Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня
магістра
(назва освітньо-кваліфікаційного рівня)

студента	*Тарана Данила Григоровича*
	(ПІБ)
академічної групи	*121М-20-1*
	(шифр)
спеціальності	*121 Інженерія програмного забезпечення*
	(код і назва спеціальності)
освітньої програми	*«Інженерія програмного забезпечення»*
	(назва освітньої програми)
на тему:	*Розробка програмного забезпечення просодичної модифікації мовного сигналу*

_____*Д.Г. Таран*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
розділів кваліфікаційної роботи				
спеціальний	*Проф. Якунін А.О.*			
економічний	*Проф. Вагонова О.Г.*			
Рецензент	*Проф. Байбуз О.Г.*			
Нормоконтролер	*Доц. Приходченко С.Д.*			

Дніпро
2022

Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
Завідувач кафедри
Програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 20 21 Року

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності *121 Інженерія програмного забезпечення*
(код і назва спеціальності)

студенту *121м-20-1* *Тарану Данилу Григоровичу*
(група) (прізвище та ініціали)

Тема кваліфікаційної роботи *Розробка програмного забезпечення просодичної модифікації мовного сигналу*

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора НТУ «Дніпровська політехніка» від ___.___.2021 р. № _____

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень – методи просодичної модифікації та розпізнавання мовного сигналу.

Предмет досліджень – використання існуючих програмних пакетів для обробки мовного сигналу, розробка програного забезпечення модифікації мовного сигналу.

Методи дослідження: методи просодичної модифікації, використання існуючих програмних пакетів для обробки мовного сигналу, розробка програного забезпечення модифікації мовного сигналу.

Мета роботи – дослідження методів просодичної модифікації мовного сигналу та можливості їх застосування для покращення якості розпізнавання мовного сигналу.

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Новизна запропонованих рішень визначається тим, що розроблено новий оригінальний алгоритм для попередньої обробки мовного сигналу перед розпізнаванням.

Практична цінність результатів полягає у тому, що в результаті проведеного дослідження було спроектовано алгоритм покращення якості розпізнавання мовного сигналу на основі методів просодичної модифікації – алгоритмів PSOLA та ERN.

4 ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Вхідними даними для програми є аудіо файл, що декодується до послідовності значень. Ці значення відповідають значенням амплітуди сигналу у часовому просторі, але представлені у більш зручному форматі для зберігання та відтворення звуку. Після проходження необхідної попередньої обробки проводиться аналіз сигналу, що полягає у вилученні інформативних ознак. На базі проведеного аналізу та отриманих від користувача параметрів встановлюються межі на рівень перетворень і повинна виконуватись модифікація даних за алгоритмами PSOLA та ERN.

5 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	**Строки виконання робіт (початок – кінець**
Аналіз джерел та постановка задачі	12.09.2021 - 30.09.2021
Побудова математичних моделей та розробка алгоритму	01.10.2021 - 31.10.2021
Розробка та тестування програмного забезпечення для просодичної модифікації мовного сигналу	01.11.2021 - 30.12.2021

Завдання видав _____ *Якунін А.О.*
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ *Таран Д.Г.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *12.09.2021 р.*

Термін подання кваліфікаційної роботи до ЕК *20.01.2021*

РЕФЕРАТ

Пояснювальна записка: ___ стор., ___ рис., ___ таблиці, ___ додатка, ___ джерел.

Об’єкт досліджень – методи просодичної модифікації та розпізнавання мовного сигналу.

Предмет досліджень – використання існуючих програмних пакетів для обробки мовного сигналу, розробка програного забезпечення модифікації мовного сигналу.

Мета роботи – дослідження методів просодичної модифікації мовного сигналу та можливості їх застосування для покращення якості розпізнавання мовного сигналу.

Новизна запропонованих рішень визначається тим, що розроблено новий оригінальний алгоритм для попередньої обробки мовного сигналу перед розпізнаванням.

Практична цінність результатів полягає у тому, що в результаті проведеного дослідження було спроектовано алгоритм покращення якості розпізнавання мовного сигналу на основі методів просодичної модифікації – алгоритмів PSOLA та ERN.

В основній частині роботи розглянуті актуальність роботи, питання покращення якості розпізнавання мовного сигналу, основні поняття і методи просодичної модифікації та розпізнавання мовного сигналу.

Результати роботи містять опис методів просодичної модифікації, що буди досліджені, принципи та перспективи застосування алгоритму покращення якості розпізнавання на основі методів просодичної модифікації.

У розділі «Економіка» проведені розрахунки трудомісткості розробки програмного забезпечення, витрат на створення ПЗ і тривалості його розробки, а також проведені маркетингові дослідження ринку збуту створеного програмного продукту.

Список ключових слів: РОЗПІЗНАВАННЯ МОВИ, ЯКІСТЬ РОЗПІЗНАВАННЯ, ОБРОБКА МОВНОГО СИГНАЛУ, ПРОСОДИЧНА МОДИФІКАЦІЯ, ЗМІЩЕННЯ ЧАСТОТИ ОСНОВНОГО ТОНУ, PSOLA, НОРМАЛІЗАЦІЯ ЕНЕРГІЇ, ERN.

ABSTRACT

Explanatory note: ___ pages, ___ figures, ___ tables, ___ appendices, ___ sources.

The object of research is methods of prosodic modification and recognition of speech signal.

The subject of research is the use of existing software packages for speech signal processing, development of software for speech signal modification.

The aim of the work is to study the methods of prosodic modification of the speech signal and the possibility of their application to improve the quality of speech signal recognition.

The novelty of the proposed solutions is determined by the fact that a new original algorithm has been developed for pre-processing of the speech signal before recognition.

The practical value of the results is that as a result of the study an algorithm was designed to improve the quality of speech signal recognition based on prosodic modification methods - PSOLA and ERN algorithms.

The main part of the paper considers the relevance of the work, improving the quality of speech signal recognition, basic concepts and methods of prosodic modification and speech signal recognition.

The results of the work contain a description of prosodic modification methods to be studied, principles and prospects of application of the algorithm for improving the quality of recognition based on prosodic modification methods.

In the section "Economics" calculations of the complexity of software development, the cost of creating software and the duration of its development, as well as marketing research of the market for the software product.

List of key words: LANGUAGE RECOGNITION, QUALITY RECOGNITION, SPEECH SIGNAL PROCESSING, PROSODIC MODIFICATION, BASIC FREQUENCY SHIFT, ERNOR TANNING, PSOL.

ЗМІСТ

ВСТУП

Мова – сигнал багатовимірний та варіативний. Припустимо, що кількість існуючих фонем для мови є визначеною. Для кожної фонеми можна застосувати різну гучність та довжину. Діапазон цих параметрів звуку визначається більш високорівневими характеристиками мови, такими як індивідуальність голосу людини, акцент, особливості вимови та фізичний і емоційний стан людини на час вимови. Так кожна фонема може набути чималу кількість тональностей, а слова мають багато можливих інтонацій і фонетичних особливостей при вимові.

Для забезпечення якості система розпізнавання мови повинна врахувати якомога більше особливостей і для цього вона укомплектовується відповідними алгоритмами. Існують алгоритми, що враховують наявність шуму та помилок при вимові, налаштовуються на індивідуальність голосу та визначають акцент.

Поліпшення якості одночасно досягається і за рахунок розширення навчальної вибірки, для цього існує декілька підходів:

1) штучне розширення – різноманітне зашумлення мовного сигналу. При цьому інший тип шумів, що не було враховано при навчанні, буде у подальшому зумовлювати помилки при розпізнаванні;

2) природне розширення – накопичення великої кількості нестандартних фрагментів мови, так як деякі акустичні характеристики користувача можуть не збігатися з типовими голосами, що використовуються у вибірці. Проблема особливо актуальна при розпізнаванні мови на нерідній для диктора мові, при порушення мовотворення та розпізнаванні дитячого мовлення;

3) адаптація – розширення навчальної вибірки за рахунок великої кількості фрагментів мови одного диктора, щоб врахувати варіації голосу. Система з таким підходом називається залежною від диктора, тому що добре працює лише з одним диктором.

Окремо можна виділити вплив емоцій на інтонацію висловлювання. Мова має непередбачуваний емоційний контент, тож колекціонування різних емоційних еталонів для систем розпізнавання є довгим та витратним. Так перспективним постає підхід, що дозволяє не накопичувати навчальну вибірку, а перетворювати вхідне висловлювання так, щоб максимізувати точність подальшого розпізнавання. Для впровадження такого перетворення пропонується застосувати методи просодичної модифікації мовного сигналу.

Дослідження використання просодичної модифікації в контексті розпізнавання мовного сигналу поетапно виконано у трьох розділах даної кваліфікаційної роботи.

Перший розділ містить проблематику поставленої задачі та перспективи її вирішення. В другому розділі розглянуто теоретичні основи предметної області, проведено огляд існуючих методів вирішення поставленої задачі, пояснення та математичні основи алгоритмів, що використовуються для вирішенні задачі. У третьому розділі представлено загальну схему роботи та структуру алгоритму, спроектованого для вирішення поставленої задачі, з використанням програмного забезпечення. У четвертому розділі подано результати роботи – скріншоти роботи розробленого алгоритму та порівняльну характеристику впливу розробленого алгоритму на якість розпізнавання мовного силналу.

РОЗДІЛ 1
АНАЛІЗ МЕТОДІВ ТА АЛГОРИТМІВ ОБРОБКИ МОВНИХ СИГНАЛІВ

1.1. Проблеми покращення якості розпізнавання мовного сигналу.

У контексті розпізнавання мови, покращення якості розпізнавання досягається за рахунок розширення навчальної вибірки та додаткових алгоритмів обробки мовного сигналу, що враховують наявність шуму та помилок при вимові, налаштовуються на індивідуальність голосу та визначають акцент [1].

Емоційний стан людини при вимові значно та непередбачувано впливає та накладається на інтонацію висловлювання, тому колекціонування різних емоційних еталонів для систем не можна визнати ефективним.

Перші дослідження зв’язку між емоціями та мовленням людини було проведено ще у XIX столітті. Інтерес до теми проявляли природознавці у наукових та філософських роботах. Так Г. Спенсер детально описав зміни голосу під впливом різних емоцій у філософському есе про походження музики. Ч. Дарвін у своїх роботах наводив спостереження щодо схожості вираження емоції у людини та тварин, але підкреслював, що його дослідження не є достатніми, щоб повністю освітити тему передачі емоцій голосом людини. Деякі дослідники зверталися до цієї теми і впродовж XX століття, але не досягли значних успіхів. Попри те, що невербальна поведінка активно досліджувалася у психології, однозначних висновків та науково достовірних результатів отримано не було. Це в певній мірі пов’язують з недостатньою технічною базою для досліджень.

За останні кілька десятиліть просодична модифікація займає місце однієї із головних тем, що цікавлять дослідників у галузі обробки мовного сигналу. Просодична модифікація (prosodic modification або prosody modification) – це процес інтонаційного перетворення мовного сигналу, який впливає на частоту основного тону (ЧОТ) та довжину сигналу, але запобігає спектральній

деформації [2], порушенню семантики повідомлення та зберігає натуральність звучання мови [3].

Чимало досліджень проведено щодо застосування просодичної модифікації на основі методів зміщення ЧОТ у контексті синтезу мови (text-to-speech, TTS) для надання штучно сгенерованим висловлюванням природної та доцільної експресивності.

У [4] представлено методи просодичної модифікації для застосування у TTS системах, що працюють режимі реального часу. У таких системах край важливо мінімізувати обчислювальну складність і, таким чином, час обробки даних та швидкості відповіді системи на запит користувача. Саме тому було запропоновано виконувати пошук вокалізованих ділянок висловлювання та спиратися на модифікацію лише окремих частин цих ділянок. Таких підхід дозволив дослідникам зменшити обчислювальну складність на 75-90% від базового підходу до модифікації, проте було відзначено, що такий напрям досі є недостатньо вивченим для практичного застосування. «Швидкий» підхід, що базувався на модифікації частоти основного тону мовного сигналу, успішно застосовували і інші дослідники [5]. Отримані результати відображали достатній рівень якості модифікованого сигналу, проте було підкреслено, що при значному зміщенні ЧОТ може виникнути необхідність використання додаткових алгоритмів нормалізації гучності звуку.

Пізніше дослідники почали акцентувати увагу на зв’язку між емоційністю мови та інтонацією, на зміну якої і націлені методи просодичної модифікації, для впровадження нового підходу для оптимального пошуку та модифікації ділянок висловлювання [6].

Деякі з досліджень останніх років почали прицільно вивчати просодичну модифікацію зі зворотної сторони – галузі розпізнавання мови (speech-to-text), проте не позбавилися від контексту впливу емоцій на мовний сигнал [2].

У даній роботі просодична модифікація розглядається як перспективний підхід для попередньої обробки сигналу з метою покращення якості подальшого розпізнавання мовного сигналу; розглянуто алгоритми розпізнавання мови, зміщення ЧОТ та нормалізації енергії (гучності).

1.2. Алгоритми розпізнавання мови

Розпізнавання мови – одна із задач, яка завжди давалася людям легше, ніж комп’ютерам. Проте дослідження у даному напрямку тривають вже кілька десятиліть [7]. За останні роки було здійснено значний прорив у підвищенні якості розпізнавання. Цей прорив зумовлений активним використанням глибокого навчання, яке прийшло на зміну традиційній архітектурі систем автоматичного розпізнавання мови (ASR).

У свій час використання нейронної мережі як акустичної моделі для традиційної ASR покращило точність розпізнавання при незмінному обсязі навчальних даних та розмірі моделі. Найбільш ефективним було поєднання рекурентної нейронної мережі (RNN) і прихованої марківської моделі (HMM) у, так званому, гібридному підході. Однак така архітектура не тільки успадковувала недоліки HMM, але і не дозволяла в повній мірі використовувати потенціал RNN для моделювання послідовностей. Пізніше були представлені end-to-end ASR [8][9], які були засновані на нейронних мережах і завдяки специфічному алгоритму навчання мали можливість напряму передбачати текст отримуючи на вхід набір акустичних ознак певного висловлювання. Саме такі моделі набули популярності в останні роки.

Нейронні мережі стали невід’ємною частиною ASR. Для розпізнавання мови, як правило, використовують RNN, іноді у поєднанні зі згортковими нейронними мережами (CNN).

Розрізняють два типи end-to-end моделей для розпізнавання мови: connectionist temporal classification та sequence-to-sequence [10]. Обидва ці підходи end-to-end розпізнавання мови, як правило, передбачають послідовність літер. Sequence-to-sequence модель кодує послідовність акустичних ознак у один вектор, а потім декодує цей вектор у послідовність символів (літер). Модель оснащена механізмом уваги, що покращує цей метод, обумовлюючи різне резюме вхідної послідовності на кожному етапі декодування.

Connectionist temporal classification [11] модель приймає акустичні ознаки на вхід і навчається передбачати символи для кожного фрейму. Символи зазвичай представляють собою літери або звуки. Потім алгоритм проходить по всіх можливих послідовностях символів, які підходять до транскрипції, та обирає найбільш імовірну.

