Національний університет ”Львівська політехніка”
Інститут комп’ютерних наук та інформаційних технологій
Кафедра систем штучного інтелекту
Спеціальність 122 «Комп’ютерні науки»

ПОЯСНЮВАЛЬНА ЗАПИСКА

до магістерської кваліфікаційної роботи на тему:

«Порівняльний аналіз методів аугментації даних для різних модальностей»

Студента(ки) групи КНСШ-22 Бохонко А.В.

Керівник роботи _____ (__Мельникова Н. І.___)
(підпис)

Консультанти _____ (_____)
(підпис)

_____ (_____)
(підпис)

Завідувач
кафедри СШІ _____ (__Шаховська Н. Б.___)

Львів - 2023

Національний університет “Львівська політехніка”

Інститут комп’ютерних наук та інформаційних технологій

Кафедра систем штучного інтелекту

Спеціальність 122 «Комп’ютерні науки»

ЗАТВЕРДЖУЮ:

Завідувач кафедри СШІ _____

“____” _____2023р.

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Бохонко Андрію Віталійовичу

1. Тема роботи: «Порівняльний аналіз методів аугментації даних для різних модальностей» затверджена наказом університету № 4420-4-06 від 13.10.2023

2. Термін подання закінченої роботи: _30.11.2023 _____

3. Вихідні дані до (проекту) роботи: використовувались три датасети з простору Kaggle різних модальностей: текст, зображення, аудіо

4. Зміст розрахунково-пояснювальної записки: (перелік питань, що належить розробити): аналітичний огляд літературних та інших джерел, системний аналіз та обґрунтування проблеми, методи та засоби вирішення проблеми, практична реалізація.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень): зображення результатів реалізації рішення.

6. Консультанти по роботі, із зазначенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Консультант з іноземної мови			

7. Дата видачі завдання 10.03.2023

Керівник роботи _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломної роботи	Термін виконання етапів роботи	Примітка
1	Огляд літератури	10.03.2023 – 03.04.2023	
2	Створення архітектури	03.04.2023 – 15.05.2023	
3	Розробка алгоритмічного та програмного забезпечення	15.05.2023 – 11.09.2023	
4	Експериментальні дослідження	11.09.2023 – 06.11.2023	
5	Оформлення записки	06.11.2023 – 30.11.2023	

Керівник роботи _____
(підпис)

Студент _____
(підпис)

АНОТАЦІЯ

Магістерська кваліфікаційна робота виконана студентом групи КНСШ-12 Бохонко Андрієм Віталійовичем. Тема “Порівняльний аналіз методів аугментації даних для різних модальностей”. Робота направлена на здобуття ступеня магістр за спеціальністю 122 «Комп’ютерні науки».

Об’єктом дослідження є процеси прогнозування у випадку коротких наборів табличних даних.

Предметом досліджень є методи аугментації даних для різних модальностей.

Досягнення мети відбувається перш за все з дослідження існуючого інструментарію машинного навчання та методів аугментації даних для різних модальностей. Подальша розробка програмного забезпечення для реалізації різних методів аугментації даних та моделей машинного навчання для різних модальностей. Апробацію роботи здійснено за допомогою аналізу ефективності різних методів аугментації даних для різних модальностей з використанням метрик якості та статистичних методів.

У результаті виконання дипломної роботи проведено аналіз впливу різноманітних методів аугментації даних на ефективність роботи класифікаторів на різних модальностях.

Загальний обсяг роботи: 92 сторінки, 49 рисунків, 25 посилань.

Ключові слова: модальність даних, класифікація, аугментація даних, розширення вибірки, машинне навчання.

ABSTRACT

Master’s degree work of the student of the group CSAI-22 Bokhonko Andrii Vitaliyovych. The topic is "Comparative analysis of data augmentation methods for different modalities". The work is aimed at obtaining a master's degree in 122 "Computer Science".

The object of research is forecasting processes in the case of short sets of tabular data.

The subject of research is data augmentation methods for various modalities.

Achieving the goal occurs primarily from the study of existing machine learning tools and data augmentation methods for various modalities. Further software development to implement various data augmentation methods and machine learning models for various modalities. Approbation of the work was carried out by analyzing the effectiveness of various methods of data augmentation for various modalities using quality metrics and statistical methods.

As a result of the diploma work, an analysis of the influence of various methods of data augmentation on the effectiveness of classifiers in various modalities was carried out.

The total volume of work: 92 pages, 49 figures, 25 references.

Keywords: data modality, classification, data augmentation, sample expansion, machine learning.

ЗМІСТ

ВСТУП

В сучасному світі машинне навчання стало ключовим інструментом для аналізу та обробки даних в різних сферах, від медицини до транспорту. Однак, щоб досягти високої точності та надійності в прогнозуванні, машинне навчання потребує великої кількості даних. Проте, у багатьох випадках наявність достатньої кількості даних є неможливою, що ускладнює розробку ефективних моделей машинного навчання.

Однією з важливих проблем у машинному навчанні є аналіз коротких наборів даних. У таких випадках, розробка ефективної моделі машинного навчання може бути викликом, оскільки точність таких моделей зазвичай залежить від обсягу даних. Але застосування методів аугментації даних може збільшити обсяг даних, необхідних для ефективного навчання моделей.

Метою роботи є порівняльний аналіз методів аугментації даних для різних модальностей. У рамках роботи будуть досліджені різні методи аугментації даних для табличних даних, зображень та аудіо даних. Дослідження будуть виконані за допомогою різних алгоритмів та моделей машинного навчання. Порівняння результатів досліджень дозволить визначити ефективність різних методів аугментації даних для різних модальностей.

Для досягнення поставленої мети будуть розглянуті наступні завдання:

- Дослідження існуючого інструментарію машинного навчання та методів аугментації даних для різних модальностей.
- Розробка програмного забезпечення для реалізації базових методів аугментації даних та моделей машинного навчання для різних модальностей.
- Порівняльний аналіз ефективності різних методів аугментації даних для різних модальностей за допомогою метрик якості та статистичних методів.

Об’єктом дослідження є процеси прогнозування у випадку коротких наборів табличних даних.

Предметом дослідження є методи аугментації даних для різних модальностей.

Методами дослідження є алгоритми машинного навчання, які використовуються для роботи з різними модальностями.

Апробація результатів роботи. На основі проведених наукових досліджень буде опубліковано наукову статтю по темі магістерської роботи. На даних момент стаття надіслана в редакцію та прийнята до розгляду.

1. АНАЛІТИЧНИЙ РОЗДІЛ

1.1. Методологія пошуку наукових джерел.

У цій роботі проведено аналітичний огляд наукових джерел згідно з стандартизованою методологією PRISMA [1], що містить загальні рекомендації щодо огляду наукових та базових особливостей мета-аналізу.

Методологія PRISMA включає у себе певний порядок відбору досліджень для огляду, критерії включення та виключення, методи пошуку та аналізу даних. Вона дозволяє проводити огляди та мета-аналізи відповідно до встановлених стандартів, що дозволяє зменшити ризик спотворення результатів та покращити якість звітування.

PRISMA рекомендує проводити пошук даних у різних базах даних, детально описувати критерії включення та виключення досліджень, проводити детальний аналіз якості досліджень та рівня доказової бази.

Методологія PRISMA використовується для проведення систематичних оглядів та мета-аналізів наукових досліджень в різних галузях науки. PRISMA дозволяє забезпечити стандартизований та об'єктивний підхід до проведення оглядів та мета-аналізів, що підвищує їх наукову значущість.

Для пошуку релевантних публікацій скористаємось базами даних Scopus та Google Scholar. Ці бази даних є двома з найбільш відомих та використовуваних баз даних у науковому світі.

Scopus - це міжнародна база даних, яка включає більше 75 мільйонів записів з більш ніж 24 тисяч журналів, конференцій та книг з усього світу. Вона охоплює різні наукові галузі, такі як медицина, біологія, інженерія, науки про Землю, соціологія, економіка та інші. База даних Scopus надає можливість для пошуку, перегляду та аналізу наукових статей та конференцій, а також показує метрики цитувань, які вказують на вплив статті та її авторів в науковому світі.

Google Scholar - це безкоштовна база даних, яка містить велику кількість наукових робіт та літератури. Вона дає змогу знайти статті, книги, тези, наукові праці та інші наукові джерела з усього світу. Google Scholar дозволяє пошук по різних наукових галузях та виданнях, а також надає можливість для збереження

та організації знайдених джерел. База даних Google Scholar також надає метрики цитувань та інші дані про наукові джерела.

Основна відмінність між Scopus та Google Scholar полягає в їхніх обсягах та підходах до збору та обробки даних. Scopus фокусується на вибірковому зборі матеріалів відомих наукових видань, тоді як Google Scholar збирає матеріали зі всіх джерел в Інтернеті, включаючи незалежні блоги та веб-сайти авторів. Також Scopus надає точніші та більш детальні метрики цитувань та більш точні дані про журнали, а Google Scholar надає доступ до більшої кількості джерел, включаючи роботи, які не опубліковані у відомих наукових журналах.

Здійснимо пошук релевантних статей у науковій базі даних Scopus за запитом на *Рис. 1.1*:

Рис. 1.1. Початковий запит пошуку релевантних до теми статей.

Для початкової фільтрації публікацій скористаємось ключовими словами по темі роботи. Сферою пошуку *“machine-learing”* або *“deep learning”*. Також додамо як головний термін ключові слова *“data augmentation”*. Вони будуть пов’язані за допомогою оператора AND щоб включити обидва цих терміни до пошуку. Також за допомогою оператора AND додамо фільтрацію за ключовими слова *“small data”, “little data”, “short data”, “limitted data”*. Ці ключові слова відносяться до розміру датасату, тому вони були об’єднані за допомогою оператора OR.

Рис. 1.2. Результати фільтрування за початковим запитом

Для подальшої корекції результаті розширимо наш запит для того щоб робити пошук лише серед заголовків, ключових слів та анотацій.

Рис. 1.3. Результати фільтрування лише серед заголовків, ключових слів та анотації

Отримати 256 публікацій після покращеного пошуку. Для подальшої мінімізації результатів скористаємось фільтрами.

Список фільтрів, які були використані:

- Open access. Обираємо лише відкриті публікації.
- Subject Area. Обираємо публікації з предметної області “Computer Science”.
- Language. Обираємо лише публікації англійською.
- Document Type. Обираємо документи типу “Article” та “Conference paper”.
- Keyword. Обмежимо пошук лише за ключовим словом “Data augmentation”.
- Source Type. Обираємо джерело типу “Journal” та “Conference proceeding”.
- Publication Stage. Обираємо публікації, які вже були видані.
- Year. Обираємо публікації виданні не пізніше ніж 2018.

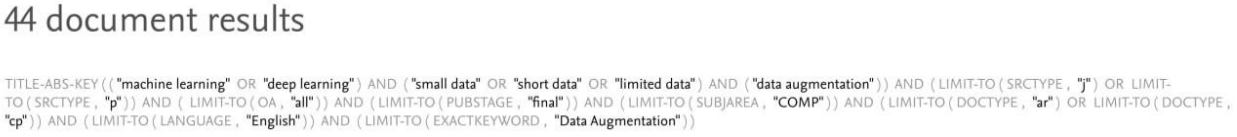


Рис. 1.4. Результати після пошуку з фільтрами

Тепер проведемо пошук публікацій з використанням баз даних Google Scholar. Повторимо запит Scopus у Google Scholar з фільтром по даті публікації.

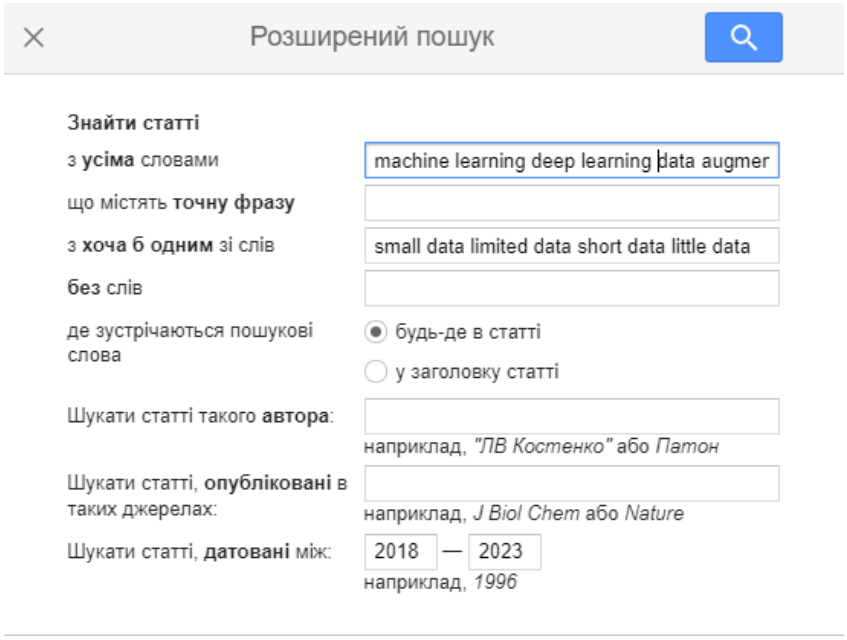


Рис. 1.5. Пошук за допомогою Google Scholar

Після виконання запиту було отримано 22700 публікацій. Оскільки подальших можливостей немає, щоб омежити кількість результатів немає, тому обремо перших 20 публікацій посортованих за релевантністю.

Рис. 1.6. Результати пошуку у Google Scholar

Після проведемо мета-аналізу публікацій з використання бази даних Scopus було відібірано 44 статті та 20 після аналізу з допомогою Google Scholar. Отже 64 публікації будуть аналізуватись з використанням методолгії PRISMA.

Для отримання більш чіткого та якісного розуміння того, як відбираються відповідні тематиці робіт проведемо PRISMA аналіз. Ця технологія передбачає створення схеми на основі встановлених критеріїв, які визначають, які публікації включаються або виключаються. Створена схема показує процес прийняття рішень та визначає кількість наукових джерел, які можуть бути використані для аналізу.

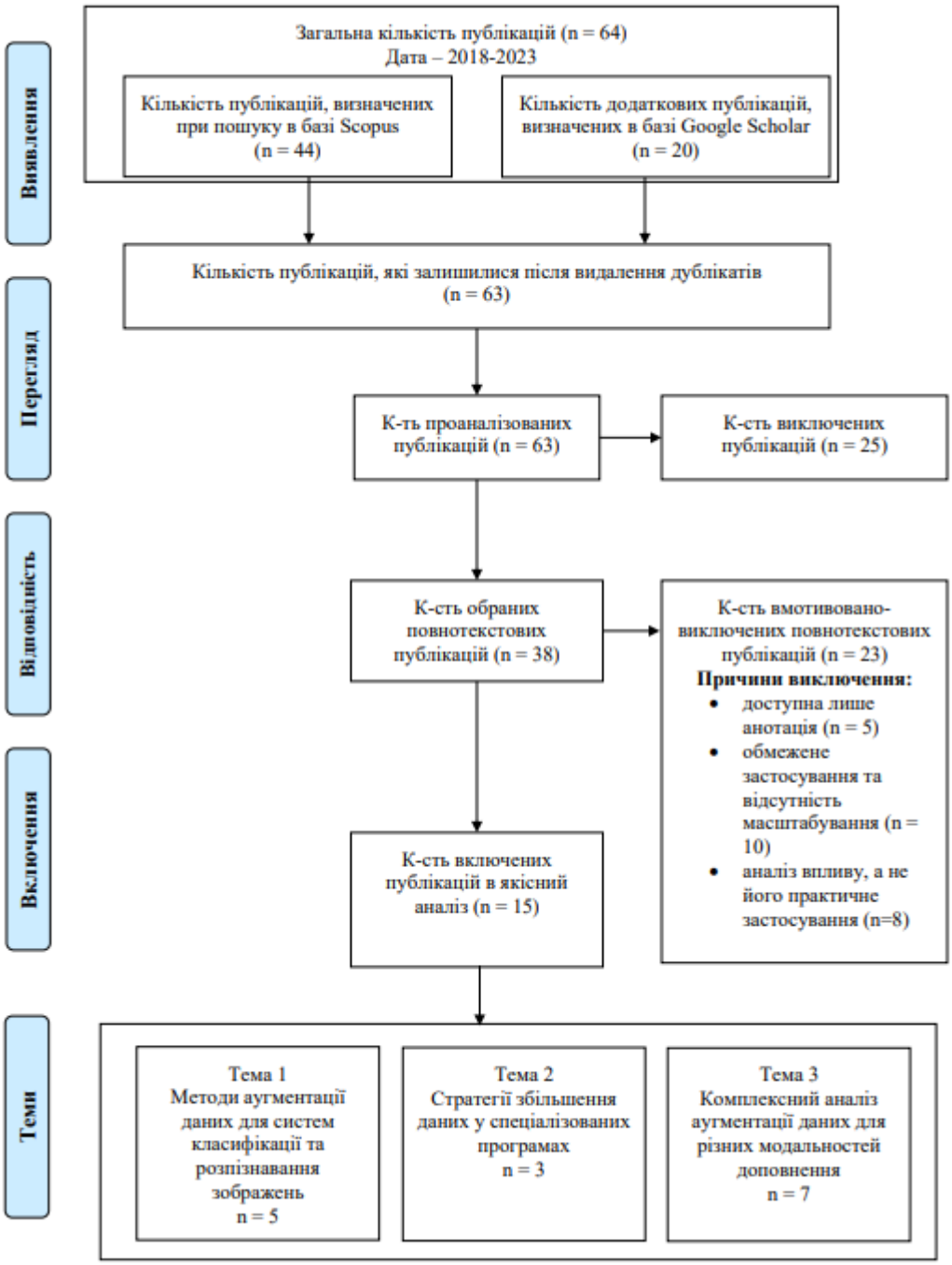


Рис. 1.7. Діаграма PRISMA

1.2. Огляд літературних джерел.

На основі проведеного пошуку літературних джерел за темою магістерської роботи, обрано 15 статей, які безпосередньо стосуються тематики моїх досліджень. Їх зведено у *Табл. 1.1.*

Таблиця 1.1. Огляд існуючих робіт

№	Назва	Інструментарій	Задача	Рік	Посилання
1	Tuning of data augmentation hyperparameters in deep learning to building construction image classification with small datasets	CNN, Logistic regression, Data augmentation	Застосуванні методології налаштування гіперпараметрі в аугментації даних для покращення продуктивності згорткових нейронних мереж у класифікації зображень.	2022	[2]
2	The effects of augmented training dataset on performance of convolutional neural networks in face recognition system	CNN, Data augmentation	Дослідження ефективності методів аугментації даних на прикладі задачі розпізнавання облич.	2019	[3]
3	BiLSTM with Data Augmentation using Interpolation Methods to Improve Early	CNN, LSTM, Logistic Regression, Cubic SVM	Поліпшення раннього виявлення хвороби Паркінсона за	2020	[4]

	Detection of Parkinson Disease		допомогою методів машинного навчання та аугментації даних.		
4	Improving deep learning with generic data augmentation	CNN	Порівняння різних популярних схем аугментації даних та оцінка їх впливу на продуктивність задачі згорткової нейронної мережі.	2018	[5]
5	Efficient Data Augmentation Techniques for Improved Classification in Limited Data Set of Oral Squamous Cell Carcinoma	GAN, CNN	Розробка нового методу аугментації даних, що може збільшити розмір навчального набору, та досягти більшої точності за допомогою глибокого	2022	[6]

			навчання		
6	Deep learning-based data augmentation for hydraulic condition monitoring system	CNN, Data augmentation	Розробка нового методу аугментації даних для збільшення кількості даних, необхідних для моніторингу стану гідравлічної системи.	2020	[7]
7	Data augmentation in natural language processing: a novel text generation approach for long and short text classifiers	EDA	Розробка та оцінка методу генерації тексту, який може поліпшити ефективність класифікаторів для коротких та довгих текстів шляхом аугментації тренувальних даних.	2023	[8]
8	Data augmentation for improving deep learning in image classification	CNN, DNN, GAN.	Порівняння та аналіз різних методів аугментації	2018	[9]

	problem		даних для класифікації зображень, від класичних трансформацій зображень, до передачі стилю та генеративних змагальних мереж.		
9	Data Augmentation for Deep-Learning-Based Multiclass Structural Damage Detection Using Limited Information	CNN, GAN	Дослідження ефективності архітектури згорткової нейронної мережі з використанням синтетичних зображень, створених за допомогою, для багатокласового виявлення пошкоджень бетонних поверхонь	2022	[10]
10	Copypaste: An augmentation method for speech	ResNet model, Noise augmentation,	Перевірка гіпотези про покращення	2021	[11]

	emotion recognition	SER	ефективності SER за допомогою згенерованих конкатенованих висловів (емоційних і нейтральних) під час тренування моделі		
11	An Improved Deep Learning Model of Chili Disease Recognition with Small Dataset	CNN, ResNet	Застосування методу аугментації даних на невеликому наборі даних про здорові та хворі листки перцю.	2022	[12]
12	Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling	LSTM, CNN-LSTM, NBSVM	Виявлення агресії в текстах соціальних медіа за допомогою глибоких нейронних мереж, а також порівнянні їх	2018	[13]

			ефективності з базовою моделлю NBSVM, заснованою на символьних n-грамах.		
13	A survey on Image Data Augmentation for Deep Learning	GAN, DCGAN, CNN, Data augmentation	Систематизація існуючих методів аугментації даних, охоплюючи геометричні перетворення, зміну колірного простору, використання фільтрів, змішування зображень, випадкове стирання.	2019	[14]
14	A kernel theory of modern data augmentation	Data augmentation, Markov process	Загальна модель аугментації як процесу Маркова, з врахуванням виникнення	2019	[15]

			ядер.		
15	A close look at deep learning with small data	CNN, ResNet-20, SGD	Аналіз впливу складності моделі на результуючу продуктивність у задачах з обмеженими тренувальними даними.	2020	[16]

Далі, проведемо детальний аналіз отриманих робіт з *Табл. 1.1*.

