Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Гібридна система детекції та інтерпретації ознак синтезованого мовлення за сукупністю спектрально-фазових та SSL-характеристик
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-22-1
Володимир РОСТОВЦЕВ
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)

Керівник ас. Сергій ГРИНЬОВ
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Лариса ЧАЛА
(підпис) (власне ім'я, прізвище)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

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

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Ростовцеву Володимиру Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Гібридна система детекції та інтерпретації ознак синтезованого мовлення за сукупністю спектрально-фазових та SSL-характеристик

затверджена наказом університету від 5 червня 20 26 р. № 488Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 10 липня 20 26 р.

3. Вихідні дані до роботи науково-технічні публікації та Інтернет-джерела з питань детекції синтезованого мовлення, deepfake-аудіо та anti-spoofing-систем; матеріали наборів даних ASVspoof 2019 LA, ASVspoof 2021 LA та ASVspoof5; документація моделей WavLM, wav2vec2 і RawNet2; документація з мови програмування Python, фреймворку PyTorch та бібліотек для цифрової обробки аудіосигналів.

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Обґрунтування обраних технологій та методів розробки
3) Програмна реалізація гібридної системи детекції синтезованого мовлення
4) Експериментальне дослідження та аналіз результатів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.06.2026	виконано
2	Аналіз предметної галузі	09.06.2026 – 12.06.2026	виконано
3	Обгрунтування обраних технологій та методів		
	розробки	13.06.2026 – 14.06.2026	виконано
4	Реалізація модулів попередньої обробки аудіо,		
	виділення ознак, навчання та оцінювання моделі	14.06.2026 – 16.06.2026	виконано
5	Проведення експериментального дослідження	17.06.2026 – 19.06.2026	виконано
6	Аналіз отриманих результатів, порівняння		
	конфігурацій моделі та проведення fine-tuning	20.06.2026 – 21.06.2026	виконано
7	Оформлення пояснювальної записки, висновків,		
	переліку джерел та додатків	22.06.2026 – 24.06.2026	виконано
8	Проходження нормконтролю	25.06.2026-26.06.2026	виконано
9	Рецензування	26.06.2026	виконано
10	Перевірка на академічний плагіат	27.06.2026-28.06.2026	виконано
11	Підготовка презентації та доповіді	28.06.2026-29.06.2026	виконано
12	Попередній захист	30.06.2026	виконано
13	Захист роботи	10.07.2026	

Дата видачі завдання 8 червня 20 26 р.

Здобувач _____
(підпис)

Керівник роботи _____ ас. Сергій ГРИНЬОВ
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 103 с., 10 рис., 6 табл., 1 дод., 25 джерел.

АВТЕНТИЧНІСТЬ АУДІО, ГІБРИДНА СИСТЕМА, ДЕТЕКЦІЯ СИНТЕЗОВАНОГО МОВЛЕННЯ, ПРОСОДИЧНІ ПАРАМЕТРИ, СПЕКТРАЛЬНІ ОЗНАКИ, ФАЗОВІ ХАРАКТЕРИСТИКИ, DEEPFAKE-АУДІО, LFCC, SSL-ХАРАКТЕРИСТИКИ, WAVLM.

Об’єкт дослідження – процес автоматичного виявлення синтезованого мовлення в аудіосигналах.

Предмет дослідження – методи аналізу, виділення та інтеграції спектральних, фазових, просодичних і SSL-характеристик для визначення автентичності мовного аудіосигналу.

Мета роботи – розробити концепцію гібридної системи детекції та інтерпретації ознак синтезованого мовлення, що дозволить визначати штучне походження аудіозапису на основі комплексного аналізу акустичних характеристик і нейромережевих представлень мовлення.

Методи дослідження – порівняльний аналіз сучасних підходів до детекції синтезованого мовлення, аналіз методів цифрової обробки аудіосигналів, дослідження спектральних, фазових і просодичних ознак, а також аналіз моделей самокерованого навчання.

У звіті розглянуто сучасні виклики виявлення синтезованого мовлення, пов’язані з розвитком нейронного синтезу, клонування голосу та deepfake-аудіо. Проаналізовано основні групи методів детекції, зокрема підходи на основі кепстральних ознак, нейромережевих архітектур, моделей самокерованого навчання та гібридного поєднання різних типів представлень. Обґрунтовано доцільність створення системи, яка враховує не лише спектральну структуру аудіосигналу, а й фазові, просодичні та SSL-характеристики.

ABSTRACT

Bachelor’s thesis contains: 103 pp., 10 fig., 6 tabl., 1 ann., 25 references.

AUDIO AUTHENTICITY, DEEPFAKE AUDIO, HYBRID SYSTEM, LFCC, PHASE CHARACTERISTICS, PROSODIC PARAMETERS, SPECTRAL FEATURES, SSL CHARACTERISTICS, SYNTHETIC SPEECH DETECTION, WAVLM.

Research objective – the process of automatically detecting synthesised speech in audio signals.

Research subject – methods for analysing, extracting and integrating spectral, phase, prosodic and SSL characteristics to determine the authenticity of a speech audio signal.

The aim of the work is to develop a concept for a hybrid system for the detection and interpretation of features of synthesised speech, which will enable the artificial origin of an audio recording to be determined on the basis of a comprehensive analysis of acoustic characteristics and neural network representations of speech.

Research methods – comparative analysis of modern approaches to the detection of synthesised speech, analysis of methods for digital audio signal processing, investigation of spectral, phase and prosodic features, as well as analysis of self-supervised learning models.

The report examines current challenges in synthetic speech detection caused by the rapid development of neural speech synthesis, voice cloning and deepfake audio. It reviews the main detection approaches, including cepstral features, neural network models, self-supervised learning methods and hybrid architectures. The work also justifies the use of a hybrid system that combines spectral, phase, prosodic and SSL-based characteristics of the audio signal.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 8
Вступ 9
1 Аналіз предметної галузі та постановка задачі 11
1.1 Опис предметної галузі 11
1.2 Опис проблем та викликів 13
1.3 Огляд існуючих методів детекції синтезованого мовлення 16
1.4 Інноваційність та переваги розроблюваної системи 19
1.5 Мета роботи та постановка задачі 21
2 Обґрунтування обраних технологій та методів розробки 24
2.1 Вибір програмних технологій 24
2.2 Методи підготовки та попередньої обробки аудіоданих 27
2.3 Вибір ознакових представлень аудіосигналу 29
2.4 Архітектура моделі та механізм інтеграції ознак 32
2.5 Методи навчання, оцінювання та практичної реалізації 35
3 Програмна реалізація гібридної системи детекції синтезованого мовлення 39
3.1 Архітектура системи 39
3.2 Підготовка та попередня обробка аудіоданих 42
3.3 Формування спектральних ознак LFCC 46
3.4 Отримання SSL-представлень мовлення 50
3.5 Фазова гілка 54
3.6 Просодична гілка 56
3.7 RawNet2 гілка 57
3.8 Інтеграція ознак і класифікаційний модуль 60
3.9 Програмна реалізація системи 63
3.10 Навчання моделі 68
3.11 Реалізація серверної частини та користувацького інтерфейсу 73
4 Експериментальне дослідження та аналіз результатів 78

4.1 Використані набори даних 78
4.2 Конфігурація експериментів 81
4.3 Результати на ASVspoof 2019 LA 84
4.4 Дослідження на ASVspoof 2021 LA 87
4.5 Перевірка перенесення моделі на ASVspoof5 90
4.6 Fine-tuning моделі на ASVspoof5 93
Висновки 96
Перелік джерел посилання 100
Додаток А Відомість кваліфікаційної роботи 103

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ASV – Automatic Speaker Verification – автоматична верифікація мовця;

AUC ROC – Area Under the Receiver Operating Characteristic Curve – площа під ROC-кривою;

CNN – Convolutional Neural Network – згорткова нейронна мережа;

CQCC – Constant-Q Cepstral Coefficients – кепстральні коефіцієнти на основі Constant-Q перетворення;

EER – Equal Error Rate – рівень рівної помилки;

LA – Logical Access – сценарій логічного доступу в наборах ASVspoof;

LFCC – Linear Frequency Cepstral Coefficients – лінійні частотні кепстральні коефіцієнти;

MFCC – Mel-Frequency Cepstral Coefficients – мел-частотні кепстральні коефіцієнти;

MLP – Multilayer Perceptron – багатошаровий перцептрон;

RawNet2 – нейромережева архітектура для аналізу аудіосигналу безпосередньо у хвильовій формі;

ROC – Receiver Operating Characteristic – характеристична крива приймача;

SSL – Self-Supervised Learning – самонавчання без ручної розмітки даних;

TTS – Text-to-Speech – синтез мовлення з тексту;

VC – Voice Conversion – перетворення голосу;

wav2vec2 – self-supervised learning модель для отримання мовленнєвих представлень;

WavLM – self-supervised learning модель для аналізу мовленнєвих сигналів.

ВСТУП

Розвиток генеративних технологій штучного інтелекту суттєво змінив підходи до створення мовного аудіоконтенту. Сучасні системи синтезу мовлення здатні формувати голосові записи, які за звучанням наближаються до природного людського мовлення. Такі технології використовуються у голосових асистентах, сервісах автоматичного озвучування текстів, медіаплатформах, системах доступності, телекомунікаційних застосунках та персоналізованих цифрових сервісах.

Разом із корисними можливостями поширення синтезованого та клонованого голосу створює нові ризики. Штучно згенероване мовлення може застосовуватися для підробки голосових повідомлень, обходу систем голосової автентифікації, шахрайських дзвінків, маніпуляції аудіозаписами та поширення дезінформації. Через це зростає потреба у засобах, які здатні автоматично перевіряти автентичність мовного сигналу та виявляти ознаки його штучного походження.

Складність цієї задачі пов’язана з тим, що сучасні моделі синтезу дедалі точніше відтворюють тембр, ритм, інтонацію та загальну структуру природного мовлення. У результаті очевидні артефакти, які раніше дозволяли легко відрізнити синтезований голос від справжнього, стають менш помітними. Тому для надійної детекції недостатньо орієнтуватися лише на суб’єктивне сприйняття або на одну групу акустичних ознак.

Виявлення синтезованого мовлення потребує комплексного аналізу аудіосигналу. Потенційні відмінності можуть проявлятися у спектральній структурі, фазовій узгодженості, часовій стабільності, просодичних особливостях та високорівневих представленнях мовлення. Кожна з цих груп характеристик описує сигнал з різного боку, тому їх поєднання дозволяє отримати більш повну картину його походження.

У зв’язку з цим доцільним є використання гібридного підходу, який поєднує класичні методи цифрової обробки сигналів із сучасними

нейромережевими представленнями. Спектральні та фазові характеристики можуть допомагати виявляти локальні акустичні артефакти, тоді як SSL-представлення дають змогу враховувати ширший контекст мовлення. Така комбінація є перспективною для побудови систем, здатних працювати з різними типами синтезованого аудіо.

Під час передатестаційної практики було опрацьовано матеріали, необхідні для подальшого виконання кваліфікаційної роботи, розглянуто сучасні виклики детекції синтезованого мовлення, проаналізовано основні групи методів, визначено переваги гібридного підходу та сформовано загальну логіку побудови системи на основі спектрально-фазових і SSL-характеристик.

Практичне значення такої системи полягає у можливості її застосування для перевірки достовірності аудіоконтенту, захисту голосових біометричних сервісів, аналізу потенційно підроблених записів і підвищення рівня інформаційної безпеки. Автоматизація виявлення синтезованого мовлення може стати важливим інструментом у цифровому середовищі, де якість штучно створеного голосу постійно зростає.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис предметної галузі

Сфера детекції синтезованого мовлення знаходиться на перетині цифрової обробки сигналів, машинного навчання, мовних технологій та інформаційної безпеки. Її основне завдання полягає у визначенні походження аудіосигналу, чи був він сформований природним мовцем, чи створений або модифікований за допомогою алгоритмів синтезу, перетворення або клонування голосу.

У сучасних системах синтезу мовлення зазвичай використовуються нейромережеві моделі, які перетворюють текст або проміжне акустичне представлення у мовний сигнал. Важливу роль у цьому процесі відіграють вокодери, які відповідають за відновлення хвильової форми аудіо. Наприклад, HiFi-GAN демонструє можливість ефективного високоякісного синтезу мовлення на основі генеративно-змагального підходу [1]. Саме розвиток таких моделей робить задачу автоматичного виявлення синтезованого аудіо технічно складною.

У задачах детекції мовлення часто використовується поділ аудіозаписів на два класи – bonafide та spoof. Клас bonafide відповідає справжньому природному мовленню, тоді як spoof позначає аудіо, створене або змінене за допомогою синтезу, voice conversion чи інших методів підробки. Така постановка використовується у benchmark-наборі ASVspoof 2019, який призначений для оцінювання систем виявлення синтезованого, перетвореного та відтвореного мовлення [2].

Аудіосигнал може аналізуватися на різних рівнях. Найнижчий рівень пов’язаний із безпосередніми акустичними характеристиками: амплітудою, частотною структурою, фазою, енергією та часовою динамікою. На цьому рівні застосовуються методи цифрової обробки сигналів, які дозволяють отримати числові ознаки для подальшого аналізу. До таких ознак належать,

зокрема, кепстральні коефіцієнти, які використовуються для опису спектральної структури мовлення.

Спектральні ознаки є важливою складовою аналізу, оскільки синтезоване мовлення може містити відмінності у розподілі енергії за частотами. Наприклад, артефакти можуть проявлятися у високочастотній області, у надмірно згладжених ділянках спектра або в нетиповій регулярності гармонічної структури. Саме тому в системах детекції використовуються ознаки, здатні відображати частотні властивості сигналу, зокрема LFCC та інші кепстральні представлення.

Фазові характеристики доповнюють спектральний аналіз, оскільки дозволяють враховувати не лише розподіл енергії за частотами, а й часову узгодженість компонентів сигналу. У природному мовленні фазова структура формується фізичними процесами мовлення, тоді як під час синтезу вона може відтворюватися неточно. Тому аналіз фазової стабільності може бути корисним для виявлення локальних порушень, які не завжди помітні лише за амплітудним спектром.

Окрему групу становлять просодичні характеристики. Вони описують інтонацію, ритм, основну частоту, мікроколивання голосу та амплітудну варіативність. Для природного мовлення характерна певна нерівномірність і варіативність цих параметрів, тоді як синтезоване мовлення може демонструвати надмірну згладженість або регулярність (рисунок 1.1). Через це просодичні ознаки можуть використовуватись як додаткове джерело інформації про природність мовного сигналу.

Поряд із класичними акустичними характеристиками дедалі більшого значення набувають представлення, отримані за допомогою моделей самокерованого навчання. Такі моделі навчаються на великих масивах немаркованого мовлення та формують узагальнені векторні представлення аудіосигналу. WavLM було запропоновано як self-supervised модель для широкого спектра задач обробки мовлення, що робить її придатною для отримання високорівневих аудіопредставлень [3].

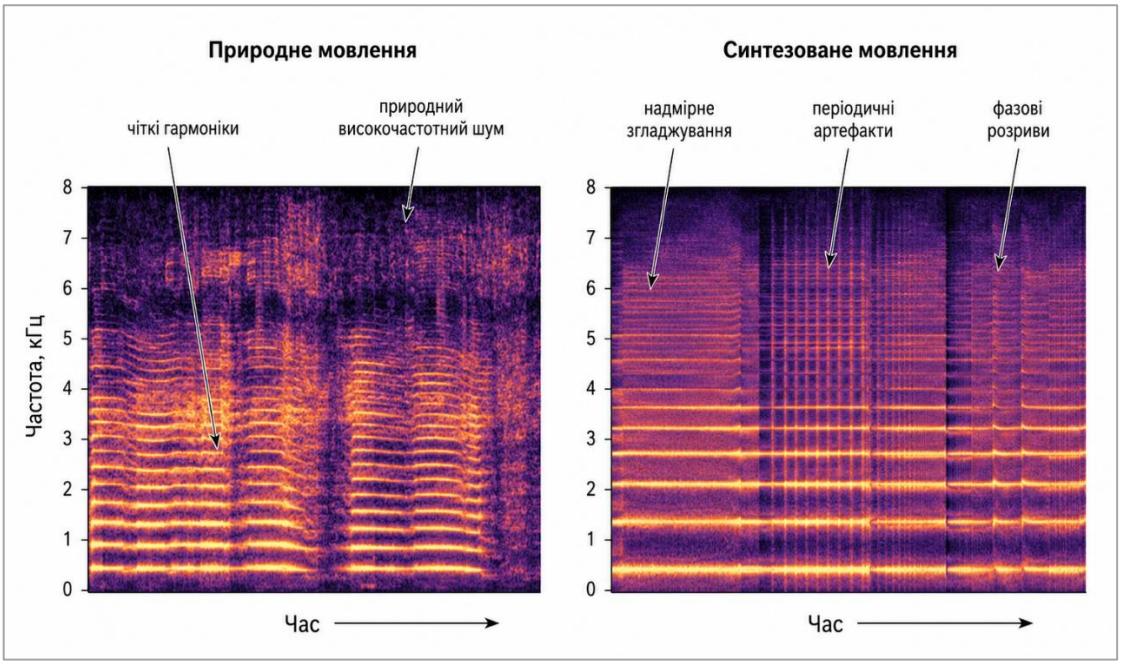


Рисунок 1.1 – Відмінності природного та синтезованого мовлення

Таким чином, предметна область детекції синтезованого мовлення охоплює як аналіз фізичних властивостей аудіосигналу, так і використання сучасних нейромережевих представлень. Спектральні, фазові, просодичні та SSL-характеристики описують мовлення з різних боків, тому їхнє поєднання створює основу для побудови більш стійких та інформативних систем виявлення синтезованого аудіо.

1.2 Опис проблем та викликів

Одним із головних викликів у задачі детекції синтезованого мовлення є швидкий розвиток самих генеративних технологій. Моделі синтезу постійно вдосконалюються, тому ознаки, які раніше дозволяли відносно легко відрізняти штучний голос від природного, поступово стають менш помітними. У результаті система детекції має працювати не лише з очевидними спотвореннями, а й з більш тонкими відмінностями в структурі аудіосигналу.

Особливо складною є проблема узагальнення на нові типи атак. Модель може показувати високі результати на тих способах синтезу, які були представлені у навчальних даних, але втрачати якість при появі нових генераторів мовлення. Це означає, що система не повинна «запам’ятовувати» лише особливості конкретного набору даних або конкретної моделі синтезу. Вона має виявляти більш загальні закономірності, які відрізняють природне мовлення від штучно сформованого.

Ще одним важливим викликом є вплив реальних умов запису та передачі аудіо. У практичних сценаріях мовлення може бути записане через мікрофон низької якості, передане через телефонний канал, стиснуте аудіокодеком або змінене під час пересилання через месенджери. Такі перетворення можуть частково приховувати корисні ознаки або створювати додаткові спотворення, які не пов’язані із синтезом. Саме тому в ASVspoof 2021 було окремо розширено увагу до складніших сценаріїв оцінювання, зокрема до deepfake-аудіо та умов, ближчих до практичного використання.

Окрему складність становить різноманіття самих атак. Синтезоване мовлення може бути створене за допомогою Text-to-Speech, voice conversion, клонування голосу або комбінованих підходів [4]. Кожен із цих варіантів може залишати різні сліди в аудіосигналі. Наприклад, одні артефакти можуть проявлятися у спектральній області, інші – у часовій структурі або просодичних характеристиках. Через це використання лише одного типу ознак може бути недостатнім для стабільної роботи системи.

Сучасні набори даних також демонструють, що задача поступово наближається до більш реалістичних умов. Наприклад, ASVspoof5 розширює постановку за рахунок «deepfake-атак», «crowdsourced speech data» та «adversarial attacks», що підкреслює зростання складності задачі детекції синтезованого мовлення [5]. Це означає, що майбутні системи

повинні бути стійкими не тільки до відомих алгоритмів синтезу, а й до нових способів створення та маскування підробленого аудіо [6].

Важливою проблемою є вибір інформативних ознак. Спектральні характеристики дозволяють аналізувати частотну структуру сигналу, але не завжди достатньо описують часову узгодженість мовлення. Фазові характеристики можуть доповнювати спектральний аналіз, однак вони є чутливими до параметрів обробки сигналу та якості запису. Просодичні параметри відображають природність інтонації, ритму й мікроваріацій голосу, але можуть залежати від мовця, емоційного стану та умов запису. Тому кожна група ознак має власні переваги й обмеження.

Ще один виклик пов’язаний з інтеграцією різнорідних представлень. Спектральні, фазові, просодичні та SSL-характеристики мають різну природу, розмірність і рівень абстракції. Їх не можна ефективно поєднати лише механічним додаванням до одного вектора без урахування їхньої інформативності. Невдале об’єднання ознак може призвести до надмірної розмірності, втрати важливої інформації або перенавчання моделі.

Не менш важливою є проблема інтерпретованості. У задачах, пов’язаних з інформаційною та біометричною безпекою, недостатньо лише отримати відповідь, що аудіозапис є справжнім або синтезованим. Важливо також мати можливість пояснити, які саме властивості сигналу вплинули на рішення системи. Це можуть бути спектральні аномалії, фазова нестабільність, неприродна просодика або особливості високорівневих представлень мовлення.

Також складним залишається коректне порівняння результатів різних досліджень. Навіть за використання одного еталонного набору результати можуть залежати від попередньої обробки аудіо, довжини фрагментів, способу балансування даних, параметрів навчання, архітектури моделі та обраних метрик. Тому числові показники потрібно аналізувати не ізольовано, а з урахуванням умов експерименту.

Таким чином, основні проблеми предметної області пов’язані з високою якістю сучасного синтезу, появою нових типів атак, впливом реальних умов запису, вибором інформативних ознак, складністю їх інтеграції та потребою в інтерпретованості результатів. Саме ці виклики обґрунтовують необхідність гібридного підходу, у якому різні групи характеристик доповнюють одна одну та дозволяють аналізувати аудіосигнал більш повно.

1.3 Огляд існуючих методів детекції синтезованого мовлення

Існуючі методи детекції синтезованого мовлення можна розглядати за тим, як саме вони представляють аудіосигнал перед класифікацією. Одні підходи спираються на заздалегідь визначені акустичні ознаки, інші використовують нейромережі для автоматичного виділення прихованих закономірностей, а сучасніші рішення дедалі частіше застосовують попередньо навчені моделі мовлення.