Розглянемо детально алгоритм connectionist temporal classification (CTC), що дозволяє ефективно використовувати нейронні мережі для розпізнавання мови у end-to-end системах. Алгоритм, що використовує виходи RNN для розпізнавання мови отримав назву *connectionist temporal classification (CTC)*. А процес незалежного маркування послідовності нейронною мережею на кожному часовому кроці, або фреймі, вхідної послідовності називається *пофреймова класифікація (framewise classification).*

Важливий крок для забезпечення сумісного використання RNN та CTC складається у перетворенні виходів нейронної мережі у розподіл умовної ймовірності послідовностей маркерів. Таким чином, виходи нейронної мережі стають вхідними даними для CTC алгоритму. Отримана *CTC мережа* є класифікатором, що дозволяє обирати найбільш ймовірне маркування для поданої вхідної послідовності.

1.3 Постановка задачі

Мета роботи: дослідження методів просодичної модифікації мовного сигналу та можливості їх застосування для покращення якості розпізнавання.

Для досягнення мети необхідно вирішити наступні завдання:

– виконати літературний огляд питання якості та покращення якості розпізнавання мовного сигналу;

– розглянути алгоритми розпізнавання мови та методи просодичної модифікації, обрати алгоритми та методи для дослідження та розглянути принцип їх роботи;

– спроектувати та розробити алгоритм покращення якості розпізнавання на основі методів просодичної модифікації;

– зробити висновки щодо досяжності мети дослідження.

РОЗДІЛ 2
МЕТОДИ ТА АЛГОРИТМИ АНАЛІЗУ МОВНОГО СИГНАЛУ

2.1. Маркування несегментованих послідовностей даних

Задача маркування несегментованих послідовностей даних – це *часова класифікація (temporal classification).*

Нехай S – це множина навчальних прикладів, що взяті з фіксованого розподілу $D_{X\times Z}$. Вхідний простір $X = (R^m)^*$ представляє собою множину всіх послідовностей m-вимірних дійсних векторів. Цільовий простір $Z = L^*$ – це множина усіх послідовностей над скінченним алфавітом з L маркерів. Елементи L^* називаються *послідовностями маркерів* або *маркуваннями*. Кожний елемент множини S складається з пари послідовностей (x, z). Довжина цільової послідовності $z = (z_1, z_2, \dots, z_U)$ не більша, ніж довжина вхідної послідовності $x = (x_1, x_2, \dots, x_T)$, тобто $U \leq T$. Так як x та z мають різну довжину, не існує апріорного методу для їх порівняння.

Необхідно побудувати *часовий класифікатор* $h : X \longmapsto Z$ на основі множини S та навчити його розпізнавати нові вхідні послідовності таким чином, щоб мінімізувати метрику помилки, визначений задачею.

2.2. Частота помилкового маркування

Для заданої тестової множини $S' \subset D_{X\times Z}$, яка не перетинається із множиною S *частота помилкового маркування (label error rate, LER)* часового класифікатора h визначається як нормалізована відстань редагування між виданими класифікаціями і цільовими маркуваннями на множині S', тобто

$$LER(h, S') = \frac{1}{Z}\sum_{(x,z)\,\in\, S'} ED\big(h(x)\big)\,, \tag{2.1}$$

де Z – загальна кількість цільових маркувань в S, а $ED(p, q)$ – відстань редагування між двома послідовностями p та q, тобто мінімальна кількість вставок, замін та видалень необхідна для отримання q із p.

Для даної задачі використовується саме така метрика помилки; вона є природною для задач, в яких потрібно мінімізувати частоту помилок транскрипції.

2.3. Від виходів мережі до маркувань

CTC мережа має вихідний шар *softmax* (застосовується як узагальнення логістичної функції для задач класифікації, коли кількість можливих класів більше двох; функція softmax перетворює кожне значення z_i вхідного вектору Z у ймовірність належності до класу i) з кількістю нейронів на один більше, ніж кількість міток в L. Передаточні функції перших $|L|$ нейронів інтерпретуються як ймовірності спостереження відповідних маркерів у конкретні моменти часу. Передаточна функція додаткового нейрона – це ймовірність спостереження «порожнього» маркера ($blank$). Разом ці виходи визначають ймовірності всіх можливих способів порівняння всіх можливих послідовностей маркерів та вхідної послідовності. Повна ймовірність будь-якої послідовності маркерів може бути знайдена як сума імовірностей всіх її порівнянь.

Більш формально це означає, що для вхідної послідовності x довжиною T визначена RNN, яка має m вхідних нейронів, n вихідних і ваговій вектор w – неперервне відображення $N_w : (R^m)^T \longmapsto (R^n)^T$. Нехай $y = N_w(x)$ – послідовність виходів мережі, а y_k^t – передаточна функція вихідного нейрона k в момент часу t. Тоді y_k^t – це імовірність спостереження маркеру k в момент часу t, яка визначає розподіл послідовностей довжиною T на множині L'^T, тобто *шляхів* $\pi \in L'^T$, отриманих з алфавіту $L' = L \cup \{blank\}$:

$$p(\pi|x) = \prod_{t=1}^{T} y_{\pi_t}^t \,,\ \forall \pi \in {L'}^T. \qquad (2.2)$$

Слід зазначити, що для CTC мережі, яка заснована на фонемах, алфавіт $L' = L \cup \{blank\}$, а для CTC мережі на графемах – $L' = L \cup \{blank, space\}$, де $blank$ – порожній маркер, $space$ – пауза між словами.

У (2.2) передбачається, що виходи мережі в різні моменти часу умовно незалежні для даного внутрішнього стану мережі. Це забезпечується вимогою відсутності зворотних зв'язків у вихідному шарі і зворотних зв'язків між вихідним шаром та мережею.

Далі будується таблиця багато-до-одного $\mathcal{B} : {L'}^T \mapsto L^{\leq T}$, де $L^{\leq T}$ – множина можливих маркувань (тобто множина послідовностей довжиною не більше ніж T, складених з алфавіту вихідних маркерів L). Це здійснюється простим видаленням всіх «порожніх» та повторюваних маркерів зі шляхів, наприклад, $\mathcal{B}(a\_ab\_) = \mathcal{B}(\_aa\_\ \_abb) = aab$.

Інтуїтивно, це відповідає поверненню наступного маркеру, коли мережа перемикається з передбачення «порожнього» маркеру до передбачення маркеру або від передбачення одного маркеру до передбачення іншого (рис 2.1).

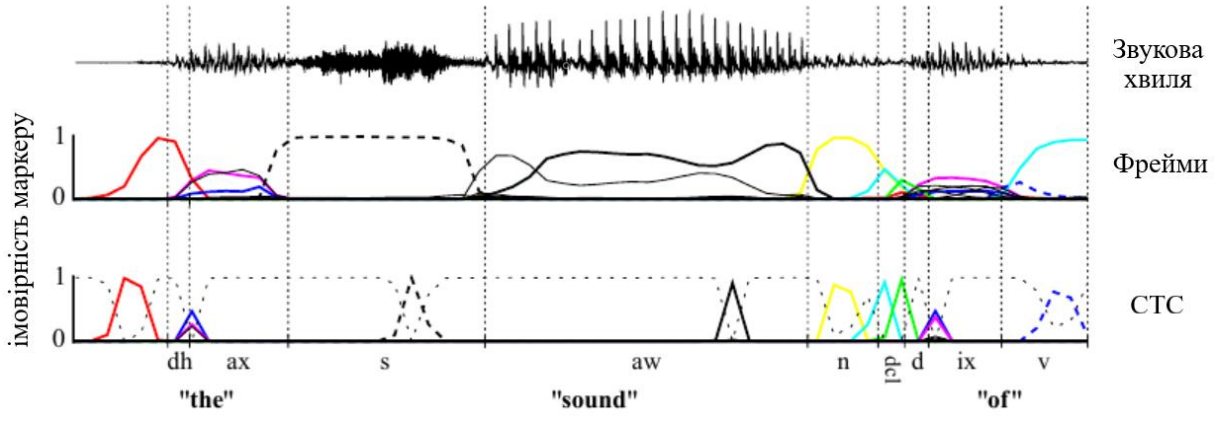

Рис. 2.1 Класифікація мовного сигналу пофреймовою та CTC мережею

Кольорові лінії – це виходи мережі, що відповідають імовірностям спостереження фонем у конкретні моменти часу. CTC мережа передбачає лише послідовність фонем (зазвичай виглядає як ряд піків, розділених «порожніми» або нульовими передбаченнями), коли пофреймова мережа намагається порівнювати їх з ручною сегментацією (вертикальні лінії). Пофреймова мережа визначає помилку, коли межі сегментів не співпадають, навіть якщо фонема передбачена вірно (наприклад, «dh»). CTC мережа, як правило, передбачає їх подвійним піком фонеми, які завжди ідуть поряд («dsl», що закінчується на «d»). Обране маркування може бути отримано напряму з виходів CTC мережі (за піками), в той час як передбаченням пофреймової мережі вимагають постобробки.

Очевидно, що за таблицею $\mathcal{B}$ є багато шляхів, що відповідають єдиному маркуванню. Таблиця $\mathcal{B}$ використовується для визначення умовної ймовірності отриманого маркування $l \in L^{\leq T}$ як суми ймовірностей всіх шляхів з яких складається маркування:

$$p(l|x) = \sum_{\pi \in \mathcal{B}^{-1}(l)} p(\pi|x). \tag{2.3}$$

Відповідно до вищеподаного формулювання, класифікатор повинен повертати найбільш ймовірне маркування для вхідної послідовності:

$$h(x) = \arg\max_{l \in L^{\leq T}} p(l|x). \tag{2.4}$$

Задача знаходження цього маркування називається *декодуванням*. Наступні два наближених методи забезпечують хороші результати у цій задачі.

Перший метод (*декодування кращого шляху, best path decoding*) заснований на припущенні, що найбільш ймовірний шлях буде відповідати найбільш ймовірному маркуванню:

$$h(x) \approx \mathcal{B}(\pi^*),$$
$$\text{де } \pi^* = arg \max_{\pi \in N^t} p(\pi|x). \tag{2.5}$$

Декодування кращого шляху обчислюється тривіально, так як π^* є просто конкатенацією найбільш активних виходів на кожному часовому кроці. Однак це не гарантує знаходження найбільш імовірного маркування.

Інший метод (*декодування префіксним пошуком, prefix search decoding*) покладається на той факт, що, модифікуючи алгоритм прямого-зворотного ходу, можна ефективно обчислювати ймовірності послідовних розширень префіксів маркування (рис. 2.2). Кожний префікс або закінчується («e»), або розширює префікс батьківської вершини. Вказані значення вузлів – це ймовірності того, що маркери починаються зі вказаного (батьківського) префіксу. Значення кінцевого вузла означає ймовірність того, що маркування закінчується на батьківському вузлі. На кожній ітерації знаходяться розширення найбільш ймовірного префіксу з тих, що лишилися. Пошук закінчується, коли одне маркування є більш ймовірним, ніж будь-який префікс, що лишився.

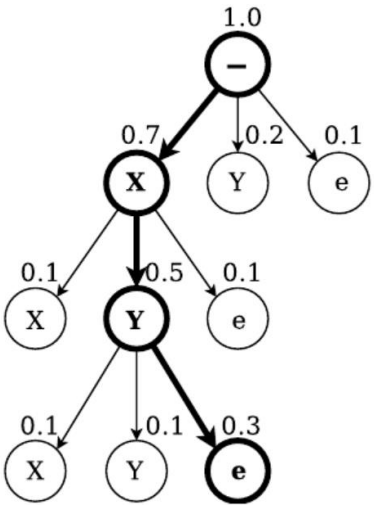

Рис. 2.2. Префіксний пошук на алфавіти X,Y

При достатній кількості часу, декодування префіксним пошуком завжди знаходить найбільш ймовірне маркування. Однак, максимальне число префіксів, які алгоритм повинен перевірити, зростає експоненціально відносно довжини вхідної послідовності. Таким чином, необхідно застосування евристики, щоб завершити роботу алгоритму за розумну кількість часу.

Помічено, що виходи навченої CTC мережі мають тенденцію формувати ряди піків, розділені «порожніми» маркерами, які мають велику ймовірність передбачення (рис. 2.1). Завдяки цьому вхідна послідовність ділитися на секції, які, швидше за все, починаються і закінчуються на «порожній» маркер. Поділ на секції відбувається шляхом вибору крайових точок, в яких імовірність спостереження «порожнього» маркеру вище певного порогу. Потім обчислюється найбільш ймовірний маркер для кожної секції індивідуально і ці маркери об'єднуються для отримання остаточної класифікації.

На практиці префіксний пошук добре працює з цією евристикою і, здебільшого, знаходить найкращий шлях. Однак у деяких випадках він зазнає невдачі, наприклад, якщо одна і та ж мітка передбачається з малою ймовірністю на обох кінцях секції.

2.4. Алгоритм прямого-зворотного ходу для CTC

З огляду на (2.3), запропонувати ефективний спосіб обчислення умовних ймовірностей $p(l|x)$ індивідуальних маркувань виявляється проблематичним: це сума всіх шляхів, відповідних до даного маркування, і, як правило, їх дуже багато.

Проблема може бути вирішена за допомогою алгоритму динамічного програмування, схожого на алгоритм прямого-зворотного ходу.

Алгоритм прямого-зворотного ходу – алгоритм для обчислення апостеріорних ймовірностей послідовності станів за наявності послідовності спостережень. Інакше кажучи, алгоритм, що обчислює ймовірність специфічної

послідовності спостережень. Основна ідея полягає у тому, що сума шляхів, відповідних до маркування, може бути розбита на ітеративну суму шляхів, що відповідають префіксам цього маркування.

Такі ітерації можуть бути ефективно обчислені рекурсивними *прямими* та *зворотними* змінними.

Для деякої послідовності q довжини r, $q_{1:p}$ та $q_{r-\,p:r}$ – це перші і останні p символів відповідно. Пряма змінна $\alpha_t(s)$ для маркування l – це повна ймовірність $l_{1:s}$ в момент часу t, тобто

$$\alpha_t(s) \stackrel{\text{def}}{=} \sum_{\substack{\pi \in N^T: \\ \mathcal{B}(\pi_{1:t})=l_{1:s}}} \prod_{t'=1}^{t} y_{\pi_{t'}}^{t'}. \tag{2.6}$$

Очевидно, що $\alpha_t(s)$ можна обчислити рекурсивно за $\alpha_{t-1}(s)$ та $\alpha_{t-1}(s-1)$.

Додамо «порожні» маркери у вихідні шляхи і розглянемо модифіковану послідовність маркерів l' з «порожніми» маркерами на початку та в кінці, а також між кожною парою маркерів. Довжина l' становить $2|l|+1$. При обчисленні ймовірностей префіксів l' додамо всі переходи між «порожніми» і «непорожніми», і між будь-якою парою різних «непорожніх» маркерів. Додамо всі префікси, що починаються з «порожнього» (b) або з першого символу в $l(l_1)$.

