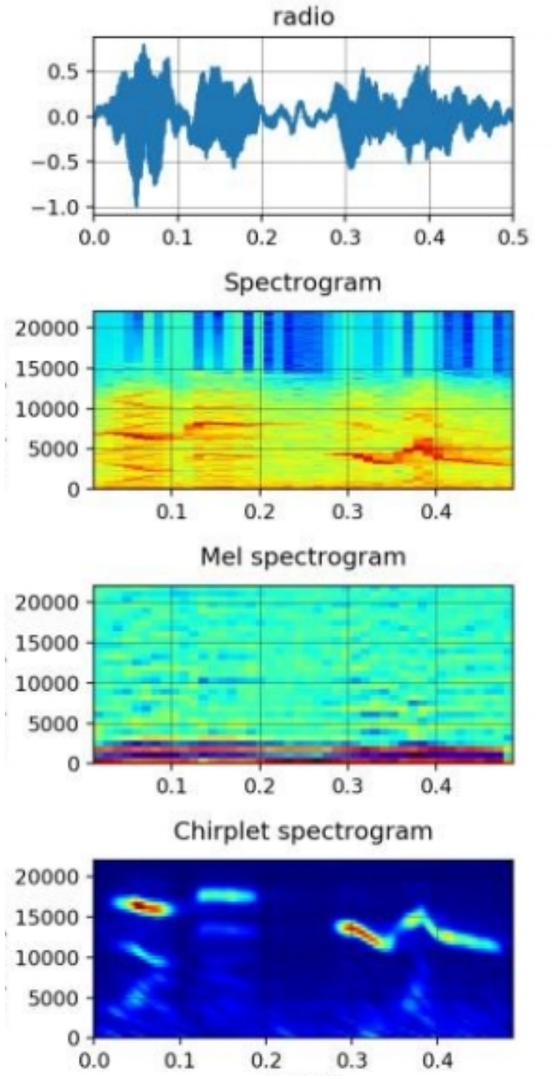
База даних



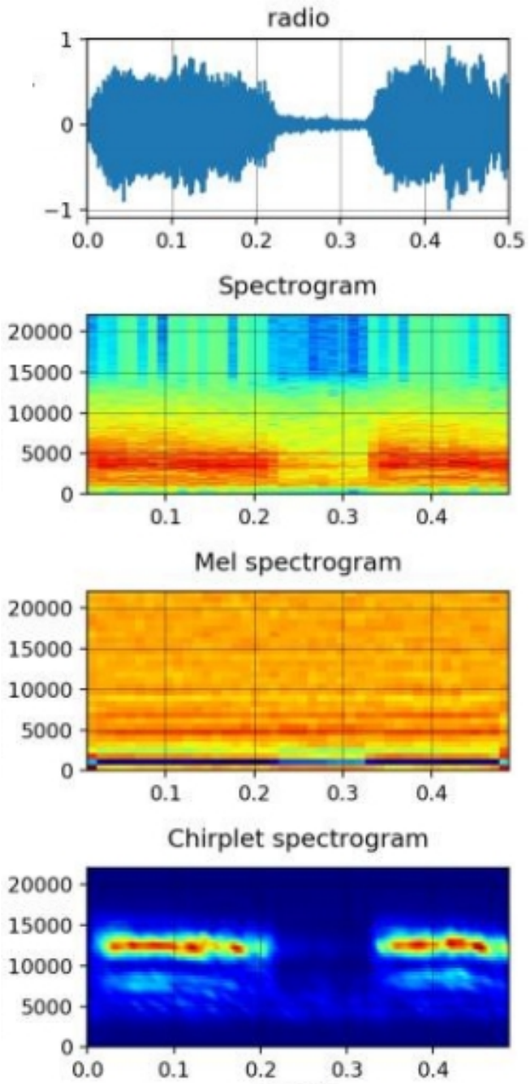
Демонстраційний плакат №__
до магістерської дисертації на тему
„Система розпізнавання звуків живої природи за допомогою нейронної мережі”