Одним із ранніх і поширених напрямів є використання кепстральних ознак. Такі ознаки дозволяють стисло описати спектральну структуру мовного сигналу та передати її у вигляді числового вектора.

Для задачі «anti-spoofing» було запропоновано використання Constant Q Cepstral Coefficients, CQCC, які базуються на constant-Q перетворенні та застосовувалися саме для виявлення підробок у системах автоматичної верифікації мовця [7]. Перевагою таких ознак є відносна інтерпретованість і зв’язок із фізичною структурою сигналу.

Іншим поширеним варіантом є використання Linear Frequency Cepstral Coefficients, LFCC. На відміну від ознак, побудованих із використанням мел-шкали, LFCC зберігають лінійне представлення частотної області. Це робить їх корисними тоді, коли важливо не втратити інформацію у високочастотній частині спектра. У контексті ASVspoof 2019 кепстральні ознаки та пов’язані з ними контрзаходи розглядалися як частина базових

підходів до виявлення синтезованого, перетвореного та відтвореного мовлення [8].

Окрему групу становлять моделі, які працюють безпосередньо з хвильовою формою сигналу. Їхня ідея полягає в тому, щоб не задавати всі ознаки вручну, а дозволити нейромережі самостійно виділяти інформативні закономірності з аудіо. Прикладом такого підходу є RawNet2, адаптований до задачі наскрізного виявлення підробленого або синтезованого мовлення без окремого ручного виділення ознак [9]. Перевагою таких моделей є здатність працювати з більш необробленим представленням сигналу, однак вони зазвичай потребують якісного навчання та достатньої кількості даних.

Подальший розвиток нейромережевих методів пов’язаний із використанням механізмів уваги та моделюванням спектро-часових зв’язків. Однією з відомих архітектур цього напряму є AASIST, в якій для виявлення підробленого аудіо застосовуються інтегровані графові мережі уваги для аналізу спектро-часових зв’язків [10]. Такий підхід дозволяє аналізувати не лише окремі локальні ознаки, а й зв’язки між різними ділянками сигналу, що є важливим для виявлення складніших артефактів синтезу.

Сучасний напрям розвитку систем детекції пов’язаний із моделями самокерованого навчання. На відміну від класичних ознак, такі моделі попередньо навчаються на великих масивах мовлення та формують узагальнені представлення аудіосигналу. Wav2vec 2.0 є одним із базових прикладів такого підходу, оскільки демонструє можливість отримання мовних представлень з необробленого аудіо без повної залежності від ручної розмітки [11].

Серед моделей самокерованого навчання важливе місце займає WavLM. Вона була запропонована як модель для широкого спектра задач обробки мовлення, зокрема тих, де важливо враховувати не лише локальні акустичні властивості, а й ширший контекст сигналу [3]. Для задачі детекції синтезованого мовлення такі представлення є корисними, оскільки можуть

містити інформацію, яку складно явно задати через окремі, сформовані вручну, ознаки.

Окремі сучасні роботи вже використовують WavLM безпосередньо для виявлення підробленого аудіо. Зокрема, підхід на основі multi-fusion attentive classifier – класифікатор з механізмом уваги та багаторівневим злиттям ознак, і WavLM розглядає використання SSL-представлень разом із механізмом інтеграції ознак для задачі детекції підробленого аудіо [12]. Це підтверджує загальну тенденцію, ефективні системи дедалі частіше не обмежуються одним типом ознак, а комбінують різні рівні подання аудіосигналу.

Важливим питанням у таких системах є спосіб об’єднання ознак. Якщо використовуються різнорідні представлення, наприклад акустичні параметри та SSL-ембеддинги, їх потрібно узгодити в єдиному просторі. Найпростішим способом є конкатенація, однак вона не враховує різну інформативність окремих груп ознак. Більш гнучким підходом є кероване злиття ознак, де модель може регулювати внесок різних джерел інформації. Ідея керованих мультимодальних блоків, gated multimodal units, була запропонована саме для поєднання різних модальностей у спільному представленні [13].

Таким чином, розвиток методів детекції синтезованого мовлення рухається від класичних акустичних ознак до складніших нейромережевих і SSL-підходів (таблиця 1.1).

Класичні методи залишаються корисними завдяки своїй зрозумілості та зв’язку з фізичними властивостями сигналу. Нейромережеві архітектури краще виявляють складні закономірності, а SSL-моделі дають ширше представлення мовлення. Саме тому доцільним напрямом є побудова гібридної системи, у якій різні типи характеристик не замінюють, а доповнюють одна одну.

Таблиця 1.1 – Порівняння основних підходів до детекції синтезованого мовлення

Підхід	Приклади	Основна ідея	Переваги	Обмеження
Кепстральні ознаки	CQCC, LFCC	Аналіз спектральної структури сигналу	Відносна простота та інтерпретованість	Можуть бути недостатньо стійкими до нових типів синтезу
Raw waveform	RawNet2	Аналіз хвильової форми без повного ручного задання ознак	Автоматичне виділення закономірностей із сигналу	Вищі вимоги до даних і навчання
Attention-based моделі	AASIST	Аналіз спектро-часових зв’язків за допомогою механізмів уваги	Добре враховують складні залежності в аудіо	Складніша архітектура та менша прозорість
SSL-підходи	wav2vec 2.0, WavLM	Використання попередньо навчених мовних представлень	Узагальнене подання сигналу	Висока обчислювальна складність
Гібридні системи	Акустичні ознаки + SSL-представлення	Поєднання різних рівнів аналізу	Баланс між локальними ознаками та контекстом	Потребують коректної інтеграції представлень

1.4 Інноваційність та переваги розроблюваної системи

Розроблювана система орієнтована на комплексний аналіз мовного аудіосигналу, а не на використання лише одного типу ознак [14]. Це є важливим, оскільки синтезоване мовлення може залишати сліди на різних

рівнях: у спектральній структурі, фазовій узгодженості, часовій динаміці та високорівневих представленнях сигналу. Якщо система спирається тільки на одну групу характеристик, вона може втратити частину важливої інформації, особливо у випадку нових або якісніших моделей синтезу.

Ключова ідея розроблюваного підходу полягає у поєднанні спектрально-фазових характеристик із SSL-представленнями. Спектральні ознаки дозволяють аналізувати розподіл енергії за частотами та виявляти можливі аномалії у частотній області. Фазові характеристики доповнюють цей аналіз, оскільки відображають часову узгодженість компонентів сигналу. Це важливо, тому що синтезований голос може звучати природно на рівні амплітудного спектра, але водночас містити менш очевидні порушення у фазовій або часовій структурі.

Використання SSL-характеристик дає змогу перейти від аналізу окремих локальних ознак до більш узагальненого подання мовлення. Моделі самокерованого навчання формують представлення на основі великих масивів мовних даних, тому можуть враховувати не лише низькорівневі акустичні властивості, а й ширший контекст сигналу. Зокрема, WavLM була запропонована як модель для широкого кола задач обробки мовлення, що робить її доцільною для отримання високорівневих аудіопредставлень [3].

Перевага гібридної системи полягає в тому, що різні типи ознак не замінюють один одного, а доповнюють. Спектрально-фазовий аналіз дозволяє зосередитися на фізичних властивостях аудіосигналу, тоді як SSL-представлення дають змогу врахувати більш загальні закономірності мовлення. Завдяки цьому система потенційно може бути стійкішою до різних способів синтезу, оскільки її рішення формується не за одним джерелом інформації, а на основі сукупності ознак.

Важливою перевагою є також можливість інтерпретації результатів на рівні груп характеристик. Для практичних задач недостатньо лише визначити, що аудіозапис є синтезованим. Бажано також розуміти, які саме

ознаки могли вплинути на рішення – спектральні відхилення, фазова нестабільність або особливості SSL-представлення. Такий підхід підвищує довіру до системи та робить її більш придатною для використання у сферах, де важлива обґрунтованість рішення.

Оскільки система має працювати з різнорідними ознаками, окреме значення має спосіб їх інтеграції. Просте об’єднання векторів не завжди є достатнім, адже різні групи характеристик можуть мати різну інформативність для конкретного аудіозапису. Одним із можливих рішень є використання механізмів керованого злиття ознак, які дозволяють моделі регулювати внесок окремих представлень. Ідея керованих мультимодальних блоків була запропонована саме для об’єднання різних модальностей у спільному представленні [13].

Практична цінність розроблюваної системи полягає у можливості її застосування для перевірки автентичності аудіозаписів, аналізу підозрілих голосових повідомлень, захисту голосових біометричних сервісів і виявлення підробленого аудіо. У таких сценаріях важливими є не лише точність класифікації, а й стійкість до різних умов запису та можливість пояснити отриманий результат.

Таким чином, інноваційність розроблюваної системи полягає у поєднанні кількох рівнів аналізу мовного сигналу спектрального, фазового та SSL-рівня. Такий підхід дозволяє розглядати синтезоване мовлення як складний аудіосигнал, у якому штучне походження може проявлятися не в одній конкретній ознаці, а в сукупності взаємопов’язаних характеристик.

1.5 Мета роботи та постановка задачі

Після аналізу предметної області, основних викликів та існуючих методів детекції синтезованого мовлення можна сформулювати наступну мету – розробити гібридну систему, яка поєднує різні рівні аналізу аудіосигналу. Така система має не обмежуватися лише одним типом ознак,

а враховувати спектральні, фазові та високорівневі SSL-характеристики, оскільки штучне походження мовлення може проявлятися у різних властивостях сигналу.

Доцільність гібридного підходу зумовлена тим, що окремі групи ознак описують аудіосигнал з різних боків. Спектральні характеристики дозволяють аналізувати частотну структуру мовлення, фазові ознаки відображають часову узгодженість компонентів сигналу, просодичні характеристики описують енергетично-часову поведінку, в той час як SSL-представлення дають змогу враховувати більш високорівневі мовленнєві закономірності. Водночас додаткове використання raw waveform-компонента дозволяє моделі працювати ближче до хвильової форми сигналу та виділяти закономірності, які не завжди повністю відображаються через заздалегідь сформовані акустичні ознаки.

Об’єктом дослідження є процес автоматичного виявлення синтезованого мовлення в аудіосигналах.

Предметом дослідження є методи аналізу, виділення, інтеграції та інтерпретації спектрально-фазових і SSL-характеристик для визначення автентичності мовного аудіосигналу.

Основна задача полягає у побудові гібридної системи детекції синтезованого мовлення, яка здатна виконувати бінарну класифікацію аудіозаписів на справжні та синтезовані. У межах стандартних наборів даних для цієї задачі така класифікація зазвичай подається через класи bonafide та spoof, що використовується, зокрема, у ASVspoof 2019.

Для розв’язання поставленої задачі необхідно виконати такі етапи:

– проаналізувати сучасні технології синтезу мовлення та характерні ознаки, які можуть свідчити про штучне походження аудіосигналу;

– дослідити спектральні характеристики мовлення, зокрема кепстральні ознаки, які дозволяють описувати частотну структуру сигналу;

– розглянути фазові характеристики аудіосигналу як додаткове джерело інформації про часову узгодженість і стабільність мовлення;

– проаналізувати можливості SSL-моделей для формування високорівневих представлень мовного сигналу;

– визначити спосіб поєднання різнорідних ознак у спільному просторі представлення;

– розробити загальну архітектуру системи, що забезпечує інтеграцію спектрально-фазових та SSL-характеристик;

– передбачити можливість інтерпретації результатів класифікації через аналіз внеску окремих груп ознак;

– визначити метрики та умови експериментального оцінювання якості системи.

Очікуваним результатом є формування гібридного підходу до детекції синтезованого мовлення, який дозволяє аналізувати аудіосигнал комплексно через його спектральну структуру, фазову поведінку та високорівневі нейромережеві представлення. Така постановка задачі створює основу для подальшої розробки системи, здатної не лише класифікувати аудіозапис, а й надавати більш змістовне пояснення ознак, що вплинули на прийняте рішення.

Практичне значення роботи полягає в тому, що розроблена система може бути використана як основа для перевірки автентичності мовного аудіоконтенту, аналізу потенційно підроблених записів і дослідження стійкості моделей до різних типів синтезованого мовлення. Окреме значення має можливість адаптації системи до нових наборів даних за допомогою fine-tuning, оскільки в реальних умовах модель може стикатися з новими генераторами, іншими дикторами, різною якістю запису та зміненими умовами формування аудіо.

2 ОБҐРУНТУВАННЯ ОБРАНИХ ТЕХНОЛОГІЙ ТА МЕТОДІВ РОЗРОБКИ

2.1 Вибір програмних технологій

Для розроблення гібридної системи детекції синтезованого мовлення було обрано набір програмних технологій, які забезпечують зручну роботу з аудіоданими, побудову нейромережевих моделей, проведення експериментів та подальше використання системи через прикладний інтерфейс. Оскільки задача роботи поєднує цифрову обробку аудіосигналів, машинне навчання, використання попередньо навчених SSL-моделей і практичну реалізацію детектора, обрані технології мають підтримувати як дослідницьку, так і прикладну частину системи.

Основною мовою програмування було обрано Python. Такий вибір є доцільним, оскільки Python широко використовується в задачах машинного навчання, обробки сигналів і аналізу даних. Він має розвинену екосистему бібліотек для роботи з аудіо, числовими масивами, нейронними мережами та експериментальними pipeline. Крім того, Python дозволяє швидко змінювати конфігурацію моделі, додавати нові ознакові гілки, запускати навчання та оцінювання на різних наборах даних. Це особливо важливо для роботи, у якій необхідно порівнювати кілька конфігурацій гібридної моделі.

Для реалізації нейромережевої частини системи доцільним є використання PyTorch [15]. Цей фреймворк забезпечує гнучке створення моделей, роботу з tensor-операціями, підтримку GPU-прискорення та зручну організацію процесу навчання. У межах розробленої системи PyTorch використовується для реалізації ознакових гілок, проєкційних шарів, механізму злиття представлень, класифікаційного модуля, а також для навчання, fine-tuning і збереження checkpoint моделі. Гнучкість PyTorch є важливою, оскільки архітектура системи є модульною і передбачає

можливість змінювати набір активних компонентів залежно від експерименту.

Для роботи з аудіосигналами потрібні інструменти, які дозволяють завантажувати аудіофайли, приводити їх до єдиного формату, виконувати перетворення частоти дискретизації та формувати ознаки. У межах такої задачі доцільно використовувати бібліотеки для цифрової обробки аудіо, зокрема torchaudio та librosa [16]. Вони дають змогу працювати з хвильовою формою сигналу, виконувати попередню обробку, обчислювати спектральні характеристики та готувати аудіодані до подальшого передавання в модель. Завдяки цьому система може працювати з аудіозаписами різної тривалості та формату, приводячи їх до уніфікованого подання.

Окреме значення має підтримка попередньо навчених моделей самонавчання. Для отримання SSL-представлень у роботі використовується WavLM, яка формує високорівневі векторні представлення мовного сигналу. Використання таких моделей є доцільним, оскільки вони дають змогу враховувати не лише локальні акустичні властивості аудіо, а й ширший мовленнєвий контекст. Для інтеграції SSL-моделей у Python проєкт можуть використовуватися готові інструменти для завантаження попередньо навчених моделей і отримання embedding-представлень аудіосигналу. Це спрощує використання WavLM як окремої ознакової гілки в гібридній архітектурі.

Для роботи з таблицями метаданих і результатами експериментів доцільно використовувати бібліотеки для аналізу даних, зокрема pandas і NumPy. Вони дозволяють формувати manifest-таблиці, зберігати шляхи до аудіофайлів, мітки класів, назви split-частин, результати прогнозування та score моделі. Таке табличне представлення є зручним для запуску навчання й оцінювання на різних наборах даних, зокрема ASVspoof 2019 LA, ASVspoof 2021 LA та ASVspoof5. Крім того, воно полегшує аналіз помилок і порівняння різних конфігурацій системи.

Важливою вимогою до програмної реалізації була можливість відтворюваного проведення експериментів. Для цього система має підтримувати конфігурований запуск, у якому можна задавати активні гілки моделі, параметри навчання, шляхи до даних, режим оцінювання та checkpoint.

Такий підхід дозволяє не змінювати основний код для кожного нового експерименту, а керувати поведінкою системи через конфігураційні параметри. Це є важливим для порівняння моделей, оскільки результати мають отримуватися в однакових умовах обробки даних та оцінювання.

Окрім дослідницької частини, у роботі передбачено практичне використання системи через серверно-клієнтський застосунок. Для реалізації серверної частини доцільним є використання FastAPI. Цей фреймворк дозволяє створити API для завантаження аудіо- або відеофайлу, запуску попередньої обробки, передавання аудіосигналу в модель і повернення результату класифікації користувачу. Такий підхід відокремлює модель від інтерфейсу і робить систему придатною не лише для експериментів, а й для демонстрації роботи на окремих файлах.

Клієнтська частина системи потрібна для зручної взаємодії користувача з детектором. Вона дозволяє завантажити файл, отримати результат класифікації, числовий score та інтерпретацію відповіді у зрозумілій формі. Наявність такого інтерфейсу показує, що розроблена система може працювати не лише як набір скриптів для навчання та оцінювання, а і як прикладний інструмент для перевірки аудіоконтенту.

Цим, вибір програмних технологій обумовлений вимогами задачі детекції синтезованого мовлення. Python забезпечує гнучкість і зручність розробки, PyTorch дозволяє реалізувати та навчати нейромережеву модель, бібліотеки для обробки аудіо забезпечують підготовку сигналу й виділення ознак, а FastAPI та клієнтський інтерфейс дають змогу продемонструвати практичне використання системи.

Сукупність цих технологій створює основу для реалізації повного циклу роботи, від підготовки аудіоданих і навчання моделі до оцінювання результатів та перевірки окремих файлів.

2.2 Методи підготовки та попередньої обробки аудіоданих

Підготовка аудіоданих є одним із важливих етапів розроблення системи детекції синтезованого мовлення, оскільки якість роботи моделі залежить не лише від архітектури, а й від узгодженості вхідних даних. Аудіозаписи з різних наборів можуть відрізнятися форматом файлів, тривалістю, частотою дискретизації, способом подання міток, структурою каталогів і правилами розділення на навчальну та тестову частини. Якщо не привести ці дані до спільного формату, одна й та сама модель не зможе стабільно працювати з різними джерелами аудіо.

У розробленій системі підготовка даних ґрунтується на уніфікованому представленні аудіофайлів через manifest-таблиці. Такий підхід дозволяє відокремити логіку роботи моделі від особливостей конкретного набору даних. Замість того, щоб для кожного набору окремо змінювати код завантаження файлів, дані приводяться до спільної структури, яка містить шлях до аудіофайлу, мітку класу та назву split-частини. Це спрощує роботу з ASVspoof 2019 LA, ASVspoof 2021 LA, ASVspoof5 та іншими можливими джерелами аудіоданих [17].

Використання manifest-таблиць є доцільним ще й тому, що в задачі детекції синтезованого мовлення важливо чітко контролювати відповідність між аудіофайлом і його міткою. У межах роботи використовується бінарна постановка задачі, справжні записи належать до класу bonafide, а синтезовані або підроблені – до класу spoof. Для подальшого навчання ці класи подаються у числовому вигляді, що дозволяє використовувати стандартні методи бінарної класифікації. Така уніфікація зменшує ризик

помилок при переході між різними наборами даних і робить експерименти більш відтворюваними.

Окрему роль відіграє поділ даних на train, dev та eval-частини. Навчальна частина використовується для налаштування параметрів моделі, dev-частина – для проміжного контролю якості та вибору налаштувань, а eval-частина – для фінального оцінювання. Такий поділ потрібний для того, щоб результати не були завищені через перевірку моделі на тих самих даних, на яких вона навчалася. У задачах anti-spoofing це особливо важливо, оскільки модель може легко переорієнтуватися на особливості конкретних атак або конкретного набору даних.

Перед передаванням аудіосигналу в модель необхідно привести його до узгодженого технічного формату. Аудіофайли можуть мати різну частоту дискретизації, кількість каналів, амплітудний масштаб і тривалість. Для стабільної роботи ознакових гілок сигнал потрібно уніфікувати – привести до спільної частоти дискретизації, за потреби перетворити в mono-формат, нормалізувати амплітуду та підготувати до подальшого виділення ознак. Це дозволяє зменшити вплив технічних відмінностей між файлами й зосередити модель саме на властивостях мовлення.

Контроль тривалості аудіозаписів також є важливим елементом підготовки даних. У реальних і benchmark-наборах аудіофайли можуть бути як короткими, так і довшими, що ускладнює формування batch-ів під час навчання. Для нейромережевої моделі потрібно забезпечити узгоджене подання вхідних даних, тому застосовуються механізми обрізання, доповнення або групування аудіо за довжиною. Це дозволяє ефективніше використовувати пам’ять, зменшити кількість зайвих обчислень і зробити процес навчання стабільнішим.

Попередня обробка також необхідна через те, що різні ознакові гілки мають різні вимоги до вхідного сигналу. LFCC-гілка потребує сигналу, придатного для спектрального аналізу. SSL-гілка на основі WavLM працює з аудіо у форматі, сумісному з попередньо навченою моделлю. Фазові,

просодичні та RawNet2-inspired компоненти також використовують власні способи представлення сигналу. Тому попередня обробка має забезпечити такий формат аудіо, який можна коректно передати в усі активні гілки системи.

Важливою перевагою уніфікованого pipeline є можливість запускати однакову модель на різних наборах даних. Це має значення для перевірки узагальнювальної здатності системи. Якщо дані ASVspoof 2019 LA, ASVspoof 2021 LA та ASVspoof5 приводяться до спільного внутрішнього формату, тоді різниця в результатах більшою мірою відображає саме відмінності між доменами, а не випадкові відмінності в обробці файлів. Такий підхід робить експериментальне порівняння більш коректним.

Крім того, підготовка даних повинна підтримувати подальший аналіз результатів. Для цього до manifest-таблиць можуть додаватися службові поля: ідентифікатор запису, тип атаки, назва набору даних, split, тривалість аудіо або шлях до попередньо обчислених ознак. Такі дані не завжди використовуються безпосередньо під час класифікації, але вони корисні для аналізу помилок, порівняння поведінки моделі на різних групах записів і дослідження впливу окремих типів атак.

