ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Кваліфікаційна наукова
праця на правах рукопису

КРИВОХАТА АНАСТАСІЯ ГРИГОРІВНА

УДК 519.876.5:004.94(043.3)

ДИСЕРТАЦІЯ

НЕЙРОМЕРЕЖЕВІ МАТЕМАТИЧНІ МОДЕЛІ ЗВУКОВИХ СИГНАЛІВ У ЗАДАЧАХ РОЗПІЗНАВАННЯ

01.05.02 – Математичне моделювання та обчислювальні методи
01 − фізико-математичні науки

Подається на здобуття наукового ступеня
кандидата фізико-математичних наук

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
_____А. Г. Кривохата

Науковий керівник
Чопоров Сергій Вікторович
доктор технічних наук, доцент

Запоріжжя – 2020

АНОТАЦІЯ

Кривохата А. Г. Нейромережеві математичні моделі звукових сигналів у задачах розпізнавання. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня кандидата фізико-математичних наук за спеціальністю 01.05.02 «Математичне моделювання та обчислювальні методи» (01 – фізико-математичні науки). – Запорізький національний університет Міністерства освіти і науки України, Запоріжжя, 2020.

Дисертаційну роботу присвячено розв'язанню задач класифікації звукових сигналів засобами гібридних нейронних мереж із шарами згортки та автокодувальників з оптимізацією їх структури генетичними алгоритмами.

У першому розділі надано аналітичний огляд публікацій із застосування нейронних мереж у задачах розпізнавання звукових даних; наведено основні поняття машинного навчання; розглянуто підходи із застосування згорткових нейронних мереж, автокодувальників та їх ансамблів у задачах класифікації. Розглянуто публікації із застосування генетичних алгоритмів для структурної оптимізації нейронних мереж. Переважну більшість розглянутих публікацій орієнтовано на оптимізацію одного з аспектів систем машинного слуху, наприклад, набору ознак або структури та типів класифікаторів. Значно менше розглянуто комплексний підхід, при якому використання ансамблевого навчання із застосуванням класифікаторів на основі ознак та глибинних нейронних мереж поєднано з оптимізацією за допомогою генетичних алгоритмів. З аналізу публікацій зроблено висновок, про відсутність інструментальної системи класифікації звукових даних, яка б поєднувала різні методи та етапи попередньої обробки, класифікації та постпроцесорного представлення результатів роботи. Зроблено висновок, що нейромережеві моделі є одним із найбільш адаптивних методів розпізнавання, однак використання цих методів на практиці вимагає налаштування великої

кількості параметрів нейронних мереж та їх ансамблів. Одним із підходів до автоматичного вибору структури нейромереж та налаштування їхніх гіперпараметрів є генетичні алгоритми.

У другому розділі роботи виконано формальну постановку задачі класифікації звукових сигналів та сформульовано нейромережеву математичну модель класифікації звуків, яку реалізовано в наступних розділах в інструментальній системі класифікації звукових сигналів. Запропоновано схему системи класифікації звуків, яка складається з таких етапів: попередня обробка звукових даних, обчислення інформативних ознак, навчання нейронної мережі, застосування підходу Snapshot навчання ансамблю нейронних мереж, отримання вектора з прогнозом класів даних. До попередньої обробки даних віднесено такі операції: фільтрація даних; застосування перетворень довжини та тембру звукового сигналу; розширення наборів даних шляхом генерування нових звуків за допомогою нейронної мережі автокодувальник. Для найбільш раціонального представлення акустичного сигналу використано класичні методи цифрової обробки сигналів: перетворення Фур'є, мел-частотні кепстральні коефіцієнти (англ. Mel-frequency cepstral coefficients, MFCC), частотні коефіцієнти перетворення гамматон (англ. Gammatone frequency cepstral coefficients, GFCC), коефіцієнти константного Q-перетворення (англ. Constant-Q transform, CQT), хромаграми (англ. chromagram) тощо. Ці ознаки сигналу є вхідними даними для нейронних мереж. На етапі навчання мережі застосовано автокодувальник, метою якого є отримання в прихованому шарі деякий абстрактний простір ознак, уникаючи в цьому разі простого копіювання вхідного сигналу у вихідний. Особливістю згорткових нейронних мереж є декілька типів шарів (згортки, субдискретизації, повнозв'язні шари), які дозволяють виконувати безпосередньо класифікацію даних і моделювати інформативні для розв'язання задачі ознаки. Цей тип мереж є базовою моделлю в ансамблі класифікаторів розробленої інструментальної системи. У шарах згортки виконано перетворення вхідного шару та отримано мапу ознак, яка містить певні локальні ознаки вхідних даних.

Для тестування моделі класифікації в роботі використано такі колекції звукових даних: ESC, UrbanSound8k. Вказані набори обрано з двох причин. По-перше, їх широко використовують інші автори, отже можливе порівняння з іншими моделями. По-друге, дані відрізняються за кількістю записів на один клас, довжиною записів тощо, тобто, система класифікації, яка зможе демонструвати задовільну точність на цих наборах, є досить гнучкою з погляду різноманітності вхідних даних.

Наведено математичні моделі згорткових нейронних мереж та автокодувальника та особливості використання у задачах класифікації аудіоданих.

Запропоновано використання гібридних моделей нейронних мереж, які передбачають довільне поєднання декількох архітектур або типів мереж з метою підвищення точності прогнозування. Таке поєднання передбачає навчання гібридної топології як цілої системи.

Формалізовано математичну модель гібридної нейромережі класифікації. Сформульовано і доведено теорему про збіжність гібридної нейронної мережі з шарами автокодувальника та згортки.

У третьому розділі розроблено нейромережеві моделі класифікації звукових даних на основі ознак, які розглянуто у другому розділі. Виконано дослідження впливу гіперпараметрів класифікаторів на точність прогнозування моделей. Серед гіперпараметрів, що підлягають оптимізації: кількість пакетів згортки-субдискретизації, наявність та кількість додаткових шарів у пакетах, кількість фільтрів, кількість повнозв’язних шарів та нейронів у них, вид функцій активації, метод початкової ініціалізації шарів, вид оптимізатора тощо.

Проведено обчислювальні експерименти з базовими архітектурами згорткових нейромереж на відкритому наборі звукових даних ESC. Досліджено декілька варіантів топології з різною кількістю пакетів шарів згортки-субдискретизації та фільтрів. Проілюстровано точність прогнозування нейромережевих моделей за класами. Використано такі функції активації: для

всіх внутрішніх шарів зазначених моделей – Relu, а SoftMax – для останнього шару Dense. Для підвищення точності класифікації використано додаткові шари «уваги». Наведено архітектуру базової гібридної згорткової нейронної моделі класифікації звукових наборів, яка використовується у наступному розділі для ініціалізації початкової популяції генетичного алгоритму. Для оптимізації множини ознак у гібридній нейромережі використовуються шари автокодувальника, яким на вхід подаються різні типи спектрограм а у прихованому шарі генерується абстрактне представлення отриманих ознак. Отже, автокодувальник генерує абстрактні ознаки, які мають меншу розмірність ніж вхідні дані, що призводить до зменшення розмірності задачі класифікації, яка виконується згортковою нейронною мережею.

Використання ансамблевого навчання призводить до поліпшення якості класифікації, що свідчить про можливість використання наведених архітектур нейронних мереж у четвертому розділі як початкових топологій для генетичних алгоритмів. Ефективність класифікаторів підвищено за рахунок використання методу ансамблювання Snapshot, особливістю якого є отримання декількох класифікаторів під час навчання однієї мережі, що дало змогу прискорити час тренування ансамблю. Наведено результати обчислювальних експериментів з різною кількістю шарів та фільтрів. Базові архітектури нейронних мереж, які отримано після проведення обчислювальних експериментів, використано в четвертому розділі для ініціалізації початкової популяції генетичної оптимізації.

У четвертому розділі дисертаційної роботи розроблено схему кодування хромосоми та генетичні оператори для оптимізації гібридних нейромережевих моделей класифікації звукових даних з використанням генетичних алгоритмів. Особливістю запропонованого підходу є те, що оптимізуються не тільки топології класифікаторів, а й їх ансамблі. Розроблено інструментальну систему класифікації звукових даних на основі згорткових нейронних мереж та автокодувальників з оптимізацією їх параметрів за допомогою генетичних

алгоритмів. Для тестування запропонованих підходів використано набір звукових даних UrbanSound8k.

Для оптимізації гіперпараметрів моделей та їх ансамблів реалізовано генетичний алгоритм, у якому використано числові хромосоми. Розроблено словникову структуру даних для збереження області значень елементів хромосом, у яку покладено попередню експертну оцінку гіперпараметрів мереж. Оператор ініціалізації визначає початкову популяцію шляхом випадкової генерації необхідної кількості хромосом. В основі оператора відбору лежить метод ранжування, який полягає в оцінці цільової функції з подальшим сортуванням за зменшенням значень функції втрат. Для оператора кросовера передбачено такі методи: плоске схрещування, арифметичне та геометричне схрещування, жадібний метод. Для оператора мутації використано метод простої мутації, при якому змінюється обрана позиція хромосоми на довільне значення з можливих у словнику. Для формування нового покоління застосовано декілька підходів: формування нового покоління тільки з нащадків, використання принципу елітизму, витіснення в новому поколінні ідентичних за структурою хромосом.

Описані генетичні оператори та підходи до побудови генетичних алгоритмів реалізовано в інструментальній системі класифікації звуку. Розроблена система містить програмні засоби для підтримки всіх етапів класифікації звукових даних: попередньої обробки, вилучення ознак, підготовки даних для класифікації, класифікації, оцінки якості класифікатора.

Програмну реалізацію наведеної інструментальної системи написано мовою програмування Python. Використано такі бібліотеки Python: Keras, DEAP, Librosa, Spafe, Pandas. Розроблене програмне забезпечення є універсальним для розпізнавання звуків різного походження. Можуть використовуватись набори даних із не збалансованою за класами кількістю записів та різною довжиною звуку.

Основним результатом роботи є удосконалення моделей та методів підвищення ефективності розпізнавання аудіосигналів засобами нейронних

мереж за рахунок автоматичного налаштування параметрів нейромережевих моделей та їх ансамблів за допомогою генетичних алгоритмів. Отримано такі результати:

– у роботі виконано аналіз сучасного стану задачі розпізнавання аудіосигналів;

– уперше розроблено гібридні нейромережеві моделі класифікації звукових даних із використанням генетичних алгоритмів для оптимізації параметрів моделей та їх ансамблів, що дало змогу підвищити точність розпізнавання класів звуків до 96%;

– набув подальшого розвитку генетичний алгоритм структурної оптимізації в частині поліпшення архітектури ансамблів нейронних мереж, що дало змогу підвищити точність прогнозування;

– удосконалено нейромережеві моделі класифікації даних у частині оптимізації множини ознак звукових даних, що дозволило мінімізувати розмірність довільної задачі класифікації;

– уперше запропоновано інструментальну систему класифікації звукових даних на основі згорткових нейронних мереж та автокодувальників з оптимізацією їхніх параметрів за допомогою генетичних алгоритмів;

– достовірність ґрунтується на обчислювальних експериментах з використанням відкритих даних. Усі розроблені математичні моделі реалізовано в інструментальній системі класифікації звукових даних.

Отримані результати дослідження впроваджено в навчальний процес при вивченні дисциплін «Емпіричні методи програмної інженерії», «Засоби машинного навчання», «Нейронні мережі» та виконанні кваліфікаційних робіт студентами спеціальності 121 «Інженерія програмного забезпечення» Запорізького національного університету.

Ключові слова: звуковий сигнал, класифікація, гіперпараметри, згорткова нейронна мережа, автокодувальник, генетичний алгоритм.

ABSTRACT

Kryvokhata A. G. Neural networks mathematical models for the sound signals recognition problems. – Manuscript.

Thesis for the degree of a candidate of physical and mathematical sciences in specialty 01.05.02 «Mathematical Modeling and Computational methods» (01 – Physical and Mathematical Sciences). – Zaporizhzhia National University, Ministry of Education and Science of Ukraine, Zaporizhzhia, 2020.

The thesis is devoted to solving sound classification problems using convolutional neural networks and autoencoders with genetic algorithms for the structure optimization.

The first chapter provides an analytical review of publications on the neural networks applications in the sound data recognition problems; the basic concepts of machine learning are given; approaches to convolutional neural networks using, autoencoders and their ensembles in classification problems are considered. Publications on the application of genetic algorithms for the structural optimization of neural networks are presented. The vast majority of the publications are focused on the optimization of one aspect of machine hearing systems, such as a set of features or structure and types of classifiers. An approaches that use ensemble training for classifiers based on features and deep neural networks and applications of the genetic algorithms optimization were considered much less.

From the publications analysis, it was concluded that there is no instrumental system for audio data classification which would combine different methods and stages of pre-processing, classification and post-processing. It is concluded that neural network models are one of the most adaptive methods of recognition but using these methods in practice requires the large number of neural networks and their ensembles parameters adjustment. Genetic algorithm is widely used for automatic selection of the neural networks structure and their hyperparameters adjustment.

In the second chapter, mathematical models of the sound classification problem and neural networks of different types are formulated. These mathematical models are implemented in the next sections as the instrumental sound signal classification system. The scheme of classification system is suggested. It consists of the following stages: the sound data pre-processing, features evaluation, neural network training, application of the Snapshot method for neural network ensemble training, obtaining vector with data class forecast, and accuracy evaluation. The following operations are included in the preliminary data processing: data filtering; implementation of the sound length and timbre transformations; data augmentation by generating new sounds using the autoencoder neural network.

Sound features are used for signal representation. There are following features: the Fourier transform, Mel-frequency cepstral coefficients (MFCC), Gammatone frequency cepstral coefficients (GFCC), constant Q-transform coefficients (CQT), chromagrams, etc. These signal features are an input data for neural networks. Autoencoders are used for data augmentation and feature map reduction. Convolutional neural networks are used to classify sound data. Local data features are calculated in the convolutional and pooling layers. These network types are used as the basic model in the further chapters. The Snapshot ensemble method is used for both neural networks in the next chapter. The attention layer is described. It is used for prediction accuracy increasing.

The following sound data collections were used to test the classification models: ESC, UrbanSound8k. These sets were selected because, firstly, they are widely used by other authors, so it is possible to compare with other models. Secondly, the data differ in the number of records per class, the length of the records, etc. The classification system, which can demonstrate the sufficient accuracy on these sets, is quite flexible in terms of the variety of input data.

The use of hybrid models of neural networks is suggested in order to provide an arbitrary combination of several networks architectures. This combination is used to increase the prediction accuracy. It assumes learning a hybrid topology as a whole system.

The mathematical model of the hybrid neural network of classification is formalized. The theorem on the convergence of a hybrid neural network with the layers of the autoencoder and convolution is formulated and proved.

In the third chapter the neural network models for sound data classification are developed. It is assumed that the features and network architectures from previous chapter are used. A study of the influence of classifier hyperparameters on the accuracy of model prediction is performed. The hyperparameters are following: the number of convolution-pooling packets, the number of additional layers in the packets, the number of filters, the number of fully connected layers and neurons in them, the type of activation functions, the layer initialization method, the optimizer type, etc.

Computational experiments with the basic architectures of convolutional neural networks on an open ESC audio dataset are performed. The experiments consider different neural networks topologies and hyperparameters. The accuracy of forecasting neural network models with confusion matrices is illustrated. The following activation functions are used: Relu for all hidden layers, and SoftMax for the output Dense layer. Additional layers of «attention» were used to increase the classification accuracy. The basic hybrid convolutional neural model for sound classification is presented. This topology is used in the next section to initialize the initial population of the genetic algorithm. There are different types of spectrograms that are fed to the autoencoder layers. The abstract representations of the obtained features is generated in the hidden layer. These representations are used to optimize the features set in the hybrid neural network. Thus, the autoencoder generates abstract features that have a smaller dimension than the input data, which reduces the dimensionality of the classification task performed by the convolutional neural network.

The use of ensemble learning improves the classification accuracy which indicates the possibility of using the above neural network architectures in the fourth section as the initial topologies for genetic algorithms. The efficiency of classifiers is increased due to the use of the Snapshot ensemble method. The time of

ensemble learning is equal to the training time of one network which is significant for deep neural networks. The results of computational experiments with different number of layers and filters are presented. The basic architectures of neural networks, which were obtained after computational experiments, were used in the fourth chapter to initialize the initial population of genetic optimization. The methods for data augmentation are described. They are used to treat imbalanced learn problem. There are over- and under-sampling methods.

The chromosome coding scheme and genetic operators are developed in the fourth chapter of the dissertation. The genetic algorithms requires it for hybrid neural network sound classification models optimization. The ensembles are optimized also in this approach which is vital for further applications. An instrumental system of sound data classification based on convolutional neural networks and autoencoders with optimization of their parameters using genetic algorithms has been developed. The UrbanSound8k audio data set was used to test the proposed approaches.

The genetic algorithm was implemented using numerical chromosomes in order to optimize the hyperparameters of the models and their ensembles. A dictionary data structure has been developed to preserve the values range of chromosome elements. This data structure requires expert assessment for hyperparameters possible values range determining. The initialization operator determines the initial population by randomly generating the required number of chromosomes. The selection operator is based on the ranking method, which consists in estimating the objective function with subsequent sorting by decreasing the values of the loss function. The following methods are provided for the crossover operator: flat crossing, arithmetic and geometric crossing, greedy method. For the mutation operator, the method of simple mutation is used, in which the selected position of the chromosome is changed to an arbitrary value from those possible in the dictionary. Several approaches have been used to form a new generation: the formation of a new generation only from descendants, the use of the elitism principle, the displacement of identical chromosomes in the new generation.

These genetic operators and approaches to the genetic algorithms are implemented in the instrumental sound classification system. The developed system contains software to support all stages of the audio data classification: pre-processing, features extraction, data extraction for classification, classification, classifier accuracy assessment. The pre-processing stage implies that a raw signal is the subject of segmentation into shorter signal chunks by some windowing process. A common approach is to convert the original acoustic signal to the frames of a certain length.

The aim of the feature extraction stage is to receive a compact representation of the acoustic characteristics of a signal. This stage exploits special coefficients such as Zero-crossing rate, Spectrum shape, Short-Time Fourier Transform, Mel-frequency cepstral coefficients, gammatone frequency cepstral coefficients etc

The software for instrumental system is developed using Python programming language. There are used the following Python libraries: Keras, DEAP, Librosa, Spafe, Pandas. The developed software is universal for different sources sounds classification. Datasets can be used with an unbalanced number of recordings and different audio lengths.

The instrumental system could be used as high level framework for rapid developing of the embedded sound classification modules. Computational experiments were provide using both graphics and tensor processing units. Machine hearing systems are quite useful in such areas as audio surveillance, automatic medicine auscultation, acoustic scene detection in urban environments etc. The main requirements for such systems are real time classification with high accuracy and ability to generalization on unseen data. The classification system described at the fourth chapter of the thesis meets these requirements.

The main result of the thesis is the improvement of models and methods to increase the efficiency of recognition of audio signals by means of neural networks by automatically adjusting the parameters of neural network models and their ensembles using genetic algorithms. The following results were obtained:

– the analysis of the current state of the problem of audio signal recognition is performed in the work;

– for the first time hybrid neural network models of sound data classification were developed using genetic algorithms to optimize the parameters of models and their ensembles, which allowed to increase the accuracy of recognition of sound classes to 96%;

– the genetic algorithm of structural optimization was further developed in terms of improving the architecture of neural network ensembles, which made it possible to increase the accuracy of prediction;

– improved neural network models of data classification in terms of optimizing the set of features of audio data, which minimized the dimensionality of an arbitrary classification problem;

– for the first time an instrumental system of sound data classification on the basis of convolutional neural networks and autoencoders with optimization of their parameters with the help of genetic algorithms was proposed;

– reliability is based on computational experiments using open data. All developed mathematical models are implemented in the instrumental system of sound data classification.

The obtained research results are introduced into the educational process in the study of disciplines «Empirical methods of software engineering», «Machine learning tools», «Neural networks» and qualification work of students majoring in 121 «Software Engineering» Zaporizhzhia National University.

Keywords: sound data, classification, hyperparameters, convolutional neural network, autoencoder, genetic algorithm.

СПИСОК ПУБЛІКАЦІЙ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Праці, в яких опубліковані основні наукові результати:

1. Кривохата А. Г., Кудін О. В., Давидовський М. В., Лісняк А. О. Застосування ансамблевого навчання в задачах класифікації акустичних даних. *Вісник Запорізького національного університету*. *Фізико-математичні науки*. Запоріжжя, 2018. № 1. С. 50–62.

2. Чопорова О. В., Кривохата А. Г., Оптимізація згорткових нейронних мереж та їх ансамблів. *Вісник Запорізького національного університету*. *Фізико-математичні науки*. Запоріжжя, 2019. № 1. С. 107–115.

3. Kryvokhata A., Kudin O., Gorbenko V. Design Sound Classification IoT System with Genetic Algorithm. *Visnyk of Zaporizhzhia National University. Physical and Mathematical Sciences*. Zaporizhzhia, 2019. № 2. P. 69−74.

4. Кривохата А. Г., Кудін О. В., Чопоров С. В. Нейромережеві математичні моделі у задачах обробки звукових сигналів: монографія. Херсон : Видавничий дім «Гельветика», 2020. 120 с.

5. Kryvokhata A. The sound classification system based on neural networks with attention mechanism and autoencoders. *International Journal of Mathematics and Statistics Invention (IJMSI).* 2020. Volume 8, Issue 6. P. 24−28.

6. Kryvokhata A. Evolutionary methods in environmental sound classification. *International Journal of Mathematics and Computer Research.* 2020. Volume 8, Issue 7. P. 2091−2095.

Праці, які засвідчують апробацію матеріалів дисертації:

7. Кудін О. В. Кривохата А. Г. Методи класифікації акустичних даних. *Актуальні проблеми математики та інформатики* : збірка тез та доповідей Дев'ятої Всеукраїнської, шістнадцятої регіональної наукової конференції молодих дослідників. (Запоріжжя, 26-27 квітня 2018). Запоріжжя, 2018. С. 37–38.

8. Кривохата А. Г. Оптимізація згорткової нейронної мережі при розв'язанні задачі класифікації акустичних даних. *Інновації науки XXI століття* : збірник наукових матеріалів XXXVI Міжнародної науково-практичної інтернет-конференції. (Вінниця, 18 листопада 2019). Вінниця, 2019. С. 23–27.

9. Кудін О. В., Кривохата А. Г. Генетичні алгоритми оптимізації ансамблів згорткових нейронних мереж. *Комп'ютерні науки, інформаційні технології та системи управління* : матеріали Міжнародної науково-технічної конференції здобувачів вищої освіти та молодих вчених. (Івано-Франківськ, 27-29 листопада 2019). Івано-Франківськ, 2019. С. 28.

10. Кривохата А. Г., Кудін О. В., Лісняк А. О. Методи глибинного навчання у задачах машинного слуху. *Міжнародна конференція з математичного моделювання* : матеріали XIX міжнародної конференції з математичного моделювання, присвяченої 250-річчю з дня народження Жана Батиста Фур'є. (Херсон, 17-21 вересня 2018). Херсон : ХНТУ, 2018. С. 70–71.

11. Кудін О. В., Кривохата А. Г. Застосування згорткових нейронних мереж в задачах класифікації акустичних даних. *Комп'ютерні науки, інформаційні технології та системи управління* : матеріали міжнародної науково-технічної конференції молодих вчених, аспірантів та студентів. (Івано-Франківськ, 28-30 листопада 2018). Івано-Франківськ : Прикарпатський національний, 2018. С. 145–146.

12. Kudin O., Kryvokhata A., Gorbenko V. Developing a Deep Learning Sound Classification System for a Smart Farming. *237th ECS Meeting with the 18th*

International Meeting on Chemical Sensors (IMCS 2020) : Meeting Abstracts of The Electrochemical Society. (Canada, May 2020). Canada, 2020. Vol. 2020-01, N. 26. P. 1853. DOI: 10.1149/ma2020-01261853mtgabs.

13. Кривохата А. Г. Класифікація звуків з використанням згорткових автокодувальників. *Інформаційні технології та комп'ютерне моделювання* : матеріали міжнародної науково-практичної конференції. (Івано-Франківськ, 18-22 травня 2020). Івано-Франківськ, 2020. С. 181–182.

Праці, які додатково відображають наукові результати дисертації:

14. Кривохата А. Г., Кудін О. В., Лісняк А. О. Огляд методів машинного навчання для класифікації акустичних даних. *Вісник Херсонського національного технічного університету*. Херсон, 2018. Т. 1, № 3 (66). С. 327–331.

15. Тимофєєва А. Є., Кудін О. В., Кривохата А. Г., Лісняк А. О. Автоматичне анотування зображень за допомогою нейронних мереж. *Вчені записки Таврійського національного університету імені В.І. Вернадського*. Київ, 2019. Т. 30 (69), № 2, Ч. 1. С. 214–220.

ЗМІСТ

ВСТУП

Сучасний розвиток засобів телекомунікації, соціальних мереж, поширеність вебресурсів та мобільних застосунків зумовлює накопичення великої кількості текстових та мультимедійних даних, зокрема, аудіофайлів. Прикладом звукових даних є записи музики, лекцій, доповідей, звуків різного походження тощо. Для можливості пошуку в таких колекціях, зазвичай, використовуються метадані, які описують у текстовому вигляді вміст відповідного аудіофайлу. Часто формування метаданих виконують вручну, що не завжди зручно при обробці великих обсягів даних.

Сьогодні, розробка машин з чуттєвими можливостями, такими як зір та слух, є однією з визнаних складних проблем науки й техніки. Системи, які мають можливість визначати сенс із аудіовізуальної інформації, все частіше використовуються як у наукових застосунках, так й у промисловості. Отже, безумовно, існує необхідність в ефективних підходах автоматичного розпізнавання звукових, графічних та відеоданих. Роботу присвячено розвиненню методів класифікації акустичних даних та розробці відповідних інструментальних систем.

Актуальність теми дисертаційного дослідження. Програмне забезпечення розпізнавання звуків, зображень, числових даних, тексту практично застосовується у різних галузях. Класифікацію зображень можна застосовувати для автоматичної обробки супутникових зображень, у медичній діагностиці, системах біометричної автентифікації, контролю якості виробництва тощо. Системи розпізнавання аудіоданих інтегровано в сучасну урбаністику, ідентифікацію правопорушень, аускультацію серця та легень тощо.

Усі наведені застосування автоматичної класифікації зводяться до задач розпізнавання образів, для вирішення яких є досить багато підходів, однак

найбільш універсальними методами, з погляду можливості адаптації, є нейронні мережі.

У розвиток теорії та практики розробки нейромережевих моделей розв'язання задач розпізнавання суттєвий внесок зробили такі вчені: Безсонов О. О., Бодянський Є. В., Зайченко Ю. П., Згуровський М. З., Нестеренко Б. Б., Новотарський М. А., Олійник О. О., Руденко О. Г., Синєглазов В. М., Субботін С. О., Тимощук П. В., Чумаченко О. І., Ямпольський Л. С., Bengio Y., Goodfellow I., Hinton G., LeCun Y., Ng A., Rosenblatt F., Schmidhuber J. та ін.

Істотний внесок у розвиток еволюційних методів та їх використання в задачах дискретної оптимізації зробили такі науковці, як-от: Безсонов О. О., Козін І. В., Олійник А. О., Пічугіна О. С., Перепелиця В. О., Субботін С. О., Скобцов Ю. А., Holland J., Fogel L., Rechenberg I., Schwefel H-P. та ін.

Попри значне поширення нейромережевих моделей, не існує загальних рекомендацій та схем їх застосування на практиці. Досить велику кількість параметрів таких моделей дослідники часто налаштовують у процесі виконання серії обчислювальних експериментів. Використання ансамблю нейронних мереж суттєво ускладнює знаходження оптимальної множини параметрів. Наприклад, процес класифікації даних потребує налаштування декількох десятків або сотень класифікаторів.

Отже, актуальною задачею є розробка методів та підходів до автоматичної оптимізації нейронних мереж та їх ансамблів із метою підвищення точності розпізнавання звуків.

Зв'язок роботи з науковими програмами, планами, темами. Роботу виконано відповідно до плану наукових досліджень математичного факультету Запорізького національного університету за держбюджетними науково-дослідними роботами «Розробка математичного забезпечення для інженерного аналізу об'єктів аерокосмічної техніки на базі хмарних технологій» (№ ДР 0117U007204, 2017-2020 рр.) та «Математичне та

програмне забезпечення автоматизованого проектування аерокосмічної техніки» (№ ДР 0118U00210, 2018-2020 рр.).

Мета і задачі дослідження. Метою роботи є розробка математичних моделей звукових сигналів у задачах розпізнавання на базі нейронних мереж.

Задачі, які необхідно розв’язати для досягнення поставленої мети:

– провести аналіз сучасного стану проблеми розпізнавання аудіосигналів;

– розробити гібридні нейромережеві моделі класифікації звукових даних із використанням генетичних алгоритмів оптимізації параметрів моделі;

– удосконалити генетичний алгоритм структурної оптимізації архітектури ансамблів нейронних мереж;

– удосконалити нейромережеві моделі в задачах оптимізації множини ознак звукових даних;

– розробити інструментальну систему класифікації звукових даних на основі згорткових мереж та автокодувальників з оптимізацією їх параметрів за допомогою генетичних алгоритмів.

Об’єкт дослідження – процес розпізнавання звукових сигналів.

Предмет дослідження – нейромережеві математичні моделі звукових сигналів.

Методи дослідження. Для моделювання звукових сигналів у роботі використовуються методи цифрової обробки сигналів; методи теорії нейронних мереж використовуються для розробки математичних моделей розпізнавання; методи теорії генетичних алгоритмів застосовуються при структурній оптимізації нейронних мереж у задачах розпізнавання.

Наукова новизна одержаних результатів:

– уперше розроблено гібридні нейромережеві моделі класифікації звукових даних із використанням генетичних алгоритмів для оптимізації параметрів моделей та їх ансамблів, що дає змогу підвищити точність розпізнавання довільних класів звуків;

– уперше запропоновано інструментальну систему класифікації звукових даних на основі згорткових нейронних мереж та автокодувальників з оптимізацією їхніх параметрів за допомогою генетичних алгоритмів;

– набув подальшого розвитку генетичний алгоритм структурної оптимізації в частині поліпшення архітектури ансамблів нейронних мереж, що дало змогу підвищити точність прогнозування;

– удосконалено нейромережеві моделі класифікації даних у частині оптимізації множини ознак звукових даних, що дало змогу мінімізувати розмірність довільної задачі класифікації аудіозаписів.

Наукові результати, які отримано в дисертаційній роботі, відповідають формулі спеціальності 01.05.02 та напрямам досліджень № 3, 4, 5 і 6 згідно з паспортом спеціальності, що свідчить про створення та вдосконалення математичних моделей класифікації звукових даних у межах інструментальної системи розпізнавання.

Практичне значення одержаних результатів. Розроблені математичні моделі надають можливість розв’язання задачі класифікації звукових даних у таких галузях: медична діагностика, системи машинного слуху, спостереження за об’єктами тваринництва, біометричної автентифікації. Підвищення точності розпізнавання досягнуто за рахунок побудови гібридних нейромережевих моделей з оптимізацією їхніх параметрів за допомогою генетичних алгоритмів. Запропоновані методи та моделі реалізовано в інструментальній системі класифікації звукових даних.

Результати, що отримані в дисертаційному дослідженні, упроваджені в навчальному процесі при вивченні дисциплін «Емпіричні методи програмної інженерії», «Засоби машинного навчання», «Нейронні мережі» та виконанні кваліфікаційних робіт студентами спеціальності «Інженерія програмного забезпечення».

Особистий внесок. Усі результати дисертаційної роботи, що виносяться на захист, отримано автором особисто. Чотири наукові праці опубліковано одноосібно [16, 17, 103, 104]. У роботах, написаних у співавторстві,

здобувачеві належать: розробка гібридних нейромережевих моделей та їх реалізація в інструментальній системі класифікації звукових даних [21]; розробка математичних моделей нейромережевих класифікаторів та їх ансамблів [20, 105]; розробка метода оптимізації архітектури згорткової нейронної мережі на базі генетичного алгоритму [55]; дослідження методів побудови ансамблів нейронних мереж для класифікації акустичних даних [18]; дослідження впливу гіперпараметрів нейронної мережі на точність класифікації [47]; програмна реалізація бібліотеки класифікації акустичних даних [22]; реалізація генетичного алгоритму оптимізації згорткових нейронних мереж та їх ансамблів у задачах медичної діагностики [23]; запропоновано підхід до побудови ансамблів нейронних мереж [19]; реалізація алгоритму побудови ансамблю нейронних мереж [24]; архітектура вбудованої системи розпізнавання звуку для сільського господарства [106].

Апробація результатів дисертації. Основні результати, що отримані у дисертаційному дослідженні доповідались на конференціях:

Дев’ята Всеукраїнська, шістнадцята регіональна наукова конференція молодих дослідників «Актуальні проблеми математики та інформатики» м. Запоріжжя, 26 – 27 квітня, 2018 р.

XIX міжнародна конференція з математичного моделювання, присвячена 250-річчю з дня народження Жана Батиста Фур’є, м. Херсон, 17 – 21 вересня, 2018 р.

Міжнародна науково-технічна конференція молодих вчених, аспірантів та студентів «Комп’ютерні науки, інформаційні технології та системи управління», м. Івано-Франківськ, 28 – 30 листопада, 2018 р.

XXXVI Міжнародна науково-практична інтернет-конференція «Інновації кауки XXI століття», м. Вінниця, 18 листопада, 2019 р.

Міжнародна науково-технічна конференція здобувачів вищої освіти та молодих вчених «Комп’ютерні науки, інформаційні технології та системи управління», м. Івано-Франківськ, 27 – 29 листопада, 2019 р.

237th ECS Meeting with the 18th International Meeting on Chemical Sensors (IMCS 2020), Montreal, Canada, 10 – 14 May 2020.

Міжнародна науково-практична конференція «Інформаційні технології та комп'ютерне моделювання», м. Івано-Франківськ, 18 – 22 травня, 2020 р.

Дисертація в цілому розглядалася на науковому міжвузівському семінарі Запорізького національного університету «Актуальні проблеми прикладної математики та механіки» під керівництвом д.т.н., професора В. З. Грищака в Запорізькому національному університеті.

Публікації. За результатами досліджень опубліковано 15 робіт: серед них – 1 монографія [21]; 5 статей у наукових періодичних виданнях (з них 2 статті – у іноземних спеціалізованих виданнях [103, 104]; 3 статті – у журналах включених до Переліку наукових фахових видань України з фізико-математичних наук [20, 55, 105]); 2 статті – у журналах включених до Переліку наукових фахових видань України з технічних наук, які додатково висвітлюють наукові результати [18, 47]; 7 робіт – у збірниках матеріалів конференцій та тез доповідей [16, 17, 19, 22, 23, 24, 106] (з них 1 – у збірнику матеріалів закордонної конференції [106]).

Структура і обсяг дисертації. Дисертаційна робота складається з анотації, вступу, чотирьох розділів, висновків, списку використаних джерел, додатків. Загальний обсяг роботи складає 160 сторінок. Робота містить 39 рисунків, 8 таблиць, список використаних джерел із 176 найменувань (на 20 сторінках) та 3 додатки.

РОЗДІЛ 1
АНАЛІЗ СТАНУ ПРОБЛЕМИ РОЗРОБКИ НЕЙРОМЕРЕЖЕВИХ МОДЕЛЕЙ РОЗПІЗНАВАННЯ ЗВУКОВИХ ДАНИХ

Розробка математичних моделей та автоматизованих систем розпізнавання звукових даних є актуальною задачею у різних галузях, таких як медична діагностика, урбаністика, сільське господарство. Наприклад, системи класифікації звукових даних у застосуванні до обробки обстежень легень та серця дозволяють автоматизувати виявлення захворювань, потенційні правопорушення можуть бути класифіковані при обробці міських звуків. Ці та інші задачі розв'язують при розробці різноманітних систем машинного слуху [119]. Однією із задач цього напряму є розробка ефективних методів класифікації аудіоданих різного походження, наприклад, мови, музики, природних звуків тощо. У цьому разі, найбільш досліджуваними є саме задачі аналізу музики та мови [59]. Іншим напрямом, що досить часто розглядають автори, є виявлення звукових подій. Цю задачу спрямовано на обробку неперервного акустичного сигналу та перетворення його у символічний опис змісту звукового запису [101].

Отже, більшість задач машинного слуху зводять до виділення з сигналу акустичних подій, їх перетворення та класифікації. Водночас використовують методи обробки цифрових сигналів, математичної статистики, теорії нейронних мереж та дискретної оптимізації.

Детерміновані методи обробки звуку передбачають використання вейвлет та Фур'є перетворень, звукових фільтрів тощо [5, 59, 109]. Ці алгоритми використовують для виділення ознак звуку та попередньої його обробки з метою формування такого представлення, яке буде найбільш оптимальним для подальшої класифікації. Такі підходи до попередньої

обробки звуку та виділення ознак добре себе зарекомендували та використовуються більшістю авторів.

Як свідчить аналіз публікацій останніх років, огляд яких проведено нижче у даному розділі, для побудови систем класифікації звуку автори використовують різні набори ознак та класифікаторів, а однією з вимог до таких моделей є здатність до узагальнення або генералізації, тобто задовільної точності роботи системи на нових даних, які не були використані протягом розробки алгоритмів обробки звуку.

Такої адаптивності можна досягти за допомогою нейронних мереж, які використовують на етапах попередньої обробки даних та безпосередньо класифікації. Оскільки під час розв'язання практичних задач можна застосовувати різні нейромережеві моделі, актуальною науковою задачею є розробка методів створення гібридних систем, які поєднують у собі декілька типів нейронних мереж. Ще одним методом підвищення точності прогнозування є поєднання нейромережевих моделей у ансамблі, які поєднують декілька мереж спільної або різної архітектури з наступним формуванням спільного прогнозу.

Важливим питанням також є розробка методів пошуку оптимальної архітектури нейронних мереж для розв'язання конкретних задач класифікації. Такі параметри, як кількість та тип шарів, кількість нейронів, види функцій активації тощо, можуть суттєво впливати на точність моделі. Для розв'язання цієї задачі зазвичай використовують еволюційні методи, за допомогою яких виконується направлений пошук кращої архітектури.

У наступному підрозділі наведено деякі базові поняття машинного навчання, що зазвичай використовуються у публікаціях. Основні статті із застосування нейронних мереж та їх ансамблів у задачах класифікації звуку розглянуто у другому підрозділі. Особливості використання генетичних алгоритмів для пошуку оптимальних структур нейронних мереж, які відображено в останніх публікаціях, розглянуто у третьому підрозділі.

1.1 Основні поняття машинного навчання

Під машинним навчанням розуміють набір алгоритмів і методів для статистичної апроксимації функціональних залежностей [86], використовуючи при цьому обмежений набір даних – навчальну вибірку. При так званому «навчанні з учителем», цей набір є впорядкованими парами числових характеристик (ознак) деяких об’єктів та відповідних їм міток [45]. Найчастіше, при використанні дискретних цілочисельних міток розв’язують задачу класифікації, а у разі дійсних значень – задачу регресії.

Кожну пару з навчальної вибірки називають спостереженням або прецедентом [26]. У практиці машинного навчання множину міток об’єктів часто називають множиною відповідей [45].

Значення множини відповідей визначають такі варіанти задачі класифікації [28, 45]:

– бінарна класифікація (наприклад, при класифікації звуків на природні та штучного походження);

– багатокласова класифікація (прикладом може бути класифікація звуків різних тварин);

– багатокласова класифікація з класами, що перетинаються (визначення звукової сцени, коли один аудіозапис може містити звуки з різних джерел).

Окрім навчання з учителем, розглянутого вище, виділяють ще часткове навчання, при якому значення цільової функції відомі не для всіх об’єктів навчальної вибірки; навчання без учителя, коли значення цільової функції відсутні.

Здебільшого задачі машинного навчання зводять до пошуку деякої нелінійної функції з набором параметрів, що оптимізують під час навчання алгоритму.

Відповідність шуканої функції постановці задачі та даним навчальної вибірки визначено функціоналом похибки, а процес навчання зводять до

мінімізації цього функціоналу шляхом підбору оптимального набору параметрів алгоритму навчання.