Розробив: _____
Прийняв: _____

Типові спектрограми тварин



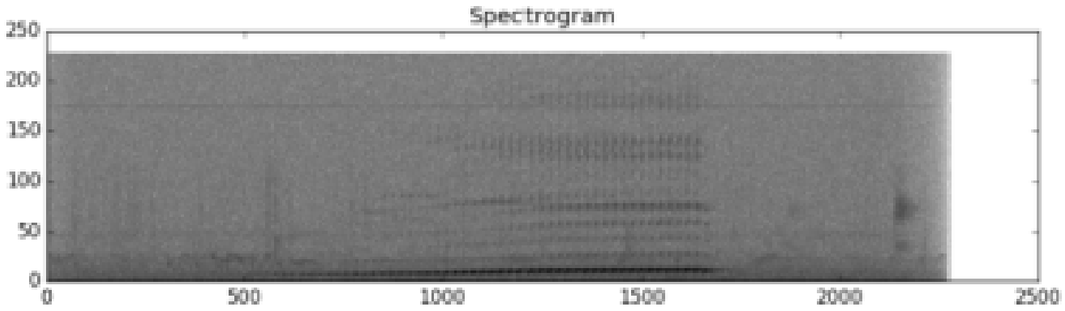
синиця велика

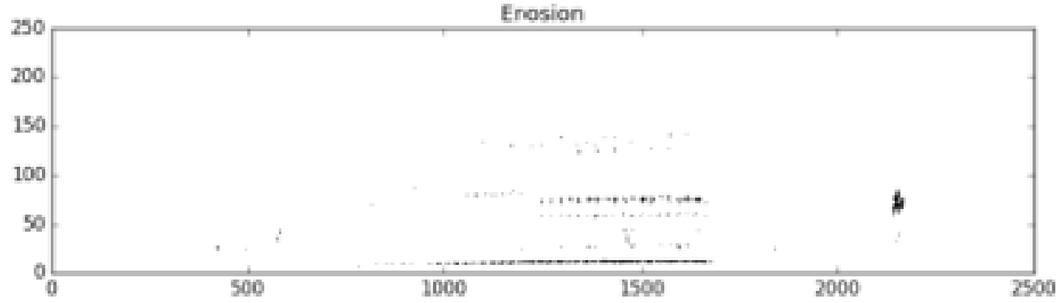


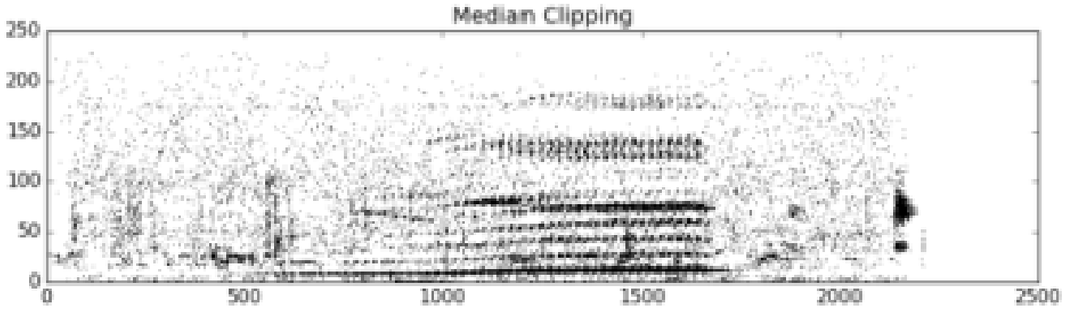
Горобець польовий

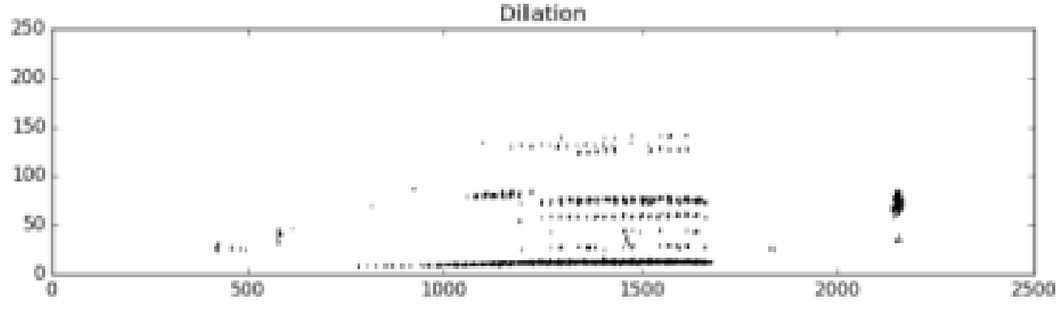
Демонстраційний плакат №__
до магістерської дисертації на тему
„Система розпізнавання звуків живої природи за допомогою нейронної мережі”
Розробив: _____
Прийняв: _____

Зміна бінарного зображення









Показано, як бінарне зображення змінюється після кожного кроку : (Spectrogram) спектрограмма сигналу, використовуваного як еталон, (Median Clipping) бінарне зображення після медіанного відсікання амплітудної спектрограми, (Erosion) бінарне зображення після виконання бінарної ерозії, і (Dilation) бінарне зображення після виконання бінарного розширення. Потім маска виходить з остаточного бінарного зображення шляхом перевірки того, які стовпці містять хоча б одне нульове значення.

Демонстраційний плакат №__
до магістерської дисертації на тему
„Система розпізнавання звуків живої природи за допомогою нейронної мережі”

Розробив: _____
Прийняв: _____