Отже, методи підготовки та попередньої обробки аудіоданих обрані з урахуванням потреб гібридної системи. Вони забезпечують уніфікацію вхідних файлів, коректне подання міток, підтримку різних наборів даних, стабільне формування batch-ів і сумісність із різними ознаковими гілками. Така організація даних є необхідною умовою для відтворюваного навчання, коректного оцінювання та подальшої адаптації системи до нових доменів.

2.3 Вибір ознакових представлень аудіосигналу

Для задачі детекції синтезованого мовлення важливо аналізувати аудіосигнал не лише за одним типом характеристик, а за сукупністю взаємодоповнюючих представлень [18]. Це пов’язано з тим, що ознаки

штучного походження мовлення можуть проявлятися на різних рівнях сигналу: у спектральній структурі, енергетично-часовій поведінці, фазовій узгодженості, високорівневих мовленнєвих закономірностях або безпосередньо у хвильовій формі [19]. Тому в розробленій системі використано гібридний підхід, який поєднує кілька груп ознак.

Одним із базових представлень у системі є LFCC, тобто лінійні частотні кепстральні коефіцієнти. Їх використання є доцільним, оскільки вони описують спектральну структуру мовного сигналу та дозволяють подати частотні властивості аудіо у компактній числовій формі. На відміну від ознак, побудованих на мел-шкалі, LFCC зберігають лінійне представлення частотної області. Це важливо для задачі anti-spoofing, оскільки артефакти синтезованого мовлення можуть проявлятися не лише в низько- та середньочастотній області, а й у високочастотних компонентах сигналу.

LFCC-гілка виконує роль спектральної основи системи. Вона дозволяє моделі враховувати розподіл енергії за частотами, локальні спектральні спотворення, нетипову гладкість або регулярність спектра. Такі властивості можуть бути корисними для виявлення синтезованого мовлення, оскільки генеративні моделі не завжди точно відтворюють усі фізичні особливості природного голосу [20]. Водночас LFCC не розглядаються як єдине достатнє джерело інформації, а використовуються як одна з гілок гібридної архітектури.

Другим важливим компонентом є SSL-представлення на основі WavLM. Self-supervised learning моделі навчаються на великих обсягах мовленнєвих даних і формують узагальнені векторні представлення аудіосигналу. На відміну від класичних акустичних ознак, такі представлення можуть містити інформацію не лише про локальні спектральні властивості, а й про ширший мовленнєвий контекст, структуру фрагмента, особливості вимови та приховані закономірності сигналу.

У фінальній конфігурації системи WavLM використовується як основна SSL-гілка. Такий вибір обґрунтований тим, що WavLM забезпечує інформативне високорівневе подання мовлення та добре доповнює класичні акустичні ознаки. У межах експериментального дослідження також розглядалося альтернативне SSL-представлення wav2vec2. Воно використовувалося для порівняння, однак основною для фінальної системи було обрано саме WavLM-орієнтовану конфігурацію, оскільки вона показала кращий баланс узагальнення під час перевірки на іншому домені.

Фазові характеристики використано як додаткове джерело інформації про часову узгодженість компонентів аудіосигналу. Спектральні ознаки переважно описують розподіл енергії за частотами, однак не повністю відображають фазову поведінку сигналу. У природному мовленні фазова структура формується фізичними процесами голосоутворення, тоді як під час синтезу вона може відтворюватися менш точно. Тому фазова гілка може допомагати виявляти локальні порушення, які не завжди помітні лише за амплітудним спектром або кепстральними коефіцієнтами.

Просодичні характеристики в системі використовуються для опису енергетично-часової поведінки мовлення. Вони дозволяють враховувати узагальнені властивості сигналу, пов’язані з енергією, часовою стабільністю та варіативністю аудіозапису. Для природного мовлення характерна певна нерівномірність і динаміка, тоді як синтезовані записи можуть демонструвати надмірну згладженість або стабільність окремих параметрів. У межах розробленої системи просодична гілка розглядається як компактне додаткове джерело інформації, яке доповнює спектральні, фазові та SSL ознаки.

Окрему роль у фінальній конфігурації відіграє RawNet2-inspired гілка. Її призначення полягає в тому, щоб аналізувати аудіосигнал ближче до хвильової форми, без повної залежності від заздалегідь сформованих акустичних ознак [21]. Такий компонент може виділяти закономірності, які не повністю представлені в LFCC або SSL-embedding [22]. Завдяки цьому

модель отримує ще один рівень аналізу сигналу, що є корисним для виявлення складніших або менш очевидних артефактів синтезу [23].

Поєднання LFCC, WavLM, фазових, просодичних та RawNet2-inspired ознак дозволяє системі аналізувати аудіозапис комплексно. LFCC відповідають за спектральну структуру, WavLM – за високорівневі мовленнєві представлення, фазові характеристики – за часову узгодженість, просодичні статистики – за енергетично-часову поведінку, а RawNet2 гілка – за інформацію, отриману безпосередньо з хвильової форми. Кожне з цих представлень описує мовлення з іншого боку, тому їх поєднання є більш інформативним, ніж використання лише однієї групи ознак.

Таким чином, вибір ознакових представлень обумовлений самою природою задачі детекції синтезованого мовлення. Оскільки штучне походження аудіосигналу може проявлятися у різних властивостях мовлення, система має враховувати кілька рівнів аналізу. Саме тому фінальна архітектура побудована як WavLM-орієнтована гібридна модель, яка поєднує класичні акустичні ознаки, SSL-представлення та raw waveform-компонент.

2.4 Архітектура моделі та механізм інтеграції ознак

Архітектура розробленої системи побудована як гібридна модель із кількома паралельними ознаковими гілками. Такий підхід обрано тому, що синтезоване мовлення не завжди можна надійно виявити за одним типом характеристик. Артефакти штучного походження можуть проявлятися в спектральній структурі, фазовій поведінці, енергетично-часовій динаміці, високорівневих мовленнєвих представленнях або безпосередньо у хвильовій формі аудіосигналу. Тому модель має отримувати інформацію з кількох джерел і формувати рішення на основі їх спільного аналізу.

Використання паралельних гілок дозволяє розділити процес виділення ознак за їхньою природою. LFCC-гілка відповідає за спектральне

представлення сигналу, SSL-гілка на основі WavLM формує високорівневі мовленнєві ознаки, фазова гілка враховує часову узгодженість компонентів сигналу, просодична гілка описує енергетично-часові властивості, а RawNet2-inspired компонент працює ближче до хвильової форми. Така організація є доцільною, оскільки кожна група ознак має власне призначення і може бути інформативною для різних типів аудіозаписів.

Важливою перевагою модульної архітектури є її конфігурованість. Набір активних гілок може змінюватися залежно від експерименту, що дозволяє порівнювати різні варіанти системи та оцінювати внесок окремих компонентів. Наприклад, можна перевірити якість моделі без SSL-гілки, порівняти WavLM і wav2vec2, оцінити вплив RawNet2-inspired компонента або дослідити роботу повної WavLM-орієнтованої конфігурації. Такий підхід робить систему придатною не лише для фінального використання, а й для експериментального аналізу.

Оскільки різні гілки формують представлення різної природи та розмірності, їх не можна безпосередньо об’єднати без попереднього узгодження. Наприклад, LFCC-ознаки описують спектральну структуру, WavLM формує високорівневі embedding-представлення, а просодичні характеристики можуть бути подані у вигляді компактного статистичного вектора. Якщо просто об’єднати такі ознаки без обробки, модель може надмірно залежати від представлень із більшою розмірністю або втрачати частину корисної інформації з менших гілок.

Для розв’язання цієї проблеми в архітектурі використовуються проєкційні шари. Їхнє призначення полягає в тому, щоб перетворити вихід кожної ознакової гілки до спільного простору фіксованої розмірності. Завдяки цьому різнорідні представлення стають сумісними для подальшого злиття. Проєкційний модуль також виконує роль додаткового навчального компонента, який дозволяє моделі навчитися виділяти з кожної групи ознак саме ту інформацію, яка є корисною для класифікації.

Після приведення ознак до спільного простору використовується механізм gated attention fusion. Його застосування обґрунтоване тим, що інформативність окремих гілок може відрізнятися для різних аудіозаписів. В одному випадку більш корисними можуть бути спектральні ознаки, в іншому – SSL-представлення або інформація з хвильової форми. Тому модель повинна не лише об’єднувати всі доступні вектори, а й мати можливість регулювати їхній внесок у фінальне представлення.

Механізм gated attention fusion поєднує дві важливі ідеї – керування внеском ознак і механізм уваги. Керувальна частина дозволяє моделі посилювати або послаблювати окремі представлення, а attention-компонент допомагає зосередитися на більш інформативних джерелах інформації. Це є важливим для гібридної системи, оскільки різні ознаки не завжди мають однакову значущість. Наприклад, для одного типу синтезу сильніші відмінності можуть бути помітні у спектральній області, а для іншого – у SSL-представленнях або raw waveform-гілці.

Порівняно з простим об’єднанням векторів, gated attention fusion є більш гнучким способом інтеграції ознак. Проста конкатенація лише передає всі ознаки далі до класифікатора, але не враховує їхню відносну інформативність. Натомість механізм керованого злиття дозволяє моделі сформувати більш збалансоване фінальне представлення аудіозапису. Це особливо важливо в умовах domain shift, коли корисність окремих груп ознак може змінюватися при переході від одного набору даних до іншого.

Після інтеграції ознак сформоване спільне представлення передається до класифікаційного модуля. У системі для цього використовується багатошаровий перцептрон, який виконує бінарну класифікацію аудіозаписів на bonafide та spoof. Такий класифікатор є доцільним, оскільки на цьому етапі модель уже отримує узгоджений вектор ознак, а завдання полягає у визначенні межі між двома класами. Вихід класифікатора подається як score, який надалі може інтерпретуватися через поріг класифікації.

Використання одного вихідного score є зручним для задачі anti-spoofing, оскільки дозволяє не лише отримати фінальну мітку класу, а й аналізувати поведінку моделі при різних threshold. Це важливо для розрахунку AUC ROC, EER, precision, recall і F1-score. Такий підхід дає змогу оцінювати модель як за загальною здатністю розділяти класи, так і за її поведінкою при конкретному порозі прийняття рішення.

Обрана архітектура також підтримує подальший fine-tuning. Оскільки система побудована модульно, її можна адаптувати до нового домену без повної зміни структури. При переході на інший набір даних зберігається загальна логіка роботи моделі, тобто виділення ознак у паралельних гілках, проєкція до спільного простору, gated attention fusion і класифікація. Змінюються лише параметри trainable-компонентів, що дозволяє уточнити модель відповідно до нового розподілу аудіоданих.

Таким чином, архітектура моделі обґрунтована потребою комплексного аналізу мовного сигналу. Паралельні гілки дозволяють отримувати різні типи ознак, проєкційні шари узгоджують їх у спільному просторі, gated attention fusion забезпечує гнучке злиття представлень, а MLP-класифікатор формує фінальне рішення. Така структура відповідає задачі детекції синтезованого мовлення, оскільки дозволяє враховувати різнорівневі ознаки аудіосигналу та адаптувати систему до різних експериментальних сценаріїв.

2.5 Методи навчання, оцінювання та практичної реалізації

Для розробленої системи детекції синтезованого мовлення було обрано підхід навчання з учителем, оскільки задача формулюється як бінарна класифікація аудіозаписів на два класи: bonafide та spoof. Кожному аудіофайлу відповідає мітка, яка визначає його походження, справжнє мовлення або синтезований чи підроблений запис. Така постановка є природною для anti-spoofing-задач, оскільки система має навчитися

відокремлювати автентичне мовлення від штучно сформованого на основі розмічених прикладів.

Процес навчання організовано у вигляді конфігурованого pipeline. Це означає, що основні параметри експерименту задаються через конфігурацію: шлях до даних, активні ознакові гілки, параметри моделі, режим навчання, кількість епох, batch size, learning rate, шлях для збереження checkpoint та режим оцінювання. Такий підхід є доцільним, оскільки в роботі досліджуються різні конфігурації моделі. Замість ручної зміни коду для кожного експерименту можна змінювати лише відповідні параметри запуску, що підвищує відтворюваність і зручність порівняння результатів.

Під час навчання оптимізуються параметри trainable-компонентів системи: проєкційних шарів, механізму gated attention fusion, класифікаційного модуля та додаткових ознакових гілок. WavLM у фінальній конфігурації використовується як попередньо навчений SSL екстрактор високорівневих мовленнєвих представлень. Такий підхід дозволяє використати знання, отримані моделлю під час попереднього навчання на великих обсягах мовлення, і водночас зменшити обчислювальні витрати та ризик перенавчання на обмеженому наборі anti-spoofing-даних.

Важливим елементом навчального процесу є збереження checkpoint моделі. Checkpoint фіксує стан моделі після певного етапу навчання або після досягнення кращого значення контрольної метрики. Це дає змогу відновити модель для подальшого оцінювання, порівняння з іншими конфігураціями або fine-tuning на новому наборі даних. Для експериментальної роботи це особливо важливо, оскільки одна й та сама архітектура може перевірятися на різних доменах, а результати мають бути відтворюваними.

Окремим режимом роботи системи є fine-tuning. У межах цієї роботи fine-tuning розглядається як метод доменної адаптації, тобто додаткового навчання вже підготовленої моделі на даних нового набору. Такий підхід є

доцільним, оскільки між різними наборами даних може існувати domain shift: змінюються типи атак, властивості мовців, умови запису, якість аудіо, формат даних і співвідношення класів. Fine-tuning дозволяє уточнити параметри моделі відповідно до нового розподілу аудіосигналів без повного навчання системи з нуля [24], [25].

Для оцінювання якості системи використовується набір метрик, які дозволяють проаналізувати модель з різних боків. Однією з основних метрик є AUC ROC, яка характеризує здатність моделі ранжувати аудіозаписи за ймовірністю належності до spoof-класу. Ця метрика не залежить від одного фіксованого порога класифікації, тому є корисною для загальної оцінки роздільної здатності score моделі.

Важливою метрикою для anti-spoofing-задач є EER, або Equal Error Rate. Вона показує рівень помилки в точці, де частка хибного прийняття та частка хибного відхилення збалансовані. На відміну від accuracy, EER краще відображає поведінку моделі в задачах безпеки, де важливо враховувати обидва типи помилок. Саме тому EER використовується як одна з ключових метрик при порівнянні конфігурацій моделі та оцінюванні її здатності до узагальнення.

Крім AUC ROC та EER, у роботі використовуються precision, recall і F1-score. Precision показує, наскільки надійними є позитивні передбачення моделі, тобто скільки записів, визначених як spoof, справді належать до цього класу. Recall характеризує здатність моделі знаходити spoof-записи серед усіх наявних підроблених записів. F1-score поєднує precision і recall в одну метрику, що дозволяє оцінити баланс між цими двома характеристиками.

Accuracy у цій задачі використовується як допоміжна метрика. Вона може бути зручною для загального уявлення про частку правильних відповідей, але її не слід розглядати ізольовано. У наборах для детекції синтезованого мовлення часто існує помітний дисбаланс між bonafide та spoof-записами. Через це високе значення accuracy не завжди означає, що

модель однаково добре працює з обома класами. Саме тому основна увага приділяється метрикам AUC ROC, EER, precision, recall і F1-score.

Для практичного використання системи було обґрунтовано створення серверно-клієнтської частини. Дослідницькі скрипти зручні для навчання та оцінювання моделі, однак вони не є достатньо зручними для перевірки окремих файлів користувачем. Тому серверна частина на основі FastAPI використовується як прикладний інтерфейс до моделі. Вона приймає аудіо або відеофайл, виконує необхідну попередню обробку, передає аудіосигнал у модель і повертає результат класифікації.

Клієнтський інтерфейс виконує роль зручної оболонки для взаємодії з системою. Користувач може завантажити файл і отримати відповідь у зрозумілому форматі: клас аудіозапису, score та інтерпретацію результату. Така реалізація демонструє, що розроблена система може використовуватися не лише в межах експериментального середовища, а й як прикладний інструмент для перевірки аудіоконтенту. Водночас серверно-клієнтська частина не змінює саму модель, а лише забезпечує доступ до її функціональності.

Отже, методи навчання, оцінювання та практичної реалізації обрані відповідно до задачі детекції синтезованого мовлення. Навчання з учителем дозволяє сформувати модель для бінарної класифікації bonafide та spoof записів, fine-tuning забезпечує адаптацію до нового домену, а набір метрик дає змогу комплексно оцінити якість системи. Практична реалізація через FastAPI та клієнтський інтерфейс демонструє можливість використання розробленої моделі для перевірки окремих аудіо- та відеофайлів.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ГІБРИДНОЇ СИСТЕМИ ДЕТЕКЦІЇ СИНТЕЗОВАНОГО МОВЛЕННЯ

3.1 Архітектура системи

Розроблена система призначена для автоматичного визначення походження мовного аудіосигналу та розв’язує задачу бінарної класифікації аудіозаписів на два класи: справжнє мовлення та синтезоване мовлення. У межах реалізації ці класи подаються як bonafide та spoof відповідно. На вхід системи подається аудіофайл, після чого виконується його попередня обробка, формування кількох груп ознак, інтеграція отриманих представлень і прийняття фінального рішення класифікаційним модулем.

Загальна ідея системи полягає в тому, що аудіосигнал аналізується не лише з одного боку, а через кілька взаємодоповнюючих представлень. Такий підхід обрано через те, що ознаки синтезованого мовлення можуть проявлятися на різних рівнях сигналу. Частина відмінностей може бути пов’язана зі спектральною структурою, частина – з часовою або фазовою поведінкою, а частина – з високорівневими закономірностями, які виділяються попередньо навченими моделями мовлення. Тому архітектура системи побудована як гібридна, вона дає змогу поєднувати класичні акустичні ознаки та нейромережеві представлення в одному класифікаційному процесі.

Узагальнено роботу системи можна подати як послідовність таких етапів: завантаження аудіофайлу, попередня обробка аудіосигналу, виділення ознак у паралельних гілках, приведення ознак до спільного простору, злиття представлень та класифікація (рисунок 3.1). На першому етапі аудіофайл зчитується із файлової системи або надходить через користувацький інтерфейс. Далі сигнал приводиться до єдиного формату, придатного для подальшої обробки. Після цього він передається в набір

ознакових гілок, кожна з яких формує окреме числове представлення аудіозапису.

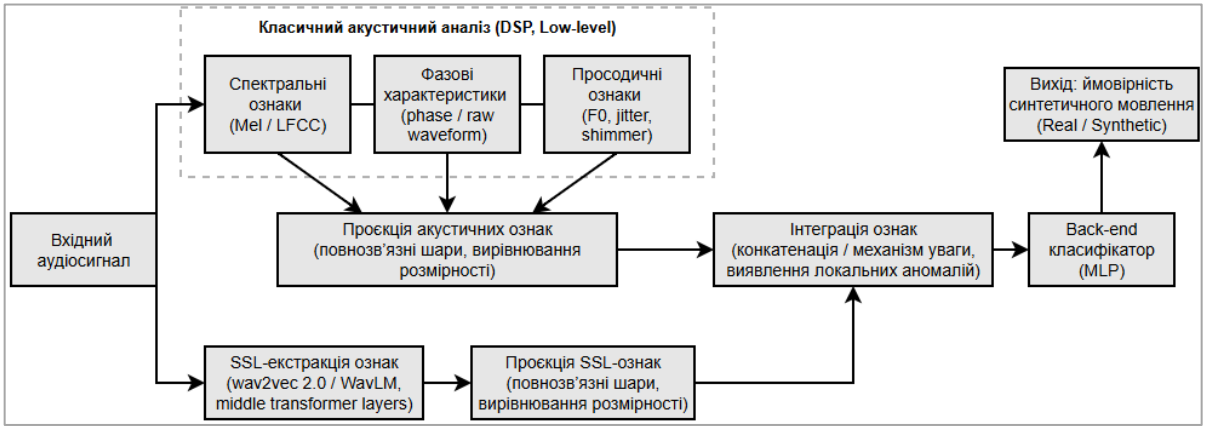


Рисунок 3.1 – Загальна структура гібридної системи детекції синтезованого мовлення

Основним вхідним об'єктом системи є аудіосигнал, представлений у вигляді хвильової форми. Перед передаванням у модель він проходить попередню обробку, яка забезпечує уніфікацію вхідних даних. Це необхідно тому, що аудіозаписи можуть мати різну тривалість, формат, частоту дискретизації та амплітудний масштаб. Уніфіковане подання сигналу дозволяє використовувати однакову модель для різних аудіофайлів і зменшує вплив технічних відмінностей між записами.

Після попередньої обробки аудіосигнал подається до паралельних гілок виділення ознак. У базовій конфігурації система використовує спектральну гілку на основі просодичних ознак та SSL-гілку на основі WavLM. Спектральна гілка призначена для отримання низькорівневих акустичних характеристик, пов'язаних із частотною структурою мовлення. SSL-гілка, навпаки, формує високорівневе представлення аудіосигналу за допомогою попередньо навченої моделі мовлення. Таке поєднання дозволяє враховувати як локальні властивості сигналу, так і більш узагальнені закономірності мовлення.

Архітектура реалізована конфігурованою, тому набір активних гілок може змінюватися залежно від експерименту. Окрім базового варіанта, який поєднує: LFCC, WavLM, phase, prosody, у проєкті передбачено можливість використання інших ознакових компонентів, зокрема wav2vec2 представлень та RawNet2-подібної гілки, яка працює безпосередньо з хвильовою формою сигналу. Завдяки цьому одна й та сама програмна структура може застосовуватися як для основної моделі, так і для порівняльних експериментів та досліджень.

Кожна гілка ознак формує власний вектор характеристик. Оскільки ці вектори мають різну природу та різну розмірність, їх не можна безпосередньо об’єднати без попереднього узгодження. Для цього в системі використовується проєкційний модуль, який перетворює вихід кожної гілки у спільний простір фіксованої розмірності. Такий крок дозволяє надалі порівнювати та об’єднувати різні типи ознак у межах єдиного механізму злиття.