1.3. Критичний аналіз літературних джерел.

У статті описується застосування згорткових нейронних мереж у малих наборах даних. У даній статті запропоновано ретельну методологію налаштування гіперпараметрів аугментації даних для глибокого навчання у класифікації зображень будівельного будівництва, особливо для розпізнавання рослинності на фасадах та аналізі конструкцій дахів. Для цього були використані моделі логістичної регресії для аналізу продуктивності згорткових нейронних мереж, навчених за допомогою 128 комбінацій трансформацій зображень.

Ця стаття може бути корисною тим, що вона розглядає методи налаштування гіперпараметрів аугментації даних у глибокому навчанні та їх застосування у класифікації зображень. Це дозволить мені вивчити запропоновану методологію та застосувати за потреби до власного дослідження.

У статті [3] розглядається важливість системи розпізнавання обличчя. Автор описує основні факторами, які ускладнюють розпізнавання. Це може бути напрямок світла, відбивання, емоційні та фізичні зміни в обличчі. Тренування системи з доступними даними в невеликих наборах даних є важливим фактором,

який негативно впливає на продуктивність. Модель згорткової нейронної мережі (CNN) є глибокою архітектурою навчання, що використовується для великих обсягів тренувальних даних. У цьому дослідженні мале число зображень співробітників невеликої компанії було збільшено за допомогою застосування різних фільтрів. Крім того у роботі визначається, які опції аугментації даних мають більший ефект на розпізнавання обличчя.

Дослідження може дати корисну інформацію про те, які методи аугментації є найбільш ефективними для покращення роботи систем розпізнавання облич. Результати дослідження можуть допомогти вам зробити висновки щодо того, які методи аугментації можна використовувати для найкраще використовувати для вирішення цієї задачі. Автори пропонують використання сплайн-інтерполяції та інтерполяції за допомогою кусково кубічних поліномів Ерміта (Pchip) для генерації синтетичних даних. Також досліджується зменшення розмірності ознак для ефективної та оперативної класифікації, з урахуванням обчислювальної складності в реальних мобільних пристроях.

Стаття [4] присвячена проблемі раннього виявлення хвороби Паркінсона (PD) за допомогою методів машинного навчання та аугментації даних для дуже малих наборів даних. В роботі розглядаються виклики, такі як маленький розмір набору даних, дисбаланс класів, перенавчання, високий рівень помилкового виявлення тощо.

Стаття [5] може бути корисною для даної роботи, оскільки вона досліджує аугментацію даних на прикладі дуже малих наборів даних, що може бути актуальним для різних модальностей.

Робота розглядає використання аугментації даних для збільшення обсягу навчальних даних для глибоких штучних нейронних мереж, зокрема згорткових нейронних мереж (CNN). Автори оцінюють різні популярні схеми аугментації даних, щоб допомогти дослідникам приймати обґрунтовані рішення щодо вибору найбільш підходящих методів для своїх наборів даних.

Стаття [6] присвячена вивченню методів аугментації даних для різних доменів, таких як охорона здоров'я, де зібрати великі набори даних є складно

через етичність та приватність пацієнтів. Автори розглядають використання генеративних суперницьких мереж (GAN) для аугментації даних, що може допомогти збільшити точність класифікації зображень у галузі охорони здоров'я.

Задача, яка вирішується у статті - розробка нового методу аугментації даних, що може збільшити розмір навчального набору, та досягти більшої точності за допомогою глибокого навчання. Автори порівнюють ефективність класифікаторів зображень з використанням стандартних методів аугментації та GAN.

Ця стаття може бути корисною для даної роботи, оскільки вона досліджує використання GAN для аугментації даних у галузі охорони здоров'я та демонструє збільшення точності класифікації зображень. Результати дослідження підтверджують перевагу GAN над стандартними методами аугментації, що може бути актуальним для різних модальностей та специфічних областей застосування, де збір великого набору даних є проблематичним.

Стаття [7] акцентує увагу на важливості класифікації аномальних даних, зокрема часових рядів, які збираються з виробничих об'єктів для виявлення несправностей. Глибоке навчання виявило свою ефективність у вирішенні складних задач класифікації, проте потребує великої кількості даних для навчання. Для випадків з обмеженою кількістю даних може застосовуватись аугментація даних. В статті запропоновано новий метод аугментації даних для контролю стану гідравлічної системи.

Задача, яка вирішується у статті - розробка нового методу аугментації даних для збільшення кількості даних, необхідних для моніторингу стану гідравлічної системи. Застосовуючи запропонований метод, автори застосовують модель глибокого навчання для класифікації даних гідравлічної системи.

Ця стаття може бути корисною для даної роботи, оскільки вона розглядає аугментацію даних у контексті гідравлічних систем. Запропонований метод може слугувати основою для адаптації аугментації даних у різних модальностях та інших областях застосування, де збір великої кількості даних є складним або неможливим. Враховуючи успішність запропонованого методу для класифікації

гідравлічних систем, можна сподіватися його корисності й у інших сферах.

У статті [8] розглядається важливість розробки тренувальних даних у випадках машинного навчання, наголошуючи, що вони можуть бути більш важливими, ніж вибір та моделювання класифікаторів. В статті представлено та оцінено метод генерації тексту, який призначений для підвищення ефективності класифікаторів для коротких та довгих текстів. У сфері обробки природної мови (NLP) важливим є встановлення універсальних правил текстових трансформацій, які надають нові мовні шаблони.

Задача, яка вирішується у статті - розробка та оцінка методу генерації тексту, який може поліпшити ефективність класифікаторів для коротких та довгих текстів шляхом аугментації тренувальних даних.

Ця стаття може бути корисною для даної роботи, оскільки вона досліджує метод аугментації даних у контексті обробки природної мови. Метод генерації тексту, запропонований авторами, може бути адаптований для різних модальностей та застосувань, де збір великої кількості даних є складним. Результати дослідження, проведені на 11 різних наборах даних, допомагають зрозуміти, в яких ситуаціях метод може бути підходящим або непридатним, та обговорюють наслідки та закономірності успішного застосування запропонованого підходу на різних типах наборів даних.

Стаття [9] присвячена глибокому навчанню та глибоким нейронним мережам (DNN), зокрема, зосереджуючись на згорткових нейронних мережах (CNN) як основному інструменті для аналізу та класифікації зображень. Одним з найбільш поширених проблем у сфері машинного навчання є недостатня кількість тренувальних даних або нерівномірний баланс класів у наборах даних. Щоб вирішити цю проблему, використовують аугментацію даних.

Задача, яка вирішується у статті - порівняння та аналіз різних методів аугментації даних для класифікації зображень, від класичних трансформацій зображень (поворот, обрізка, зумування, методи на основі гістограм) до передачі стилю та генеративних змагальних мереж (GAN). Автори представляють свій власний метод аугментації даних, заснований на передачі стилю зображення, який дозволяє генерувати нові зображення високої якості на основі змішування

змісту базового зображення з виглядом інших.

Ця стаття може бути корисною для даної роботи, оскільки вона досліджує та порівнює різні методи аугментації даних у завданні класифікації зображень. Зокрема, можна адаптувати запропонований метод аугментації на основі передачі стилю для різних модальностей. Метод перевірено на трьох медичних випадках (діагностика шкірних меланом, аналіз гістопатологічних зображень та дослідження МРТ грудної клітки), де дефіцит даних є однією з найбільш актуальних проблем.

Стаття [10] присвячена проблемі старіння інфраструктури та пошкодженню споруд в результаті стихійних лих. Це спонукало наукове середовище вдосконалити методи моніторингу структурного здоров'я (SHM). Хоча глибоке навчання (DL) активно вивчалося в останній десятиліття для автоматизації інспекції споруд, проблема дефіциту даних у сфері SHM все ще актуальна. Генеративні змагальні мережі (GAN) привернули увагу дослідників як інструмент для аугментації даних, що сприяє покращенню класифікації пошкоджень.

Задача, яка вирішується в статті - дослідження ефективності архітектури згорткової нейронної мережі (CNN) з використанням синтетичних зображень, створених за допомогою GAN, для багатокласового виявлення пошкоджень бетонних поверхонь.

Ця стаття може бути корисною для даної роботи, оскільки вона досліджує використання синтетичних даних, згенерованих з GAN, для поліпшення класифікації пошкоджень за допомогою згорткової нейронної мережі. Можна адаптувати ідеї, представлені в статті, для порівняння методів аугментації даних для різних модальностей, що вивчаються в вашій магістерській роботі. Окрім того, стаття досліджує зв'язок між точністю класифікації та кількістю та різноманітністю синтетичних даних, які використовуються для аугментації.

Стаття [11] присвячена методу аугментації даних, названому Copy-Paste, який розроблений для покращення роботи моделей розпізнавання емоцій за мовленням (SER). Цей метод допомагає частково подолати проблему обмеженої кількості даних у задачах, де збір даних є дорогою та складною процедурою.

Задача, яка вирішується в статті, полягає в перевірці гіпотези про покращення ефективності SER за допомогою згенерованих конкатенованих висловів (емоційних і нейтральних) під час тренування моделі. Щоб перевірити це, автори тестують три схеми CopyPaste на двох глибоких навчальних моделях: однієї, яка навчається самостійно, та іншої, яка використовує передавання навчання від x-vector моделі.

Ця стаття може бути корисною для даної роботи, оскільки вона представляє новий метод аугментації даних, Copy-Paste, який успішно застосовується для задач розпізнавання емоцій за мовленням. Можна вивчити та порівняти цей метод з іншими методами аугментації даних, які використовуються для різних модальностей, і дослідити можливість його адаптації для вашого дослідження. Крім того, результати статті показують, що Copy-Paste працює краще, ніж аугментація шумом, і його спільне використання забезпечує подальше покращення результатів SER.

Стаття [12] присвячена визначенню стану здоров'я перцю через застосування глибокого навчання для підвищення сільськогосподарської продуктивності. Оскільки глибокі навчальні моделі потребують великої кількості даних для ефективної роботи, автори пропонують метод аугментації даних для подолання цієї проблеми.

Задача, яка вирішується в статті, полягає в застосуванні методу аугментації даних на невеликому наборі даних про здорові та хворі листки перцю. Метод геометричної трансформації використовується для збільшення розміру набору даних.

Ця стаття може бути корисною для даної теми роботи, оскільки вона досліджує використання методів аугментації даних для покращення результатів глибоких навчальних моделей у випадку обмеженої кількості даних. Можна цей метод аугментації та порівняти його з іншими методами для різних модальностей. Крім того, результати статті показують покращення середньої точності (97%) для обох моделей (CNN та ResNet-18) при використанні аугментованих та оригінальних наборів даних.

Стаття [13] присвячена виявленню агресії в публікаціях соціальних медіа.

Автори досліджують ефективність глибоких нейронних мереж різної складності для автоматичного визначення агресії в повідомленнях.

Задача, яка вирішується у статті, полягає у виявленні агресії в текстах соціальних медіа за допомогою глибоких нейронних мереж, а також порівнянні їх ефективності з базовою моделлю NBSVM, заснованою на символьних n-грамах.

Ця стаття може бути корисною для даної роботи, оскільки вона досліджує використання глибоких нейронних мереж для аналізу текстів у соціальних медіа та визначення агресії. Ви можете вивчити методи аугментації даних та псевдо-міток, які автори застосовували для покращення результатів своїх моделей. Також може бути корисним порівняння ефективності глибоких нейронних мереж з лінійною базовою моделлю для кращого розуміння можливостей та обмежень різних методів.

Стаття [14] є оглядом методів аугментації даних, які застосовуються для покращення роботи глибоких згорткових нейронних мереж у задачах комп'ютерного зору, особливо в умовах обмеженого обсягу даних, як-от в медичному зображенні.

Основна задача статті - систематизувати існуючі методи аугментації даних, охоплюючи геометричні перетворення, зміну колірного простору, використання фільтрів, змішування зображень, випадкове стирання, аугментацію у просторі ознак, адверсарний тренування, генеративно-суперечливі мережі, перенесення стилів та метанавчання. Особливу увагу приділено методам аугментації, заснованим на генеративно-суперечливих мережах (GAN).

Ця стаття буде корисною для даної роботи, оскільки вона розглядає різні методи аугментації даних, що можуть бути використані для різних модальностей. Вам буде цікаво дізнатися про різні підходи до аугментації та як вони можуть допомогти покращити результати моделей глибокого навчання, а також розширити обмежені набори даних для використання потенціалу великих даних. Окрім того, можуть бути корисними обговорення інших характеристик аугментації даних, таких як аугментація в часі тестування, вплив роздільної здатності, розмір фінального набору даних та послідовне навчання.

Стаття [15] присвячена розробці теоретичного підґрунтя для розуміння аугментації даних, техніки розширення набору навчальних даних за допомогою трансформацій, що зберігають класи.

Задача, що вирішується у статті, полягає в розгляді аугментації даних з двох позицій: спочатку представлено загальну модель аугментації як процесу Маркова, з врахуванням виникнення ядер у цій моделі, а потім досліджується вплив аугментації на класифікатори з ядрами, показуючи, що аугментація даних може бути апроксимована компонентами усереднення ознак першого порядку та регуляризації дисперсії другого порядку.

Ця стаття може бути корисною для даної роботи, оскільки вона пропонує теоретичні рамки для аналізу аугментації даних, розкриваючи шляхи, якими вона впливає на моделі навчання. Отримані результати створюють нові зв'язки між попередніми дослідженнями інваріантних ядер, тангенціальної пропагації та робастної оптимізації. Крім того, стаття надає декілька практичних застосувань, що демонструють корисність розробленої теорії для прискорення робочих процесів машинного навчання, таких як зменшення обчислювального навантаження при навчанні на аугментованих даних та прогнозування ефективності трансформацій до початку навчання.

Стаття [16] описує проведення численних експериментів з різними архітектурами глибокого навчання на наборах даних обмеженого розміру. Згідно з результатами дослідження, автори демонструють, що складність моделі має критичне значення, коли доступно лише кілька зразків кожного класу.

Задача, що вирішується у статті, полягає в аналізі впливу складності моделі на результуючу продуктивність у задачах з обмеженими тренувальними даними.

Ця стаття може бути корисною для даної роботи, оскільки вона розглядає вплив складності моделі на задачі з обмеженим набором даних та показує, що за певних умов низькоскладні архітектури можуть бути кращими, ніж більш сучасні архітектури. Також дослідження підтверджує, що стандартна аугментація даних може суттєво підвищити продуктивність розпізнавання, що свідчить про необхідність розробки більш складних підходів до генерації та аугментації даних у випадках, коли дані є обмеженими.

1.4. Постановка проблеми на її обґрунтування.

У сучасному світі обсяги доступних даних зростають з кожним роком, але у багатьох випадках якість та кількість даних може бути недостатньою для ефективного навчання різноманітних моделей машинного навчання та штучного інтелекту. Тому для підвищення продуктивності та точності цих моделей з'являється потреба в розробці ефективних методів аугментації даних.

Аугментація даних - це процес створення нових даних на основі вже наявних, який допомагає покращити представлення різних аспектів та характеристик даних. Важливість аугментації даних висока у різних сферах, таких як комп'ютерне зору, розпізнавання мови та аналітика даних, і може відігравати вирішальну роль у побудові точних та ефективних моделей.

Дані проблеми унеможливлюють досягнення високої точності та продуктивності моделей машинного навчання та штучного інтелекту. Без адекватного розуміння та застосування ефективних методів аугментації даних для різних модальностей, є велика ймовірність зіткнення з такими проблемами, як:

1. Перенавчання (overfitting): моделі можуть виявитися занадто специфічними до навчального набору даних і неадекватно узагальнювати свої передбачення на нових даних.
2. Недостатня кількість даних: відсутність достатньої кількості та якості даних може призвести до поганих результатів навчання та непридатності моделей для вирішення практичних задач.
3. Часові та обчислювальні витрати: без використання оптимальних методів аугментації даних, навчання моделей може стати затратним як по часу, так і по обчислювальних ресурсах.

Усі ці проблеми можуть призвести до низької якості результатів, витрачених на розробку моделей, а також до неефективного використання обчислювальних ресурсів. Це підкреслює важливість проведення досліджень та розробки методів аугментації даних для різних модальностей, які допоможуть вирішити зазначені вище проблеми і досягти кращих результатів у побудові та

застосуванні моделей машинного навчання та штучного інтелекту.

Отже, методи аугментації даних допомагають збільшити обсяг та різноманітність навчальних даних, забезпечуючи краще узагальнення моделей і зменшуючи ризик перенавчання. Проте, аугментація даних зазвичай залежить від модальності даних, тому важливо дослідити та зрозуміти, які методи аугментації є ефективними для різних модальностей, таких як табличні дані, зображення та аудіо дані.

1.5. Формулювання мети і задач дослідження.

Мета дослідження: провести порівняльний аналіз методів аугментації даних для різних модальностей, зокрема табличних даних, зображень та аудіо даних, з метою виявлення найбільш ефективних підходів для покращення якості навчання моделей машинного навчання та штучного інтелекту.

Задачі дослідження:

1. Проаналізувати наукову літературу та визначити ключові методи аугментації даних для кожної з модальностей: табличні дані, зображення та аудіо дані.
2. Розробити критерії для оцінки ефективності методів аугментації даних, враховуючи специфіку різних модельних задач та модальностей.
3. Провести експерименти з різними методами аугментації даних, використовуючи зазначені критерії для оцінки їх ефективності на різних датасетах і задачах.
4. Проаналізувати результати експериментів і визначити найбільш ефективні методи аугментації даних для кожної модальності, враховуючи їх вплив на точність та узагальнюючу здатність моделей.
5. Сформулювати рекомендації щодо використання ефективних методів аугментації даних для різних модельних задач та модальностей, з метою покращення якості навчання моделей

машинного навчання та штучного інтелекту.

1.6. Висновки до першого розділу.

У даному розділі було здйснено дослідження наукової літератури з метою аналізу методів аугментації даних для різних модальностей.

Методологія PRISMA була використана для ефективного пошуку наукових джерел на платформах Scopus та Google Scholar. На основі огляду знайденої літератури, ми провели критичний аналіз 15 ключових літературних джерел, що допомогло отримати глибше розуміння проблеми та ідентифікувати основні виклики та тенденції в області аугментації даних.

Після детального розгляду відповідної літератури та аналізу різних аспектів аугментації даних, було окреслено основну проблему та її обґрунтування, визначено різні ефективні методи аугментації даних для кожної з модальностей: текстові дані, зображення та аудіо дані та їх вплив на точність моделей машинного навчання та штучного інтелекту.

З урахуванням отриманих результатів, сформульовано мету і задачі дослідження, спрямовані на аналіз методів аугментації, проведення експериментів, оцінювання ефективності цих методів.

У подальших розділах даної роботи буде здійснено детальний аналіз методів аугментації даних для кожної модальності, розроблено критерії оцінки ефективності цих методів, а також проведено експерименти з метою виявлення найбільш оптимальних підходів. Результати цих досліджень будуть використані для формулювання рекомендацій щодо використання ефективних методів аугментації даних у різних областях машинного навчання та штучного інтелекту, а також для визначення перспектив подальших наукових досліджень в галузі аугментації даних.

2. ДОСЛІДНИЦЬКИЙ РОЗДІЛ

2.1. Задача класифікації.

Класифікація — це один із різновидів навчання під наглядом в галузі машинного навчання, метою якого є розпізнати (класифікувати) екземпляри або зразки в один із кількох попередньо визначених класів або категорій на основі їхніх особливостей або характеристик.

Задача класифікації [17] передбачає побудову набору функцій або правил, які можуть відображати екземпляри вхідних даних, представлені як вектори ознак, на відповідні мітки класу. Ці функції призначені для охоплення основної структури та зв'язків у даних, щоб розрізняти та розділяти класи.

Математично це можна представити наступним чином:

- Вхідні дані: Вхідними даними для проблеми класифікації зазвичай є вектор $X = (x1,\ x2,\ \ldots, x_n)$, що містить числові, категоріальні значення, що представляють ознаки або атрибути об'єкта чи спостереження. Цей вектор належить простору ознак, позначеному як X.
- Вихідні дані (класи): вихідними результатами проблеми класифікації є дискретна змінна Y, яка має набір попередньо визначених категорій або класів $C = \{C1,\ C2,\ \ldots, C_k\}$. Мета полягає в тому, щоб знайти функцію (алгоритм або модель), яка призначає деякий клас C одному із вхідних векторів X.
- Набір даних: надається набір даних D, який зазвичай містить низку спостережень або зразків (випадків). Кожен екземпляр представлено вектором ознак X_i та його відповідною міткою класу Y_i, позначеною як $(X_i,\ Y_i)$ для i = 1, 2, ..., N, де N — загальна кількість екземплярів у наборі даних.