Ефективність навчання переважно залежить від методу (нейронні мережі, дерева рішень, висновок Баєса та ін.), функціоналу похибки та даних навчальної вибірки. У обробці аудіовізуальних даних особливо важливим етапом є вибір інформативних ознак звуку. Для розв'язання цієї задачі застосовують класичні статистичні алгоритми, наприклад, метод головних компонент, проте також використовуються мультиагентний підхід та нейронні мережі.

1.2 Огляд робіт із застосування нейронних мереж у задачах класифікації звуку

Здебільшого процес аналізу звуку включає декілька етапів. Спочатку виконують попередню обробку неперервного акустичного сигналу з метою представлення його у дискретному цифровому вигляді. Водночас, зазвичай, використовують ряд стандартних підходів [71, 129], які полягають у дискретизації та квантуванні вихідного акустичного сигналу. Далі виділяють ознаки звуку, суттєві для розв'язання поставленої задачі класифікації або кластеризації. Серед найбільш широко вживаних ознак використовують коефіцієнти перетворення Фур'є та автокореляції, мел-кепстральні коефіцієнти, хромаграми тощо [109]. Після цього отримані ознаки використовують як вхідні значення математичної моделі класифікації (наприклад, алгоритму класифікації, кластеризації або нейронної мережі). На заключному етапі виконують верифікацію отриманих результатів та впроваджують розроблену систему акустичного аналізу.

Наразі тематиці розробки систем машинного слуху присвячено велику кількість публікацій та продовжують з'являтися нові. Серед оглядових робіт,

можна виділити декілька, які є найбільш загальними. Так, в оглядових статтях [59, 68, 150] наведено опис компонент системи автоматичної класифікації звуків, яка містить модулі попередньої обробки, екстракції ознак, алгоритм навчання та модуль визначення якості моделі класифікації.

У [59] детально розглянуто підходи до виділення ознак звуку. Наведено критерії, за якими можна класифікувати мову, музику та природні звуки. Виділено методи, що засновані на фізичних властивостях сигналів та особливостях людського сприйняття звуків. Частіше за все використовують методи виділення ознак, які представляють акустичний сигнал у таких областях: часовій, частотній, кепстральній та вейвлет.

Огляди [68, 150] містять аналіз загальних підходів і публікацій з автоматичної класифікації музичних записів за жанрами. Запропоновано множину найбільш інформативних міток, які можна використовувати як класи при навчанні класифікаторів. Розглянуто найбільш вживані джерела розмічених акустичних даних, що застосовують у системах навчання з учителем. Переважно це відкриті музичні бази даних у мережі Інтернет, розмічені користувачами записи. Наприклад у соціальних мережах та дані, які згенеровано спеціально для розв’язання задач машинного слуху. У роботі [150] додатково розглянуто питання оцінки ефективності систем класифікації музичних файлів за жанрами.

Попри наявність доволі докладних оглядів із класифікації звуків, більшість з них базуються на класичному підході до аналізу даних засобами машинного навчання, який має такі етапи: виділення ознак, навчання системи, верифікація та впровадження системи. Недостатню увагу приділено методам глибинного машинного навчання, які включають нейронні мережі з великим числом прихованих шарів та можливості побудови гібридних систем з використанням декількох типів нейронних мереж.

Для класифікації даних, які описано своїми ознаками, можуть використовувати як статистичні методи (класифікатор Баєса [77], дискримінантний аналіз [70, 91], EM алгоритм тощо), так і методи, які

засновано на мірах схожості та відмінності (метод *k*-середніх [71, 96, 135], метод опорних векторів [71, 91, 124], метод *k* найближчих сусідів [71, 75] тощо). Потенційні складнощі використання зазначених методів машинного навчання, такі як недостатня генералізація, залежні ознаки тощо наведено у [126].

У роботі [77] розв'язано задачу класифікації довільних аудіоданих. Запропоновано автоматичну систему, яка розподіляє вхідні аудіодані на сім класів, зокрема різні типи музики, мова одного чи декількох людей, звуки зовнішнього середовища. Виконано порівняння декількох ознак класифікації, показано, що найліпший результат досягається при використанні мел-частотних кепстральних коефіцієнтів та коефіцієнтів кодування з лінійним предиктором [129]. Процес аналізу аудіо в даній роботі має такі етапи: вилучення ознак, ідентифікація пауз, сегментація на фрагменти та класифікація фрагментів. Для класифікації використовують класифікатор Баєса.

Автоматичний аналіз акустичних даних застосовується також і в комп'ютерній лінгвістиці. У статті [91] автори виконують лінгвістичне дослідження впливу наголосу на певні частини речення в англійській мові на сприйняття змісту. Описана у роботі методологія дослідження передбачає використання акустичного класифікатора (метод опорних векторів та лінійний дискримінантний аналіз). Для визначення ознак, що мають найбільший вплив на результат класифікації, використовують метод Борута.

У роботі [96] введено поняття матриці часової залежності, яка містить дані про кількість повторень певних типів інтервалів у звукових даних. Кластеризацію аудіоданих на основі отриманої матриці часової залежності виконують за допомогою метода *k*-середніх. Проілюстровано використання розроблених алгоритмів для кластеризації аудіо та відео документів.

Порівняння підходів на базі методів опорних векторів та багатошарових перцептронів наведено у [98].

Розробка системи автоматичного вибору суттєвих для подальшої класифікації ознак звуку є досить актуальним напрямом, якому присвячено велику кількість досліджень, наприклад роботи [60-62, 79, 80, 82, 89, 97, 113, 141, 158, 164, 167, 174]. Ознаки генерують детермінованим алгоритмом, нейронною мережею, із застосуванням методів навчання без учителя або еволюційних алгоритмів.

Досить велику кількість сучасних публікацій з класифікації даних присвячено використанню нейронних мереж на різних етапах, наприклад при вилученні ознак, при класифікації отриманих з ознак представлень. Це можна пояснити здатністю таких моделей до апроксимації нелінійних функцій довільної складності. Отже, побудувавши оптимальну архітектуру нейронної мережі, можна досягти відображення будь-якого вхідного сигналу в потрібний вихідний.

Серед різноманіття нейромережевих моделей для задач класифікації зазвичай використовуються такі: рекурентні нейронні мережі, згорткові, автокодувальники та мережі прямого поширення (багатошаровий перцептрон).

Атомарним елементом нейронної мережі є, власне, штучний нейрон. Типи штучних нейронів, які входять до мережі, та способи їх поєднання у шари визначають специфічну архітектуру нейронних мереж. Більш детально моделі нейронів та їх комбінації буде розглянуто в наступних розділах.

Багатошарова архітектура нейронних мереж, яка може містити шари різних типів, представляє собою послідовність нелінійних функцій активації, що застосовуються до початкового сигналу. Результатом застосування нейронних мереж є підвищення точності класифікації у практичних застосуваннях.

Завдяки своїй архітектурі глибинні нейронні мережі при вдалому налаштуванні можуть визначати закономірності у вихідних даних та використовувати їх для розв’язання задач класифікації. Сучасні архітектури згорткових нейронних мереж можуть використовувати десятки шарів. Так,

архітектура VGG (англ. Visual Geometry Group) використовує 19 шарів «згортка-нелінійність-субдискретизація», а GoogLeNet та DenseNet (англ. Dense Convolutional Network) відповідно 22 та 250 шарів [31]. Кількість шарів сучасних ResNet (англ. Residual Convolutional Network) вже близько тисячі. Також, розробляються спеціальні архітектури для пристроїв з обмеженими ресурсами, наприклад MobileNets, SqueezeNet [31]. Застосовуються також генеративні типи мереж [160].

Реалізація глибинних нейронних мереж зазвичай потребує системи паралельних та розподілених обчислень, часто із залученням графічних процесорів (наприклад, CUDA для Nvidia), та може займати тижні [31, 84]. На даний момент, існує декілька програмних бібліотек (TensorFlow, Keras, Theano, Caffe, PyTorch), в яких реалізовано основні архітектури глибинних нейронних мереж, зокрема згорткові мережі.

Далі розглянемо роботи, в яких використовують ту чи іншу нейронну архітектуру.

Лінійні та нелінійні моделі нейронів для виділення ознак з аудіоданих наведено у роботі [64]. Досліджено вплив нелінійної функції активації нейронів на результат класифікації за шістьма класами. Як класифікатор використано три методи: класифікатор Баєса, метод опорних векторів та метод гаусових сумішей. Показано, що використання нелінійної моделі нейронів дозволяє підвищити якість класифікації на 15%.

Метод дискримінантного аналізу звукових даних розглянуто в [70]. Застосування цього методу до попередньо обробленого звукового сигналу дозволяє отримати стійкий до шуму вектор ознак. Кожен шар деформаційного дискримінантного аналізу, у свою чергу, використовує метод головних компонент та згорткові нейронні мережі [171] для генерування вектору ознак з вхідного звукового файлу та класифікації. Як приклад описано архітектуру системи, що дозволяє знаходити звукові відрізки в потоковому аудіо.

У статті [75] запропоновано підхід до автоматичної класифікації музичних записів за жанрами. Особливістю підходу є те, що вихідні музичні

файли розділено на три частини, для яких обчислено власні 15-мірні вектори ознак. Після цього кожен з векторів надходить на вхід класифікатора. Результат отримано шляхом голосування моделей. На етапі визначення ознак використано швидке перетворення Фур'є та обчислено такі ознаки, як спектральний центроїд та інші. Застосовано два типи класифікаторів: метод *k* найближчих сусідів та багатошарова нейронна мережа з одним прихованим шаром.

У роботах [57, 111] використовують 1D згорткову мережу, на вхід якої подано аудіосигнал без обробки. Чотири згорткових шари дають змогу вилучити абстрактні ознаки з амплітудно-часового представлення. Використання 1D мережі має меншу кількість параметрів навчання у порівнянні з 2D і 3D шарами.

Однак більш поширеним є двомірне представлення звукового сигналу. У цьому випадку природнім є використання 2D згорткових мереж на етапі класифікації та вилучення ознак, як це продемонстровано в роботах [58, 73, 74, 94, 100, 101, 102, 135, 142, 144, 151, 154, 163, 166, 172, 173, 175]. Відмінними у різних реалізаціях є особливості певних шарів або способів їх поєднання.

Наприклад, у [73] використано розріджену згорткову матрицю при формуванні фільтрів. Такий підхід дає змогу розширити поле, з якого відбувається зчитування характерних ознак.

У роботі [94] відмінність від класичної згорткової архітектури полягає у поєднанні кожного згорткового шару безпосередньо зі всіма наступними шарами.

Статті [101, 102] описують застосування згорткових мереж до вилучення ознак із даних, а в [101] додатково застосовано рекурентні нейронні мережі з вентильними рекурентними вузлами для класифікації акустичних даних. Проведено обчислювальні експерименти на базі звуків довкілля та розв'язано задачу виявлення специфічних звукових подій,

наприклад плачу дитини, звуку сигналізації або пострілів у публічних місцях тощо.

Роботу [135] присвячено розв'язанню задачі класифікації музики та мовлення. Запропоновано метод оптимізації кількості ознак для задачі класифікації аудіозаписів музики та мовлення у форматі MPEG. Для класифікації використано метод *k*-середніх та нечітка нейронна мережа. Для виділення тих ознак, які найбільше впливають на результат класифікації, застосовано такі два підходи: аналіз головних компонент та генетичний алгоритм.

Одним із практичних застосувань класифікаторів звуків є розробка вбудованих систем [67, 105, 155].

Автори робіт [142, 154, 173] пропонують комбінування класичної згорткової архітектури з так званим механізмом уваги нейронних мереж. Завдяки цьому підходу можна визначити, які саме фільтри будуть передавати сигнали на наступні шари мережі, а які – не будуть.

У [148] розглянуто задачу розпізнавання музичних записів та співу птахів. Для класифікації використано комбінацію методу кластеризації за допомогою самоорганізованих карт Кохонена та класифікатор, який працює на основі нейронних мереж векторного квантування.

Автори статті [162] досліджують випадок задачі класифікації аудіо даних, при якому дані для навчання включають фоновий шум, а дані тестового набору записано без шуму. У цьому разі розподіли ознак відповідних даних для тренування та тестування можуть бути схожі, проте зміщенні відносно один одного. Розроблено модифікацію логістичної регресії для нівелювання статистичних зміщень при навчанні та тестуванні класифікатора.

Ефективність класифікаторів можна підвищити за рахунок використання ансамблевого навчання, яке передбачає об'єднання декількох моделей в одну, з наступним узгодженням результатів всіх класифікаторів за деяким алгоритмом. Дослідження свідчать, що ефективність ансамблю,

зазвичай, вище ефективності окремих моделей [84]. Водночас висувають вимогу відсутності кореляції між моделями, що входять до ансамблю.

Одним з підходів до побудови ансамблів є використання різних типів класифікаторів (наприклад дерева прийняття рішень, SVM, класифікатор Баєса тощо) та формування спільного прогнозу шляхом простого голосування, обчислення середнього або спеціального алгоритму машинного навчання. Альтернативою може бути навчання однакових типів класифікаторів на різних підмножинах тренувальних даних з подальшим усередненням результату прогнозу [84].

Останнім часом саме використання ансамблів глибинних нейронних мереж зумовлює розвиток практичного застосування машинного навчання [31, 84]. Проте, незважаючи на високу точність, ансамблеве навчання нейронних мереж застосовують не так широко, як ансамблі більш класичних методів машинного навчання. Це можна пояснити високою вимогливістю до часових та просторових ресурсів.

Більшість робіт із застосування ансамблю нейронних мереж спрямовано на дослідження методів генерації спільного результату з отриманих результатів навчених класифікаторів.

У роботі [93] запропоновано замість навчання M нейронних мереж навчати одну мережу. Основна ідея полягає в тому, що при застосуванні методу стохастичного градієнтного спуску запам’ятовують значення вагової матриці при потраплянні у задану кількість точок локального мінімуму. Після цього для кожної з M вагових матриць генерують відповідну нейронну мережу. Отже, час навчання ансамблю майже не відрізняється від часу навчання однієї нейронної мережі [93].

Під автокодувальником розуміють нейронну мережу, яка здатна генерувати на виході сигнал, що подібний до того, що надходить на вхід мережі у сенсі заданої метрики [31, 86]. Метрикою може бути середнє квадратичне відхилення, середнє абсолютне відхилення, відносна похибка тощо. Тобто, протягом навчання така нейронна мережа підлаштовує свої

параметри таким чином, щоб імітувати вхідні дані. Поширеними застосуваннями автокодувальників є розширення набору даних для навчання нейронних мереж та зменшення простору ознак. Такі нейронні мережі мають симетричну структуру і кількість нейронів на виході дорівнює кількості нейронів на вході. Середній шар автокодувальника найчастіше має меншу кількість нейронів та представляє собою абстрактні ознаки вхідного сигналу.

Автокодувальники зазвичай використовують у поєднанні з іншими типами нейронних мереж, що дозволяє підвищити ефективність систем класифікації. Розглянемо типові застосування автокодувальників у задачах розпізнавання звуків.

Одночасне використання автокодувальників для вилучення ознак та\або для розширення даних на етапі навчання реалізовано в статтях [58, 63, 92, 99, 136, 143, 163, 176].

У роботі [58] представлено клієнт-серверну систему класифікації звуку. Реалізовано послідовність автокодувальників для компресії ознак сигналу, які передано на сторону сервера. На сервері працює декодер, який відновлює сигнал та застосовує класифікатор.

Алгоритм балансування незбалансованої вибірки для навчання запропоновано в роботі [130].

Для виділення інформативних ознак застосовують майже усі типи автокодувальників (звичайні, рекурентні, варіаційні тощо). Порівняння різних типів цих мереж у задачі класифікації музики виконано у роботі [136].

У задачі класифікації з одним класом у роботі [143] запропоновано реалізацію автокодувальника, який розширює набір числових даних для навчання.

Ще одним можливим застосуванням у задачах класифікації є фільтрування шуму [176]. При цьому підході навчальну вибірку формують з оригінальних та зашумлених даних, які утворюють додаванням випадкової послідовності.

Суттєвий внесок у дослідження нейромережевих моделей та методів їх застосування зробили представники наукових шкіл України. Доволі багато робіт присвячено розвитку нейро-нечітких підходів і теоретичним засадам застосування нейронних мереж у дискретних та неперервних задачах. Деякі основні результати опубліковано в монографіях і навчальних посібниках [3, 8, 11, 12, 29, 32, 37, 39, 48-51]

Окремим напрямом є дослідження методів поєднання нейромереж з виведенням на основі нечіткої логіки [4, 7, 9, 43]. Так, у [4] проаналізовано можливості застосування технології інтелектуальних агентів до гібридних нейро-нечітких систем, а в роботі [7] розглянуто еволюційні підходи оптимізації вейвлент-нейро-нечітких нейронних мереж.

У роботі [30], запропоновано метод сегментації супутникових зображень, для підвищення якості якої застосовано морфологічну обробку, яка забезпечує зменшення кількості аналізованих областей. Забезпечує високу точність розпізнавання образів метод розроблений у [54], який полягає у переході з простору первинних ознак об’єктів у простір похибок за допомогою нейронної мережі.

Питання зменшення часу синтезу прогнозуючих нейромережевих моделей у системах обробки інформації та управління для підтримки прийняття рішень розглянуто у роботі У роботі [33]. Для оцінки міри якості навчальної вибірки запропоновано такі характеристики: нерівномірність, яку характеризує середньоквадратичне відхилення, суперечливість, як середнє арифметичне суперечностей двох навчальних наборів, середньоквадратична помилка перетворення вхідних даних, кількість розпізнаваних класів. Надано критерій якісної навчальної вибірки.

Методику підвищення якості та швидкості синтезу нейромережевих моделей для розпізнавання образів розглянуто у роботі [34]. Запропоновано еволюційний метод відбору ознак з групуванням, в якому для пошуку інформативної комбінації ознак на етапі ініціалізації виконано групування ознак за ступенем їх близькості, а також еволюційний метод структурно-

параметричного синтезу, нейромоделей для розпізнавання образів. Наведено послідовність відбору ознак за допомогою еволюційного методу з фіксацією частини простору пошуку. Еволюційний метод, який розроблено, дозволяє рівномірно покривати простір пошуку. Оптимізацію структури виконано за допомогою еволюційного методу спрощення нейронної мережі.

Алгоритм вирішення задач класифікації об’єктів, які задані набором числових ознак запропоновано у [38]. Для визначення кількості кластерів при кластеризації об’єктів запропоновано метод, який базується на використанні нейронної мережі Кохонена, критерію якості отриманих кластерів та методі ідеальної точки.

У роботі [41] удосконалено методи навчання нейронних мереж для вирішення задач стиснення зображень, запропоновано конкурентну нейронну мережу, яка в якості вхідного сигналу використовує фрагменти зображень. За допомогою фільтра Калмана-Мейна додатково аналізується інформація, що оброблюється, в режимі реального часу.

Роботи [10, 46], присвячено задачам інтелектуального аналізу даних. У [10] запропоновано гібридну спайк-нейронну мережу для нечіткої кластеризації даних, у [46] – модель тришарової автоасоціативної нейронної мережі, за допомогою якої розв’язано задачі обробки та класифікації біомедичних даних. Створенню математичного апарату для застосування в медико-біологічних дослідженнях присвячено роботу [13]. Метод рівномірного розподілу нейромережевої обробки даних викладено у [25], пропонується нова модель оцінювання прискорення навчання та функціонування багатошарових штучних нейронних мереж.

Одним з підходів до автоматичного синтезу оптимальних нейромереж та вибору множини суттєвих ознак є застосування інтелектуальних агентів [36]. Основна ідея таких методів полягає у випадковому пошуку у просторі параметрів. Завдяки тому, що моделювання імітує поведінку групи агентів, ймовірність стагнації у локальних мінімумах зменшується. Однією з переваг також є відсутність вимоги диференційованості цільової функції, що збільшує

діапазон можливих застосувань. Робота [36] розглядає метод відбору інформативних ознак на основі їх подання у вигляді графу пошуку та застосування мультиагентного підходу з непрямим зв'язком між агентами.

Дослідженню та реалізації алгоритмів попередньої обробки зображень на базі нейронних середовищ присвячено роботу [53]. Розглянуто алгоритм реалізації швидких перетворень Фур'є та Адамара на однорідних обчислювальних середовищах. Також досліджено методи попередньої обробки зображень такі як: лінійна фільтрація, медіанна фільтрація, пошук об'єктів заданого класу з використанням узгодженої фільтрації.

У роботі [56] розроблено методологію структурно-параметричного синтезу гібридних нейронних мереж та їх модулів. Запропоновано підхід, при якому на першому етапі генерується базова топологія мережі, яка у подальшому модифікується в результаті розв'язання задачі багатокритеріальної оптимізації. Передбачено використання еволюційного алгоритму для локалізації зони пошуку оптимальної топології нейромережі.

Одним із ефективних еволюційних підходів до розв'язання задач дискретної оптимізації є генетичні алгоритми, використання яких у синтезі нейромереж розглядається у наступному підрозділі.

1.3 Застосування генетичних алгоритмів для структурної оптимізації нейронних мереж та ансамблів

Теорію еволюційних, зокрема, генетичних алгоритмів на даний час добре розроблено, наприклад, у роботах [14, 35, 40, 44, 165], в яких формалізовано основні поняття та описано приклади застосувань до задач дискретної оптимізації. Оскільки генетичні алгоритми є метаевристиками, досягнення глобального оптимуму неможливо гарантувати, однак ці підходи добре зарекомендували себе у низці задач дискретної математики. У роботі [2]

досліджено можливість застосування не тільки генетичних, а й інших евристичних алгоритмів для пошуку оптимальної архітектури нейронних мереж.

Загальна схема застосування генетичних алгоритмів до нейронних мереж полягає у наступному. На першому етапі слід обрати спосіб кодування суттєвих параметрів нейронної мережі у вигляді бінарного або числового вектора. Такі вектори формують деяку базову множину розв'язків *X,* в якій здійснюється пошук оптимального розв'язку. Скінченні непорожні множини *X* називають популяціями [14].

Наступним кроком є вибір цільової функції $f : X \to \mathbb{R}^1$, яка визначатиме найбільш вдалу архітектуру окремих нейронних мереж та структуру ансамблю в цілому. Така функція має бути близькою за значенням до функції втрат, що використовують при навчанні нейромереж.

Далі визначають стандартні для генетичних алгоритмів оператори: селекції, кросовера, мутації та відбору. Докладно зміст та призначення цих операторів викладено у монографіях [14, 35]. Існують також різноманітні варіації наведеного підходу, наприклад еволюційно-фрагментарний алгоритм [14, 15].

У роботах [168-170] розглянуто застосування генетичних алгоритмів до задач комбінаторної оптимізації.

Методи еволюційної оптимізації нейронних мереж використовують як для пошуку оптимальної архітектури, так і для оптимізації вагових коефіцієнтів. Водночас, на практиці використовується не тільки дарвінівська модель еволюції, а й інші теорії. Роботи [2, 6] присвячено застосуванню різних еволюційних підходів та алгоритмів до задачі оптимізації мереж прямого поширення сигналу.

Оскільки саме згорткові нейронні мережі відіграють суттєву роль у задачах розпізнавання образів, розглянемо далі останні публікації із застосування генетичних алгоритмів до структурної та параметричної оптимізації цього типу мереж.

Огляд підходів до використання генетичних алгоритмів для оптимізації згорткових нейронних мереж розглянуто в [42, 66, 69, 85, 90, 114, 116, 118, 122, 123, 125, 133, 134, 152, 153]. У практичних застосуваннях генетичних алгоритмів до нейронних мереж розв'язано задачі вибору найбільш оптимальної структури шарів, оптимізації гіперпараметрів та вибір функції втрат. Основними задачами у цьому разі є вибір методу кодування можливих розв'язків та виду генетичних операторів. Слід зазначити, що кількість потенційних мережевих структур експоненційно збільшується з кількістю шарів у мережі [114], що робить генетичні алгоритми ефективним підходом для пошуку у такому великому просторі можливих розв'язків.

Роботу [152] присвячено застосуванню методів генетичного програмування для оптимізації архітектури згорткової нейронної мережі, яка у свою чергу, представлена направленим ациклічним графом. Перевагою цього методу подання є його гнучкість, оскільки він може представляти мережеві архітектури змінної довжини. Параметричну оптимізацію нейронних мереж базової множини виконано методом стохастичного градієнтного спуску. Функція втрат при навчанні мережі є цільовою функцією генетичного програмування.

Автори [122] досліджують застосування генетичних алгоритмів до оптимізації глибинних нейронних мереж. Множину можливих розв'язків розділено на підмножини параметрів для згорткових шарів та повнозв'язних шарів. Виділено такі параметри як розміри ядра згортки у шарах згортки, типи функцій активації, розмір вікна шару субдискретизації, кількість нейронів у повнозв'язних шарах. Генетичний алгоритм використано у комбінації з методом прогнозування ефективності згенерованої структури мережі. Передбачена за допомогою нейронної мережі ефективність визначає перспективність використання архітектури, згенерованої генетичним алгоритмом.

У роботі [153] запропоновано спосіб кодування можливих розв'язків, при якому кількість шарів нейронної мережі є необмеженою, також

використано блок, з пропусками з’єднань між нейронами для уникнення проблеми зникання градієнту.

У статті [124] розроблено автоматичну систему екстракції ознак, в якій використано генетичні алгоритми для отримання оптимальної, з точки зору певної задачі, комбінації операторів. Операторами можуть виступати різноманітні перетворення сигналу з метою отримання додаткових характеристик: алгоритми перетворення Фур’є, фільтри, алгоритми визначення інтервалів тощо. Функцію пристосованості генетичного алгоритму обчислено як міру якості класифікації аудіоданих. Для класифікації використано метод опорних векторів.

Підходи до оптимізації гіперпараметрів згорткових нейронних мереж розглянуто у роботах [118, 123]. Запропоновано підхід вибору найкращої конфігурації. У роботі [123] замість стандартного для згорткових мереж ядра згортки використано ядро Габора, яке відповідно застосовує перетворення Габора до певної підмножини даних. Відзначено, що використання ядер Габора зменшує час навчання нейронних мереж.

У роботах [90, 125] відзначено, що у практичних застосунках тільки частина гіперпараметрів нейронної мережі суттєво впливає на точність прогнозування, а для різних наборів вхідних даних, оптимальними можуть бути різні набори гіперпараметрів. Розглянуто задачу класифікації зображень та досліджено зв’язок між характеристиками зображень, які надходять на вхід нейронної мережі та її гіперпараметрами.

Окрім застосування класичних варіантів генетичного алгоритму, поширеними є методи, які розроблено суто для оптимізації нейронних мереж. Особливістю таких підходів є як оптимізація архітектури та гіперпараметрів мереж, так і оптимізація ваг нейронів. Це сімейство методів нейроеволюції з розширенням топології (англ. NeuroEvolution of Augmenting Topologies, NEAT) [95, 128, 146, 147].

Огляди робіт з нейроеволюційних методів та метаевристик пошуку оптимальної архітектури нейронних мереж наведено у роботах [78, 140].

При автоматизації пошуку оптимальної топології важливим питанням є вибір способу кодування архітектури мережі. У разі прямого кодування потенційних рішень у популяції кожен ген відповідає за певний гіперпараметр. Отже, такий підхід не є зручним при моделюванні глибинних нейронних мереж, оскільки розмір опису одного рішення та популяції в цілому дуже швидко збільшується [128]. Одним зі шляхів вирішення цієї проблеми є кодування стандартизованих блоків, наприклад у статтях [65, 153] запропоновано такі блоки для згорткових нейронних мереж.

Іншим підходом до розв'язання проблеми кодування структури нейронних мереж у генетичному алгоритмі є метод нейроеволюції з розширенням топології на базі гіперкубу (англ. Hypercube-based NEAT, HyperNEAT). Для кодування топології мереж запропоновано використання нейромережі генерування складних паттернів (англ. Compositional pattern-producing networks, CPPN), що вперше розглянуто в роботі [146].

Питання представлення нейронних мереж для подальшого аналізу та оптимізації розглянуто також у роботах [107, 132]. Запропоновано підхід визначення подібності нейронних мереж з однаковим значенням цільової функції. Це дає змогу використовувати у популяціях генетичного алгоритму найбільш прості архітектури мереж, що зменшує час обчислень. У [159] використано так звану коеволюцію, або спільну еволюцію двох різних популяцій для представлення глибинних нейронних мереж.

Крім методів дискретної оптимізації в задачах пошуку оптимальних архітектур нейромереж використовують також алгоритми неперервної оптимізації. В роботі [115] запропоновано алгоритм DARTS, який зводить простір пошуку оптимальної архітектури до неперервного виду, що дозволяє застосовувати метод градієнтного спуску.

Поєднання описаних вище генетичних алгоритмів та чисельних методів оптимізації описано в роботах [87, 117]. Метаевристики, а саме варіанти алгоритму імітації відпалу (англ. Simulated annealing) та розподіленого генетичного алгоритму, застосовано у [88, 120].

Автори роботи [76] використовують метод пошуку «новизни» як альтернативу поняттю цільової функції в генетичному алгоритмі.

Одним із підходів до підвищення точності прогнозування нейронних мереж є побудова ансамблів декількох моделей. Зазвичай при створенні таких систем виникають дві основні задачі: генерація окремих нейронних мереж ансамблю та обчислення спільного результату прогнозування [84].

Окремі нейронні мережі, які входять до ансамблю, повинні розрізнятись між собою за структурою та параметрами. Цього досягають або використанням різних даних на етапі навчання, або диференціюванням параметрів мережі, або залученням різних алгоритмів параметричної оптимізації.

Ансамблевому навчанню нейронних мереж присвячено, зокрема, роботи [112, 139, 145, 156, 161]. У статтях [112, 156] детально розглянуто основні етапи побудови ансамблів нейромереж. Запропоновано підхід до генерації моделей ансамблю, з яких методом *k*-середніх проводиться відбір для подальшого навчання.

У роботі [161] розглянуто метод спільного навчання як альтернативу ансамблевому навчанню. Введено поняття функції втрат для оптимізації нейронних мереж саме при спільному навчанні.

Роботи [139, 145] присвячено особливостям використання генетичних алгоритмів при створенні ансамблів. Автори [139] пропонують підхід, у якому нейронні мережі та структура ансамблю поєднують в одну популяцію. По-перше, створюють набір нейронних мереж з високим ступенем різноманітності. Для цього використовують різні набори навчальних даних для кожної моделі, а також варіюють архітектуру шляхом зміни кількості прихованих нейронів, функцій активації та ініціалізації ваг. По-друге, генетичний алгоритм використовують для вибору як найбільш ефективної підмножини синтезованих нейронних мереж, так і оптимальної комбінаційної стратегії для забезпечення точності та надійності ансамблю. Така схема

застосування генетичних алгоритмів до нейронних мереж та їх ансамблів є типовою.

Одним із недоліків використання ансамблів нейронних мереж є висока вимогливість до часових та просторових ресурсів, оскільки час навчання збільшується пропорційно до кількості нейронних мереж в ансамблі. Особливо цей недолік стосується глибинних нейронних мереж. Застосування сучасних обчислювальних засобів, таких як графічні та тензорні процесори, суттєво прискорюють процес тренування нейромереж. Однак розробка ефективних методів ансамблювання та пошуку оптимальних топологій мереж дасть змогу отримувати нові результати у теорії та отримувати інноваційні застосування на практиці.

1.4 Висновки до розділу 1

З аналізу літературних джерел можна зробити висновок, що задача класифікації акустичних даних і загалом розробки систем машинного слуху є досить актуальною. Опубліковані на даний момент наукові статті можна умовно розділити на три категорії.

До першої категорії включено роботи, в яких автори виконують попередню обробку сигналу з метою сегментації та вилучення ознак, потім навчають класифікатор, на вхід якого подають вектори ознак. У цих роботах зазвичай застосовують перетворення Фур'є, обчислення мел-частотних кепстральних коефіцієнтів та інших частотних або спектральних характеристик сигналу. Із класифікаторів найчастіше застосовують метод опорних векторів, *k*-найближчих сусідів, дерева прийняття рішень, метод *k*-середніх, нейронні мережі та ін. Можуть також застосовувати ансамбль декількох класифікаторів, у цьому разі клас-переможець визначають шляхом голосування.

До другої категорії включено публікації, в яких автори прагнуть автоматизувати процес побудови оптимального набору ознак для застосування класифікаторів. Серед підходів, які застосовують для такої автоматизації, можна виділити генетичні алгоритми та нейронні мережі. Класифікатори використовують ті ж, що й у публікаціях попередньої категорії.

У публікаціях третьої категорії автори застосовують підходи глибинних нейронних мереж. Часто це згорткові нейронні мережі, на вхід яких можна подати як дані без попередньої обробки, так і після спеціальної підготовки. Ефективність такого підходу можна пояснити багатошаровою архітектурою згорткових нейронних мереж. Передбачено наявність декількох типів шарів: шари згортки, в яких виділяють певного виду ознаки; агрегувальні шари, які застосовують для зменшення розмірності; та декілька повністю зв’язних шарів, в яких виконують класифікацію. До недоліків такого підходу можна віднести складність налаштування нейронних мереж зі складною архітектурою та вимогливість до обчислювальних ресурсів. Реалізація глибинних нейронних мереж зазвичай потребує системи паралельних та розподілених обчислень, залучення графічних процесорів (GPU).

Отже, переважну більшість розглянутих публікацій орієнтовано на оптимізацію одного з аспектів систем машинного слуху, наприклад, набору ознак або структури та типів класифікаторів. Практично не розглянуто комплексний підхід, при якому використання ансамблевого навчання із застосуванням нейронних мереж поєднано з оптимізацією за допомогою генетичних алгоритмів.

Також на даний момент відсутня інструментальна система класифікації звукових даних, яка б поєднувала різні методи та етапи попередньої обробки, класифікації та постпроцесорного представлення результатів роботи.

Основні результати першого розділу опубліковано у роботах [16-24, 47, 55, 103-106].

РОЗДІЛ 2
МАТЕМАТИЧНІ МОДЕЛІ ЗВУКОВИХ СИГНАЛІВ У ЗАДАЧАХ КЛАСИФІКАЦІЇ

Моделювання звукового сигналу, тобто вибір такого представлення вхідного звуку, яке є найбільш оптимальним для подальшої класифікації, суттєво може вплинути на ефективність системи розпізнавання. Розробка автоматичних методів синтезу абстрактних ознак може відбуватися у напрямі поєднання певних природних ознак безпосередньо в шарах нейронних мереж із подальшою класифікацією. Таке завдання виконують шари згортки відповідних нейромереж або автокодувальники, які додатково зменшують розмірність множини ознак.

У даному розділі використано методи теорії обробки сигналів, які є фактично прикладами класичних математичних моделей. Описано основні технічні характеристики наборів даних, що використано в подальших розділах для тестування моделі класифікації та проілюстровано обчислення деяких основних ознак аудіоданих.

Виконано формальну постановку задачі класифікації звукових сигналів на основі понять теорії розпізнавання. Розроблено гібридні нейромережеві математичні моделі класифікації звуків із використанням шарів згортки та автокодувальників. Спільне навчання такої гібридної архітектури може значно підвищити точність прогнозування в цілому. Водночас різні шари можуть виконувати відмінні одна від одної задачі, наприклад, шари згортки застосовуються як для отримання абстрактної множини ознак, так і для розширення навчальної вибірки. Розроблені формалізації дали змогу сформулювати і довести теорему про збіжність гібридної нейромережі.

Розроблені математичні моделі гібридних нейронних мереж розпізнавання аудіоданих використано в наступних розділах при реалізації інструментальної системи класифікації.

2.1 Ознаки звукових сигналів

Загальну схему системи класифікації звуків, що запропоновано у даній роботі, наведено на рисунку 2.1.

На етапі попередньої обробки даних виконуються операції фільтрації, застосування перетворень довжини та тембру звукового сигналу, розширення наборів даних шляхом генерування нових звуків. Даний етап є суттєвим для універсальної системи класифікації звуків, оскільки вихідна база аудіозаписів може містити звукові файли різної довжини та частоти дискретизації. Операція розширення даних шляхом генерації нових звуків застосовується за умови різної кількості спостережень для різних категорій. Для вирівнювання зразків даних класів навчальної вибірки у дисертаційній роботі використано нейронні мережі автокодувальники а також інші методи балансування класів на основі методу *k*-найближчих сусідів та випадкових вибірок.

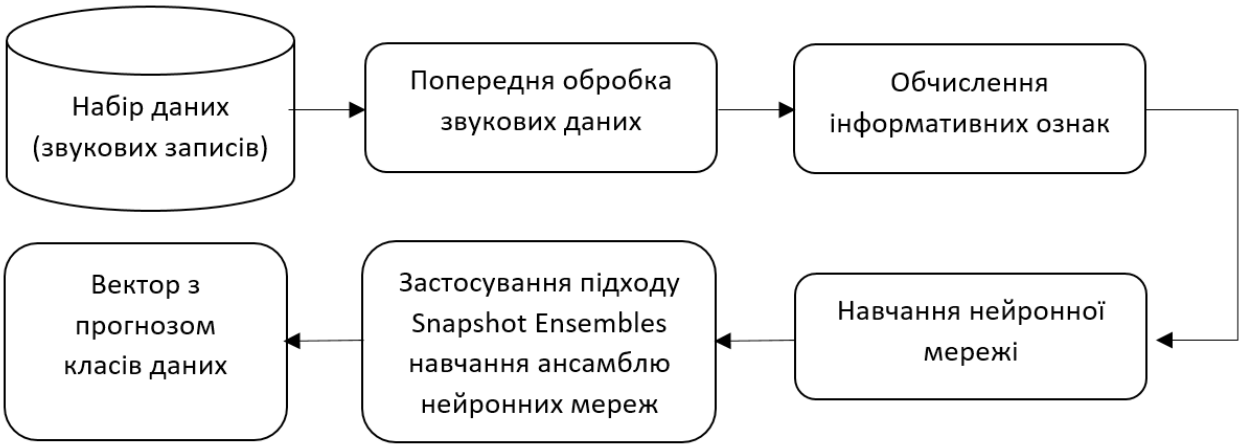


Рисунок 2.1 – Схема системи класифікації звуку

Застосування тих чи інших перетворень залежить від конкретних наборів даних, які використано для навчання та тестування моделей класифікації. Для цього, зазвичай, використовують відкриті дані, які містять записи звуків різного походження та з різними характеристиками.

У подальших розділах роботи використано такі колекції звукових даних: Free sound [81], ESC-50, ESC-10 [131], UrbanSound8k [138]. Зазначені набори обрано з двох причин. По-перше, їх широко використовують інші автори, отже можливе порівняння з іншими моделями. По-друге, дані відрізняються за кількістю записів на один клас, довжиною записів тощо, тобто, система класифікації, яка зможе демонструвати задовільну точність на цих наборах, є досить гнучкою з точки зору різноманітності вхідних даних.

Результати порівняння за такими характеристиками як кількість записів на один клас, довжина записів, частота дискретизації, наведено у таблиці 2.1.

Таблиця 2.1 – Порівняння наборів звукових даних

Назва	Кількість записів	Кількість класів	Рівні класи за об’ємом	Довжина записів, с	Частота дискретизації, Гц
Free sound	11073	41	Ні	$\approx 6{,}7$	≈ 44100
ESC-10	400	10	Так	$=5$	≈ 44100
ESC-50	2000	50	Так	$=5$	≈ 44100
UrbanSound8k	8732	10	Ні	$\approx 3{,}6$	≈ 48456

Як можна побачити з таблиці 2.1 набори Free sound та UrbanSound8k є незбалансованими за кількістю спостережень у кожному класі та довжиною записів.

Загалом для довільних наборів даних може виникнути ситуація, коли кількість спостережень деяких класів навчальної вибірки є порівняно малою (міноритарні класи), а інших – досить великою (мажоритарні класи). Точність класифікаторів, що побудовано на таких незбалансованих навчальних вибірках, не є зазвичай достатньо високою.

Для балансування даних за кількістю спостережень у роботі застосовано методи *k*-найближчих сусідів, ADASYN, SMOTE, випадкове балансування, автокодувальники [130]. Можна класифікувати існуючі методи за стратегіями видалення спостережень мажоритарних класів або синтезу нових зразків

даних у міноритарних класах. Наприклад, методи балансування видаленням спостережень з мажоритарних класів базуються на пошуку близьких точок даних, видалення яких не змінить суттєво розподілення спостережень. Більш детально підходи, які застосовано у роботі для балансування вибірки тренування наведено у підрозділі 3.3.