Після проєкції ознаки передаються до модуля інтеграції. У розробленій системі використовується «gated attention fusion», тобто злиття ознак на основі механізмів керування та уваги. Його призначення полягає в тому, щоб надати моделі можливість регулювати внесок кожного джерела інформації під час формування фінального представлення аудіозапису. Це важливо, оскільки інформативність окремих ознак може відрізнятися для різних аудіосигналів. Наприклад, в одному випадку більш корисними можуть бути спектральні характеристики, а в іншому – SSL-представлення або ознаки, пов’язані з часовою структурою сигналу.

Отримане після злиття представлення передається до класифікаційного модуля. Класифікатор реалізовано як багатошаровий перцептрон, який формує один вихідний «logit», необроблене числове значення на виході моделі перед застосуванням функції активації, для задачі бінарної класифікації. Після застосування сигмоїдної функції цей «logit» інтерпретується як оцінка класу «spoof», тобто як оцінка ймовірності того,

що аудіозапис має синтетичне походження. Якщо отримане значення перевищує заданий поріг, запис класифікується як «spoof»; інакше він належить до класу «bonafide». При цьому під час експериментального оцінювання поріг може підбиратися окремо, зокрема для визначення точки рівності помилок.

Окремою особливістю системи є те, що вона не обмежується лише навчанням моделі. У межах проєкту також реалізовано засоби для оцінювання якості, збереження результатів, побудови ROC-кривої (Receiver Operating Characteristic curve), формування матриці помилок та отримання прогнозу для окремого аудіофайлу. Це дозволяє використовувати систему не тільки для тренування моделі, а й для повного циклу експериментального дослідження, від підготовки даних до аналізу отриманих результатів.

Крім дослідницької частини, у проєкті передбачено демонстраційний програмний інтерфейс. Для цього розроблено серверну частину, яка приймає аудіо або відео файл, виконує необхідну обробку та передає аудіосигнал до моделі. Клієнтська частина надає користувачу можливість завантажити файл і отримати результат класифікації у вигляді мітки, оцінки та впевненості. Такий інтерфейс не є окремою моделлю, а виконує роль практичної оболонки над розробленим детектором, демонструючи можливість його застосування поза межами експериментального середовища.

3.2 Підготовка та попередня обробка аудіоданих

Якість роботи системи детекції синтезованого мовлення значною мірою залежить не лише від архітектури моделі, а й від того, наскільки узгоджено підготовлено вхідні аудіодані. Аудіозаписи в наборах даних можуть мати різну тривалість, різні ідентифікатори, різний спосіб подання міток, а також відрізнятися технічними характеристиками файлів. Через це

перед навчанням і оцінюванням моделі необхідно привести дані до єдиного формату, який може бути використаний усіма компонентами системи.

У розробленій системі підготовка даних виконується через уніфіковане подання у вигляді таблиці з метаданими аудіофайлів. Такий підхід дозволяє відокремити логіку роботи моделі від особливостей конкретного набору даних. Замість того, щоб у кожному модулі окремо обробляти структуру ASVspoof 2019, ASVspoof 2021, ASVspoof5 або іншого набору, система перетворює вхідні файли протоколів розмітки або готові CSV/JSON-файли до спільного формату. Це спрощує запуск експериментів і дозволяє використовувати однаковий конвеєр обробки для різних джерел аудіоданих.

Основними полями таблиці з метаданими аудіофайлів є шлях до аудіофайлу, числова мітка класу та назва частини. Поле «path» містить шлях до відповідного аудіозапису, поле «label» визначає клас аудіосигналу, а поле «split» вказує, до якої частини набору належить запис: «train», «dev», «eval» або іншої службової частини. У межах реалізації використовується бінарне кодування класів, де значення 0 відповідає справжньому мовленню, тобто «bonafide», а значення 1 – синтезованому або підробленому мовленню, тобто «spoof». Такий формат є зручним для подальшого використання в задачі бінарної класифікації.

Окрім обов’язкових полів, таблиця з метаданими може містити додаткову інформацію про аудіозапис. До таких полів належать ідентифікатор запису, ідентифікатор атаки, ідентифікатор мовця, назва набору даних, шлях до попередньо обчисленого SSL представлення та тривалість запису. Наявність цих полів не є обов’язковою для роботи моделі, однак вони корисні для аналізу результатів, побудови порівняльних експериментів, групування помилок за типами атак і повторного використання вже підготовлених ознак.

Для роботи з наборами ASVspoof у системі передбачено окремий модуль розбору файлів протоколів розмітки. Його завдання полягає в тому,

щоб прочитати службові файли з описом записів, виділити ідентифікатор аудіо, мітку класу, інформацію про атаку та сформувати повний шлях до аудіофайлу. Такий модуль враховує відмінності між форматами файлів протоколів і приводить їх до єдиного внутрішнього представлення. Це важливо, оскільки різні версії ASVspoof можуть мати різну структуру службових записів, але для моделі ці відмінності не повинні впливати на спосіб завантаження даних.

Після формування таблиць метаданих для кожного аудіозапису визначається його довжина у дискретних відліках. Якщо в таблиці вже є тривалість аудіо в секундах, вона перетворюється у кількість відліків з урахуванням цільової частоти дискретизації. Якщо тривалість не задана, система намагається отримати її з метаданих аудіофайлу. Такий крок потрібен не тільки для статистичного опису даних, а й для організації ефективного групування під час навчання, оскільки аудіозаписи можуть суттєво відрізнятися за довжиною.

Завантаження аудіофайлів реалізовано з використанням кількох послідовних способів. Спочатку система намагається прочитати аудіо за допомогою бібліотеки «soundfile». Якщо це не вдається, використовується «torchaudio». Якщо й цей спосіб не спрацьовує, застосовується «ffmpeg» як резервний варіант декодування. Така послідовність підвищує стійкість системи до різних форматів аудіо та контейнерів. Це особливо важливо для практичного використання, оскільки вхідні файли можуть бути не лише у форматі FLAC або WAV, а й у стиснених або контейнерних форматах.

Після завантаження аудіосигнал приводиться до єдиного формату. Якщо запис має кілька каналів, він перетворюється в моно шляхом усереднення каналів. Це дозволяє уникнути залежності моделі від кількості каналів у файлі та забезпечує однакову форму вхідного сигналу для всіх записів. Далі виконується передискретизація до цільової частоти дискретизації. У реалізованій системі як базове значення використовується

частота 16000 Гц, що є типовим варіантом для задач обробки мовлення та сумісним із більшістю використаних аудіомоделей.

Наступним етапом є нормалізація амплітуди. У системі використовується пікова нормалізація, за якої сигнал масштабується відносно максимального абсолютного значення амплітуди. Це дозволяє зменшити вплив різної гучності записів і подати аудіосигнали в більш узгодженому масштабі. Водночас така нормалізація не змінює часову структуру сигналу і не вносить додаткових перетворень, пов’язаних зі зміною спектрального складу аудіо.

У конфігурації також передбачена можливість обрізання тиші. Цей параметр є опційним, оскільки видалення тиші може мати як позитивний, так і негативний вплив залежно від конкретного експерименту. З одного боку, воно дозволяє прибрати неінформативні ділянки на початку або в кінці аудіозапису. З іншого боку, паузи та неактивні ділянки можуть бути частиною реального розподілу даних і впливати на роботу моделі в практичних умовах. Тому обрізання тиші не є обов’язковим етапом, а керується через конфігураційні параметри.

Оскільки аудіофайли мають різну тривалість, під час формування груп необхідно привести їх до однакової довжини. Для цього в межах однієї групи усі хвильові форми доповнюються нулями до довжини найдовшого запису в цій групі. Одночасно формується маска доповнення, яка позначає доповнені ділянки. Така маска потрібна для того, щоб модель і пов’язані з нею компоненти могли відрізняти реальні семпли аудіосигналу від штучно доданих нульових значень.

Для зменшення обчислювальних витрат використовується групування з урахуванням довжини. Його ідея полягає в тому, щоб групувати аудіозаписи близької довжини в межах однієї групи. Якщо в одну групу потрапляють записи з дуже різною тривалістю, короткі аудіо доводиться значно доповнювати нулями, що збільшує обсяг зайвих обчислень. Групування за довжиною дозволяє зменшити кількість доповнених

елементів, ефективніше використовувати пам’ять GPU та стабілізувати процес навчання.

У результаті роботи блоку підготовки даних кожен елемент вибірки подається у вигляді структури, що містить хвильову форму, частоту дискретизації, довжину сигналу, числову мітку класу, ідентифікатор запису, шлях до файлу та додаткові метадані за наявності. Після об’єднання кількох елементів у групу система формує тензор аудіосигналів, маску доповнення, вектор довжин, вектор міток і списки службових ідентифікаторів. Саме в такому вигляді дані передаються до моделі під час навчання, оцінювання та інференсу.

Важливо, що блок підготовки даних не прив’язаний лише до одного конкретного набору. Якщо дані вже подані у форматі метаданих, система може використовувати їх напряму. Але якщо ж використовується ASVspoof-подібна структура з файлами протоколів, вони попередньо перетворюються до того самого представлення з метаданими. Завдяки цьому реалізація залишається гнучкою, до неї можна додавати нові набори даних, не змінюючи архітектуру моделі та основну логіку навчання.

3.3 Формування спектральних ознак LFCC

Однією з основних гілок розробленої системи є спектральна гілка на основі LFCC. Її призначення полягає в отриманні компактного числового опису частотної структури мовного сигналу. На відміну від безпосередньої роботи з хвильовою формою, спектральні ознаки дозволяють подати аудіосигнал у вигляді послідовності коефіцієнтів, що відображають розподіл енергії в частотній області. Це є важливим для задачі детекції синтезованого мовлення, оскільки артефакти генерації можуть проявлятися саме у спектральній структурі сигналу.

У задачах виявлення підробленого або синтезованого мовлення кепстральні ознаки часто використовуються як базове акустичне

представлення. Їхня перевага полягає в тому, що вони формують відносно невеликий вектор характеристик, але при цьому зберігають інформацію про частотну організацію мовлення. У контексті ASVspoof-подібних задач LFCC належать до поширених ознак для побудови базових вимірювальних систем, оскільки вони дозволяють аналізувати спектральні відмінності між природним і штучно сформованим аудіосигналом [8].

У розробленій системі LFCC-гілка реалізована як окремий модуль виділення ознак (рисунок 3.2). На вхід вона отримує групу аудіосигналів у вигляді хвильових форм і відповідну маску доповнення, яка позначає реальні та доповнені ділянки сигналу. Наявність маски є важливою, оскільки записи в групі можуть мати різну початкову довжину, а після доповнення нулями не всі семпли повинні брати участь у формуванні фінального представлення. Завдяки цьому статистичне узагальнення ознак виконується тільки за валідними ділянками аудіо.

```python
class LFCCBranch(nn.Module):
    """LFCC feature extraction branch with stats pooling."""

    def __init__(self, sample_rate: int, n_lfcc: int, n_fft: int, win_length: int, hop_length: int) -> None:
        super().__init__()
        self.hop_length = hop_length
        self.transform = torchaudio.transforms.LFCC(
            sample_rate=sample_rate,
            n_lfcc=n_lfcc,
            speckwargs={
                "n_fft": n_fft,
                "win_length": win_length,
                "hop_length": hop_length,
                "center": True,
            },
        )
        self.output_dim = n_lfcc * 2
```

Рисунок 3.2 – Приклад класу LFCC-гілки

Для обчислення LFCC використовується перетворення, реалізоване засобами бібліотеки «torchaudio». Основні параметри LFCC-гілки наведено в таблиці 3.1.

Таблиця 3.1 – Параметри формування LFCC-ознак

Параметр	Значення	Призначення
sample_rate	16000	цільова частота дискретизації аудіосигналу
n_lfcc	60	кількість LFCC-коефіцієнтів
n_fft	512	розмір FFT-перетворення
win_length	400	довжина аналізованого вікна в семплах
hop_length	160	крок між сусідніми вікнами в семплах
center	true	центрування фреймів під час спектрального аналізу

За частоти дискретизації 16000 Гц довжина вікна 400 семплів відповідає приблизно 25 мс аудіосигналу, а крок 160 семплів – приблизно 10 мс. Такий вибір параметрів є виправданим, оскільки мовний сигнал на коротких часових інтервалах можна розглядати як відносно стабільний. Це дозволяє аналізувати локальні спектральні властивості мовлення без втрати часової структури запису.

Після застосування LFCC-перетворення для кожного аудіозапису формується послідовність фреймових ознак. Тобто замість одного вектора на весь аудіофайл спочатку отримується матриця, у якій кожен часовий фрейм описується набором LFCC-коефіцієнтів. Така форма є зручною для збереження локальної інформації про спектральну структуру сигналу, однак класифікаційний модуль системи працює з узагальненим представленням запису. Тому фреймові ознаки потрібно перетворити на вектор фіксованої розмірності.

Для цього в LFCC-гілці використовується статистичне узагальнення. За всіма валідними фреймами обчислюються два типи статистик – середнє значення та стандартне відхилення. Середнє значення описує усереднений спектральний профіль аудіозапису, тоді як стандартне відхилення відображає варіативність LFCC-коефіцієнтів у часі. Поєднання цих двох статистик дозволяє зберегти не тільки загальну частотну характеристику сигналу, а й інформацію про її зміну протягом запису.

Оскільки в реалізації використовується 60 LFCC-коефіцієнтів, після обчислення середнього значення формується вектор розмірності 60. Аналогічно стандартне відхилення також має розмірність 60. Після конкатенації цих двох частин вихідний вектор LFCC-гілки має розмірність 120. Саме цей вектор надалі передається до проєкційного модуля, де приводиться до спільного простору представлень разом з іншими гілками моделі.

Важливою деталлю є те, що узагальнення виконується з урахуванням фактичної довжини аудіозапису. Для цього на основі маски доповнення визначається кількість реальних семплів, після чого обчислюється відповідна кількість валідних фреймів. Фрейми, які відповідають штучно доповненим нульовим ділянкам, не повинні впливати на середнє значення та стандартне відхилення. Такий підхід дозволяє коректно обробляти записи різної тривалості та зменшує ризик спотворення ознак через доповнення.

У межах гібридної архітектури LFCC-гілка виконує роль джерела низькорівневих спектральних характеристик. Вона не замінює SSL представлення та інші ознаки, а доповнює їх. Якщо SSL-гілка формує більш абстрактне представлення мовлення, отримане з попередньо навченої моделі, то LFCC-гілка безпосередньо описує частотну структуру конкретного аудіосигналу. Завдяки цьому модель може використовувати як високорівневі нейромережеві ознаки, так і більш прозорі акустичні характеристики.

У практичному сенсі використання LFCC також є корисним через відносно невелику обчислювальну складність. На відміну від великих SSL моделей, LFCC-ознаки не потребують проходження аудіосигналу через глибоку нейромережу. Це робить їх зручними для базового аналізу і порівняльних експериментів з більш складними представленнями. Крім того, LFCC-гілка може працювати як самостійний baseline або як один із компонентів гібридної моделі.

3.4 Отримання SSL-представлень мовлення

Другою ключовою групою ознак у розробленій системі є SSL представлення мовлення. На відміну від LFCC-ознак, які описують аудіосигнал через спектральні коефіцієнти, SSL-представлення формуються за допомогою попередньо навчених нейромережевих моделей мовлення. Такі моделі навчаються на великих обсягах аудіоданих і здатні виділяти узагальнені характеристики мовного сигналу, які можуть бути корисними не лише для розпізнавання мовлення, а й для суміжних задач аналізу аудіо.

У межах задачі детекції синтезованого мовлення SSL-підхід є корисним тому, що він дозволяє отримати не тільки низькорівневі акустичні параметри, а й більш абстрактні представлення мовного сигналу. Синтезоване мовлення може бути дуже схожим на природне за окремими спектральними характеристиками, однак відмінності можуть проявлятися у складніших закономірностях, наприклад у часовій структурі, стабільності артикуляційних патернів, характері переходів між фрагментами мовлення або у внутрішніх залежностях між різними ділянками сигналу. Попередньо навчені SSL-моделі можуть частково відображати такі закономірності у своїх прихованих станах.

У розробленій системі основною SSL-моделлю є WavLM-base. WavLM належить до моделей самокерованого навчання для обробки мовлення та була запропонована як універсальна модель для широкого спектра мовленнєвих задач [3]. Її використання в цій роботі пов’язане з тим, що модель формує багаторівневе представлення аудіосигналу, де різні шари можуть містити інформацію різного рівня узагальнення. Нижчі шари зазвичай ближчі до акустичної структури сигналу, тоді як середні й вищі шари можуть відображати складніші мовленнєві закономірності.

Додатково в системі передбачена можливість використання wav2vec2 як альтернативної або додаткової SSL-гілки. Модель wav2vec 2.0 також належить до підходів самокерованого навчання мовленнєвих представлень

і була запропонована як фреймворк для отримання корисних ознак із сирого аудіосигналу без потреби у повністю розмічених даних на етапі попереднього навчання [11]. У межах реалізованого прототипу гілка wav2vec2 використовується як експериментальний компонент, що дозволяє порівняти або поєднати різні типи SSL-представлень.

Практична реалізація SSL-гілки побудована на використанні моделей із бібліотеки «Transformers» (рисунок 3.3). Аудіосигнал після попередньої обробки передається до відповідної SSL-моделі, яка повертає послідовність прихованих станів.

```python
class SSLBranch(nn.Module):
    """Frozen or fine-tuned SSL branch using HuggingFace audio models."""

    def __init__(self, model_name: str, selected_layers: list[int], frozen: bool = True) -> None:
        super().__init__()
        self.model = AutoModel.from_pretrained(model_name, output_hidden_states=True)
        self.selected_layers = selected_layers
        self.frozen = frozen
        self.output_dim = int(self.model.config.hidden_size)
        conv_stride = getattr(self.model.config, "conv_stride", [5, 2, 2, 2, 2, 2, 2])
        self.total_stride = int(np.prod(conv_stride))

        if frozen:
            for parameter in self.model.parameters():
                parameter.requires_grad = False
            self.model.eval()

    def forward(self, waveform: torch.Tensor, padding_mask: torch.Tensor) -> torch.Tensor:
        attention_mask = (~padding_mask).long()
        context = torch.no_grad() if self.frozen else nullcontext()
        with context:
            outputs = self.model(
                input_values=waveform,
                attention_mask=attention_mask,
                output_hidden_states=True,
                return_dict=True,
            )
        hidden_states = outputs.hidden_states
        selected = [hidden_states[index] for index in self.selected_layers]
        features = torch.stack(selected, dim=0).mean(dim=0)
        valid_mask = frame_valid_mask(padding_mask, self.total_stride, features.size(1))
        return masked_mean(features, valid_mask)
```

Рисунок 3.3 – Приклад класу SSL-гілки

Кожен прихований стан можна розглядати як часову послідовність векторів представлень, де кожен вектор описує певний фрагмент аудіосигналу у внутрішньому просторі моделі. Оскільки на виході SSL

моделі формується не один вектор, а послідовність представлень, необхідно додатково визначити, які шари використовувати та як узагальнювати інформацію в межах усього аудіозапису.

У розробленій системі використовуються приховані стани шарів 5, 6 і 7. Такий вибір орієнтований на використання середніх шарів моделі, які можуть містити баланс між акустичною інформацією та більш узагальненими мовленнєвими ознаками. Використання не одного, а кількох шарів дозволяє отримати представлення, яке враховує інформацію з різних рівнів SSL-моделі. Для кожного вибраного шару формується окреме часово-залежне представлення, після чого ці представлення об’єднуються для подальшого узагальнення.

На відміну від LFCC-гілки, де ознаки обчислюються через явне спектральне перетворення, SSL-гілка отримує ознаки шляхом проходження аудіосигналу через глибоку нейромережеву модель. Це робить її більш обчислювально складною, але водночас дає можливість використати знання, набуті моделлю під час попереднього навчання на великих наборах мовлення. У контексті виявлення штучного аудіо подібні SSL представлення є перспективними, оскільки вони можуть виявляти не лише прості спектральні артефакти, а й складніші відмінності у структурі мовлення. Зокрема, у роботах з виявлення штучного аудіо досліджується використання WavLM разом із механізмами багаторівневого або уваги злиття ознак [12].

У базовому режимі роботи SSL-модель у системі використовується як заморожений модуль виділення ознак. Це означає, що параметри WavLM або wav2vec2 не оновлюються під час навчання основного класифікатора. Такий підхід має кілька практичних переваг. По-перше, він зменшує потребу в обчислювальних ресурсах, оскільки повне донавчання SSL моделі є досить вартісним. По-друге, заморожування знижує ризик перенавчання на обмеженій навчальній вибірці. По-третє, це дозволяє використовувати

SSL-модель як стабільне джерело ознак, а навчати – лише проєкційні шари, шари злиття ознак та класифікаційні шари.

Разом із цим у реалізації передбачена можливість часткового донавчання SSL-моделі. Для цього конфігурація може задавати кількість останніх шарів, які розморожуються під час донавчання. Такий режим доцільно використовувати тоді, коли базова заморожена модель уже показала стабільний результат, але потрібно адаптувати її до нового домену даних. Наприклад, під час переходу від ASVspoof 2019 до ASVspoof5 модель може стикатися з іншими спікерами, умовами запису або типами атак. У такому разі часткове донавчання останніх шарів SSL-моделі може допомогти краще узгодити її представлення з новим набором даних.

Оскільки SSL-модель повертає часову послідовність векторів представлень, для отримання одного представлення всього аудіозапису використовується агрегація ознак. У реалізованій системі представлення узагальнюється за часовою віссю з урахуванням валідної довжини сигналу. Це важливо через те, що аудіозаписи в групі мають різну тривалість і можуть містити доповнені ділянки. Якщо не враховувати маску доповнення, штучно додані нульові ділянки могли б впливати на фінальне представлення. Тому при агрегуванні SSL-представлень використовуються лише ті часові позиції, що відповідають реальному аудіосигналу.

Після агрегації формується вектор SSL-ознак фіксованої розмірності. Цей вектор не передається безпосередньо до фінального класифікатора, а спочатку надходить у проєкційний модуль. Проєкція потрібна для того, щоб привести SSL-представлення до спільного простору з іншими гілками системи. Оскільки LFCC, phase, prosody, wav2vec2, WavLM і RawNet2 гілка можуть мати різні вихідні розмірності, проєкційні шари вирівнюють ці представлення перед їх подальшим злиттям.