Цим зумовлені наступні правила ініціалізації та рекурсії

$$\begin{aligned} \alpha_1(1) &= y_b^1 \\ \alpha_1(2) &= y_{l_1}^1 \\ \alpha_1(s) &= 0, \quad \forall s > 2 \end{aligned} \tag{2.7}$$

$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) y^t_{l'_s}, & l'_s = b \text{ або } l'_{s-2} = l'_s \\ \left(\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)\right) y^t_{l'_s}, & \text{у інших випадках} \end{cases}, \qquad (2.8)$$

де

$$\bar{\alpha}_t(s) \stackrel{\text{def}}{=} \alpha_{t-1}(s) + \alpha_{t-1}(s-1). \qquad (2.9)$$

Слід зауважити, що $\alpha_t(s) = 0, \forall s < |l'| - 2(\mathrm{T} - \mathrm{t}) - 1$, тому що ці змінні відповідають станам для яких залишається недостатньо часових кроків для укомплектування послідовності (вершини без зв'язків у правому верхньому куті на рис. 2.3). І також $\alpha_t(s) = 0, \forall s < 1$.

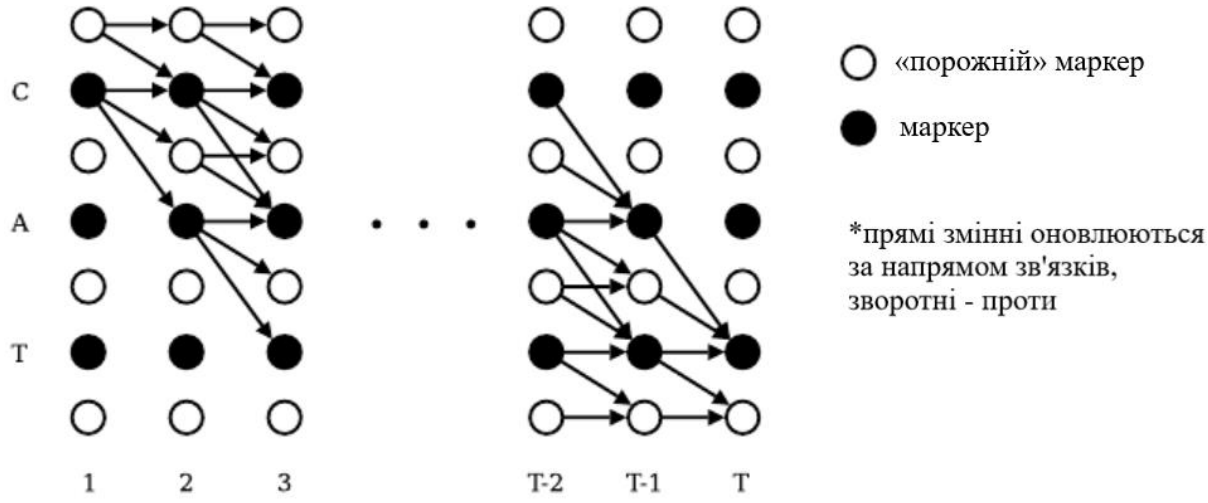

Рис 2.3. Ілюстрація алгоритму прямого-зворотного ходу
для маркування «CAT»

Таким чином, імовірність l – це сума повних ймовірностей l' з та без кінцевого «порожнього» маркеру в момент часу T.

$$p(l|x) = \alpha_T(|l'|) + \alpha_T(|l'| - 1). \qquad (2.10)$$

Аналогічно, зворотні змінні $\beta_t(s)$ визначаються як повна ймовірність $l_{s:|l|}$ в момент часу t.

$$\beta_t(s) \stackrel{\text{def}}{=} \sum_{\substack{\pi \in N^T: \\ \mathcal{B}(\pi_{t:T})=l_{s:|l|}}} \prod_{t'=t}^{T} y_{\pi_{t'}}^{t'} \tag{2.11}$$

$$\begin{aligned} \beta_T(|l'|) &= y_b^T \\ \beta_T(|l'|-1) &= y_{l_{|l|}}^T \\ \beta_T(s) &= 0, \forall s < |l'|-1 \end{aligned}$$

$$\beta_t(s) = \begin{cases} \bar{\beta}_t(s) y_{l'_s}^t, & l'_s = b \text{ або } l'_{s+2} = l'_s \\ \left(\bar{\beta}_t(s) + \beta_{t+1}(s+2)\right) y_{l'_s}^t, & \text{у інших випадках} \end{cases}, \tag{2.12}$$

де

$$\bar{\beta}_t(s) \stackrel{\text{def}}{=} \beta_{t+1}(s) + \beta_{t+1}(s+1). \tag{2.13}$$

Слід зауважити, що $\beta_t(s) = 0, \forall s > 2t$ (вершини без зв'язків у лівому нижньому куті на рис. 2.3) та $\forall s > |l'|$.

На практиці, вищеописані рекурсії незабаром приведуть до перевантаження комп'ютера. Один зі способів уникнути цього полягає у перемасштабуванні прямих та зворотних змінних. Якщо визначити

$$C_t \stackrel{\text{def}}{=} \sum_s \alpha_t(s), \qquad \hat{\alpha}_t(s) \stackrel{\text{def}}{=} \frac{\alpha_t(s)}{C_t} \tag{2.14}$$

та замінити α на $\hat{\alpha}$ у правій частині (2.8) та (2.9), прямі змінні будуть залишатися у обчислюваному діапазоні.

Аналогічно, для зворотних змінних можна визначити

$$D_t \stackrel{\text{def}}{=} \sum_s \beta_t(s), \qquad \hat{\beta}_t(s) \stackrel{\text{def}}{=} \frac{\beta_t(s)}{D_t} \tag{2.15}$$

та замінити β на $\hat{\beta}$ в правій частині (2.11) та (2.12).

Для визначення помилки максимальної правдоподібності необхідно використовувати натуральні логарифми ймовірностей цільових маркувань. Після перемасштабування змінних вони приймають дуже просту форму:

$$\ln\big(p(l|x)\big) = \sum_{t=1}^{T} \ln(C_t) \tag{2.16}$$

2.5. Навчання з урахуванням максимальної правдоподібності

Мета навчання з урахуванням максимальної правдоподібності в одночасній максимізації логарифмів ймовірностей усіх вірних класифікацій навчальної множини. У даному випадку це означає мінімізацію наступної цільової функції, що називається *функцією втрат (CTC loss)*:

$$O^{ML}(S, N_w) = -\sum_{(x,z) \in S} \ln\big(p(z|x)\big) \tag{2.17}$$

Для навчання мережі методом градієнтного спуску, необхідно диференціювати (2.17) відносно виходів нейронної мережі.

Так як навчальні приклади незалежні, їх можна розглядати окремо один від одного:

$$\frac{\partial O^{ML}(\{(x,\ z)\}, N_w)}{\partial y_k^t} = -\frac{\partial \ln\big(p(z|x)\big)}{\partial y_k^t} \tag{2.18}$$

Розглянемо застосування алгоритму для обчислення (2.18). Ключовий момент полягає в тому, що для маркування l добуток прямих та зворотних змінних, при заданих s та t, це ймовірність всіх шляхів відповідних до l, які проходять через символ s в момент часу t. А саме, з (2.5) та (2.12) отримуємо:

$$\alpha_t(s)\,\beta_t(s) = \sum_{\substack{\pi \in \mathcal{B}^{-1}(l):\\ \pi_t = l'_s}} y^t_{l'_s} \prod_{t=1}^{T} y^t_{\pi_t} . \tag{2.19}$$

Перегрупування та підстановка виразів з (2.1) дає

$$\frac{\alpha_t(s)\,\beta_t(s)}{y^t_{l'_s}} = \sum_{\substack{\pi \in \mathcal{B}^{-1}(l):\\ \pi_t = l'_s}} p(\pi|x) . \tag{2.20}$$

З (2.4) видно, що частка повної ймовірності $p(l|x)$, завдяки цим шляхам, проходить через l'_s в момент часу t. Для будь-якого t, можна таким чином визначити суму по всім s, щоб отримати:

$$p(l|x) = \sum_{s=1}^{|l'|} \frac{\alpha_t(s)\,\beta_t(s)}{y^t_{l'_s}} . \tag{2.21}$$

Для диференціювання відносно y^t_k, необхідно лише розглянути ті шляхи, що проходять через маркер k в момент часу t. З огляду на, що один і той самий маркер (включаючи «порожній» маркер) може повторюватися кілька разів для одного маркування l, визначимо множину позицій, де мітка k зустрічається в якості $lab(l,\ k) = \{s :\ l'_s = k\}$, яке може бути і порожнім.

Потім, диференціюючи (2.21), отримуємо:

$$\frac{\partial p(l|x)}{\partial y^t_k} = \frac{1}{{y^t_k}^2} \sum_{s \in lab(l,k)} \alpha_t(s)\,\beta_t(s). \tag{2.22}$$

Варто зазначити, що

$$\frac{\partial \ln\left(p(l|x)\right)}{\partial y_k^t} = \frac{1}{p(l|x)} \frac{\partial p(l|x)}{\partial y_k^t} \quad (2.23)$$

можна встановити $l = z$ і замінити (2.10) і (2.22) на (2.23) для диференціювання цільової функції.

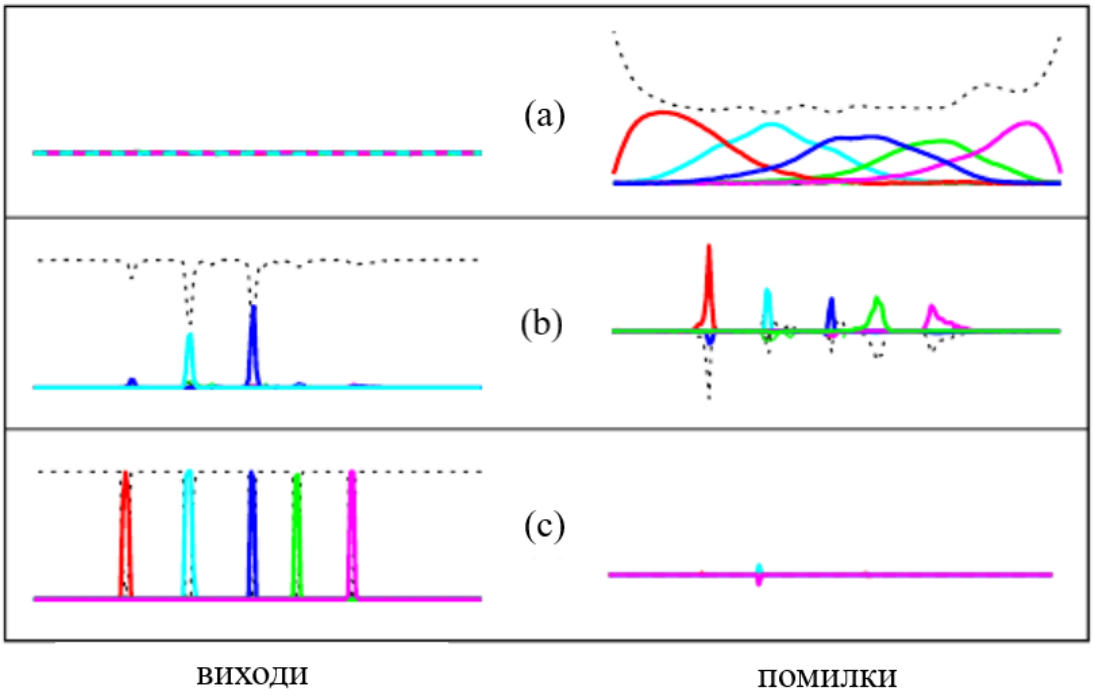

Рис. 2.4. Еволюція сигналу помилки CTC у процессі навчання

І нарешті, для застосування градієнтного алгоритму зворотного поширення помилки на шарі softmax, необхідно використовувати похідну функцію від цільової з урахуванням *ненормалізованих* виходів u_k^t.

Якщо використовується перемасштабування, то виходить:

$$\frac{\partial O^{ML}(\{(x,\ z)\}, N_w)}{\partial u_k^t} = y_k^t - \frac{1}{y_k^t\ Z_t} \sum_{s \in lab(z,k)} \hat{\alpha}_t(s)\ \hat{\beta}_t(s), \quad (2.23)$$

де

$$Z_t \stackrel{\text{def}}{=} \sum_{s=1}^{|l'|} \frac{\hat{\alpha}_t(s)\,\hat{\beta}_t(s)}{y^t_{l'_s}}. \qquad (2.24)$$

Рівняння (2.23) – це «сигнал помилки» отриманий мережею під час навчання (рис. 2.4). Пунктирна лінія – це нейрон «порожнього» маркування. Помилки зростають та спадають відносно вихідних значень. (a) На початку навчання мережа має малі випадкові вагові коефіцієнти, а помилка визначається лише цільовою послідовністю. (b) Мережа починає робити змістовні передбачення та локалізувати помилку навколо них. (c) Мережа добре передбачає вірні маркування і помилка фактично зникає.

2.6. Процес розпізнавання мови

CTC позиціонується як метод навчання RNN для маркування послідовностей несегментованих даних безпосередньо, в межах єдиної мережевої архітектури.

На вхід мережі подаються мінімально оброблені дані мовного сигналу, тобто вхідні дані на кожному часовому кроці як правило представляють спектр фрейму або навіть сам фрейм (рис. 2.5-2.6).