Обчислення ознак звукового сигналу необхідно виконувати у кожній системі класифікації. Це пов’язано з тим, що безпосереднє представлення звуку у часовій області (залежність амплітуда-час) є не досить ефективним і вимагає додаткових часових та просторових ресурсів для збереження й обробки даних. Для найбільш раціонального представлення акустичного сигналу у дисертаційній роботі використано класичні методи цифрової обробки сигналів [5]. Серед них можна виділити перетворення, які розкладають сигнал за ортогональними базисними функціями: перетворення Фур’є, Хартлі, Мелліна, вейвлет тощо, а також різноманітні ознаки сигналу, які обчислюють на базі цих перетворень, наприклад, мел-частотні кепстральні коефіцієнти (англ. Mel-frequency cepstral coefficients, MFCC), частотні коефіцієнти перетворення гамматон (англ. Gammatone frequency cepstral coefficients, GFCC), коефіцієнти константного Q-перетворення (англ. Constant-Q transform, CQT), хромаграми (англ. chromagram) тощо [59]. Алгоритми обчислення наведених ознак детально описано у роботі [109].

Наведені характеристики сигналу є досить поширеними та їх часто використовують для розв’язання задач класифікації звуків різного походження. Наприклад, у роботі [142] запропоновано поєднання чотирьох ознак: MFCC, GFCC, CQT та хромаграм в один чотиривимірний масив. Застосовують також архітектури нейронних мереж, які передбачають наявність декількох вхідних гілок [113, 151]. Це дає змогу використовувати входи різної вимірності. На відміну від цього підходу, для поєднання декількох ознак у роботі використано автокодувальники, де у середньому шарі обчислюється компактне абстрактне представлення довільних вхідних ознак, яке згодом використовується у класифікаторі.

Обчислення спектру та пов’язаного поняття кепстр відбувається за схемою, зображеною на рисунку 2.2.

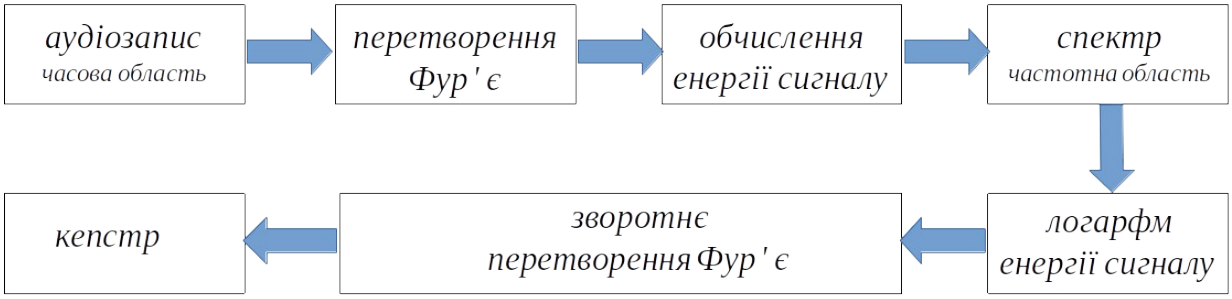


Рисунок 2.2 – Схема перетворення спектр/кепстр

Спектральний аналіз дає змогу виявити основні гармоніки аудіосигналу, що є досить важливим для задач класифікації. На базі представлення звуку у вигляді спектру/кепстру обчислюють дійсні коефіцієнти одного зі стандартних перетворень звуку $F()$, наприклад, мел-частотні, барк-частотні або коефіцієнти гамматон фільтру (див. рис. 2.3), що є більш зручним ніж використання комплексних значень після перетворення Фур’є.

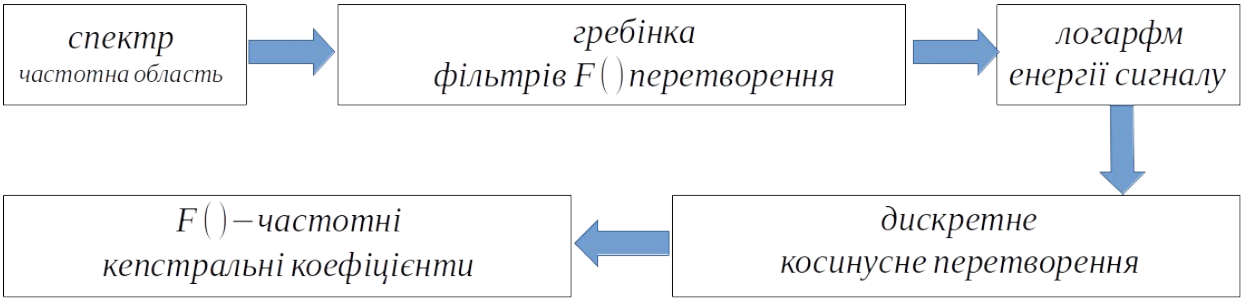


Рисунок 2.3 – Обчислення мел/гамматон-частотних кепстральних коефіцієнтів

Використання хромаграм (тональних класів, англ. Pitch Class Profile) для представлення аудіозапису відповідає розподіленню за частотами із застосуванням дванадцятитонового представлення, що застосовують у музиці, проте також може бути залученим для аналізу звуків природного походження. Загальну схему обчислення наведено на рисунку 2.4.

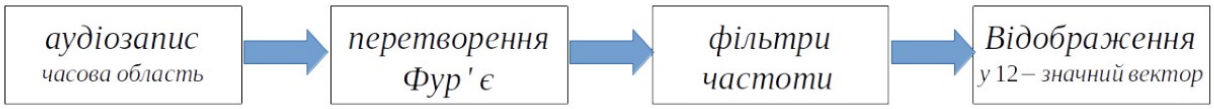


Рисунок 2.4 – Обчислення хромаграми

Константне Q-перетворення є варіантом вейвлет перетворення Габора-Морлета у представленні час-частота [109]. Завдяки цьому перетворенню можна змінювати рівень деталізації в залежності від частоти.

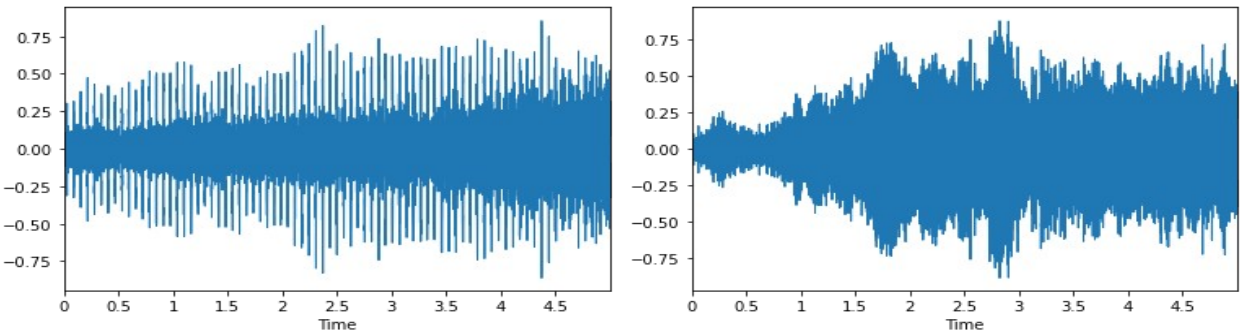


Рисунок 2.5 – Ознаки даних з набору ESC-10 (пила, гелікоптер).

Розглянемо далі деякі приклади обчислення ознак для прикладу звуку з набору даних ESC-10. Мел-частотні кепстральні коефіцієнти для наведених на рисунку 2.5 звуків обчислено за схемою рисунку 2.3, діаграму MFCC представлено на рисунку 2.6. Візуально можна побачити відмінності між звуками різного походження. При навчанні нейронної мережі ці відмінності використовуються як суттєві для класифікації.

Перевагою використання наведених ознак у спектральній та кепстральних областях є більш компактне представлення вихідного звукового сигналу. Більш ефективним підходом є поєднання декількох ознак в одну структуру даних, що може зумовити підвищення точності прогнозування.

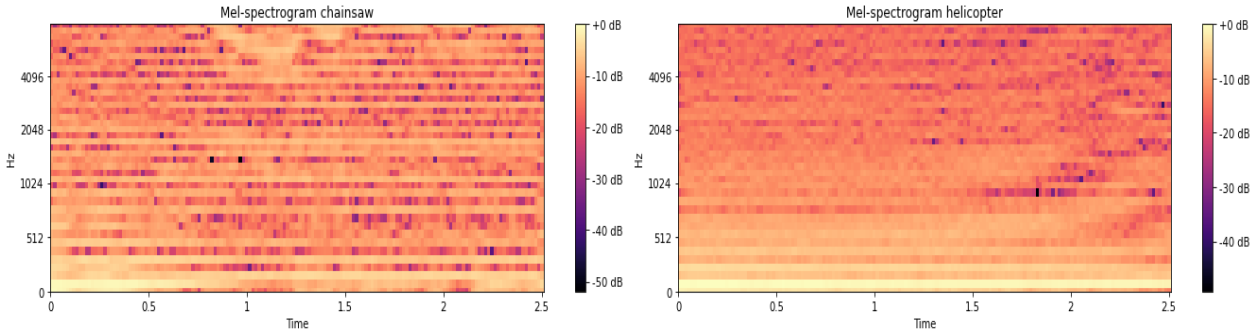


Рисунок 2.6 – Мел-частотні кепстральні коефіцієнти

Хромаграми, які обчислено за схемою рисунку 2.4, наведено на рисунках 2.7 та 2.8. Досить суттєва різниця у візуальному представленні коефіцієнтів свідчить про потенціал застосування цих ознак у задачах розпізнавання.

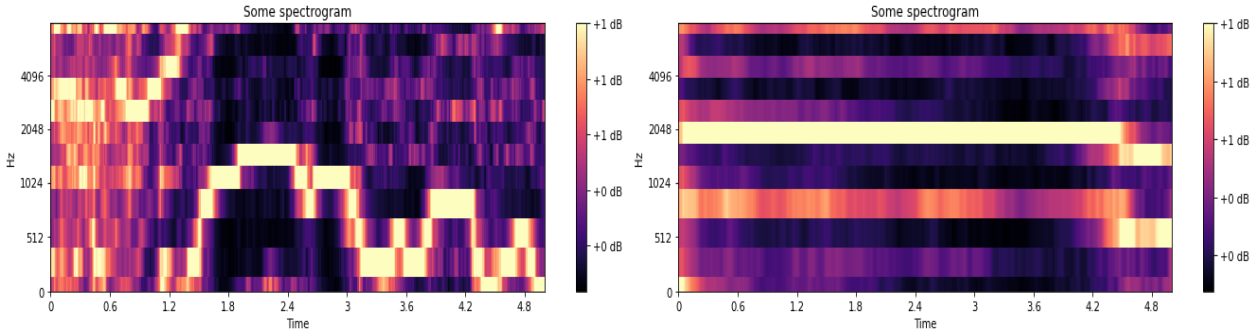


Рисунок 2.7 – Представлення хромаграм

Разом з мел шкалою для аудіоданих застосовують барк шкалу та гамматон фільтр, які встановлюють нелінійну відповідність між частотою звуку та сприйняттям тону [5]. Використання наведених хромаграм у поєднанні з іншими ознаками звуку дозволяє підвищити точність розпізнавання оскільки в результаті можна отримати вираження одного звуку у різних шкалах. Недоліком такого підходу є збільшення вимірності навчальної вибірки. Саме для зменшення розмірності даних у роботі використано нейронні мережі автокодувальники. З іншого боку, такий підхід оптимізує множину ознак звукових даних, внаслідок чого зменшується розмірність задачі класифікації та час роботи класифікатора.

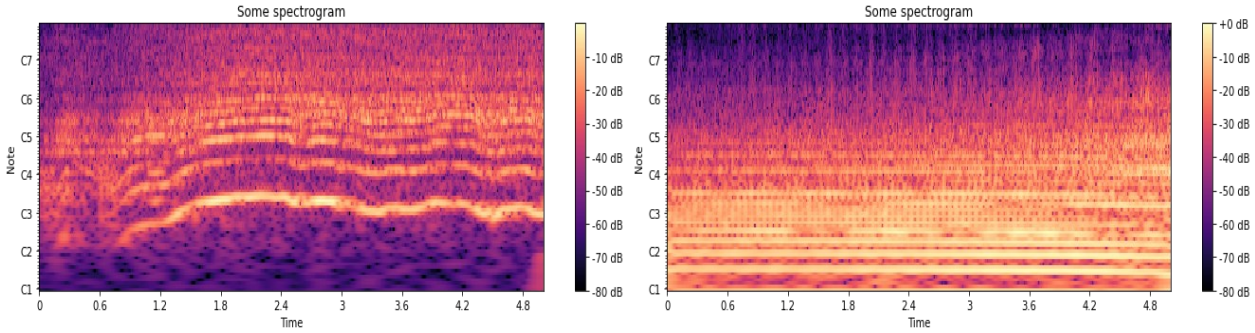


Рисунок 2.8 – Хромаграми на основі коефіцієнтів константного Q-перетворення

Діаграми барк та гамматон коефіцієнтів наведено на рисунках 2.9 та 2.10 відповідно. Психоакустична природа цих одиниць вимірювання частот пов’язана з особливостями сприйняття звуку людиною. Отже, мел, барк представлення та гамматон фільтри також широко застосовують у системах класифікації.

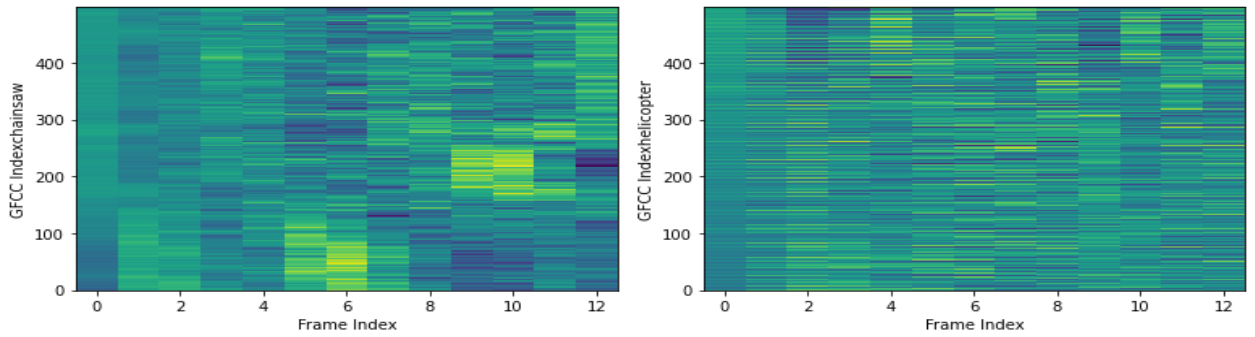


Рисунок 2.9 – Гамматон-частотні кепстральні коефіцієнти

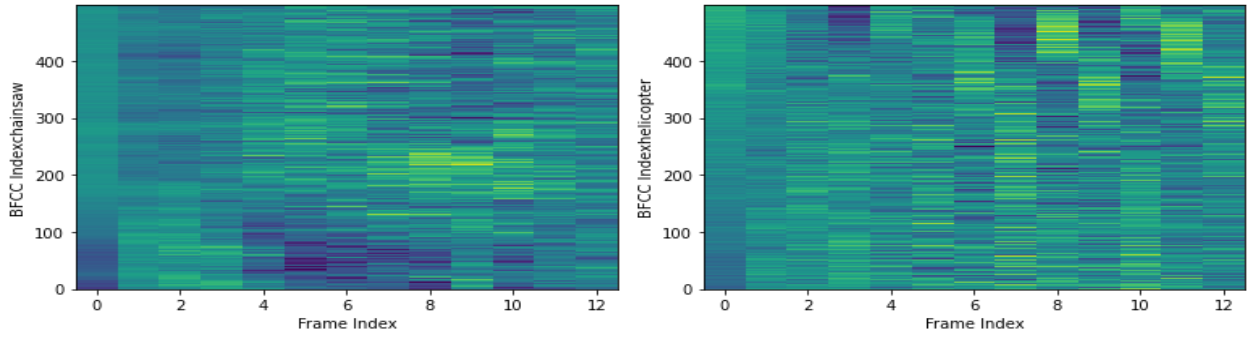


Рисунок 2.10 – Барк-частотні кепстральні коефіцієнти

Усі наведені на рисунках 2.7–2.10 ознаки використано у роботі для представлення звуку у вигляді, зручному для подальшого розпізнавання.

Після виконання попередньої обробки звуку та обчислення векторів характерних ознак застосовано згорткові нейронні мережі, математичну модель яких наведено в підрозділі 3.2.

2.2 Формалізація задачі класифікації акустичних сигналів

Розглянемо формальну постановку задачі багатокласової класифікації, використовуючи поняття теорії розпізнавання образів [26-28].

Нехай $U=\{u_1,\dots,u_N\}$ – множина об'єктів, які підлягають класифікації, наприклад, аудіозаписів. $\Omega=\{\omega_1,..,\omega_k\}$ – множина класів – розподілення елементів множини U за певним принципом класифікації, наприклад, за джерелами звуків в аудіозаписах. До множини класів задачі багатокласової класифікації висуваємо такі вимоги:

$$\cup_{i=1}^{k}\omega_i=U,\ \omega_i\cap\omega_j=\varnothing\ \forall i\neq j. \tag{2.1}$$

Оскільки об'єктами множини U можуть виступати довільні аудіозаписи, введемо функцію $F()$ для обчислення числових ознак кожного об'єкту:

$$F(U):U\to X, \tag{2.2}$$

де $X=\{X_1,\dots,X_N\}, X_i\in\mathbb{R}^{M\times K}$ – $M\times K$ -вимірний простір ознак.

У даному випадку функція $F()$ реалізує один із методів обчислень ознак звукового сигналу, описаних у підрозділі 2.1: GFCC, MFCC, CQT, BFCC тощо. Результатом застосування функції $F()$ є множина X двовимірних матриць дійсних чисел – ознак об'єктів, де M – кількість коефіцієнтів перетворення $F()$, K – кількість кроків дискретизації вихідного сигналу.

Функція $G()$ ставить у відповідність множині класів Ω певний числовий вектор, зазвичай бінарний або числове значення:

$$G(\Omega):\Omega \to Y, \tag{2.3}$$

де $Y=\{Y_1,...,Y_N\}, Y_j \in \mathbb{R}^k$ – множина міток, яка в залежності від задачі, що розв'язують, приймає такий вигляд:

- $Y=\{0,1\}$ – бінарна класифікація;
- $Y=\{1,\ldots,k\}, k>2$, – багатокласова класифікація;
- $Y=\{0,1\}^k$ – багатокласова класифікація з класами, що мають перетин.

Елементи множин ознак об'єктів X та відповідних міток Y утворюють навчальну вибірку

$$T=\{(x_1,y_1),\ldots(x_N,y_N)\}, \tag{2.4}$$

де $(x_i,y_i), i\in[1,N]$ – спостереження навчальної вибірки або прецеденти;

N – об'єм навчальної вибірки.

Задачу класифікації можна звести до визначення функції класифікатора – деякого алгоритму $A()$ та його параметрів

$$A(W,X):X \to Y, \tag{2.5}$$

де $W=\{w_1,...,w_N\}, w_i \in \mathbb{R}$ – це вектор параметрів алгоритму навчання.

Параметри класифікатора можна визначити з мінімізації функції похибки прогнозування $L()$:

$$L(A(W,X),Y)\rightarrow min_W . \quad (2.6)$$

У залежності від типу класифікації визначають і вид алгоритму $A()$ та функції похибок $L()$.

У випадку задачі бінарної класифікації алгоритмом $A()$ може виступати сигмоїдна функція, яка для заданих вхідних значень нейронів генерує одну з двох можливих відповідей:

$$A(W,X)=\sigma(W\cdot X+w_0)=\sigma\left(\sum_{j=1}^{N} w_j x_j+w_0\right), \quad (2.7)$$

де $W=\{w_1,...w_N\}, w_i\in\mathbb{R}$ – ваговий вектор у даному випадку;

$w_0\in\mathbb{R}$ – зсув;

$\sigma(z)=\frac{1}{1+e^{-z}}$ – сигмоїдна функція.

Функція похибок для бінарної класифікації приймає вигляд перехресної ентропії [84]:

$$L(A(W,X),Y)=-(Y\log(A(W,X))+(1-Y)\log(1-A(W,X)))\rightarrow min_W . \quad (2.8)$$

Для багатокласової класифікації замість сигмоїдної функції використовують функцію *SoftMax* [86]:

$$SoftMax(z_1,\ldots,z_k)=\left(\frac{e^{z_1}}{\sum_{i=1}^{k} e^{z_i}},\ldots,\frac{e^{z_k}}{\sum_{i=1}^{k} e^{z_i}}\right). \quad (2.9)$$

У цьому разі ймовірність *i*-го класу можна визначити за формулою:

$$P\langle y=i|x,w\rangle=\frac{e^{\left(w_i\cdot x+w_{0,i}\right)}}{\sum_{j=1}^{k}e^{\left(w_j\cdot x+w_{0,j}\right)}}\ . \qquad (2.10)$$

Отже, коефіцієнти вагового вектора W можна визначити з умови мінімізації функціоналу похибки, який для задач багатокласової класифікації має такий узагальнений вигляд [86]:

$$L\left(A\left(W,X\right),Y\right)=-\sum_{i}^{N}y_i\log\left(A\left(W,x_i\right)\right)\rightarrow min\ . \qquad (2.11)$$

Зазначені функції, що визначають алгоритм $A()$, реалізовуватимуться штучними нейронами та їх об'єднаннями – нейронними мережами. Найбільш типовими архітектурами в задачах класифікації зображень та звуку є багатошарові перцептрони, згорткові мережі, автокодувальники. У роботі запропоновано топологію гібридних нейронних мереж із використанням шарів згортки, автокодувальника та перцептронів. Побудована в такий спосіб нейронна мережа містить досить велику кількість параметрів, які вимагають подальшого налаштування.

Отже, задачу побудови нейромережевої моделі можна звести до розв'язання задачі багатовимірної оптимізації відносно параметрів W, для рішення якої використовують як прямі або градієнтні методи, так й евристичні підходи на базі еволюційних алгоритмів. Якість класифікатора визначають за допомогою спеціальних метрик: матриці похибок, коефіцієнта точності, повноти, *F*-метрики [84]. Слід зазначити, що в залежності від метрики похибки, що використовується під час навчання, буде змінюватись і час навчання, що є суттєвим для глибинних нейромереж.

Матрицю похибок, яка дає змогу оцінити точність як комбінацію істинної відповіді з вибірки навчання та відповіді алгоритму для бінарної класифікації, наведено в таблиці 2.2.

Таблиця 2.2 – Матриця похибок

	$y=1$	$y=0$
$A(x)=1$	True Positive (TP)	False Positive
$A(x)=0$	False Negative (FN)	True Negative

З матриці похибок випливають похідні метрики, які часто використовують на практиці для оцінки точності розпізнавання.

Частка правильних відповідей визначається за такою формулою [84]:

$$accuracy=\frac{TP+TN}{TP+FP+FN+TN}\ . \tag{2.12}$$

Цю метрику використовують у разі, коли кількість прецедентів у кожному класі є приблизно рівною. При великій кількості класів, використання цієї метрики призводить до завищених значень точності класифікації.

Більш точними метриками є точність, повнота та їх комбінації. Точність визначають у такий спосіб [84]:

$$precision=\frac{TP}{TP+FP}\ . \tag{2.13}$$

Повнота визначає, яку кількість позитивних відповідей було обчислено класифікатором [84].

$$recall=\frac{TP}{TP+FN}\ . \tag{2.14}$$

Найбільш інформативною є *F*-метрика – гармонічне середнє точності й повноти [84]:

$$F=\frac{2\cdot precision\cdot recall}{precision+recall}\ . \tag{2.15}$$

У випадку багатокласової класифікації наведені метрики обчислюються для кожного класу окремо. Для отримання узагальненої точності необхідно обчислити статистичні характеристики середнього та розкиду.

При досить великій кількості прецедентів у навчальній вибірці реалізацію обчислювальної схеми, наведеної вище, можна виконати за допомогою нейронних мереж.

2.3 Математична модель нейронних мереж автокодувальників

Обчислювальним елементом будь-якої нейронної мережі є штучний нейрон. Зазвичай, на вхід нейрона надходить значення множини ознак, нейрон обчислює зважену суму, після чого застосовує нелінійну функцію активації (див. рис. 2.11), де $X_1,\dots,X_N$ ознаки, що характеризують вихідні об’єкти. Ваги зв’язків $W=\{w_1,w_2,\dots,w_N\}$, які налаштовуватимуться протягом навчання, визначають вагу певного входу з погляду розв’язання задачі класифікації.

Вид функції активації залежить від типу задачі, що розв’язують. Для задач класифікації – це сигмоїдна функція або SoftMax (див. табл. 2.3), однак не виключено і застосування інших активацій у проміжних шарах нейронної мережі. Наприклад, вид функції активації для задач регресії залежить від діапазону значень, які необхідно згенерувати на виході мережі. Розглянемо далі активаційні функції, які найбільш часто застосовуються на практиці. До

наведених функцій висувають вимогу диференційованості, оскільки основним підходом навчання нейронних мереж є метод градієнтного спуску [84, 86].

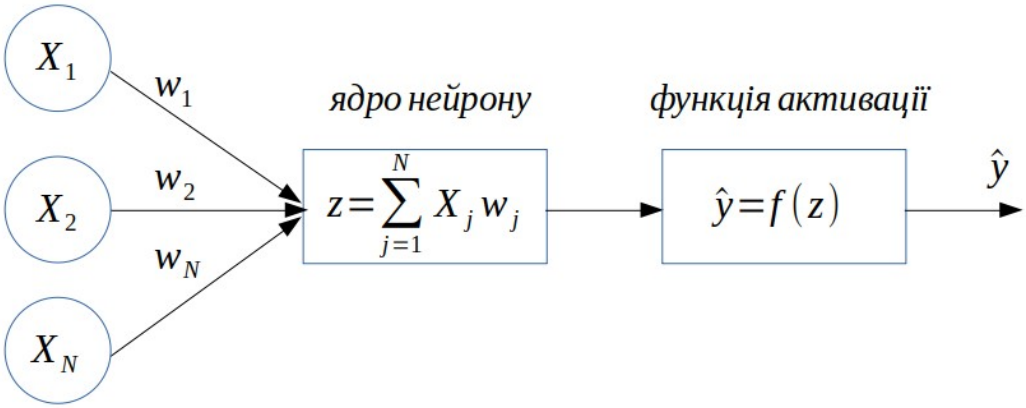


Рисунок 2.11 – Штучний нейрон

У таблиці 2.3 наведено відповідність між функціями активації та формальними задачами.

Таблиця 2.3 – Функції активації та застосування

Назва	Вираз	Задача
Сигмоїда	$f(x)=\frac{1}{1+e^{-x}}$	Бінарна класифікація
SoftMax	$f(x)=\sigma_i(x)=\frac{e^{x_i}}{\sum_{i=1}^{N} e^{x_i}}$	Багатокласова класифікація
Softplus	$f(x)=\ln(1+e^{x})$	Регресія
Softsign	$f(x)=\frac{x}{1+(x)}$	Регресія
Elu	$f(\alpha,x)=\begin{cases}\alpha(e^{x}-1), x\leqslant 0\\ x, x>0\end{cases}$	Регресія
Selu	$f(\alpha,x)=\begin{cases}\alpha(e^{x}-1), x\leqslant 0\\ x, x>0\end{cases}$	Регресія
Relu	$f(x)=\begin{cases}0, x<0\\ x, x\geq 0\end{cases}$	Регресія

Слід зазначити, що оскільки більшість застосувань можна звести до класифікації або регресії, важливим є вибір коректної та швидкої активаційної функції.

Отже, розглянуті в попередньому підрозділі приклади класифікаторів, які задано формулами 2.7, 2.10, можна реалізувати як один нейрон, а поєднання нейронів у багатошарову архітектуру нейронної мережі теоретично дозволяє розв'язувати задачі розпізнавання будь-якої складності [86]. Наприклад, теореми Розенблатта про збіжність перцептрону та теорема Цибенка свідчать про теоретичну можливість розробити архітектуру нейронної мереж для наближення будь-якої нелінійної функції та розв'язання довільної задачі класифікації за скінченний проміжок часу.

Процес навчання нейронної мережі полягає в оптимізації матриці ваг W таким чином, щоб значення, які генерує останній шар максимально близько за обраною метрикою співпадали зі значеннями навчальної вибірки. Отже, нейронні мережі в цьому разі є прикладом навчання з учителем, а результуюча модель визначається даними, які надходять на вхід мережі.

Розглянемо далі найбільш типові способи організації нейронів у мережі. Наприклад, схему мережі штучних нейронів прямого поширення сигналу з l шарами для класифікації на k класів, зображено на рисунку 2.12.

Сигнал на виході j-го нейрону багатошарової мережі обчислюємо за формулою:

$$\hat{y}_j = (f_1 \circ f_2 \circ ... \circ f_l) X_j = f_1(f_2(\dots f_l(X_j))), \qquad (2.16)$$

де $f_1 \circ f_2 \circ ... \circ f_l$ – композиція функцій активації шарів нейромережі.

Для навчання мереж прямого поширення сигналу застосовують, зазвичай, градієнтні методи, наприклад, метод зворотного поширення помилки та його варіації [86]. На практиці використовуються також еволюційні методи, які дозволяють зменшити ймовірність стагнації алгоритму навчання в точках локального мінімуму.

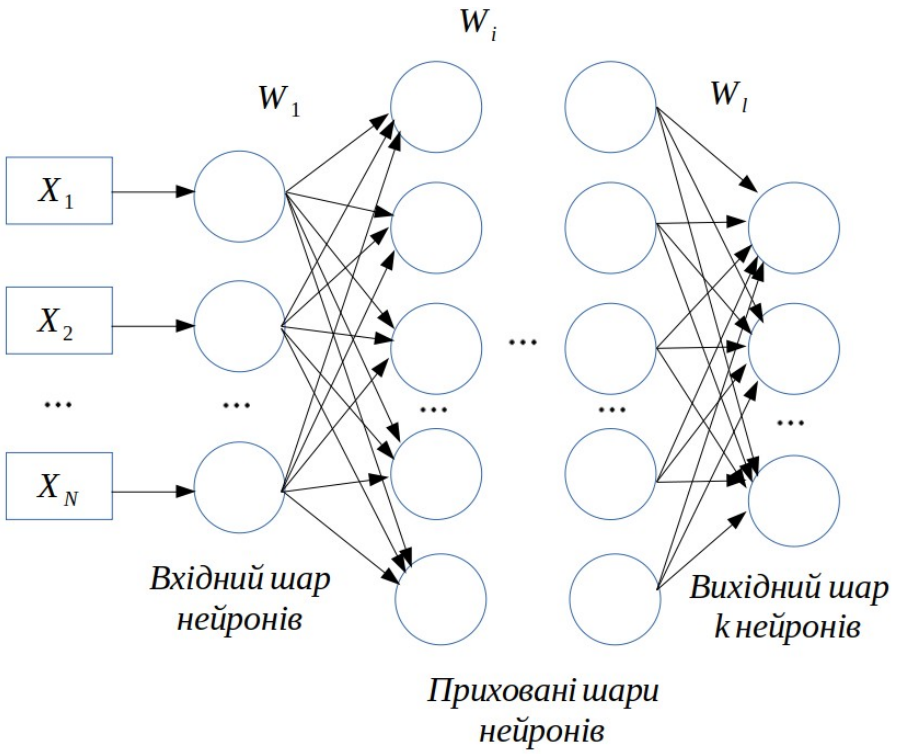


Рисунок 2.12 – Мережа прямого поширення сигналу

У задачах розпізнавання застосовують спеціалізовані мережі прямого поширення сигналу, наприклад, автокодувальники. У даній роботі ці нейромережі використовуємо для зменшення розмірності простору ознак та генерації нових прецедентів, або зразків даних.

Типову архітектуру автокодувальника наведено на рисунку 2.13. Мережа має дві частини: шари кодувальника, які реалізують функції кодування $h=f(X)$ та шари декодування, які генерують сигнал $\widetilde{X}$, схожий за своїми характеристиками із сигналом X: $\widetilde{X}=g(h)$. Значення h формується у прихованому шарі (див. рис. 2.13) та зазвичай має меншу або більшу розмірність порівняно з X. Отже, метою застосування автокодувальника є оптимізація вектору параметрів θ такого перетворення [84]:

$$g(f(X,\theta),\theta)=\widetilde{X},\ X\approx\widetilde{X}. \qquad (2.17)$$

Фактично автокодувальники є прикладом систем, які навчаються без вчителя, оскільки при навчанні нейромереж цього типу дані надходять у нерозміченому вигляді.

Для аналізу аудіоданих у роботі використано два типи автокодувальників: понижувальні (англ. undercomplete autoencoder) та знешумлювальні автокодувальники (англ. denoising autoencoder).

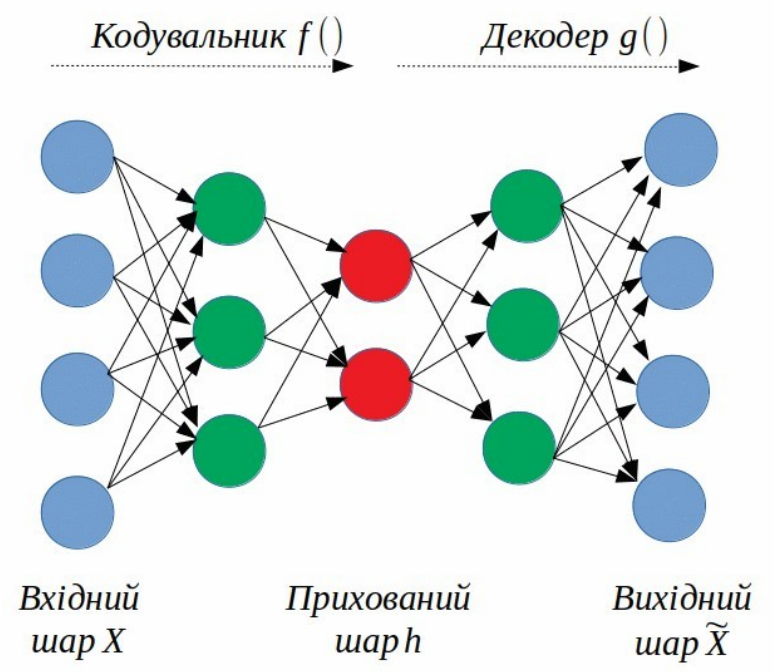


Рисунок 2.13 – Понижувальний автокодувальник

Мета застосування понижувального автокодувальника полягає в мінімізації деякого функціонала похибки $L()$:

$$L\left(X, g\left(f\left(X,\theta\right),\theta\right)\right) \rightarrow min_{\theta}, \tag{2.18}$$

водночас оптимізуємо параметри θ функцій $f()$ та $g()$, які фактично є функціями активації нейронів автокодувальника [31].

Навчання даної мережі можна розглядати як приклад навчанням з учителем із такими змінами. Множини вхідних сигналів також формують

простір ознак $X \in \mathbb{R}^d$, проте множина міток тотожно дорівнює множині ознак $Y \equiv X$, тоді $T = \{(x_i, y_i)\}, i \in [1, N]$ – навчальна вибірка автокодувальника.

Прихований шар h після навчання містить абстрактні інформативні ознаки, які можна використовувати, наприклад, при розв'язанні задачі класифікації. Крім того, згідно з [86], якщо декодер є лінійною функцією, а функціонал $L()$ – середньою квадратичною похибкою, у прихованому шарі буде сформовано аналогічний підпростір ознак, що і при застосуванні методу головних компонент.

Отже, результатом застосування автокодувальника до аудіоданих є отримання в прихованому шарі деякого абстрактного простору ознак. Водночас необхідно уникнути простого копіювання вхідного сигналу у вихідний, тому кількість нейронів шару h повинна бути нижчою за вхід та вихід.

При роботі знешумлювального автокодувальника на вхід надходить початковий сигнал з додаванням шуму – деякого випадкового розподілення $rand(X)$, зазвичай це гаусів шум з нульовим середнім значенням та невеликою дисперсією, яка задає рівень шуму:

Отже, вхідний сигнал автокодувальника буде сформовано як $X + rand(X)$, а перетворення 2.17 набуває такого вигляду [31]:

$$g(f(X + rand(X), \theta), \theta) = \widetilde{X}, \; X \approx \widetilde{X}. \qquad (2.19)$$

Структура нейронної мережі у цьому разі подібна до рисунку 2.13. Знешумлювальні автокодувальники ефективно застосовують в обробці аудіозаписів звуків довкілля або міста, коли є необхідність очистити сигнал. Додатково також шляхом накладання шуму на початковий звук генерується додаткова навчальна вибірка у випадку незбалансованості за класами.

Наразі, навчання нейронних мереж може відбуватися з використанням класичних оптимізаційних градієнтних методів, наприклад, зворотного

поширення помилки [86], але також застосовуються евристичні методи, наприклад, генетичні алгоритми [2].

Далі, в роботі при розпізнаванні аудіоданих використано саме метод зворотного поширення помилки, в якому сигнал поширюється в прямому напрямі вздовж мережі, а в зворотному напрямку повертаються значення похибок, які коригують ваги нейронів.

При навчанні глибинних нейронних мереж із метою зменшення часових витрат використовують такі модифікації методу градієнтного спуску: стохастичний градієнтний спуск, метод пришвидшених градієнтів Нестерова (англ. Nesterov Accelerated Gradient), метод адаптивної оцінки моментів (англ. Adaptive Moment Estimation, ADAM) та інші [84].

Оскільки розмірність даних у задачах класифікації звуків може бути досить великою, застосування евристичного пошуку може призвести до часових втрат.

Загалом, процес зворотного поширення помилки повторюється для кожного із навчальних зразків, отже сила зв’язків між нейронами ітерація за ітерацією наближається до оптимального набору значень.

Одним із недоліків градієнтних методів є можливість стагнації у локальному мінімумі. Критичним параметром тут є коефіцієнт антиградієнту, або швидкості навчання. Деякі варіанти методу зворотного поширення помилки використовують адаптивну зміну величини кроку.

2.4 Математична модель згорткових нейронних мереж

Згорткові нейронні мережі є прикладом нейронних мереж прямого поширення, їхньою особливістю є декілька типів шарів, які зазвичай дозволяють як виконувати безпосередньо класифікацію даних, так і моделювати інформативні для розв’язання задачі ознаки [108].

Наприклад, у шарах згортки виконується перетворення вхідного шару, який називають мапою ознак, у вихідний, який є, відповідно, входом для наступного шару нейронів. Нехай у l-му шарі згортки виконується лінійне перетворення двомірних вхідних даних, які представлені матрицею X^{l}_{MxN} у вихідну матрицю $Y^{l}_{\grave{M}x\grave{N}}$ ($\grave{M} \leqslant M, \grave{N} \leqslant N$), за допомогою вагової матриці W_{dxd} $(d<M, N)$ [31]:

$$y^{l}_{\grave{i},\grave{j}} = \sum_{a,b=0}^{d-1} W_{a,b}\, x^{l}_{i+a,\, j+b}\,. \tag{2.20}$$

Матриця $W_{d\times d}$ – це ядро згортки, а її значення обчислюємо алгоритмом навчання нейронної мережі. Зазвичай, це варіанти методу градієнтного спуску такі, як пакетний або стохастичний градієнтний спуск. Фактично, ядро згортки – це вікно перетворень, яке переміщується мапою ознак, обчислюючи результуючі значення. Коефіцієнти ядра згортки можна застосовувати декілька разів до різних областей мапи ознак. Також, операція згортки зумовлює те, що кожне значення на виході згорткового шару залежить тільки від декількох вхідних значень, водночас у повнозв'язній нейронній мережі кожне значення залежить від всіх входів. Отже, операцію згортки можна інтерпретувати як виділення певних локальних ознак вхідних даних. Чим більше в нейронній мережі шарів згортки, тим більше локальних ознак може бути виділено [31, 84, 108, 171].

Роботу шару згортки проілюстровано на рисунку 2.14 [73].

Різні режими роботи визначають, чи може сковзне вікно ядра згортки виходити за межі матриці ознак, наприклад у режимі 1 (див. рис. 2.14) вихідний фільтр мапи ознак має меншу розмірність, ніж початкова матриця.

У роботі після обчислення згортки до мапи ознак застосовуємо нелінійну функцію активації. Це може бути одна з функцій, які наведено в таблиці 2.3: логістична функція, гіперболічний тангенс, ReLU тощо.