Проблема класифікації має на меті оцінити функцію f: X → Y так, щоб для будь-якого вхідного екземпляра x, f(x) передбачила правильну мітку класу y для цього екземпляра. В ідеалі функція f повинна бути в змозі мінімізувати помилку класифікації екземплярів даних, зберігаючи при цьому прийнятний рівень

складності.

У машинному навчанні процес класифікації виглядає наступним чином:

- **Збір даних і попередня обробка**. Потрібно отримати набір даних, що містить вектори ознак та їхні відповідні мітки класів. Попередня обробка може передбачати очищення, нормалізацію або розробку функцій для забезпечення якості та придатності даних для моделювання.
- **Вибір ознак**. Потрібно визначити найбільш релевантні особливості, які значною мірою сприяють класовому поділу та дискримінації. Цей крок може допомогти покращити продуктивність моделі та зменшити обчислювальну складність шляхом усунення нерелевантних або зайвих функцій.
- **Вибір моделі**. Тоді потрібно обрати відповідний класифікатор на основі характеристик даних і вимог проблеми. Приклади алгоритмів класифікації включають логістичну регресію, опорні векторні машини (SVM), дерева рішень, випадкові ліси, k-найближчих сусідів (KNN) і нейронні мережі.
- **Навчання моделі**. Позначений набір даних розділяється на набір для навчання та перевірки (або тестування). Використовуйте навчальний набір, щоб «навчити» класифікатор взаємозв’язкам між вхідними ознаками та мітками класу, регулюючи його внутрішні параметри. Залежно від вибраного алгоритму, цей крок може включати оптимізацію функції витрат, побудову межі рішення або вивчення ваг ознак.
- **Оцінка моделі**. Після навчання потрібно перевірити навчену модель за допомогою тестувального набору, який містить невидимі екземпляри для оцінки продуктивності моделі та її здатності до узагальнення. Такі показники оцінки, як accuracy, precision, recall, f1-score можна використовувати для кількісної оцінки успіху класифікатора у передбаченні правильних позначок класу.

Крім того, при класифікації екземплярів ми можемо зіткнутись з проблемою недостатньої кількості даних. Відсутність даних у процесі класифікації може серйозно вплинути на продуктивність та ефективність моделей класифікації. Обмежений набір даних не лише створює труднощі при розробці моделі, але також може призвести до певних проблем під час процесу класифікації.

Перелік проблем до яких може привести обмежена вибірка даних:

- **Перенавчання.** З обмеженою кількістю навчальних даних моделі можуть перенавчитись [18], що означає, що вони починають запам’ятовувати навчальні зразки замість того, щоб вчитися узагальнювати шаблони даних. У результаті моделі добре працюватимуть на навчальних даних, але не зможуть забезпечити точні прогнози на невидимих даних.
- **Недостатнє представлення.** Менший набір даних може не охоплювати весь простір функцій і належним чином розподілити класи, що призведе до поганої моделі. Такі моделі не можуть відобразити справжні характеристики проблеми та можуть бути неефективними або давати упереджені прогнози.
- **Дисбаланс класів.** У сценаріях, коли набір даних невеликий і розподіл класів спотворений, процес навчання моделі буде зміщений у бік більшості класів. Це призводить до поганої ефективності прогнозування для класу меншості, оскільки модель недостатньо піддається їхнім відмінним рисам і шаблонам.
- **Варіативність і невизначеність.** Невеликий набір даних може не містити достатньо різноманітних зразків, що призводить до високої мінливості та невизначеності в прогнозах моделі. У результаті продуктивність будь-якого конкретного тестового набору даних може бути далекою від очікуваної загальної продуктивності, що призведе до ненадійних результатів.

Один із способів вирішення проблеми недостатньої кількості даних є їх аугментація.

2.2. Аугментація даних різних модальностей.

Аугментація (збільшення) даних — це техніка, яка використовується в машинному навчанні, особливо в контексті глибокого навчання, для розширення та збагачення навчального набору даних шляхом створення нових екземплярів даних із застосуванням різноманітних перетворень до існуючих даних. Ці перетворення призначені для збереження вихідних міток класів, одночасно вносячи варіації та різноманітність даних, які можуть імітувати сценарії реального світу.

У деталях збільшення даних виконує кілька завдань, щоб допомогти вирішити проблему нестачі даних:

- **Збільшення розміру набору даних.** Застосовуючи різні перетворення до оригінальних екземплярів у навчальному наборі даних, розширення даних створює більший набір різноманітних зразків. Ці додаткові дані допомагають зменшити ризик перенавчання та дають змогу моделям вивчати та отримувати більше відмінних ознак для покращеного узагальнення невидимих даних.
- **Підвищення різноманітності даних.** Різноманітні варіації, введені завдяки збільшенню даних, ефективно розширюють простір функцій і дають більш повне представлення розподілу даних. Завдяки навчанню на більш різноманітному наборі даних моделі стають надійними та адаптованими до реальних ситуацій, що забезпечує вищу точність прогнозів.
- **Вирішення проблеми дисбалансу класів.** Доповнення даних може допомогти усунути дисбаланс класів шляхом вибіркового створення більшої кількості екземплярів для недостатньо представлених класів. Це врівноважує розподіл класів і сприяє кращому вивченню особливостей, пов’язаних з класом меншості, що в кінцевому

підсумку призводить до покращення продуктивності передбачення в усіх класах.

- **Незмінність до перетворень.** Розширення даних за допомогою таких операцій, як обертання, масштабування та перевертання, дозволяє моделі навчитися бути стійкою до цих перетворень і зберігати точність прогнозування, незважаючи на наявність таких змін у екземплярах даних реального світу.
- **Неявна регулярізація.** Збільшення даних також служить формою неявної регулярізації, оскільки воно змушує моделі вивчати надійні та незмінні функції, а не запам'ятовувати навчальні дані. Це може допомогти зменшити ненавчання та покращити узагальнення моделі на нових екземплярах даних.

Залежно від модальності даних, різні техніки штучного збільшення даних можуть бути використані. Розглянемо різні модальності даних.

2.2.1 Методи аугментації текстових даних.

Аугментація текстових даних [19] – це процес, який створює нові текстові зразки з існуючого набору даних шляхом застосування різних методів. Розширення покращує продуктивність обробки природної мови (NLP) і пов'язаних з текстом моделей машинного навчання шляхом збільшення розміру та різноманітності набору даних. Розглянемо приклади збільшення текстових даних:

- **Заміна синонімів.** Це техніка доповнення текстових даних, яка передбачає заміну слів у реченні їхніми синонімами, генеруючи нові речення зі схожими значеннями, але дещо відмінним словниковим запасом. Щоб виконати заміну синонімів у певному реченні, спочатку речення токенізується. Тоді потрібно обрати цільові слова для заміни, особливо зосереджуючись на словах які відіграють важливу роль у змісті речення. Наступним кроком потрібно знайти контекстуально релевантні синоніми. Отримавши синоніми можемо

замінити вихідні слова визначеними синонімами. В результаті отримуємо нове речення. Цей процес можна повторювати допоки не досягнемо бажану різноманітність речень.

- **Випадкова зміна позиції.** Це техніка змінює оригінальні речення, не змінюючи їх загального значення. У цій техніці випадкові пари слів міняються місцями в реченні, що призводить до дещо змінених речень, зберігаючи загальний контекст і наміри.
- **Випадкове вставлення**. Це техніка доповнення текстових даних, за якої слова чи фрази довільно вставляють у речення, створюючи нові варіації речень, які зберігають загальний контекст і значення вихідного тексту. Реалізація випадкової вставки спочатку також вимагає токенізації вхідного речення. Також потрібно визначати ліміт вставок, щоб не вставити забагато слів. За допомогою мовних моделей визначати відповідні слова для вставлення. Тоді довільно обираються позиції в реченні, куди будуть вставлені нові слова чи фрази, переконавшись, що вони не змінюють граматичну структуру речення чи загальну зв’язність.

2.2.2 Методи аугментації зображень.

Доповнення даних зображення [14] відіграє вирішальну роль у покращенні продуктивності моделей машинного навчання, зокрема в задачах комп’ютерного зору, таких як класифікація зображень, виявлення об’єктів і сегментація. Застосовуючи різні перетворення до оригінальних зображень, створюються нові навчальні зразки, що робить набір даних більшим і різноманітнішим. Це допомагає моделі вивчати більш узагальнені функції та ставати стійкішою до варіацій даних реального світу. Розглянемо приклади аугментації зображень:

- **Випадкове обертання.** Ця техніка передбачає випадкове обертання зображень на заданий кут у визначеному діапазоні. Для може бути корисно для навчання моделей для розпізнавання об’єктів з різною орієнтацією.

- **Зсув по ширині та по висоті.** Ці методи передбачають випадковий горизонтальний або вертикальний зсув зображень на певну частку загальної ширини або висоти зображення. Ці техніки допомагають моделі розпізнавати об’єкти, незважаючи на невеликі зміщення.
- **Зміна яскравості.** Цей метод змінює яскравість зображень шляхом випадкового налаштування рівнів освітлення в межах заданого діапазону. Це дозволяє моделі вивчати функції, які менше залежать від умов освітлення.
- **Зсув каналу.** Ця техніка передбачає зміну каналів кольорів зображень шляхом випадкового зсуву їх інтенсивності в межах заданого діапазону. Це допомагає моделі стати більш інваріантною до колірних варіацій.
- **Трансформація зсуву.** Ця операція спотворює зображення шляхом застосування випадкового кута зсуву в заданому діапазоні. Зсув зображень допомагає моделям розпізнавати об’єкти, які можуть виглядати деформованими або розтягнутими.
- **Масштаб**. Це техніка довільного збільшення або зменшення масштабу в межах заданого діапазону, яка може допомогти моделі розпізнавати об’єкти в різних масштабах і на різних відстанях.
- **Горизонтальне та вертикальне перевертання.** Це техніка довільного перевертання зображень по горизонталі або вертикалі створює нові зображення з різними ракурсами та допомагає моделі узагальнювати різні точки зору.

Поєднання різних методів збільшення зображення може бути корисним і дуже ефективним у покращенні продуктивності моделей машинного навчання, зокрема в задачах комп’ютерного зору. Поєднання кількох методів доповнення допомагає створити багатший і різноманітніший набір даних, дозволяючи моделі вивчати більш надійні, узагальнені та інваріантні функції. Хоча поєднання різних методів розширення може запропонувати численні переваги, важливо враховувати конкретне завдання та природу набору даних. Деякі комбінації

можуть не підвищити продуктивність для певних завдань або навіть зменшити здатність моделей вивчати важливі функції. Ретельний вибір, застосування та оцінка методів збільшення забезпечує оптимальне поєднання для даної проблеми.

2.2.3 Методи аугментації аудіозаписів.

Аугментація аудіозаписів [20] – це процес створення нових зразків аудіо з наявного набору даних шляхом застосування різноманітних перетворень до вихідних записів. Аудіодоповнення допомагає збільшити розмір і різноманітність набору даних, покращуючи продуктивність і можливості узагальнення моделей машинного навчання в завданнях, пов'язаних із аудіо, такими як автоматичне розпізнавання мовлення (ASR), ідентифікація мовця та класифікація звуку. Розглянемо приклади аугментації аудіо:

- **Розтяг аудіо по часовій прямі.** Це метод аугментації аудіо, який передбачає зміну тривалості аудіосигналу без впливу на його висоту або основну частоту. Цей метод генерує нові зразки звуку, які зберігають той самий вміст, що й вихідний сигнал, але зі зміненою тривалістю. Один із способів реалізації цього методу аугментації є наступний алгоритм:
  1. *Короткочасне перетворення Фур'є (STFT)*. Фазовий вокодер спочатку обчислює STFT вхідного аудіосигналу, перетворюючи сигнал часової області в частотне подання. STFT досягається застосуванням віконної функції до коротких сегментів аудіосигналу, що перекриваються, а потім перетворенням Фур'є кожного сегмента.
  2. *Масштабування STFT:* Далі фазовий вокодер масштабує часову вісь STFT за наданим коефіцієнтом розтягування. Це фактично змінює тривалість аудіо, зберігаючи вміст частоти.
  3. *Зворотне короткочасне перетворення Фур'є (iSTFT).* Нарешті, фазовий вокодер обчислює зворотне STFT масштабованого

частотного представлення, щоб отримати розтягнутий у часі аудіосигнал. Зворотний STFT передбачає рекомбінацію модифікованого частотного представлення назад у часову область.

- **Зміщення висоти**. Ця техніка спотворює висоту аудіосигналу, транспонуючи його вгору або вниз у певному діапазоні півтонів. Це допомагає моделі стати більш стійкою до змін висоти гучномовця та частотного вмісту. Досягнути цього можна наступним чином:
  1. *Розтягування часу за допомогою фазового вокодера*. Спочатку вхідний аудіосигнал піддається розтягуванню часу на основі бажаного зсуву висоти. Це розтягує або стискає тривалість аудіо без впливу на висоту за допомогою методу фазового вокодера, який передбачає обчислення короткочасного перетворення Фур'є (STFT), масштабування часової області, а потім обчислення зворотного STFT для синтезу розтягнутого в часі аудіосигналу.
  2. *Повторна дискретизація*. Після кроку розтягування часу створюється звук зі зміщенням висоти шляхом регулювання частоти дискретизації звуку. Якщо цільове зміщення висоти звуку є збільшенням, звук повторно дискретизується до нижчої частоти дискретизації, фактично підвищуючи висоту тону. І навпаки, якщо метою є зменшення висоти звуку, аудіо дискретизується до вищої частоти дискретизації, що фактично знижує висоту тону.
- **Випадковий зсув**. Це техніка аудіодоповнення, яка передбачає зміну положення аудіоподій у вихідному сигналі шляхом зсуву часу початку або завершення сигналу. Це створює нові зразки аудіо з різним вирівнюванням часу для подій і вносить варіативність у набір даних. Один з варіантів зсуву – це зсув у часі. Зсув у часі передбачає переміщення аудіоподій у межах оригінального запису шляхом зміщення часу початку, зсуву часу завершення або обох одночасно. Випадкове зміщення, яке може бути додатнім або від'ємним, застосовується до звукових подій, завдяки чому вони починаються

раніше чи пізніше в межах сигналу.

2.3. Алгоритми класифікації даних різних модальностей.

Вибір відповідного алгоритму машинного навчання для завдань класифікації залежить від різних факторів, включаючи модальність даних, розмір набору даних, представлення функцій, обчислювальні ресурси та бажані показники продуктивності. Різні модальності даних, такі як зображення, текст, аудіо та числові дані, демонструють унікальні характеристики та вимагають спеціальних методів виділення ознак і навчання.

2.3.1 Класифікатор текстових даних.

Один з методів машинного навчання для класифікації текстових даних є Логістична Регресія [21]. Це широко поширений контрольований алгоритм машинного навчання для завдань бінарної та багатокласової класифікації. Це узагальнена лінійна модель (GLM), яка моделює ймовірність певного результату за допомогою логістичної функції. Логістична регресія особливо корисна при роботі з категоріальними залежними змінними та безперервними або дискретними незалежними змінними.

Логістична регресія може бути ефективним методом для завдань класифікації тексту, особливо у випадках з обмеженими обчислювальними ресурсами або коли потрібна проста, інтерпретативна модель. Ряд переваг логістичної регресії для класифікації тексту:

- **Багатовимірні та розріджені дані.** Текстові дані після виділення ознак зазвичай призводять до багатовимірних і розріджених представлень, де кожен вимір відповідає унікальному слову або лексемі зі словника. Логістична регресія добре працює в таких ситуаціях, оскільки вона може обробляти велику кількість функцій, зберігаючи при цьому відносно просту лінійну модель, яку можна інтерпретувати.
- **Масштабованість.** логістична регресія ефективна з точки зору

обчислень і може добре масштабуватися відповідно до розміру набору даних і кількості функцій. Це робить його придатним вибором для завдань класифікації тексту, особливо при роботі з обмеженими обчислювальними ресурсами або коли потрібна швидка модель класифікації.

Існує кілька стратегій розширення логістичної регресії для проблем багатокласової класифікації, розглянемо їх:

- **Стратегія «One vs Rest» (OVR) або «One vs All» (OVA).** У цьому підході ми навчаємо декілька класифікаторів бінарної логістичної регресії, по одному для кожного класу. Для кожного класифікатора ми вважаємо один клас позитивним (цільовий клас), а решту – негативним (нецільовий клас). Щоб зробити прогноз, ми пропускаємо екземпляр тесту через кожен класифікатор і вибираємо клас, який відповідає класифікатору з найвищою оцінкою ймовірності.
- **Стратегія One vs One (OVO).** У стратегії «один проти одного» ми навчаємо класифікатор бінарної логістичної регресії для кожної пари класів, у результаті чого отримуємо $N \cdot \frac{N-1}{2}$ класифікатори для N класів. Класифікатори навчаються лише на екземплярах з пари класів, за які вони відповідають. Щоб зробити прогноз для екземпляра, ми розглядаємо результати всіх класифікаторів і вибираємо клас, який отримує найбільше «голосів» окремих класифікаторів.
- **Регресія Softmax.** Softmax є розширенням логістичної регресії для проблем багатокласової класифікації. Замість тренування кількох бінарних класифікаторів ми безпосередньо моделюємо ймовірність кожного класу. Функція softmax, також відома як нормалізована експоненціальна функція, застосовується до лінійних вихідних результатів, щоб отримати розподіл ймовірностей за цільовими класами. Як і двійкова логістична регресія, ми використовуємо

принцип оцінки максимальної правдоподібності (MLE) для оцінки параметрів моделі.

2.3.1 Класифікатор зображень.

Згорткова нейронна мережа (CNN) [22] — це тип моделі глибокого навчання, що спеціалізується на обробці сіткових даних, наприклад зображень, за допомогою згорткових шарів для захоплення локальних просторових моделей і абстракцій. CNN стали основною моделлю для численних завдань класифікації зображень завдяки їхній здатності вивчати ієрархічні функції, стійкості до шуму та варіацій у вхідних даних.

Основними шарами CNN для класифікації зображень є:

- **Input Layer**. Вхідний рівень відповідає за отримання вихідних даних зображення. Кожне вхідне зображення зазвичай представляється як тривимірний тензор (ширина, висота та кількість колірних каналів - зазвичай 3 для зображень RGB).
- **Convolutional Layers.** Ці шари виконують операцію згортки, яка передбачає застосування набору доступних для навчання фільтрів до вхідного зображення або виводу попередніх шарів. Під час згортки фільтр проходить по зображенню, поелементно множачи фільтр на локальне оточення з подальшим підсумовуванням результатів для створення вихідної карти ознак. Кожен фільтр виявляє певні особливості або візерунки у вхідних даних, такі як грані, лінії, текстури або більш складні структури на більш глибоких рівнях мережі.
- **Нелінійність (функції активації).** Функції активації вводять нелінійність у мережу, дозволяючи їй вивчати та апроксимувати складні, нелінійні функції. Найпоширенішою функцією активації є Rectified Linear Unit (ReLU), яка зберігає додатні значення та встановлює від’ємні значення на нуль.
- **Pooling Layers.** Шари об’єднування зменшують просторові розміри

та обчислювальну складність, роблячи модель більш ефективною з точки зору обчислень та інваріантною до змін. Найбільш часто використовуваною технікою об’єднання є Max-Pooling, яке бере максимальне значення з певного локального околу (зазвичай 2x2) і рухається по зображенню.

- **Fully Connected payers.** Повнозв’язані шари, використовуються для відображення результатів згорткового шару та шару об’єднання в остаточну оцінку класу або розподіл ймовірностей. У типовій архітектурі може існувати один або кілька повнозв’язаних рівнів, за якими слідує функція активації softmax для виведення ймовірностей для кожного класу.
- **Output Layer.** Вихідний рівень виробляє кінцеві ймовірності класу для класифікації зображень. Для багатокласових задач часто використовується активація softmax, тоді як для бінарної класифікації можна використовувати сигмоїдну активаційну функцію. Клас із найвищою ймовірністю вибирається як остаточний прогноз мережі.

Мережа CNN також додатково може бути покращенна з використанням наступним шарів:

- **Batch Normalization.** Це техніка, яка часто використовується в CNN для швидшої конвергенції та кращого узагальнення. Він передбачає нормалізацію результату шару за допомогою вивчених параметрів масштабування та зсуву, щоб результат мав стабільне, попередньо визначене середнє значення та стандартне відхилення.
- **Dropout.** Щоб зменшити перенавчання та покращити узагальнення, до архітектур CNN іноді додають рівні вилучення. Шари вилучення випадковим чином «скидають» певний відсоток нейронів під час навчання, змушуючи мережу вивчати більш надійні функції.
- **Регуляризація**. Це також важливою технікою в машинному навчанні, яка спрямована на запобігання перенавчанню шляхом

додавання штрафу до функції втрат. Перенавчання, як правило, відбувається, коли модель навчилась вловлювати шум у навчальних даних, що призводить до поганого узагальнення невидимих даних. У згорткових нейронних мережах (CNN) методи регуляризації допомагають покращити узагальнення та підвищити стабільність процесу навчання.

Усі ці техніки використовуються для побудови моделі, яка буде добре вміти працювати з різними вхідними даними. З використанням цих правил в парі з аугментацією даних - є хорошою практикою, яка веде до кращого узагальнення, покращеної продуктивності моделі та підвищеної стійкості до варіацій вхідних даних.

2.3.3 Класифікатор аудіозаписів.