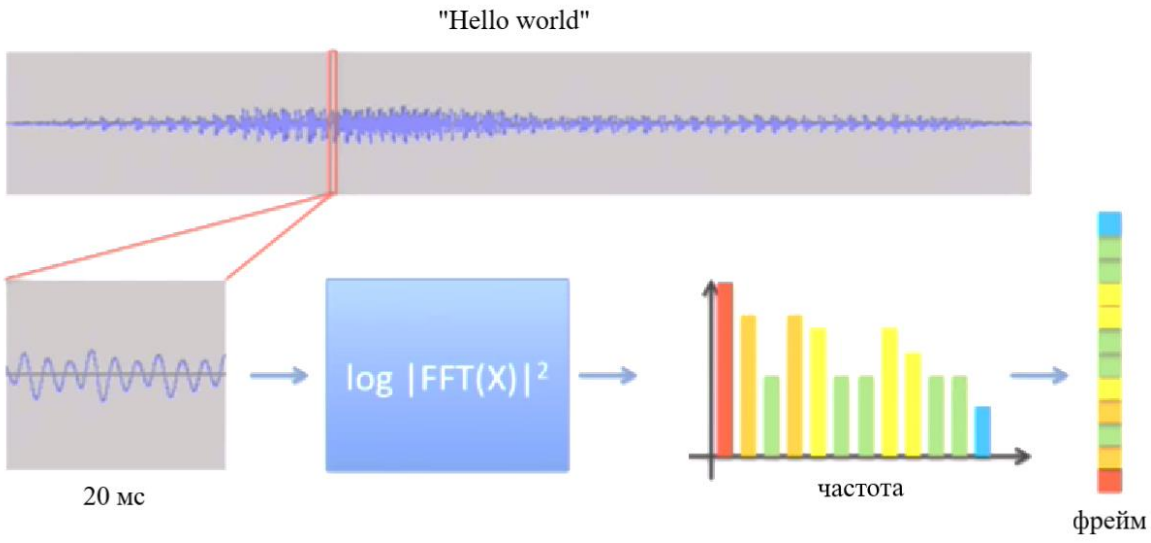

Рис. 2.5. Вилучення спектру

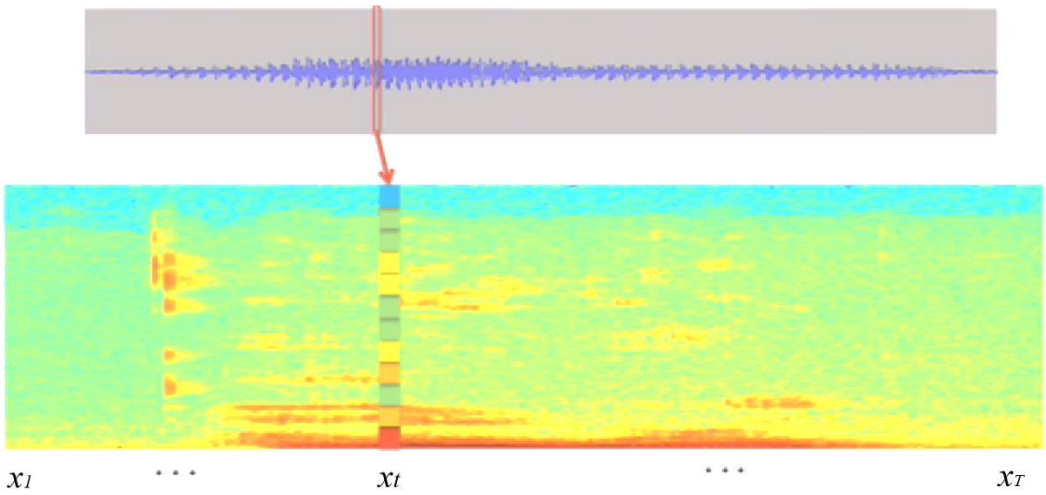

Рис. 2.6. Спектрограма та спектр сигналу

Цільова функція RNN може бути перетворена так, щоб з даного розподілу безпосередньо максимізувати ймовірності правильних маркерів (рис. 2.7).

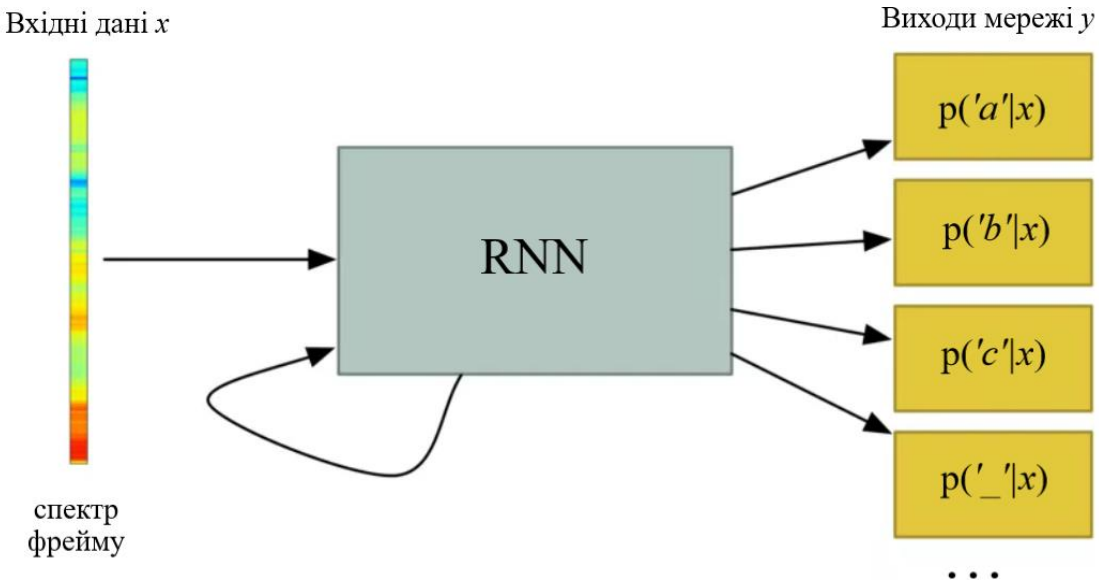

Рис. 2.7. Передбачення маркеру для фрейма

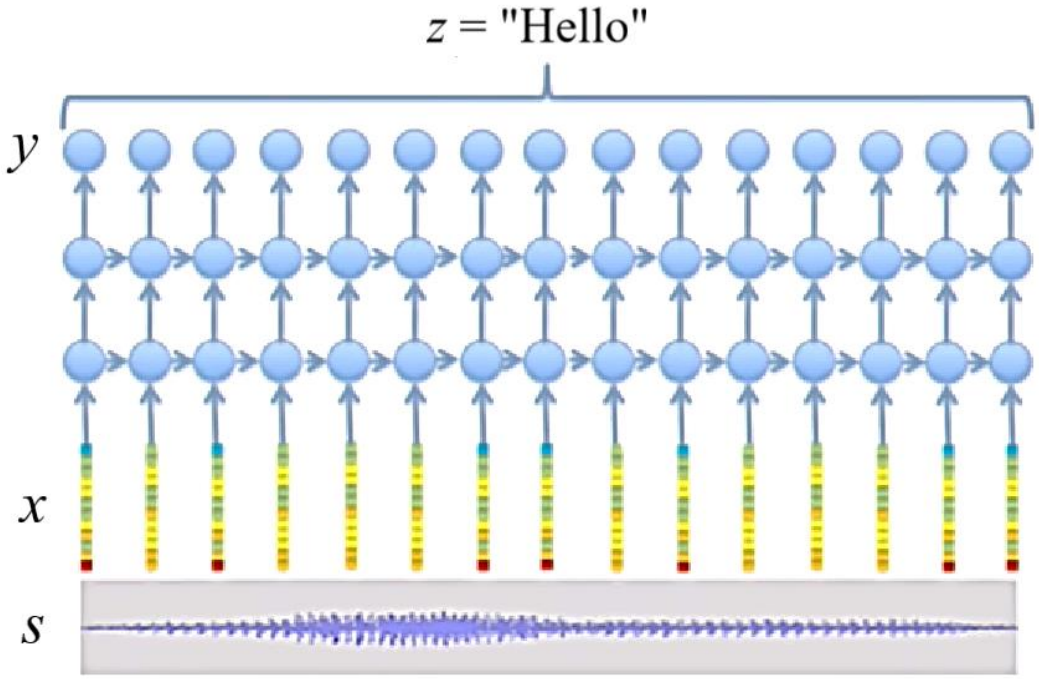

Рис. 2.8. Маркування послідовності фреймів CTC мережею

Виходи мережі інтерпретуються як імовірнісний розподіл усіх можливих послідовностей маркерів, обумовлених даною вхідною послідовністю. Застосовуючи префіксний пошук з евристикою у вигляді алгоритму прямого-зворотного ходу, можна ефективно обчислити найбільш імовірну послідовність маркерів (рис. 2.8-2.9). Так як цільова функція диференційовна, мережа може бути навчена класичним алгоритмом зворотного поширення помилки в часі.

CTC є зрілим алгоритмом, якому присвячено багато важливих досліджень. Принцип роботи CTC мережі усуває необхідність попередньої сегментації навчальних даних та постобробки виходів. Саме з цих причин застосування нейронних мереж для розпізнавання мови до певного моменту було обмежено.

Заснована на нейронній мережі CTC модель, як правило, робасна до часового та просторового шуму. Для роботи з RNN необхідно вибрати представлення для вхідних та вихідних даних. Важливо, що такі моделі не отримують явне фонетичне спостереження, на відміну від традиційних систем,

які зазвичай спираються на акустичну модель, навчену передбачати фонетичні одиниці. Однак, інтуїтивно зрозуміло, що end-to-end моделі повинні згенерувати деяке внутрішнє уявлення, яке дозволить їм абстрагуватися від фонологічних одиниць (фонем).

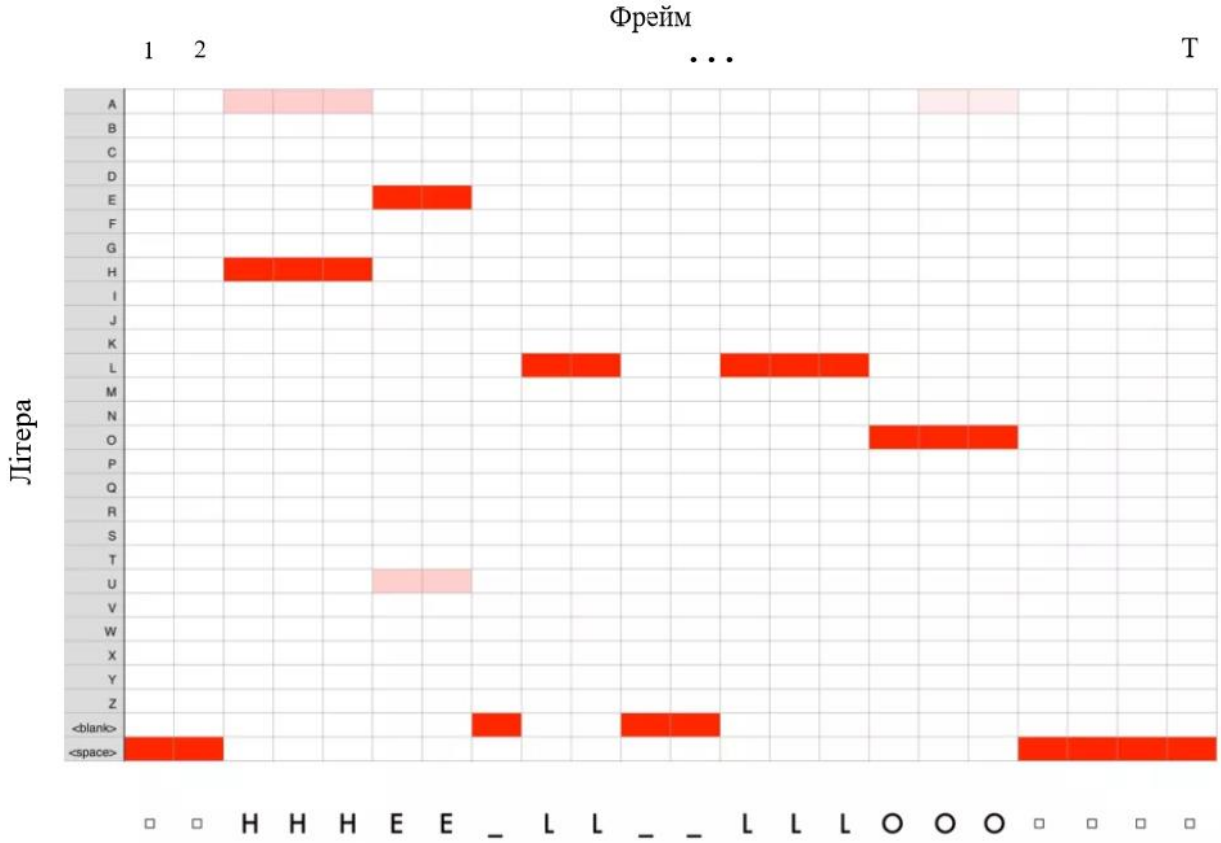

Рис. 2.9. Схема маркування послідовності фреймів

Сам алгоритм достатньо простий для розуміння та застосування. End-to-end ASR на основі CTC концептуально більш витончена та проста ніж традиційні ASR, але, при роботі з глибокими нейронними мережами, дослідники очікувано стикаються з тим, що інтерпретація навчених моделей є далеко не очевидною. Проте, це не заважає успішному використанню end-to-end систем для розпізнавання мови. CTC, як правило, передбачає послідовність літер або звуків, хоча спочатку були й спроби безпосередньо передбачати слова. За наявності

значної кількості навчальних даних вони показують хороші результати та перевершують точність традиційних та гібридних ASR.

2.7. Методи просодичної модифікації

2.7.1 Алгоритм PSOLA

Алгоритм PSOLA (Pitch – Synchronous OverLap and Add) призначений для зміщення частоти основного тону сигналу [12]. Це потенційно може зробити звук вхідного голосу вище або нижче. Технологія PSOLA може бути застосована до зміни просодії мовного сигналу. Алгоритм PSOLA використовується у синтезі мови. Він належить до алгоритмів зміщення частоти основного тону і представляє собою композицію більш простих методів модифікації мовного сигналу (розтягнення у часі та повторної дискретизації).

Існує велика кількість класичних алгоритмів оцінки частоти основного тону (Pitch Detection Algorithms, PDA) мовного сигналу та їх модифікацій [13]. Вибір певного алгоритму для оцінки частоти основного тону залежить від цільового використання і завжди являє собою певний компроміс між частотно-часовим розширенням, стійкістю до помилок, алгоритмічною затримкою і обчислювальною складністю.

Якщо допустити, що аналізований сигнал є строго періодичним, то частота основного тону може бути визначена як величина зворотна до довжини його періоду. Період в свою чергу визначається як мінімальний часовий зсув, який зберігає вихідний сигнал. Майже всі сигнали, з обробкою яких зустрічаються в практичних додатках, є не строго періодичними, а квазіперіодичними.

В обробці мовного сигналу зазвичай вважають, що частота основного тону відповідає частоті коливань голосових зв'язок. Передбачається, що хоча їх коливання і є квазіперіодичними, то, у всякому разі, на деякому нетривалому часовому інтервалі можна спостерігати майже повторювані фрагменти.

Алгоритм AMDF (Average Mean Difference Function) [12][13] визначають як вдосконалений вид автокореційної функції (ACF). Він вимагає попередньої обробки мовного сигналу, поділення сигналу на послідовність фреймів та проведення віконного перетворення над кожним фреймом.

У загальному виді AMDF функція виглядає наступним чином:

$$F_{0,i} = \left(\frac{argmin_{\,t}\,(D(l))}{f_s}\right)^{-1}, \tag{2.25}$$

де i – номер фрейму; $F_{0,i}$ – оцінка частоти основного тону i – го фрейму; f_s – частота дискретизації сигналу.

Різницева функція (DF) алгоритму AMDF визначається наступною формулою:

$$D(l) = \frac{1}{N-1-l}\sum_{t=0}^{N-1-l}|x(t) - x(t+l)|^j, \tag{2.26}$$

де N – довжина фрейму; l – час затримки у секундах $0 \leq l \leq$ N– 1; $x(t)$ – значення сигналу у дискретний момент часу t; t – порядок функції [18].

Однак, AMDF схильний до двох помилок – це спадаюча тенденція та подвійний пік, тому оцінки отримані за допомогою AMDF не є кращими.

Проте, як варіація AMDF, був запропонований алгоритм CAMDF (Circular Average Mean Difference Function), що в достатній мірі вирішує ці проблеми і значно покращує результати. Різницева функція алгоритму CAMDF дещо відрізняється визначається наступним виразом:

$$D(l) = \sum_{t=0}^{N-1}|x(mod(t+l, N)) - x(t)|. \tag{2.27}$$

Слід відзначити певні практичні моменти щодо вибору параметру l. По перше, на практиці значення затримки l може бути виражено не у секундах, а у кількості точок даних. У такому випадку отримані за формулою значення вже

будуть визначені в одиницях частоти і немає необхідності масштабувати їх до частоти дискретизації сигналу f_s. По друге, зауважимо, що при застосуванні алгоритмів оцінки частоти основного тону (ЧОТ) на мовному сигналі розглядають лише певний діапазон значень параметру l, що властивий саме ЧОТ голосу людини при вимові. З огляду на те, що цей діапазон є відносно вузьким, це може суттєво вплинути на часову ефективність обробки даних алгоритмом.