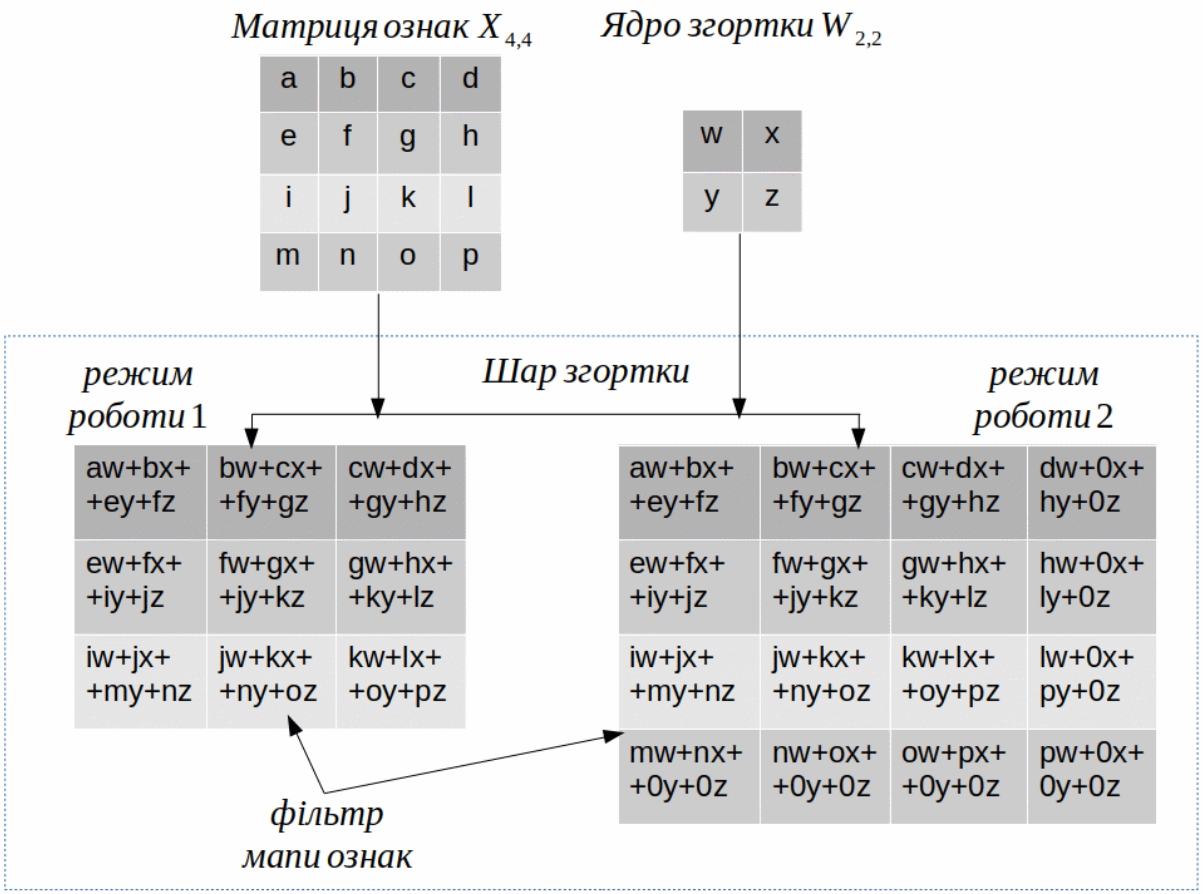


Рисунок 2.14 – Робота шару згортки

Зрештою, результатом роботи шарів згортки, що застосовуються до спектрограм аудіоданих, є множина фільтрів, яку можна інтерпретувати як різні представлення вихідного зображення мапи ознак. Отже, для безпосередньо класифікації, яка виконується у повнозв'язних шарах, формується досить великий набір абстрактних ознак, що описують початковий звук з різних точок зору.

Шари субдискретизації виконують зменшення розмірності мапи ознак, яка надходить на вхід з попереднього шару згортки та після застосування функції активації. Зазвичай, це перетворення виконують шляхом проходження по мапі ознак вікном, як це роблять на етапі згортки, однак при субдискретизації обирають максимальне (MaxPooling) або середнє (AveragePooling) значення серед тих, що попали у вікно. Цей процес можна

інтерпретувати як узагальнення ознак та, як наслідок, зменшення мапи ознак [108] (див рис. 2.15).

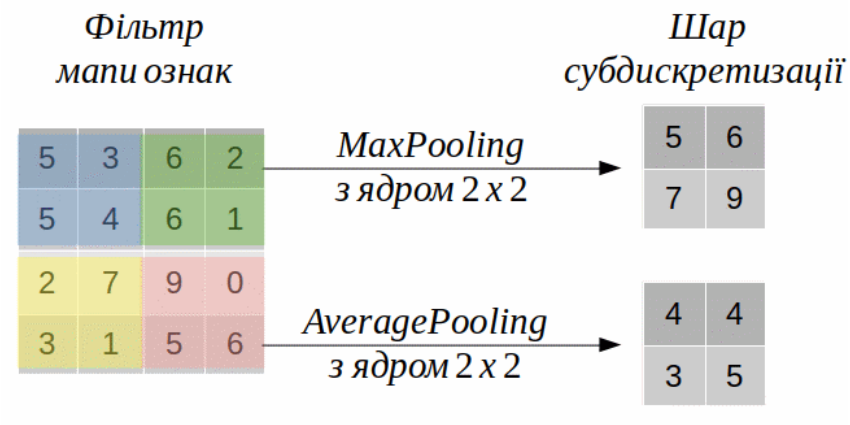


Рисунок 2.15 – Шар субдискретизації

Роботу шарів згортки та субдискретизації проілюстровано на рисунках 2.16 – 2.18. На рисунку 2.16 зображено приклад спектрограми звуку з набору ESC [131]. Далі, спектрограми оригінальних звуків надходять на вхід згорткової мережі. Після проходження блоків шарів згортки та субдискретизації отримують деякі абстрактні набори ознак, які називають фільтрами.

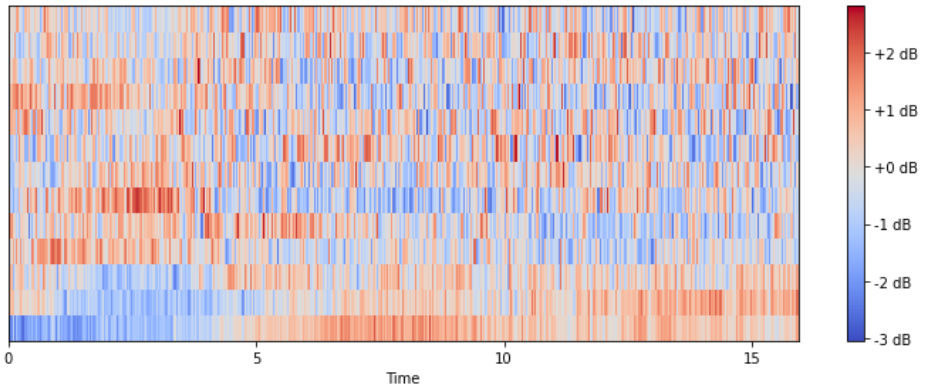


Рисунок 2.16 – Оригінальна спектрограма звуку

На рисунку 2.17 зображено два таких фільтри після першого блоку згортки-субдискретизації. Кожен фільтр – це частина вихідної спектрограми.

Оскільки після застосування операції субдискретизації розмір фільтрів зменшується, збільшення їх кількості призводить до більш повного представлення вихідних даних.

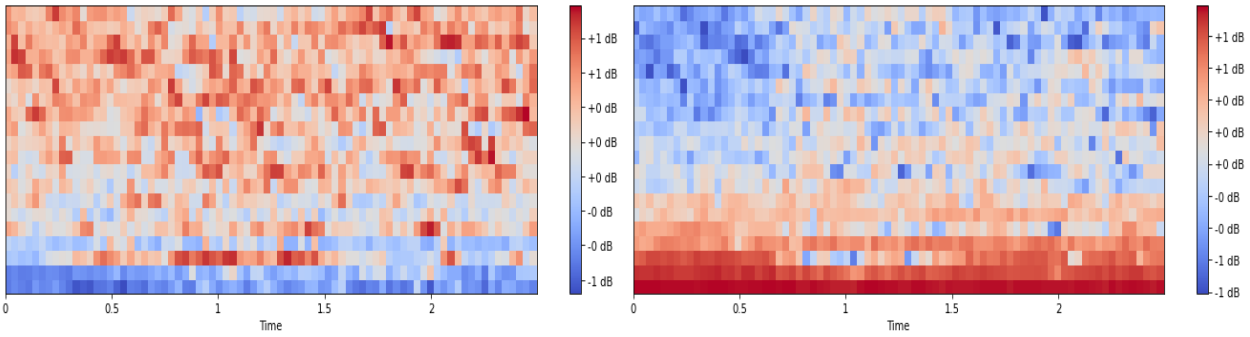


Рисунок 2.17 – Фільтри після 1-го блоку шарів

Наприклад, на рисунку 2.18 зображено фільтри після третього блоку шарів згортки, які є більш абстрактними.

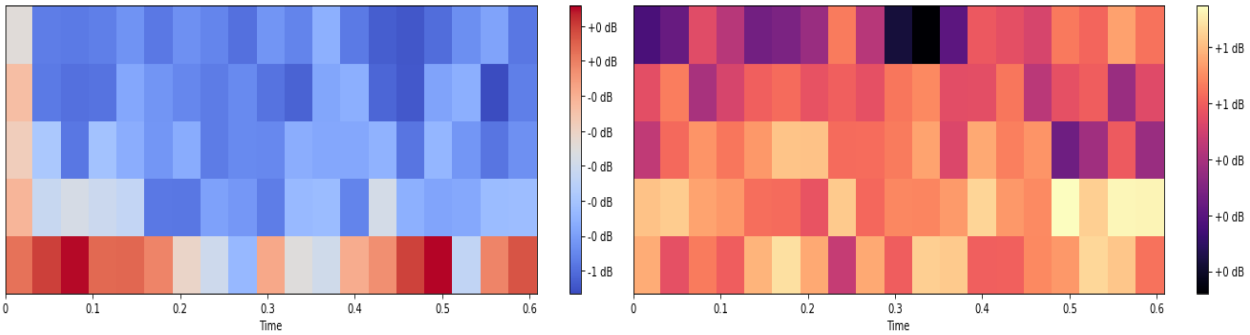


Рисунок 2.18 – Фільтри після 3-го блоку шарів

Останні шари згорткових нейронних мереж є багатошаровим перцептроном. Після декількох ітерацій послідовного виконання згортки та субдискретизації, на вхід перцептрона надходить набір досить абстрактних ознак, які використовують для навчання при класифікації [31, 84, 108, 171].

2.5 Математична модель гібридної нейронної мережі

Під гібридними нейронними мережами розуміють системи, які складаються з різних підсистем (типів нейромереж), об'єднаних для підвищення якості прогнозування. Зазвичай, передбачається не тільки механічне об'єднання декількох предикторів, але й механізм спільного навчання. Наприклад, в роботі [56] запропоновано підходи оптимізації структури деякої базової топології нейромережі. Базова структура варіюється та будується ансамбль нейромереж із подальшою оптимізацією генетичними алгоритмами.

У дисертаційній роботі запропоновано поєднання архітектур, що описано у підрозділах 2.3, 2.4: згорткових нейронних мереж та автокодувальників. Тобто, запропоновано гібридну топологію, що складається із шарів автокодувальника та згортки. На відміну від роботи [56] використано шари автокодувальника та ансамбль Snapshot, що дозволяє підвищити швидкість генерування ансамблю мереж.

Шари автокодувальника дозволяють виконувати розширення спостережень навчальної вибірки з метою балансування навчальної вибірки та генерації абстрактних інформативних ознак. Особливістю використання автокодувальника в цьому разі є попереднє поєднання окремих ознак звуку у більш складну структуру даних.

Додатковою перевагою застосування понижувального або знешумлювального автокодувальника до такого поєднання ознак є з одного боку зниження вимірності вихідних ознак, а з іншого – оптимізація множини ознак, які використовуються для класифікації.

Після шарів автокодувальника для класифікації застосовано згорткові нейронні мережі. Для ознак GFCC, MFCC, CQT, BFCC: X_{GFCC}, X_{MFCC}, X_{CQT}, X_{BFCC}, остаточні перетворення, враховуючи (2.17) будуть такими:

$$X_h = g(f(X_{GFCC} \oplus X_{MFCC} \oplus X_{CQT} \oplus X_{BFCC}, \theta), \theta), \quad (2.21)$$

де X_h – матриця ознак прихованого шару автокодувальника;

$X_{GFCC} \oplus X_{MFCC} \oplus X_{CQT} \oplus X_{BFCC}$ – операція конкатенації ознак.

До X_h в гібридній архітектурі застосовуються певна кількість шарів згортки та субдискретизації.

Для обґрунтування збіжності такої гібридної архітектури сформулюємо й доведемо теорему.

Теорема 2.1. Гібридна нейронна мережа із шарами автокодувальника та згортки знаходить класифікацію $C(U)$ для простору звукових сигналів U протягом скінченного проміжку часу.

Доведення. Нехай оператор A визначає гібридну нейромережу класифікації; множина звуків $U = \{u_1, \ldots, u_N\}$; $\Omega = \{\omega_1, .., \omega_k\}$ множина класів; $X = \{x_1, \ldots, x_N\}, x_i \in \mathbb{R}^{M \times K}$ – $M \times K$ -вимірний простір ознак.

Нехай оператор $F()$ переводить кожне спостереження у відповідну ознаку:

$$\forall u_i \in U \; \exists x_i \in X : x_i = F(u_i), \; i \in [1, N], \quad (2.22)$$

тоді, дію шарів $AutoEnc(U)$ автокодувальника нейромережі можна представити таким оператором:

$$\forall x_i \in X \; \exists \widetilde{x}_i \in \widetilde{X} : \widetilde{x}_i = AutoEnc(x_i), \; i \in [1, N], \quad (2.23)$$

де $\widetilde{X} = \{\widetilde{x}_1, \ldots, \widetilde{x}_N\}$ простір ознак після застосування автокодувальника, у загальному випадку $\widetilde{x}_i \in \mathbb{R}^{M \times K}$.

Дію шарів згортки можна опишемо таким оператором $Conv(U)$:

$$\forall \widetilde{x}_i \in \widetilde{X}\ \exists z_i \in Z : z_i = Conv(\widetilde{x}_i),\ i \in [1, N], \tag{2.24}$$

де $Z = \{z_1, \ldots, z_N\}$ в загальному випадку простір векторів фільтрів згорткового шару, $z_i \in \mathbb{R}^M$.

Тоді, оператор гібридної нейронної мережі можна представити як таку послідовність операторів

$$A : U \to F(U) \to AutoEnc(U) \to Conv(U) \to C(U), \tag{2.25}$$

де $C(U)$ – шари повнозв'язних шарів перцептрону, які генерують розподілення за класами.

Вважається, що оператори дії шарів визначають як дію одного типового шару, так і декількох.

Оскільки результатом послідовного виконання шарів автокодувальника та/або шарів згортки є деяка множина ознак, яка є вхідною послідовністю для шарів перцептрону, то за основною теоремою Ф. Розенблатта про збіжність перцептрону [137] можна стверджувати, що класифікацію $C(U)$ буде знайдена протягом скінченного проміжку часу.

2.6 Висновки до розділу 2

У даному розділі запропоновано математичні моделі класифікаторів на базі нейронних мереж. Виконано математичну постановку задачі розпізнавання аудіосигналу, яка дозволяє застосовувати нейронні мережі для багатокласової класифікації без перетину класів. Наведено математичні моделі мереж прямого поширення сигналу, зокрема: згорткових нейронних мереж та автокодувальників.

Запропоновано математичну модель гібридної нейронної мережі розпізнавання аудіосигналів, яка включає в себе шари автокодувальника та згорткової мережі. Математичні моделі нейромереж та ознак звукових даних базуються на теорії розпізнавання та теорії обробки сигналів і логічно випливають з попередніх досліджень інших авторів та є формалізацією основних понять, які використано у подальших розділах.

Сформульовано та доведено теорему про збіжність гібридної нейронної мережі з шарами автокодувальника та згортки.

Основні результати другого розділу опубліковано у роботах [19-22].

РОЗДІЛ 3
РОЗВ'ЯЗАННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ ЗА ДОПОМОГОЮ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

Після визначення інформативних ознак суттєвим етапом розв'язання задачі розпізнавання є вибір та оптимізація алгоритму класифікації. У даному розділі розроблено нейромережеві моделі класифікації звукових даних на основі ознак, які розглянуто у розділі 2.1. Виконано дослідження впливу гіперпараметрів класифікаторів на точність прогнозування моделей.

Згорткові нейронні мережі є одним із основних методів класифікації даних з просторовою структурою (зображень спектрограм або темпограм звуку тощо). Оскільки на практиці можна застосовувати моделі з великою кількістю шарів та різноманітною структурою блоків згортки-субдискретизації актуальною задачею є оптимізація набору гіперпараметрів, що повністю задають модель класифікації.

Крім структури та кількості шарів згортки-субдискретизації у згорткових мережах можливе використання додаткових шарів, наприклад, шарів «уваги», які можуть підвищити точність класифікації за рахунок виділення найбільш інформативних областей даних з точки зору класифікації. Також можна застосовувати шар радіально-базисних функцій замість повнозв'язних шарів [1, 52]. У цьому випадку можна перейти вид задачі параметричної апроксимації результуючого вектору класів до непараметричної задачі, коли результат представляється комбінацією набору радіально-базисних функцій.

Не менш важливим є питання множини ознак, які використано як вихідні дані для класифікатора. У деяких випадках таку оптимізацію може проводити експерт-дослідник, адже на цьому етапі суттєвим є розуміння предметної області. В автоматизованих системах для генерації абстрактних ознак, які можна далі передавати класифікатору, використовують нейронні

мережі автокодувальники. Найпростіший варіант такої нейромережі – лінійний двошаровий автокодувальник, який генерує у середньому шарі абстрактні ознаки, аналогічні тим, що генеруються методом головних компонент. Ще одним застосуванням автокодувальників є синтез додаткових зразків даних у випадку незбалансованих навчальних вибірок.

Отже, перспективним є використання гібридних нейронних мереж з використанням згорткових архітектур та автокодувальників. Це дасть змогу виконувати попередню обробку даних та безпосередньо класифікацію у рамках однієї нейромережевої архітектури.

Отримані конфігурації нейронних мереж використано далі для розробки методів автоматичного налаштування гіперпараметрів у наступному розділі.

3.1 Згорткова модель класифікації

Згорткові нейронні мережі є базовою архітектурою при розв’язанні задач класифікації широкого діапазону вхідних даних. У залежності від специфіки вхідних даних до базової структури згорткових мереж можна додавати так звані шари «уваги», шари радіально-базисних функцій тощо. Додаткові шари можуть зумовити покращення точності класифікації.

Базову згорткову нейромережеву модель, яку можна застосовувати для класифікації звукових наборів даних Free Sond [81], ESC [131], UrbanSound8k [138], зображено на рисунку 3.1.

Серед гіперпараметрів, які підлягають оптимізації, деякі відносять до архітектури нейромережі: кількість пакетів згортки-субдискретизації, наявність та кількість додаткових шарів у пакетах, кількість фільтрів, розмірність матриць згортки та субдискретизації, кількість повнозв’язних шарів та нейронів у них. Інші гіперпараметри характеризують обчислення в

шарах: вид функцій активацій, метод початкової ініціалізації шарів, вид оптимізатора тощо.

Отже, вибір базової моделі класифікації можна робити на основі попередніх експертних оцінок про архітектуру нейромережевої моделі та структуру множини ознак, яку використано для класифікації.

Однією з основних задач нейронної мережі на цьому етапі є визначення ознак сигналів, які надходять на вхід нейронної мережі.

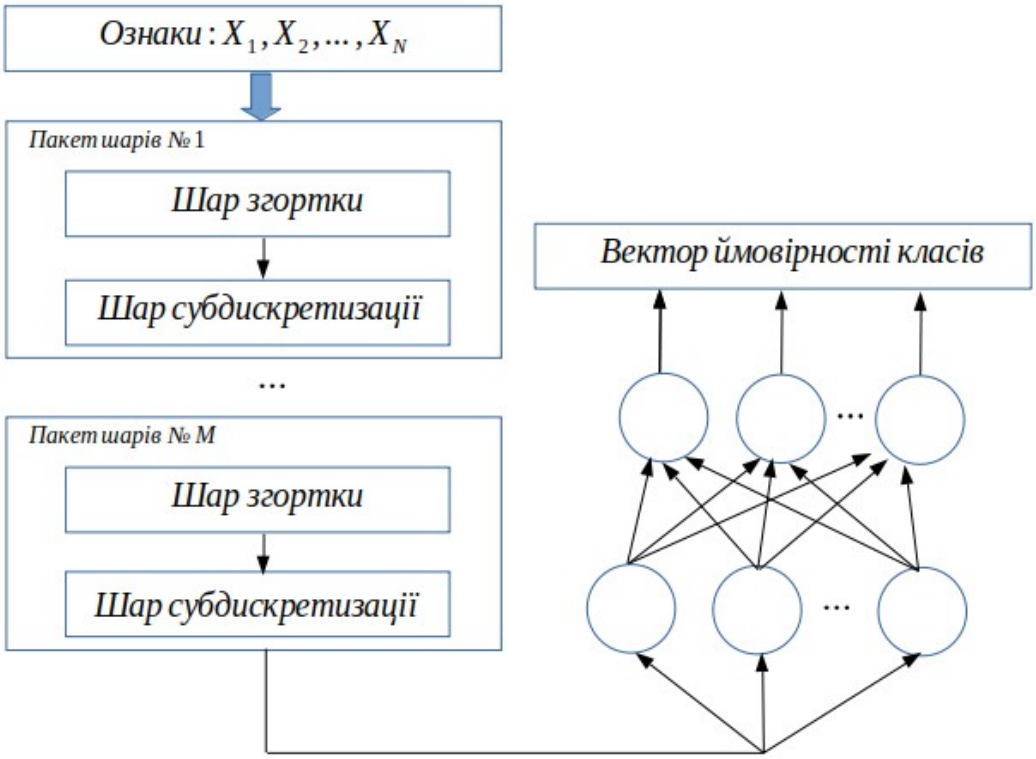


Рисунок 3.1 – Згорткова нейронна мережа

Розглянемо далі використання деяких базових архітектур для класифікації набору даних ESC-10 [131], який містить десять категорій звуків, які пронумеровано наступним чином: 0 – лай собаки, 1 – звук бензопили, 2 – годинник, 3 – тріск вогню, 4 – плач дитини, 5 – вертоліт, 6 – дощ, 7 – крик півня, 8 – морські хвилі, 9 – чхання.

Спектрограми звуків є найбільш компактними та інформативними ознаками. На рисунках 3.2–3.4 наводяться відповідні мел-частотні спектрограми необроблених звуків з кожного класу набору даних ESC-10.

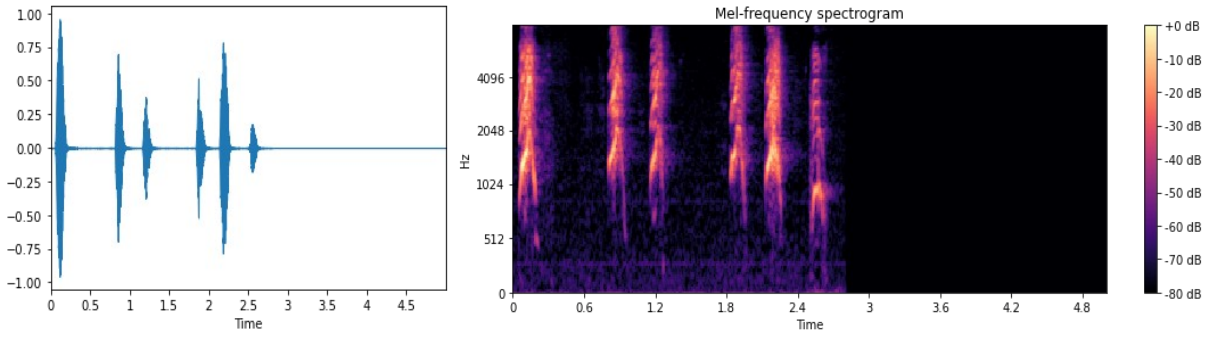


Рисунок 3.2 – Залежність амплітуда-час і мел-спектрограма лаю собаки

З рисунку 3.2 можна побачити, що значний час займає сигнал з низькою енергією (майже тиша), отже значна частина запису не несе інформації. Водночас, різні записи цього класу можуть мати довільну довжину інформативного сигналу, отже для таких звуків необхідно застосувати метод сегментації початкових записів на відрізки довжиною 0,005 – 1,5 секунди з відповідним маркуванням звуку та тиші.

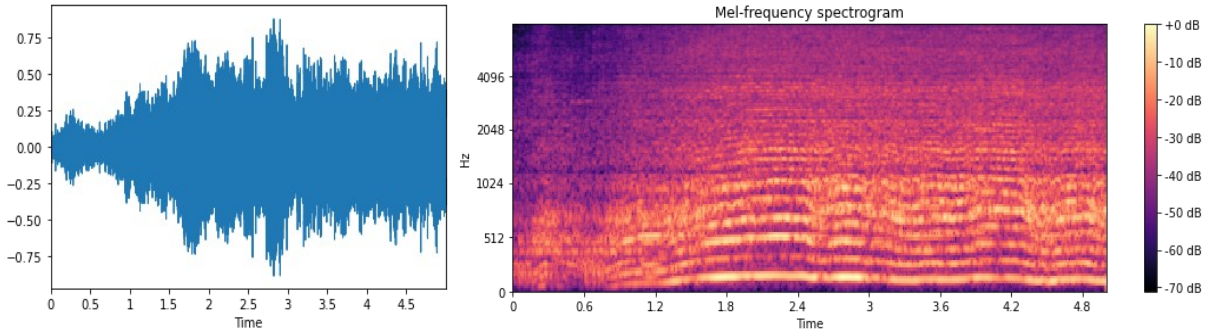


Рисунок 3.3 – Залежність амплітуда-час і мел-спектрограма бензопили

Приклад, зображений на рисунку 3.3 є характерним для цього класу звуків, оскільки на спектрограмі заповнено майже всю часову область. Отже, сегментація та додаткова розмітка сегментів таких звукових записів не призведе до зростання кількості спостережень класу тиші, що призводить до незбалансованої вибірки навчання. У роботі для розв’язання задачі балансування значень за класами запропоновано використання автокодувальників.

Більш складну структуру мають людські звуки, наприклад плач дитини (див. рис. 3.4). Інтенсивність на розподілення частот у записах цього класу більше варіюється порівняно з іншими прикладами звуків.

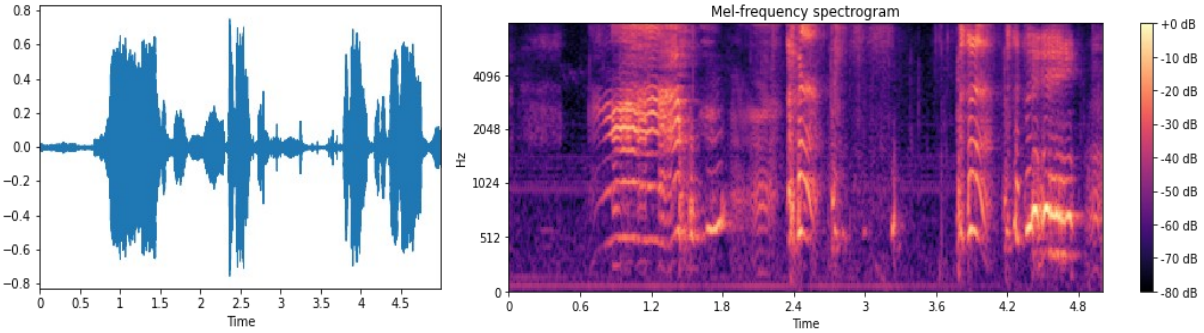


Рисунок 3.4 – Залежність амплітуда-час і мел-спектрограма плачу дитини

Схожими на людські звуки за структурою спектрограми є звуки тварин (див. рис. 3.5).

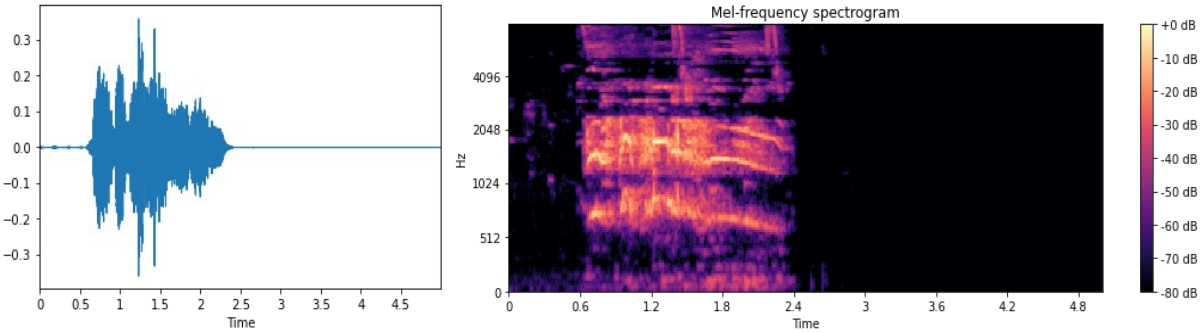


Рисунок 3.5 – Залежність амплітуда-час і мел-спектрограма крику півня

З рисунків 3.6, 3.7 видно, що наведені класи звуків набору даних ESC-10, наприклад, «дощ» та «морські хвилі» мають схожі спектрограми та розподіл частот.

Класи схожих за структурою спектрограм звуків прогнозовано можуть не дуже якісно розпізнаватися автоматизованою системою з вхідними ознаками тільки у вигляді спектрограм.

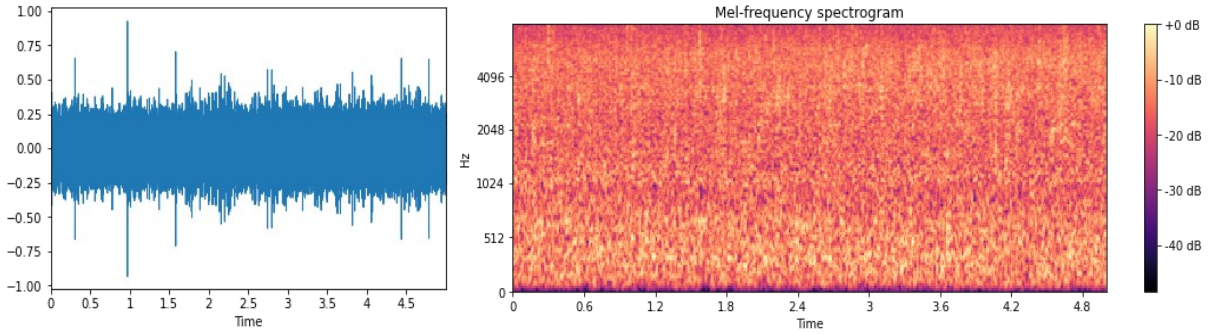


Рисунок 3.6 – Залежність амплітуда-час і мел-спектрограма звуку дощу

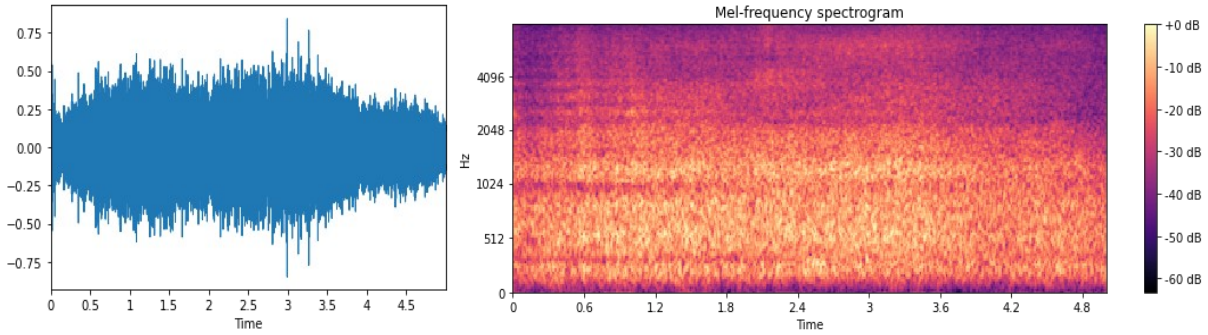


Рисунок 3.8 – Залежність амплітуда-час і мел-спектрограма морських хвиль

Для розширення кількості записів використано вікно довжиною у 5 мілісекунд. Отже, з’явився додатковий клас, який містить звук з низькою енергією (майже тиша). Водночас, таке перетворення дозволяє розширити кількість спостережень у кожному класі, що є перевагою при застосуванні нейронних мереж.

Результати обчислювальних експериментів із класичною згортковою мережею наведено далі у вигляді матриць розбіжностей 3.1–3.4. Досліджено декілька варіантів топології з різною кількістю пакетів шарів згортки-субдискретизації та фільтрів. Прийнято такі позначення: Conv – шар згортки; MaxPooling – шар субдискретизації з пошуком максимального значення у вікні шару; Dense – повнозв’язний шар.

Операція додавання у формулах архітектур нейронних мереж нижче означає, що сигнал передається послідовного від одного шару до іншого, а операція множення – послідовність однакових шарів.

Згорткова мережа №1 (ЗМ №1). Має таку структуру шарів: Conv(32)+Conv(64)+ MaxPooling+ 2xConv(64)+ MaxPooling+ 2xConv(128)+ MaxPooling+3xDense(256)+Dense(11).

Матриця розбіжностей моделі ЗМ №1 має такий вигляд:

$$\begin{pmatrix} 136 & 4 & 40 & 0 & 5 & 0 & 0 & 5 & 0 & 10 & 0 \\ 0 & 130 & 5 & 0 & 30 & 0 & 5 & 10 & 5 & 15 & 0 \\ 0 & 0 & 114 & 10 & 32 & 7 & 5 & 0 & 20 & 12 & 0 \\ 0 & 0 & 5 & 100 & 10 & 10 & 20 & 15 & 30 & 10 & 0 \\ 0 & 0 & 0 & 0 & 192 & 0 & 0 & 0 & 3 & 3 & 2 \\ 0 & 43 & 0 & 47 & 0 & 70 & 20 & 10 & 0 & 10 & 0 \\ 1 & 9 & 0 & 0 & 0 & 0 & 150 & 0 & 20 & 10 & 10 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 169 & 14 & 17 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 20 & 0 & 145 & 20 & 15 \\ 0 & 0 & 0 & 0 & 0 & 0 & 20 & 0 & 10 & 155 & 15 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 190 \end{pmatrix} \quad (3.1)$$

Згорткова мережа №2 (ЗМ №2). Має таку структуру шарів: Conv(24)+Conv(48)+ MaxPooling+ 2xConv(48)+ MaxPooling+ 2xConv(96)+ MaxPooling+3xDense(192)+Dense(11).

$$\begin{pmatrix} 143 & 3 & 34 & 0 & 5 & 0 & 0 & 5 & 0 & 10 & 0 \\ 0 & 157 & 8 & 0 & 10 & 0 & 5 & 10 & 5 & 5 & 0 \\ 0 & 0 & 110 & 14 & 32 & 7 & 5 & 0 & 20 & 12 & 0 \\ 0 & 0 & 5 & 90 & 10 & 20 & 20 & 15 & 30 & 10 & 0 \\ 0 & 0 & 0 & 0 & 192 & 0 & 0 & 0 & 3 & 3 & 2 \\ 0 & 43 & 0 & 57 & 0 & 65 & 20 & 10 & 0 & 5 & 0 \\ 1 & 9 & 0 & 0 & 0 & 0 & 145 & 0 & 20 & 15 & 10 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 169 & 14 & 17 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 173 & 10 & 17 \\ 0 & 0 & 0 & 0 & 3 & 0 & 18 & 0 & 10 & 162 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 7 & 193 \end{pmatrix} \quad (3.2)$$

Згорткова мережа №3 (ЗМ №3). Має таку структуру шарів: Conv(32)+Conv(64)+ MaxPooling+ 2xConv(64)+ MaxPooling+ 2xConv(128)+ MaxPooling+3xDense(256)+Dense(11).

$$\begin{pmatrix} 138 & 3 & 34 & 0 & 10 & 0 & 0 & 5 & 0 & 10 & 0 \\ 0 & 139 & 6 & 7 & 17 & 3 & 5 & 13 & 5 & 5 & 0 \\ 0 & 0 & 135 & 14 & 22 & 7 & 5 & 6 & 5 & 6 & 0 \\ 0 & 0 & 5 & 93 & 10 & 20 & 20 & 15 & 27 & 10 & 0 \\ 0 & 0 & 0 & 0 & 150 & 0 & 3 & 6 & 14 & 17 & 10 \\ 0 & 43 & 0 & 57 & 0 & 68 & 15 & 0 & 1 & 10 & 6 \\ 1 & 9 & 0 & 0 & 0 & 0 & 145 & 0 & 20 & 15 & 10 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 200 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 142 & 29 & 24 \\ 0 & 0 & 0 & 0 & 3 & 0 & 18 & 0 & 10 & 134 & 35 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 190 \end{pmatrix} \quad (3.3)$$

Згорткова мережа №4 (ЗМ №4). Має таку структуру шарів: Conv(32)+Conv(64)+ MaxPooling+ 2xConv(64)+ MaxPooling+ 2xConv(128)+ MaxPooling+4xDense(512)+Dense(11).

$$\begin{pmatrix} 156 & 11 & 16 & 0 & 2 & 0 & 0 & 5 & 0 & 10 & 0 \\ 0 & 139 & 6 & 7 & 17 & 3 & 5 & 13 & 5 & 5 & 0 \\ 0 & 0 & 135 & 14 & 22 & 7 & 5 & 6 & 5 & 6 & 0 \\ 0 & 0 & 5 & 115 & 10 & 10 & 18 & 15 & 27 & 0 & 0 \\ 0 & 0 & 0 & 0 & 160 & 0 & 3 & 6 & 14 & 7 & 10 \\ 0 & 43 & 0 & 50 & 0 & 56 & 17 & 7 & 1 & 20 & 6 \\ 1 & 9 & 0 & 0 & 0 & 0 & 115 & 0 & 30 & 25 & 20 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 200 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 4 & 0 & 127 & 36 & 28 \\ 0 & 0 & 0 & 0 & 3 & 0 & 18 & 0 & 6 & 140 & 33 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 200 \end{pmatrix} \quad (3.4)$$

Функція активації для всіх внутрішніх шарів зазначених моделей – Relu та функція SoftMax (див. табл. 2.3) для останнього шару Dense. Слід зазначити, що таку порівняно невисоку точність побудованих моделей можна пояснити незбалансованістю даних, що використано при навчанні та тестуванні.

Точність прогнозування в обчислювальних експериментах з наведеними архітектурами згорткових мереж наведено в таблиці 3.1. Точність у цьому разі визначається за формулою (2.15).

Таблиця 3.1 – Точність за класами згорткових мереж

№ класу	0	1	2	3	4	5	6	7	8	9	10
ЗМ №1	0,81	0,66	0,62	0,56	0,84	0,48	0,68	0,83	0,63	0,63	0,87
ЗМ №2	0,83	0,76	0,62	0,50	0,85	0,43	0,66	0,82	0,70	0,71	0,90
ЗМ №3	0,81	0,70	0,71	0,50	0,71	0,46	0,71	0,90	0,67	0,50	0,83
ЗМ №4	0,88	0,69	0,74	0,61	0,76	0,39	0,58	0,87	0,60	0,62	0,85

Одним із методів підвищення точності прогнозування для просторових даних є застосування шарів «уваги». Наприклад, у роботі [173] шар «уваги» використовується в поєднанні з рекурентними нейронними мережами.

Архітектуру шару «уваги» зображено на рисунку 3.9.