Класифікація аудіо — це процес класифікації та позначення різних типів аудіосигналів у окремі класи чи категорії на основі їхнього вмісту чи характеристик. Основна мета аудіокласифікації полягає в тому, щоб автоматично аналізувати, розуміти та впорядковувати великі обсяги аудіоданих. Існує кілька компонентів і методів, які відіграють вирішальну роль у розробці ефективної та надійної системи класифікації звуку. Один з них це етап виділення ознак. Етап виділення ознак зазвичай є перший етам у задачі класифікації аудіозаписів. Спочатку відбувається вилучення релевантних ознак із необробленого аудіосигналу. Вибір ознак суттєво впливає на ефективність класифікації, і залежно від конкретної програми можуть використовуватися різні типи ознак. Зазвичай використовувані функції включають спектрограми, кепстральні коефіцієнти Mel-Frequency (MFCC), функції кольоровості та спектральний контраст.

Кепстральні коефіцієнти Mel-Frequency (MFCC) [23] — це широко використовуваний набір функцій у обробці аудіосигналів, зокрема для аналізу мови та музики. Вони вперше були представлені в 1980-х роках як засіб представлення спектральних характеристик аудіосигналів у компактний та ефективний спосіб. Основна ідея MFCC полягає в тісному наслідуванні системи

нелінійного слухового сприйняття людини, яка має тенденцію бути більш чутливою до одних діапазонів частот, ніж до інших.

Розрахунок MFCC включає кілька ключових кроків:

- **Попередня обробка.** Необроблений аудіосигнал спочатку попередньо обробляється шляхом застосування фільтра високих частот для посилення високочастотних компонентів і зменшення ефекту шуму. Це робиться для того, щоб підкреслити частоти, які є більш доречними для розрізнення звуків мови.
- **Кадрування та вікна**. Аудіосигнал потім розділяється на невеликі кадри, що перекриваються, зазвичай близько 20-40 мілісекунд кожен. Це робиться для захоплення локальних характеристик аудіосигналу, припускаючи, що спектральні властивості сигналу є відносно стабільними протягом такої короткої тривалості. Віконна функція, як правило, вікно Хеммінга або Ханнінга, застосовується до кожного кадру, щоб мінімізувати ефекти бічних пелюсток у частотній області.
- **Перетворення Фур'є та спектр потужності.** Дискретне перетворення Фур'є (ДПФ) застосовується до кожного віконного кадру для перетворення його в частотну область. Це призводить до комплексної величини та фазового спектру кожного кадру. Потім розраховується квадрат величини (або ступінь) спектру величини, формуючи основу для подальшого аналізу.
- **Фільтри Мела.** Щоб отримати частотну шкалу, яка більше відповідає людському слуховому сприйняттю, до спектру потужності застосовується банк фільтрів Мела. Шкала Мел — це логарифмічна шкала частоти, яка намагається приблизно оцінити людське сприйняття різниць частот. Група фільтрів зазвичай складається з трикутних фільтрів, що перекриваються, рівномірно розподілених за шкалою Мела та охоплюючи весь частотний діапазон спектру потужності. Вихід кожного фільтра представляє

енергію, присутню у відповідній смузі частот у шкалі Мела.

- **Логарифм і дискретне косинусне перетворення (DCT).** Логарифмічна функція застосовується до виходу банку фільтрів Мела для отримання значень логарифмічної енергії. Цей крок наближає людське сприйняття інтенсивності звуку. Після цього дискретне косинусне перетворення (DCT) застосовується до значень логарифмічної енергії, щоб отримати кепстр. DCT допомагає декорелювати спектральну інформацію, створюючи компактне представлення. Кілька перших коефіцієнтів кепстра (зазвичай між 12-20) зберігаються як остаточні характеристики MFCC, оскільки ці коефіцієнти фіксують найбільш релевантну інформацію про спектральну оболонку.

Виділення ознак є важливим кроком у класифікації аудіо, оскільки воно допомагає перетворити необроблені дані на значуще та компактне представлення, придатне для завдань аналізу та класифікації.

Для класифікації аудіозаписів будемо використовувати багатошаровий персептрон (MLP) [24]. Це тип штучної нейронної мережі з прямою архітектурою, що складається з кількох шарів взаємопов’язаних нейронів. Це контрольований алгоритм навчання, який використовує навчальні дані з мітками для вивчення зв’язку між вхідними функціями та відповідними вихідними мітками. MLP широко використовується для різних завдань класифікації та регресії. Нейрони в MLP пов’язані між собою, і кожне з’єднання має вагу, яка визначає силу зв’язку між нейронами. Під час процесу навчання ці ваги коригуються за допомогою алгоритму навчання, наприклад зворотного поширення та градієнтного спуску, щоб мінімізувати помилку при прогнозуванні вихідних міток.

Базова структура персептрона складається з наступний шарів:

- **Input Layer.** Вхідний рівень відповідає за отримання вхідних функцій і передачу їх на наступний рівень. Кількість нейронів у цьому шарі відповідає кількості ознак у вхідних даних.
- **Hidden Layes.** Приховані шари розташовані між вхідним і вихідним

шарами. MLP може мати один або кілька прихованих рівнів. Нейрони в прихованих шарах обробляють і перетворюють інформацію, отриману з попереднього шару, і передають результат на наступний шар. Складність вивченої функції зростає зі збільшенням кількості прихованих шарів і кількості нейронів на шар.

- **Output Layer.** Рівень виводу виробляє кінцевий вихід для заданого введення. Кількість нейронів у вихідному шарі залежить від типу та складності проблеми. Для бінарної класифікації вихідний рівень зазвичай має один нейрон із сигмоподібною функцією активації, тоді як для багатокласової класифікації вихідний рівень використовує функцію активації softmax із нейроном для кожного класу.

2.4. Метрики оцінювання якості прогнозів.

Оцінка якості прогнозів, зроблених моделлю машинного навчання, є вирішальним кроком у процесі розробки моделі. Різні показники допомагають оцінити ефективність і продуктивність моделі для створення точних і надійних прогнозів. Є кілька показників, які можна використовувати для вимірювання ефективності моделі машинного навчання. Розглянемо деякі метрики, які будемо використовувати для оцінки ефективності моделей [25].

- **Accuracy.** Метрика є широко використовуваним показником для оцінки продуктивності моделі класифікації. Вона вимірює частку правильних прогнозів, зроблених моделлю, порівняно із загальною кількістю зразків. Іншими словами, він забезпечує оцінку того, наскільки добре модель може правильно класифікувати екземпляри з заданого набору даних. Хоча метрика accuracy є простим і легким для розуміння показником, він має певні обмеження, особливо при роботі з незбалансованими наборами даних. У незбалансованому наборі даних пропорція вибірок із різних класів нерівномірна, що призводить до неоднакового акценту на кожному класі. Як результат, модель може мати високу точність, просто передбачаючи більшість

класів, навіть якщо вона погано прогнозує інші класи. У таких випадках для оцінки продуктивності класифікатора більше підходять інші показники, такі як precision, recall та F1-score.

$$accuracy = \frac{TN+TP}{TP+FP+TN+FN} \quad (2.1.)$$

- **precision** оцінює, наскільки точна модель у прогнозуванні позитивних значень. Precision відповідає на запитання: скільки разів модель передбачала позитивний результат, як часто він був правильним. Вона особливо корисна коли набір даних незбалансований.

$$precision = \frac{TP}{TP+FP} \quad (2.2.)$$

- **recall** вимірює скільки фактичних позитивних випадків модель правильно передбачила як позитивні. Recall часто використовується в поєднанні з precision, яка вимірює частку справжніх позитивних випадків серед усіх випадків, прогнозованих як позитивні моделлю. Поєднання precision та recall допомагає вирішити проблему дисбалансу класів або коли вартість помилкових негативів переважає над помилково позитивними результатами. Одним з обмежень використання recall як окремого показника є те, що він не враховує хибні спрацьовування (випадки, неправильно передбачені як позитивні). Щоб врахувати баланс між precision та recall потрібно використовувати показник F1-score.

$$recall = \frac{TP}{TP+FN} \quad (2.3.)$$

- **F1-score.** Це метрика оцінювання, яка є особливо ефективною, коли мова йде про незбалансовані набори даних або коли виникають як помилкові позитивні, так і помилкові негативні результати. Це середнє гармонійне значення precision та **recall** , яке поєднує ці два показники, щоб забезпечити збалансоване вимірювання ефективності моделі.

$$f1 = \frac{2 \cdot precision \cdot recall}{precision+recall} \quad (2.4.)$$

- **Confusion matrix.** Це таблиця, яка відображає ефективність моделі класифікації шляхом порівняння її прогнозованих міток із фактичними мітками (основна правда) у форматі перехресної таблиці. Він надає загальне уявлення про продуктивність моделі, дозволяючи ідентифікувати конкретні області, у яких модель має недостатню продуктивність або неправильно класифікує екземпляри. Кожен рядок і стовпець у матриці представляють прогнозований і фактичний класи відповідно. Значення клітинок підраховують кількість випадків відповідних передбачень.

2.5. Висновки до другого розділу.

У цьому розділі ми обговорили різні аспекти завдань класифікації, застосованих до різних модальностей даних, включаючи текст, зображення та аудіо.

Було описано проблеми класифікації, яка передбачає призначення однієї з кількох попередньо визначених міток класу вхідній вибірці. Однією з ключових проблем у проблемах класифікації є робота з незбалансованими даними, коли певні класи мають значно менше вибірок, ніж інші. Цей дисбаланс може призвести до упередженої роботи класифікатора, надаючи перевагу класу більшості та нехтуючи класом меншості. Щоб усунути цю проблему, можна застосувати методи доповнення даних, щоб штучно збільшити кількість вибірок у недостатньо представлених класах. Ці методи, які відрізняються залежно від модальності даних, включають створення нових зразків шляхом перетворення або рекомбінації наявних даних, що в кінцевому підсумку призводить до більш збалансованого набору даних.

Для вирішення проблем класифікації можна використовувати різні класифікатори, починаючи від традиційних алгоритмів машинного навчання, таких як логістична регресія, до більш складних підходів глибокого навчання, таких як згорткові нейронні мережі (CNN. Вибір класифікатора залежить від таких факторів, як складність проблеми, розмір набору даних і обчислювальні

ресурси.

Оцінка якості моделей класифікації є життєво важливою для визначення їх ефективності. Було розглянуто декілька показників ефективності, які є варіативними до різних модельних задач та модальностей, включаючи accuracy, precision, recall, f1-score та confusion matrix. Ці показники надають різні погляди на продуктивність моделі, причому деякі з них більше підходять для певних сценаріїв, таких як незбалансовані набори даних або випадки, коли, наприклад, помилкові позитивні та помилкові негативні результати мають нерівну вартість.

3. РОЗДІЛ АПРОБАЦІЙ ТА РЕЗУЛЬТАТІВ

3.1. Засоби реалізації.

Програмна реалізація була виконана з використання хмарного середовища Google Colaboratory та було використано мову програмування Python. Для реалізація задачі було використано ряд допоміжних бібліотек:

- **Textaugment**. Це бібліотека обробки природної мови (NLP), який надає різні методи доповнення текстових даних. За допомогою заміни синонімів, випадкового видалення, випадкового вставлення та інших процесів обробки тексту бібліотека TextAugment пропонує дієвий і ефективний підхід до покращення наборів даних NLP, таким чином потенційно покращуючи продуктивність і можливість узагальнення алгоритмів машинного навчання.
- **Nltk.** Це потужна бібліотека Python для роботи з даними людської мови. Він розроблений, щоб полегшити завдання обробки природної мови (NLP), такі як токенізація, визначення основи, розбір, класифікація та аналіз настроїв, серед інших функцій. Розроблений як проект з відкритим кодом, NLTK пропонує повний набір інструментів і ресурсів, включаючи велику кількість готових модулів, корпусів і лексичних ресурсів.
- **Numpy.** Це бібліотека з відкритим кодом для мови програмування Python, яка забезпечує потужну та гнучку підтримку математичних операцій над багатовимірними масивами та матрицями.
- **Pandas.** Це потужна бібліотека, яка надає гнучкі, високопродуктивні інструменти обробки та аналізу даних. Його розроблено, щоб зробити роботу зі структурованими даними, такими як табличні, часові ряди та матричні дані, одночасно легкою та ефективною. Pandas побудовано на основі бібліотеки NumPy. Ключовою особливістю Pandas є DataFrame, двовимірна таблична структура даних, схожа на електронну таблицю або таблицю SQL, з позначеними осями для рядків і стовпців. Це дає змогу легко

виконувати широкий спектр операцій, таких як очищення даних, перетворення, агрегації та візуалізації.

- **OpenCV.** Це бібліотека комп'ютерного бачення, обробки зображень і машинного навчання. Бібліотека написаний на C/C++, тим самим забезпечує швидке виконання коду. Крім того OpenCV може похвалитися широким набором функцій і алгоритмів для обробки, аналізу та обробки зображень і відео в режимі реального часу.
- **Sklearn.** Ця бібліотека надає повний набір інструментів машинного навчання для Python. Він побудований на основі NumPy, SciPy і Matplotlib. Scikit-learn широко використовується для таких завдань, як класифікація, регресія, кластеризація, зменшення розмірності та оцінка моделі, тощо.
- **Keras.** Бібліотека для роботи з нейронними мережами. Вона розроблена, щоб забезпечити швидкий, зручний і модульний підхід до створення моделей глибокого навчання. Keras діє як інтерфейс для бібліотеки TensorFlow, яка є її серверною частиною за замовчуванням.
- **Librosa.** Бібліотека, яка призначена для аналізу та обробки аудіо та музичних даних. Він надає широкий спектр інструментів і функцій для різноманітних завдань аналізу музики та аудіо, таких як виділення ознак, обробка аудіосигналу та пошук музичної інформації (MIR). Librosa створена на основі інших потужних бібліотек, таких як NumPy, SciPy і scikit-learn.
- **Matplotlib.** Це бібліотека графічних зображень, яка надає широкий спектр інструментів візуалізації та функцій для створення високоякісних статичних, анімованих та інтерактивних 2D та 3D графіків і фігур.

3.2. Огляд наборів даних та їх аугментація.

Розглянемо три модальності даних та будемо проводити експерименти з

використанням аугментації:

- Текст.
- Зображення.
- Аудіо.

3.2.1. Аугментація текстових даних.

Для досліджень впливу аугментації на текстові дані було обрано датасет про відгуки покупців товарів на Amazon та настрій цих відгуків.

Поглянемо на датасет. (Рис. 3.1.)

	sentiments	cleaned_review	cleaned_review_length	review_score
0	positive	i wish would have gotten one earlier love it a...	19	5.0
1	neutral	i ve learned this lesson again open the packag...	88	1.0
2	neutral	it is so slow and lags find better option	9	2.0
3	neutral	roller ball stopped working within months of m...	12	1.0
4	neutral	i like the color and size but it few days out ...	21	1.0
...	...	...	...	...
17335	positive	i love this speaker and love can take it anywh...	30	5.0
17336	positive	i use it in my house easy to connect and loud ...	13	4.0
17337	positive	the bass is good and the battery is amazing mu...	41	5.0
17338	positive	love it	2	5.0
17339	neutral	mono speaker	2	5.0

Рис. 3.1. Датасет відгук на товари

Вибірка містить 17340 рядків та складається з наступних колонок:

- **Sentiments.** Цільова колонка датасету – настрій відгуку покупця. Може мати одне з трьох значень: позитивний, негавний чи нейтральний настрій.
- **Cleared_review.** Головна ознака датасету - текст відгуку.
- **Cleared_review_length.** Довжина відгуку.
- **Review_score.** Оцінка товару.

В даному випадку класифікатор буде навчатись для передбачення настрою відгуків. Тому для нас цікаві лише колонки Sentiments та Cleared_review. Крім того потрібно провести попередню обробку тексту. Текст потрібно перетворити

в нижній регістр та видалити стоп слова. Після обробки датасет виглядає наступний чином. (Рис. 3.2.)

	text
0	wish would gotten one earlier love makes worki...
1	learned lesson open package use product right ...
2	slow lags find better option
3	roller ball stopped working within months mini...
4	like color size days return period hold charge
...	...
17335	love speaker love take anywhere charge phone w...
17336	use house easy connect loud clear music
17337	bass good battery amazing much better charge t...
17338	love
17339	mono speaker

Рис. 3.2. Датасет відгуків на товари

Поглянемо на баланс класів в датасеті (Рис. 3.3.).

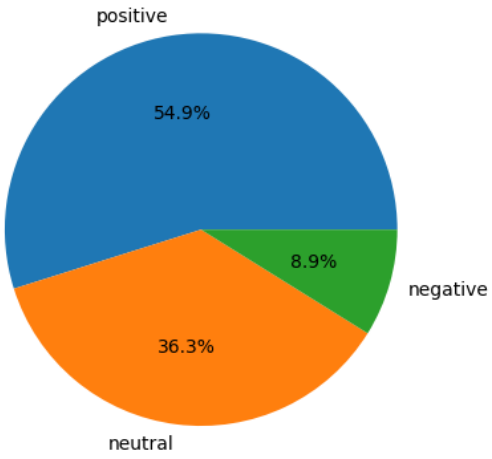


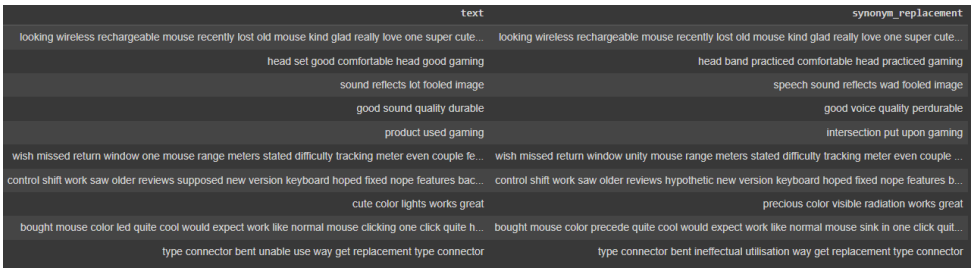
Рис. 3.3. Розподіл класів у датасеті

З графіка видно, що більшість класів це позитивні та нейтральні відгуки. В той час як негативні відгуки мають суттєву меншість. Це може негативно вплинути на процес навчання та фінальні передбачення моделі, тому для цієї вибірки буде доцільно застосувати аугментацію.

Розглянемо як будуть виглядати дані після аугментації з використанням різних методів збільшення текстових даних. Кожним метод аугментації застосувався два рази на реченні. Для цього будемо використовувати бібліотеку

textaugment та її клас EasyDataAugmentation. Вона надає різні функції для маніпуляції з реченням.

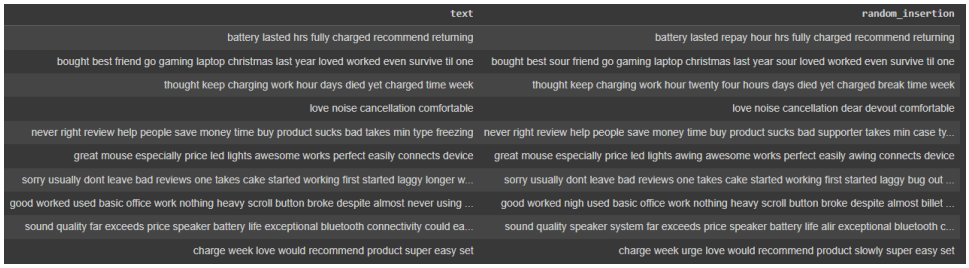
- **Заміна синонімів**. Можемо спостерігати, що нові речення містять інші синоніми схожі за змістом замість старих.



text	synonym_replacement
looking wireless rechargeable mouse recently lost old mouse kind glad really love one super cute...	looking wireless rechargeable mouse recently lost old mouse kind glad really love one super cute...
head set good comfortable head good gaming	head band practiced comfortable head practiced gaming
sound reflects lot fooled image	speech sound reflects wad fooled image
good sound quality durable	good voice quality perdurable
product used gaming	intersection put upon gaming
wish missed return window one mouse range meters stated difficulty tracking meter even couple fe...	wish missed return window unity mouse range meters stated difficulty tracking meter even couple ...
control shift work saw older reviews supposed new version keyboard hoped fixed nope features bac...	control shift work saw older reviews hypothetic new version keyboard hoped fixed nope features b...
cute color lights works great	precious color visible radiation works great
bought mouse color led quite cool would expect work like normal mouse clicking one click quite h...	bought mouse color precede quite cool would expect work like normal mouse sink in one click quit...
type connector bent unable use way get replacement type connector	type connector bent ineffectual utilisation way get replacement type connector

Рис. 3.4. Датасет аугментований з використанням методу заміни синонімів

- **Випадкове вставлення.** На випадкових позиціях в речені з’являються нові слова.



text	random_insertion
battery lasted hrs fully charged recommend returning	battery lasted repay hour hrs fully charged recommend returning
bought best friend go gaming laptop christmas last year loved worked even survive til one	bought best sour friend go gaming laptop christmas last year sour loved worked even survive til one
thought keep charging work hour days died yet charged time week	thought keep charging work hour twenty four hours days died yet charged break time week
love noise cancellation comfortable	love noise cancellation dear devout comfortable
never right review help people save money time buy product sucks bad takes min type freezing	never right review help people save money time buy product sucks bad supporter takes min case ty...
great mouse especially price led lights awesome works perfect easily connects device	great mouse especially price led lights awing awesome works perfect easily awing connects device
sorry usually dont leave bad reviews one takes cake started working first started laggy longer w...	sorry usually dont leave bad reviews one takes cake started working first started laggy bug out ...
good worked used basic office work nothing heavy scroll button broke despite almost never using ...	good worked nigh used basic office work nothing heavy scroll button broke despite almost billet ...
sound quality far exceeds price speaker battery life exceptional bluetooth connectivity could ea...	sound quality speaker system far exceeds price speaker battery life alir exceptional bluetooth c...
charge week love would recommend product super easy set	charge week urge love would recommend product slowly super easy set

Рис. 3.5. Датасет аугментований з використанням методу випадкового вставлення

- **Випадкові перестановки.** Слова у реченні змінюють свої позиції.

text	random_swap
cheaply made another get pay suggest pass	get cheaply another made pay suggest pass
lasted last months	last months lasted
great bass rich amazing sound	sound rich bass amazing great
broke weeks poorly put together	put broke poorly weeks together
packaging damaged looked used	looked damaged used packaging
less months purchasing scrolling wheel stopped working	months less working scrolling wheel stopped purchasing
sensitive mouse maybe new works fine	sensitive maybe works new mouse fine
mouse cool comfortable use fan sleep mode annoying still like mouse love color shifting feature	annoying cool comfortable use fan sleep mode mouse feature like mouse love color shifting still
always wanted mouse similar apple one found use windows laptop stylish rgb light looks pretty co...	like wanted mouse saving apple one found use windows laptop stylish rgb light looks pretty cool ...
love mine hear house	love house mine hear

Рис. 3.6. Датасет аугментований з використанням методу Випадкові перестановки

3.2.2. Аугментація зображень.