Окремою практичною можливістю є використання кешування ембеддингів SSL. Обчислення WavLM або wav2vec2-представлень для кожного аудіозапису є значно дорожчим, ніж обчислення простих

спектральних або статистичних ознак. Якщо SSL-модель використовується в замороженому режимі, її виходи для конкретних аудіофайлів не змінюються між епохами навчання. У такому випадку доцільно один раз обчислити SSL-ембеддинги та зберегти їх у cache, а під час подальшого навчання завантажувати вже готові представлення. Це дозволяє зменшити час навчання та ефективніше використовувати обчислювальні ресурси.

У системі SSL-cache є опційним механізмом. Якщо в таблиці метаданих заданий шлях до збереженого файлу представлень, модель може використовувати попередньо обчислене SSL-представлення замість повторного проходження аудіосигналу через WavLM або wav2vec2. Якщо кеш не використовується, представлення обчислюються безпосередньо під час навчання або оцінювання. Такий підхід зберігає гнучкість системи, на етапі швидких експериментів можна застосовувати кеш, а під час донавчання або зміни SSL-конфігурації – обчислювати представлення заново.

3.5 Фазова гілка

Фазова гілка призначена для аналізу характеристик аудіосигналу, пов’язаних із фазовою структурою його спектрального представлення. У класичному спектральному аналізі значна увага часто приділяється амплітудному спектру, оскільки він показує розподіл енергії сигналу за частотами. Водночас фазова складова також містить інформацію про часову організацію сигналу та узгодженість його частотних компонентів. Для синтезованого мовлення це може бути корисним, оскільки нейронні генератори не завжди відтворюють фазову поведінку так само, як вона формується у природному мовленні.

У реалізованій системі фазова гілка є легкою статистичною гілкою (рисунок 3.4). Вона не використовує окрему глибоку нейромережу, а формує набір характеристик на основі короткочасного перетворення

Фур'є. Спочатку для вхідної хвильової форми обчислюється STFT, після чого з комплексного спектра виділяється фазова компонента. Далі розраховується різниця фаз між сусідніми часовими фреймами. Така величина дозволяє оцінити не абсолютну фазу окремого фрейму, а зміну фазової структури сигналу в часі.

```python
class PhaseBranch(nn.Module):
    """Compact STFT phase branch.

    It uses sin/cos phase, wrapped time differences, and wrapped frequency
    differences. The frequency axis is pooled to a fixed number of bins so
    the branch remains light enough for Colab experiments.
    """

    def __init__(…

    @staticmethod
    def _wrap_delta(delta: torch.Tensor) -> torch.Tensor:…

    def _resize_frequency(self, tensor: torch.Tensor) -> torch.Tensor:…

    def forward(self, waveform: torch.Tensor, padding_mask: torch.Tensor) -> torch.Tensor:
        window = torch.hann_window(self.win_length, device=waveform.device, dtype=waveform.dtype)
        stft = torch.stft(
            waveform,
            n_fft=self.n_fft,
            hop_length=self.hop_length,
            win_length=self.win_length,
            window=window,
            center=True,
            return_complex=True,
        )
        phase = torch.angle(stft)
        delta_time = F.pad(self._wrap_delta(torch.diff(phase, dim=2)), (1, 0))
        delta_freq = F.pad(self._wrap_delta(torch.diff(phase, dim=1)), (0, 0, 1, 0))

        components = [
            torch.sin(phase),
            torch.cos(phase),
            torch.sin(delta_time),
            torch.cos(delta_time),
            torch.sin(delta_freq),
            torch.cos(delta_freq),
        ]
        resized = [self._resize_frequency(component) for component in components]
        features = torch.cat(resized, dim=1).transpose(1, 2)
        valid_mask = frame_valid_mask(padding_mask, self.hop_length, features.size(1))
        return torch.cat([masked_mean(features, valid_mask), masked_std(features, valid_mask)], dim=-1)
```

Рисунок 3.4 – Приклад класу фазової гілки

За значення n_fft = 512 кількість частотних компонентів дорівнює 257. У системі використовуються середнє значення та стандартне відхилення фазової різниці. Середнє значення дозволяє отримати узагальнену характеристику фазової зміни, а стандартне відхилення описує її варіативність упродовж аудіозапису. Після конкатенації цих статистик формується вектор фазових ознак розмірності 514.

Під час обчислення фазових ознак також враховується маска доповнень, сформована на етапі підготовки даних. Це потрібно для того, щоб статистики не враховували фрейми, які відповідають штучно доповненим нульовим ділянкам. Таким чином, фазове представлення формується тільки за валідною частиною аудіосигналу, що дозволяє коректно обробляти записи різної тривалості.

3.6 Просодична гілка

Просодична гілка призначена для отримання характеристик, пов’язаних із загальною динамікою мовного сигналу. У широкому розумінні просодія охоплює такі властивості мовлення, як інтонація, темп, паузи, енергія, ритмічна організація та зміна висоти тону.

У межах цієї гілки аудіосигнал поділяється на фрейми фіксованої довжини (рисунок 3.5).

```python
class ProsodyBranch(nn.Module):
    """Prosodic branch with energy, ZCR, voiced ratio, and optional F0 stats."""

    def __init__(self, sample_rate: int, win_length: int, hop_length: int) -> None:···

    def _pitch_track(self, waveform: torch.Tensor, num_frames: int) -> torch.Tensor:···

    def forward(self, waveform: torch.Tensor, padding_mask: torch.Tensor) -> torch.Tensor:
        if waveform.size(1) < self.win_length:
            pad = self.win_length - waveform.size(1)
            waveform = F.pad(waveform, (0, pad))
            padding_mask = F.pad(padding_mask, (0, pad), value=True)

        frames = waveform.unfold(dimension=1, size=self.win_length, step=self.hop_length)
        num_frames = frames.size(1)
        valid_mask = frame_valid_mask(padding_mask, self.hop_length, num_frames)

        rms = frames.pow(2).mean(dim=-1).clamp_min(1e-8).sqrt()
        log_energy = torch.log(rms.clamp_min(1e-8))
        signs = torch.sign(frames)
        zcr = (signs[:, :, 1:] * signs[:, :, :-1] < 0).to(waveform.dtype).mean(dim=-1)

        pitch = self._pitch_track(waveform, num_frames)
        voiced = (pitch > 1.0).to(waveform.dtype)
        log_f0 = torch.log(pitch.clamp_min(1.0)) * voiced
        delta_energy = F.pad(torch.diff(log_energy, dim=1), (1, 0))
        delta_f0 = F.pad(torch.diff(log_f0, dim=1), (1, 0)) * voiced

        features = torch.stack(
            [log_energy, rms, zcr, log_f0, voiced, delta_energy, delta_f0],
            dim=-1,
        )
        return torch.cat([masked_mean(features, valid_mask), masked_std(features, valid_mask)], dim=-1)
```

Рисунок 3.5 – Приклад класу просодичної гілки

Для кожного фрейму обчислюються енергетичні та динамічні характеристики, зокрема RMS-енергія, логарифмічна енергія, зміна енергії між сусідніми фреймами, абсолютна зміна енергії, частота перетинів нульового рівня та зміна частоти перетинів нульового рівня. Такі ознаки не описують повну інтонаційну структуру мовлення, але дають змогу оцінити загальну енергетичну динаміку, різкість змін сигналу та характер його часової варіативності.

Додатково в гілці обчислюється частка голосоподібних ділянок сигналу. Ця характеристика не є повноцінною оцінкою частки аудіосигналу на основі фундаментальної частоти, але дозволяє приблизно оцінити частку фреймів, у яких енергія сигналу перевищує відносний поріг. Така оцінка може бути корисною для опису загальної активності мовлення в записі та співвідношення енергійних і слабко виражених фрагментів.

Після обчислення фреймових характеристик вони узагальнюються за допомогою статистичного pooling. Як і в інших гілках, використовуються середнє значення та стандартне відхилення за валідними фреймами. У результаті формується вектор просодичних ознак фіксованої розмірності. У реалізації ця гілка повертає 12-вимірне представлення, яке далі може бути передане до проєкційного модуля та об’єднане з іншими гілками моделі.

Роль просодичної гілки полягає в тому, щоб додати до системи прості, але інформативні часово-енергетичні характеристики. Для синтезованого мовлення такі характеристики можуть бути корисними, оскільки генеративні системи можуть відрізнятися від природного мовлення за рівномірністю енергії, характером пауз, різкістю переходів або динамікою сигналу.

3.7 RawNet2 гілка

Ще одним компонентом системи є RawNet2 гілка, яка працює безпосередньо з хвильовою формою аудіосигналу. Її призначення полягає в

тому, щоб дати моделі можливість виділяти ознаки з raw waveform без попереднього перетворення сигналу в LFCC, спектрограму або інше явно задане представлення. Такий підхід є важливим, оскільки частина артефактів синтезованого мовлення може проявлятися на рівні самої хвильової форми та бути не повністю відображеною в ручних ознаках.

Ідея роботи з raw waveform використовується в підходах «end-to-end anti-spoofing». Зокрема, RawNet2 було запропоновано як «end-to-end» модель для задачі виявлення атак із підробкою мовного аудіо, що працює безпосередньо з аудіосигналом [9].

Першим етапом є SincConv1d-шар (рисунок 3.6). Цей шар реалізує набір навчаних смугових фільтрів, які застосовуються до сирої хвильової форми. На відміну від фіксованих спектральних ознак, такі фільтри можуть адаптуватися під задачу під час навчання. У реалізації використовується 64 sinc-фільтри з розміром ядра 251 семпл. Після цього сигнал проходить через нормалізацію, нелінійність і pooling, що формує початкове згорткове представлення.

```python
class SincConv1d(nn.Module):
    """Learnable band-pass convolution used by RawNet/SincNet-style models."""

    def __init__(…

    def forward(self, waveform: torch.Tensor) -> torch.Tensor:
        low = self.min_low_hz + self.low_hz_.abs()
        low = low.clamp(max=self.sample_rate / 2 - self.min_band_hz - 1.0)
        high = (low + self.min_band_hz + self.band_hz_.abs()).clamp(max=self.sample_rate / 2 - 1.0)
        band = (high - low).clamp_min(self.min_band_hz)

        low_times = 2.0 * np.pi * low * self.time
        high_times = 2.0 * np.pi * high * self.time
        left = (torch.sin(high_times) - torch.sin(low_times)) / (np.pi * self.time)
        center = 2.0 * band
        filters = torch.cat([left.flip(dims=[1]), center, left], dim=1)
        filters = filters * self.window
        filters = filters / filters.abs().sum(dim=1, keepdim=True).clamp_min(1e-8)
        return F.conv1d(waveform, filters.unsqueeze(1), padding=self.kernel_size // 2)
```

Рисунок 3.6 – Приклад класу SincConv1d

Далі використовуються залишкові часові блоки.

Кожен такий блок містить послідовність одномірних згорток, групову нормалізацію, LeakyReLU, агрегацію і метод регуляризації «dropout». Наявність швидкого з'єднання дозволяє зберігати інформацію з попереднього рівня та полегшує навчання глибшого представлення. У конфігурації гілки використовується послідовність каналів 64, 128 і 128, що дозволяє поступово збільшувати кількість ознакових карт під час обробки сигналу.

Після згорткових блоків отримана послідовність ознак передається до GRU-шару. Його завдання полягає в моделюванні часових залежностей між локальними представленнями сигналу. У реалізованій гілці використовується двонапрямлений GRU з прихованою розмірністю 128. Завдяки двонапрямленій обробці модель може враховувати контекст як з попередніх, так і з наступних часових позицій у межах аудіозапису.

Фінальне представлення RawNet2 гілки формується за допомогою статистичної агрегації виходу GRU (рисунок 3.7).

```python
class RawNet2Branch(nn.Module):
    """RawNet2-inspired branch operating directly on the waveform.

    The original RawNet2 family uses raw-audio convolutional filters and
    residual temporal modeling. This compact version keeps that spirit while
    staying light enough to combine with LFCC, WavLM, phase, and prosody.
    """

    def __init__(
        self,
        sample_rate: int,
        sinc_filters: int = 64,
        sinc_kernel_size: int = 251,
        channels: list[int] | None = None,
        gru_hidden: int = 128,
        dropout: float = 0.1,
    ) -> None:
        super().__init__()
        channels = channels or [64, 128, 128]
        self.total_stride = 1
        self.sinc = SincConv1d(…
        self.frontend = nn.Sequential(…
        self.total_stride *= 3

        blocks: list[nn.Module] = []
        in_channels = sinc_filters
        for out_channels in channels:…
        self.blocks = nn.Sequential(*blocks)
        self.gru = nn.GRU(…
        self.output_dim = gru_hidden * 4

    def forward(self, waveform: torch.Tensor, padding_mask: torch.Tensor) -> torch.Tensor:
        features = self.sinc(waveform.unsqueeze(1))
        features = self.frontend(features)
        features = self.blocks(features).transpose(1, 2)
        valid_mask = frame_valid_mask(padding_mask, self.total_stride, features.size(1))
        sequence, _ = self.gru(features)
        return torch.cat([masked_mean(sequence, valid_mask), masked_std(sequence, valid_mask)], dim=-1)
```

Рисунок 3.7 – Приклад класу RawNet2Branch

Як і в інших гілках, обчислюються середнє значення та стандартне відхилення за валідними часовими позиціями. У результаті формується вектор фіксованої розмірності, який описує аудіозапис на основі хвильових ознак. За прихованої розмірності GRU 128 і двонапрямленої обробки фінальний вектор має розмірність 512.

Під час роботи цієї гілки також враховується маска доповнень. Це потрібно через те, що після згорткових і операцій агрегації довжина послідовності змінюється, але модель все одно повинна розуміти, які часові позиції відповідають реальному аудіосигналу. Для цього фактична довжина сигналу перераховується з урахуванням загального кроку згортки гілки, після чого формується маска для валідних часових кроків. Такий механізм дозволяє коректно виконувати pooling навіть для записів різної тривалості.

Ця гілка є найбільш складною серед додаткових акустичних компонентів, оскільки вона містить навчальні фільтри, згорткові блоки та рекурентний шар. Через це її використання збільшує кількість параметрів моделі та обчислювальне навантаження. Водночас вона може надати моделі доступ до інформації, яка не завжди повністю зберігається після ручного перетворення сигналу в кепстральні або статистичні ознаки. Тому її доцільно розглядати як розширення базової архітектури.

3.8 Інтеграція ознак і класифікаційний модуль

Після формування ознак у паралельних гілках системи виникає задача їх узгодження та об’єднання. Кожна гілка описує аудіосигнал з окремого боку – LFCC-гілка формує спектральне представлення, SSL-гілка надає високорівневі нейромережеві ознаки, фазова гілка характеризує зміну фазової структури, просодична гілка описує енергетичну та часову динаміку, а RawNet2-подібна гілка виділяє ознаки безпосередньо з хвильової форми. Оскільки ці представлення мають різну природу та різну

розмірність, перед класифікацією їх необхідно привести до спільного формату.

У розробленій системі інтеграція ознак виконується у кілька послідовних етапів. Спочатку кожна активна гілка формує власний вектор ознак. Далі цей вектор передається до проєкційного модуля, який переводить його у спільний простір фіксованої розмірності. Після цього отримані проєктовані представлення об’єднуються за допомогою механізму уваги. Фінальне злиття передається до класифікаційного модуля, який формує вихідний «logit» для задачі бінарної класифікації.

Необхідність проєкції пов’язана з тим, що виходи різних гілок не є однорідними. Наприклад, LFCC-гілка після статистичного pooling формує вектор розмірності 120, фазова гілка – значно більший вектор на основі статистик фазових бінів, просодична гілка – компактне 12-вимірне представлення, а SSL і RawNet2 подібні гілки мають власні розмірності, визначені їх внутрішньою архітектурою. Без проміжного узгодження такі ознаки було б складно коректно об’єднати, оскільки гілки з більшою розмірністю могли б непропорційно впливати на фінальне представлення.

Проєкційний модуль вирішує цю проблему шляхом перетворення кожного вхідного вектора в представлення однакової розмірності. У реалізації такий модуль складається з нормалізації, лінійного шару, нелінійної функції активації та механізму «dropout». Нормалізація допомагає стабілізувати масштаб ознак, лінійний шар виконує власне перетворення у спільний простір, нелінійність підвищує виразність перетворення, а «dropout» використовується як регуляризаційний механізм. Завдяки цьому вихід кожної гілки подається до модуля злиття ознак у вигляді узгодженого векторного представлення.

Після проєкції система отримує набір embedding-векторів, кожен із яких відповідає окремій активній гілці. Якщо в експерименті використовується лише LFCC та WavLM, модуль злиття ознак працює з двома представленнями. Якщо активовано додаткові компоненти, кількість

представлень збільшується. Така організація робить архітектуру гнучкою, вона не потребує переписування класифікатора при зміні набору ознак, оскільки всі гілки попередньо приводяться до спільного формату.

Для об’єднання гілок у системі використовується механізм уваги. Ідея такого підходу полягає в тому, що внесок кожної ознакової гілки не задається вручну, а визначається самою моделлю. Механізм злиття обчислює ваги для окремих гілок і використовує їх для формування спільного представлення аудіозапису. У загальному вигляді такий підхід пов’язаний з ідеєю керованого злиття мультимодальних представлень, де модель навчається визначати, які джерела інформації є більш важливими для конкретної задачі [13].

У контексті цієї роботи механізм уваги використовується не для класичної мультимодальності на кшталт тексту, зображень і аудіо, а для поєднання різних типів аудіоознак. Кожна гілка розглядається як окреме джерело інформації про той самий аудіосигнал. Це дозволяє застосувати ідею зваженого злиття до гібридної аудіомоделі, де система може навчатися надавати більший або менший внесок LFCC, SSL, фазовим, просодичним або хвильовим ознакам залежно від їхньої корисності для класифікації.

На відміну від простого об’єднання ознак через конкатенацію, цей метод додає проміжний механізм оцінювання важливості гілок. Якщо всі представлення просто сконкатенувати, модель отримує один великий вектор, але не має явного механізму для регулювання внеску кожної групи ознак на рівні злиття. Після цього для кожної гілки формується коефіцієнт важливості, який впливає на її внесок у фінальне представлення. Це робить злиття більш керованим і дає можливість аналізувати відносну роль різних гілок.

Після злиття ознак фінальне представлення надходить до класифікаційного модуля. У розробленій системі класифікатор реалізовано як багатошаровий перцептрон. Його завдання полягає в тому, щоб на основі об’єднаного вектора представлень визначити, чи належить аудіозапис до

класу «bonafide» або «spoof». Оскільки задача є бінарною, класифікатор формує один вихідний «logit». Цей «logit» є ненормалізованою оцінкою, яка надалі може бути перетворена на ймовірнісну інтерпретацію за допомогою сигмоїдної функції.

Фінальне рішення може прийматися на основі заданого порога. Наприклад, для демонстраційного використання може застосовуватися стандартний поріг 0.5, якщо оцінка перевищує це значення, запис класифікується як «spoof»; якщо ні – як «bonafide». Водночас під час експериментального оцінювання якості такий поріг не обов’язково є оптимальним. Для розрахунку EER використовується окремий поріг, за якого частоти помилкового прийняття та помилкового відхилення наближаються одна до одної. Тому оцінку моделі слід розглядати як неперервну оцінку, а не тільки як готову бінарну відповідь.

Загалом інтеграційний блок виконує роль центрального компонента гібридної архітектури. Саме він поєднує різнорідні представлення аудіосигналу в єдиний простір ознак, придатний для класифікації. Завдяки проєкційним шарам система узгоджує розмірності виходів різних гілок, механізм уваги регулює їхній внесок, а MLP-класифікатор формує фінальну оцінку синтетичного походження запису. Така структура дозволяє гнучко змінювати набір активних ознак і оцінювати їх вплив на якість детекції.

3.9 Програмна реалізація системи

Програмна реалізація системи побудована як модульний Python проєкт, у якому окремі частини відповідають за підготовку даних, виділення ознак, побудову моделі, навчання, оцінювання та запуск інференсу. Такий підхід дозволяє розділити логіку роботи системи на незалежні компоненти та спростити подальше розширення архітектури. Замість того, щоб реалізовувати весь pipeline в одному файлі, функціональність винесена в окремі модулі, кожен із яких має чітко визначене призначення.

Основним пакетом системи є «spoof_hybrid». У ньому зосереджено код, пов’язаний із машинним навчанням – підготовка аудіоданих, формування ознак, архітектура моделі та допоміжні утиліти. Така організація дозволяє використовувати один і той самий пакет у різних сценаріях, під час навчання, оцінювання, інференсу окремого файлу або запуску серверної частини застосунку.

Папка «spoof_hybrid/data» містить модулі для роботи з вхідними аудіоданими. Її функціональність охоплює завантаження аудіофайлів, приведення сигналу до потрібного формату, обробку файлів протоколу, створення представлення, формування dataset-об’єктів і підготовку груп. Цей блок є початковою частиною конвеєр обробки, оскільки саме він відповідає за те, щоб різні набори даних були подані в однаковому форматі для подальшої роботи моделі.

У цій папці реалізовано кілька окремих файлів. Модуль «audio.py» відповідає за завантаження та базову обробку аудіосигналу. Модуль «parsers.py» використовується для перетворення файлів протоколу або інших описів наборів даних у внутрішній формат. Модуль «dataset.py» реалізує dataset-рівень, який повертає окремі приклади для моделі. Модуль «collate.py» відповідає за об’єднання окремих прикладів у групі, а «sampler.py» використовується для ефективнішого групування записів під час навчання.

Папка «spoof_hybrid/features» містить модулі виділення ознак. У ній реалізовано окремі гілки для різних типів представлень аудіосигналу. Модуль «lfcc.py» відповідає за формування LFCC-ознак. Модуль «ssl.py» використовується для отримання SSL-представлень за допомогою моделей WavLM або wav2vec2. Модуль «phase.py» реалізує фазові характеристики, а «prosody.py» – компактні енергетичні та часові характеристики. Додатково в проєкті є «rawnet2.py», який реалізує RawNet2-подібну гілку для роботи з хвильовою формою сигналу.

Окрему роль у папці «features відіграє модуль «pooling.py». Його призначення полягає в узагальненні часових послідовностей ознак у вектори фіксованої розмірності. Це потрібно тому, що більшість аудіоознак спочатку формуються як послідовність фреймових або часових представлень, а класифікаційний модуль потребує одного вектора для всього аудіозапису. Завдяки використанню «pooling» система може працювати із записами різної довжини.