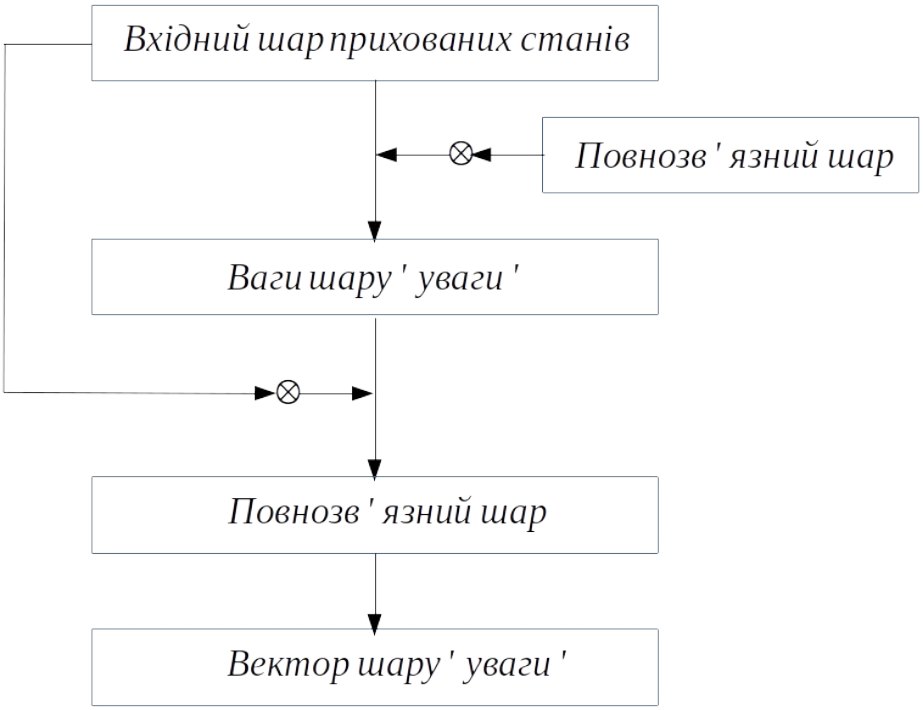


Рисунок 3.9 – Блок шарів «уваги»

Головною метою імплементації цього компонента є визначення тих областей у прихованому стані нейронної мережі, які найбільш суттєво впливають на результат класифікації. Під прихованим станом нейромережі розуміємо набір ваг будь-якого із внутрішніх шарів. Припустимо, що на вхід шару «уваги» надійшов результат обчислень після одного з повнозв'язних шарів. Отже, всі операції в блоці «уваги» виконано над числовими векторами, а $\otimes$ – визначає векторний добуток. Отже, за умови, що функцією активації є *Softmax* (2.18), вектор «уваги» v можна визначити такими рівняннями [173]:

$$\beta_t = \frac{e^{W * h_t}}{\sum_{i=1}^{T} e^{W * h_t}} \ , \ v = \sum_{t=1}^{T} \beta_t h_t \ , \tag{3.5}$$

де h_t – вектор прихованих станів повнозв'язного шару;

W – ваговий вектор шару «уваги»;

β_t – нормалізований вектор ознак.

На відміну від статті [173] в даній роботі використано комбіновані ознаки, отримані після застосування шару автокодувальника (див. рис. 3.11), а шар «уваги» застосовується після згорткових шарів. Також слід зазначити, що ваговий вектор «уваги» W формується при проходженні сигналу через повнозв'язний шар, отже, гіперпараметром є кількість нейронів всередині шару «уваги».

У публікаціях [113, 142, 173] використовуються шари «уваги» у рекурентних згорткових топологіях для класифікації звуків, проте кількість та параметри цих шарів є статичними, що удосконалюється у даній роботі, де оптимізацію структури шарів виконано генетичним алгоритмом.

Результати обчислювальних експериментів наведено у формулах 3.6-3.19. У формулах структури нейронних мереж, що наведено нижче, шар «уваги» позначено як Attention( ).

Згорткова модель з шаром уваги №1 (ЗМШУ №1, див. рис. 3.17). Має таку структуру шарів: Conv(32)+Conv(64)+ MaxPooling+ 2xConv(64)+ MaxPooling+ 2xConv(128)+ MaxPooling+3xDense(256)+Attention()+Dense(11).

$$\begin{pmatrix} 150 & 10 & 10 & 6 & 2 & 0 & 3 & 5 & 2 & 10 & 2 \\ 0 & 173 & 0 & 7 & 7 & 3 & 5 & 0 & 0 & 5 & 0 \\ 0 & 0 & 120 & 19 & 22 & 7 & 5 & 6 & 5 & 6 & 10 \\ 0 & 0 & 25 & 129 & 0 & 10 & 8 & 5 & 23 & 0 & 0 \\ 0 & 0 & 0 & 0 & 186 & 0 & 3 & 3 & 1 & 3 & 4 \\ 0 & 41 & 0 & 50 & 2 & 95 & 7 & 0 & 1 & 0 & 4 \\ 1 & 9 & 0 & 0 & 0 & 0 & 135 & 5 & 25 & 25 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 190 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 2 & 0 & 170 & 8 & 15 \\ 2 & 0 & 2 & 0 & 3 & 2 & 18 & 2 & 6 & 137 & 28 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 200 \end{pmatrix} \quad (3.6)$$

Згорткова модель з шаром уваги №2 (ЗМШУ №2, див. рис. 3.18). Має таку структуру шарів: 2xConv(32)+Conv(64)+ MaxPooling+ 3xConv(128)+ MaxPooling+ 3xConv(256)+ MaxPooling+3xDense(256)+ Attention()+ Dense(11).

$$\begin{pmatrix} 155 & 10 & 11 & 0 & 2 & 0 & 3 & 4 & 3 & 10 & 2 \\ 0 & 172 & 1 & 7 & 8 & 3 & 4 & 0 & 0 & 5 & 0 \\ 0 & 0 & 117 & 19 & 23 & 7 & 6 & 8 & 1 & 10 & 9 \\ 0 & 0 & 25 & 123 & 1 & 10 & 8 & 5 & 24 & 0 & 4 \\ 0 & 0 & 0 & 0 & 193 & 0 & 3 & 0 & 1 & 3 & 0 \\ 0 & 40 & 0 & 50 & 5 & 93 & 7 & 0 & 1 & 0 & 4 \\ 1 & 6 & 0 & 0 & 0 & 0 & 143 & 0 & 25 & 25 & 0 \\ 1 & 1 & 1 & 1 & 0 & 4 & 1 & 177 & 4 & 4 & 6 \\ 0 & 0 & 0 & 0 & 5 & 0 & 2 & 0 & 170 & 8 & 15 \\ 2 & 0 & 2 & 0 & 3 & 2 & 0 & 0 & 6 & 185 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 200 \end{pmatrix} \quad (3.7)$$

Згорткова модель з шаром уваги №3 (ЗМШУ №3, див. рис. 3.19). Має таку структуру шарів: Conv(32)+Conv(64)+ MaxPooling+ 2xConv(64)+ MaxPooling+ 2xConv(128)+ MaxPooling+3xDense(256)+Attention()+Dense(11).

$$\begin{pmatrix} 160 & 2 & 11 & 10 & 2 & 3 & 3 & 4 & 3 & 0 & 2 \\ 0 & 143 & 11 & 15 & 10 & 3 & 4 & 3 & 4 & 5 & 2 \\ 0 & 1 & 140 & 12 & 24 & 2 & 1 & 8 & 3 & 0 & 9 \\ 0 & 13 & 25 & 98 & 1 & 10 & 8 & 5 & 24 & 10 & 6 \\ 0 & 0 & 0 & 0 & 165 & 0 & 3 & 0 & 11 & 18 & 3 \\ 0 & 40 & 0 & 50 & 18 & 80 & 7 & 0 & 1 & 0 & 4 \\ 1 & 6 & 0 & 0 & 0 & 0 & 143 & 0 & 25 & 25 & 0 \\ 1 & 1 & 1 & 1 & 0 & 4 & 1 & 177 & 4 & 4 & 6 \\ 0 & 0 & 0 & 0 & 5 & 0 & 2 & 0 & 170 & 8 & 15 \\ 7 & 0 & 2 & 0 & 3 & 2 & 0 & 0 & 26 & 160 & 0 \\ 20 & 0 & 0 & 0 & 15 & 0 & 0 & 0 & 0 & 0 & 165 \end{pmatrix} \quad (3.8)$$

Після застосування шару «уваги» формується вектор «уваги», який кодує суттєві для розпізнавання елементи вхідних ознак.

Точність прогнозування в обчислювальних експериментах з наведеними архітектурами згорткових мереж наведено в таблиці 3.2.

Таблиця 3.2 – Точність прогнозування за класами згорткових мереж з шарами «уваги»

№ класу	0	1	2	3	4	5	6	7	8	9	10
ЗМШУ 1	0,85	0,80	0,67	0,63	0,88	0,60	0,70	0,90	0,77	0,66	0,95
ЗМШУ 2	0,86	0,80	0,65	0,61	0,89	0,58	0,73	0,87	0,76	0,81	0,93
ЗМШУ 3	0,81	0,70	0,71	0,50	0,71	0,46	0,71	0,90	0,67	0,50	0,83
ЗМШУ 4	0,91	0,71	0,79	0,70	0,82	0,63	0,64	0,92	0,76	0,83	0,90

Результати обчислювальних експериментів із різною кількістю шарів та фільтрів свідчать про те, що задача підбору оптимального набору гіперпараметрів є досить складною, і потребує засобів автоматизації. Також, можна зробити висновок, що класи з ідентифікаторами 2, 3, 5, 9 мають найменшу точність у різних конфігураціях як класичної згорткової архітектури, так і з використанням шару «уваги». Отже, необхідно додатково оптимізувати множину ознак, що використано при навчанні системи класифікації.

3.2 Гібридна модель з використанням згорткових мереж та автокодувальників

Гібридні моделі нейронних мереж передбачають довільне поєднання декількох архітектур або типів мереж з метою підвищення точності прогнозування. Таке поєднання передбачає навчання гібридної топології як цілої системи. Недоліком такого підходу є досить велика кількість гіперпараметрів та необхідність структурної оптимізації кожної підсистеми. На практиці зустрічаються різноманітні гібридні архітектури, наприклад, нейро-нечіткі системи [4, 46], які передбачають різні способи поєднання предикторів у одну систему. Наприклад, обчислення можуть виконуватись спочатку однією мережею, а результати передаються далі як вхідні в іншу архітектуру. Для задач розпізнавання звуків перспективним є поєднання автокодувальників та згорткових мереж.

Загальне поняття нейронної мережі автокодувальника наведено в підрозділі 2.3 як один із прикладів нейромереж прямого поширення сигналу. Структура різних архітектур автокодувальників відповідає рисунку 2.13. Однак, в залежності від типів шарів, що використовуються, можна розрізняти автокодувальники з повнозв’язними шарами та згортковими.

При обробці двовимірних ознак аудіоданих, наприклад спектрограм або темпограм, у роботі використано згорткові автокодувальники (див. рис. 3.10). В залежності від практичних задач можна застосовувати довільну кількість шарів згортки, проте вона повинна дорівнювати кількості шарів оберненої згортки. Нейромережу, яку зображено на рисунку 3.10, використано для генерування нових зразків даних, наприклад спектрограм звуків з класу, який має порівняно мало спостережень.

У роботі використано підхід, при якому спектрограма звуку (див. рис. 3.10) складається з декількох типів спектрограм.

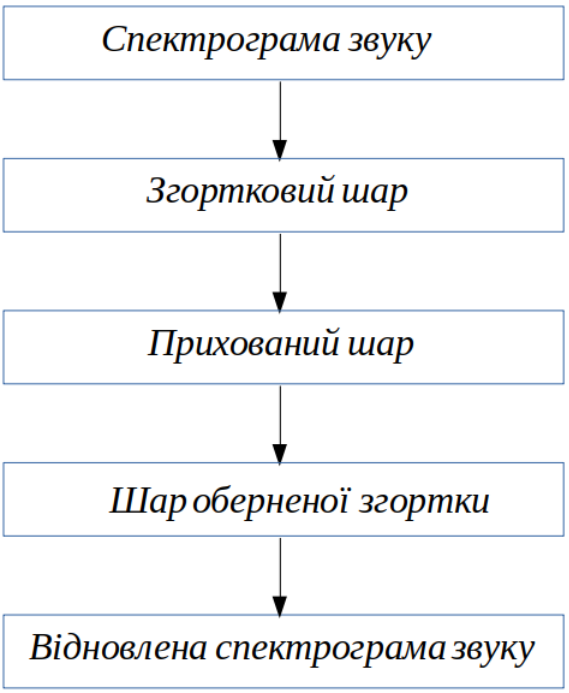


Рисунок 3.10 – Згортковий автокодувальник

Базову гібридну згорткову нейромережеву модель класифікації звуку зображено на рисунку 3.11. Автокодувальник використано для генерування шару абстрактних ознак, що дає змогу мінімізувати розмірність задачі класифікації. Отже, база даних 2D ознак є колекцією ознак для кожного звукового спостереження. Такий підхід можна використовувати для декількох вхідних ознак аудіоданих, наприклад, спектрограми, темпограми та хромаграми. Таким чином, прихований шар автокодувальника міститиме стиснене представлення, яке передається класифікатору. Запропонована схема може бути використана також для інших типів автокодувальників.

Отже, гібридна нейронна мережа складається з двох типів мереж, які оброблюють вхідний сигнал послідовно. Недоліком такої архітектури є порівняно велика кількість гіперпараметрів, які потребують налаштування. Для автоматизації цього процесу в даній роботі пропонується використання генетичного алгоритму, який описано у четвертому розділі.

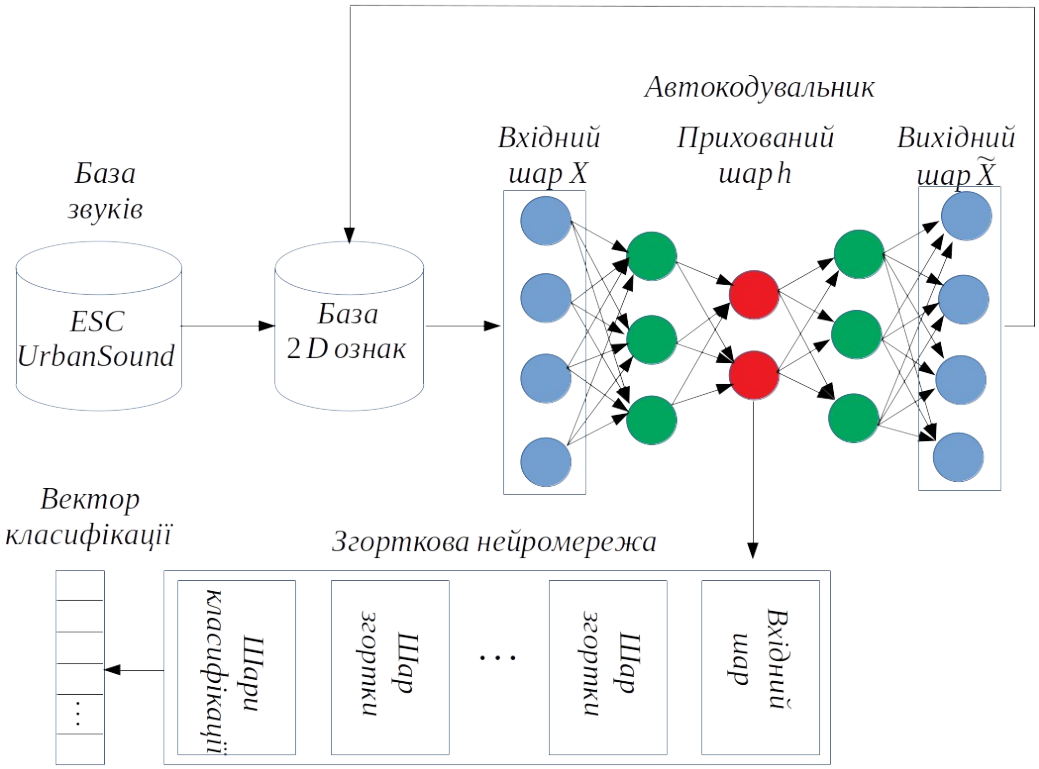


Рисунок 3.11 – Схема застосування автокодувальника

Одним із недоліків застосування автокодувальника для генерації нових зразків аудіоданих є часова складність цього підходу, оскільки для досягнення задовільної точності нейромережі необхідно використовувати декілька шарів. У роботі використано альтернативні підходи до збільшення кількості та балансування аудіозаписів за класами. Детально ці підходи у застосуванні до обробки звуку розглянуто у наступному підрозділі.

3.3 Методи балансування спостережень навчальної вибірки

Отже, після застосування методів сегментації звукових записів та додаткової розмітки отриманих сегментів можуть виникнути міноритарні класи в яких кількість спостережень деяких є порівняно малою, або мажоритарні – кількість спостережень відносно велика [79, 130]. При цьому

незбалансованість зростає для наборів даних, які містять звуки різного походження, наприклад, ESC та UrbanSound8k, або початкові дані вже не є збалансованими. Застосування класифікаторів, зазвичай, потребує балансування навчальної вибірки за кількістю спостережень, тобто додаткової генерації або вилучення деяких записів з метою отримання рівної кількості спостережень у кожному класі. Під час цього процесу виникають задачі визначення звуку, видалення якого не вплине на результат класифікації, а також дослідження методів генерації нових звуків. Загальний огляд таких підходів та посилання на інші роботи можна знайти в публікаціях [79, 130].

Одним із найпростіших підходів є випадкове видалення спостережень мажоритарного класу (англ. Random Undersampling). Для цього визначається кількість мажоритарних спостережень K_{RS} для випадкового видалення. Недоліки цього методу пов'язані з можливістю видалити суттєві для заданого розподілення звуки, що також може призвести до зменшення точності прогнозування [130].

Інші методи засновано на визначенні відстані між спостереженнями та видаленні схожих звуків.

При визначенні зв'яків Томека (англ. Tomek Links) визначаються записи, які можуть бути видалені. Нехай два спостереження X_i, X_j належать різним класам, $d(X_i, X_j)$ – відстань між спостереженнями. Оскільки на етапі попередньої обробки X_i, X_j є деякими спектрограмами, використовується відстань Евкліда для зображень. Пара X_i, X_j називається зв'язком Томека, якщо не знайдеться жодного спостереження X_l, для якого виконується [79]:

$$d(X_i, X_l) < d(X_i, X_j),\ d(X_j, X_l) < d(X_i, X_j). \qquad (3.9)$$

Отже, всі мажоритарні дані, які мають зв'язки Томека, видаляються. Очевидно, що видалення близьких у цьому сенсі спостережень не вплине на результат класифікації.

Іншим підходом є ущільнене правило найближчого сусіда (англ. Condensed Nearest Neighbor Rule). У цьому разі з вихідної навчальної вибірки формується додаткова шляхом випадкового вибору довільної кількості міноритарних звуків та меншої кількості даних з мажоритарних класів. Після чого кожна точка даних навчальної вибірки класифікується методом одного найближчого сусіда. Цей підхід дає змогу сформувати нову вибірку меншого розміру, проте збалансовану за кількістю спостережень [130].

Основна мета методів зменшення вибірки – видалити ті спостереження, які не вплинуть на результат класифікації.

При розширенні множини спостережень шляхом збільшення точок даних міноритарних класів у найпростішому випадку може використовуватись довільне дублювання спостережень, однак такий підхід може призвести до зменшення точності класифікації.

Метод SMOTE (англ. Synthetic Minority Oversampling Technique) використовує метод *k*-найближчих сусідів для пошуку схожих за метрикою відстані звуків міноритарного класу, потім обчислюється вектор різниці між спостереженням. Новим синтетичним спостереженням буде добуток одного зі спостережень міноритарного класу на вектор різниці та деякий вектор ваг [79, 130]. Такий підхід дозволяє отримати синтетичні звуки, які відрізняються від оригінальних спостережень, що зменшує ймовірність перенавчання класифікаторів .

Отже, розглянуті методи застосовано для розв’язання задачі балансування вибірки тренування з метою підвищення точності прогнозування. Подальше розширення набору даних може відбуватися засобами автокодувальників.

Описані вище методи в роботі використано для генерування додаткових звуків та реалізовано у рамках інструментальної системи класифікації, що дозволяє комбінувати їх застосування на етапі попередньої обробки даних.

3.4 Ансамблеве навчання

У даній роботі запропоновано використання методу Snapshot [93] для ансамблювання нейронних мереж. Особливістю цього підходу є варіювання коефіцієнту швидкості навчання відповідно до такої косинусної функції:

$$\alpha(t)=\frac{\alpha_0}{2}\left(\cos\left(\frac{\pi mod\left(t-1,\lceil T/M\rceil\right)}{\lceil T/M\rceil}\right)+1\right), \tag{3.9}$$

де α_0 – початкове значення коефіцієнта швидкості навчання;

t – ітерація навчання;

T – загальна кількість ітерацій навчання;

M – кількість циклів зміни коефіцієнту швидкості навчання, та, відповідно, кількість моделей, що згенеровано алгоритмом.

Згідно з [93] такий циклічний режим зміни коефіцієнту α (див. рис. 3.12) є ефективним при навчанні нейронних мереж, зокрема, згорткових. Крім того, з точки зору ансамблевого навчання, різні моделі, які обчислено на кожному циклі зміни коефіцієнту навчання, мають різні характеристики та навчаються на різних зразках даних.

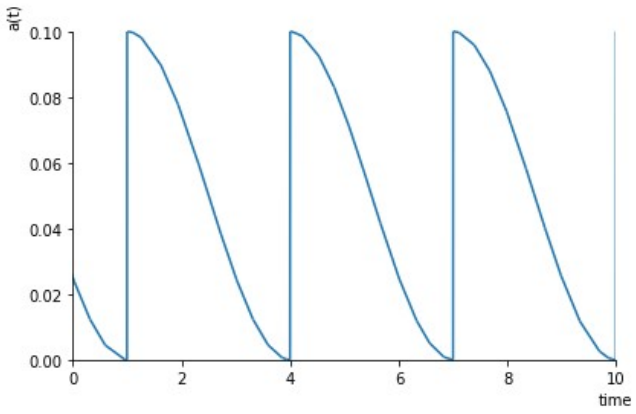
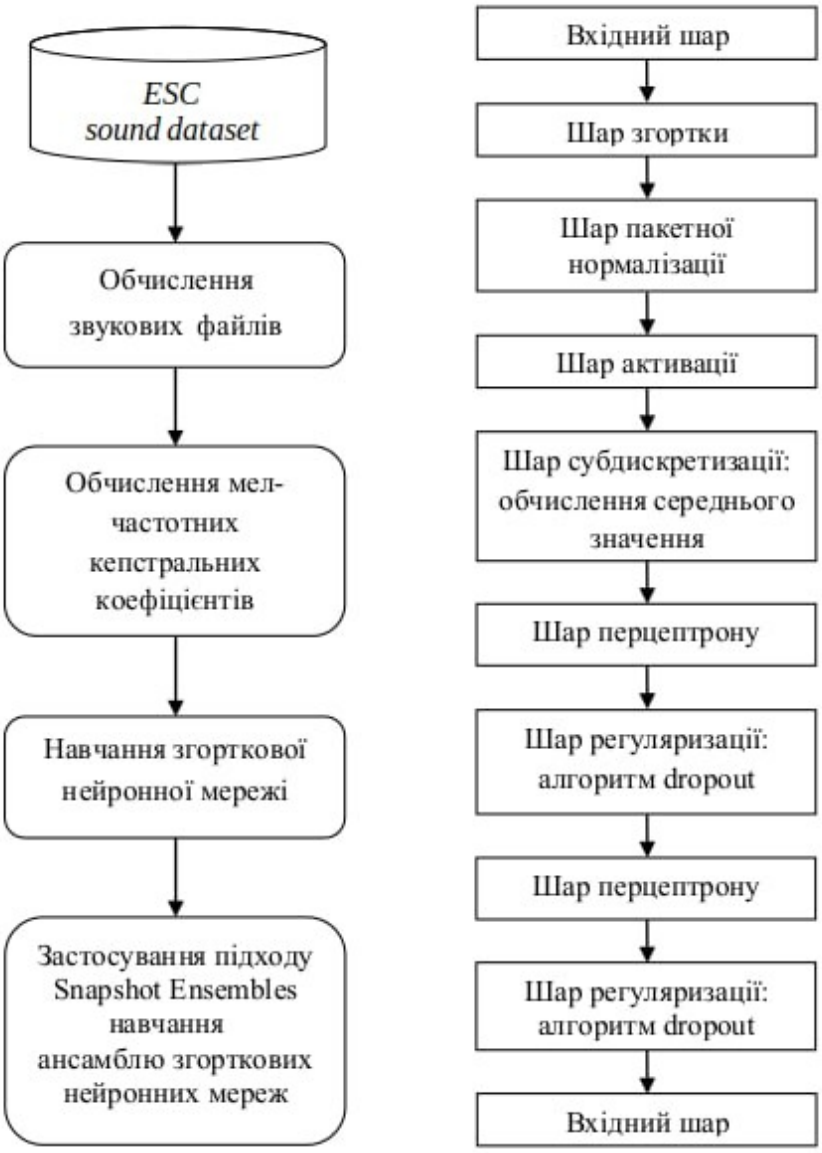


Рисунок 3.12 – Функція залежності коефіцієнту швидкості навчання

Отже, виконано вимоги, що зазвичай висувають до класифікаторів ансамблю: класифікатори розрізнено за алгоритмом або характеристиками; для навчання використано різні зразки даних.



а) б)

Рисунок 3.13 – Схема ансамблевого навчання

Основною перевагою використання даного алгоритму ансамблювання мереж є його швидкість, оскільки загальний час навчання ансамблю дорівнює часові навчання однієї мережі. Елементи ж самого ансамблю формуються зі збережених матриць станів цієї мережі.

Для тестування запропонованих підходів у цьому розділі використано дані ESC[131]. Схему застосування нейромережевих моделей та структуру мережі за шарами зображено на рисунку 3.13, а діаграму точності функції втрат отриманого класифікатору наведено на рисунку 3.14.

У формулі (3.10) наведено матрицю розбіжностей побудованого ансамблю класифікаторів моделі ЗМШУ №4. Зведені результати обчислювальних експериментів наведено в таблиці 3.3.

$$\begin{pmatrix} 194 & 0 & 0 & 1 & 0 & 0 & 2 & 1 & 0 & 0 & 2 \\ 0 & 160 & 11 & 10 & 2 & 3 & 0 & 0 & 4 & 5 & 5 \\ 0 & 8 & 172 & 6 & 1 & 2 & 1 & 2 & 3 & 0 & 5 \\ 0 & 10 & 21 & 156 & 1 & 0 & 6 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 165 & 0 & 3 & 0 & 11 & 18 & 3 \\ 0 & 37 & 0 & 44 & 18 & 96 & 0 & 0 & 1 & 0 & 4 \\ 1 & 6 & 0 & 0 & 0 & 0 & 143 & 0 & 25 & 25 & 0 \\ 1 & 1 & 1 & 1 & 0 & 4 & 1 & 180 & 4 & 4 & 3 \\ 0 & 0 & 0 & 0 & 5 & 0 & 2 & 0 & 179 & 8 & 6 \\ 9 & 0 & 2 & 0 & 3 & 0 & 0 & 0 & 16 & 170 & 0 \\ 0 & 0 & 0 & 5 & 3 & 0 & 0 & 0 & 0 & 0 & 192 \end{pmatrix} \quad (3.10)$$

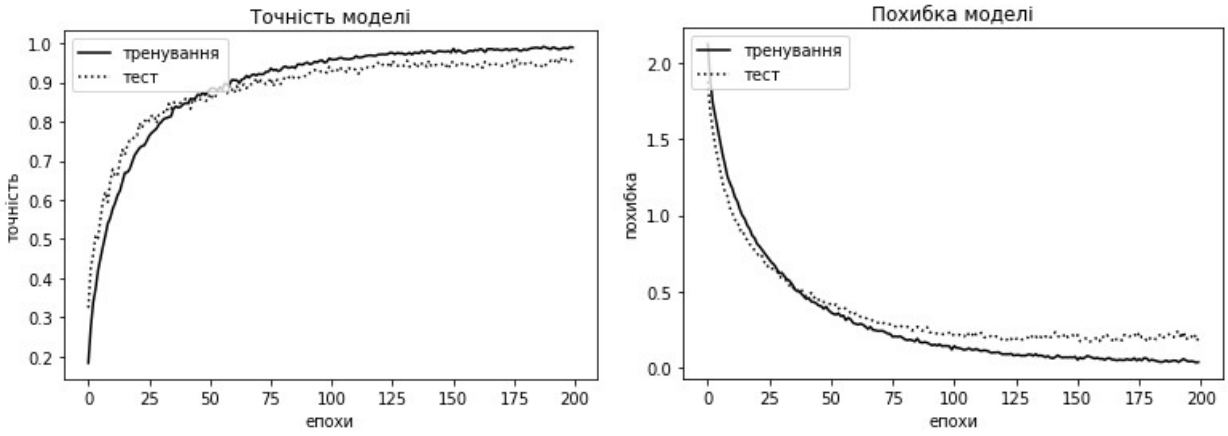


Рисунок 3.14 – Точність та похибка прогнозування ансамблю моделей

Значення матриці розбіжностей свідчать про задовільну точність класифікації. Спрогнозовані значення класів з шостого по восьмий повністю відповідають фактичними класами, які розмічені для даних звукових файлів. Результати обчислювальних експериментів наведено в таблиці 3.3.

Таблиця 3.3 – Точність за класами ансамблю згорткових мереж з шарами «уваги»

№ класу	0	1	2	3	4	5	6	7	8	9	10
ЗМШУ 1	0,89	0,85	0,70	0,68	0,90	0,66	0,74	0,90	0,81	0,68	0,94
ЗМШУ 2	0,88	0,84	0,63	0,68	0,91	0,60	0,78	0,92	0,80	0,84	0,93
ЗМШУ 3	0,86	0,78	0,75	0,61	0,76	0,53	0,76	0,93	0,69	0,57	0,87
ЗМШУ 4	0,94	0,76	0,83	0,76	0,86	0,67	0,69	0,94	0,77	0,87	0,92

Застосування ансамблевого навчання зумовило підвищення якості класифікації, що свідчить про можливість використання наведених архітектур нейронних мереж у подальшому як початкових топологій для генетичних алгоритмів.

3.5 Висновки до розділу 3

Отже, у третьому розділі розроблено нейромережеві моделі класифікації звукових даних на основі ознак, які розглянуто у другому розділі.

Проведено обчислювальні експерименти з базовими архітектурами нейронних мереж на відкритому наборі звукових даних ESC, які буде використано у четвертому розділі для ініціалізації початкової популяції генетичної оптимізації.

Досліджено вплив гіперпараметрів класифікаторів на точність прогнозування моделей. Наведно декілька варіантів топології з різною кількістю пакетів шарів згортки-субдискретизації та фільтрів. Проілюстровано точність прогнозування нейромережевих моделей за класами. Додання шарів «уваги» зумовило підвищення точності класифікації моделі. Для оптимізації множини ознак звукових даних, тобто зменшення його розмірності, в нейромережі використано шари автокодувальника.

Ефективність класифікаторів підвищено за рахунок використання методу ансамблювання Snapshot, що дало змогу прискорити час тренування ансамблю. Наведено результати обчислювальних експериментів з різною кількістю шарів та фільтрів.

Отримали подальший розвиток нейромережеві моделі класифікації звукових даних у частині розробки гібридних архітектур та оптимізації множини ознак.

Основні результати третього розділу опубліковано у роботах [16, 17, 20, 22, 24, 47, 104].

РОЗДІЛ 4
НЕЙРОЕВОЛЮЦІЙНА ГІБРИДНА СИСТЕМА КЛАСИФІКАЦІЇ

При використанні гібридних систем класифікації, що включають декілька типів нейронних мереж, наприклад, автокодувальників та згорткових, збільшується кількість гіперпараметрів, які підлягають налаштуванню. У даному розділі генетичні алгоритми використано для оптимізації множини структурних параметрів, які впливають на точність прогнозування побудованих класифікаторів.

Серед гіперпараметрів, які підлягають оптимізації, деякі відносять до архітектури нейромережі: розмір пакету навчання, кількість пакетів згортки-субдискретизації, наявність та кількість додаткових шарів в пакетах, кількість фільтрів, розмірність матриць згортки та субдискретизації, кількість повнозв'язних шарів та нейронів у них. Інші гіперпараметри характеризують обчислення у шарах: вид функцій активацій, метод початкової ініціалізації шарів, вид оптимізатора, швидкість навчання, кількість епох навчання тощо.

При використання методу ансамблювання Shapshot [93] суттєвим для точності результуючого класифікатора є початкове значення коефіцієнту швидкості навчання α_0 (3.1), кількість моделей ансамблю.

Отже, актуальною задачею є розробка методів спільної оптимізації множини гіперпараметрів нейронних мереж, що входять до ансамблю класифікаторів та параметрів самого ансамблю. Для цього використовують еволюційні методи оптимізації, наприклад, генетичні алгоритми, генетичне програмування, еволюційні стратегії, метод рою часток, алгоритм імітації відпалу та ін. У даному розділі розглянуто застосування саме генетичних алгоритмів до оптимізації нейромереж та їх ансамблів. Описано основні генетичні оператори та способи їх застосування. Наведено результати

обчислювальних експериментів із використанням набору аудіозаписів UrbanSound8k [138].

4.1 Застосування генетичних алгоритмів у оптимізації нейронних мереж та ансамблів

Загальну схему застосування генетичних алгоритмів до нейронних мереж викладено у підрозділі 1.3, а детальну теорію з цього приводу можна знайти в монографіях [35, 128]. Важливим етапом у задачі структурної оптимізації гібридної системи класифікації та побудованих ансамблів нейромереж є визначення деякої базової множини розв'язків X, в якій буде здійснено пошук оптимального розв'язку. Такі скінченні непорожні множини X називають популяціями [14], а їх елементи – хромосомами. Отже, задачею еволюційної оптимізації є пошук оптимальної хромосоми, яка кодує певну архітектуру та параметри системи класифікації, що розроблюється.

Суттєвим для подальшої роботи генетичних алгоритмів є вибір способу кодування параметрів системи, а також визначення генетичних операторів та цільової функції.

Для кодування гіперпараметрів системи класифікації використовують числові гомологічні хромосоми. Оскільки областю значень є певна скінченна множина цілих чисел або строк, які мають сенс для архітектури нейронних мереж, оптимальною структурою даних для збереження всіх параметрів є словник. Приклад такого словника наведено в таблиці 4.1, де використано такі позначення: розмір пакету навчання (batch_size), тип шару нейронів (layer_type), кількість фільтрів шарів згортки (units_filters), розмірність матриць згортки та субдискретизації (kernel_size), вид функцій активацій (activation), метод початкової ініціалізації шарів (init), тип оптимізатора (optimizer), швидкість навчання (learning_rate), кількість епох навчання

(epochs), тип регуляризатора мережі (regular), кількість мереж в ансамблі (ensemble_count). Текстові значення гіперпараметрів optimizer, layer_type, init та activation позначають відповідні символьні константи розробленої інструментальної системи.

Таблиця 4.1 – Області визначення гіперпараметрів

Гіперпараметр	Область визначення у форматі {код: значення}
batch_size	{0: 8, 1: 16, 2: 32, 3: 64, 4: 128, 5: 256, 6: 512}
learning_rate	{0: 0.1, 1: 0.01, 2: 0.001, 3: 0.0001, 4: 0.00001}
epochs	{0: 10, 1: 50, 2: 100, 3: 200}
ensemble_count	{0: 10, 1: 20, 2: 50, 3: 100, 4: 200}
optimizer	{0: 'Adam', 1: 'Adadelta', 2: 'Adagrad', 3: 'Adamax', 4: 'Nadam'}
layer_type	{0: 'Conv2D', 1: 'Dense', 2: 'MaxPolling2D', 3: 'AveragePooling2D', 4: 'Dropout', 5: 'Attention3D', 6: 'Conv2DTranspose'}
units_filters	{0: 8, 1: 1024}
stride	{0: 2, 1: 4, 2: 6}
kernel_size	{0: 2, 1: 4, 2: 6}
init	{0: 'RandomNormal', 1: 'RandomUniform', 2: 'Zeros', 3: 'Ones', 4: 'GlorotNormal', 5: 'GlorotUniform', 6: 'he_normal', 7: 'he_uniform', 8: 'lecun_normal', 9: 'lecun_uniform'}
regular	{0: 'l1', 1: 'l2', 2: 'l1_l2', 3: None}
activation	{0: 'relu', 1: 'sigmoid', 2: 'softmax', 3: 'softplus', 4: 'softsign', 5: 'tanh', 6: 'selu', 7: 'elu'}

Значення, що закодовано у структурі словника є такими, що мають сенс для певної архітектури згорткових мереж, згорткових автокодувальників та ансамблів нейромереж на основі підходу Snapshot [93]. Тобто, у словник закладено попередню експертну оцінку гіперпараметрів мереж, а множину

можливих значень обмежено особливостями програмної реалізації інструментальної системи.

Отже, якщо структура хромосоми буде фіксована, можна закодувати довільну архітектуру системи класифікації. Так, на рисунку 4.1 зображено початок коду хромосоми, яка кодує нейронну мережу з розміром пакету навчання у 64 спостереження, алгоритм оптимізації – метод Адам зі швидкістю навчання 0,01; кількість епох навчання – 100, а перший прихований шар – згортковий.

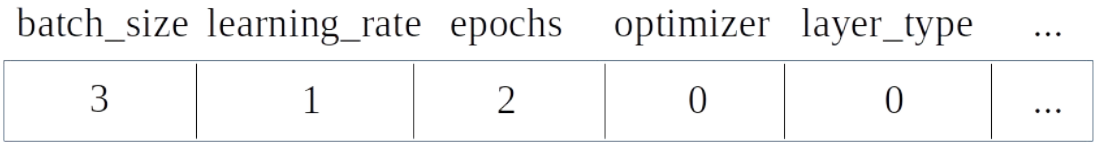


Рисунок 4.1 – Приклад кодування хромосоми

Цільову функцію визначено функціоналом похибки нейронної мережі. У випадку багатокласової класифікації використано перехресну ентропію (2.7).

Далі розглянемо визначення основних операторів генетичних алгоритмів: ініціалізації, селекції, кросовера та мутації.

При ініціалізації необхідно визначити початкову популяцію. Метод передбачає експертне задання деякої базової архітектури, яка має сенс для задачі класифікації аудіоданих. Після чого виконуємо випадкову генерацію необхідної кількості хромосом популяції, водночас забезпечуючи задане співвідношення архітектур нейромереж, які відповідають базовій хромосомі. Отже, можна варіювати «випадковість» початкової популяції та, за необхідністю, закладати в систему певні експертні оцінки.

Оператор відбору базується на методі ранжування [34]. Водночас, зроблено оцінку цільової функції (функції втрат нейромережі) кожної хромосоми та виконано сортування за зменшенням значень функції втрат з розрахунком рангу. Задану кількість хромосом відібрано для кросовера.

Для оператора кросовера в системі передбачено декілька найбільш популярних методів, а саме: плоске схрещування, арифметичне та геометричне схрещування, жадібний метод [35]. Водночас перевірено існування розв'язку, який відповідає хромосомі-нащадку, з погляду системи класифікації. Найбільш частий випадок некоректних конфігурацій – невідповідність розмірностей даних, які передано між шарами різних типів. Наприклад, від згорткового шару до повнозв'язного.

Для оператора мутації використано методи простої мутації, який засновано на зміні обраної позиції хромосоми на довільне значення з можливих у словнику. Оскільки структура хромосоми фіксована, це накладає певні обмеження на методи мутації.

Для формування нового покоління використано декілька підходів: формування нового покоління тільки з нащадків, використання принципу елітизму (деяка частина предків гарантовано переходить у нову популяцію), витіснення у новому поколінні ідентичних за структурою хромосом.

Описані вище генетичні оператори та підходи до побудови генетичних алгоритмів використано в інструментальній системі класифікації звуку для оптимізації її архітектури та гіперпараметрів.

Результати обчислювальних експериментів із системою класифікації наведено у наступному підрозділі.

4.2 Результати обчислювальних експериментів із гібридною моделлю

Генетичний алгоритм з описаними в підрозділі 4.1 операторами застосовано до гібридної архітектури. Обчислювальні експерименти проводяться з набором даних UrbanSound8k [138]. Далі наведено проміжкові

результати, які отримано під час навчання генетичного алгоритму для різних значень елементів хромосом.

Формули 4.1, 4.2 містять отримані матриці розбіжностей при дослідженні впливу кількості шарів згорткової нейронної мережі, що входить до гібридної системи, на точність класифікації.