Задля збереження індивідуальності голосу, потребується змінювати частоту основного тону без зміни формантних частот і, таким чином, ідентичності голосних. Такий результат можна отримати при застосуванні однієї з варіацій алгоритму PSOLA [14]. Цей алгоритм, що застосовується у часовій області представлення сигналу.

Основна ідея алгоритму полягає розтягненні часу на ЧОТ-мітках, при цьому форма звукової хвилі сегмента не змінюється. Відстань ЧОТ-мітки визначає період ЧОТ мовлення і, таким чином, має бути відповідним чином змінений. Цей метод є подвійною операцією для повторної дискретизації сигналу у часовій області, але в цьому випадку здійснюється повторна дискретизація короткочасної спектральної оболонки. Короткочасна спектральна оболонка описує частотну криву, що проходить через всі амплітуди гармонік. Гармоніки знову масштабуються відповідно до $f_i^{new} = \beta\, f_i^{old}$,але амплітуди гармонік $a_i^{new} = env(f_i^{new}) \neq a_i^{old}$ визначаються шляхом дискретизації спектральної оболонки. Деякі відхилення амплітуд від точної оболонки можна помітити.

Коли ми прагнемо зміщення ЧОТ за коефіцієнтом β, який визначається як відношення ЧОТ локального синтезу до оригінальної ЧОТ

$$\beta = \frac{\tilde{f}_0(\tilde{t})}{f_0(t)}, \tag{2.28}$$

новий період ЧОТ буде задаватися рівнянням

$$\tilde{P}(\tilde{t}) = \frac{P(t)}{\beta}, \tag{2.29}$$

де $\tilde{t} = t$, оскільки час не розтягується.

Алгоритм PSOLA складається з двох фаз – аналізу та синтезу, які схематично зображені на рис. 2.1-2.2.

На фазі аналізу встановлюються ЧОТ-мітки, сигнал поділяється на фрейми з перекриттям, для кожного з яких застосовується віконне перетворення.

Алгоритм аналізу:

1. Визначення періоду ЧОТ $P(t)$ вхідного сигналу та моментів часу t_i. На практиці t_i встановлюються на відстані $P(t)$, тобто

$$P(t) = P(t_i) = t_{i+1} - t_i. \tag{2.30}$$

2. Вилучення кожного сегменту центрованого на ЧОТ-мітці t_i із застосуванням вікна Гемінга та довжиною $L_i = 2P(t_i)$ для забезпечення подальшого плавного перекриття сегментів.

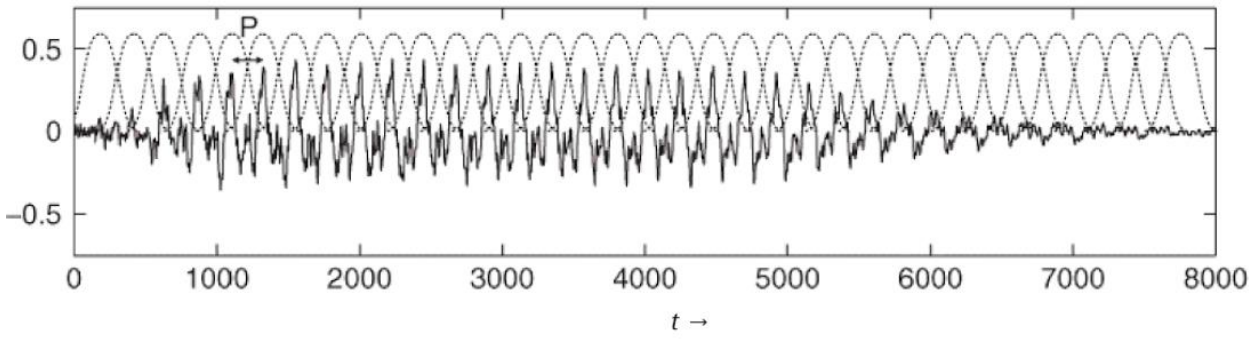

Рис. 2.10. Мовний сигнал на фазі аналізу

На фазі синтезу з фреймів формують сигнал первинної довжини, перекриваючи та додаючи фрейм за фреймом, таким чином відстань між ЧОТ-мітками, а отже і частота основного тону сигналу змінюється.

Алгоритм синтезу:

– для кожної синтезованої ЧОТ-мітки $\tilde{t}_k$:

1. Вибір відповідного i-го сегменту для аналізу (ідентифікується міткою часу t_i), що мінімізує часову відстань $|t_i - \tilde{t}_k|$.

2. Перекриття та додавання обраного сегменту. При цьому деякі вхідні сегменти будуть повторені при $\beta > 1$ (підвищення ЧОТ) або вилучені при $\beta < 1$ (зниження ЧОТ).

3. Визначення моменту часу $\tilde{t}_{k+1}$, на якому буде центровано наступний синтезований сегмент, для збереження локальної ЧОТ, за відношенням

$$\tilde{t}_{k+1} = \tilde{t}_k + \tilde{P}(\tilde{t}_k) = \tilde{t}_k + \frac{P(t_i)}{\beta}. \quad (2.30)$$

Для великих зсувів ЧОТ, доцільно компенсувати варіації амплітуди, що представлені більшим чи меншими перекриттями сегментів, множенням вихідного сигналу на $1/\beta$.

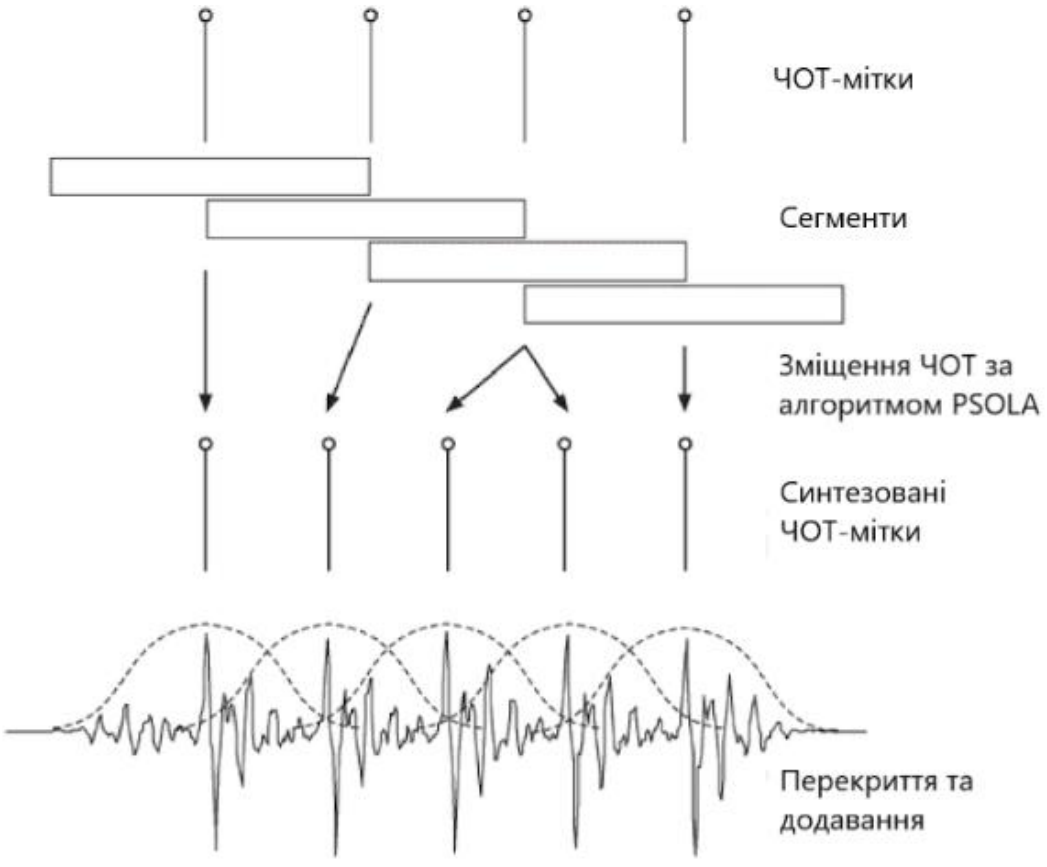

Рис. 2.11. Алгоритм синтезу

Можна поєднати розтягування часу за фактором α із зміною тону. У цьому випадку для кожної ЧОТ-мітки синтезу $\tilde{t}_k$, перший крок алгоритму синтезу, представлений вище, буде модифіковано як вибір відповідного i-го сегменту аналізу (ідентифікується міткою часу t_i), що мінімізує часову відстань $|\alpha t_i - \tilde{t}_k|$.

Внаслідок перекриття та додавання ЧОТ змінюється, при цьому вихідний сигнал залишатиметься такої ж довжини, що і вхідний.

Алгоритм PSOLA є ефективною технікою для зміщення частоти основного тону, має обґрунтовану якість та надає можливості для зміни мовного сигналу. Але оскільки метод PSOLA ґрунтується на припущеннях, які не є суто істинними, його застосування має деякий вплив на спектр і також може погіршити якість сигналу.

2.7.2. Нормалізація енергії

Нормалізація динамічного діапазону логарифму енергії або нормалізація енергії (ERN) [15] зменшує розподіл енергії, обумовлений різними рівнями фонових шумів. Цей метод базується на тому, що однаковий рівень шуму призводить до невеликих змін логарифмів енергії у сегментах високої енергії, проте веде до різких змінах на сегментах низької енергії. Припускається, що всі висловлювання всіх ораторів мають таку ж максимальну енергію, як і динамічний діапазон. При цьому припущенні цільова мінімальна логарифмічна енергія встановлюється на підставі оціночної максимальної енергії висловлювання та припущенного динамічного діапазону [16]. Якщо мінімальна логарифмічна енергія в одному висловлюванні менша, ніж обчислена мінімальна логарифмічна енергія, енергія повинна бути перевизначена до певного цільового діапазону.

Короткострокова енергія сигналу (STE, енергія) [14] обчислюється за виразом

$$STE = \frac{1}{N}\sum_{t=0}^{N-1}|x(t)|^2, \qquad (2.31)$$

де N – довжина фрейму; $x(t)$ – значення амплітуди сигналу у момент часу t.

Динамічний діапазон послідовності ознак логарифмічної енергії визначається наступним чином:

$$D.R.(dB) = 10 \times \frac{Max(Log(STE_i)_{i=1\dots n})}{Min(Log(STE_i)_{i=1\dots n})}, \quad (2.32)$$

де $Max(Log(STE_i)_{i=1\dots n})$ – це максимальне значення послідовності ознак логарифмічної енергії, $Min(Log(STE_i)_{i=1\dots n})$ – мінімальне, а n – загальна кількість фреймів у висловлюванні. Припустимо, що $Max(Log(STE_i)_{i=1\dots n})$ є однаковим для первинного і цільового динамічного діапазону. Цільове мінімальне значення T_Min може бути обчислене за вищеподаним рівнянням при заданому цільовому динамічному діапазоні [17][18].

Алгоритм нормалізації логарифму енергії:

(1) Для кожного висловлювання чистої мови, знайти $Max = Max(Log(STE_i)_{i=1\dots n})$ – максимальне значення послідовності ознак логарифмічної енергії та $Min = Min(Log(STE_i)_{i=1\dots n})$ – мінімальне, де n – загальна кількість фреймів у висловлюванні.

(2) Обчислити $T\_Min = \alpha \times Max(Log(STE_i)_{i=1\dots n})$.

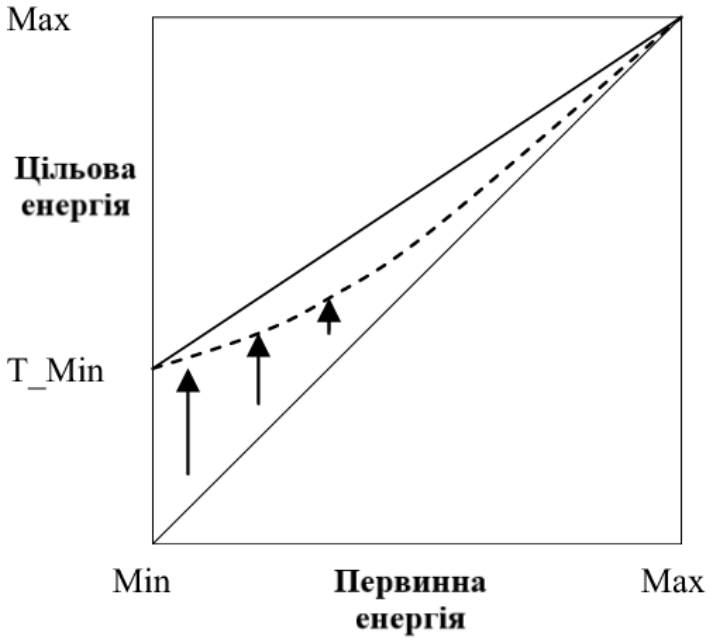

Рис. 2.12. Схематичне представлення ефекту масштабування ERN алгоритму

(3) Якщо $Min < T\_Min$, то значення ознаки логарифмічної енергії $Log(Energy_i)$ для кожного i – го фрейму обчислюється за формулою:

$$Log(STE_i) = Log(STE_i) + \frac{T\_Min - Min}{Max - Min} \times \left(Max - Log(STE_i)\right). \quad (2.33)$$

Наведений алгоритм масштабування мінімуму може бути застосований і для масштабування максимального значення енергії у висловлюванні.

Для виконання модифікації мовного сигналу на основі нормалізованого значення $Log(STE_i)$ отримане для фрейму значення перетворюються у нове значення енергії фрейму STE_i та впроваджується пропорційна модифікація значень амплітуди фрейму.

РОЗДІЛ 3
ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ПРОСОДИЧНОЇ МОДИФІКАЦІЇ МОВНОГО СИГНАЛУ

3.1. Алгоритм на основі методів просодичної модифікації

Визначимо схему навчання алгоритму перетворення на основі просодичної модифікації можна визначити наступнім чином:

1) встановлення параметрів перетворення – визначення діапазону значень емоційних ознак (частота основного тону та енергія), до якого мають бути приведені усі значення фрагменту мови;

2) перетворення мови – застосування методу або комбінації методів перетворення (PSOLA та ERN);

3) розпізнавання мови – отримання якості розпізнавання оригінального та перетвореного висловлювання за допомогою алгоритму розпізнавання СТС;

4) порівняння якості розпізнавання для встановлення напрямку подальшого пошуку кращого перетворення;

5) повторення кроків 1-4 до досягнення критерію максимального покращення якості.