Для аугметації зображень було обрано датасет рентген знімків легень з пневмонією та без. Він складається 5863 рентгенівських зображення (JPEG) і та двох категорії (пневмонія/нормальний). Рентгенівські зображення грудної клітини були відібрані з медичного центру в Гуанчжоу. Усі рентгенівські дослідження органів грудної клітки проводилися як частина звичайної клінічної допомоги пацієнтам. Для аналізу рентгенівських зображень грудної клітки всі рентгенограми грудної клітки спочатку перевіряли для контролю якості шляхом видалення всіх сканів низької якості або нечитабельних.

Приклад зображень датасету (Рис 3.7.).

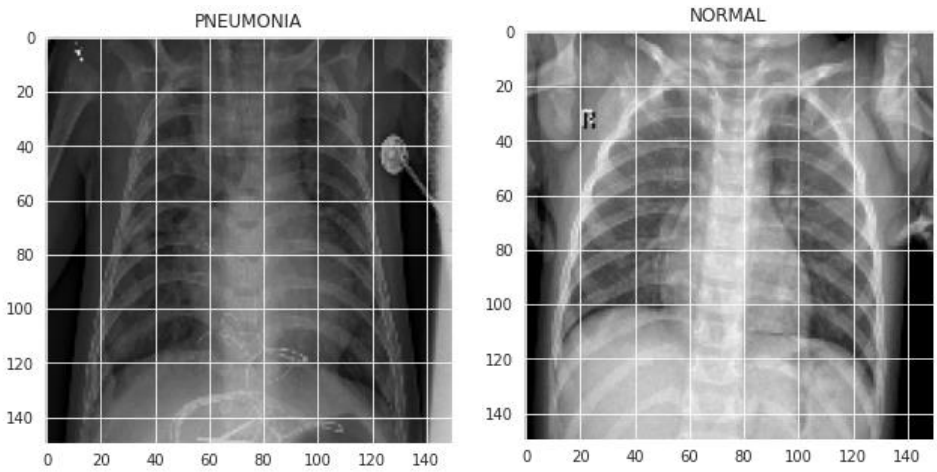


Рис. 3.7. Деякі зображення з датасету

Поглянемо на баланс класів в датасеті зображень (Рис. 3.8.).

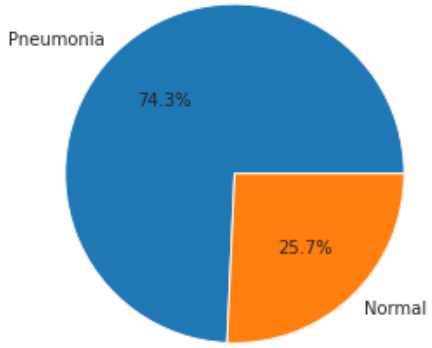


Рис. 3.8. Розподіл класів у датасеті зображень

У цій ситуації ситуація схожа. Клас зображень із захворюванням переважає здорові легені. З використанням аугментації будемо балансувати класи.

Для цього скористаємось різними техніками аугментації зображень. Використаємо для цього ImageDataGenerator з бібліотеки keras. Він працює застосовуючи різноманітні перетворення до зображень у вашому наборі даних, такі як масштабування, обертання, зміна приближення та перевертання. ImageDataGenerator доповнює дані на льоту протягом кожної епохи, надаючи пакети до нейронної мережі під час навчання. Тим самим ми гарантуємо, що модель бачить різні варіації того самого зображення в різні епохи.

Розглянемо як будуть виглядати дані після аугментації з використанням різних комбінацій параметрів для збільшення кількості зображень:

- **Комбінація поворотів зображень.** В цього випадку зображень будуть перевертатись випадково горизонтально або вертикально. Крім того вони такому можуть повертатись на певний кут.

```
model_rotation, history_rotation = run_test(x_train, y_train, x_val, y_val,
                                            rotation_range = 30,
                                            horizontal_flip = True,
                                            vertical_flip=True)
```

Рис. 3.9. Комбінація параметрів аугментації поворотами

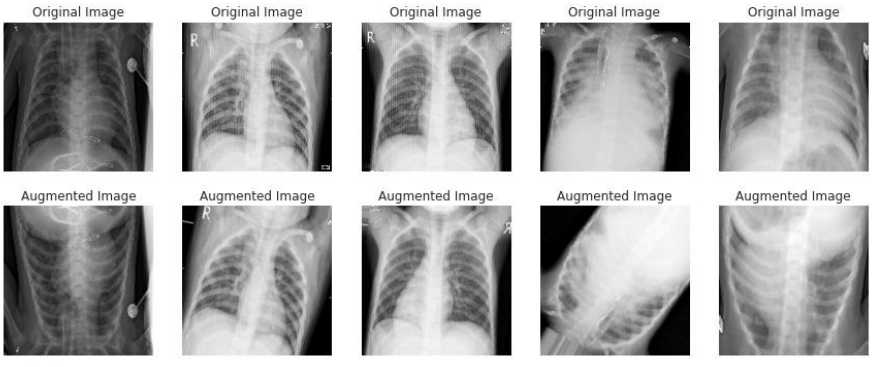


Рис. 3.10. Зображення аугментовані поворотами

- **Комбінація приближення та зсуву висоти або ширини зображень.** В цього випадку зображень будуть випадковим чином приближені або віддалені. Крім того вони можуть бути довільно зміщенні по

ширині чи довжині від загальної частки.

```
model_zoom, history_zoom = run_test(x_train, y_train, x_val, y_val,
                                    zoom_range = 0.2,
                                    width_shift_range=0.1,
                                    height_shift_range=0.1
                                   )
```

Рис. 3.11. Комбінація параметрів аугментації приближенням та зсувами

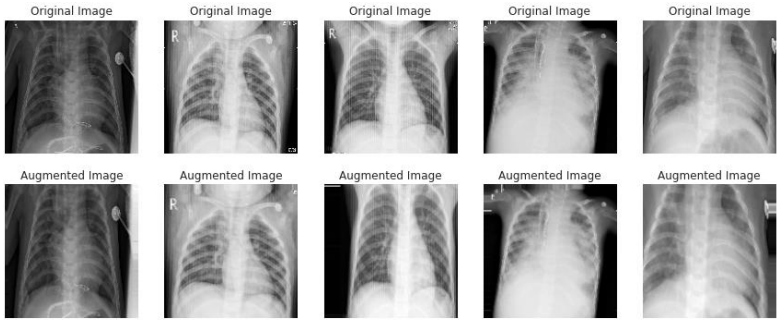


Рис. 3.12. Зображення аугментовані приближенням та зсувами

- **Комбінація зміни яскравості.** В цього випадку зображень будуть випадково міняти яскравість пікселів.

```
model_brightness, history_brightness = run_test(x_train, y_train, x_val, y_val,
                                                brightness_range=(0.5, 1.2),
                                                zoom_range = 0.2
                                                )
```

Рис. 3.13. Комбінація параметрів аугментації зміною яскравості

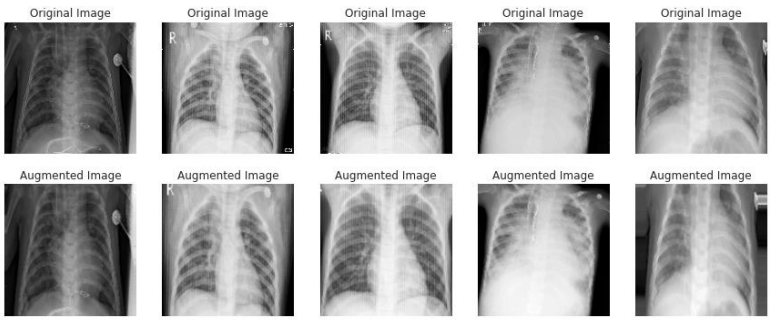


Рис. 3.14. Зображення аугментовані зміною яскравості

3.2.3. Аугментація аудіо.

Для класифікації аудіозаписів було образно датасет звуків тварин. Він містить звуки котів, собак та пташок. Переглянемо баланс класів (Рис 3.15).

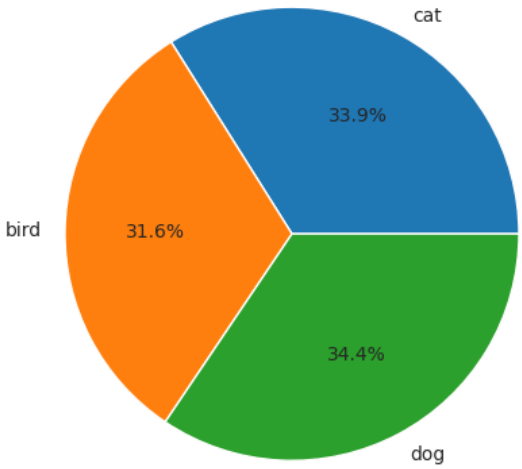


Рис. 3.15. Розподіл класів у датасеті аудіозаписів

В даному випадку класи збалансові, проте датасет налічує лише 610 екземплярів. Датасет досить маленький тому для нього також буде доцільно застосувати аугментацію даних. Проводити будемо її з використанням бібліотеки librosa. Спочатку аудіозаписи будуть аугментовані, після того з використання MFCC будуть видобуті головні ознаки аудіо.

Розглянемо як будуть виглядати аудіозаписи після аугментації з використанням різних методів:

- **Зміна швидкості.** У цьому методі аудіо буде пришвидшене або сповільнене на деяку випадкову величину з проміжку. Бачимо що аудіо має збільшену тривалість та загальна довжина також стала більшою.

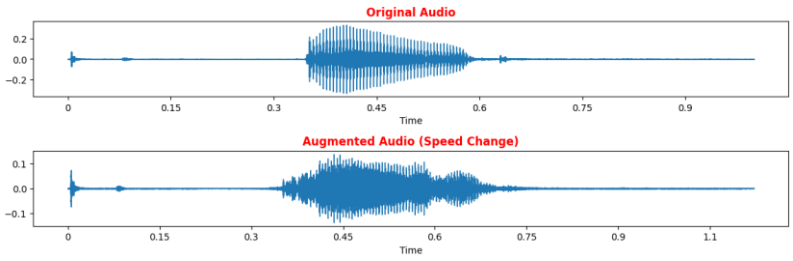


Рис. 3.16. Аудіозапис аугметований зміною швидкості

- **Зміна висоти тону.** З використання цього методу відбудеться зміна висоти або основної частоти аудіосигналу для створення модифікованої версії оригінального звуку. Можемо бачити, що на деяких ділянка частоти змінились порівняно з оригінальним звуком.

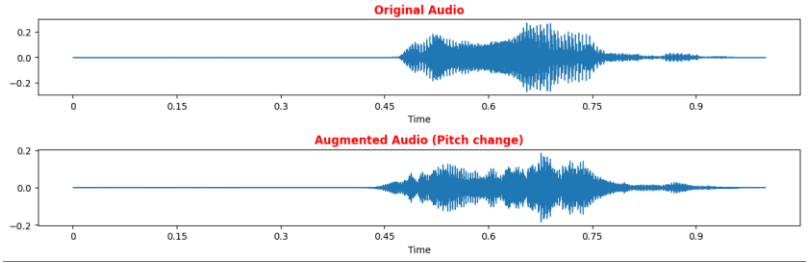


Рис. 3.17. Аудіозапис аугметований зміною швидкості

- **Випадові зсуви.** У цьому методі аудіо може випадково зсуватись по часовій лінії в певному проміжку для створення нових екземплярів. В результаті бачимо, що аудіо запис був зміщений в праву сторону по часовій лінії.

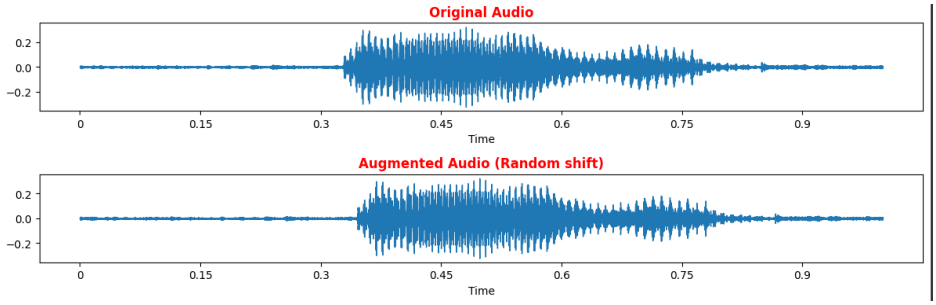


Рис. 3.18. Аудіозапис аугметований випадковим зсувом

3.3. Оцінка якості аугментованих даних.

З використання вище описаних моделей та методів машинного навчання проведемо передбачення цільового класу на оригінальний датасетах та на аугментованих датасетах з використання вище описаних методів для трьом модальностей даних: текст, зображення та аудіо. Крім того проведемо порівняльний аналіз отриманих результатів на оригінальних та штучно збільшених даних.

3.3.1. Аналіз передбачень текстових даних.

Для передбачення тональності тексту будемо використовувати Логістичну регресію. Спочатку проведемо тренування та оцінку результатів та оригінальному датасеті.

	precision	recall	f1-score	support
negative	0.87705	0.22385	0.35667	478
neutral	0.72464	0.82667	0.77230	1875
positive	0.86887	0.89698	0.88270	2844
accuracy			0.80970	5197
macro avg	0.82352	0.64916	0.67055	5197
weighted avg	0.81758	0.80970	0.79449	5197

Рис. 3.19. Класифікаційний звіт прогнозів на оригінальному датасеті

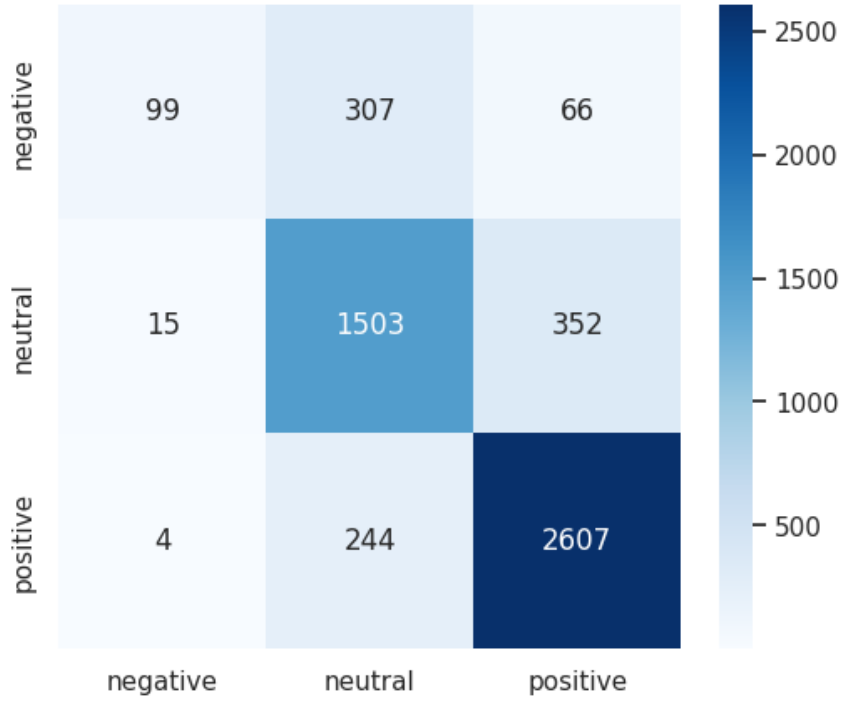


Рис. 3.20. Confusion matrix прогнозів на оригінальному датасеті

Бачимо, що модель виконує передбачення сентименту тексту з загальною точністю 80.97%.

Але є розбіжність у точності обробки для різних класів емоцій. Модель найбільш ефективно визначає позитивний сентимент (f1-score 88.27%).

Нейтральний сентимент також визначається досить добре (f1-score 77.23%).

Проте, модель значно слабше справляється з визначенням негативного сентименту, з f1-score лише 35.67%. Це говорить про те, що модель значно краще визначає позитивні й нейтральні відгуки, ніж негативні. Це також чітко видно на confusion matrix, клас негативних відгуків був передбачений найгірше.

Макроусереднений показник (macro avg) враховує незбалансованість класів і показує загальну ефективність моделі при рівному обробленні всіх класів. В цьому випадку f1-score макроусереднений показник становить 67.05%, що є досить низьким, що також підтверджує зазначені вище спостереження.

Тепер проведемо аналіз результатів на текстовому датасеті з використання вище описаних методів аугментації**.** Аугментація тексту проводилась з використання бібліотеки textaugment. З використанням кожного методу аугментації тексту, перш за все було досягнуто балансу класів та збільшено розмірність кожного класу у 2 рази. Це привело до наступних результатів:

- **Заміна синонімів**

	precision	recall	f1-score	support
negative	0.53249	0.73203	0.61651	459
neutral	0.80681	0.70471	0.75231	1849
positive	0.88614	0.90516	0.89555	2889
accuracy			0.81855	5197
macro avg	0.74181	0.78063	0.75479	5197
weighted avg	0.82668	0.81855	0.81994	5197

Рис. 3.21. Класифікаційний звіт прогнозів на аугментованому датасеті з використанням заміни синонімів

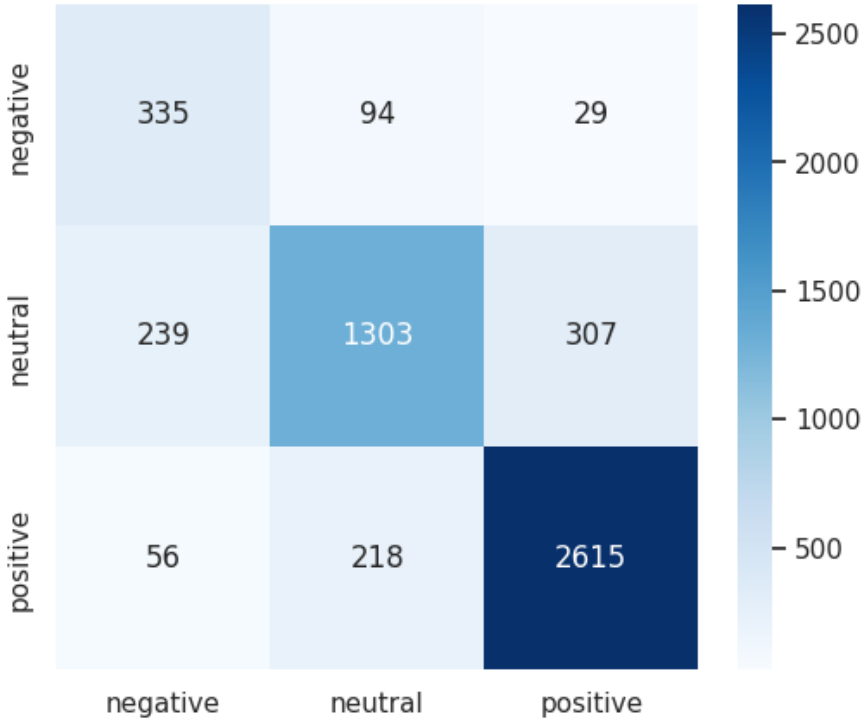


Рис. 3.22. Confusion matrix прогнозів на аугментованому датасеті з використанням заміни синонімів

Точність прогнозування позитивного класу залишилася високою (89%), і повнота також висока (91%). Це означає, що модель досить добре визначає позитивні відгуки.

Прогноз нейтрального класу покращився: точність становить 81%, а повнота - 70%.

Найбільше покращення спостерігається у прогнозуванні негативного класу. Precision знизилась до 53%, але recall підвищилася до 73%. Це означає, що модель виявляє більше негативних відгуків, навіть якщо деякі з них є помилковими.

Загальна точність моделі сягає 82%. Загалом модель стала краще прогнозувати нейтральну тональність за рахунок балансування, в той час як решта класів отримала лише невелике покращення. Це могло статись, тому що зміна деяких синонімів могла також вплинувати на тональність та змінити його тон, наприклад з позитивного на нейтральний.