Матриця розбіжностей одного шару згортки Conv2D:

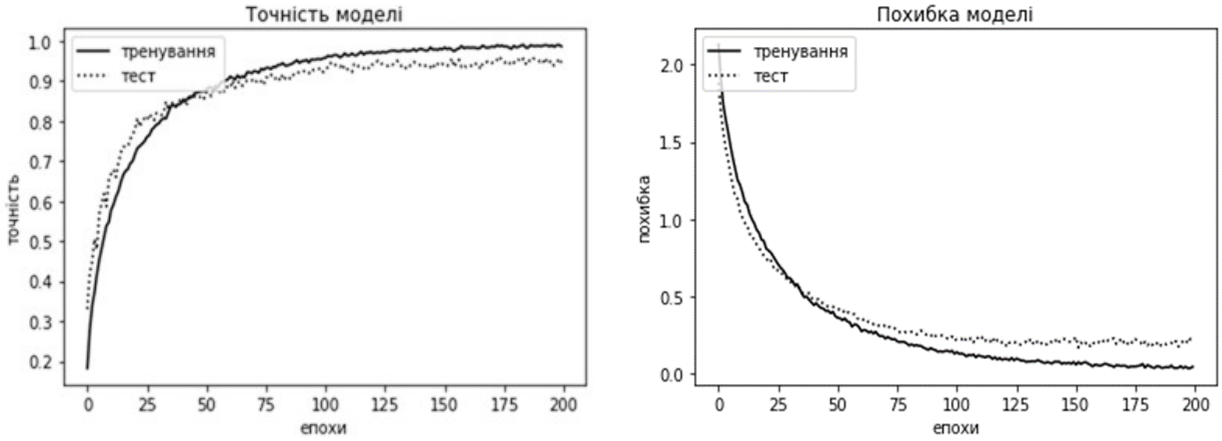
$$\begin{pmatrix} 47 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 48 & 0 & 1 & 3 & 0 & 0 & 1 & 0 \\ 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 35 & 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 47 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 42 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 48 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 43 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 15 \end{pmatrix}. \quad (4.1)$$

Матриця розбіжностей двох шарів згортки:

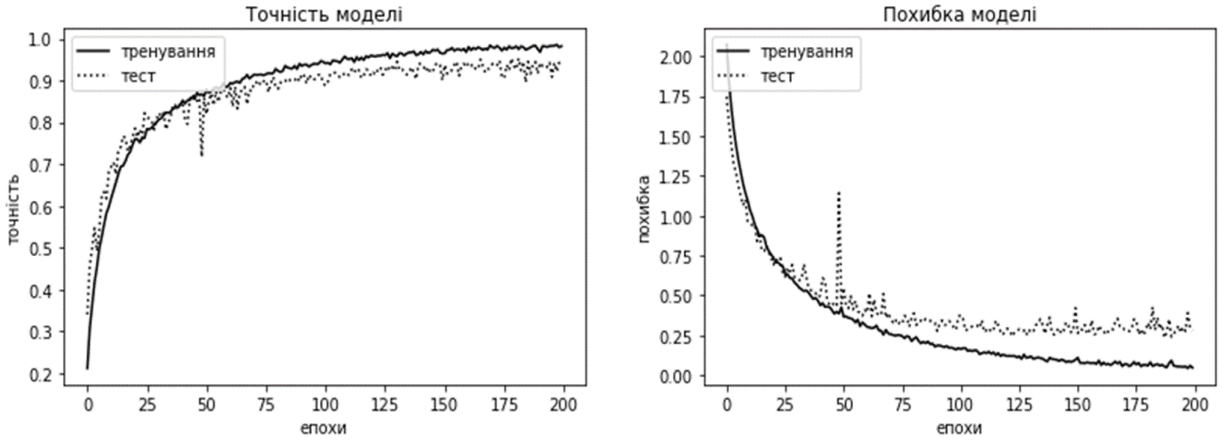
$$\begin{pmatrix} 47 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 46 & 0 & 1 & 1 & 0 & 0 & 2 & 1 \\ 1 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 1 & 0 & 36 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 1 & 3 & 43 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 42 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 48 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 42 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 15 \end{pmatrix}. \quad (4.2)$$

Аналізуючи матриці 4.1, 4.2 можна зробити висновок, що отримані значення свідчать про задовільну точність прогнозування. Зокрема, для останнього класу кількість правильно спрогнозованих класів – 15, а помилково – 2. Зазначимо, що неправильне прогнозування виконано для другого класу.

Залежність функції похибки та точності навчання від епох навчання відповідних топологій нейромереж наведено на рисунку 4.2.



а) один шар нейронної мережі



б) два шари нейронної мережі

Рисунок 4.2 – Точність та похибка прогнозування моделі з різною кількістю шарів Conv2D

Результати порівняння наведених функцій для одного та двох згорткових шарів свідчать про більш гладку практичну збіжність класифікатора саме для одного згорткового шару, при фіксованих інших гіперпараметрах. Використання класифікатора, що відповідає рисунку 4.2, б може призвести до нестійкої роботи нейромережі на практиці. Водночас, точність першого класифікатора складає 95%, другого – майже 94%. Тому, можемо зробити висновок, що одного значення точності прогнозування іноді не достатньо для повноцінної оцінки ефективності роботи нейромоделі.

Оскільки безпосередньо класифікація звуків відбувається у повнозв’язних шарах (Dense), їхня кількість є суттєвою при структурній оптимізації. На відміну від згорткових шарів, у яких відбувається вилучення суттєвих ознак аудіосигналу, кількість нейронів у Dense шарах може бути довільною. Виключенням є тільки останній повнозв’язний шар, у якому кількість нейронів повинна дорівнювати кількості класів набору звукових даних.

У формулах 4.3, 4.4 наведено результати обчислювальних експериментів із різною кількістю повнозв’язних шарів нейронів.

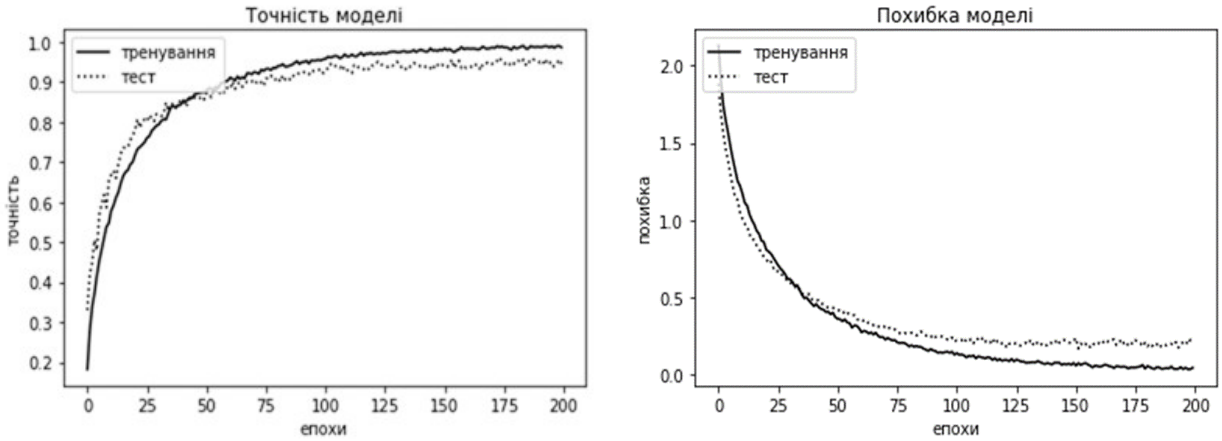
Для двох повнозв’язних шарів Dense:

$$\begin{pmatrix} 47 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 48 & 0 & 1 & 3 & 0 & 0 & 1 & 0 \\ 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 35 & 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 47 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 42 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 48 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 43 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 15 \end{pmatrix}, \quad (4.3)$$

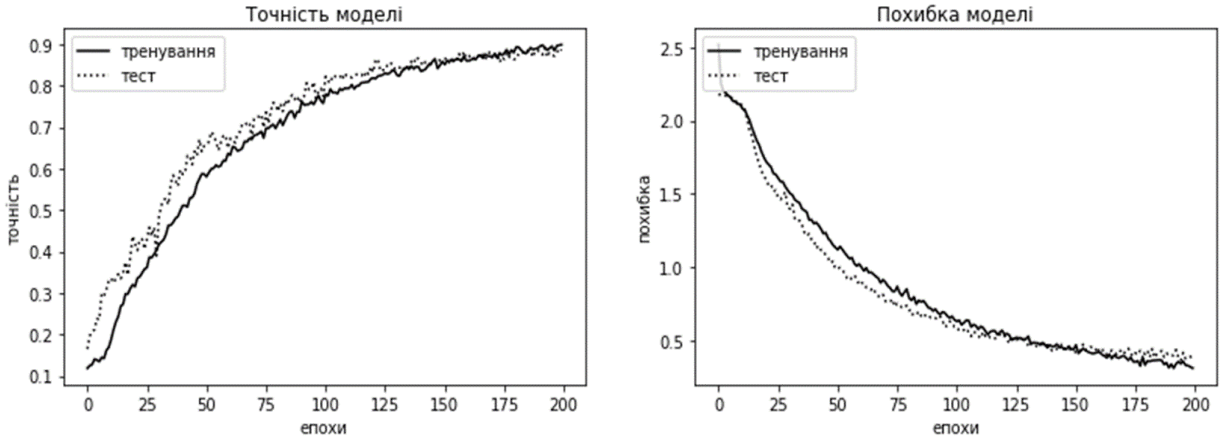
для чотирьох:

$$\begin{pmatrix} 44 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 2 & 33 & 0 & 1 & 10 & 0 & 1 & 3 & 3 \\ 0 & 1 & 40 & 0 & 1 & 0 & 0 & 0 & 0 \\ 4 & 0 & 1 & 27 & 8 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 47 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 42 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 47 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 42 & 0 \\ 0 & 2 & 0 & 0 & 1 & 1 & 0 & 0 & 13 \end{pmatrix}. \quad (4.4)$$

Залежність функцій похибки та точності від кількості епох навчання відповідних топологій нейромереж наведено на рисунку 4.3.



а) два повнозв’язних шари нейронів



б) чотири повнозв’язних шари нейронів

Рисунок 4.3 – Точність та похибка прогнозування моделі з різною кількістю шарів Dense

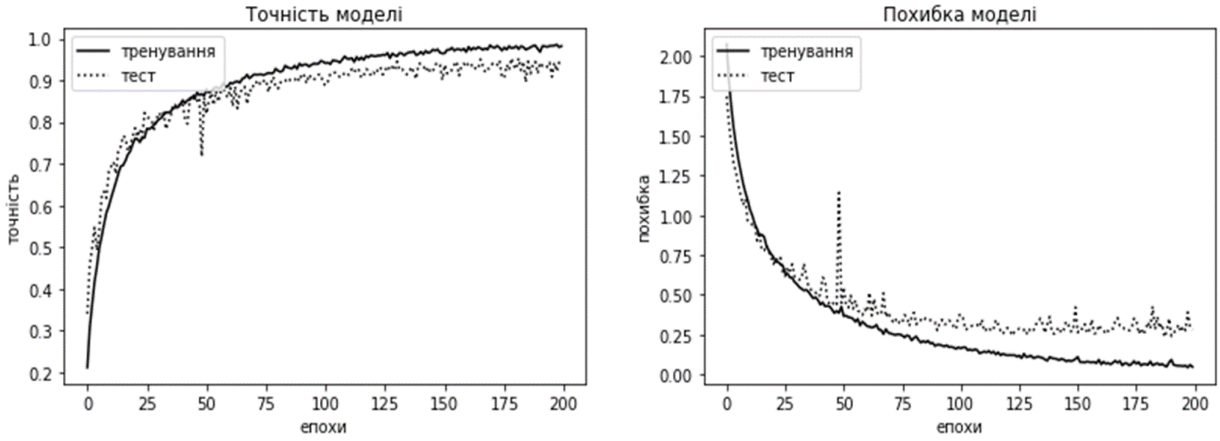
Матриця похибок для топології з двома шарами згортки Conv2D та двома шарами Dense має такий вигляд:

$$\begin{pmatrix} 47 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 44 & 0 & 1 & 3 & 0 & 0 & 3 & 0 \\ 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 2 & 1 & 31 & 4 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 48 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 42 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 48 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 43 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 15 \end{pmatrix}, \quad (4.5)$$

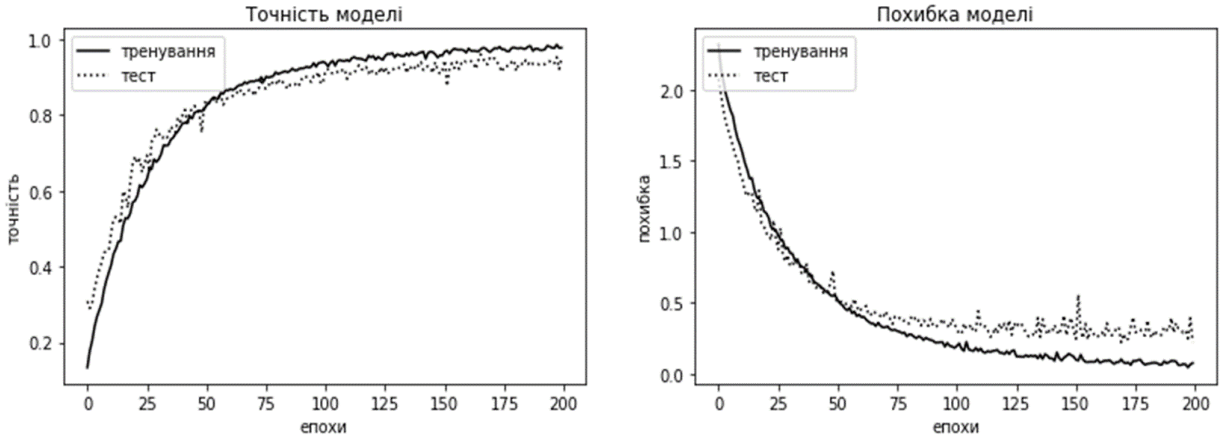
для такої самої конфігурації шарів згортки, проте з трьома повнозв’язними шарами:

$$\begin{pmatrix} 45 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 42 & 1 & 1 & 6 & 1 & 0 & 1 & 0 \\ 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 35 & 5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 47 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 41 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 47 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 42 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 15 \end{pmatrix}. \quad (4.6)$$

Залежність функції похибки та точності навчання від епох навчання відповідних топологій нейромереж наведено на рисунку 4.4.



а) 2 шари Conv2D та 2 шари Dense



б) 2 шари Conv2D та 3 шари Dense

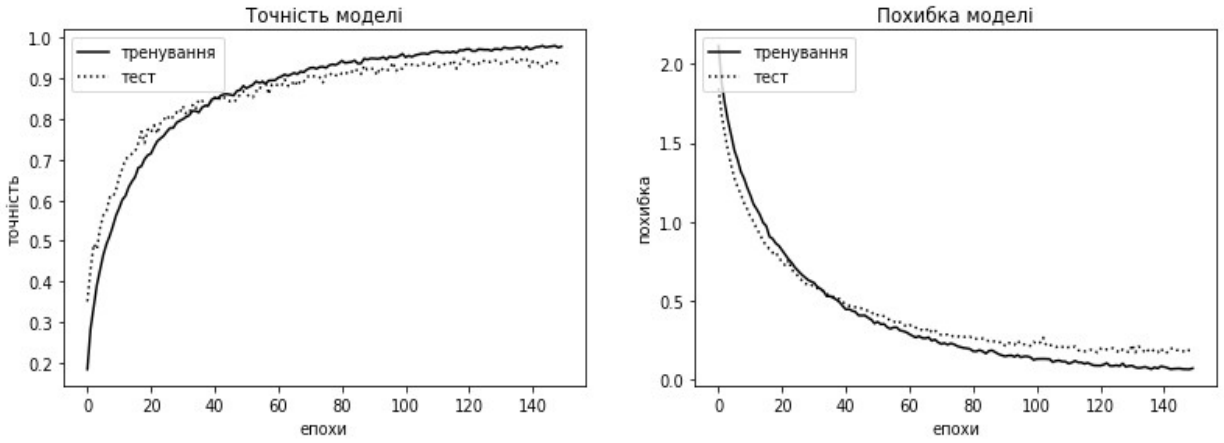
Рисунок 4.4 – Точність та похибка прогнозування моделі з різною кількістю шарів Conv2D та Dense

Матриця згортки останньої розглянутої топології зі 150 епохами навчання:

$$\begin{pmatrix} 47 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 44 & 0 & 2 & 4 & 1 & 0 & 1 & 0 \\ 1 & 0 & 40 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 35 & 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 47 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 42 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 47 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 43 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 15 \end{pmatrix}, \quad (4.7)$$

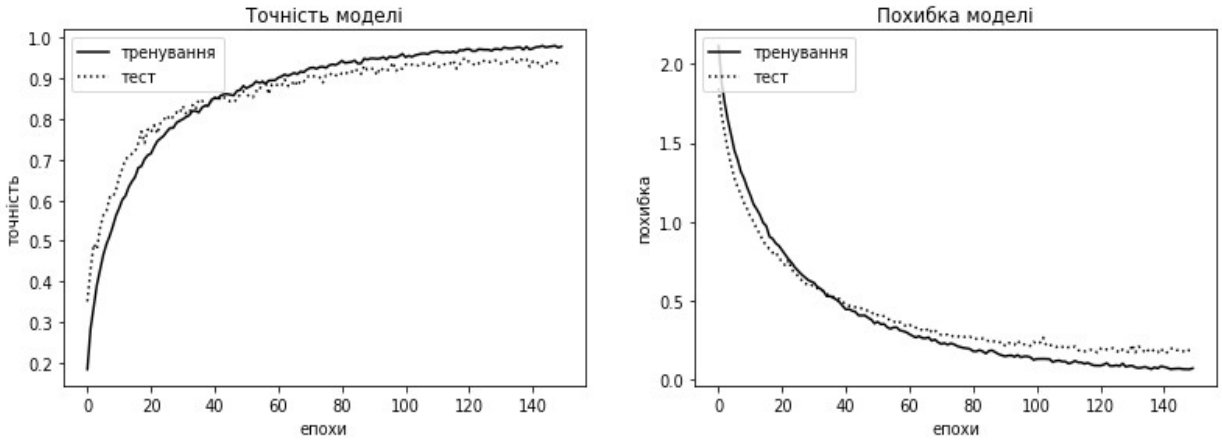
а при 200 епохах:

$$\begin{pmatrix} 47 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 48 & 0 & 1 & 3 & 0 & 0 & 1 & 0 \\ 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 35 & 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 47 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 42 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 48 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 43 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 15 \end{pmatrix}. \quad (4.8)$$



а) 150 епох навчання

Рисунок 4.5 – Точність та похибка прогнозування моделі з різною кількістю епох навчання, аркуш 1



б) 200 епох навчання

Рисунок 4.5, аркуш 2

Слід зазначити, що експерименти зі збільшенням кількості епох (більше 200) виявилися досить тривалими за часом (див. рис. 4.5). Тривалість навчання однієї епохи складала від 80 до 100 секунд з використання центрального процесору Intel Core i7-10700 2.9GHz/16MB. Тобто збільшення епох навчання не приводить до суттєвого підвищення точності прогнозування моделі, однак вимагає більших обчислювальних ресурсів. Слід зазначити, що використання графічних процесорів призводить до зменшення часу розрахунків приблизно в 10 раз, а із використанням тензорних процесорів – в 100 раз. Результати наведених обчислювальних експериментів зведено в таблицю 4.2.

Таблиця 4.2 – Етапи роботи генетичного алгоритму

Номер популяції	Кількість шарів Conv2D	Кількість шарів Dense	Кількість шарів Attention	Кількість епох	Кількість мереж у Snapshot ансамблі	Точність
2	4	1	0	10	10	0,89
3	4	3	1	50	20	0,90
4	5	4	1	100	50	0,93
5	6	2	0	50	100	0,95
6	6	4	1	200	100	0,96

Наведені значення та параметри є проміжковими результатами роботи генетичного алгоритму на різних популяціях.

У роботі [142] виконано порівняння точності класифікації моделей різних авторів на наборах даних ESC-10 та UrbanSound8k. За даними цієї публікації, найкраща точність прогнозування становить 97,75% для ESC-10 та 97,52 для UrbanSound8k. Слід зазначити, що методика оцінки ефективності моделей докладно не наводиться, в той же час, використання метрик accuracy (2.12), precision (2.13), recall (2.14) та F-метрики (2.15) призводить до різних значень точності.

4.3 Інструментальна система класифікації звукових даних

Базові архітектури згорткових нейронних мереж та автокодувальників, що описано в третьому розділі, та підходи еволюційної оптимізації з підрозділу 4.1 реалізовано в інструментальній системі класифікації звукових даних, схему компонент якої зображено на рисунку 4.6.

Модуль попередньої обробки даних містить програмну реалізацію приведення всіх аудіозаписів до однієї довжини та методи нормалізації даних для подальшої обробки алгоритмом класифікації. Також у систему імплементовано набір методів для балансування класів вибірки навчання. Ці методи базуються на випадковому генеруванні додаткових спостережень та на методі k найближчих сусідів. Застосування функцій цього модулю дає змогу отримати на виході збалансовані за кількістю спостережень класи, які містять аудіозаписи однакової довжини.

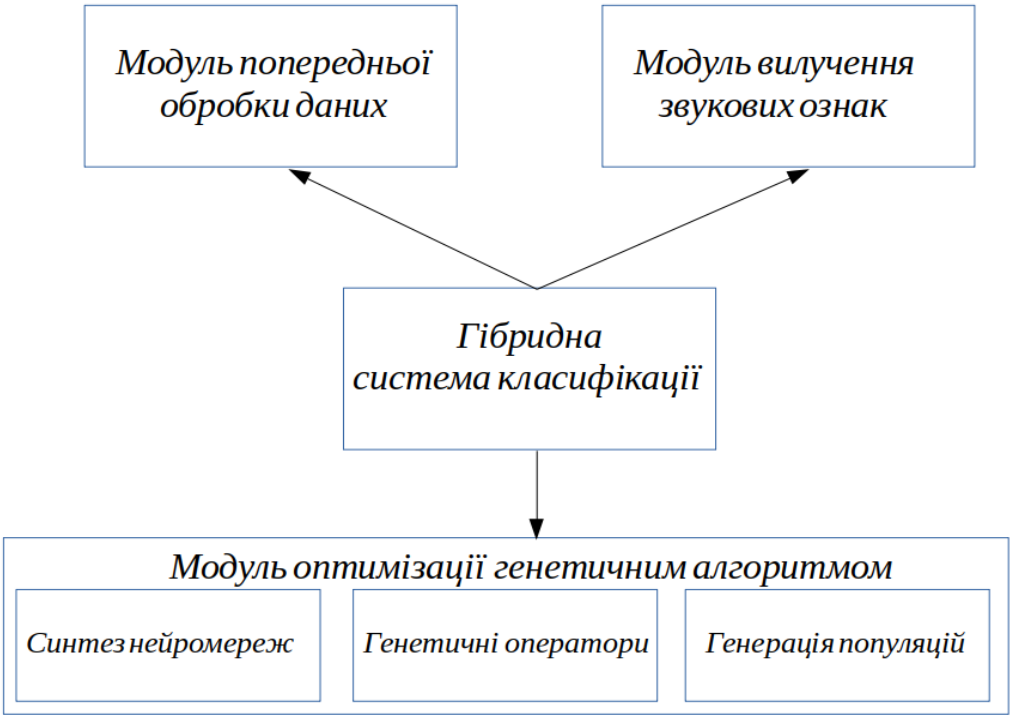


Рисунок 4.6 – Основні компоненти інструментальної системи

Методи модулю визначення ознак містять функції обчислення таких звукових ознак як мел-, барк-, гамматон-спектрограми, хромаграми, темпограми. Для оптимізації множини ознак або генерації абстрактних ознак реалізовано згортковий автокодувальник.

Модуль оптимізації за допомогою генетичного алгоритму містить декілька підсистем. Підсистему синтезу нейромереж призначено для генерації нейромережі за структурою, закодованою у хромосомі.

Підсистеми генетичних операторів та генерації популяцій реалізовують основні етапи генетичної оптимізації та застосовуються циклічно до досягнення умови зупинення.

Програмну реалізацію наведеної інструментальної системи виконано на мові програмування Python. Реалізація представляє собою бібліотеку класів та методів – набір пов’язаних між собою файлів, які розташовано у хмарному середовищі Google Colab. Особливістю роботи ресурсу є можливість використання інноваційних технологій графічних (англ. graphics processing unit, GPU) та тензорних (англ. Google Tensor Processing Unit, Google TPU)

процесорів. Використано такі бібліотеки Python: Keras для програмування структури нейронних мереж; DEAP для базової реалізації генетичного алгоритму; Librosa та Spafe для обчислення звукових ознак; Pandas для збереження таблиць даних у пам'яті комп'ютера. Програмні коди розташовано за посиланням https://github.com/AnastasiiaKryvokhata/Sound-Classification.

На рисунку 4.7 зображено програмну реалізацію процесу генерування ознак звуку.

Features methods

```
[ ]  1 def feat_specshow(S, sr, y_name, hop_len, title):
     2         plt.figure(figsize=(10, 4))
     3         librosa.display.specshow(S, sr=sr, x_axis='time',
     4                                  y_axis=y_name, hop_length = hop_len, fmax=8000)
     5         plt.colorbar(format='%+2.0f dB')
     6         plt.title(title)
     7         plt.tight_layout()
     8         plt.show()
```

```
[ ]  1 def feat_sft(y, hop_len=257, n_fft=1026):
     2     X = librosa.stft(y, n_fft=n_fft, hop_length=hop_len)
     3     S_db = librosa.amplitude_to_db(abs(X))
     4     return S_db
```

```
[ ]  1 def feat_cqt(y, sr, hop_len=256, n_fft=1026):
     2     C = np.abs(librosa.cqt(y, sr=sr, hop_length=hop_len))
     3     S_db = librosa.amplitude_to_db(C, ref=np.max)
     4     return S_db
```

```
[ ]  1 def feat_mel(y, sr, n_mfcc=40):
     2     mfccs = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=n_mfcc, dct_type=3)
     3     S_db = librosa.power_to_db(abs(mfccs), ref=np.max)
     4     return S_db
```

```
[ ]  1 def feat_chroma(y, sr):
     2     chroma = librosa.feature.chroma_stft(y=y, sr=sr)
     3     return chroma
```

Рисунок 4.7 – Генерування ознак звуку

Розроблена інструментальна система містить програмні засоби для підтримки всіх етапів класифікації звуку: попередня обробка, вилучення ознак, підготовка даних для класифікації, класифікація, оцінка якості класифікатора. Також розроблено інструменти для автоматизованого налаштування гіперпараметрів системи засобами генетичної оптимізації.

Обчислені ознаки зберігаються в хмарному середовищі та використовуються безпосередньо для класифікації або розширення даних.

Реалізацію методу конструювання базового варіанта нейронної мережі класифікатора наведено на рисунку 4.8.

```
############ Визначення шарів згорткової нейронної мережі
model = models.Sequential()
model.add(Conv2D(128, 32, 32, border_mode="same",
                 input_shape = input_shape,kernel_initializer=kernel_initializer,
                 bias_initializer=bias_initializer, kernel_regularizer=None))
model.add(BatchNormalization())
model.add(Activation(activation))
model.add(AveragePooling2D())
############ Додавання повнозв'язного шару
model.add(Flatten())
model.add(Dense(1024,kernel_initializer=kernel_initializer,bias_initializer=bias_initializer))
model.add(Activation("relu"))
model.add(Dropout(0.6))
model.add(Dense(1024,kernel_initializer=kernel_initializer,bias_initializer=bias_initializer))
model.add(Activation("relu"))
model.add(Dropout(0.8))
model.add(Dense(9,kernel_initializer=kernel_initializer,bias_initializer=bias_initializer))
model.add(Activation('softmax'))
############ Компіляція моделі
############ Головні параметри ансамблю
M = 150 # Кількість станів, які зберігаються
nb_epoch = T = 200 # Кількість епох навчання
alpha_zero = 0.0001 # Коефіцієнт швидкості навчання
model_prefix = 'Model_'
snapshot = SnapshotCallbackBuilder(T, M, alpha_zero)
optimizer = optimizers.Nadam(lr=alpha_zero, beta_1=0.9, beta_2=0.999,
                             epsilon=None, schedule_decay=0.004)
model.compile(loss = "categorical_crossentropy", optimizer = optimizer,
              metrics = ["accuracy"])
history = model.fit(X_mfcc_train, y_train, batch_size = batch_size,
                    epochs = nb_epoch, verbose=2, validation_data = (X_mfcc_test, y_test),
                    callbacks=snapshot.get_callbacks(model_prefix=model_prefix))
```

Рисунок 4.8 – Реалізація класифікатора

Деякі практичні застосування розробленої системи класифікації звуків у задачах біоакустики розглянуто в наступному підрозділі.

4.4 Застосування в біоакустиці

Підсистеми класифікації звуку впроваджуються в різноманітні системи інтернету речей (англ. Internet of Things, IoT), наприклад «розумне» місто (англ. Smart City), «розумне» сільське господарство (англ. Smart Farming) та ін. Під час розробки програмного забезпечення IoT виникають технічні складнощі, пов’язані з обмеженістю обчислювальних ресурсів та відносно невеликим об’ємом оперативної пам’яті. Отже, наступним етапом після розробки програмного забезпечення розпізнавання звуків є імплементація його до будь-якого програмно-апаратного комплексу.

Одним із напрямів практичного застосування розробленої в роботі інструментальної системи класифікації є розв’язання задач біоакустичного аналізу [21, 105, 106]. У цій області можна виділити такі напрями досліджень: моніторинг стану популяцій, ідентифікація видів тварин, аналіз словника базових звуків та ін.

Останнім часом технології машинного навчання використовуються в «розумному» господарюванні, зокрема бджільництві. Велика кількість накопичених емпіричних знань, досліджень та рекомендацій робить бджільництво важким, а часом і невизначеним. Основним завданням пасічників є визначення стану колонії та оцінка поведінки бджіл, однак ручне спостереження за вуликом може стати фактором стресу для комах.

Важливою задачею є розробка методів визначення стану колоній медоносних бджіл без втручання у вулик. Існують різні критичні проблеми для популяції, які можна визначити за допомогою вбудованих систем та відповідного програмного забезпечення. Неінвазивні методи використовують при зборі баз даних для прогнозування роїння колонії, наявності в повітрі токсичного або паразитичного кліща, відсутності маточника тощо. Мікрофон є найпростішим обладнанням для встановлення, використання та обслуговування серед різноманіття датчиків IoT. Отримання звукових даних

вулика – це найпростіша задача в більшості випадків, яка не турбує бджіл і тому може використовуватися на пасіках. Система запису звуку може бути розгорнута на одноплатних комп'ютерах, таких як RaspberryPi, Odroid або навіть Arduino, для класифікації та прогнозування в режимі реального часу.

Існують різні параметри для автоматичного визначення стану медоносних бджіл. Наприклад, вага, звук, вологість, температура, концентрація CO_2 тощо. У роботі [106] наведено огляд робіт з конструювання мультисенсорних платформ для аналізу стану вуликів. Автори цих робіт пропонують використовувати набір внутрішніх і зовнішніх датчиків вулика та RaspberryPi або Arduino як контролери. Ці вбудовані системи та спеціалізоване програмне забезпечення використовують для автоматичної класифікації стану колонії або виявлення аномалій.

Аналіз звуку вулика є джерелом різноманітних практичних застосувань. Зміни в поведінці бджіл можуть свідчити про проблеми в бджолиній сім'ї (наприклад, хвороби бджіл, відсутність маточника, наявність паразитичного кліща). Стаття [105] посилається на оригінальні дослідження, які висвітлюють кореляцію між звуком вулика та станом бджіл.

Однією з ключових проблем у такій програмі аналізу звуку є вибір відповідних акустичних ознак. Загальним підходом тут є оцінка деяких спектральних і кепстральних характеристик. Наприклад, проблема присутності бджолиної матки аналізується з використанням кепстральних мел-частотних коефіцієнтів у роботах розглянутих в монографії [21]. У роботі порівнюються опорні векторні машини та згорткові нейронні мережі для розпізнавання стану вулика за допомогою оригінальних звукових записів з вуликів. Автори повідомляють про відсутність узагальнення системи класифікації звуків щодо нових вуликів. Цей факт обмежує використання будь-якої системи класифікації звуків на пасіках навіть з одного географічного району.

Проблема роїння в колоніях медоносних бджіл також розглядається в публікаціях, розглянутих у [105]. Автори пропонують використовувати

акселерометри для збору акустичних вібрацій. Аналіз основних компонентів використовується для оптимізації функцій.

Отже, у практичних застосуваннях автоматизованих систем аналізу звуку до задач бджільництва важливою задачею є збір бази даних звуків та маркування відповідно до життєвого циклу бджолиної сім'ї.

Для тестування розробленої системи класифікації використано звуковий набір даних, який зібрано з експериментальної пасіки Запорізького національного університету [106]. Під час типового життєвого циклу медоносної бджолиної сім'ї з літа 2019 року до весни 2020 року було проведено моніторинг звуку десяти вуликів, а звукові дані періодично реєструвались у чітко визначених станах бджолосімей. Пристрої запису складалися з електретного мікрофона (50-16 кГц) підключеного до комп'ютера. Записаний сигнал було оцифровано за допомогою мікросхеми Intel QM67 Express з 1 моноканалом на 16 біт і частотою дискретизації 44,1 кГц. Набір даних містить більше 10 тисяч звукових файлів .wav для навчання та тестування системи. Навчальний зразок містить аудіофайли, які асоційовано з різними категоріями: спокійний стан медоносної бджолиної сім'ї; період інтенсивного збирання квіткового нектару та пилку; без маточних бджіл; у роєвому стані; з молодими бджолиними матками. Кількість файлів в одній категорії перевищує тисячу, тривалість кожного з них становить 6 секунд [106]. Таким чином, після навчання систему може бути розгорнуто на одноплатному комп'ютері і може використовуватися автономно як частина інтелектуальних систем «розумного» сільського господарства.

Перспективи подальших досліджень у цій області пов'язано з дослідженням можливості використання переносу навчання (англ. Transfer Learning), адже відомо, що звукові ознаки бджолиних колоній суттєво відрізняються.

4.5 Висновки до розділу 4

Отже, у даному розділі генетичний алгоритм використано з метою оптимізації гіперпараметрів розроблених гібридних нейромережевих моделей та їх ансамблів для класифікації звукових даних.

Наведено генетичні оператори та схему кодування хромосоми.

Запропоновано інструментальну систему класифікації звукових даних на основі згорткових нейронних мереж та автокодувальників з оптимізацією їх параметрів за допомогою генетичних алгоритмів. Інструментальну систему класифікації реалізовано у вигляді високорівневої бібліотеки функцій, що може використовуватись при розробці наукового програмного забезпечення.

Запропонована інструментальна система класифікації реалізує такі етапи: попередня обробка даних з метою формування інформативних ознак звуку, побудова класифікатора, конструювання ансамблю класифікаторів. Методом ансамблювання є алгоритм Snapshot, перевагою якого є швидкість навчання.

Наведено результати обчислювальних експериментів на відкритому наборі звукових даних UrbanSound8k.

Проілюстровано спосіб практичного застосування розробленої інструментальної системи у біоакустиці для класифікації звуку комах. Отже, програмне забезпечення розроблене за допомогою інструментальної системи можна інтегрувати у програмно-апаратні комплекси різного призначення.

Основні результати четвертого розділу опубліковано у роботах [21, 23, 55, 103, 105, 106].

ВИСНОВКИ

Основним результатом роботи є удосконалення моделей та методів підвищення ефективності розпізнавання аудіосигналів засобами нейронних мереж за рахунок автоматичного налаштування параметрів нейромережевих моделей та їх ансамблів за допомогою генетичних алгоритмів, а саме:

– у роботі виконано аналіз сучасного стану задачі розпізнавання аудіо сигналів, на основі якого встановлено, що нейромережеві моделі є одним з найбільш універсальних методів розпізнавання, однак використання цих методів на практиці потребує налаштування великої кількості параметрів нейронних мереж та їх ансамблів; одним з підходів до автоматичного вибору структури нейромереж та налаштування їхніх гіперпараметрів є генетичні алгоритми;

– уперше розроблено гібридні нейромережеві моделі класифікації звукових даних з використанням генетичних алгоритмів для оптимізації параметрів моделей та їх ансамблів, що дало змогу підвищити точність розпізнавання класів звуків до 96%;

– набув подальшого розвитку генетичний алгоритм структурної оптимізації в частині поліпшення архітектури ансамблів нейронних мереж, що дало змогу підвищити точність прогнозування;

– удосконалено нейромережеві моделі класифікації даних у частині оптимізації множини ознак звукових даних, що дало змогу мінімізувати розмірність довільної задачі класифікації;

– уперше запропоновано інструментальну систему класифікації звукових даних на основі згорткових нейронних мереж та автокодувальників з оптимізацією їхніх параметрів за допомогою генетичних алгоритмів;

– достовірність ґрунтується на обчислювальних експериментах із використанням відкритих даних. Усі розроблені математичні моделі реалізовано в інструментальній системі класифікації звукових даних;

– отримані результати впроваджено в навчальний процес при вивченні дисциплін «Емпіричні методи програмної інженерії», «Засоби машинного навчання», «Нейронні мережі» та при виконанні кваліфікаційних робіт студентами спеціальності 121 «Інженерія програмного забезпечення».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бархатов Н. А., Ревунов С. Е. Искусственные нейронные сети в задачах солнечно-земной физики : монография. Нижний Новгород : Типография «Поволжье», 2010. 407 с.

2. Безсонов О. О. Еволюційні штучні нейронні мережі прямого розповсюдження : архітектури, навчання, застосування : дис. ... д-ра техн. наук : 05.13.23. Харків, 2017. 40 с.

3. Белозерский Л. А. Введение в системы автоматического распознавания. Киев : Наукова думка, 2005. 434 с.

4. Бодянский Е. В. Гибридные нейро-фаззи модели и мультиагентные технологии в сложных системах : монография. Днепропетровск : Системные технологии, 2008. 403 с.

5. Бондарев В. Н., Трёстер Г., Чернега В. С. Цифровая обработка сигналов : методы и средства. Харьков : Конус, 2001. 398 с.

6. Вікторов Є. О. Гібридні еволюційні нейронні мережі та їх навчання : автореф. дис. ... канд. техн. наук : 05.13.23. Харків, 2011. 19 с.

7. Винокурова О. А. Гібридні еволюційні адаптивні вейвлет-нейро-фаззі-системи для динамічного аналізу даних : автореф. дис. ... д-ра техн. наук : 05.13.23. Харків, 2012. 40 с.

8. Гольцев А. Д. Нейронные сети с ансамблевой организацией. Киев : Наукова думка, 2005. 200 с.

9. Горшков Е. В. Классификация данных в условиях неопределённости на основе гибридных нейро-фаззи архитектур : дис. ... канд. техн. наук : 05.13.23. Харьков, 2007. 150 с.

10. Долотов А. І. Самонавчанні спайк-нейронні мережі в задачах інтелектуального аналізу даних : автореф. дис. ... канд. техн. наук : 05.13.23. Харків, 2011. 20 с.

11. Дубровін В. І., Субботін С. О. Методи оптимізації та їх застосування в задачах навчання нейронних мереж : навч. посіб. для студ. вищ. навч. закл. Запоріжжя : ЗНТУ, 2003. 136 с.

12. Іванченко Г. Ф. Системи штучного інтелекту : навч. посіб. Київ : КНЕУ, 2011. 382 с.

13. Клюшин Д. А. Непараметричні методи розпізнавання з гарантованим рівнем значущості : дис. ... д-ра фіз.-мат. наук : 01.05.01. Київ, 2008. 278 с.

14. Козин И. В. Эволюционные модели в дискретной оптимизации. Запорожье : ЗНУ, 2019. 204 с.

15. Козін І. В., Селютін Є. К. Особливості пошуку оптимальних класифікацій: еволюційні алгоритми. *Вісник Запорізького національного університету. Фізико-математичні науки.* Запоріжжя, 2019. №2. С. 62–68.

16. Кривохата А. Г. Оптимізація згорткової нейронної мережі при розв'язанні задачі класифікації акустичних даних. *Інновації науки XXI століття* : збірник наукових матеріалів XXXVI міжнародної науково-практичної інтернет-конференції. (Вінниця, 18 листопада 2019). Вінниця, 2019. № 5. С. 23–27.

17. Кривохата А. Г. Класифікація звуків з використанням згорткових автокодувальників. *Інформаційні технології та комп'ютерне моделювання* : матеріали Міжнародної науково-практичної конференції. (Івано-Франківськ, 18-22 травня, 2020). Івано-Франківськ : п. Голіней О. М., 2020. С. 181–182.

18. Кривохата А. Г., Кудін О. В., Лісняк А. О. Огляд методів машинного навчання для класифікації акустичних даних. *Вісник Херсонського національного технічного університету*. Херсон, 2018. Т. 1, № 3 (66). С. 327–331.