Слід зауважити, що передбачається, що саме перетворення мови на нейтральну за інтонацією підвищить якість розпізнавання, проте не відкидається, що цільове перетворення може бути і іншим. При цьому натуральність та чистота звучання мови не постає пріоритетною, адже не представляє практичного інтересу для подальшого використання.

3.1.1. Алгоритм PSOLA

Розглянемо детальніше алгоритм PSOLA на прикладі діаграми діяльності алгоритму (рис. 3.1).

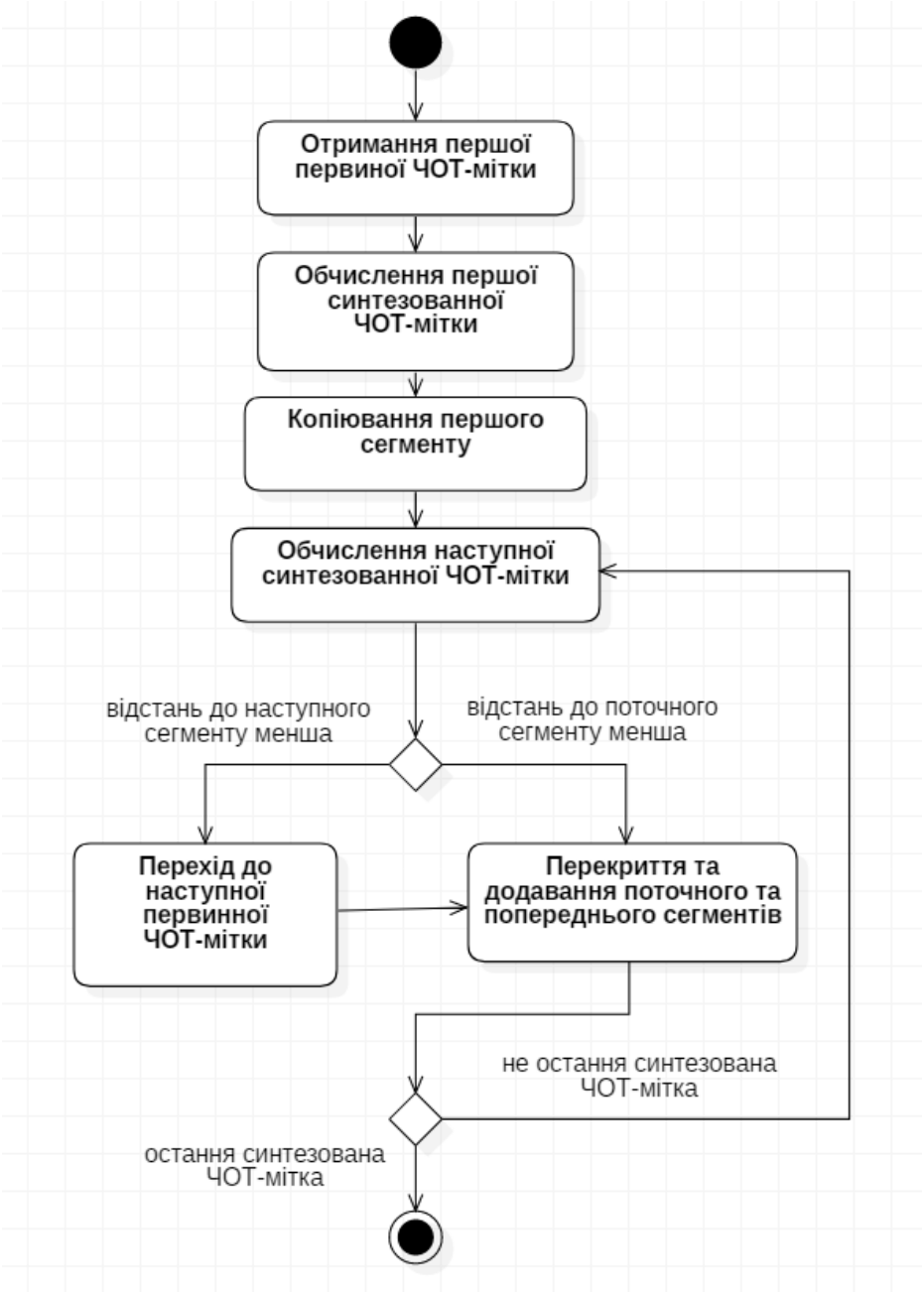

Рис. 3.1. Діаграма діяльності алгоритму PSOLA

Зміщення ЧОТ дозволяє змінювати висоту голосу. З огляду на те, що мовний сигнал не є стаціонарним і змінює ЧОТ впродовж часу, зміщення ЧОТ виконується для окремих фреймів сигналу. Отже, вхідними даними для алгоритму є дані фрейму (сегменту), що обробляється.

До параметрів алгоритму PSOLA відноситься первина ЧОТ фрейму та цільова ЧОТ, що має бути синтезована для фрейму. На основі даних та параметрів впроваджується перетворення та на виході алгоритму формується модифікований мовний сигнал, що має первинну довжину, але іншу ЧОТ.

3.1.2. Алгоритм нормалізації енергії ERN

Розглянемо детальніше алгоритм нормалізації динамічного діапазону енергії на прикладі діаграми діяльності алгоритму (рис. 3.2).

Нормалізація енергії має вплив на гучність голосу та вирівнює варіації гучності впродовж усього висловлювання. Так само як і PSOLA, алгоритм ERN працює з фреймами (сегментами), проте він потребує попереднього обчислення енергії для усіх сегментів, тому на вхід алгоритму надходитимуть дані не сегменту, а усього висловлювання, тобто набір сегментів.

Алгоритм потребує завдання параметру масштабування α. Значення мінімуму (Min), максимуму (Max) та цільового мінімуму (T_Min) логарифму енергії обчислюються під час роботи алгоритму. На основі даних та параметрів впроваджується перетворення та на виході алгоритму формується модифікований мовний сигнал, що має інший мінімум логарифму енергії. Очевидно, що для модифікованого сигналу слід провести операцію зворотну до логарифму.

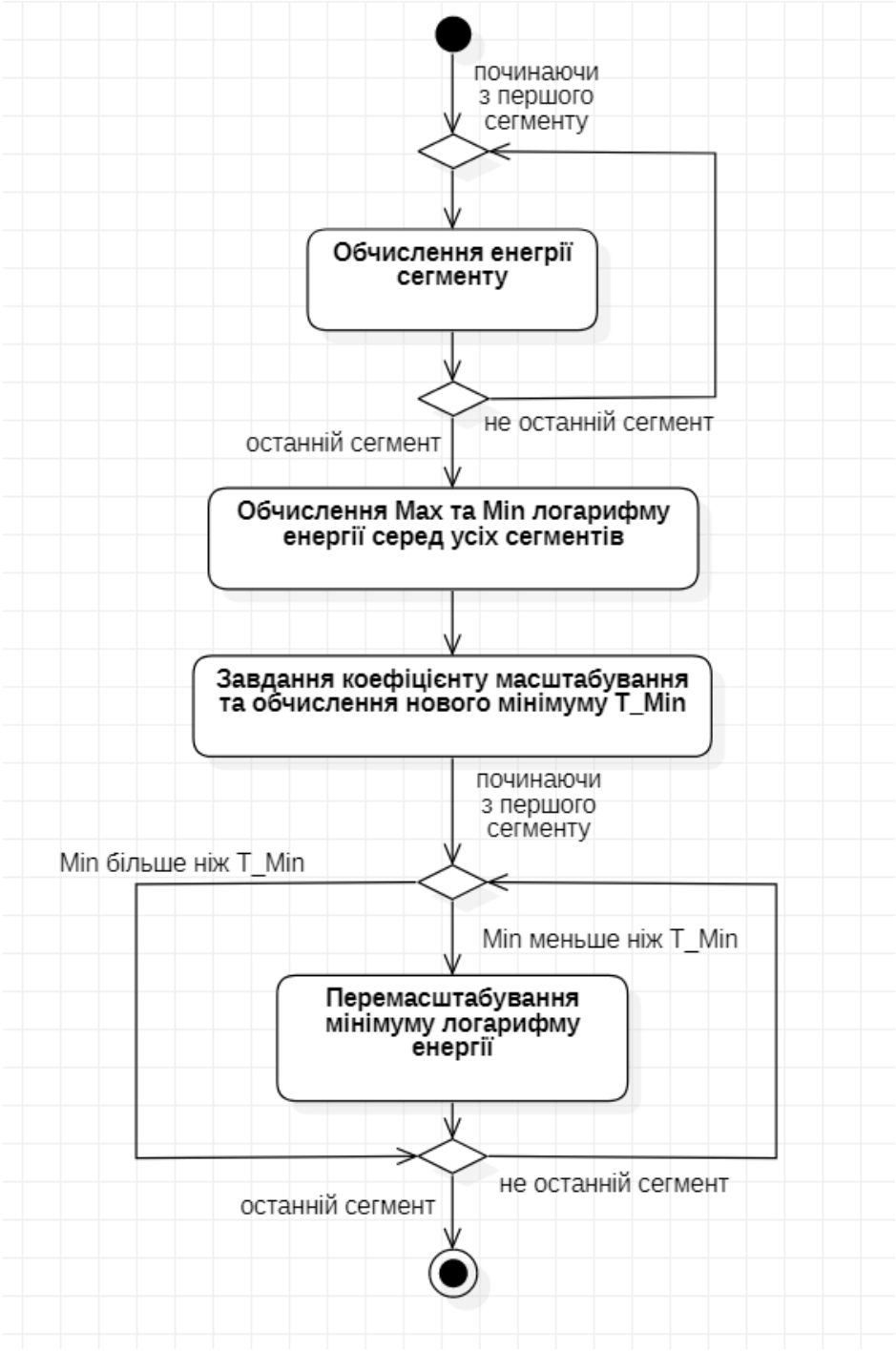

Рис. 3.2. Діаграма діяльності алгоритму ERN

3.2. Результати просодичної модифікації даних за допомогою алгоритмів PSOLA та ERN.

Вхідними даними є аудіо-файли – записи висловлювань у форматі wav. Отриманий аудіо файл декодується до послідовності значень імпульсно-кодової модуляції (PCM). Ці значення відповідають значенням амплітуди сигналу у часовому просторі, але представлені у більш зручному форматі для зберігання і відтворення звуку.

Важливою характеристикою аудіо для обробки PCM даних є частота дискретизації – кількість значень амплітуди сигналу, що кодуються за одну секунду. Так, наприклад, частота дискретизації 44100 Гц означає, що 441 значення з послідовності PCM даних буде відповідати 10 мс мовного сигналу при відтворенні аудіо файлу.

Мовні сигнали як правило проходять попередню обробку перед обчисленням ЧОТ та енергії задля підвищення точності та ефективності процесу вилучення. Для ЧОТ обробка складається з поділу на фрейми та застосування віконного перетворення, для енергії виконується лише поділ на фрейми. Поділ на фрейми перетворює масив даних мовного сигналу у набір фреймів, що аналізуються незалежно. Важливо обрати оптимальний розмір для кожного фрейму. Якщо розмір фрейму занадто великий, то неможливо вилучити характеристики аудіо сигналу зі збереженням залежності від часу. Якщо фрейм замалий, то неможливо вилучити вагомі акустичні ознаки. Зазвичай розмір фрейму дорівнює степеню двійки (256, 512, 1024), тому що такий розмір підходить для виконання швидкого перетворення Фур'є. Як правило використовують фрейм довжиною 10-30 мс, тобто 512 та 1024 точок даних. Вікно – вагова функція, яка застосовується над фреймом для зменшення витоку спектра в сигналі вхідних даних. найпоширенішими у застосуванні є віконні функції Гемінга.

Скріншоти роботи алгоритмів PSOLA та ERN на рис. 3.3-3.14. Для дослідження використано набір даних награної (акторської) емоційної мови – *Toronto Emotional Speech Set (TESS)*, розділ *YAF (Young Actor, Female)*.

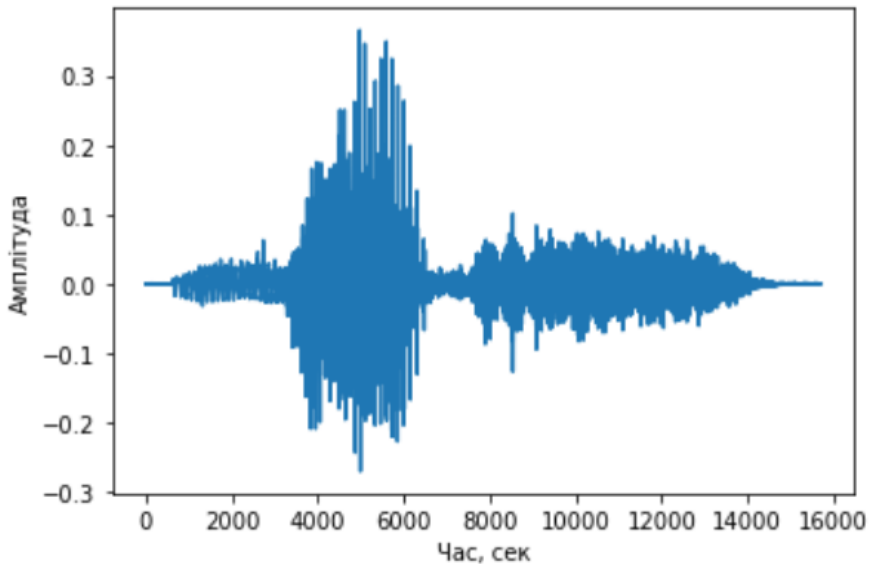

Рис. 3.3. Звукова хвиля слова «yes» з емоцією «злість»

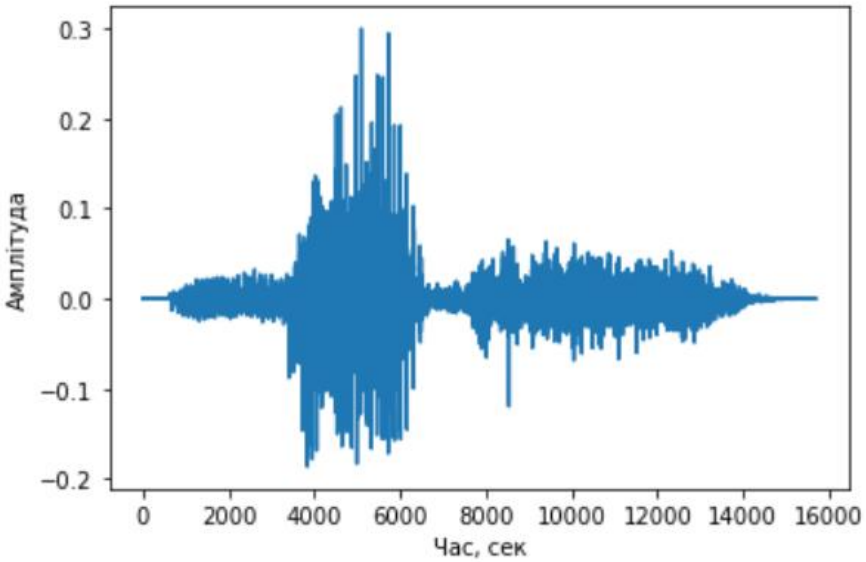

Рис. 3.4. Звукова хвиля слова «yes» з емоцією «злість» після обробки алгоритмом PSOLA