Також у проєкті передбачено механізм кешування SSL-представлень. Для цього використовується модуль «cache_ssl.py», який дозволяє попередньо обчислити SSL-ембеддинги та зберегти їх для подальшого використання. Такий підхід є корисним у тих випадках, коли SSL-модель використовується як заморожений «feature extractor» і її виходи не змінюються між епохами навчання. Кешування дозволяє зменшити час повторних експериментів і знизити навантаження на GPU.

Папка «spoof_hybrid/models» містить компоненти архітектури нейронної мережі. У ній реалізовано основний клас гібридного детектора, окремі модулі, проєкційні шари, механізм злиття та класифікатор. Така структура відображає логіку самої моделі – спочатку кожна гілка формує власне представлення, потім ці представлення проєктуються в узгоджений простір, після чого об’єднуються та передаються до класифікаційного модуля.

Файл «hybrid_detector.py» є центральним елементом моделі. Він об’єднує активні гілки ознак, викликає відповідні модулі ознак, передає їхні виходи до проєкційних шарів, виконує злиття та формує результат класифікації. Саме цей модуль задає загальний прямий прохід моделі та дозволяє змінювати набір активних гілок залежно від конфігурації експерименту.

Файл «branches.py» містить допоміжну логіку, пов’язану з організацією окремих гілок моделі. Файл «projection.py» реалізує приведення ознак різної розмірності до спільного простору. Файл

«fusion.py» відповідає за механізм уваги, тобто за зважене об’єднання представлень. Файл «classifier.py» містить класифікаційний модуль, який на основі фінального представлення формує вихідний «logit» для бінарної класифікації.

Папка «spoof_hybrid/utils» містить допоміжні засоби, які використовуються в різних частинах конвеєр обробки. До них належать модулі для роботи з конфігураціями, встановлення випадкового «seed», обчислення метрик, збереження файлів контрольних точок і логування. Винесення такої функціональності в окрему папку робить основні скрипти коротшими та зрозумілішими, оскільки вони не містять усіх службових операцій безпосередньо у своєму коді.

Окреме значення має папка «configs». У ній зберігаються всі YAML-конфігурації для різних варіантів запуску системи. Конфігураційний підхід дозволяє змінювати параметри експерименту без редагування основного коду. Через конфігурацію задаються шляхи до даних, активні гілки моделі, параметри LFCC, SSL-моделі, режим заморожування або донавчання, «batch size», «learning rate», кількість епох, параметри «optimizer», «scheduler» та інші налаштування.

Завдяки конфігураціям можна запускати різні варіанти моделі: тільки LFCC, тільки SSL, LFCC та WavLM, повну гібридну модель із додатковими гілками або варіанти для донавчання на іншому наборі даних. Це особливо важливо для експериментальної частини роботи, оскільки дозволяє проводити дослідження та порівнювати внесок різних компонентів без зміни програмної логіки моделі.

У корені проєкту розміщено основні скрипти запуску. Файл «train.py» використовується для навчання моделі відповідно до заданої YAML-конфігурації. Файл «evaluate.py» призначений для оцінювання збереженї контрольної точки на вибраному наборі. Файл «infer.py» використовується для отримання прогнозу для окремого аудіофайлу. Такий поділ є зручним, оскільки навчання, оцінювання та інференс мають різні

сценарії використання, але при цьому спираються на спільні модулі «spoof_hybrid».

Файл «requirements.txt» містить перелік основних залежностей проєкту. До них належать бібліотеки для глибокого навчання, обробки аудіо, роботи з табличними даними, побудови графіків, конфігурацій, серверної частини та тестування. Центральними інструментами є PyTorch і torchaudio, які використовуються для побудови нейронної мережі та аудіообробки, а також transformers, що забезпечує використання попередньо навчених SSL-моделей. Для роботи з даними застосовуються numpy, pandas і scikit-learn, а для завантаження та обробки аудіофайлів – librosa і soundfile.

Окрім дослідницької частини, у проєкті є папки «server» і «frontend». Вони відповідають за локальний застосунок, який дозволяє використовувати навчену модель через зручний інтерфейс. Серверна частина виконує приймання файлу, запуск моделі та повернення результату, а клієнтська частина забезпечує взаємодію з користувачем. Детальний опис цього застосунку наведено в окремому підрозділі, тому в межах опису програмної структури достатньо зазначити, що серверно-клієнтська частина є додатковим рівнем над ML-модулем.

Загалом програмна реалізація системи побудована так, щоб відокремити дослідницьку логіку від службових і прикладних компонентів.

Модулі «data», «features» і «models» формують ядро системи детекції. Папка «utils» забезпечує допоміжну функціональність, необхідну для стабільного запуску експериментів.

Конфігурації дають змогу змінювати параметри моделі та даних без редагування коду.

Окремі скрипти забезпечують навчання, оцінювання та інференс, а серверна й клієнтська частини демонструють можливість практичного використання навченого детектора.

3.10 Навчання моделі

Навчання моделі є окремим етапом роботи системи, під час якого налаштовуються параметри навчальних компонентів гібридної архітектури: проєкційних шарів, механізму злиття представлень, класифікатора та додаткових ознакових гілок. SSL-модель WavLM у фінальній конфігурації використовується як попередньо навчений екстрактор високорівневих мовних представлень, тоді як її параметри залишаються замороженими. Метою етапу навчання є отримання контрольної точки моделі, здатної відрізняти справжні аудіозаписи від синтезованих або підроблених. У межах розробленої системи процес навчання реалізовано як конфігурований конвеєр обробки, що запускається через окремий скрипт «train.py».

Використання окремого скрипта навчання дозволяє відокремити цей процес від оцінювання та інференсу. Це важливо, оскільки під час навчання потрібно виконувати операції, які не потрібні під час звичайного використання моделі: розбиття даних на train/dev, обчислення функції втрат, оновлення параметрів, контроль якості на валідаційній частині, збереження файлів контрольних точок і ведення історії навчання. Завдяки такій організації модель можна багаторазово навчати з різними конфігураціями, не змінюючи основний код архітектури.

Навчання моделі виконується на основі YAML-конфігурації. У конфігураційному файлі задаються шляхи до даних, параметри аудіообробки, активні гілки моделі, параметри ознак, налаштування optimizer, scheduler, кількість епох, batch size, learning rate, режим роботи SSL-моделі, шлях для збереження результатів та інші службові параметри. Такий підхід робить експерименти відтворюваними, оскільки кожен запуск може бути пов’язаний із конкретним конфігураційним файлом.

На початку роботи train.py зчитує конфігурацію та встановлює службові параметри запуску. До таких параметрів належать seed для відтворюваності, пристрій обчислень, шляхи до директорій з результатами

та параметри логування. Якщо доступний графічний процесор, модель і batch переносяться на GPU; інакше навчання може виконуватися на CPU, хоча для SSL-гілок і великих моделей це суттєво повільніше.

Після зчитування конфігурації виконується підготовка даних. Якщо використовується вже готовий файл з метаданими, система завантажує його та відбирає записи відповідно до потрібних split-частин. Якщо ж дані подані у вигляді ASVspoof-подібної структури з файлами протоколів, вони попередньо перетворюються до представлення метаданних. Далі на основі підготовленої таблиці створюються dataset-об’єкти для навчальної та валідаційної частин.

Для навчання використовуються завантажувачі даних, які формують групи аудіозаписів. Оскільки записи мають різну тривалість, під час об’єднання прикладів у групи виконується доповнення до максимальної довжини в межах цієї групи.

Важливим елементом навчального конвеєру є групування за тривалістю. Його використання дозволяє групувати аудіозаписи близької довжини в одну групу, зменшуючи кількість зайвого доповнення. Це особливо корисно під час роботи з мовними наборами даних, де тривалість записів може значно відрізнятися. Менша кількість доповнених елементів знижує обсяг непотрібних обчислень і робить використання пам’яті ефективнішим.

Після підготовки завантажувача даних створюється екземпляр гібридної моделі. Набір активних гілок визначається конфігурацією. Наприклад, в одному експерименті може використовуватися лише LFCC та WavLM, в іншому – LFCC, WavLM, phase та prosody, а в розширеному варіанті – додатково wav2vec2 або RawNet2-подібна гілка. Завдяки цьому той самий скрипт навчання може використовуватися для різних архітектурних варіантів.

Під час ініціалізації моделі для кожної активної гілки створюється відповідний модуль виділення ознак, проєкційний шар, модуль злиття і

класифікатор. Якщо в конфігурації задано використання SSL-моделі, завантажується відповідна попередньо навчена модель WavLM або wav2vec2. У базовому режимі SSL-модель може бути заморожена, тобто її параметри не оновлюються під час навчання. У такому випадку навчання зосереджується на проєкційних шарах, модулі злиття та класифікаторі.

Разом із цим у системі передбачена можливість часткового донавчання SSL-моделі. Якщо конфігурація задає розморожування останніх шарів, частина параметрів SSL-гілки включається до навчання. Для таких параметрів може задаватися окрема швидкість навчання, менша за основний швидкість навчання. Це потрібно через те, що попередньо навчена SSL модель уже містить корисні мовленнєві представлення, і занадто агресивне оновлення її параметрів може погіршити узагальнювальну здатність.

Для оптимізації параметрів використовується AdamW. Цей оптимізатор є зручним для навчання нейронних мереж, оскільки поєднує адаптивне оновлення параметрів із регуляризація ваг. Регуляризація ваг допомагає зменшити ризик перенавчання, особливо коли модель має велику кількість параметрів або коли навчальна вибірка містить обмежену кількість різноманітних прикладів.

Функцією втрат у задачі є «BCEWithLogitsLoss», binary cross-entropy loss with logits. Вона використовується для бінарної класифікації та працює безпосередньо з «logit» значеннями, які формує класифікатор. Такий варіант є чисельно стабільнішим, ніж окреме застосування «binary cross-entropy» і «sigmoid». У межах цієї задачі модель навчається формувати високе «logit» значення для підроблених записів і низьке значення для «bonafide» записів.

Оскільки в наборах даних кількість прикладів різних класів може бути незбалансованою, у функції втрат передбачено використання «pos_weight». Цей параметр дозволяє змінити вагу позитивного класу, тобто класу «spoof», під час обчислення втрат. Якщо позитивний і негативний класи представлені нерівномірно, «pos_weight» допомагає частково компенсувати дисбаланс і зменшити зміщення моделі в бік переважаючого класу.

У конфігурації може бути задано автоматичний розрахунок «pos_weight». У такому випадку він обчислюється на основі співвідношення кількості «bonafide» та «spoof» прикладів у навчальній частині. Це дозволяє не задавати вагу вручну для кожного нового набору даних, а розраховувати її відповідно до фактичного розподілу класів.

Під час навчання також використовується «gradient clipping». Цей механізм обмежує максимальну норму градієнтів і допомагає уникнути нестабільних оновлень параметрів. Для моделей, які містять різні типи компонентів, зокрема згорткові, рекурентні, SSL і MLP-блоки, контроль градієнтів є корисним для стабілізації процесу оптимізації.

Після кожної епохи модель оцінюється на dev-частині. Це дозволяє відстежувати не тільки значення train та loss, а й якість узагальнення на даних, які не використовуються для оновлення параметрів. На цьому етапі обчислюються валідаційні метрики, зокрема AUC ROC, EER, F1 та loss. Саме ці значення використовуються для вибору найкращої контрольної точки і контролю можливого перенавчання.

Для вибору найкращої моделі використовується моніторинг заданої валідаційної метрики. У різних експериментах як основний критерій може використовуватися «val_auc» або «val_eer». Якщо моніториться AUC ROC, кращим вважається більше значення. Якщо моніториться EER, кращим є менше значення, оскільки EER відображає рівень помилки в точці балансу між двома типами помилок. Такий механізм дозволяє зберігати не просто останній стан моделі, а саме ту контрольну точку, яка показала найкращу якість на dev-частині.

Щоб уникнути зайвого навчання після стабілізації результатів, у конвеєрі навчання передбачено дострокову зупинку. Якщо протягом заданої кількості епох валідаційна метрика не покращується, навчання може бути зупинене достроково. Це зменшує час експериментів і допомагає уникнути ситуації, коли модель починає краще запам’ятовувати train-частину, але не покращує якість на dev-даних.

Під час навчання система зберігає кілька типів контрольних точок. Поточний стан може зберігатися у файлі «train_in_progress.pt», що корисно для відновлення навчання у випадку переривання. Після завершення епохи може зберігатися «train_epoch_complete.pt». Найкраща модель за вибраною валідаційною метрикою зберігається у файлі «best.pt», а останній стан навчання – у «last.pt». Такий набір контрольних точок забезпечує гнучкість, можна або використовувати найкращу модель, або продовжити навчання з останнього стану.

Окрім контрольних точок, зберігається історія навчання. Вона містить значення loss і метрик за епохами та дозволяє проаналізувати динаміку оптимізації. Це корисно для виявлення перенавчання, нестабільності, надто великого навчальної швидкості або недостатньої кількості епох. Історія навчання також може використовуватися для побудови графіків у експериментальному розділі.

Донавчання використовується тоді, коли вже навчена модель переноситься на інший домен або інший набор даних. Наприклад, модель, навчена на одному наборі даних, може бути донавчена на ASVspoof5 або на змішаному наборі з іншими типами синтетичного мовлення. У такому випадку контрольна точка попередньої моделі використовується як стартова точка, а подальше навчання дозволяє адаптувати параметри до нових умов. Це важливо для задачі детекції синтезованого мовлення, оскільки типи атак і властивості аудіозаписів можуть суттєво відрізнятися між наборами.

Під час донавчання особливо важливо контролювати швидкість навчання, кількість розморожених шарів і критерій ранньої зупинки. Якщо навчання є занадто агресивним, модель може втратити частину вже сформованих корисних представлень. Якщо ж оновлення параметрів занадто слабке, адаптація до нового набору даних може бути недостатньою. Тому донавчання у системі організований через конфігурації, що дозволяє окремо налаштовувати режим роботи SSL-гілки, оптимізатор і параметри навчання.

Результатом навчання є збережений checkpoint моделі та набір службових файлів із метриками й історією оптимізації. Надалі найкращий checkpoint використовується для оцінювання на eval-частині, порівняння різних конфігурацій і запуску локального застосунку. Таким чином, навчальний конвеєр є центральним етапом, який пов’язує підготовлені дані, архітектуру моделі та подальший експериментальний аналіз.

Отже, процес навчання в розробленій системі реалізовано як гнучкий конфігурований конвеєр. Він охоплює підготовку даних, ініціалізацію гібридної моделі, вибір функції втрат, оптимізатор, підтримку змішаної точності, gradient clipping, early stopping, checkpointing і fine-tuning. Така організація дозволяє проводити серію відтворюваних експериментів із різними наборами ознак і режимами навчання без зміни основного програмного коду.

3.11 Реалізація серверної частини та користувацького інтерфейсу

Окрім модулів навчання та оцінювання, у межах проєкту реалізовано локальний застосунок для демонстрації роботи розробленої моделі. Його призначення полягає в тому, щоб надати користувачу можливість перевірити окремий аудіо або відео файл без прямого запуску Python-скриптів з командного рядка. Такий застосунок не змінює архітектуру моделі та не є окремим методом детекції, а виконує роль прикладної оболонки над уже навченим гібридним детектором.

Необхідність такого компонента пов’язана з тим, що результати роботи моделі мають бути доступні не лише в експериментальному середовищі, а й у більш зрозумілій для користувача формі. Під час навчання та оцінювання модель працює з файлами метаданих, контрольними точками, YAML-конфігураціями та файлами оцінки. Це зручно для дослідження, але менш придатне для швидкої перевірки окремого запису. Серверно-клієнтська частина вирішує цю задачу, надаючи простий сценарій

використання – завантажити файл, передати його на аналіз і отримати результат класифікації.

Серверна частина реалізована з використанням FastAPI. Вона відповідає за приймання файлу від клієнтської частини, перевірку формату, тимчасове збереження, підготовку аудіосигналу, запуск моделі та формування відповіді. Такий поділ є зручним, оскільки вся логіка, пов’язана з машинним навчанням, залишається на backend-рівні. Frontend при цьому не працює напряму з checkpoint або PyTorch-моделлю, а лише надсилає файл і отримує готовий результат.

У серверній частині передбачено окремі маршрути для перевірки стану застосунку, отримання інформації про модель і виконання аналізу файлу. Маршрут перевірки стану використовується для того, щоб переконатися, що backend запущений і готовий приймати запити. Маршрут інформації про модель дозволяє отримати базові відомості про активну конфігурацію або checkpoint. Основним є маршрут аналізу, через який користувацький файл передається до системи детекції.

Після отримання файлу backend перевіряє його розширення та визначає, чи належить файл до підтримуваних аудіо або відеоформатів. Для аудіофайлів сигнал може бути безпосередньо переданий до модуля попередньої обробки. Для відеофайлів спочатку необхідно виділити аудіодоріжку, оскільки розроблена система аналізує саме мовний аудіосигнал, а не візуальний ряд. Після цього подальша обробка виконується так само, як і для звичайного аудіофайлу.

Тимчасове збереження файлу використовується для того, щоб backend міг передати його до модуля інференсу. Після завершення аналізу файл видаляється, оскільки застосунок не призначений для накопичення користувацьких медіаданих. Такий підхід є важливим з практичного погляду – сервер виконує лише поточну обробку запиту й не зберігає завантажені файли довше, ніж це потрібно для отримання результату.

Запуск моделі в серверній частині організований за принципом lazy loading. Це означає, що модель і checkpoint не обов’язково завантажуються в пам’ять одразу під час старту сервера, а можуть ініціалізуватися при першому запиті або під час звернення до відповідного модуля. Такий підхід дозволяє не виконувати зайве завантаження, якщо сервер запущено, але аналіз файлів ще не проводиться. Після завантаження модель може повторно використовуватися для наступних запитів.

Перед передаванням у модель файл проходить той самий базовий pipeline аудіообробки, що й під час навчання та оцінювання: аудіосигнал зчитується, приводиться до потрібної частоти дискретизації, перетворюється в моно за потреби, нормалізується та передається до моделі у вигляді тензора. Це важливо, оскільки модель навчалася на даних, підготовлених у певному форматі. Якщо під час інференсу використовувати іншу схему підготовки сигналу, результат може бути менш стабільним.

Результатом роботи моделі є числова оцінка класу «spoof». Це значення інтерпретується як оцінка ймовірності того, що аудіозапис має синтетичне або підроблене походження. Якщо оцінка перевищує заданий поріг, застосунок повертає мітку «spoof»; якщо оцінка нижча за поріг, повертається мітка «bonafide».

Окрім мітки класу, backend повертає значення впевненості. Для запису, класифікованого як «spoof», впевненість може відповідати самій оцінці (рисунок 3.8).

Для запису, класифікованого як «bonafide», впевненість може обчислюватися як 1 мінус значення оцінки.

Таким чином, користувач бачить не лише бінарний результат, а й узагальнену оцінку впевненості моделі у прийнятому рішенні. Водночас це значення не слід трактувати як абсолютну гарантію правильності, оскільки воно залежить від навчальних даних, порога класифікації та властивостей конкретного аудіофайлу.

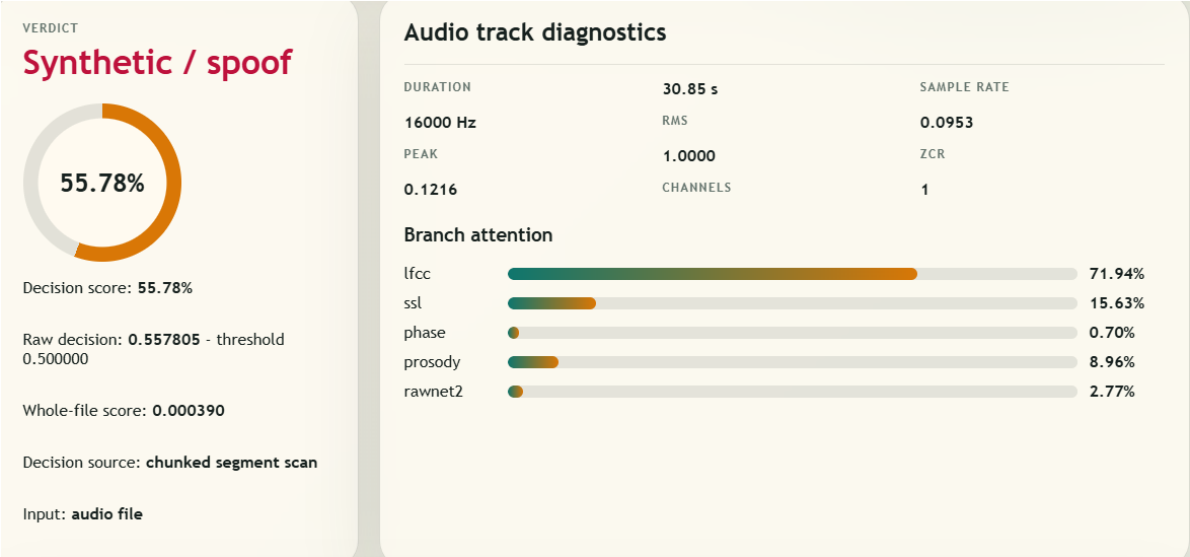


Рисунок 3.8 – Приклад користувацького інтерфейсу з показником впевненості

У відповіді backend також можуть передаватися додаткові діагностичні дані (рисунок 3.9). До них можуть належати attention weights для активних гілок, базові акустичні характеристики або службова інформація про оброблений файл.