19. Кривохата А. Г., Кудін О. В., Лісняк А. О. Методи глибинного навчання у задачах машинного слуху. *XIX Міжнародна конференція з математичного моделювання, присвячена 250-річчю з дня народження Жана Батиста Фур'є* : матеріали конференції. (Херсон, 17-21 вересня 2018). Херсон, 2018. С. 70–71.

20. Кривохата А. Г., Кудін О. В., Давидовський М. В., Лісняк А. О. Застосування ансамблевого навчання в задачах класифікації акустичних даних. *Вісник Запорізького національного університету*. *Фізико-математичні науки*. Запоріжжя, 2018. № 1. С. 49–61.

21. Кривохата А. Г., Кудін О. В., Чопоров С. В. Нейромережеві математичні моделі у задачах обробки звукових сигналів : монографія. Херсон : Видавничий дім «Гельветика», 2020. 120 с.

22. Кудін О. В., Кривохата А. Г. Методи класифікації акустичних даних. *Актуальні проблеми математики та інформатики* : збірка тез доповідей Дев'ятої Всеукраїнської, шістнадцятої регіональної наукової конференції молодих дослідників. (Запоріжжя, 26-27 квітня 2018). Запоріжжя, 2018. С. 37–38.

23. Кудін О. В., Кривохата А. Г. Генетичні алгоритми оптимізації ансамблів згорткових нейронних мереж. *Комп'ютерні науки, інформаційні технології та системи управління* : матеріали Міжнародної науково-технічної конференції здобувачів вищої освіти та молодих вчених. (Івано-Франківськ 27-29 листопада 2019). Івано-Франківськ, 2019. С. 28.

24. Кудін О. В., Кривохата А. Г. Застосування згоркових нейронних мереж в задачах класифікації акустичних даних. *Комп'ютерні науки, інформаційні технології та системи управління* : матеріали Міжнародної науково-технічної конференції молодих вчених, аспірантів та студентів. (Івано-Франківськ, 28-30 листопада, 2018). Івано-Франківськ : Прикарпатський національний університет ім. Василя Стефаника, 2018. С. 145–146.

25. Лебьодкіна А. Ю. Методи та моделі прискореної нейромережевої обробки даних у розподіленому обчислювальному середовищі : автореф. дис. ... канд. техн. наук : 05.13.23. Харків, 2012. 20 с.

26. Лепский А. Е., Броневич А. Г. Математические методы распознавания образов: Курс лекций. Таганрог : ТТИ ЮФУ, 2009. 155 с.

27. Мазуров В. Д. Математические методы распознавания образов : уч. пособие, 2-е изд. Екатеринбург : Урал. ун-т, 2010. 101 с.

28. Мерков А. Б. Распознавание образов: построение и обучение вероятностных моделей. Москва : ЛЕНАНД, 2014. 240 с.

29. Мінаєв Ю. М., Філімонова О. Ю. Розв’язання прикладних інженерних задач в нейронних мережах : навч.-метод. посібник. Ч. 1. Теоретичні основи штучних нейронних мереж та головні передумови для розв’язування прикладних задач. Київ : Національний авіаційний університет, 2003. 75 с.

30. Надеран С. В. Розпізнавання будівель на супутникових зображеннях надвисокої роздільної здатності за допомогою нейронних мереж : автореф. дис. ... канд. техн. наук : 05.12.23. Київ, 2015. 18 с.

31. Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. Санкт-Петербург : Питер, 2018. 480 с.

32. Новотарський М. А., Нестеренко Б. Б. Штучні нейронні мережі: обчислення. *Праці Інституту математики НАН України*. Київ, 2004. Т. 50. 408 с.

33. Олешко Д. Н. Информационная технология ускорения синтеза нейронных сетей для решения задач прогнозирования при принятии решений : автореф. дис. ... канд. техн. наук: 05.13.06. Одесса, 2005. 165 с.

34. Олійник А. О. Еволюційні методи відбору інформативних ознак та побудови нейромережевих моделей для розпізнання образів : автореф. дис. ... канд. техн. наук : 05.13.23. Харків, 2009. 19 с.

35. Олійник А. О., Субботін С. О., Олійник О. О. Еволюційні обчислення та програмування. Запоріжжя : ЗНТУ, 2010. 324 с.

36. Олійник О. О. Мультиагентні методи побудови нейро-нечітких мереж : автореф. дис. ... канд. техн. наук : 05.13.23. Харків, 2010. 19 с.

37. Перепелиця В. О., Козін І. В., Терещенко Е. В. Задачі класифікації: підходи, методи, алгоритми. Запоріжжя : Поліграф, 2008. 188 с.

38. Порхун О. В. Автоматична класифікація багатовимірних об’єктів із застосуванням апарату нейронних мереж : автореф. дис. ... канд. фіз.-мат. наук : 01.05.01. Київ, 2009. 19 с.

39. Руденко О. Г., Бодянський Є. В. Штучні нейронні мережі : навчальний посібник. Київ : Компанія СМІТ, 2006. 404 с.

40. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская и др.; пер. с пол. И. Д. Рудинский. Москва : Горячая линия-Телеком, 2007. 383 с.

41. Сакало Є. С. Фрагментна обробка зображень на основі штучних нейронних мереж : автореф. дис. ... канд. техн. наук : 05.13.23. Харків, 2011. 19 с.

42. Синєглазов В., Чумаченко О. Глибокі нейронні мережі для вирішення завдань розпізнавання і класифікації зображення. *Інформаційні технології та комп’ютерне моделювання 2017* : матеріали Міжнародної науково-практичної конференції. (Івано-Франківськ - Яремче, 15-20 травня 2017). Івано-Франківськ, 2017. С. 274–277. URL : http://itcm.comp-sc.if.ua/2017/Sineglazov.pdf (дата звернення: 06.12.2019).

43. Системи штучного інтелекту: нечітка логіка, нейронні мережі, нечіткі нейронні мережі, генетичний алгоритм : монографія / В. П. Лисенко, В. М. Решетюк, В. М. Штепа та ін. Київ : НУБіП України, 2014. 332 с.

44. Скобцов Ю. А. Метаэвристики : монография. Донецк : Изд-во «Ноулидж» (Донецкое отделение), 2013. 426 с.

45. Соколов Е. А., Федоров Е. Е. Машинное обучение на ФКН ВШЭ. URL : https://github.com/esokolov/ml-course-hse (дата звернення: 18.09.2019).

46. Тесленко Н. О. Нейро-фаззі моделі та системи, що самонавчаються, у задачах інтелектуального аналізу даних : автореф. дис. ... канд. техн. наук : 05.13.23. Харків, 2009. 20 с.

47. Тимофєєва А. Є., Кудін О. В., Кривохата А. Г., Лісняк А. О. Автоматичне анотування зображень за допомогою нейронних мереж. *Вчені*

записки Таврійського національного університету імені В.І. Вернадського. Київ, 2019. Т. 30(69), № 2, Ч. 1. С. 214–220.

48. Тимощук П. В. Штучні нейронні мережі : навчальний посібник. Львів : Видавництво Львівської політехніки, 2011. 444 с.

49. Тимощук П. В., Лобур М. В. Основи теорії проектування нейронних мереж : навчальний посібник. Львів : Видавництво Національного університету «Львівська політехніка», 2007. 328 с.

50. Ткаченко Р. О., Ткаченко П. Р., Ізонін І. В. Нейромережеві засоби штучного інтелекту : навчальний посібник. Львів : Видавництво Львівської політехніки, 2017. 207 с.

51. Федоров Е. Е. Искусственные нейронные сети : монография. Красноармейск : ДВНЗ «ДонНТУ», 2016. 337 с.

52. Хайкин С. Нейронные сети: полный курс, 2-е издание. Москва : «Вильямс», 2006. 1104 с.

53. Худий А. М. Ефективні методи та алгоритми обробки сигналів і даних в системних та нейронних середовищах : автореф. дис. ... канд. техн. наук : 05.13.06. Львів, 2002. 16 с.

54. Четирбок П. В. Розпізнавання об’єктів на основі векторної міри близькості образів у просторі похибок : автореф. дис. ... канд. техн. наук : 05.13.23. Київ, 2015. 19 с.

55. Чопорова О. В., Кривохата А. Г. Оптимізація згорткових нейронних мереж та їх ансамблів. *Вісник Запорізького національного університету*. *Фізико-математичні науки*. Запоріжжя, 2019. № 1. С. 107–115.

56. Чумаченко О. І. Структурно-параметричний синтез гібридних нейронних мереж : дис. д-ра техн. наук : 05.13.23. Харків, 2019. 40 с.

57. Abdoli S., Cardinal P., Koerich A. L. End-to-end environmental sound classification using a 1D convolutional neural network. *Expert Systems With Applications*. 2019. Vol. 136 (2019). P. 252–263.

58. Abeber J., Mimilakis S.-I., Gräfe R., Lukashevich H. Acoustic scene classification by combining autoencoder-based dimensionality reduction and

convolutional neural networks. *Detection and Classification of Acoustic Scenes and Events (DCASE 2017)* : proceedings of the 2017 Workshop. (Munich, 16-17 November 2017). Munich, Germany, 2017. P. 7–11.

59. Alías F., Socoró J. C., Sevillano X. A review of physical and perceptual feature extraction techniques for speech, music and environmental sounds. *Applied Sciences*. 2016. № 6(5):143. P. 1–44.

60. Deep Unsupervised Representation Learning for Abnormal Heart Sound Classification / S. Amiriparian, N. Cummins, K. Qian [et al.]. *Engineering in Medicine and Biology* : proceedings of the 40th International IEEE Conference. (Honolulu, 18-21 July 2018). Honolulu, HI, USA, 2018. P. 4776–4779.

61. Anumula J., Neil D., Delbruck T., Liu S.-C. Feature Representations for Neuromorphic Audio Spike Streams. *Frontiers in Neuroscience*. 2018. Volume 12, Article 23. P. 1–12. DOI: https://doi.org/10.3389/fnins.2018.00023.

62. Aytar Y., Vondrick C., Torralba A. SoundNet: Learning Sound Representations from Unlabeled Video. *Neural Information Processing Systems (NIPS 2016)* : proceedings of the 29th International Conference. (Barcelona, 5-10 December 2016). New York, USA, 2016. P. 892–900. DOI: https://dl.acm.org/doi/10.5555 /3157096.3157196.

63. Babaei K., Chen Z. Y., Maul T. Data Augmentation by AutoEncoders for Unsupervised Anomaly Detection. *Preprint arXiv.org*. 2019. 8 p. URL: https://arxiv.org/abs/1912.13384 (дата звернення: 12.02.2020).

64. Bach J.-H., Meyer A.-F., McElfresh D., Anemüller J. Automatic classification of audio data using nonlinear neural response models. *International Conference on Acoustics, Speech and Signal Processing (ICASSP)* : proceedings of IEEE International Conference. (Kyoto, 25-30 March 2012). Kyoto, Japan, 2012. P. 357–360.

65. Bakhshi A., Noman N., Chen Z., Zamani M., Chalup S. Fast Automatic Optimisation of CNN Architectures for Image Classification Using Genetic Algorithm. *Congress on Evolutionary Computation (CEC 2019)* : proceedings of the 2019 IEEE Congress on Evolutionary Computation. (Wellington, 10-13 June

2019). Wellington, New Zealand, 2019. P. 1283–1290. DOI: https://doi.org/10.1109/CEC.2019.8790197.

66. Baldominos A., Saez Y., Isasi P. Evolutionary Design of Convolutional Neural Networks for Human Activity Recognition in Sensor-Rich Environments. *Sensors (Basel)*. 2018. № 18(4): 1288. P. 1–24.

67. Baucas M. J., Spachos P. Using cloud and fog computing for large scale IoT-based urban sound classification. *Simulation Modelling Practice and Theory*. 2020. Volume 101, Article 102013. P. 1–11.

68. Bertin-Mahieux T., Eck D., Mandel M. Automatic tagging of audio: the state-of-the-art. *Machine Audition: Principles, Algorithms and Systems*. 2011. P. 334–352.

69. Bohrer J. S, Grisci B. I., Dorn M. Neuroevolution of Neural Network Architectures Using CoDeepNEAT and Keras. *Preprint arXiv.org.* 2020. 29 p., URL: https://arxiv.org/abs/2002.04634 (дата звернення: 10.04.2020).

70. Burges C. J. S., Platt J. C., Jana S. Extracting noise-robust features from audio data. *Acoustics, Speech and Signal Processing (ICASSP 2002)* : proceedings of the IEEE International Conference. (Orlando, 13–17 May 2002). Orlando, FL, USA, 2002. P. 1021–1024.

71. Camastra F., Vinciarelli A. Machime learning for Audio, Image and Video analysis. London : Springer-Verlag, 2015. 561 p.

72. Cecchi S., Spinsante S., Terenzi A., Orcioni A. A Smart Sensor-Based Measurement System for Advanced Bee Hive Monitoring. *Sensors.* 2020. Vol. 20 (9), Article 2726. DOI: https://doi.org/10.3390/s20092726.

73. Environmental sound classification with dilated convolutions / Y. Chen, Q. Guo, X. Liang [et al.]. *Applied Acoustics*. 2019. Vol. 148 (2019). P. 123–132.

74. Chi Z., Li Y., Che C. Deep Convolutional Neural Network Combined with Concatenated Spectrogram for Environmental Sound Classification. *Computer Science and Network Technology (ICCSNT 2019)* : proceedings of the IEEE 7th International Conference. (Dalian, 19-20 October 2019). Dalian, China, 2019. P. 251–254.

75. Costa C. H. L., Valle Jr. J. D., Koerich A. L. Automatic classification of audio data. *Systems, Man and Cybernetics* : proceedings of the IEEE International Conference. (Hague, 10-13 October 2004). Hague, Netherlands, 2004. P. 562–567.

76. Doncieux S., Paolo G., Laflaquière A., Coninx A. Novelty Search makes Evolvability Inevitable. *Preprint arXiv.org*, 2020. 9 p, URL: https://arxiv.org/abs/2005.06224 (дата звернення: 30.05.2020).

77. Classification of general audio data for content-based retrieval / D. Li, I. K. Sethi, N. Dimitrova [et al.] *Pattern Recognition Letters*. 2001. № 22(5). P. 533–544.

78. Elsken T., Metzen J. H., Hutter F. Neural Architecture Search: A Survey. *Journal of Machine Learning Research*. 2019. Vol. 20. P. 1–21.

79. Elhassan A. T., Aljourf M., Al-Mohanna F., Shoukri M. Classification of Imbalance Data using Tomek Link (T-Link) Combined with Random Under-sampling (RUS) as a Data Reduction Method. *Global Journal of Technology & Optimization*. 2017. №1. P.1–11. DOI: https://doi.org/10.4172/2229-8711.S1111

80. Faraji M. M., Shouraki S. B., Iranmehr E. Unsupervised Feature Selection for Phoneme Sound Classification using Genetic Algorithm. *International Journal of Mechatronics, Electrical and Computer Technology*. 2018. Vol. 8(27). P. 3753–3763.

81. General-purpose Tagging of Freesound Audio with AudioSet Labels: Task Description, Dataset, and Baseline / E. Fonseca, M. Plakal, F. Font [et al.] *Detection and Classification of Acoustic Scenes and Events (DCASE 2018)* : proceedings of the DCASE 2018 Workshop. (Surrey, 19-20 November 2018). Surrey, UK, 2018. P. 1–4.

82. auDeep: Unsupervised Learning of Representations from Audio with Deep Recurrent Neural Networks / M. Freitag, S. Amiriparian, S. Pugachevskiy [et al.]. *Journal of Machine Learning Research*. 2018. Vol. 18. P. 1–5.

83. Audio set: an ontology and human-labeled dataset for audio events / J. F. Gemmeke, D. P. W. Ellis, D. Freedman [et al.]. *Acoustics, Speech and Signal Processing* : proceedings of the IEEE ICASP 2017. (New Orleans, 5-9 March

2017). New Orleans, LA, USA, 2017. URL: https://research.google.com/pubs/archive /45857.pdf (дата звернення: 06.06.2018).

84. Geron A. Hands-On Machine Learning with Scikit-Learn and TensorFlow. Sebastopol : O`Reilly, 2017. 861 p.

85. Gonzalez J. A., Hurtado L.-F., Pla F. ELiRF-UPV at SemEval-2019 Task 3: Snapshot Ensemble of Hierarchical Convolutional Neural Networks for Contextual Emotion Detection. *Semantic Evaluation (SemEval-2019)* : proceedings of the 13th International Workshop. (Minneapolis, 6-7 June 2019). Minneapolis, Minnesota, USA, 2019. P. 195–199. URL: https://www.aclweb.org/anthology/S19-2031.pdf (дата звернення: 30.09.2019).

86. Goodfellow I., Bengio Y., Courville A. Deep Learning. London : MIT Press, 2016. 802 p.

87. Gottapu R. D., Dagli C. H. Efficient Architecture Search for Deep Neural Networks. *Procedia Computer Science*. 2020. Vol. 168. P. 19–25. DOI: https://doi.org/10.1016/j.procs.2020.02.246.

88. Gülcü Y., Kuş Z. Hyper-Parameter Selection in Convolutional Neural Networks Using Microcanonical Optimization Algorithm. *IEEE Access.* 2020. Vol. 8. P. 52528–52540. DOI: https://doi.org/10.1109/ACCESS.2020.2981141

89. Semi-Supervised Active Learning for Sound Classification in Hybrid Learning Environments / W. Han, E. Coutinho, H. Ruan [et al.]. *PloS ONE*. 2016. № 11(9): e0162075. P. 1–23. DOI: https://doi.org/10.1371/journal.pone.0162075.

90. Hinz T., Navarro-Guerrero N., Magg S., Wermter S. Speeding up the Hyperparameter Optimization of Deep Convolutional Neural Networks. *International Journal of Computational Intelligence and Applications*. 2018. Vol. 17, No. 2. P. 1–15. DOI: https://doi.org/10.1142/S1469026818500086.

91. Howell J., Rooth M., Wagner M. Acoustic classification of focus : on the web and in the lab. *Laboratory Phonology: Journal of the Association for Laboratory Phonology*. 2017. № 8(1), 16. P. 1–41. DOI: https://doi.org/10.5334/labphon.8.

92. Hu W., Lv J., Liu D., Chen Y. Unsupervised Feature Learning for Heart Sounds Classification Using Autoencoder. *Journal of Physics: Conference Series*. 2018. Vol. 1004. P. 1–9. DOI: https://doi.org/10.1088/1742-6596/1004/1/012002.

93. Snapshot Ensembles: Train 1, Get M for Free / G. Huang, Y. Li, G. Pleiss [et al.]. *International Conference on Learning Representations (ICLR 2017)* : proceedings of the 5th international conference. (Toulon, 24-26 April 2017). Toulon, France, 2017. P. 1–14. URL: https://openreview.net/pdf?id=BJYwwY9ll. (дата звернення: 06.06.2018).

94. Huang G., Liu Z., van der Maaten L., Weinberger K. Densely Connected Convolutional Networks. *Computer Vision and Pattern Recognition (CVPR 2017)* : proceedings of the 2017 IEEE Conference. (Honolulu, 21-26 July 2017). HI, USA, 2017. P. 1–9. URL: https://arxiv.org/abs/1608.06993. (дата звернення: 06.06.2018).

95. Iba H., Noman N. Deep Neural Evolution. Deep Learning with Evolutionary Computation. Singapore : Springer Nature, 2020. 437 p. DOI: https://doi.org/10.1007 /978-981-15-3685-4.

96. Ibrahim Z. Al A., Ferrane I., Joly P. Audio Data Analysis Using Parametric Representation of Temporal Relations. *Information and Communication Technologies: from Theory to Applications (ICTTA 2006)* : proceedings of the IEEE International Conference. (Damacus, 24-28 April 2006). Damacus, Syria, 2006. P.1337–1343 URL: https://www.researchgate.net/publication/224645341_ Audio_Data_Analysis_using_Parametric_Representation_of_Temporal_Relations (дата звернення: 06.06.2018).

97. Shuffling and mixing data augmentation for environmental sound classification / T. Inoue, P. Vinayavekhin, S. Wang [et al.]. *Detection and Classification of Acoustic Scenes and Events (DCASE 2019)* : proceedings of the Workshop DCASE 2019. (New York, 25-26 October 2019). New York, USA, 2019. P. 109–113. DOI: https://doi.org/10.33682/wgyb-bt40.

98. Jatturas C., Ayudhya P. D. N., Pankaew S., Asdornwised W. Performance Comparison of Environmental Sound Classification using Scikit-learn. *Circuits*

Systems, Computers and Communications (ITC-CSCC, 2018) : proceedings of the 33rd annual international technical conference. (Bangkok, 4-7 July 2018). Bangkok, Thailand, 2018. URL: https://www.researchgate.net/publication/327751167_ Performance_Comparison_of_Environmental_Sound_Classification_using_Scikit-learn (дата звернення: 20.09.2019).

99. Jayarathne I., Cohen M. Autoencoder-based One-class Classification. *The Society of Instrument and Control Engineers (SICE 2019)* : Proceedings of the 32th Tohoku branch workshop. (Hiroshima, 10-13 September 2019). Aizu-Wakamatsu, Japan, 2019. P. 1–6. URL: https://www.researchgate.net/publication/337929447_ Autoencoder-based_One-class_Classification (дата звернення: 20.09.2019).

100. Sound Classification Using Convolutional Neural Network and Tensor Deep Stacking Network / A. Khamparia, D. Gupta, N. G. Nhu [et al.]. *IEEE Access. Special Section on New Trends in Brain Signal Processing and Analysis*. 2019. Vol. 7. P. 7717–7727. DOI : https://doi.org/10.1109/ACCESS.2018.2888882.

101. Kong Q., Xu Y., Wang W., Plumbley M. D. A joint separation-classification model for sound event detection of weakly labelled data. *Acoustics, Speech and Signal Processing (ICASSP 2018)* : proceedings of the IEEE International Conference. (Calgary, 15-20 April 2018). Calgary, Canada, 2018. P. 321–325. DOI: https://doi.org/10.1109/ICASSP.2018.8462448.

102. Kong Q., Xu Y., Wang W., Plumbley M. D. Convolutional gated recurrent neural network incorporating spatial features for audio tagging. *Neural Networks (IJCNN 2017)* : proceedings of the International Joint Conference. (Anchorage, 14-19 May 2017). Anchorage, Alaska, 2017. P. 3461–3466. DOI: https://doi.org/10.1109 /IJCNN.2017.7966291.

103. Kryvokhata A. Evolutionary methods in environmental sound classification. *International Journal of Mathematics and Computer Research*. 2020. Vol. 8, Issue 7. P. 2091–2095.

104. Kryvokhata A. The sound classification system based on neural networks with attention mechanism and autoencoders. *International Journal of Mathematics and Statistics Invention (IJMSI)*. 2020. Vol. 8, Issue 6. P. 24–28.

105. Kryvokhata A., Kudin O., Gorbenko V. Design sound classification IoT system with genetic algorithms. *Visnyk of Zaporizhzhia National University. Physical and Mathematical Sciences*. Zaporizhzhia, 2019. № 2. P. 69–74. DOI: https://doi.org/10.26661/2413-6549-2019-2-08.

106. Kudin O., Kryvokhata A., Gorbenko V. Developing a Deep Learning Sound Classification System for a Smart Farming. *237th ECS Meeting with the 18th International Meeting on Chemical Sensors (IMCS 2020)* : Meeting Abstracts of The Electrochemical Society. (Canada, May 2020). Canada, 2020. Vol. 2020-01, N. 26. P. 1853. DOI: 10.1149/ma2020-01261853mtgabs.

107. Kwasigroch A., Grochowski M., Mikołajczyk M. Deep neural network architecture search using network morphism. *Methods and Models in Automation and Robotics (MMAR)* : proceedings of the 2019 24th International Conference. (Międzyzdroje, 26-29 August 2019). Międzyzdroje, Poland, 2019. P. 30–35. DOI: https://doi.org/10.1109/MMAR.2019.8864624.

108. LeCun Y., Bengio Y. Convolutional Networks for Images, Speech, and Time-Series. *The Handbook of Brain Theory and Neural Networks* / M. A. Arbib [et al.]. Cambridge : MIT Press, 1995. P. 255–258.

109. Lerch A. An Introduction to Audio Content Analysis. New Jersey : John Wiley & Sons, Inc., Hoboken, 2012. 248 p.

110. Lezhenin I., Bogach N., PyshkiniE. Urban Sound Classification using Long Short-Term Memory Neural Network. *Computer Science and Information Systems (FedCSIS 2019)* : proceedings of the 14th Federated Conference. (Leipzig, 1-4 September 2019). *Annals of Computer Science and Information Systems*, 2019. Vol. 18. P. 57–60. DOI: https://doi.org/10.15439/2019F185.

111. Feature extraction and classification of heart sound using 1D convolutional neural networks / F. Li, M. Liu, Y. Zhao, L. Kong [et al.]. *EURASIP Journal on Advances in Signal Processing*. 2019. Vol. 2019:59. P. 1–11.

112. A Novel Dynamic Weight Neural Network Ensemble Model / K. Li, W. Liu, K. Zhao [et al.]. *International Journal of Distributed Sensor Networks*. 2015. Vol. 11. P. 1–13. DOI: https://doi.org/10.1155/2015/862056.

113. Li X., Chebiyyam V., Kirchhoff K. Multi-stream Network With Temporal Attention For Environmental Sound Classification. *Preprint arXiv.org.* 2019. 5 p. URL: https://arxiv.org/abs/1901.08608 (дата звернення: 30.05.2019).

114. Lingxi X., Yuille A. Genetic CNN. *Computer Vision (ICCV 2017)* : proceedings of the IEEE International Conference. (Venice, 22-29 October 2017). Venice, Italy. 2017. P. 1379–1388.

115. Liu H., Simonyan K., Yang Y. DARTS: Differentiable Architecture Search. *Learning Representations (ICLR 2019)* : proceedings of the Seventh International Conference. (New Orleans, 6-9 May 2019). New Orleans, 2019. P. 1–13. URL: https://openreview.net/pdf?id=S1eYHoC5FX (дата звернення: 15.05.2020).

116. Lone M. A., Islam M. A Brief Overview of Developing Convolutional Neural Network Using Genetic Algorithm. *International Journal of Computer Sciences and Engineering*. 2019. Vol. 7, Issue 2. P. 812–818. DOI: https://doi.org/10.26438/ ijcse/v7i2.812818.

117. Lopes V., Fazendeiro P. A Hybrid Method for Training Convolutional Neural Networks. *Preprint arXiv.org*, 2020. 6 p. URL: https://arxiv.org/abs/2005.04153 (дата звернення 20.05.2020).

118. Loussaief S., Abdelkrim A. Convolutional Neural Network Hyper-Parameters Optimization based on Genetic Algorithms. *International Journal of Advanced Computer Science and Applications*. 2018. Vol. 9, No. 10. P. 252–266.

119. Lyon R. F. Machine Hearing: An Emerging Field. *IEEE Signal Processing Magazine*. 2010. Vol. 27. P. 131–139.

120. Lyu Z., Karns J., ElSaid A., Desell T. Improving Neuroevolution using Island Extinction and Repopulation. *Preprint arXiv.org*, 2020. 11 p. URL: https://arxiv.org/abs/2005.07376 (дата звернення 30.05.2020).

121. Mcloughlin M.P., Stewart R., McElligott A.G. Automated biacoustics: methods in ecology and conservation and their potential for animal welfare monitoring. *Journal of the royal society interface.* 2019. Vol. 16, Issue 155. DOI: https://doi.org/10.1098/rsif.2019.0225

122. An Experiment on the Use of Genetic Algorithms for Topology Selection in Deep Learning / F. Mattioli, D. Caetano, A. Cardoso [et al.]. *Journal of Electrical and Computer Engineering*. 2019. Vol. 2019, Art. ID 3217542. P. 1–12. DOI: https://doi.org/10.1155/2019/3217542.

123. Energy-Efficient Gabor Kernels in Neural Networks with Genetic Algorithm Training Method / F. Meng, X. Wang, F. Shao [et al.]. *Electronics*. 2019. Vol. 8(1), No. 105. DOI: https://doi.org/10.3390/electronics8010105.

124. Mierswa I., Morik K. Learning feature extraction for learning from audio data. *Technische Universität Dortmund. Technical Reports*. Dortmund, 2004. № 55. P. 1–14.

125. Molina-Cabello M. A., Accino C., López-Rubio E., Thurnhofer-Hemsi K. Optimization of Convolutional Neural Network Ensemble Classifiers by Genetic Algorithms. *IWANN 2019* : Lecture Notes in Computer Science. (Gran Canaria, 12-14 June 2019). Cham, 2019. Vol. 11507. P. 163–173.

126. Pitfalls to Avoid when Interpreting Machine Learning Models / C. Molnar, G. König, J. Herbinger, T. Freiesleben [et al.]. *Preprint arXiv.org*, 2020. 10 p. URL: https://arxiv.org/abs/2007.04131 (дата звернення: 01.08.2020).

127. Nolasco I., Benetos E. To bee or not to bee: investigating machine learning approaches for beehive sound recognition, *Preprint arXiv.org*, 2018. 5 p. URL: https://arxiv.org/abs/1811.06016 (дата звернення: 30.05.2019).

128. Omelianenko I. Hands-On Neuroevolution with Python: Build high-performing artificial neural network architectures using neuroevolution-based algorithms. Birmingham : Packt Publishing, 2019. 368 p.

129. Oppenheim A. V., Schafer R. W. Discrete-Time Signal Processing, Third Edition. Harlow : Pearson Education Limited, 2014. 1055 p.

130. Park E., Wong R. K., Chu V. W. Classifier Learning from Imbalanced Corpus by Autoencoded Over-Sampling. *PRICAI 2019: Trends in Artificial Intelligence* : proceedings of the 16th International Conference. (Cuvu, August 26-30 2019). Cuvu, Yanuca Island, Fiji, 2019. Part I. P. 16–29.

131. Piczak K. J. ESC: Dataset for Environmental Sound Classification. *ACM Multimedia* : proceedings of the 23rd Annual Conference. (Brisbane, 26-30 October 2015). New York, 2015. P. 1015–1018. DOI: https://doi.org/10.1145/2733373.2806390.

132. Radiuk P., Kutucu H. Heuristic Architecture Search Using Network Morphism for Chest X-Ray Classification. *Intelligent Information Technologies & Systems of Information Security (IntellTSIS 2020)* : proceedings of the 1st International Workshop. (Khmelnitskyi, 10-12 June 2020). Khmelnitskyi, Ukraine, 2020. P. 107–121.

133. Real E., Liang C., So D. R., Le Q. V. AutoML-Zero: Evolving Machine Learning Algorithms From Scratch. *Preprint arXiv.org*, 2020. 23 p. URL: https://arxiv.org/abs/2003.03384 (дата звернення: 01.08.2020).

134. Risi S. Stanley K. O. An Enhanced Hypercube-Based Encoding for Evolving the Placement, Density, and Connectivity of Neurons. *Artificial Life.* 2012. Vol. 18, No. 4. P. 331–363. DOI: https://doi.org/10.1162/ARTL_a_00071.

135. Rizzi A., Buccino M., Panella M., Uncini A. Optimal short-time features for music/speech classification of compressed audio data. *International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)* : proceedings of 2006 International Conference. (Sydney, 28 Nov.-1 Dec. 2006). Sydney, Australia, 2006. P. 1–7.

136. Roche F., Hueber T., Limier S., Girin L. Autoencoders for music sound modeling: a comparison of linear, shallow, deep, recurrent and variational models. *Preprint arXiv.org*. 2019. 8 p. URL: https://arxiv.org/abs/1806.04096 (дата звернення: 30.05.2020).

137. Rosenblatt F. Principles of neurodynamics: Perceptions and the theory of brain mechanism. Washington DC : Spartan Books. 1961. 616 p.

138. Salamon J., Jacoby C., Bello J. P. A dataset and taxonomy for urban sound research. *ACM international conference on Multimedia* : proceedings of the 22nd ACM international conference. (Orlando, 3-7 November 2014). New York, 2014. P. 1041–1044. DOI: http://dx.doi.org/10.1145/2647868.2655045.

139. Sallam H., Regazzoni C. S., Talkhan I., Atiya A. Evolving neural networks ensembles NNEs. *Cognitive Information Processing* : proceedings of the IAPR Workshop. (Santorini, 9-10 June 2008). Santorini, Greece, 2008. P. 142–147.

140. A Survey on the Latest Development of Machine Learning in Genetic Algorithm and Particle Swarm Optimization / A. Kulkarni, S. Satapathy, Sarmah D. K. [et al.]. *Optimization in Machine Learning and Applications.* Singapore : Springer, 2020. P. 91–112. DOI: https://doi.org/10.1007/978-981-15-0994-0_6.

141. Sengupta N., Sahidullah Md., Saha G. Optimization of cepstral features for robust lung sound classification. *INDICON -2015* : proceedings of the 12th Annual IEEE India Conference. (New Deli, 17-19 December 2015). New Delhi, 2015. P. 1–6.

142. Sharma J., Granmo O.-C., Goodwin M. Environment Sound Classification using Multiple Feature Channels and Attention based Deep Convolutional Neural Network. *Preprint arXiv.org*. 2020. 13 p. URL: https://arxiv.org/abs/1908.11219 (дата звернення: 30.05.2020).

143. Shin S. Y., Kim H.-j. Autoencoder-based One-class Classification Technique for Event Prediction. *Cloud Computing and Internet of Things CCIOT 2019* : proceedings of the 4th International Conference. (Tokyo, 20-22 September 2019). Tokyo, Japan, 2019 P. 54–58. DOI: https://doi.org/10.1145/3361821.3361831.

144. Singh J., Joshi R. Background Sound Classification in Speech Audio Segments. *Speech Technology and Human-Computer Dialogue (SpeD)* : proceedings of the International Conference. (Timisoara, 10-12 October 2019). Timisoara, Romania. 2019. P. 1–6. DOI: https://doi.org/10.1109/SPED.2019.8906597.

145. Soares S. G., Antunes C. H., Arajo R. A Genetic Algorithm for Designing Neural Network Ensembles. *Genetic and Evolutionary Computation Conference GECCO`12* : proceedings of the 14th Annual Conference. (Philadelphia, 7-11 July 2012). New York, USA, 2012. P. 681–688.

146. Stanley K. O., D'Ambrosio D. B., Gauci J. A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks. *Artificial Life.* 2009. Vol. 15, Issue 2. P. 185–212. DOI: https://doi.org/10.1162/artl.2009.15.2.15202.

147. Stanley K. O., Miikkulainen R. Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation.* 2002. Vol. 10, Number 2. P. 99–127.

148. Stastný J., Skorpil V., Fejfar J. Audio Data Classification by Means of New Algorithms. *Telecommunications and Signal Processing (TCP)* : proceedings of the 36th International Conference. (Rome, 2-4 July 2013). Rome, 2013. P. 507–511.

149. Stowell D. Computational Bioacoustic Scene Analysis. *Computational Analysis of Sound Scenes and Events* / T. Virtanen, M. Plumbley, D. Ellis [et al.] Cham : Springer, 2018. P. 303–333. DOI: https://doi.org/10.1007/978-3-319-63450-0_11

150. Sturm B. L. A Survey of Evaluation in Music Genre Recognition. *Adaptive Multimedia Retrieval: Semantics, Context, and Adaptation AMR 2012* : proceedings of the 10th International Conference. (Copenhagen, 24-25 October 2012). Cham, 2012. Vol. 8382 (2014). P. 29–66.

151. Su Y., Zhang K., Wang J., Madani K. Environment Sound Classification Using a Two-Stream CNN Based on Decision-Level Fusion. *Sensors.* 2019. Vol. 19, No. 1733. P. 1–15. DOI: https://doi.org/10.3390/s19071733.

152. Suganuma M., Shirakawa S., Nagao T. A. Genetic Programming Approach to Designing Convolutional Neural Network Architectures. *International Joint Conference on Artificial Intelligence (IJCAI-18)* : proceedings of the Twenty-Seventh International Joint Conference. (Stockholm, 13-19 July 2018). Stockholm, 2018. P. 5369–5373.

153. Sun Y., Xue B., Zhang M., Yen G. G. Automatically Designing CNN Architectures Using Genetic Algorithm for Image Classification. *IEEE Transactions on Cybernetics*. 2020. P. 1–15.

154. Deep CNN Framework for Environmental Sound Classification using Weighting Filters / B. Tang, Y. Li, X. Li., L. Xu [et al.]. *International Conference on Mechatronics and Automation (ICMA 2019)* : proceedings of 2019 IEEE International Conference. (Tianjin, 4-7 August 2019). Tianjin, China, 2019. P. 2297–2302.

155. Tangkawanit S., Pinthong C., Kanprachar S. Development of Gunfire Sound Classification System with a Smartphone using ANN. *International Conference on Digital Arts, Media and Technology (ICDAMT 2018)* : proceedings of the 3rd International Conference. (Phayao, 25-28 February 2018). Phayao, Thailand, 2018. P. 168–172.

156. Tao S. Deep Neural Network Ensembles. *Preprint arXiv.org*. 2019. 9 p. URL: https://arxiv.org/abs/1904.05488 (дата звернення: 20.03.2020).

157. Terenzi A., Cecchi S., Orcioni S., Piazza F. Features Extraction Applied to the Analysis of the Sounds Emitted by Honey Bees in a Beehive. *11th International Symposium on Image and Signal Processing and Analysis (ISPA) :* proceedings of the International Symposium. (Dubrovnik, 23-25 September 2017). IEEE, 2019. P. 3–8. DOI: https://doi.org/10.1109/ISPA.2019.8868934.

158. Thwe K. Z., War N. Environmental Sound Classification based on Time-frequency Representation. *International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)* : proceedings 18th IEEE/ACIS International Conference. (Kanazawa, 26-28 June 2017). Kanazawa, Japan, 2017. P. 251–255.

159. Tirumala S. S. Evolving deep neural networks using coevolutionary algorithms with multi-population strategy. *Neural Computing and Applications.* 2020. № 16. P. 13051–13064. DOI : https://doi.org/10.1007/s00521-020-04749-2.

160. Valle R. Hands-On Generative Adversarial Networks with Keras: Your guide to implementing next-generation generative adversarial networks. Birmingham : Packt Publishing Ltd, 2019. 316 p.

161. Joint Training of Neural Network Ensembles / A. M. Webb, C. Reynolds, D.-A. Iliescu [et al.]. *Preprint arxiv.org*. 2019. 14 p. URL: https://arxiv.org/abs/1902.04422 (дата звернення: 20.03.2020).

162. Automatic audio tagging using covariate shift adaptation / G. Wichern, M. Yamada, H. Thornburg [et al.]. *Acoustics, Speech and Signal Processing (ICASSP)* : proceedings of the IEEE International Conference. (Dallas, 14-19 March 2010). Dallas, USA, 2010. P. 253–257.

163. Wilkinghoff K., Kurth F. Open-Set Acoustic Scene Classification with Deep Convolutional Autoencoders. *Detection and Classification of Acoustic Scenes and Events 2019 (DCASE2019)* : proceedings of the 2019 Workshop. (New York, 25-26 October 2019). New York, USA, 2019. P. 258–262.

164. Wilson A., Fazenda B. M. Variation in Multitrack Mixes: Analysis of Low-level Audio Signal Features. *Journal of the Audio Engineering Society*. 2016. Vol. 64, No. 7/8. P. 466–473. DOI: http://dx.doi.org/10.17743/jaes.2016.0029.

165. Wirsansky E. Hands-On Genetic Algorithms with Python. Birmingham : Packt Publishing Ltd, 2020. 334 p.

166. A Spiking Neural Network Framework for Robust Sound Classification / J. Wu, Y. Chua, M. Zhang [et al.]. *Frontiers in Neuroscience*. 2018. Vol. 12, Article 836. P. 1–17. DOI: https://doi.org/10.3389/fnins.2018.00836.

167. Unsupervised Feature Learning Based on Deep Models for Environmental Audio Tagging / Y. Xu, Q. Huang, W. Wang [et al.]. *IEEE/ACM transactions on audio, speech and language processing*. 2017. Vol. 25 (6). P. 1230–1241.

168. Yakovlev S., Kartashov O., Pichugina O., Korobchynskyi K. Genetic Algorithms for Solving Combinatorial Mass Balancing Problem. *2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON)* : proceedings of the 2019 IEEE conference. (Lviv, 2-6 July 2019). Lviv, 2019. p. 1061–1064. DOI: https://doi.org/10.1109/UKRCON.2019.8879938.