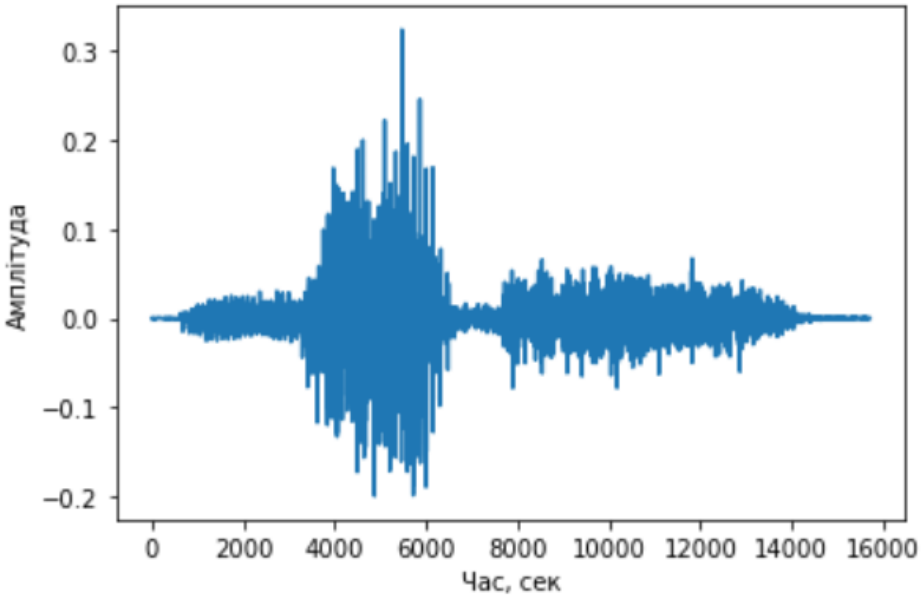

Рис. 3.5. Звукова хвиля слова «yes» з емоцією «злість» після обробки алгоритмом ERN

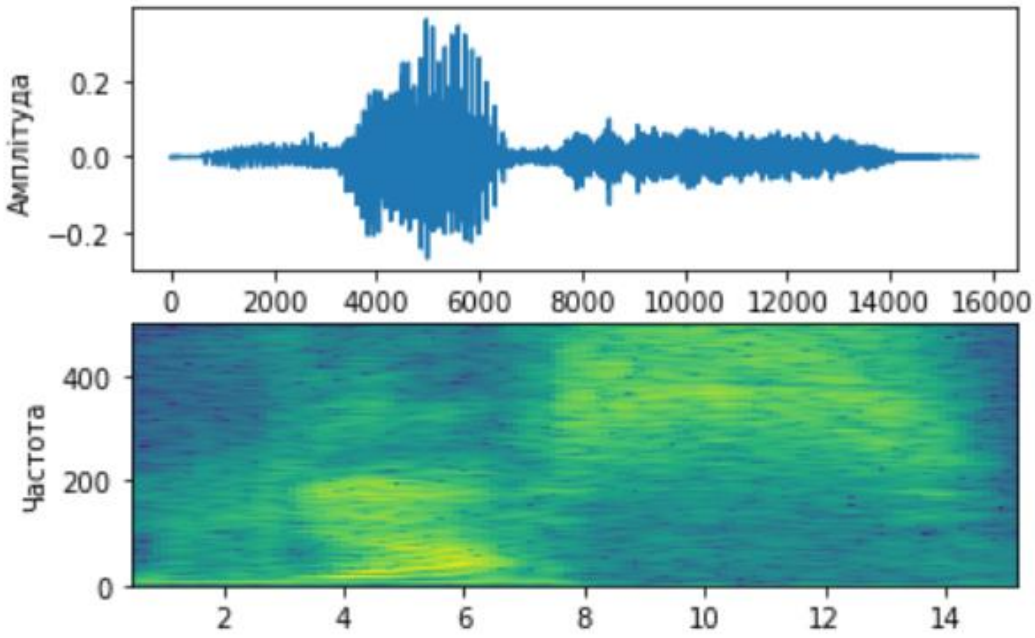

Рис. 3.6. Спектрограма слова «yes» з емоцією «злість»

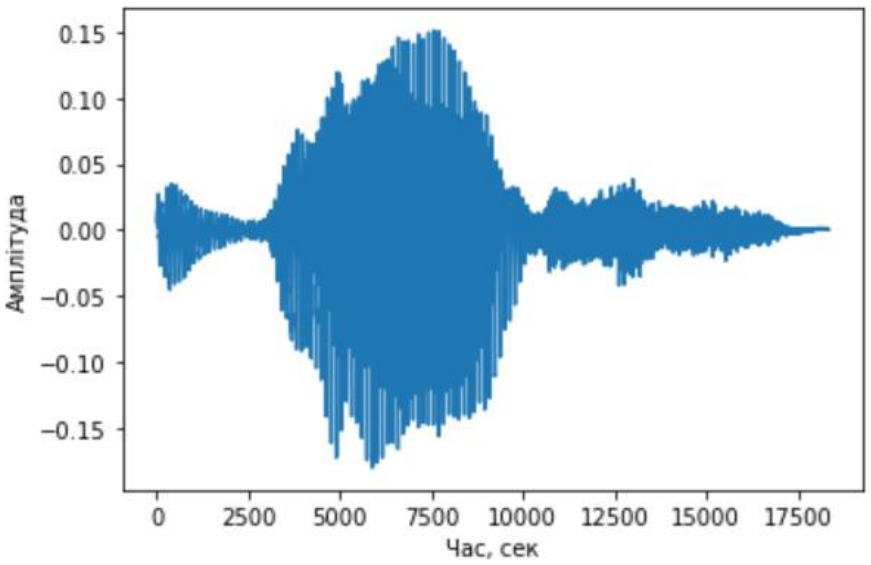

Рис. 3.7. Звукова хвиля слова «yes» з емоцією «нейтральний»

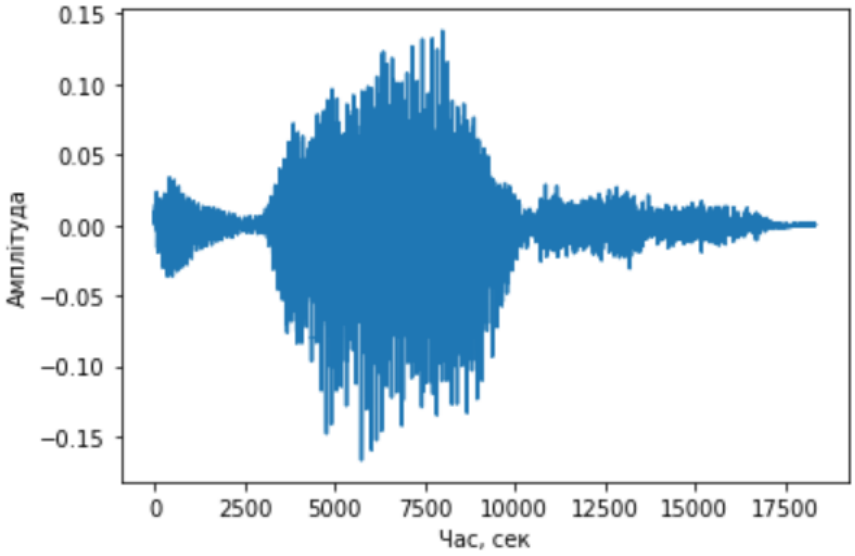

Рис. 3.8. Звукова хвиля слова «yes» з емоцією «нейтральний» після обробки алгоритмом PSOLA

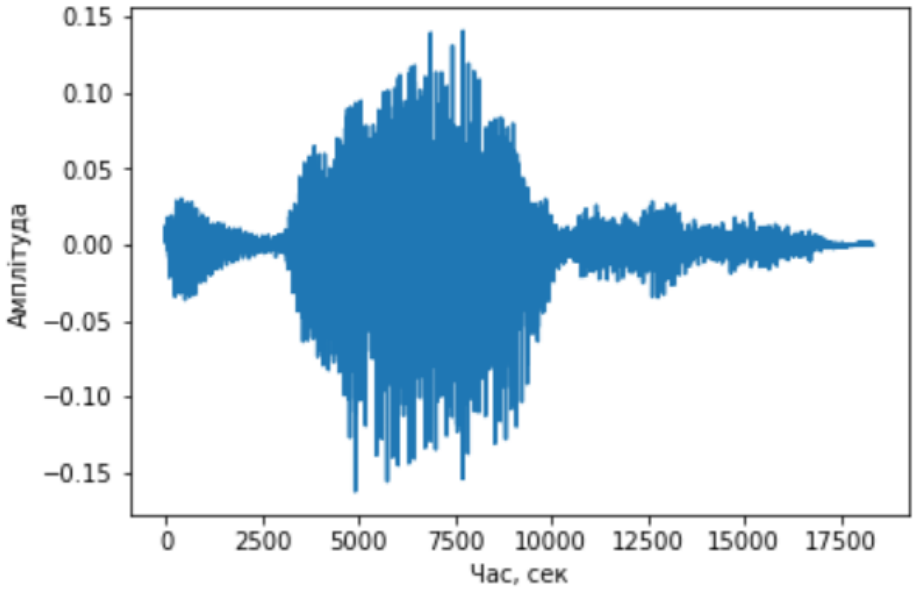

Рис. 3.9. Звукова хвиля слова «yes» з емоцією «нейтральний» після обробки алгоритмом ERN

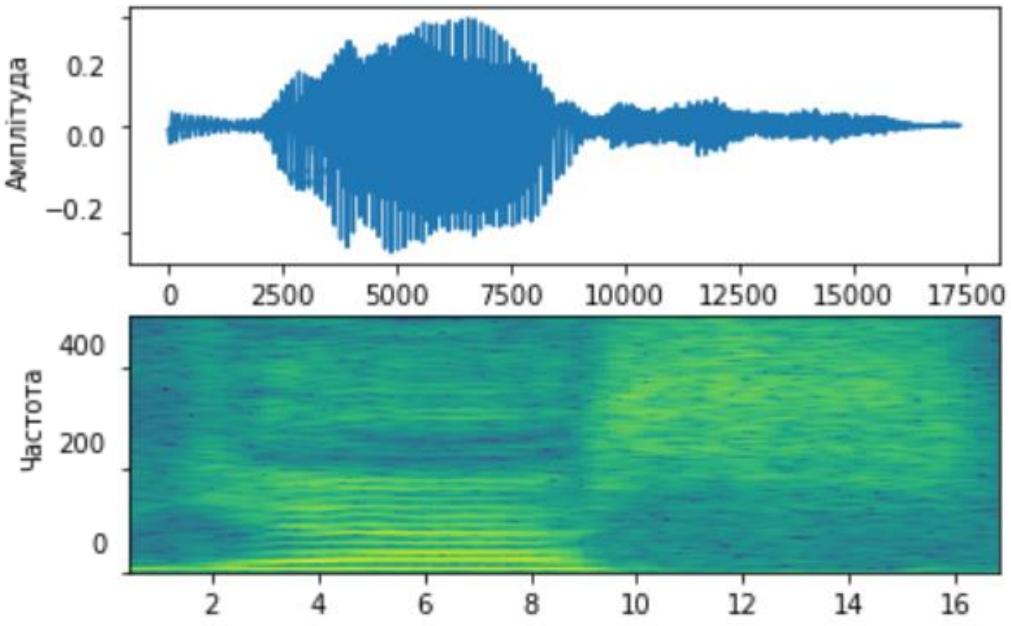

Рис. 3.10. Спектрограма слова «yes» з емоцією «нейтральний»

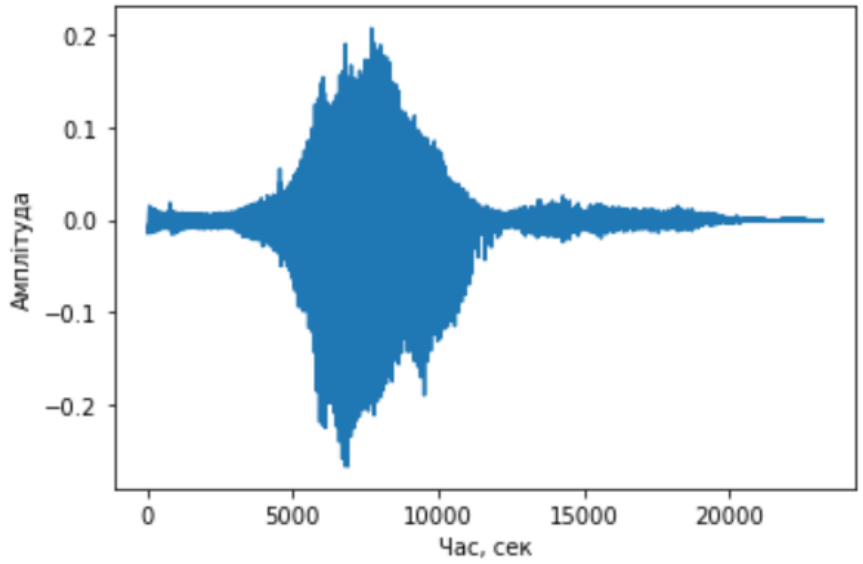

Рис. 3.11. Звукова хвиля слова «yes» з емоцією «щастя»

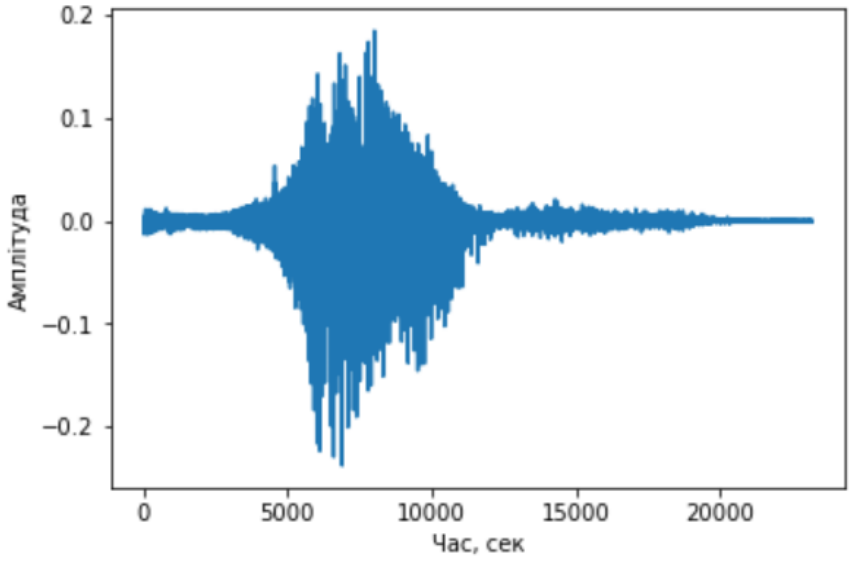

Рис. 3.12. Звукова хвиля слова «yes» з емоцією «щастя» після обробки алгоритмом PSOLA