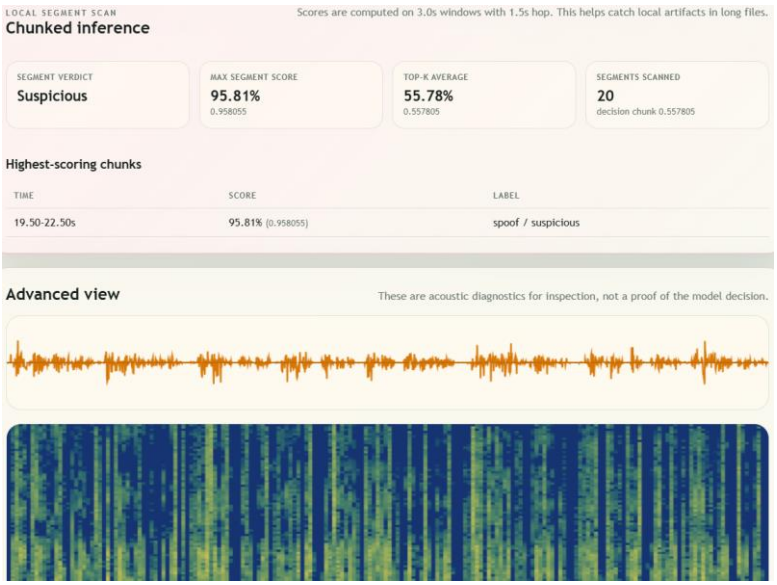


Рисунок 3.9 – Приклад користувацького інтерфейсу з додатковими діагностичними даними

Такі дані не є повною інтерпретацією рішення моделі, але можуть допомогти краще зрозуміти, як система опрацювала конкретний запис. Наприклад, ваги гілок можуть показати, які типи представлень отримали більший внесок у fusion-модулі для певного аудіо.

Клієнтська частина реалізована як frontend-застосунок, що забезпечує взаємодію користувача із системою. Її основна функція полягає в тому, щоб надати форму для завантаження файлу, передати файл на backend, дочекатися відповіді та відобразити результат у зручному вигляді. Користувачу не потрібно знати, які саме Python-модулі запускаються, де розташований checkpoint або як формується тензор для моделі. Усі ці операції приховані за інтерфейсом застосунку.

4 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

Після розроблення архітектури системи, реалізації модулів виділення ознак, навчального pipeline та засобів оцінювання необхідно перевірити якість роботи моделі на практиці. Експериментальна частина роботи спрямована на те, щоб оцінити здатність розробленої системи відрізняти справжнє мовлення від синтезованого, порівняти різні конфігурації моделі та визначити, як зміна набору ознак впливає на результат класифікації.

У цьому розділі розглядаються використані набори даних, конфігурації експериментів, метрики оцінювання та отримані результати. Основна увага приділяється не лише фінальному значенню метрик, а й порівнянню різних варіантів моделі: базової конфігурації, розширених гібридних варіантів, моделей із додатковими гілками та сценаріїв fine-tuning. Такий підхід дозволяє оцінити не тільки загальну ефективність системи, а й практичний внесок окремих компонентів.

4.1 Використані набори даних

Для навчання та оцінювання системи використовувалися набори даних, орієнтовані на задачу виявлення синтезованого або підробленого мовлення. Основними джерелами даних є набори серії ASVspoof, які застосовуються для дослідження системи протидії підробці мовлення у сфері автоматичної верифікації мовця та виявлення spoofing-атак. У межах цієї роботи вони використовуються не для самої верифікації мовця, а для навчання та перевірки моделі, яка класифікує аудіозаписи як «bonafide» або «spoof».

Основним набором для побудови та первинної перевірки моделі є ASVspoof 2019. Цей набір містить дані для дослідження атак на системи автоматичної верифікації мовця, зокрема синтезоване, конвертоване та

replay-мовлення [1]. У межах цієї роботи основний акцент зроблено на частині Logical Access, оскільки вона пов’язана з виявленням синтезованого та конвертованого мовлення. Саме ця частина є найбільш релевантною для задачі детекції штучно згенерованого аудіосигналу.

ASVspoof 2019 LA містить записи двох класів: «bonafide» та «spoof». Клас «bonafide» відповідає природному мовленню, а клас «spoof» відповідає аудіозаписам, отриманим за допомогою систем синтезу або перетворення голосу. Для моделі ці класи подаються у вигляді бінарних міток: 0 для bonafide та 1 для spoof. Таке кодування відповідає реалізованій постановці задачі, у якій модель повинна сформувати score класу spoof для кожного аудіозапису.

Набір ASVspoof 2019 використовується як базовий набір для навчання та оцінювання моделі. Його перевага полягає в тому, що він має чітку структуру, файли протоколів та поділ на частини для навчання, валідації та оцінювання. Це дозволяє побудувати відтворюваний експериментальний процес: модель навчається на train-частині, підбирається або контролюється на dev-частині, а фінальна якість перевіряється на eval-частині.

У контексті цієї роботи ASVspoof 2019 LA виконує дві функції.

По-перше, він використовується для навчання початкових конфігурацій моделі, зокрема варіантів на основі LFCC, WavLM та їх поєднання.

По-друге, він слугує основою для порівняння різних архітектурних варіантів у межах одного домену. Це важливо, оскільки результати моделей можна коректно порівнювати лише тоді, коли вони оцінюються на однаковому наборі даних і за однаковою процедурою.

Додатковим набором, який розглядається в роботі, є ASVspoof 2021. Цей набір був спрямований на подальше прискорення розвитку методів виявлення підробленого і клонованого голосу. У проєкті передбачено окремий скрипт для перевірки навченої моделі на ASVspoof 2021 LA, що дозволяє використовувати цей набір як додатковий сценарій зовнішнього

оцінювання. На відміну від перевірки всередині ASVspoof 2019, така оцінка дає змогу побачити, як модель поводиться за зміни умов і розподілу даних.

ASVspoof 2021 у цій роботі доцільно розглядати не як основний навчальний набір, а як додаткову перевірку адаптивності. Причиною можуть бути відмінності в умовах запису, типах атак, способах генерації синтезованого мовлення або в структурі eval-частини.

Окрему роль в експериментальній частині відіграє ASVspoof5. Цей набір пов’язаний з оцінюванням виявлення підробки мовного аудіо, deepfake-аудіо та змагальних атак із використанням мовленнєвих даних, зібраних від багатьох користувачів. [5]. Його використання дозволяє перевірити модель у складнішому та більш сучасному сценарії, де аудіозаписи можуть відрізнятися від даних ASVspoof 2019 за способом отримання, різноманіттям атак і загальними властивостями мовного матеріалу.

У межах роботи ASVspoof5 також використовується для донавчання. Це дозволяє перевірити, чи може модель, попередньо навчена на одному наборі, бути адаптована до іншого набору даних за рахунок додаткового навчання. Такий сценарій є практично важливим, оскільки в реальних умовах система детекції може стикатися з новими типами синтезованого мовлення, яких не було в початковому навчальному наборі.

Важливо враховувати, що різні набори даних у цій роботі виконують різні експериментальні ролі. ASVspoof 2019 LA використовується як основна база для навчання та базового порівняння моделей. ASVspoof 2021 застосовується як додаткова перевірка узагальнювальної здатності. ASVspoof5 використовується для складнішого зовнішнього оцінювання та донавчання.

Такий поділ дозволяє не змішувати результати різних експериментальних сценаріїв і коректно інтерпретувати отримані метрики.

4.2 Конфігурація експериментів

Після визначення наборів даних було сформовано кілька експериментальних конфігурацій моделі. Їхнє призначення полягало в тому, щоб не лише отримати фінальну якість класифікації, а й перевірити внесок окремих груп ознак у роботу системи. Для цього використовувалися як базові конфігурації з обмеженим набором гілок, так і розширені гібридні варіанти, у яких поєднувалися спектральні, SSL та додаткові акустичні представлення.

Усі експерименти виконувалися в межах єдиного програмного конвеєру, описаного у другому розділі. Це означає, що для різних варіантів моделі використовувалися однакова логіка підготовки даних, єдиний формат метаданих, однаковий механізм групування, спільний підхід до навчання, зберігання контрольних точок та оцінювання. Відмінність між експериментами полягала передусім у наборі активних гілок, параметрах конфігураційного файлу та використаному checkpoint.

Основою для запуску експериментів були YAML-конфігурації. Через них задавалися активні гілки моделі, параметри аудіообробки, вибір SSL моделі, режим заморожування або часткового донавчання, параметри навчання, шляхи до наборів даних і директорії для збереження результатів.

У межах роботи використовувалися кілька груп експериментів. Перша група пов’язана з базовими моделями, які використовують одну або дві основні гілки ознак. Друга група охоплює повніші гібридні конфігурації з додаванням phase, prosody, wav2vec2 або RawNet2-подібної гілки. Третя група стосується перенесення моделі на інший набір даних і донавчання, зокрема на ASVspoof5. Такий поділ дозволяє окремо розглядати базову ефективність моделі, вплив архітектурних розширень і здатність системи адаптуватися до нового домену.

Першим базовим варіантом є модель, що використовує лише LFCC ознаки. Вона потрібна як простий спектральний baseline, з яким можна

порівнювати складніші конфігурації. Такий експеримент дозволяє оцінити, наскільки добре модель працює на основі класичного акустичного представлення без використання попередньо навчених SSL-моделей. Результати цієї конфігурації є важливими для розуміння того, який рівень якості досягається лише за рахунок спектральних характеристик.

Другим базовим варіантом є LFCC з SSL конфігурація. У ній основну роль відіграє попередньо навчена модель мовлення, зокрема WavLM-base. Використання WavLM пов’язане з тим, що ця модель формує самокеровані представлення мовлення, які можуть бути корисними для широкого спектра мовленнєвих задач [3]. У межах цієї роботи SSL-гілка розглядається як джерело високорівневих ознак аудіосигналу, які доповнюють або замінюють ручні акустичні характеристики.

Наступна група експериментів пов’язана з додаванням фазової та просодичної гілок. У таких конфігураціях до LFCC і SSL-представлень додаються ознаки, що описують інші властивості сигналу. Фазова гілка враховує зміну фазової структури між часовими фреймами, а просодична гілка додає компактні енергетичні та часові характеристики. Ці експерименти потрібні для перевірки того, чи покращує додаткова акустична інформація якість класифікації, порівняно з базовою «LFCC та WavLM» моделлю.

Окремо розглядалася конфігурація з використанням двох SSL моделей – WavLM і wav2vec2. Модель wav2vec 2.0 також належить до підходів самокерованого навчання мовленнєвих представлень [11]. У межах експериментів wav2vec2 використовується як додаткове джерело SSL ознак, що дозволяє перевірити, чи дає поєднання двох різних попередньо навчених моделей кращий результат, ніж використання лише однієї SSL гілки.

Ще одним розширеним варіантом є модель із RawNet2-подібною гілкою. RawNet2 було запропоновано як end-to-end підхід до anti-spoofing, що працює безпосередньо з хвильовою формою аудіосигналу [9]. У цій роботі використовується не повна копія оригінальної архітектури, а

компактна RawNet2-inspired гілка, яка доповнює інші представлення raw waveform-ознаками. Така конфігурація дозволяє перевірити, чи є корисною інформація, отримана безпосередньо з хвильової форми, у поєднанні зі спектральними та SSL-ознаками.

Для базових експериментів SSL-модель використовувалася переважно в замороженому режимі. Це означає, що параметри WavLM або wav2vec2 не оновлювалися під час навчання, а модель застосовувалася як модуль виділення ознак. Такий режим дозволяє зменшити обчислювальні витрати та зосередити навчання на проєкційних шарах, модулі ознак й класифікаторі. Крім того, заморожування SSL-гілки зменшує ризик перенавчання, особливо коли обсяг цільових даних обмежений.

Для окремих експериментів передбачалося часткове донавчання SSL гілки. У такому режимі розморожуються лише останні шари SSL-моделі, тоді як інші залишаються незмінними. Це дозволяє адаптувати модель до нового домену, але не руйнувати повністю представлення, сформовані під час попереднього навчання. Такий підхід особливо важливий у fine-tuning сценаріях, де модель переноситься з одного набору даних на інший.

Окремий блок експериментів пов’язаний із перевіркою моделі на ASVspoof5. Тут розглядалися два сценарії. У першому сценарії модель, навчена на попередніх даних, оцінювалася на ASVspoof5 без додаткової адаптації. Це дозволяє побачити, наскільки добре вона переноситься на інший домен. У другому сценарії виконувалося донавчання на ASVspoof5, після чого модель знову оцінювалася на відповідній dev-частині. Порівняння цих двох сценаріїв дозволяє оцінити практичну користь адаптації.

Для донавчання використовувався checkpoint попередньо навченої моделі. Це означає, що навчання не починалося з випадкової ініціалізації, а стартувало з уже сформованих ваг. Такий підхід є доцільним, оскільки базова модель уже навчилася виділяти корисні ознаки для відокремлення bonafide та spoof-записів. Донавчання у цьому випадку виконує роль

доменної адаптації, модель уточнює свої параметри відповідно до властивостей нового набору даних.

Під час усіх експериментів застосовувався однаковий набір основних метрик: AUC ROC, EER, precision, recall, F1-score та confusion matrix. AUC ROC і EER використовувалися як основні метрики для оцінювання роздільної здатності моделі та якості роботи в anti-spoofing-сценарії. Precision, recall і F1-score додатково використовувалися для аналізу поведінки моделі при фіксованому порозі класифікації.

Важливо, що результати різних наборів даних не змішувалися в одній інтерпретації. Результати на ASVspoof 2019 LA характеризують якість моделі в межах основного сценарію. Результати на ASVspoof5 показують здатність моделі працювати з іншим розподілом даних і сучаснішими типами атак.

Для кожної конфігурації після навчання зберігався checkpoint, а під час оцінювання формувалися score-файли та JSON-файли з метриками. Це дозволяє повторно аналізувати результати без нового запуску моделі та порівнювати конфігурації між собою. У подальших підрозділах ці файли використовуються як основа для таблиць результатів, аналізу дослідження та оцінювання впливу fine-tuning.

4.3 Результати на ASVspoof 2019 LA

Перший основний блок експериментів було виконано на наборі ASVspoof 2019 LA. Цей набір використовувався як базовий benchmark для перевірки здатності моделі відрізняти природне мовлення від синтезованого або конвертованого мовлення [2]. Оцінювання проводилося на eval-частині набору, що дозволяє перевірити якість моделі на даних, які не використовувалися під час навчання.

У межах цього етапу було перевірено декілька конфігурацій моделі. Основну увагу приділено двом варіантам – розширеній гібридній моделі та

конфігурації з двома SSL-гілками й RawNet2-подібним компонентом. Перший варіант використовує поєднання LFCC, WavLM, фазових та просодичних ознак. Другий варіант додатково залучає wav2vec2 представлення та RawNet2-inspired гілку, що дозволяє моделі працювати не лише зі спектральними й SSL-ознаками, а й з представленнями, отриманими безпосередньо з хвильової форми.

Основні результати оцінювання наведено в таблиці 4.1.

Таблиця 4.1 – Результати оцінювання моделей на ASVspoof 2019 LA eval

Конфігурація моделі	AUC ROC	EER, %	Precision	Recall	F1-score
LFCC + Phase + Prosody	0,9393	11,98	0,9965	0,7174	0,8342
LFCC + WavLM + Phase + Prosody	0,9963	2,90	0,9992	0,9203	0,9581
LFCC + Wav2Vec2 + Phase + Prosody	0,9994	1,07	0,9997	0,9597	0,9793
LFCC + WavLM + Phase + Prosody + RawNet2-inspired	0,9979	1,74	0,9997	0,9612	0,9696

З таблиці видно, що найнижчі результати серед наведених конфігурацій показала модель «LFCC + Phase + Prosody», у якій не використовуються SSL-представлення. Для цієї конфігурації AUC ROC становить 0,9393, а EER дорівнює 11,98 %. Попри високе значення precision 0,9965, recall становить лише 0,7174, що свідчить про значну кількість пропущених spoof-записів при фіксованому порозі класифікації 0,5. Це означає, що акустичні ознаки без SSL-гілки здатні частково розділяти класи, але не забезпечують достатньо повного опису синтезованого мовлення.

Додавання WavLM-представлень суттєво покращило якість класифікації. Конфігурація «LFCC + WavLM + Phase + Prosody» досягла AUC ROC 0,9963 та EER 2,90 %. Також помітно зріс recall – до 0,9203, а F1-score становив 0,9581. Це підтверджує, що SSL-представлення дають

моделі додаткову інформацію про мовний сигнал, яка не повністю відображається через спектральні, фазові та енергетично-часові характеристики.

Окремо було перевірено конфігурацію з wav2vec2 як альтернативною SSL-гілкою. Вона показала найкращі результати саме на ASVspoof 2019 LA eval: AUC ROC 0,9994, EER 1,07 %, precision 0,9997, recall 0,9597 та F1-score 0,9793. Це свідчить про високу здатність цієї моделі розділяти bonafide та spoof-записи в межах даного тестового набору. Водночас ці результати не розглядалися ізольовано, оскільки для вибору фінальної конфігурації важливими були не лише метрики на ASVspoof 2019 LA, а й подальша здатність моделі до узагальнення на інших наборах даних.

Фінальна WavLM-орієнтована конфігурація «LFCC + WavLM + Phase + Prosody + RawNet2-inspired» показала AUC ROC 0,9979 та EER 1,74 %. Порівняно з варіантом без RawNet2-inspired гілки, EER зменшився з 2,90 % до 1,74 %, а recall зріс з 0,9203 до 0,9412. Це показує, що додавання raw waveform-гілки дозволило покращити виявлення spoof-записів і підвищити загальну якість класифікації без переходу до складнішої Dual SSL архітектури.

Окремо слід враховувати різницю між метриками, які оцінюють ранжування, і метриками, що залежать від конкретного порога. AUC ROC та EER характеризують загальну здатність score відокремлювати два класи при зміні порога. Натомість precision, recall і F1-score у цьому експерименті наведені для фіксованого порога 0,5. Тому модель може мати високе значення AUC ROC, але при неоптимальному порозі все ще пропускати частину синтезованих записів. Це особливо помітно для конфігурації «LFCC + Phase + Prosody», де precision є високим, але recall значно нижчий.

Загалом результати на ASVspoof 2019 LA eval підтверджують доцільність гібридного підходу. Перехід від акустичної конфігурації без SSL до моделей із WavLM або wav2vec2 значно покращує якість детекції.

Додавання RawNet2-inspired гілки до WavLM-орієнтованої моделі додатково підвищує recall та F1-score, що робить цю конфігурацію більш придатною для подальших експериментів із перевірки узагальнення та domain adaptation.

4.4 Дослідження на ASVspoof 2021 LA

Для додаткової перевірки здатності моделі до узагальнення було проведено оцінювання на наборі ASVspoof 2021 LA. На відміну від ASVspoof 2019 LA eval, цей набір використовується як складніший тестовий домен, оскільки містить інші умови формування та обробки аудіозаписів. Тому результати на ASVspoof 2021 LA дозволяють оцінити не лише якість класифікації в межах одного benchmark-набору, а й стійкість системи до зміни домену наведено у таблиці 4.2.

Таблиця 4.2 – Результати оцінювання моделей на ASVspoof 2021 LA eval

Конфігурація моделі	AUC ROC	EER, %	Accuracy	Коментар
LFCC + Phase + Prosody	0,7809	29,10	0,8034	Акустична конфігурація без SSL-гілки; слабке узагальнення
LFCC + WavLM + Phase + Prosody	0,9764	8,48	0,9472	Гібридна конфігурація з WavLM; кращий EER порівняно з wav2vec2
LFCC + Wav2Vec2 + Phase + Prosody	0,9742	8,83	0,9657	Альтернативна SSL-конфігурація; висока accuracy, але гірший EER
LFCC + WavLM + Phase + Prosody + RawNet2-inspired	0,9842	6,53	0,9639	Фінальна WavLM-орієнтована конфігурація з raw waveform-гілкою

У цьому експерименті особливу увагу було приділено порівнянню WavLM та wav2vec2 як SSL-гілок. На ASVspoof 2019 LA конфігурація з wav2vec2 демонструвала дуже високі метрики, однак для вибору фінальної моделі важливо враховувати не лише результат на одному тестовому наборі, а й поведінку системи при перенесенні на нові дані. Саме тому ASVspoof 2021 LA розглядався як важливий етап перевірки узагальнювальної здатності моделі.

З таблиці видно, що акустична конфігурація без SSL-гілки значно поступається гібридним моделям. Для «LFCC + Phase + Prosody» значення AUC ROC становить лише 0,7809, а EER досягає 29,10 %. Це підтверджує, що спектральні, фазові та енергетично-часові ознаки самі по собі не забезпечують достатньої стійкості при переході до складнішого тестового домену. Такий результат є важливим для обґрунтування використання SSL-представлень у складі системи.

Додавання WavLM суттєво покращило якість оцінювання на ASVspoof 2021 LA. Конфігурація «LFCC + WavLM + Phase + Prosody» досягла AUC ROC 0,9764 та EER 8,48 %. Це свідчить про те, що WavLM-представлення допомагають моделі краще зберігати здатність до розділення bonafide та spoof-записів навіть за зміни тестового домену. Порівняно з акустичною конфігурацією без SSL, зниження EER є суттєвим, що підтверджує практичну цінність SSL-гілки.

Конфігурація з wav2vec2 також показала високий результат: AUC ROC становить 0,9742, EER – 8,83 %, а accuracy при порозі 0,5 дорівнює 0,9657. Через таке співвідношення класів accuracy потрібно трактувати обережно, оскільки вона може бути менш показовою, ніж AUC ROC та EER. Важливо також, що EER-threshold для цієї конфігурації становив 0,9866, тобто оптимальний поріг суттєво відрізнявся від стандартного значення 0,5. Це свідчить про певну зміну розподілу score при перенесенні моделі на ASVspoof 2021 LA.

Попри високу accuracy конфігурації з wav2vec2, за більш релевантними для anti-spoofing задач метриками WavLM показав краще узагальнення: EER для WavLM-конфігурації становив 8,48 %, тоді як для wav2vec2 – 8,83 %. Різниця не є радикальною, однак вона важлива з погляду вибору фінальної архітектури, оскільки EER безпосередньо відображає баланс між хибним прийняттям і хибним відхиленням. Саме тому WavLM було обрано як основну SSL-гілку системи.

Найкращий результат серед WavLM-орієнтованих конфігурацій показала модель «LFCC + WavLM + Phase + Prosody + RawNet2». Для неї AUC ROC становить 0,9842, EER знижується до 6,53 %, а accuracy становить 0,9639. Це означає, що додавання raw waveform-гілки покращило здатність системи до узагальнення без переходу до Dual SSL-архітектури. Така конфігурація зберігає основну логіку роботи – поєднання WavLM представлень з акустичними ознаками – і водночас розширює аналіз аудіосигналу за рахунок безпосередньої роботи з хвильовою формою.