- **Випадкові перестановки**

	precision	recall	f1-score	support
negative	0.83898	0.20975	0.33559	472
neutral	0.73174	0.80374	0.76606	1870
positive	0.86182	0.91313	0.88673	2855
accuracy			0.80989	5197
macro avg	0.81085	0.64221	0.66279	5197
weighted avg	0.81294	0.80989	0.79326	5197

Рис. 3.23. Класифікаційний звіт прогнозів на аугментованому датасеті з використанням випадкових перестановок

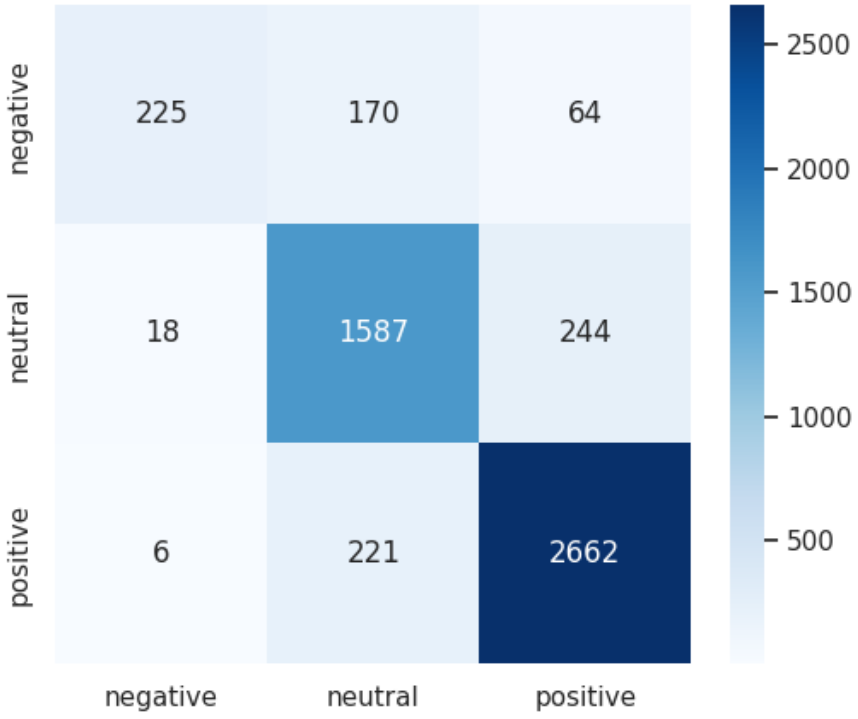


Рис. 3.24. Confusion matrix прогнозів на аугментованому датасеті з використанням випадкових перестановок

З використанням аугментації випадкових перестановок можна бачити значні покращення:

Для негативного класу: точність зросла з 0.84 до 0.90, а повнота значно збільшилася з 0.21 до 0.49. F1-міра також показала значне покращення, з 0.34 до 0.64.

Для нейтрального класу: точність зросла з 0.73 до 0.80, а повнота збільшилася з 0.80 до 0.86. F1-міра показала покращення з 0.77 до 0.83.

Для позитивного класу: значення точності зросло з 0.86 до 0.90, а повнота залишилася майже такою ж - спочатку 0.91, потім 0.92. F1-міра показала незначне покращення з 0.89 до 0.91.

Загальна точність моделі зросла з 0.81 у моделі, натренованої на оригінальному датасеті, до 0.86 у моделі, натренованої на аугментованому датасеті.

- **Випадкове вставлення**

	precision	recall	f1-score	support
negative	0.83394	0.50327	0.62772	459
neutral	0.79739	0.85776	0.82647	1849
positive	0.90242	0.91554	0.90893	2889
accuracy			0.85857	5197
macro avg	0.84458	0.75886	0.78771	5197
weighted avg	0.85900	0.85857	0.85476	5197

Рис. 3.25. Класифікаційний звіт прогнозів на аугментованому датасеті з використанням випадкових вставлень

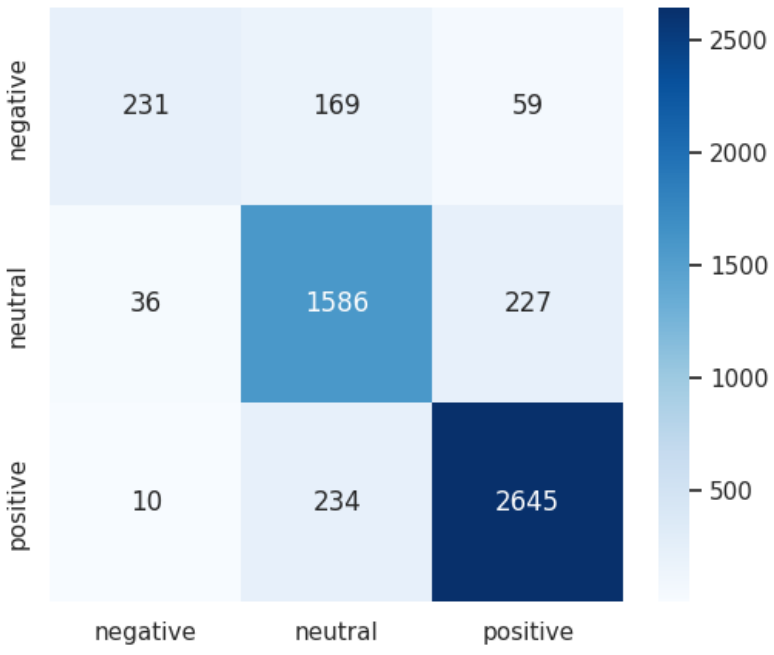


Рис. 3.26. Confusion matrix прогнозів на аугментованому датасеті з використанням випадкових вставлень

Перш за все можна побачити, що результати досить схожі до результатів отриманих з використання випадкових перестановок.

Для негативного класу: точність дещо знизилась з 0.84 до 0.83, однак повнота значно зросла з 0.21 до 0.50, що призвело до покращення F1-міри з 0.34 до 0.63.

Для нейтрального класу: точність зросла з 0.73 до 0.80, а повнота збільшилася з 0.80 до 0.86. Завдяки цьому F1-міра покращилась з 0.77 до 0.83.

Для позитивного класу: точність покращилась з 0.86 до 0.90, а повнота лише трохи зросла з 0.91 до 0.92. F1-міра показала незначне покращення з 0.89 до 0.91.

Загальна точність моделі зросла з 0.81 (на оригінальному датасеті) до 0.86 (на аугментованому датасеті).

3.3.2. Аналіз передбачень зображень.

Для виявлення хвороби пневмонії тексту будемо нейронну меред CNN. Спочатку проведемо тренування та оцінку результатів та оригінальному наборі зображень. Проводити тренування будемо на 15 епохах з використання функції зворотного виклику для зменшення кроку навчання.

	precision	recall	f1-score	support
Pneumonia (Class 0)	0.70	1.00	0.82	390
Normal (Class 1)	0.99	0.29	0.45	234
accuracy			0.73	624
macro avg	0.84	0.65	0.64	624
weighted avg	0.81	0.73	0.69	624

Рис. 3.27. Класифікаційний звіт прогнозів на оригінальному датасеті зображень

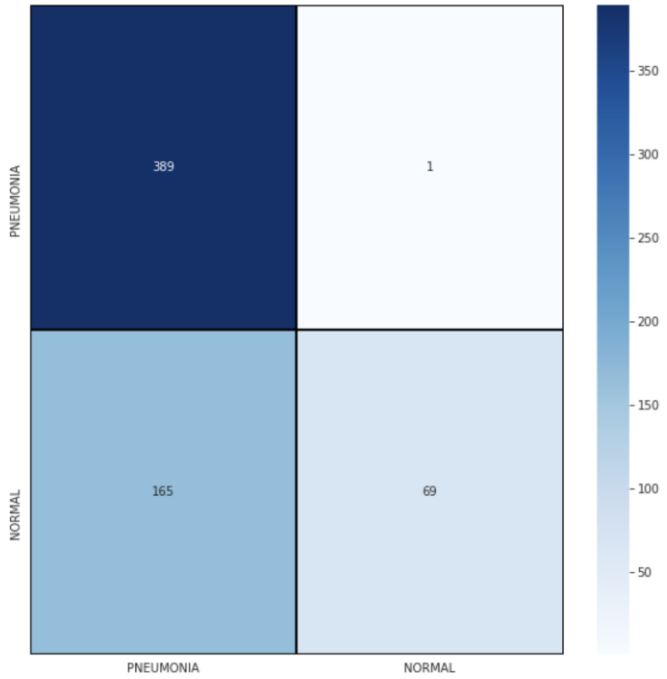


Рис. 3.28. Confusion matrix прогнозів на оригінальному датасеті зображень

За допомогою даної моделі, прогнозування пневмонії виявилося достатньо точним (Precision = 0.70) і показало високу чутливість до правильного визначення цього стану (Recall = 1.00), що призвело до узагальненої метрики F1-score = 0.82.

Однак, модель виявилася менш ефективною при визначенні випадків без пневмонії (нормальний стан - клас 1), з точністю 99%, але низькою чутливістю - лише 29%, що призвело до F1-score = 0.45.

Загальна точність моделі (accuracy) складає 73%, що хоч і вище випадкового вибору, але все ще залишає місце для покращення.

Тепер проведемо аналіз результатів прогнозування пневмонії з використання вище описаних методів аугментації**.** У якості аугментатора було використано ImageDataGenerator з бібліотеки Keras. Він під час навчання диманічно визначає класи меншості та балансує датасет. Крім того генерує нові штучні зразки кожного класу. З використання аугментації було отримано наступні результати:

- **Випадкові повороти зображень**

	precision	recall	f1-score	support
Pneumonia (Class 0)	0.88	0.93	0.90	390
Normal (Class 1)	0.87	0.78	0.82	234
accuracy			0.87	624
macro avg	0.87	0.86	0.86	624
weighted avg	0.87	0.87	0.87	624

Рис. 3.29. Класифікаційний звіт прогнозів на аугментованому датасеті зображень з використанням поворотів

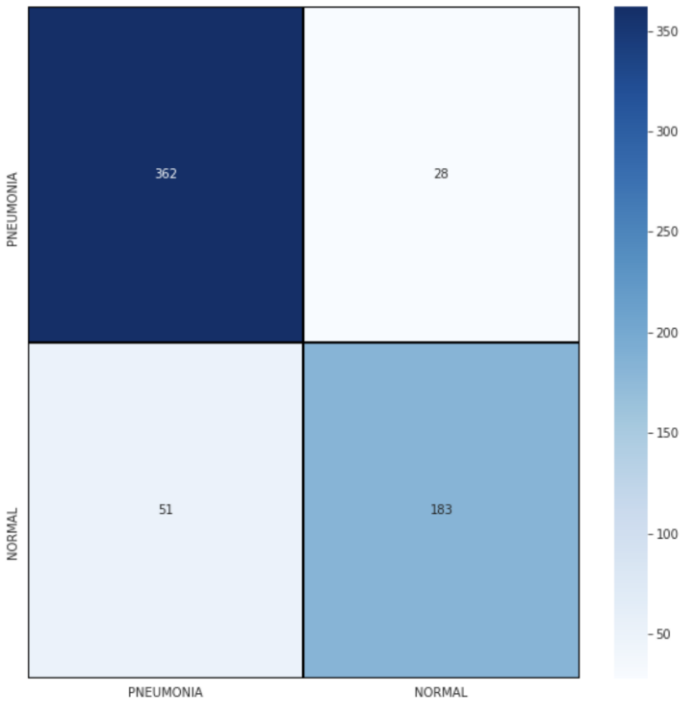


Рис. 3.30. Confusion matrix прогнозів на аугментованому датасеті зображень з використанням поворотів

У порівнянні з оригінальним датасетом, CNN, навчена на аугментованому датасеті з використанням поворотів, показала кращі результати.

Точність (precision) у визначенні пневмонії значно підвищилась, досягаючи 88%. Чутливість (recall) також зросла до 93%, забезпечуючи майже повне виявлення всіх випадків пневмонії. F1-оцінка порівняно з попередніми результатами зросла до 90%.

Щодо визначення нормального стану, модель також покращується.

Precision зросла до 87%, а recall досягла 78%. Це означає, що модель стала набагато менш "песимістичною" і розпізнавала більшу кількість здорових пацієнтів. Середнє гармонічне цих двох показників f1-score тепер становить 82%.

Загалом, точність моделі значно зросла, досягаючи 87%. Це свідчить про те, що модель значно ефективніше розпізнавала стан пацієнтів, як у разі пневмонії, так і при нормальному стані.

- **Випадкові приближення та зсуви висоти або ширини зображень**

	precision	recall	f1-score	support
Pneumonia (Class 0)	0.93	0.94	0.94	390
Normal (Class 1)	0.90	0.89	0.89	234
accuracy			0.92	624
macro avg	0.92	0.91	0.91	624
weighted avg	0.92	0.92	0.92	624

Рис. 3.31. Класифікаційний звіт прогнозів на аугментованому датасеті зображень з використанням зсувів та приближень

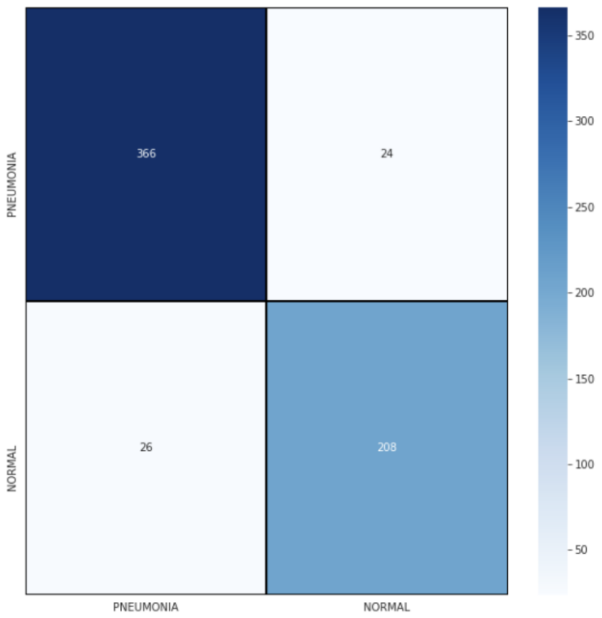


Рис. 3.32. Confusion matrix прогнозів на аугментованому датасеті зображень з використанням зсувів та приближень

У порівнянні з оригінальним датасетом результати застосування CNN для датасету аугментованої при випадкових зсувах та приближенях показали ще

кращі результати.

Для класу пневмонії precision зросла до 93%, а recall досягла 94%. Це означає, що модель визначила практично всі випадки пневмонії, і при цьому більшість передбачень були правильними. Загальний показник f1-score, який є середнім гармонічним між precision і recall, становить 94%.

У випадку щодо визначення нормального стану, точність збільшилася до 90%, чутливість досягла 89%. Це означає, що модель більш точно впізнає нормальний стан пацієнтів і рідше помиляється, призначаючи їм діагноз пневмонії. F1-оцінка цих результатів досягла 89%.

Загальна точність моделі (accuracy) зросла до 92%, що демонструє значне покращення її ефективності у визначенні стану пацієнтів.

- **Випадкові зміни яскравості**

	precision	recall	f1-score	support
Pneumonia (Class 0)	0.62	0.99	0.77	390
Normal (Class 1)	0.00	0.00	0.00	234
accuracy			0.62	624
macro avg	0.31	0.50	0.38	624
weighted avg	0.39	0.62	0.48	624

Рис. 3.33. Класифікаційний звіт прогнозів на аугментованому датасеті зображень з використанням зміни яскравості

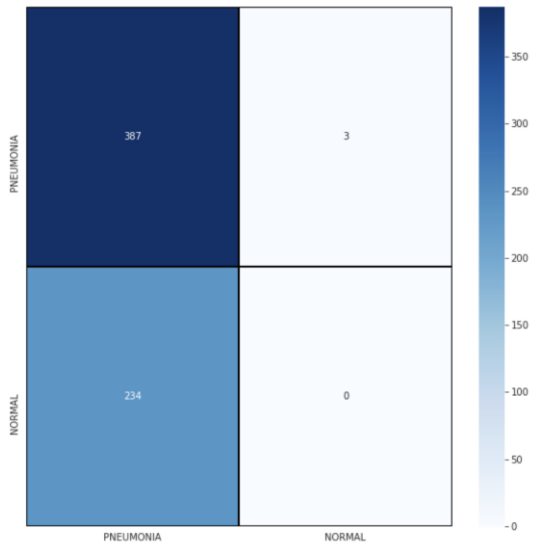


Рис. 3.34. Confusion matrix прогнозів на аугментованому датасеті зображень з використанням зміни яскравості

У порівнянні з оригінальним датасетом, результати використання CNN на аугментованому датасеті зі зміною яскравості значно гірші.

Precision виявлення пневмонії зменшилася до 62%, але водночас recall зросла і складає 99%. Це означає, що модель дійшла до крайності в тому, що передбачає пневмонію у великої кількості випадків, включаючи здорових пацієнтів. Як результат, f1-оцінка для цього класу становить 77%.

У випадку визначення нормального стану, модель досягла нульового значення precision, recall і f1-оцінки. Це означає, що модель абсолютно не розпізнала здорових пацієнтів, хоча в датасеті їх було 234.

Загальна точність моделі складає 62%, а загальна f1-оцінка - 38%.

Такий поганий результат моделі може бути пов'язаний з тим, що зміна яскравості впливає на контрастність зображень, яка є ключовим фактором для розпізнавання пневмонії на рентгенівських знімках. Таким чином, використання зміни яскравості в цій конкретній задачі, швидше за все, спотворило характеристики, на які модель повинна була звернути увагу, що призвело до дуже слабких результатів.

3.3.2. Аналіз передбачень аудіозаписів

Тренування будемо проводити з використанням персептрона. Перед тренування аудіозаписи були оброблені з використанням MFCC, щоб видобувати найважливіші ознака кожного аудіо. Процес тренування моделі для передбачення аудіозаписів є довгим процесом, тому модель буде тренуватись протягом 30 епох. Перш за все проведемо аналіз результаті на оригінальному датасеті.

	precision	recall	f1-score	support
bird	0.85000	0.80952	0.82927	42
cat	0.81481	0.57895	0.67692	38
dog	0.67273	0.88095	0.76289	42
accuracy			0.76230	122
macro avg	0.77918	0.75647	0.75636	122
weighted avg	0.77801	0.76230	0.75896	122

Рис. 3.35. Класифікаційний звіт прогнозів на оригінальному датасеті аудіо

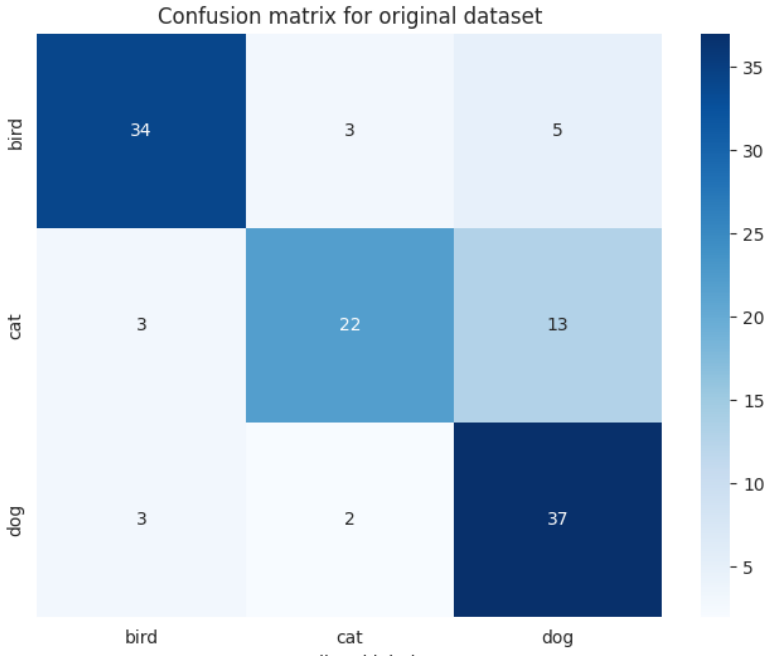


Рис. 3.36. Confusion matrix прогнозів на оригінальному датасеті аудіо

Результати показують, що модель передбачає голоси із загальною точністю 76.2%.

Для звуків птаха, модель має precision 85%, що означає, що з усіх звуків, які модель визначила як звуки птаха, правильними було 85%. Для звуків кота - 81%, а для собаки - 67%, що є найнижчим серед всіх класів.

Recall вказує, скільки від загальної кількості певного класу було правильно визначено. Так, для птахів було правильно визначено 80,95%, для котів - лише 57.89% і для собак - 88.1%.

Загальне значення f1-score для всіх трьох категорій є досить низьким (76,9%), що може свідчити про переважну зосередженість персептрону на одних категоріях за рахунок інших.

У деталях, модель має високу precision для птахів (85%), нижчу для котів (81.5%) і ще нижчу для собак (67.3%). Можемо побачити на confusion matrix, що передбачення для собак є нижчими.