169. Yakovlev S., Kartashov O., Pichugina O. Optimization on Combinatorial Configurations Using Genetic Algorithms. *Computer Modeling and*

Intelligent Systems (CMIS-2019) : proceedings of the Second International Workshop. (Zaporizhzhia, 15-19 April 2019). CEUR, 2019. Vol. 2353, urn:nbn:de:0074-2353-0. P. 28–40. URL: http://ceur-ws.org/Vol-2353/paper3.pdf

170. Yakovlev S., Pichugina O., Yarovaya O. Polyhedral-spherical configurations in discrete optimization problems. *Journal of Automation and Information Sciences*. 2019. Vol. 51, Issue 1. P. 26-40.

171. Zaccone G., Karim Md. R., Menshawy A. Deep learning with TensorFlow. Birmingham : Packt Publishing Ltd, 2018. 767 p.

172. Zhang X., Zou Y., Wang W. LD-CNN: A Lightweight Dilated Convolutional Neural Network for Environmental Sound Classification. *Pattern Recognition (ICPR 2018)* : proceedings of the 24th International Conference. (Beijing, 20-24 August 2018). Beijing, China, 2018. P. 373–378.

173. Attention based Convolutional Recurrent Neural Network for Environmental Sound Classification / Z. Zhang, S. Xu, T. Qiao, S. Zhang [et al.]. *Pattern Recognition and Computer Vision (PRCV 2019)* : Lecture Notes in Computer Science. (Xi'an, 8-11 November 2019). Cham : Springer, 2019. Vol. 11857. P. 261–271. DOI: https://doi.org/10.1007/ 978-3-030-31654-9_23.

174. Snore-GANs: Improving Automatic Snore Sound Classification With Synthesized Data / Z. Zhang, J. Han, K. W. Qian [et al.]. *IEEE Journal of Biomedical and Health Informatics*. 2019. Vol. 24, Issue 1. P. 300–310.

175. Zhao H., Huang X., Liu W., Yang L. Environmental sound classification based on feature fusion. *MATEC Web of Conferences*. 2018. Vol. 173, Art. 03059. P.1–5.

176. Zhou J., Peng L., Chen X., Yang D. Robust sound event classification by using denoising autoencoder. *Multimedia Signal Processing (MMSP 2016)* : proceedings of the IEEE 18th International Workshop. (Montreal, 21-23 September 2016). Montreal, Canada, 2016. P. 1–6.

Додаток А

Список публікацій за темою дисертації та відомості про апробацію результатів дисертації

Праці, в яких опубліковані основні наукові результати:

1. Кривохата А. Г., Кудін О. В., Давидовський М. В., Лісняк А. О. Застосування ансамблевого навчання в задачах класифікації акустичних даних. *Вісник Запорізького національного університету*. *Фізико-математичні науки*. Запоріжжя, 2018. № 1. С. 50–62.

2. Чопорова О. В., Кривохата А. Г., Оптимізація згорткових нейронних мереж та їх ансамблів. *Вісник Запорізького національного університету*. *Фізико-математичні науки*. Запоріжжя, 2019. № 1. С. 107–115.

3. Kryvokhata A., Kudin O., Gorbenko V. Design Sound Classification IoT System with Genetic Algorithm. *Visnyk of Zaporizhzhia National University. Physical and Mathematical Sciences*. Zaporizhzhia, 2019. № 2. P. 69−74.

4. Кривохата А. Г., Кудін О. В., Чопоров С. В. Нейромережеві математичні моделі у задачах обробки звукових сигналів : монографія. Херсон : Видавничий дім «Гельветика», 2020. 120 с.

5. Kryvokhata A. The sound classification system based on neural networks with attention mechanism and autoencoders. *International Journal of Mathematics and Statistics Invention (IJMSI).* 2020. Volume 8, Issue 6. P. 24−28.

6. Kryvokhata A. Evolutionary methods in environmental sound classification. *International Journal of Mathematics and Computer Research.* 2020. Volume 8, Issue 7. P. 2091−2095.

Праці, які засвідчують апробацію матеріалів дисертації:

7. Кудін О. В. Кривохата А. Г. Методи класифікації акустичних даних. *Актуальні проблеми математики та інформатики* : збірка тез та доповідей Дев'ятої Всеукраїнської, шістнадцятої регіональної наукової конференції молодих дослідників. (Запоріжжя, 26-27 квітня 2018). Запоріжжя, 2018. С. 37–38.

8. Кривохата А. Г. Оптимізація згорткової нейронної мережі при розв'язанні задачі класифікації акустичних даних. *Інновації науки XXI століття* : збірник наукових матеріалів XXXVI Міжнародної науково-практичної інтернет-конференції. (Вінниця, 18 листопада 2019). Вінниця, 2019. С. 23–27.

9. Кудін О. В., Кривохата А. Г. Генетичні алгоритми оптимізації ансамблів згорткових нейронних мереж. *Комп'ютерні науки, інформаційні технології та системи управління* : матеріали Міжнародної науково-технічної конференції здобувачів вищої освіти та молодих вчених. (Івано-Франківськ, 27-29 листопада 2019). Івано-Франківськ, 2019. С. 28.

10. Кривохата А. Г., Кудін О. В., Лісняк А. О. Методи глибинного навчання у задачах машинного слуху. *Міжнародна конференція з математичного моделювання* : матеріали XIX міжнародної конференції з математичного моделювання, присвяченої 250-річчю з дня народження Жана Батиста Фур'є. (Херсон, 17-21 вересня 2018). Херсон : ХНТУ, 2018. С. 70–71.

11. Кудін О. В., Кривохата А. Г. Застосування згорткових нейронних мереж в задачах класифікації акустичних даних. *Комп'ютерні науки, інформаційні технології та системи управління* : матеріали міжнародної науково-технічної конференції молодих вчених, аспірантів та студентів. (Івано-Франківськ, 28-30 листопада 2018). Івано-Франківськ : Прикарпатський національний, 2018. С. 145–146.

12. Kudin O., Kryvokhata A., Gorbenko V. Developing a Deep Learning Sound Classification System for a Smart Farming. *237th ECS Meeting with the 18th*

International Meeting on Chemical Sensors (IMCS 2020) : Meeting Abstracts of The Electrochemical Society. (Canada, May 2020). Canada, 2020. Vol. 2020-01, N. 26. P. 1853. DOI: 10.1149/ma2020-01261853mtgabs.

13. Кривохата А. Г. Класифікація звуків з використанням згорткових автокодувальників. *Інформаційні технології та комп'ютерне моделювання* : матеріали міжнародної науково-практичної конференції. (Івано-Франківськ, 18-22 травня 2020). Івано-Франківськ, 2020. С. 181–182.

Праці, які додатково відображають результати дисертації:

14. Кривохата А. Г., Кудін О. В., Лісняк А. О. Огляд методів машинного навчання для класифікації акустичних даних. *Вісник Херсонського національного технічного університету*. Херсон, 2018. Т. 1. № 3 (66). С. 327–331.

15. Тимофєєва А. Є. Кудін О. В. Кривохата А. Г. Лісняк А. О. Автоматичне анотування зображень за допомогою нейронних мереж. *Вчені записки Таврійського національного університету імені В.І. Вернадського*. Київ, 2019. Т. 30 (69). С. 214–220.

Додаток Б

Акт впровадження у навчальний процес Запорізького національного університету

«ЗАТВЕРДЖУЮ»
Ректор Запорізького національного університету
_____ М. О. Фролов
«11» ____06____ 2020 р.

Акт

про впровадження результатів кандидатської дисертації
КРИВОХАТИ АНАСТАСІЇ ГРИГОРІВНИ
«Нейромережеві математичні моделі звукових сигналів у задачах розпізнавання»

Комісія в складі: д.т.н., професора Гоменюка С. І. (голова комісії), к.ф.-м.н., доцента Лісняка А. О., д.т.н., доцента Гребенюка С. М., склала теперішній акт про те, що наукові дослідження, виконані Кривохатою А. Г. у кандидатській дисертації, впроваджені в навчальний процес та застосовуються при виконанні кваліфікаційних робіт студентами Запорізького національного університету. Результати досліджень використовуються при викладанні курсів «Емпіричні методи програмної інженерії», «Засоби машинного навчання», «Нейронні мережі» для студентів спеціальності 121 – «Інженерія програмного забезпечення».

Члени комісії:

Декан математичного факультету, д.т.н., професор — С. І. Гоменюк

Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент — А. О. Лісняк

Завідувач кафедри фундаментальної математики, д.т.н., доцент — С. М. Гребенюк

Додаток В

Програмна реалізація інструментальної системи

Програмна реалізація є високорівневою бібліотекою функцій обробки звукових даних у форматі .wav, побудови нейромережевих моделей, ансамблів класифікаторів, генетичних алгоритмів.

Підключення низькорівневих бібліотек обробки звуку.

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import sys
import IPython
import librosa
import seaborn as sb
import scipy
import matplotlib.pyplot as plt
from scipy import signal
import IPython.display as ipd
import os
from sklearn.model_selection import StratifiedShuffleSplit
#from sklearn.cross_validation import StratifiedKFold
from tensorflow.keras import losses, models, optimizers, regularizers,
initializers
from tensorflow.keras.activations import relu, softmax
from tensorflow.keras.callbacks import (EarlyStopping, LearningRateSche
duler,
             ModelCheckpoint, TensorBoard, ReduceLROnPlateau)
from tensorflow.keras.layers import (Convolution2D, Dense, Dropout, Glo
balAveragePooling2D,
             GlobalMaxPool2D, Input, MaxPool2D, concatenate, Activation,
             MaxPooling2D,Flatten,BatchNormalization, Conv2D,
             AveragePooling2D, GRU, Bidirectional, Reshape, BatchNormali
zation,
              Conv2DTranspose, LeakyReLU)
from tensorflow.keras.layers import (Convolution2D, Dense, Dropout, Glo
balAveragePooling2D,
```

```
            GlobalMaxPool2D, Input, MaxPool2D, concatenate, Activation,
            MaxPooling2D,Flatten,BatchNormalization, Conv2D,AveragePoo
ling2D,
            UpSampling2D,LocallyConnected2D, ConvLSTM2D,  MaxPooling3D,
            AveragePooling3D, TimeDistributed)
from tensorflow.keras.utils import Sequence, to_categorical
from tensorflow.keras.optimizers import SGD, Adam
```

Функції обчислення та візуалізації ознак звуку.

```
def feat_specshow(S, sr, y_name, hop_len, title):
    plt.figure(figsize=(10, 4))
    librosa.display.specshow(S, sr=sr, x_axis='time',
                y_axis=y_name, hop_length = hop_len, fmax=8000)
    plt.colorbar(format='%+2.0f dB')
    plt.title(title)
    plt.tight_layout()
    plt.show()
def feature_vis(name1, name2, y1, y2, sr1, sr2, method):
  hop_length=256
  if method == 'sft':
    S_db1 = feat_sft(y1)
    S_db2 = feat_sft(y2)
    feat_specshow(S_db1, sr1, 'linear', hop_length, 'SFT Spectrogram ' +
name1)
    feat_specshow(S_db2, sr2, 'linear', hop_length, 'SFT Spectrogram ' +
name2)
  if method == 'cqt':
    S_db1 = feat_cqt(y1, sr1)
    S_db2 = feat_cqt(y2, sr1)
    feat_specshow(S_db1, sr1, 'cqt_note', hop_length, 'Constant-Q transf
orm ' + name1)
    feat_specshow(S_db2, sr2, 'cqt_note', hop_length, 'Constant-Q transf
orm ' + name2)
  if method == 'mel':
    ##### MFCC
    S_db1 = feat_mel(y1, sr1, n_mfcc=40)
    S_db2 = feat_mel(y2, sr2, n_mfcc=40)
    feat_specshow(S_db1, sr1, 'mel', hop_length, 'Mel-spectrogram ' + na
me1)
    feat_specshow(S_db2, sr2, 'mel', hop_length, 'Mel-spectrogram ' + na
me2)
  if method == 'chroma':
    ##### Chroma
    S_db1 = feat_chroma(y1, sr1)
```

```
    S_db2 = feat_chroma(y2, sr2)
    feat_specshow(S_db1, sr1, 'chroma', hop_length, 'Chromagram' + name1
)
    feat_specshow(S_db2, sr2, 'chroma', hop_length, 'Chromagram' + name2
)
  if method == 'gfcc':
    S_db1 = feat_gfcc(y1, sr1)
    S_db2 = feat_gfcc(y2, sr2)
    vis.visualize_features(S_db1, 'GFCC Index' + name1, 'Frame Index')
    vis.visualize_features(S_db2, 'GFCC Index' + name2, 'Frame Index')
  if method == 'bfcc':
    S_db1 = feat_bfcc(y1, sr1)
    S_db2 = feat_bfcc(y2, sr2)
    vis.visualize_features(S_db1, 'BFCC Index' + name1, 'Frame Index')
    vis.visualize_features(S_db2, 'BFCC Index' + name2, 'Frame Index')
  if method == 'lfcc':
    S_db1 = feat_lfcc(y1, sr1)
    S_db2 = feat_lfcc(y2, sr2)
    vis.visualize_features(S_db1, 'LFCC Index' + name1, 'Frame Index')
    vis.visualize_features(S_db2, 'LFCC Index' + name2, 'Frame Index')
  if method == 'ngcc':
    S_db1 = feat_ngcc(y1, sr1)
    S_db2 = feat_ngcc(y2, sr2)
    # visualize features
    vis.visualize_features(S_db1, 'NGCC Index' + name1, 'Frame Index')
    vis.visualize_features(S_db2, 'NGCC Index' + name2, 'Frame Index')
  if method == 'msrcc':
    S_db1 = feat_msrcc(y1, sr1)
    S_db2 = feat_msrcc(y2, sr2)
    # visualize features
    vis.visualize_features(S_db1, 'MSRCC Index' + name1, 'Frame Index')
    vis.visualize_features(S_db2, 'MSRCC Index' + name2, 'Frame Index')
  if method == 'psrcc':
    S_db1 = feat_psrcc(y1, sr1)
    S_db2 = feat_psrcc(y2, sr2)
    # visualize features
    vis.visualize_features(S_db1, 'PSRCC Index' + name1, 'Frame Index')
    vis.visualize_features(S_db2, 'PSRCC Index' + name2, 'Frame Index')
  if method == 'tempo':
    S_db1 = feat_te(y1, sr1)
    S_db2 = feat_psrcc(y2, sr2)
    # visualize features
    vis.visualize_features(S_db1, 'PSRCC Index' + name1, 'Frame Index')
    vis.visualize_features(S_db2, 'PSRCC Index' + name2, 'Frame Index')
  print(np.shape(S_db1))
  print(np.shape(S_db2))
```

```
def x_features(df, wav_path, mode, method, stretch=False):
  """ Evaluating of X np.array with features
  Args:
    df: Data Frame, data frame from description csv
    wav_path: string, path to wav files
    mode: string, alias for certain dataset 'esc', 'us', 'fsd'
    method: string,  'sft', 'cqt', 'mel', 'chroma', 'gfcc', 'bfcc', 'lfc
c', 'ngcc', 'msrcc', 'psrcc'
  Returns:
    x_train np.array with features
  """
  from sklearn import preprocessing
  df = df.reset_index(drop=True)
  for i in range(len(df)):
    if mode == 'esc':
      fname = wav_path + df.loc[df.index == i]['filename'].values[0]
    if mode == 'us':
      fname = wav_path + 'fold' + str(df.loc[df.index == i]
['fold'].values[0]) + '/' + df.loc[df.index == i]
['slice_file_name'].values[0]
    if mode == 'fsd':
      fname = wav_path + df.loc[df.index == i]['fname'].values[0]
    if i % 100 == 0:
      print(str(i) + ' obs. from ' + str(len(df)) + ' are processed')
    # print(fname)
    y, sr = librosa.load(fname,sr=None)
    new_sr = 16000
    y = librosa.resample(y, orig_sr=sr, target_sr=new_sr)
    y2 = y
    # y2 = non_silent(y, sr)
    if (stretch):
      if (mode == 'us'):
        file_name = df.loc[df.index == i]['slice_file_name'].values[0]
        print(file_name)
        old_time = df.loc[df.slice_file_name == file_name]
['time'].values[0]
        print(old_time)
        if (old_time!= 4.0):
          # old_time = df.loc[df.slice_file_name == fname]
['time'].values[0]
          rate = old_time / 4
          y2 = librosa.effects.time_stretch(y2, rate=rate)
          print(np.shape(y2))
      if (mode == 'fsd'):
        file_name = df.loc[df.index == i]['fname'].values[0]
        print(file_name)
```

```
        old_time = df.loc[df.fname == file_name]['time'].values[0]
        print(old_time)
        if (old_time!= 10.0):
          # old_time = df.loc[df.slice_file_name == fname]
['time'].values[0]
          rate = old_time / 10
          y2 = librosa.effects.time_stretch(y2, rate=rate)
          print(np.shape(y2))
      if method == 'sft':
       spec = feat_sft(y2)
      if method == 'cqt':
       spec = feat_cqt(y2, new_sr)
    if method == 'mel':
      spec = feat_mel(y2, new_sr, n_mfcc=40)
    if method == 'chroma':
      spec = feat_chroma(y2, new_sr)
    if method == 'gfcc':
      y2 = np.reshape(y2, (np.shape(y2)[0],1))
      spec = feat_gfcc(y2, new_sr)
    if method == 'bfcc':
      y2 = np.reshape(y2, (np.shape(y2)[0],1))
      spec = feat_bfcc(y2, new_sr)
    if method == 'lfcc':
      y2 = np.reshape(y2, (np.shape(y2)[0],1))
      spec = feat_lfcc(y2, new_sr)
    if method == 'ngcc':
      y2 = np.reshape(y2, (np.shape(y2)[0],1))
      spec = feat_ngcc(y2, new_sr)
    if method == 'msrcc':
      y2 = np.reshape(y2, (np.shape(y2)[0],1))
      spec = feat_msrcc(y2, new_sr)
    if method == 'psrcc':
      y2 = np.reshape(y2, (np.shape(y2)[0],1))
      spec = feat_psrcc(y2, new_sr)
    scaler = preprocessing.StandardScaler().fit(spec)
    spec = scaler.transform(spec)
    print(np.shape(y2))
    print(np.shape(spec))
    if i == 0:
      x_train = np.empty(shape=(df.shape[0], spec.shape[0], spec.shape[1
], 1))
    spec = np.reshape(spec,(1, spec.shape[0],spec.shape[1], 1))
    x_train[i,] = spec
  return x_train, scaler
```

Формування вибірок тренування та тестування

```
def train_test_(df, target, test_size):
  """ Train\test splitting
  Args:
    df: Data Frame, data frame from description csv
    target: string, column name with category code
    test_size: float, train\test ratio
  Returns:
    train\test dataframes
  """
  split = StratifiedShuffleSplit(n_splits=1, test_size=test_size, rando
m_state=42)
  for train_index, test_index in split.split(df,df[target]):
    strat_train = df.loc[train_index]
    strat_test = df.loc[test_index]
  strat_train = strat_train.dropna()
  strat_test = strat_test.dropna()
  return strat_train, strat_test
```

Функції обчислення точності прогнозування

```
def f1(y_true, y_pred):
  y_pred = K.round(y_pred)
  tp = K.sum(K.cast(y_true*y_pred, 'float'), axis=0)
  tn = K.sum(K.cast((1-y_true)*(1-y_pred), 'float'), axis=0)
  fp = K.sum(K.cast((1-y_true)*y_pred, 'float'), axis=0)
  fn = K.sum(K.cast(y_true*(1-y_pred), 'float'), axis=0)
  p = tp / (tp + fp + K.epsilon())
  r = tp / (tp + fn + K.epsilon())
  f1 = 2*p*r / (p+r+K.epsilon())
  # f1 = tf.where(tf.is_nan(f1), tf.zeros_like(f1), f1)
  return K.mean(f1)
def f1_loss(y_true, y_pred):
  tp = K.sum(K.cast(y_true*y_pred, 'float'), axis=0)
  tn = K.sum(K.cast((1-y_true)*(1-y_pred), 'float'), axis=0)
  fp = K.sum(K.cast((1-y_true)*y_pred, 'float'), axis=0)
  fn = K.sum(K.cast(y_true*(1-y_pred), 'float'), axis=0)
  p = tp / (tp + fp + K.epsilon())
  r = tp / (tp + fn + K.epsilon())
  f1 = 2*p*r / (p+r+K.epsilon())
```

```
  # f1 = tf.where(tf.is_nan(f1), tf.zeros_like(f1), f1)
  return 1 - K.mean(f1)
def accuracy_(conf_matrix):
  acc = []
  true_positive_rate  = []
  positive_predictive_value  = []
  f1 = []
  for i in range(len(conf_matrix)):
    TN = conf_matrix[i][0][0]
    TP = conf_matrix[i][1][1]
    FN = conf_matrix[i][1][0]
    FP = conf_matrix[i][0][1]
    true_positive_rate.append(TP/(TP+FN))
    positive_predictive_value.append(TP/(TP+FP))
    f1.append(2*TP/(2*TP+FP+FN))
    acc.append((TP)/(TN+TP+FN+FP))
  return acc, true_positive_rate, positive_predictive_value, f1
```

Функція обчислення шару «уваги».

```
def attention_3d_block(hidden_states):
  """
  Many-to-one attention mechanism for Keras.
  @param hidden_states: 3D tensor with shape (batch_size, time_steps, i
nput_dim).
  @return: 2D tensor with shape (batch_size, 128)
  @author: felixhao28.
  https://github.com/philipperemy/keras-attention-mechanism
  """
  hidden_size = int(hidden_states.shape[2])
  # Inside dense layer
  #              hidden_states            dot               W
 =>           score_first_part
  # (batch_size, time_steps, hidden_size) dot (hidden_size, hidden_size
) => (batch_size, time_steps, hidden_size)
  # W is the trainable weight matrix of attention Luong's multiplicativ
e style score
  score_first_part = Dense(hidden_size, use_bias=False, name='attention
_score_vec')(hidden_states)
  #            score_first_part           dot        last_hidden_state
   => attention_weights
  # (batch_size, time_steps, hidden_size) dot   (batch_size, hidden_siz
e)  => (batch_size, time_steps)
```

```
  h_t = Lambda(lambda x: x[:, -1, :], output_shape=(hidden_size,), name
='last_hidden_state')(hidden_states)
  score = dot([score_first_part, h_t], [2, 1], name='attention_score')
  attention_weights = Activation('softmax', name='attention_weight')
(score)
  # (batch_size, time_steps, hidden_size) dot (batch_size, time_steps)
=> (batch_size, hidden_size)
  context_vector = dot([hidden_states, attention_weights], [1, 1], name
='context_vector')
  pre_activation = concatenate([context_vector, h_t], name='attention_o
utput')
  attention_vector = Dense(512, use_bias=False, name='attention_vector'
)(pre_activation)
  return attention_vector
```

Функції створення моделі класифікатора

```
def create_block(mode, input, ch):
  if mode == 0:
    x = Conv2D(ch, 3, padding="same")(input)
    x = BatchNormalization(trainable=True)(x)
    return Activation("relu")(x)
  elif mode == 1:
    x = DepthwiseConv2D(3, padding="same")(input)
    x = BatchNormalization(trainable=True)(x)
    return Activation("relu")(x)
  elif mode == 2:
    x = SeparableConv2D(ch, 3, padding="same")(input)
    x = BatchNormalization(trainable=True)(x)
    return Activation("relu")(x)
def create_adjusting_block(input, ch):
  x = Conv2D(ch, 3, padding="same")(input)
  x = BatchNormalization(trainable=True)(x)
  x = Activation("relu")(x)
  x = Conv2D(ch, 3, padding="same")(x)
  x = BatchNormalization(trainable=True)(x)
  return Activation("relu")(x)
def create_model(mode):
  input = Input(shape=(input_shape))
  x = create_adjusting_block(input, 64)
  x = create_block(mode, x, 64)
  x = MaxPooling2D(2)(x)
  x = create_adjusting_block(x, 128)
  x = create_block(mode, x, 128)
```

```
  x = MaxPooling2D(2)(x)
  x = create_adjusting_block(x, 256)
  x = create_block(mode, x, 256)
  #x = GlobalMaxPooling2D()(x)
  x = Flatten()(x)
  x = Dense(256, activation="relu")(x)
  x = Dense(256, activation="relu")(x)
  x = Dense(256, activation="relu")(x)
  x = Dropout(0.5)(x)
  x = Dense(n_classes, activation="softmax")(x)

  model = Model(input, x)
  return model

model = create_model(mode)
```

Функція побудови автокодувальника

```
def conv_autoencoder(xTrain_input, xTest_input):
  """Creating autoencoder. Compiling and fitting
  Args:
    df: The dataset after preprocessing
  Returns:
    history: The history of model fitting
    autoencoder: Keras model
  """
  batch_size = 64
  kernel_initializer=initializers.lecun_normal()
  bias_initializer='zeros'
  kernel_regularizer=None
  activation = 'relu'
  data_rows = xTrain_input.shape[1]
  data_cols = xTrain_input.shape[2]
  n_epoch = 100
  input_shape = (data_rows, data_cols, 1)
  n_units = data_rows*data_cols
  encoding_dim = np.round(n_units)
  loss = 'mean_squared_error'
  # Encoder
  input_img = Input(shape=input_shape)
  x = Conv2D(32, (3,6), padding='same',
       data_format="channels_last", kernel_initializer=kernel_initializ
er,
```

```
      bias_initializer=bias_initializer, kernel_regularizer=kernel_reg
ularizer)(input_img)
  x = LeakyReLU(0.2)(x)
  x = AveragePooling2D((2,4), padding='same')(x)
  # x = BatchNormalization(x)
  x = Conv2D(32, (3,6), padding='same',
       kernel_initializer=kernel_initializer,
       bias_initializer=bias_initializer, kernel_regularizer=kernel_reg
ularizer)(x)
  x = LeakyReLU(0.2)(x)
  x = AveragePooling2D((2,4), padding='same')(x)
  # x = BatchNormalization(x)
  x = Conv2D(32, (3,6), padding='same',
     kernel_initializer=kernel_initializer,
     bias_initializer=bias_initializer, kernel_regularizer=kernel_regul
arizer)(x)
  x = LeakyReLU(0.2)(x)
  x = AveragePooling2D((2,4))(x)
  # x = BatchNormalization(x)
  x = Conv2D(32, (3,6), padding='same',
       kernel_initializer=kernel_initializer,
       bias_initializer=bias_initializer, kernel_regularizer=kernel_reg
ularizer)(x)
  x = LeakyReLU(0.2)(x)
  encoded = AveragePooling2D((2,1), padding='same')(x)
  # Decoder
  x = Conv2D(32, (3,6), padding='same',
       kernel_initializer=kernel_initializer,
       bias_initializer=bias_initializer, kernel_regularizer=kernel_reg
ularizer)(encoded)
  x = LeakyReLU(0.2)(x)
  # x = AttentionAugmentation2D(1, 30, 1)(x)
  x = UpSampling2D((2,4))(x)
  x = Conv2D(32, (3,6), padding='same',
     kernel_initializer=kernel_initializer,
     bias_initializer=bias_initializer, kernel_regularizer=kernel_regul
arizer)(x)
  x = LeakyReLU(0.2)(x)
  x = UpSampling2D((2,3))(x)
  x = Conv2D(32, (3,6), padding='same',
       kernel_initializer=kernel_initializer,
       bias_initializer=bias_initializer, kernel_regularizer=kernel_reg
ularizer)(x)
  x = LeakyReLU(0.2)(x)
  x = UpSampling2D((2,3))(x)
  x = Conv2D(32, (3,6), padding='same',
```

```
        kernel_initializer=kernel_initializer,
        bias_initializer=bias_initializer, kernel_regularizer=kernel_reg
ularizer)(x)
  x = LeakyReLU(0.2)(x)
  x = UpSampling2D((3,2))(x)
  decoded = Conv2D(1, kernel_size=(5,2), strides=1, padding='valid',
          kernel_initializer=kernel_initializer,
          bias_initializer=bias_initializer, kernel_regularizer=kernel_r
egularizer)(x)
  # decoded = augmented_conv2d(decoded, filters=1)
  # Model composing
  autoencoder = models.Model(input_img, decoded, name="autoencoder")
  # optimizer = optimizers.Nadam(learning_rate=0.001, beta_1=0.9, beta_
2=0.999)
  optimizer = optimizers.Adam(learning_rate=0.001, beta_1=0.9, beta_2=0
.999, amsgrad=True)
  # optimizer = optimizers.Ftrl(learning_rate=0.0001, learning_rate_pow
er=-0.5, initial_accumulator_value=0.1, l1_regularization_strength=0.0,
l2_regularization_strength=0.0)
  autoencoder.compile(optimizer=optimizer, loss=loss)
  autoencoder.summary()
  history = autoencoder.fit(xTrain_input, xTrain_input,
              epochs=n_epoch,
              batch_size=batch_size, verbose=2,
              shuffle=True,validation_data=(xTest_input, xTest_input) )
  model = autoencoder
  return history, model
```

Функції реалізації генетичного алгоритму

```
class Genom:
  """ Represents genes and genom structure.
  """
  def __init__(self):
    """ Chromosome example with 1xConv2D layer:
      [1, 3, 150, 4, 0, 28, 2, 1, 1, 0, 1, 0, 0]
    """
    self.batch_size = {0: 8, 1: 16, 2: 32, 3: 64, 4: 128, 5: 256, 6: 512
}
    self.learning_rate = {0: 0.1, 1: 0.01, 2: 0.001, 3: 0.0001, 4: 0.000
01}
    self.epochs = {0: 10, 1: 300}
    self.optimizer = {0: 'Adam', 1: 'Adadelta', 2: 'Adagrad', 3: 'Adamax
',
            4: 'Nadam'}
```

```
    self.layer_type = {0: 'Conv2D', 1: 'Dense', 2: 'MaxPolling2D',
            3: 'AveragePooling2D', 4: 'Dropout', 5: 'Attention3D'}
    self.units_filters = {0: 8, 1: 1024}
    self.stride = {0: 2, 1: 4, 2: 6}
    self.kernel_size = {0: 2, 1: 4, 2: 6}
    self.init = {0: 'RandomNormal', 1: 'RandomUniform', 2: 'Zeros',
          3: 'Ones', 4: 'GlorotNormal', 5: 'GlorotUniform', 6: 'he_norma
l',
          7: 'he_uniform', 8: 'lecun_normal', 9: 'lecun_uniform'}
    self.regular = {0: 'l1', 1: 'l2', 2: 'l1_l2', 3: None}
    self.activation = {0: 'relu', 1: 'sigmoid', 2: 'softmax', 3: 'softpl
us',
            4: 'softsign', 5: 'tanh', 6: 'selu', 7: 'elu'}
```

class Individual:

```
  """ Represents a chromosome in GA scope.
  """
   def __init__(self, phenotype=None):
    self.phenotype = phenotype
    self.model = None
    self.score = 0
    self.n_classes = 50
  def generate(self, input_shape):
    """Creating keras model from phenopype
    Returns:
      model: Keras model
    """
    param_list = self.phenotype
    gen = Genom()
    self.batch_size = gen.batch_size[param_list[0]]
    print(self.batch_size)
    learning_rate = gen.learning_rate[param_list[1]]
    self.n_epochs = param_list[2]
    if gen.optimizer[param_list[3]] == 'Adam':
      optimizer = optimizers.Adam(learning_rate=learning_rate, beta_1=0.
9, beta_2=0.999, amsgrad=False)
    if gen.optimizer[param_list[3]] == 'Adadelta':
      optimizer = optimizers.Adadelta(learning_rate=learning_rate, rho=0
.95, epsilon=1e-07)
    if gen.optimizer[param_list[3]] == 'Adagrad':
      optimizer = optimizers.Adagrad(learning_rate=learning_rate, initia
l_accumulator_value=0.1, epsilon=1e-07)
    if gen.optimizer[param_list[3]] == 'Adamax':
      optimizer = optimizers.Adamax(learning_rate=learning_rate, beta_1=
0.9, beta_2=0.999, epsilon=1e-07)
    if gen.optimizer[param_list[3]] == 'Nadam':
```

```
      optimizer = optimizers.Nadam( learning_rate=learning_rate, beta_1=
0.9, beta_2=0.999, epsilon=1e-07)
    inp = Input(shape = (input_shape))
    x = inp
    for i in range(4, len(param_list), 9):
      print('i = ', i)
      if gen.layer_type[param_list[i]] == 'Conv2D':
        print(gen.layer_type[param_list[i]])
        print(param_list[i+1])
        filters = param_list[i+1]
        kernel_size = gen.kernel_size[param_list[i+2]]
        stride = gen.stride[param_list[i+3]]
        kernel_init = gen.init[param_list[i+4]]
        bias_init = gen.init[param_list[i+5]]
        kernel_reg = gen.regular[param_list[i+6]]
        bias_reg = gen.regular[param_list[i+7]]
        activation = gen.activation[param_list[i+7]]
        x = Conv2D(filters, (kernel_size, kernel_size),
             strides = (1, 1), padding="same",
             data_format="channels_last", activation = activation,
             kernel_initializer=kernel_init,
             bias_initializer=bias_init,
             kernel_regularizer=kernel_reg,
             bias_regularizer=bias_reg)(x)
      if gen.layer_type[param_list[i]] == 'Dense':
        units = param_list[i+1]
        # kernel_size = param_list[i+2]
        # stride = param_list[i+3]
        kernel_init = param_list[i+4]
        bias_init = param_list[i+5]
        kernel_reg = param_list[i+6]
        bias_reg = param_list[i+7]
        activation = param_list[i+7]
        x = Flatten()(x)
        x = Dense(unuts, activation = activation,
             kernel_initializer=kernel_init,
             bias_initializer=bias_init,
             kernel_regularizer=kernel_reg,
             bias_regularizer=bias_reg)(x)
      if gen.layer_type[param_list[i]] == 'MaxPolling2D':
        # filters = param_list[i+1]
        kernel_size = param_list[i+2]
        # stride = param_list[i+3]
        # kernel_init = param_list[i+4]
        # bias_init = param_list[i+5]
        # kernel_reg = param_list[i+6]
```

```
        # bias_reg = param_list[i+7]
        # activation = param_list[i+7]
        x = MaxPooling2D((kernel_size, kernel_size))(x)
      if gen.layer_type[param_list[i]] == 'AveragePooling2D':
        # filters = param_list[i+1]
        kernel_size = param_list[i+2]
        # stride = param_list[i+3]
        # kernel_init = param_list[i+4]
        # bias_init = param_list[i+5]
        # kernel_reg = param_list[i+6]
        # bias_reg = param_list[i+7]
        # activation = param_list[i+7]
        x = AveragePooling2D((kernel_size, kernel_size))(x)
      if gen.layer_type[param_list[i]] == 'Dropout':
        filters = param_list[i+1]
        # kernel_size = param_list[i+2]
        # stride = param_list[i+3]
        # kernel_init = param_list[i+4]
        # bias_init = param_list[i+5]
        # kernel_reg = param_list[i+6]
        # bias_reg = param_list[i+7]
        # activation = param_list[i+7]
        x = Dropout(1 - filters/10)(x)
      if gen.layer_type[param_list[i]] == 'Attention3D':
        pass
        out = Flatten()(x)
    out = Dense(256, kernel_initializer=kernel_init, bias_initializer=bi
as_init)(out)
    out = Activation('sigmoid')(out)
    out = Dense(self.n_classes, kernel_initializer=kernel_init, bias_ini
tializer=bias_init)(out)
    model = models.Model(inputs = inp, outputs = out)
    print(model.summary())
    # model.compile(loss='categorical_crossentropy', metrics=['accuracy'
], optimizer=optimizer)
    self.model = model
    del model
    return 0
  def fitness(self, data:Dataset):
    history = self.model.fit(data.train[0], data.train[1],
        batch_size=self.batch_size,
        epochs=self.n_epochs,
        shuffle=False,validation_data = (data.valid[0], data.valid[1]))
    y_pred = self.model.predict(data.test[0])
    y_true = data.test[1]
    from sklearn.metrics import confusion_matrix
```

```
    y_pred1 = []
    y_true1 = []
    y_pred1 = [y1.argmax() for y1 in y_pred]
    y_true1 = [y2.argmax() for y2 in y_true]
    matr = confusion_matrix(y_true1, y_pred1)
    mean_acc = []
    for i in range(self.n_classes):
      mean_acc.append(matr[i][i]/sum(matr[i]))
     del self.model
    import gc
    import torch
    for i in range(20):
      gc.collect()
      torch.cuda.empty_cache()
      K.clear_session()
    self.score = np.mean(mean_acc)
    return 0
```

```
class Network():
  def __init__(self):
    self._epochs = np.random.randint(1, 15)
    self._units1 = np.random.randint(1, 500)
    self._units2 = np.random.randint(1, 500)
    self._act1 = random.choice(['sigmoid', 'relu', 'softmax', 'tanh', 'e
lu', 'selu', 'linear'])
    self._act2 = random.choice(['sigmoid', 'relu', 'softmax', 'tanh', 'e
lu', 'selu', 'linear'])
    self._act3 = random.choice(['sigmoid', 'relu', 'softmax', 'tanh', 'e
lu', 'selu', 'linear'])
    self._loss = random.choice([
      'categorical_crossentropy',
      'binary_crossentropy',
      'mean_squared_error',
      'mean_absolute_error',
      'sparse_categorical_crossentropy'
    ])
    self._opt = random.choice(['sgd', 'rmsprop', 'adagrad', 'adadelta',
'adam', 'adamax', 'nadam'])
    self._accuracy = 0
  def init_hyperparams(self):
    hyperparams = {
      'epochs': self._epochs,
      'units1': self._units1,
      'act1': self._act1,
      'units2': self._units2,
      'act2': self._act2,
```

```
      'act3': self._act3,
      'loss': self._loss,
      'optimizer': self._opt
    }
    return hyperparams
def init_networks(population):
  return [Network() for _ in range(population)]
def fitness(networks):
  for network in networks:
    hyperparams = network.init_hyperparams()
    epochs = hyperparams['epochs']
    units1 = hyperparams['units1']
    act1 = hyperparams['act1']
    units2 = hyperparams['units2']
    act2 = hyperparams['act2']
    act3 = hyperparams['act3']
    loss = hyperparams['loss']
    opt = hyperparams['optimizer']
    try:
      model = serve_model(epochs, units1, act1, units2, act2, classes, a
ct3, loss, opt, x_train, y_train)
      accuracy = model.evaluate(x_test, y_test, verbose=0)[1]
      network._accuracy = accuracy
      print ('Accuracy: {}'.format(network._accuracy))
    except:
      network._accuracy = 0
      print ('Build failed.')
  return networks
def selection(networks):
  networks = sorted(networks, key=lambda network: network._accuracy, re
verse=True)
  networks = networks[:int(0.2 * len(networks))]
  return networks
def crossover(networks):
  offspring = []
  for _ in range(int((population - len(networks)) / 2)):
    parent1 = random.choice(networks)
    parent2 = random.choice(networks)
    child1 = Network()
    child2 = Network()
    # Crossing over parent hyper-params
    child1._epochs = int(parent1._epochs/4) + int(parent2._epochs/2)
    child2._epochs = int(parent1._epochs/2) + int(parent2._epochs/4)
    child1._units1 = int(parent1._units1/4) + int(parent2._units1/2)
    child2._units1 = int(parent1._units1/2) + int(parent2._units1/4)
    child1._units2 = int(parent1._units2/4) + int(parent2._units2/2)
```

```
    child2._units2 = int(parent1._units2/2) + int(parent2._units2/4)
    child1._act1 = parent2._act2
    child2._act1 = parent1._act2
    child1._act2 = parent2._act1
    child2._act2 = parent1._act1
    child1._act3 = parent2._act2
    child2._act3 = parent1._act2
    offspring.append(child1)
        offspring.append(child2)
  networks.extend(offspring)
  return networks
def mutate(networks):
  for network in networks:
    if np.random.uniform(0, 1) <= 0.1:
      network._epochs += np.random.randint(0,100)
      network._units1 += np.random.randint(0,100)
      network._units2 += np.random.randint(0,100)
  return networks
def main():
  networks = init_networks(population)
  for gen in range(generations):
    print ('Generation {}'.format(gen+1))
    networks = fitness(networks)
    networks = selection(networks)
    networks = crossover(networks)
    networks = mutate(networks)
    for network in networks:
      if network._accuracy > threshold:
        print ('Threshold met')
        print (network.init_hyperparams())
        print ('Best accuracy: {}'.format(network._accuracy))
        exit(0)
```