Результати на ASVspoof 2021 LA показують, що вибір фінальної моделі не може базуватися лише на максимальних метриках на ASVspoof 2019 LA. Конфігурація з wav2vec2 була дуже сильною на ASVspoof 2019 LA, але при перенесенні на ASVspoof 2021 LA WavLM продемонстрував кращий баланс за EER. Це підтверджує, що WavLM є більш доцільною основою для фінальної системи, якщо враховувати не лише якість на одному наборі даних, а й здатність моделі працювати в умовах зміни домену.

Таким чином, дослідження на ASVspoof 2021 LA підтвердило ефективність обраної WavLM-орієнтованої гібридної архітектури. Система не лише показала високі метрики на ASVspoof 2019 LA, а й зберегла стійкість при оцінюванні на складнішому cross-year наборі. Саме це стало підставою для вибору конфігурації «LFCC + WavLM + Phase + Prosody + RawNet2» як основної фінальної моделі для подальших експериментів з domain adaptation.

4.5 Перевірка перенесення моделі на ASVspoof5

Після оцінювання системи на ASVspoof 2019 LA та ASVspoof 2021 LA було проведено додаткову перевірку її роботи на новішому наборі ASVspoof5. Цей набір орієнтований на оцінювання методів виявлення spoofing, deepfake та adversarial attack detection із використанням crowdsourced speech [5]. Такий етап є важливим, оскільки високі результати на одному benchmark-наборі не гарантують збереження тієї самої якості на даних з іншим розподілом, іншими типами атак та іншими умовами формування аудіозаписів.

На відміну від оцінювання всередині ASVspoof 2019 LA, перевірка на ASVspoof5 є сценарієм перенесення моделі на інший домен. У такому випадку змінюються не лише конкретні аудіофайли, а й загальні властивості даних: типи синтезованого мовлення, характеристики дикторів, умови запису, способи формування атак, формат аудіоданих і співвідношення класів. Тому результат на ASVspoof5 слід розглядати не як пряме продовження експериментів на ASVspoof 2019 LA, а як окрему перевірку узагальнювальної здатності системи.

Для початкової перевірки використовувалася модель без додаткової адаптації до ASVspoof5. Тобто checkpoint, отриманий у попередньому експериментальному сценарії, оцінювався на dev-частині ASVspoof5 без fine-tuning на даних цього набору. Метою такого експерименту було визначити, наскільки добре система зберігає здатність відокремлювати bonafide та spoof-записи при переході до нового домену. Якщо якість у такому режимі знижується, це не свідчить про неспроможність архітектури, а вказує на наявність domain shift між наборами даних.

Результати оцінювання на ASVspoof5 dev без додаткової адаптації наведено в таблиці 4.3.

Таблиця 4.3 – Результати оцінювання на ASVspoof5 dev без донавчання

Метрика	Значення
AUC ROC	0,9332
EER, %	15,12
Accuracy при threshold = 0,5	0,8575
Precision	0,8454
Recall	0,9996
F1-score	0,9160
Кількість записів	140950
Кількість bonafide-записів	31334
Кількість spoof-записів	109616

Отримані результати показують, що навіть без додаткового навчання на ASVspoof5 система зберігає корисну роздільну здатність – AUC ROC становить 0,9332. Це означає, що score моделі все ще містить інформативний сигнал для відокремлення spoof записів від bonafide записів. Водночас EER зростає до 15,12 %, що помітно вище, ніж у межах ASVspoof 2019 LA та ASVspoof 2021 LA. Така різниця є очікуваною для сценарію перенесення на інший домен і демонструє, що ASVspoof5 є значно складнішим тестом узагальнення.

Окремо слід звернути увагу на співвідношення precision і recall. Значення recall становить 0,9996, тобто при використанні порога 0,5 система майже не пропускає spoof-записи. Водночас precision дорівнює 0,8454, що вказує на наявність помилкових спрацьовувань на bonafide-записах.

Іншими словами, у цьому режимі модель поводиться обережно з погляду безпеки, вона майже всі підроблені записи відносить до класу spoof, але частину справжніх аудіозаписів також може помилково позначати як spoof.

Така поведінка може бути пов’язана зі зміною score-розподілів між наборами даних. Модель, навчена на одному домені, може мати добре налаштований вихідний score для ASVspoof 2019 LA або ASVspoof 2021 LA, але при переході до ASVspoof5 цей розподіл зміщується.

Важливо, що результат на ASVspoof5 без адаптації не слід трактувати як недолік розробленої системи. Навпаки, цей експеримент показує реалістичну ситуацію, з якою стикаються моделі детекції синтезованого мовлення, при появі нових генераторів, нових мовців, інших каналів запису або інших способів обробки аудіо якість може змінюватися. Саме тому перевірка на зовнішньому наборі є важливою частиною експериментального дослідження, оскільки вона дозволяє оцінити не лише якість класифікації, а й стійкість системи до зміни домену.

Отриманий результат також підтверджує, що для практичного використання на нових типах даних важливим є не тільки вибір архітектури, а й адаптація до цільового домену. Високий recall у цьому експерименті є корисним для сценаріїв, де критично мінімізувати пропуск spoof-записів.

Однак зниження precision може бути небажаним у користувацьких сервісах, де помилкове позначення справжнього мовлення як підробленого знижує довіру до системи. Тому для нового домену доцільно або підбирати поріг класифікації, або виконувати додаткове навчання на репрезентативних даних.

Таким чином, перевірка на ASVspoof5 dev без fine-tuning показала, що система зберігає здатність розділяти класи на новому наборі, але якість без адаптації є нижчою через domain shift. Цей результат логічно підводить до наступного етапу експериментів – fine-tuning WavLM-орієнтованої фінальної конфігурації на даних ASVspoof5.

Такий етап дозволяє перевірити, чи може додаткове навчання зменшити вплив зміщення домену та підвищити якість роботи системи на новішому й складнішому наборі даних.

4.6 Fine-tuning моделі на ASVspoof5

Результати перенесення системи на ASVspoof5 без додаткової адаптації показали наявність помітного domain shift між попередніми наборами даних та новішим доменом ASVspoof5. Система зберігала здатність розділяти bonafide та spoof-записи, однак значення EER суттєво зростало. Це означає, що ознаки, сформовані під час попереднього навчання, залишалися інформативними, але потребували уточнення відповідно до нового розподілу аудіоданих. Для перевірки можливості такої адаптації було проведено fine-tuning фінальної WavLM-орієнтованої конфігурації на даних ASVspoof5.

У цій роботі fine-tuning розглядається як етап доменної адаптації. На відміну від навчання з випадкової ініціалізації, цей підхід використовує вже навчену модель як початкову точку. Це є доцільним, оскільки система вже навчилася виділяти ознаки, пов’язані з відмінностями між справжнім і синтезованим мовленням.

Додаткове навчання на ASVspoof5 дозволяє не перебудовувати модель повністю, а адаптувати її score-розподіли та класифікаційні межі до властивостей нового домену.

Для fine-tuning було використано фінальну конфігурацію «LFCC + WavLM + Phase + Prosody + RawNet2». Вона поєднує спектральні, фазові, енергетично-часові, SSL та raw waveform-представлення аудіосигналу. Така структура є придатною для адаптації, оскільки модель аналізує сигнал на кількох рівнях, через акустичні характеристики, високорівневі WavLM-представлення та безпосередню хвильову форму. Це дозволяє системі краще враховувати особливості нового набору даних без переходу до надмірно ускладненої Dual SSL-архітектури.

Порівняння результатів до та після fine-tuning наведено в таблиці 4.4.

Таблиця 4.4 – Порівняння результатів на ASVspoof5 dev до та після fine-tuning

Метрика	До fine-tuning	Після fine-tuning
AUC ROC	0,9332	0,9956
EER, %	15,12	3,15
Accuracy при threshold = 0,5	0,8575	0,9465
Precision	0,8454	0,9951
Recall	0,9996	0,9358
F1-score	0,9160	0,9646
Кількість записів	140950	140950
Кількість bonafide-записів	31334	31334
Кількість spoof-записів	109616	109616

З таблиці видно, що fine-tuning суттєво покращив якість роботи системи на ASVspoof5 dev. Значення AUC ROC зросло з 0,9332 до 0,9956, що свідчить про значно кращу здатність моделі ранжувати аудіозаписи за ймовірністю належності до spoof-класу. Найбільш показовим є зменшення EER з 15,12 % до 3,15 %. Це означає, що після адаптації система набагато краще балансує між хибним прийняттям spoof-записів як bonafide та хибним відхиленням справжнього мовлення.

До fine-tuning модель мала дуже високий recall – 0,9996. Це означало, що при порозі 0,5 вона майже не пропускала spoof-записи. Водночас precision становив 0,8454, тобто частина bonafide-записів помилково визначалася як spoof. Така поведінка є характерною для перенесення моделі на новий домен без адаптації, система зберігає високу чутливість до підробленого мовлення, але score-розподіл для справжніх записів зміщується, що призводить до більшої кількості помилкових спрацьовувань.

Після fine-tuning precision зріс до 0,9951, що свідчить про значне зменшення кількості помилкових спрацьовувань на bonafide-класі. При

цьому recall становив 0,9358. Такий результат показує, що модель стала більш збалансованою – вона набагато точніше відокремлює справжнє мовлення від підробленого, хоча при фіксованому порозі 0,5 частина spoof записів усе ще може бути пропущена. Саме тому для повної оцінки якості важливо враховувати не лише precision і recall, а й AUC ROC та EER, які не прив’язані до одного фіксованого порога.

Отримані результати підтверджують, що зниження якості без fine-tuning було пов’язане не з обмеженням архітектури, а з відмінністю між доменами. Після додаткового навчання на ASVspoof5 система змогла ефективно адаптуватися до нового розподілу аудіоданих – EER зменшився майже у п’ять разів, AUC ROC наблизився до 1, а F1-score зріс з 0,9160 до 0,9646. Це демонструє, що запропонована WavLM-орієнтована гібридна модель має не лише високу якість на вихідних наборах даних, а й добру здатність до доменної адаптації.

Окремо слід зазначити, що fine-tuning не змінює загальну ідею системи, а лише уточнює її параметри для нового набору даних. Основна архітектура залишається сталою, модель продовжує використовувати поєднання LFCC, WavLM, фазових, енергетично-часових та RawNet2 ознак. Саме така модульна структура дозволяє ефективно переносити систему на нові домени та адаптувати її до інших типів синтезованого мовлення.

Таким чином, fine-tuning на ASVspoof5 підтвердив практичну цінність запропонованої системи. Без адаптації модель уже зберігала корисну роздільну здатність, однак після додаткового навчання її якість істотно зросла. Це показує, що розроблена система не обмежується роботою в межах одного benchmark-набору, а може бути адаптована до складніших і новіших сценаріїв детекції синтезованого мовлення.

ВИСНОВКИ

У кваліфікаційній роботі було розглянуто задачу автоматичної детекції синтезованого мовлення та розроблено гібридну систему, яка поєднує спектральні, SSL, фазові, енергетично-часові та raw waveform представлення аудіосигналу. Актуальність цієї задачі зумовлена швидким розвитком сучасних систем синтезу мовлення, deepfake-аудіо та voice conversion, які здатні створювати записи, близькі за звучанням до природного людського голосу. У таких умовах виникає потреба в автоматизованих методах аналізу, здатних виявляти ознаки синтетичного походження аудіосигналу.

У першому розділі було проаналізовано предметну область детекції синтезованого мовлення, основні типи загроз, пов’язаних із використанням штучно згенерованого аудіо, а також існуючі підходи до побудови anti-spoofing-систем. Було розглянуто методи на основі кепстральних ознак, raw waveform-підходи, SSL-моделі та гібридні архітектури. На основі цього аналізу було обґрунтовано доцільність використання гібридного підходу, у якому різні типи ознак доповнюють одне одного та дозволяють аналізувати мовний сигнал на кількох рівнях.

У другому розділі було описано розроблену систему детекції синтезованого мовлення. Її архітектура побудована за модульним принципом і містить блок попередньої обробки аудіо, паралельні гілки виділення ознак, проєкційні шари, механізм gated attention fusion та класифікаційний модуль. Основною фінальною конфігурацією системи є WavLM-орієнтована гібридна модель, яка поєднує LFCC, WavLM, фазові, енергетично-часові та RawNet2-inspired ознаки. Така структура дозволяє враховувати як низькорівневі акустичні характеристики сигналу, так і високорівневі мовленнєві представлення.

У межах програмної реалізації було створено Python-проєкт із чітким поділом на модулі для роботи з даними, ознаками, моделлю,

конфігураціями, навчанням і оцінюванням. Реалізація підтримує конфігурований запуск експериментів, зміну набору активних гілок, навчання моделі, fine-tuning, збереження checkpoint, розрахунок метрик і формування score-файлів. Додатково було реалізовано локальний серверно-клієнтський застосунок із FastAPI backend і frontend-інтерфейсом, який дозволяє завантажити аудіо- або відеофайл і отримати результат класифікації.

У третьому розділі було проведено експериментальне дослідження системи. На ASVspoof 2019 LA було показано, що акустична конфігурація без SSL-гілки має обмежену якість узагальнення – для «LFCC + Phase + Prosody» було отримано AUC ROC 0,9393 та EER 11,98 %. Додавання SSL представлень суттєво покращило результати. Для конфігурації «LFCC + WavLM + Phase + Prosody» AUC ROC становив 0,9963, а EER – 2,90 %. Додавання RawNet2-inspired гілки до WavLM-орієнтованої моделі додатково покращило якість, для конфігурації «LFCC + WavLM + Phase + Prosody + RawNet2» було отримано AUC ROC 0,9979 та EER 1,74 %.

Окремо було проведено порівняння WavLM та wav2vec2 як SSL-гілок. Конфігурація з wav2vec2 показала дуже високі результати на ASVspoof 2019 LA, однак під час перевірки на ASVspoof 2021 LA WavLM продемонстрував кращий баланс за EER. Для «LFCC + WavLM + Phase + Prosody» на ASVspoof 2021 LA було отримано AUC ROC 0,9764 та EER 8,48 %, тоді як для «LFCC + Wav2Vec2 + Phase + Prosody» EER становив 8,83 %. Після додавання RawNet2-inspired гілки WavLM-орієнтована конфігурація досягла AUC ROC 0,9842 та EER 6,53 %. Саме тому для фінальної системи було обрано WavLM-орієнтовану архітектуру як більш збалансований варіант з погляду якості та здатності до перенесення на інший домен.

Також було перевірено здатність системи працювати на новішому наборі ASVspoof5. Без додаткової адаптації модель зберігала корисну роздільну здатність, однак якість знижувалася через domain shift: AUC ROC становив 0,9332, а EER – 15,12 %. Після fine-tuning фінальної WavLM

орієнтованої конфігурації на ASVspoof5 якість суттєво покращилася: AUC ROC зріс до 0,9956, EER зменшився до 3,15 %, precision становив 0,9951, recall – 0,9358, а F1-score – 0,9646. Це підтвердило, що розроблена система може бути ефективно адаптована до нового домену за рахунок додаткового навчання.

У процесі аналізу було встановлено, що якість детекції залежить не лише від архітектури моделі, а й від домену даних, складу навчальної вибірки, співвідношення класів, вибору threshold та режиму fine-tuning. Було показано, що різні конфігурації можуть мати різний баланс між precision і recall: одні краще мінімізують помилкові спрацьовування на bonafide-класі, інші – краще виявляють spoof-записи. Тому для практичного застосування важливо враховувати, який тип помилки є більш критичним у конкретному сценарії.

Розроблена система має низку переваг. Вона є модульною, підтримує кілька типів ознак, дозволяє проводити порівняльні експерименти, може бути адаптована до нових наборів даних і має демонстраційний інтерфейс для перевірки окремих аудіо- та відеофайлів. Водночас система має й обмеження. SSL-гілка потребує значних обчислювальних ресурсів, score моделі не слід трактувати як абсолютну гарантію синтетичного походження аудіо, а результати на одному benchmark-наборі не гарантують такої самої якості на іншому домені без додаткової адаптації або калібрування.

Перспективами подальшого розвитку є розширення навчальних даних за рахунок новіших типів синтезованого мовлення, проведення повнішого ablation-дослідження з окремою оцінкою внеску кожної гілки, покращення калібрування score, підбір threshold для різних сценаріїв використання, а також розширення просодичної гілки повнішими характеристиками F0, jitter, shimmer та voice quality.

Отже, у роботі було розроблено та експериментально перевірено гібридну систему детекції синтезованого мовлення, яка поєднує класичні акустичні ознаки, WavLM-представлення та raw waveform-компонент.

Запропонована архітектура дозволяє аналізувати аудіосигнал на кількох рівнях – через спектральні, фазові, енергетично-часові характеристики, високорівневі SSL-представлення та ознаки, отримані безпосередньо з хвильової форми.

Отримані результати підтверджують ефективність такого підходу для виявлення синтетичного мовлення та показують перевагу комбінування різнорідних представлень порівняно з використанням лише окремих груп ознак. Також було встановлено, що fine-tuning є важливим етапом для перенесення системи на новіші й складніші набори даних, оскільки дає змогу зменшити вплив domain shift і підвищити якість роботи моделі в новому домені. Це підтверджує практичну доцільність розробленої системи та можливість її подальшого розвитку для задач перевірки автентичності мовного аудіоконтенту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Kong J., Kim J., Bae J. HiFi-GAN: generative adversarial networks for efficient and high fidelity speech synthesis. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2010.05646 (date of access: 19.05.2026).

2. ASVspoof 2019: a large-scale public database of synthesized, converted and replayed speech / X. Wang et al. *computer speech & language*. 2020. Vol. 64. P. 101114. URL: https://doi.org/10.1016/j.csl.2020.101114 (date of access: 19.05.2026).

3. WavLM: large-scale self-supervised pre-training for full stack speech processing / S. Chen et al. *IEEE journal of selected topics in signal processing*. 2022. P. 1–14. URL: https://doi.org/10.1109/jstsp.2022.3188113 (date of access: 19.05.2026).

4. Yi J., Fu R. ADD 2022: the first audio deep synthesis detection challenge. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2202.08433 (date of access: 26.06.2026).

5. ASVspoof 5: evaluation of spoofing, deepfake, and adversarial attack detection using crowdsourced speech / X. Wang et al. *IEEE transactions on audio, speech and language processing*. 2026. P. 1–14. URL: https://doi.org/10.1109/taslpro.2026.3682962 (date of access: 19.05.2026).

6. Yi J., Fu R. ADD 2023: the second audio deepfake detection challenge. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2305.13774 (date of access: 26.06.2026).

7. Todisco M., Delgado H., Evans N. A new feature for automatic speaker verification anti-spoofing: constant q cepstral coefficients. *Odyssey 2016*. 2016. URL: https://doi.org/10.21437/odyssey.2016-41 (date of access: 19.05.2026).

8. ASVspoof 2019: spoofing countermeasures for the detection of synthesized, converted and replayed speech / A. Nautsch et al. *IEEE transactions on biometrics, behavior, and identity science*. 2021. Vol. 3, no. 2. P. 252–265. URL: https://doi.org/10.1109/tbiom.2021.3059479 (date of access: 19.05.2026).

9. End-to-End anti-spoofing with RawNet2 / H. Tak et al. *ICASSP 2021 – 2021 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, Toronto, ON, Canada, 6–11 June 2021. 2021. URL: https://doi.org/10.1109/icassp39728.2021.9414234 (date of access: 19.05.2026).

10. AASIST: audio anti-spoofing using integrated spectro-temporal graph attention networks / J.-w. Jung et al. *ICASSP 2022 – 2022 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, Singapore, Singapore, 23–27 May 2022. 2022. URL: https://doi.org/10.1109/icassp43922.2022.9747766 (date of access: 19.05.2026).

11. wav2vec 2.0: a framework for self-supervised learning of speech representations / A. Baevski et al. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2006.11477 (date of access: 19.05.2026).

12. Audio deepfake detection with self-supervised WavLM and multi-fusion attentive classifier / Y. Guo et al. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2312.08089 (date of access: 19.05.2026).

13. Arevalo J., Solorio T., Montes-y-Gómez M. Gated multimodal units for information fusion. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.1702.01992 (date of access: 19.05.2026).

14. Ростовцев В.В., Гриньов С.А. Гібридна система детекції синтезованого мовлення на основі SSL-ембеддінгів. *Радіоелектроніка та молодь у XXI столітті* : матеріали 30-го Міжнар. молодіж. форуму, 22 – 24 квітня 2026 р. Харків : ХНУРЕ, 2026. Т. 6. С. 105–107.

15. Paszke A., Gross S. PyTorch: an imperative style, high-performance deep learning library. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.1912.01703 (date of access: 26.06.2026).

16. Yang Y.-Y., Hira M. TorchAudio: building blocks for audio and speech processing. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2110.15018 (date of access: 26.06.2026).

17. Yamagishi J., Wang X. ASVspoof 2021: accelerating progress in spoofed and deepfake speech detection. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2109.00537 (date of access: 26.06.2026)..

18. Yi J., Wang C. Audio deepfake detection: a survey. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2308.14970 (date of access: 26.06.2026).

19. The partialspoof database and countermeasures for the detection of short fake speech segments embedded in an utterance / L. Zhang et al. *IEEE/ACM transactions on audio, speech, and language processing*. 2022. P. 1–13. URL: https://doi.org/10.1109/taslp.2022.3233236 (date of access: 26.06.2026).

20. Shen J., Pang R. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.1712.05884 (date of access: 26.06.2026).

21. Dieleman S., Zen H. WaveNet: a generative model for raw audio. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.1609.03499 (date of access: 26.06.2026).

22. Kim J., Kong J. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2106.06103 (date of access: 26.06.2026).

23. Yamamoto R., Song E., Kim J.-M. Parallel WaveGAN: a fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.1910.11480 (date of access: 26.06.2026).

24. Frank J., Schönherr L. WaveFake: a data set to facilitate audio deepfake detection. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2111.02813 (date of access: 26.06.2026).

25. Khalid H., Tariq S. FakeAVCeleb: a novel audio-video multimodal deepfake dataset. *arXiv.org*. URL: https://doi.org/10.48550/arXiv.2108.05080 (date of access: 26.06.2026).