Тепер з використанням вище описаних методів аугментації проведемо аналіз у порівнянні з оригінальним датасетом. Для реалізації методів аугментації аудіо використовувалась бібліотека librosa. Для усіх методів коефіцієнт штучного розширення вибірки дорівнював 2. Отримано наступні результати з використання методів аугментації:

- **Зміна швидкості**

	precision	recall	f1-score	support
bird	0.82222	0.88095	0.85057	42
cat	0.85294	0.76316	0.80556	38
dog	0.81395	0.83333	0.82353	42
accuracy			0.82787	122
macro avg	0.82971	0.82581	0.82655	122
weighted avg	0.82894	0.82787	0.82724	122

Рис. 3.37. Класифікаційний звіт прогнозів на аугментованому датасеті аудіо з використанням зміни швидкості

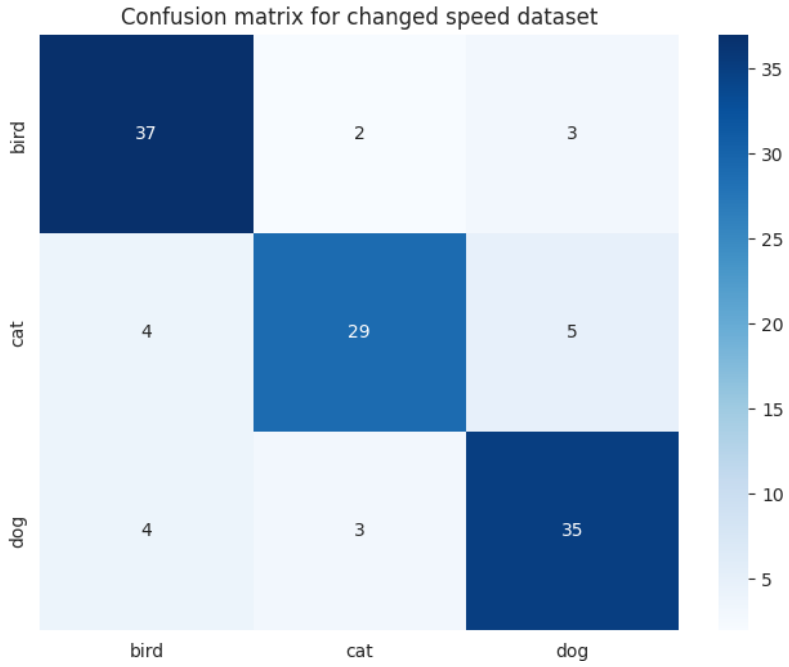


Рис. 3.38. Confusion matrix прогнозів на аугментованому датасеті аудіо з використанням зміни швидкості

У порівнянні з оригінальним датасетом маємо наступні зміни.

Для класу птахів precision зменшилася (з 85% до 82%), але recall і F1-оцінка зросли (з 81% до 88% і з 83% до 85% відповідно). Загалом, модель стала краще визначати пташині звуки.

Класів котів у всіх метриках показали поліпшення. Precision зросла з 81% до 85%, recall з 58% до 76%, а F1-оцінка із 68% до 80%. Це свідчить про значне покращення у визначенні котячих звуків.

Щодо класу собак, precision зросла з 67% до 81%, але повернення зменшилась із 88% до 83%. Це може вказувати на те, що модель стала краще розпізнавати собачі звуки, але іноді вона помилялася, вважаючи інші звуки за собачі.

Також загальна точність моделі значно зросла з 76% до 83%.

- **Зміна висоти тону**

	precision	recall	f1-score	support
bird	0.82927	0.80952	0.81928	42
cat	0.84211	0.84211	0.84211	38
dog	0.86047	0.88095	0.87059	42
accuracy			0.84426	122
macro avg	0.84395	0.84419	0.84399	122
weighted avg	0.84401	0.84426	0.84405	122

Рис. 3.39. Класифікаційний звіт прогнозів на аугментованому датасеті аудіо з використанням зміни тону

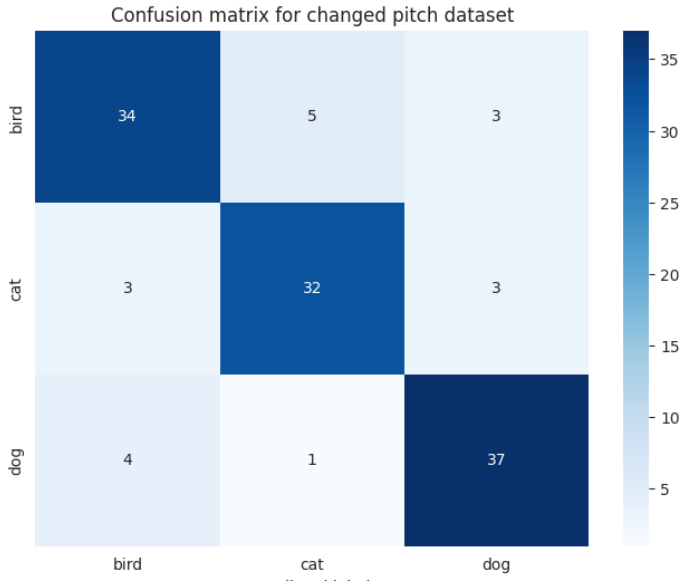


Рис. 3.40. Confusion matrix прогнозів на аугментованому датасеті аудіо з використанням зміни тону

Розглянемо висновки у порівнянні з оригінальним датасетом без аугментації.

Precision мінімально знизилася (з 85% до 83%), але вже ще знаходиться на

високому рівні для класу птахів. Recall скоротилося з 81% до 80.95%.

Класів котів у всіх метриках показали поліпшення. Precision збільшилася з 81% до 84.21%, доволі значне підвищення recall з 58% до 84.21% і F1-оцінка покращена з 68% до 84.21%. Це показує значне підвищення ефективності в пізнаванні звуків котів.

Precision підвищена з 67% до 86.04%, recall майже не змінилося (з 88% до 88.09%), а F1-оцінка покращена з 76% до 87.05% для звуків собак. Це свідчить про значне підвищення ефективності пізнавання собачих звуків.

Загальна точність моделі показує велике підвищення з 76% до 84.43%.

- **Випадові зсуви**

	precision	recall	f1-score	support
bird	0.82609	0.90476	0.86364	42
cat	0.85714	0.78947	0.82192	38
dog	0.82927	0.80952	0.81928	42
accuracy			0.83607	122
macro avg	0.83750	0.83459	0.83494	122
weighted avg	0.83686	0.83607	0.83537	122

Рис. 3.41. Класифікаційний звіт прогнозів на аугментованому датасеті аудіо з використанням випадкових зсувів

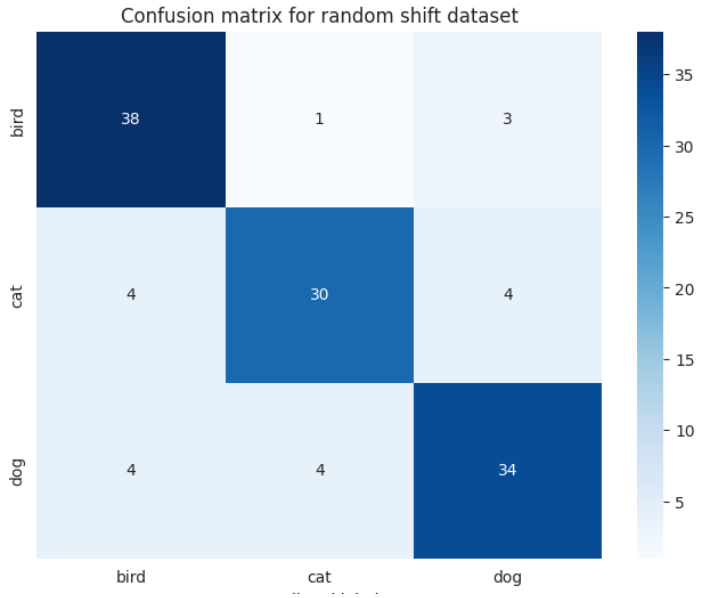


Рис. 3.42. Confusion matrix прогнозів на аугментованому датасеті аудіо з використанням випадкових зсувів

Результати випакових зсувів у порівнянні з оригінальним датасетом без аугментації:

Precision зменшилася (з 85% до 82.61%), але recall зросло (з 81% до 90.48%) для класу птахів. Загалом, модель краще визначає пташині звуки.

Precision зросла (з 81% до 85.71% %) та recall значно збільшилась (з 58% до 78.85%). а F1-оцінка також виросла з 68% до 82.19%. Загалом це свідчить про покращення ефективності моделі у визначенні котячих звуків.

Загалом, показники покращилися для собачих звуків, з точністю, яка зросла з 67% до 82.93%. Однак повернення незначно скоротилося з 88% до 80.95%.

Крім того загальна accuracy моделі показала покращення, зростаючи з 76% до 84%.

3.4. Висновки до третього розділу

Для трьох модальностей даних були проведені експерименти з використанням різних методів аугментації. Після застосування аугментації було проведено аналіз ефективності моделі та якості аугментованих даних з використанням вище зазначених метрик. Результати отримані з використанням аугментації були порівняні з результатами отримані на оригінальному датасеті, Для кожного порівняння, побудуємо таблиці для кожної модальності, які покажуть вплив аугментації та відсоток покращення метрик.

Розрахунок зміни результатів буде проводити за формулою:

$$Variation = (m_{\text{аугментований датасет}} - m_{\text{оригінальний датасет}}) \cdot 100\% \qquad (3.1.)$$

Таблиці показують на скільки відсотків покращились (зелений колір) або погіршились (червоний колір) результати:

- **Текст**

Таблиця 3.1 Зміна метрик текстових даних

Аугментація / Метрика	Заміна синонімів	Випадкові перестановки	Випадкові вставлення
Accuracy	0.9%	5.1%	4.9%
Precision	-6.9%	5.7%	3.4%
Recall	13.8%	11.4%	11.7%
F1-score	9.2%	12.8%	12.5%

Бачимо, що заміна синонімів показала дала найменший вплив на результати, у метриці precision значення навпаки погіршились. Це може бути, що заміна деяких синонімів вплинула на тональність речення та призвела до хибних результатів. В той час як аугментація з використанням випадкових перестановок чи вставлень добре підійшла для цього набору даних, і в результаті значно покращилась якість прогнозів.

- **Зображення**

Таблиця 3.2. Зміна метрик на вибірці зображень

Аугментація / Метрика	Випадкові повороти	Випадкові зсуви	Випадкові зміни яскравості
Accuracy	14%	19%	-11%
Precision	3%	8%	-53%
Recall	21%	26%	-15%
F1-score	22%	27%	-26%

З використання змін яскравості ми отримуємо значні погіршення. Це пов’язано з тим, що зображення рентгену дуже чутливе до таких змін, і в деяких випадках ми втрачаємо дані про пневмонію, або навпаки вважаємо, що пневмонія присутня на здорових знімках. Проте аугментація з використання поворотів та зсув дуже добре підійшли для цього набору даних і це дало нам

значне підвищення результатів передбачень.

- **Аудіозаписи**

Таблиця 3.3. Зміна метрик на вибірці аудіозаписів

Аугментація / **Метрика**	Зміна швидкості	Зміни висоти тону	Випадові зсуви
Accuracy	6.5%	8.2%	7.3%
Precision	5.2%	6.5%	5.8%
Recall	6.9%	8.77%	7.81%
F1-score	7%	8.76%	7.86%

Аугментація для датасету аудіозаписів повела себе схожим чином. Усі методи аугметнації показали подібні покращення. Використані методи добре підійшли для даного датасету, та не вплинули на зміну важливих ознака датесету.

Підсумовуючи, важливо сказати, що потрібно обирати метод аугментації відповідно до датасету, його характеристик, щоб досягнути найкращих результатів. Важливо, щоб аугментовані дані не були дуже спотвореними, що призведе до передбачення некоректного класу. Тому потрібно використовувати розумно параметри, які контролюють випадковий ступінь зміни одиниці даних, Також потрібно обирати метод залежно від даних. Неправильно підібрані методи можуть навпаки привести до суттєвого погіршення якості даних, та відповідно результатів. Наприклад для текстових даних контекст деяких синонімів може відрізнятись, що призведе до спотворення змісту. Для чорно-білих зображень зміна яскравості чи контрасту може вплинути негативно, у зв’язку з утворення білих або темних плям після аугментації. Отже, зазначимо, що потрібно проводити експерименти з оптимальними методами та параметрами, комбінувати деякі методи аугментації для виявлення найкращих методів додавання штучних даних для вирішення конкретної проблеми класифікації.

ВИСНОВКИ

В ході виконання даної магістерської кваліфікаційної роботи щодо порівняльного аналізу методів аугментації даних було здійсненно дослідження наукової літератури про методи аугментації даних для різних модальностей, за допомогою методології PRISMA на платформах Scopus та Google Scholar, здійснено критичний аналіз ключових джерел. Результатом стала ідентифікація основною проблеми аугментації даних і визначення досліджень потрібних для збільшення точності та узагальнюючої здатності моделей машинного навчання та штучного інтелекту. Подальший аналіз, розробка критеріїв оцінки методів аугментації та експерименти, результати яких будуть використані для визначення найбільш оптимальних підходів аугментації даних, забезпечать підставу для подальших досліджень.

У другому розділі обговорено аспекти класифікації застосованої до різних модальностей даних, включаючи текст, зображення, аудіо. Висвітлено проблематику класифікації в контексті роботи з незбалансованими даними, наслідки якої можна усунути застосуванням методів штучного доповнення даних для недостатньо представлених класів. Різні класифікатори, від логістичної регресії до згорткових нейронних мереж, можуть бути використані в класифікації залежно від ряду факторів. Було описано важливість оцінки якості моделей за допомогою різних показників ефективності для визначення їх продуктивності у відповідних сценаріях.

У третьому розділі були проведені експерименти для трьох модальностей даних з використанням трьом різних аугментації для кожної із модальнестей. Експерименти проводились на оригінальному датасеті та аугментованих. Після цього був проведений порівняльний аналіз покращень передбачень для визначення впливу кожного з методів на якісь передбачень. Крім того було обрано найкращі методи аугментації для кожного датасету.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] «PRISMA Methodology for Systematic Review», DistillerSR. Дата звернення: 12, Листопад 2023. [Online]. Доступний у: https://www.distillersr.com/resources/systematic-literature-reviews/prisma-methodology-for-systematic-review

[2] A. L. C. Ottoni, R. M. de Amorim, M. S. Novo, і D. B. Costa, «Tuning of data augmentation hyperparameters in deep learning to building construction image classification with small datasets», *Intl. J. Mach. Learn. Cybern.*, вип. 14, вип. 1, Art. вип. 1, 2023, doi: 10.1007/s13042-022-01555-1.

[3] M. A. Kutlugun, Y. Sirin, і M. Karakaya, «The effects of augmented training dataset on performance of convolutional neural networks in face recognition system», в *Proc. Fed. Conf. Comput. Sci. Inf. Syst., FedCSIS*, Ganzha M., Maciaszek L., Maciaszek L., і Paprzycki M., Ред., Institute of Electrical and Electronics Engineers Inc., 2019, с. 929–932. doi: 10.15439/2019F181.

[4] O. O. Abayomi-Alli, R. Damasevicius, R. Maskeliunas, і A. Abayomi-Alli, «BiLSTM with Data Augmentation using Interpolation Methods to Improve Early Detection of Parkinson Disease», в *Proc. Fed. Conf. Comput. Sci. Inf. Syst., FedCSIS*, Ganzha M., Maciaszek L., Maciaszek L., і Paprzycki M., Ред., Institute of Electrical and Electronics Engineers Inc., 2020, с. 371–380. doi: 10.15439/2020F188.

[5] L. Taylor і G. Nitschke, «Improving deep learning with generic data augmentation», в *2018 IEEE symposium series on computational intelligence (SSCI)*, IEEE, 2018, с. 1542–1547.

[6] W. Alosaimi і M. I. Uddin, «Efficient Data Augmentation Techniques for Improved Classification in Limited Data Set of Oral Squamous Cell Carcinoma», *CMES Comput. Model. Eng. Sci.*, вип. 131, вип. 3, Art. вип. 3, 2022, doi: 10.32604/cmes.2022.018433.

[7] K. Kim і J. Jeong, «Deep learning-based data augmentation for hydraulic condition monitoring system», в *Procedia Comput. Sci.*, Shakshuki E., Yasar A-U-H., і Malik H., Ред., Elsevier B.V., 2020, с. 20–27. doi: 10.1016/j.procs.2020.07.007.

[8] M. Bayer, M.-A. Kaufhold, B. Buchhold, M. Keller, J. Dallmeyer, і C. Reuter, «Data augmentation in natural language processing: a novel text generation approach for long and short text classifiers», *Intl. J. Mach. Learn. Cybern.*, вип. 14, вип. 1, Art. вип. 1, 2023, doi: 10.1007/s13042-022-01553-3.

[9] A. Mikołajczyk і M. Grochowski, «Data augmentation for improving deep learning in image classification problem», в *2018 international interdisciplinary PhD workshop (IIPhDW)*, IEEE, 2018, с. 117–122.

[10] K. Dunphy, M. N. Fekri, K. Grolinger, і A. Sadhu, «Data Augmentation for Deep-Learning-Based Multiclass Structural Damage Detection Using Limited Information», *Sensors*, вип. 22, вип. 16, Art. вип. 16, 2022, doi: 10.3390/s22166193.

[11] R. Pappagari, J. Villalba, P. Zelasko, L. Moro-Velazquez, і N. Dehak, «Copypaste: An augmentation method for speech emotion recognition», в *ICASSP IEEE Int Conf Acoust Speech Signal Process Proc*, Institute of Electrical and Electronics

Engineers Inc., 2021, с. 6324–6328. doi: 10.1109/ICASSP39728.2021.9415077.
[12] N. F. Aminuddin, Z. Tukiran, A. Joret, R. Tomari, і M. Morsin, «An Improved Deep Learning Model of Chili Disease Recognition with Small Dataset», *Intl. J. Adv. Comput. Sci. Appl.*, вип. 13, вип. 7, Art. вип. 7, 2022, doi: 10.14569/IJACSA.2022.0130750.
[13] S. T. Aroyehun і A. Gelbukh, «Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling», в *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, 2018, с. 90–97.
[14] C. Shorten і T. M. Khoshgoftaar, «A survey on Image Data Augmentation for Deep Learning», *J. Big Data*, вип. 6, вип. 1, Art. вип. 1, 2019, doi: 10.1186/s40537-019-0197-0.
[15] T. Dao, A. Gu, A. Ratner, V. Smith, C. De Sa, і C. Ré, «A kernel theory of modern data augmentation», в *International Conference on Machine Learning*, PMLR, 2019, с. 1528–1537.
[16] L. Brigato і L. Iocchi, «A close look at deep learning with small data», в *Proc. Int. Conf. Pattern Recognit.*, Institute of Electrical and Electronics Engineers Inc., 2020, с. 2490–2497. doi: 10.1109/ICPR48806.2021.9412492.
[17] «Statistical classification», *Wikipedia*. 30, Жовтень 2023. Дата звернення: 12, Листопад 2023. [Online]. Доступний у: https://en.wikipedia.org/w/index.php?title=Statistical_classification&oldid=1182690049
[18] T. Dietterich, «Overfitting and undercomputing in machine learning», *ACM Comput. Surv.*, вип. 27, вип. 3, с. 326–327, Вер 1995, doi: 10.1145/212094.212114.
[19] C. Shorten, T. M. Khoshgoftaar, і B. Furht, «Text Data Augmentation for Deep Learning», *J Big Data*, вип. 8, вип. 1, Art. вип. 1, Груд 2021, doi: 10.1186/s40537-021-00492-0.
[20] A. Jain, P. R. Samala, D. Mittal, P. Jyoti, і M. Singh, «SpliceOut: A Simple and Efficient Audio Augmentation Method», arXiv.org. Дата звернення: 12, Листопад 2023. [Online]. Доступний у: https://arxiv.org/abs/2110.00046v2
[21] «Logistic Regression - an overview | ScienceDirect Topics». Дата звернення: 12, Листопад 2023. [Online]. Доступний у: https://www.sciencedirect.com/topics/computer-science/logistic-regression
[22] «A Guide to Convolutional Neural Networks — the ELI5 way | Saturn Cloud Blog». Дата звернення: 12, Листопад 2023. [Online]. Доступний у: https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/
[23] U. Kiran, «MFCC Technique for Speech Recognition», Analytics Vidhya. Дата звернення: 12, Листопад 2023. [Online]. Доступний у: https://www.analyticsvidhya.com/blog/2021/06/mfcc-technique-for-speech-recognition/
[24] alexandre, «Perceptron: Concept, function, and applications», Data Science Courses | DataScientest. Дата звернення: 12, Листопад 2023. [Online]. Доступний у: https://datascientest.com/en/perceptron-definition-and-use-cases
[25] T. Srivastava, «12 Important Model Evaluation Metrics for Machine Learning

Everyone Should Know (Updated 2023)», Analytics Vidhya. Дата звернення: 12, Листопад 2023. [Online]. Доступний у: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/

ДОДАТОК А.

```
# !pip install nlpaug
# !pip install transformers
# !pip install textaugment

import os
import nltk
import random
import pandas as pd
import shutil
import numpy as np
import librosa
import seaborn as sns
import matplotlib.pyplot as plt

from PIL import Image
from nltk.corpus import stopwords
from textaugment import EDA
from sklearn.svm import SVC
from sklearn.feature_extraction.text import CountVectorizer
from nltk.tokenize import word tokenize
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, ParameterGrid
from sklearn.metrics import accuracy_score, f1_score, classification_report
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import LinearSVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.utils import shuffle
from collections import Counter

from google.colab import drive
drive.mount('/content/drive', force_remount=True)

from google.colab import files
uploaded = files.upload()

# !mkdir -p ~/.kaggle && mv kaggle.json ~/.kaggle/ && chmod 600 ~/.kaggle/kaggle.json

"""# Text augmentation"""

nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

def plot_results(history_original, history_augmented, method):
  fig, ax = plt.subplots(1, 2, figsize=(12, 5))

  ax[0].plot(history_original.history['loss'], label='Training Loss original')
  ax[0].plot(history_original.history['val_loss'], label='Validation Loss original')

  ax[0].plot(history_augmented.history['loss'], label=f'Training Loss augmened using {method}')
  ax[0].plot(history_augmented.history['val_loss'], label=f'Validation Loss augmened using
{method}')

  ax[0].legend()
  ax[0].grid()
  ax[0].set_title('Loss')

  ax[1].plot(history_original.history['accuracy'], label='Training Accuracy original')
  ax[1].plot(history_original.history['val_accuracy'], label='Validation Accuracy original')
  ax[1].plot(history_augmented.history['accuracy'], label=f'Training Accuracy augmened using
{method}')
  ax[1].plot(history_augmented.history['val_accuracy'], label=f'Validation Accuracy augmened using
{method}')

  ax[1].legend()
  ax[1].grid()
  ax[1].set_title('Accuracy')

  plt.tight_layout()
  plt.show()


def valid_string(sentence):
    return isinstance(sentence, str) and len(sentence.strip()) > 0
```

```
def get_confusion_matrix_for_text(model, y_pred, y_test,title):
  cm = confusion matrix(y test, y pred)

  plt.figure(figsize=(6, 5))

  class_names = np.unique(y_test)

  sns.set(font_scale=1)  # for label size
  sns.heatmap(cm, cmap= "Blues", linecolor = 'black', annot = True, fmt='', xticklabels=class_names,
yticklabels=class_names)

  plt.xlabel("Predicted Labels", fontsize=16)
  plt.ylabel("True Labels", fontsize=16)

  plt.title(title)
  plt.show()

def custom_synonym_replacement(sentence, n):
    tokens = nltk.word_tokenize(sentence)
    num_replacements = min(n, len(tokens))

    for _ in range(num_replacements):
        words_perturbed = set()
        while len(words_perturbed) < num_replacements:
            if len(words_perturbed) == len(tokens):
                break
            word_idx = random.randint(0, len(tokens) - 1)
            token = tokens[word_idx]
            if token in words_perturbed or token not in eda.model.wv.vocab or token in
eda.stop_words:
                continue
            synonyms = eda._synonyms(token)
            if len(synonyms) > 0:
                tokens[word_idx] = random.sample(synonyms, 1)[0]
            words_perturbed.add(token)

    return " ".join(tokens)

def augment_and_balance(X, y, factor=1, augmentation_func=None, n=1):
    augmented_X = list(X)
    augmented_y = list(y)

    if not augmentation_func:
      augmenter = EDA()
      augmentation_func = augmenter.synonym_replacement

    if factor != 1:
      for i in range(len(X)):
          for _ in range(factor):
              try:
                  new_text = augmentation_func(X[i], n)
                  augmented X.append(new text)
                  augmented_y.append(y[i])
              except:
                  continue

    classes = np.unique(augmented_y)
    class counts = Counter(augmented y)
    max_count = max(class_counts.values())

    for class_ in classes:
        class_count = class_counts[class_]
        class_samples = [x for x, label in zip(augmented_X, augmented_y) if label == class_]
        if class_count < max_count:
            sampling_indices = np.random.choice(list(range(class_count)), size=max_count-
class_count)

            additional_samples = []
            for i in sampling_indices:
              try:
                  new_sample = augmentation_func(class_samples[i], n)
                  additional_samples.append(new_sample)
              except Exception as e:
                  print(f"Failed to augment sample {class_samples[i]} due to: {str(e)}")

            additional_labels = [class_] * len(additional_samples)

            augmented_X.extend(additional_samples)
            augmented_y.extend(additional_labels)

    augmented_X, augmented_y = shuffle(augmented_X, augmented_y)
```

```
    return augmented_X, augmented_y

def load_text_data():
  data_raw = pd.read_csv("/content/drive/MyDrive/cleaned_reviews.csv")
  stop_words = set(stopwords.words('english'))

  data = data_raw.dropna(subset=['cleaned_review'])
  data = data[data['cleaned_review'].apply(valid_string)]

  data['text'] = data['cleaned_review'].str.lower()
  data['text'] = data['text'].apply(lambda x: ' '.join([word for word in x.split() if word not in
stop_words]))

  return data_raw, data['text'], data['sentiments']

data_raw, X, y = load_text_data()

data_raw['sentiments'].value_counts()

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)

def run_text_augmentation_test(X_train, y_train, X_test, y_test, factor, augm_func, n, method):
  X_train_augmented, y_train_augmented = augment_and_balance(X_train, y_train, factor, augm_func, n)

  X_train_augmented = vectorizer.transform(X_train_augmented)

  model = LogisticRegression(max_iter=1000)
  model.fit(X_train_augmented, y_train_augmented)

  y_pred = model.predict(X_test)

  print(f"Model trained on dataset augmented by {method}:")
  print(classification_report(y_test, y_pred, digits=5))

  get_confusion_matrix_for_text(model, y_pred, y_test, f'Confusion matrix for trained on dataset
with {method}')

category_counts = pd.DataFrame(y)['sentiments'].value_counts()

# Plot the count as a pie chart
plt.pie(category_counts, labels=category_counts.index, autopct='%.1f%%')
plt.title('Category Counts')

# Show the plot
plt.show()

eda = EDA()

vectorizer = TfidfVectorizer()
X_train_original_1 = vectorizer.fit_transform(X_train)
X_test_original_1 = vectorizer.transform(X_test)

model_original = LogisticRegression(max_iter=1000)
model_original.fit(X_train_original_1, y_train)
y_pred_original = model_original.predict(X_test_original_1)

print("Model trained on original dataset:")
print(classification report(y test, y pred original, digits=5))

get_confusion_matrix_for_text(model_original, y_pred_original, y_test, 'Confusion matrix for trained
on original dataset')

run_text_augmentation_test(X_train, y_train, X_test_original_1, y_test, 2, eda.synonym_replacement,
2, 'synonym replacement')

run_text_augmentation_test(X_train, y_train, X_test_original_1, y_test, 2, eda.random_swap, 3,
'random swap')

run_text_augmentation_test(X_train, y_train, X_test_original_1, y_test, 2, eda.random_insertion, 2,
'random insertion')

"""# Image augmentation"""

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import keras
from keras.models import Sequential
from keras.layers import Dense, Conv2D , MaxPool2D , Flatten , Dropout , BatchNormalization
from keras.preprocessing.image import ImageDataGenerator
```

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,confusion_matrix
from keras.callbacks import ReduceLROnPlateau
import cv2
import os
import random

labels = ['PNEUMONIA', 'NORMAL']
img_size = 150
def get_training_data(data_dir):
    data = []
    for label in labels:
        path = os.path.join(data_dir, label)
        class_num = labels.index(label)
        for img in os.listdir(path):
            try:
                img_arr = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
                resized_arr = cv2.resize(img_arr, (img_size, img_size)) # Reshaping images to
preferred size
                data.append([resized_arr, class_num])
            except Exception as e:
                print(e)
    return np.array(data)

train = get_training_data('/content/drive/MyDrive/chest_xray/train')
test = get_training_data('/content/drive/MyDrive/chest_xray/test')
val = get_training_data('/content/drive/MyDrive/chest_xray/val')

l = []
for i in train:
    if(i[1] == 0):
        l.append("Pneumonia")
    else:
        l.append("Normal")

# Calculate counts for each category
counts = {i: l.count(i) for i in set(l)}

# Plot pie chart
sns.set_style('darkgrid')
plt.pie(counts.values(), labels=counts.keys(), autopct='%.1f%%')
plt.axis('equal')
plt.show()

plt.figure(figsize = (5,5))
plt.imshow(train[0][0], cmap='gray')
plt.title(labels[train[0][1]])

x_train = []
y_train = []

x_val = []
y_val = []

x_test = []
y_test = []

for feature, label in train:
    x_train.append(feature)
    y_train.append(label)

for feature, label in test:
    x_test.append(feature)
    y_test.append(label)

for feature, label in val:
    x_val.append(feature)
    y_val.append(label)

x_train = np.array(x_train) / 255
x_val = np.array(x_val) / 255
x_test = np.array(x_test) / 255

x_train = x_train.reshape(-1, img_size, img_size, 1)
y_train = np.array(y_train)

x_val = x_val.reshape(-1, img_size, img_size, 1)
y_val = np.array(y_val)

x_test = x_test.reshape(-1, img_size, img_size, 1)
y_test = np.array(y_test)
```

```
def display_augmented_images(datagen, x_train, y_train, num_pairs=5):
    fig = plt.figure(figsize=(3 * num_pairs, 6))

    for i in range(num_pairs):
        original_image, original_label = x_train[i], y_train[i]
        generator = datagen.flow(np.expand_dims(original_image, axis=0),
                                 np.expand_dims(original_label, axis=0),
                                 batch_size=1)
        augmented_image, augmented_label = generator.next()

        plt.subplot(2, num_pairs, i + 1)
        plt.imshow(np.squeeze(original_image), cmap='gray')
        plt.title("Original Image")
        plt.axis('off')

        plt.subplot(2, num_pairs, i + num_pairs + 1)
        plt.imshow(np.squeeze(augmented_image), cmap='gray')
        plt.title("Augmented Image")
        plt.axis('off')

    plt.show()

def create_custom_datagen(x_train, **kwargs):
    datagen = ImageDataGenerator(**kwargs)

    datagen.fit(x_train)

    return datagen

def run_test(x_train, y_train, x_val, y_val,**kwargs):
    datagen = create_custom_datagen(x_train, **kwargs)

    display_augmented_images(datagen, x_train, y_train)

    learning_rate_reduction = ReduceLROnPlateau(monitor='val_accuracy', patience = 2,
verbose=1,factor=0.3, min_lr=0.000001)

    model = get_cnn_model()
    history = model.fit(datagen.flow(x_train,y_train, batch_size = 32) ,epochs = 12 ,
validation_data = datagen.flow(x_val, y_val) ,callbacks = [learning_rate_reduction])

    print("Loss of the model is - " , model.evaluate(x_test,y_test)[0])
    print("Accuracy of the model is - " , model.evaluate(x_test,y_test)[1]*100 , "%")

    return model, history

def get predictions audio(model, x test, y test):
    predictions = model.predict_classes(x_test)
    predictions = predictions.reshape(1,-1)[0]

    print(classification_report(y_test, predictions, target_names = ['Pneumonia (Class 0)','Normal
(Class 1)']))

    cm = confusion_matrix(y_test,predictions)
    cm = pd.DataFrame(cm , index = ['0','1'] , columns = ['0','1'])

    plt.figure(figsize = (10,10))
    sns.heatmap(cm,cmap= "Blues", linecolor = 'black' , linewidth = 1 , annot = True,
fmt='',xticklabels = labels,yticklabels = labels)

    plt.show()

datagen = ImageDataGenerator()
datagen.fit(x_train)

display_augmented_images(datagen, x_train, y_train)

def get_cnn_model():
    model = Sequential()

    model.add(Conv2D(32 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu' , input_shape
= (150,150,1)))
    model.add(BatchNormalization())
    model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
    model.add(Conv2D(64 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
    model.add(Dropout(0.1))
    model.add(BatchNormalization())
    model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
    model.add(Conv2D(64 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
    model.add(BatchNormalization())
    model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
    model.add(Conv2D(128 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
```

```
    model.add(Dropout(0.2))
    model.add(BatchNormalization())
    model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
    model.add(Conv2D(256 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
    model.add(Dropout(0.2))
    model.add(BatchNormalization())
    model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
    model.add(Flatten())
    model.add(Dense(units = 128 , activation = 'relu'))
    model.add(Dropout(0.2))
    model.add(Dense(units = 1 , activation = 'sigmoid'))

    model.compile(optimizer = "adam" , loss = 'binary_crossentropy' , metrics = ['accuracy'])

    return model

learning_rate_reduction = ReduceLROnPlateau(monitor='val_accuracy', patience = 2,
verbose=1,factor=0.3, min_lr=0.000001)

model_original = get_cnn_model()

history_original = model_original.fit(datagen.flow(x_train,y_train, batch_size = 32) ,epochs = 12 ,
validation_data = datagen.flow(x_val, y_val) ,callbacks = [learning_rate_reduction])

get_predictions_audio(model_original, x_test, y_test)

model_rotation, history_rotation = run_test(x_train, y_train, x_val, y_val,
                                            rotation_range = 30,
                                            horizontal_flip = True,
                                            vertical_flip=True)

get_predictions_audio(model_rotation, x_test, y_test)

model_zoom, history_zoom = run_test(x_train, y_train, x_val, y_val,
                                            zoom_range = 0.2,
                                            width_shift_range=0.1,
                                            height_shift_range=0.1
                                           )

get_predictions_audio(model_zoom, x_test, y_test)

model_brightness, history_brightness = run_test(x_train, y_train, x_val, y_val,
                                            brightness_range=(0.5, 1.2),
                                            zoom_range = 0.2
                                            )

get predictions audio(model brightness, x test, y test)

"""# Audio augmentation"""

# !kaggle datasets download -d warcoder/cats-vs-dogs-vs-birds-audio-classification
#
# !unzip cats-vs-dogs-vs-birds-audio-classification.zip

import IPython.display as ipd
import os
import numpy as np
import librosa
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.utils import to_categorical

def extract_features(audio, sample_rate):
    try:
        mfccs = librosa.feature.mfcc(y=audio, sr=sample_rate, n_mfcc=40)
        mfccs_scaled = np.mean(mfccs.T, axis=0)
    except Exception as e:
        print("Error encountered while parsing file")
        return None

    return mfccs_scaled

def change_speed(audio, sample_rate):
    speed_change = np.random.uniform(low=0.8, high=1.2)
    tmp = librosa.effects.time_stretch(audio, rate=speed_change)
    return tmp

def change_pitch(audio, sample_rate):
    pitch_change = np.random.uniform(low=-5, high=5)
    return librosa.effects.pitch_shift(audio, sr=sample_rate, n_steps=pitch_change)
```

```
def random_shift(audio, sample_rate):
    shift = np.random.randint(low=-600, high=600)
    return np.roll(audio, shift)

def generate_augmented_data(dataset, labels, augmentation_func=None, num_augmentations=1, factor=1):
    augmented_data = []
    augmented_labels = []

    for _ in range(factor):
      for i, (features, audio, sample_rate) in enumerate(dataset):
          augmented_data.append(features)

          if augmentation_func:
              for _ in range(num_augmentations):
                  audio_changed = augmentation_func(audio, sample_rate)

                  features_changed = extract_features(audio_changed, sample_rate)
                  augmented_data.append(features_changed)
                  augmented_labels.append(labels[i])

          augmented_labels.append(labels[i])

    return np.array(augmented_data), np.array(augmented_labels)

def get_model(X_train):
  model = Sequential()
  model.add(Dense(256, activation='relu', input_shape=(X_train.shape[1],)))
  model.add(Dropout(0.5))
  model.add(Dense(256, activation='relu'))
  model.add(Dropout(0.5))
  model.add(Dense(len(labels), activation='softmax'))

  model.compile(loss="sparse_categorical_crossentropy",
                optimizer="adam",
                metrics=["accuracy"])

  return model

def plot_confusion_matrix_audio(y_true, y_pred, classes, title=None, cmap=plt.cm.Blues):
    cm = confusion_matrix(y_true, y_pred)

    plt.figure(figsize=(8, 6))
    sns.heatmap(cm,cmap= "Blues", linecolor = 'black', annot = True, fmt='', xticklabels = classes,
yticklabels = classes)
    plt.xlabel('Predicted label')
    plt.ylabel('True label')
    if title:
        plt.title(title)
    plt.show()

def train_with_augmentation(X_train_augmented_cnn, y_train_augmented, X_test_cnn, y_test,
augmentation name, epochs=20, model=None):
    print(f"Training with {augmentation_name} augmentation.")

    if model is None:
      model = get_model(X_train_augmented_cnn)

    history = model.fit(X train augmented cnn, y train augmented, epochs=epochs,
batch_size=32,validation_data=(X_test_cnn, y_test))
    score = model.evaluate(X_test_cnn, y_test, batch_size=32, verbose=1)
    augmented_accuracy = score[1]

    print(f"Test accuracy ({augmentation_name} dataset):", augmented_accuracy)

    return model, history

def get_predictions(model, X_test, y_test, title=None):
  y_pred = model.predict(X_test)

  y_pred_classes = np.argmax(y_pred, axis=1)

  print(classification_report(y_test, y_pred_classes, digits=5,
target_names=label_encoder.classes_))

  plot_confusion_matrix_audio(y_test, y_pred_classes, classes=label_encoder.classes_, title=title)

def plot_waveplots(X_train, augmented_fn, title):
  original_audio = X_train[random.randint(0, len(X_train))]
  augmented_audio = augmented_fn(original_audio[1], original_audio[2])

  plt.figure(figsize=(12, 4))
```

```
  plt.subplot(2, 1, 1)
  librosa.display.waveshow(original audio[1], sr=sample rate)
  plt.title('Original Audio', fontweight='bold', color='red')

  plt.subplot(2, 1, 2)
  librosa.display.waveshow(augmented_audio, sr=sample_rate)
  plt.title(f'Augmented Audio ({title})', fontweight='bold', color='red')

  plt.tight_layout()
  plt.show()

# Load dataset
data_path = '/content/Animals/'
subfolders = os.listdir(data_path)
labels = []
dataset = []

for subfolder in subfolders:
    for file_name in os.listdir(data_path + subfolder):
        if file_name.endswith('.wav'):
            file_path = os.path.join(data_path, subfolder, file_name)
            audio, sample_rate = librosa.load(file_path)
            features = extract_features(audio, sample_rate)
            dataset.append((features, audio, sample_rate))
            labels.append(subfolder)

len(labels)

counts = {i: labels.count(i) for i in set(labels)}

sns.set_style('darkgrid')
plt.pie(counts.values(), labels=counts.keys(), autopct='%.1f%%')
plt.axis('equal')
plt.show()

label_encoder = LabelEncoder()
label_encoded = label_encoder.fit_transform(labels)

X_train, X_test, y_train, y_test = train_test_split(
    np.array(dataset, dtype=object), label_encoded,  test_size=0.2, random_state=42
)

X_train_features = np.array([x[0] for x in X_train])
X_test_features = np.array([x[0] for x in X_test])

X train cnn = X train features[:, :, np.newaxis]
X_test_cnn = X_test_features[:, :, np.newaxis]

model_cnn_1_orig = get_model(X_train_cnn)

history_audio_original = model_cnn_1_orig.fit(X_train_cnn, y_train, epochs=30, batch_size=32,
validation data=(X test cnn, y test))

score = model_cnn_1_orig.evaluate(X_test_cnn, y_test, batch_size=32, verbose=1)

original_accuracy = score[1]
print("\n\nTest accuracy (original dataset):", original_accuracy)

get_predictions(model_cnn_1_orig, X_test_cnn, y_test, title='Confusion matrix for original dataset')

augmentations = [change_speed, change_pitch, random_shift]
augmentation_names = ['Speed Change', 'Pitch Change', 'Random Shift']

X_train_augmented_speed, y_train_augmented_speed = generate_augmented_data(X_train, y_train,
augmentation_func=augmentations[0], num_augmentations=2, factor=1)
X_train_augmented_speed_cnn = X_train_augmented_speed[:, :, np.newaxis]

plot_waveplots(X_train, change_speed, 'Speed Change')

model_cnn_1_speed, history_cnn_1_speed = train_with_augmentation(X_train_augmented_speed_cnn,
y_train_augmented_speed, X_test_cnn, y_test, augmentation_names[0], 30)

get_predictions(model_cnn_1_speed, X_test_cnn, y_test, title='Confusion matrix for changed speed
dataset')

plot_results(history_audio_original, history_cnn_1_speed, 'Speed change')

X_train_augmented_pitch, y_train_augmented_pitch = generate_augmented_data(X_train, y_train,
augmentation_func=augmentations[1], num_augmentations=2, factor=1)
X_train_augmented_pitch_cnn = X_train_augmented_pitch[:, :, np.newaxis]
```

```
plot_waveplots(X_train, change_pitch, 'Pitch change')

model cnn 1 pitch, history cnn 1 pitch = train with augmentation(X train augmented pitch,
y_train_augmented_pitch, X_test_cnn, y_test, augmentation_names[1], 30)

get_predictions(model_cnn_1_pitch, X_test_cnn, y_test, title='Confusion matrix for changed pitch
dataset')

plot_results(history_audio_original, history_cnn_1_pitch, 'Pitch change')

X_train_augmented_shift, y_train_augmented_shift = generate_augmented_data(X_train, y_train,
augmentation_func=augmentations[2], num_augmentations=2)
X_train_augmented_speed_cnn = X_train_augmented_shift[:, :, np.newaxis]

plot_waveplots(X_train, random_shift, 'Random shift')

model_cnn_1_shift, history_cnn_1_shift = train_with_augmentation(X_train_augmented_shift,
y_train_augmented_shift, X_test_cnn, y_test, augmentation_names[2], 30)

get_predictions(model_cnn_1_shift, X_test_cnn, y_test, title='Confusion matrix for random shift
dataset')

plot_results(history_audio_original, history_cnn_1_shift, 'Shift change')
```