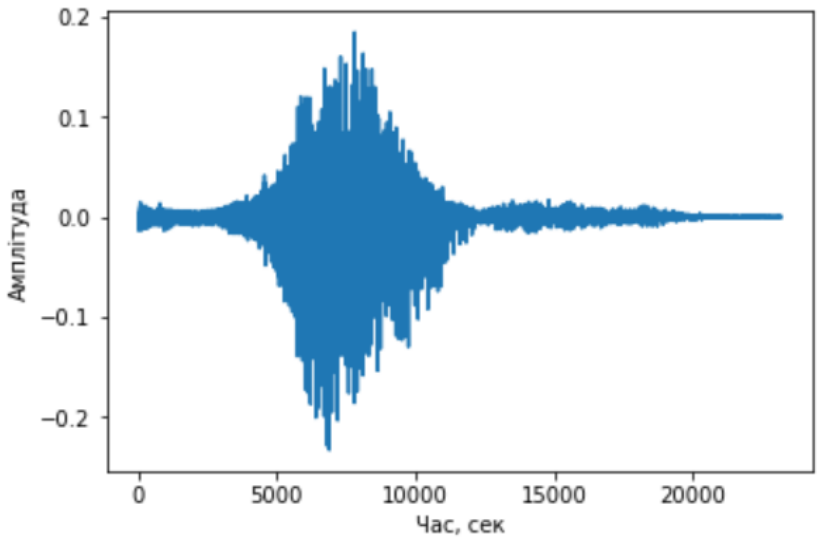

Рис. 3.13. Звукова хвиля слова «yes» з емоцією «щастя» після обробки алгоритмом ERN

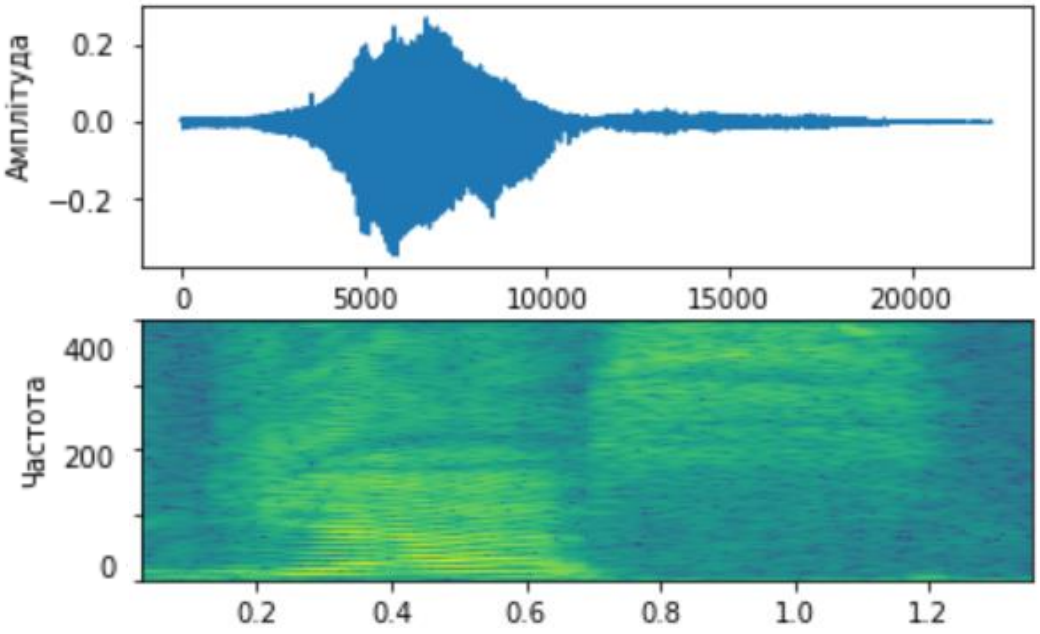

Рис. 3.14. Спектрограма слова «yes» з емоцією «щастя»

На базі проведенного аналізу та отриманих від користувача параметрів встановлюються межі на рівень перетворень і виконується просодична модифікація даних за допомогою алгоритмів PSOLA та ERN.

3.3 Інтерпретація впливу алгоритму на розпізнавання мови

Для представлення та аналізу результатів дослідження частота помилкового слова (WER, word error rate), використовується як метрика якості розпізнавання. У рамках дослідження використано набір даних награної (акторської) емоційної мови – *Toronto Emotional Speech Set (TESS)*, розділ *YAF (Young Actor, Female)*.

Таблиця 3.1

Порівняльна характеристика впливу розробленого алгоритму на якість розпізнавання мовного сигналу, відносно метрики WER (%)

Емоція	Без обробки	*PSOLA*	*ERN*	*PSOLA + ERN*
Злість	31.76	26.79	22.22	24.52
Відраза	25.64	23.09	24.79	25.22
Страх	18.87	18.74	20.85	19.34
Сум	22.56	23.40	20.94	19.80
Нейтральний	16.27	16.38	16.14	16.82
Приємний подив	26.56	27.37	24.65	25.08
Щастя	25.64	23.09	24.43	21.37

Спираючись на отримані результати (табл. 3.1), можна визначити, що алгоритми просодичної модифікації PSOLA та ERN мають достатній вплив на розпізнавання мови відносно метрики WER. Проте помічено, що модифікація не завжди має бажаний результат (покращення якості). Незважаючи на це, слід визначити, що, за умов проведення більш детального дослідження процессу обрання параметрів для просодичної модифікації, запропонований алгоритм має перспективи у контексті задачі розпізнавання мовного синалу.

РОЗДІЛ 4
ЕКОНОМІЧНИЙ РОЗДІЛ

4.1 Розрахунок трудомісткості і вартості розробки програмного продукту

Вхідні дані:

- ✓ передбачуване число операторів - 2100
- ✓ коефіцієнт складності програми - 1,6
- ✓ коефіцієнт корекції програми в ході її розробки - 0,07
- ✓ часовая заработная плата программиста, грн/г - 50,0

У процесі створення ПЗ нормування праці ускладнено в силу творчого характеру праці програміста. Тому, трудомісткість розробки ПЗ розраховується на основі системи моделей з різною точністю оцінки.

$$t = t_u + t_a + t_n + t_{отл} + t_д, \text{ чол.-г} \qquad (4.1)$$

де t_u – затрати праці на дослідження алгоритму розв'язання задачі, чол.-г;

t_a – затрати праці на розробку блок-схеми алгоритму, чол.-г;

t_n – затрати праці на програмування по готовій блок–схемі, чол.-г;

$t_{отл}$ – затрати праці на налагодження програми на ЕОМ, чол.-г;

$t_д$ – затрати праці на підготовку документації по завданню, чол.-г.

Ці затрати праці визначаються через умовне число операторів при розробці ПЗ, в число яких входять ті оператори , які необхідно написати в процесі роботи над програмою з урахуванням можливих уточнень у постановці завдання і вдосконалення алгоритму .

Умовне число операторів в програмі обчислюється за формулою:

$$Q = qC(1 + p), \qquad (4.2)$$

де q - передбачуване число операторів q = 2100 ;

c - коефіцієнт складності програми c = 1,6;

p - коефіцієнт кореляції програми в ході ее розробки p = 0,07.

Q = 2100 * 1,6 (1+ 0,07) = 3595

Затрати праці на вивчення опису завдання $t_и$ визначається з урахуванням уточнення опису та кваліфікації програміста.

$$t_u = \frac{QB}{(75..85)K} = \frac{3595*1,3}{77*1,2} = 50,58 \quad \text{чол.-год.,} \qquad (4.3)$$

де В - коефіцієнт збільшення затрат праці внаслідок недостатнього опису завдання:

$$B = 1,2. . . . 1,5;$$

К – коефіцієнт кваліфікації програміста, який визначається залежно від стажу роботи за даною спеціальністю. Він становить при стажі роботи, роки:

до 2 – 0,8;

від 2 до 3 – 1,0;

від 3 до 5 - 1,1 . . . 1,2;

від 5 до 7 - 1,3 . . . 1,4;

вище 7 – 1,5 . . . 1,6.

Затрати праці на розробку алгоритма для рішення задачі:

$$t_a = \frac{Q}{(20...25)K} = \frac{3595}{22*1,2} = 136,17 \quad \text{чол.-год.} \qquad (4.4)$$

Витрати на складання програми по готовій блок -схемі:

$$t_n = \frac{Q}{(20..25)K} = \frac{3595}{22*1,2} = 136,17 \quad \text{чол.-год.} \qquad (4.5)$$

Затрати праці на налагодження програми на ЕОМ:

$$t_{отл} = \frac{Q}{(4..5)K} = \frac{3595}{4*1,2} = 748,96 \text{ чол.-год.} \qquad (4.6)$$

$$t_{отл}^{K} = 1,5t_{отл} = 1,5*748,96 = 1123,44 \text{ чол.-год.} \qquad (4.7)$$

Витрати на підготовку документації:

$$t_д = t_{др} + t_{до} \quad \text{чол.-год.,} \qquad (4.8)$$

де $t_{др}$ – трудомісткість підготовки матеріалів:

$$t_{др} = \frac{Q}{(15...20)K} = \frac{3595}{17*1,2} = 206,61 \text{ чол.-год.} \quad (4.9)$$

$t_{до}$ - трудомісткість редагування, друку та оформлення документації:

$$t_{до} = 0,75 t_{др} = 0,75 * 206,61 = 154,96 \text{ чол.-год.} \quad (4.10)$$

$t_д = t_{др} + t_{до} = 206,61 + 154,96 = 361,57$ чол.-год.

У підсумку отримуємо, що трудомісткість розробки ПЗ становить:

t = 50,58+136,17+136,17+1123,44+361,57 = 1807,93 чол.-год.

4.2 Затрати на створення програмного забезпечення

Витрати на створення ПО (Кпо) включають витрати на заробітну плату виконавців програми (Зз/п), визначену множенням сумарної трудомісткості розробки ПО (t) на середню зарплату з нарахуваннями програміста і вартості машинного часу, необхідного для відладки програми на ЕОМ (Змв), визначеною виходячи з вартості 1-го машинного години конкретного типу ЕОМ, і витрат машинного часу на налагодження.

$$К_{ПО} = З_{ЗП} + З_{МВ} \text{ , грн.} \quad (4.11)$$

Заробітна плата виконавців визначається за формулою:

$$З_{ЗП} = t * С_{ПР} = 1807,93 * 50,0 = 90396,5 \text{ грн.,} \quad (4.12)$$

де t - загальна трудомісткість, чол.-г.;

$С_{пр}$ - середня годинна заробітна плата програміста, грн./год;

$С_{пр}$ =50,0 грн./год.

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{мв} = t_{отл} * С_{мч} = 1123,44 * 1,2 = 1348,13 \text{ грн.,} \quad (4.13)$$

де $t_{отл}$ - трудомісткість налагодження програми на ЕОМ, год.;

$С_{мч}$ – вартість машинного часу ЕОМ, грн./год.

Кпо = 90396,5 + 1348,13 = 91744,63 грн. (4.14)

Визначені таким чином витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення АСУТП . Очікуваний період розробки ПЗ:

$$T = \frac{t}{B_k \bullet F_p} \quad міс., \qquad (4.15)$$

де B_k - кількість виконавців;

F_p - місячний фонд робочого часу (при 40-ка годинному робочому тиждні F_p=176 годин).

$$T = \frac{1807,93}{1 * 176} = 10,27 \approx 10 \text{ мес.}$$

Таким чином, період розробки програми складе приблизно 10 місяців.

методу.

4.3 Маркетингові дослідження ринку

Перші дослідження зв’язку між емоціями та мовленням людини було проведено ще у XIX столітті. Інтерес до теми проявляли природознавці у наукових та філософських роботах. Так Г. Спенсер детально описав зміни голосу під впливом різних емоцій у філософському есе про походження музики. Ч. Дарвін у своїх роботах наводив спостереження щодо схожості вираження емоції у людини та тварин, але підкреслював, що його дослідження не є достатніми, щоб повністю освітити тему передачі емоцій голосом людини. Деякі дослідники зверталися до цієї теми і впродовж XX століття, але не досягли значних успіхів. Попри те, що невербальна поведінка активно досліджувалася у психології, однозначних висновків та науково достовірних результатів отримано не було. Це в певній мірі пов’язують з недостатньою технічною базою для досліджень.

За останні кілька десятиліть просодична модифікація займає місце однієї із головних головних тем, що цікавлять дослідників у галузі обробки мовного сигналу. Просодична модифікація (prosodic modification або prosody modification) – це процес інтонаційного перетворення мовного сигналу, який впливає на частоту основного тону (ЧОТ) та довжину сигналу, але запобігає спектральній деформації, порушенню семантики повідомлення та зберігає натуральність звучання мови.

Чимало досліджень проведено щодо застосування просодичну модифікацію на основі методів зміщення ЧОТ у контексті синтезу мови (text-to-speech, TTS) для надання штучно сгенерованим висловлюванням природної та доцільної експресивності.

У роботі представлено методи просодичної модифікації для застосування у TTS системах, що працюють режимі реального часу. У таких системах край важливо мінімізувати обчислювальну складність і, таким чином, час обробки даних та швидкості відповіді системи на запит користувача. Саме тому було запропоновано виконувати пошук вокалізованих ділянок висловлювання та спиратися на модифікацію лише окремих частин цих ділянок. Таких підхід дозволив дослідникам зменшити обчислювальну складність на 75-90% від базового підходу до модифікації, проте було відзначено, що такий напрям досі є недостатньо вивченим для практичного застосування. «Швидкий» підхід, що базувався на модифікації частоти основного тону мовного сигналу, успішно застосовували і інші дослідники. Отримані результати відображали достатній рівень якості модифікованого сигналу, проте було підкреслено, що при значному зміщенні ЧОТ може виникнути необхідність використання додаткових алгоритмів нормалізації гучності звуку.

Пізніше дослідники почали акцентувати увагу на зв’язку між емоційністю мови та інтонацією, на зміну якої і націлені методи просодичної модифікації, для

впровадження нового підходу для оптимального пошуку та модифікації ділянок висловлювання.

Деякі з досліджень останніх років почали прицільно вивчати просодичну модифікацію зі зворотної сторони – галузі розпізнавання мови (speech-to-text), проте не позбавилися від контексту впливу емоцій на мовний сигнал.

У даній роботі просодична модифікація розглядається як перспективний підхід для попередньої обробки сигналу з метою покращення якості подальшого розпізнавання мовного сигналу; розглянуто алгоритми розпізнавання мови, зміщення ЧОТ та нормалізації енергії (гучності).

ВИСНОВКИ

Реалізація програмної частини виконана за допомогою мови програмування Python та графічного програмного середовища Jupyter Notebook, з використанням бібліотек: wave, numpy та matplolib.

Вхідними даними для програми є аудіо файл, що декодується до послідовності значень. Ці значення відповідають значенням амплітуди сигналу у часовому просторі, але представлені у більш зручному форматі для зберігання та відтворення звуку. Після проходження необхідної попередньої обробки проводиться аналіз сигналу, що полягає у вилученні інформативних ознак. На базі проведенного аналізу та отриманих від користувача параметрів встановлюються межі на рівень перетворень і виконується модифікація даних за алгоритмами PSOLA та ERN.

Алгоритм PSOLA дозволяє вирівняти фонеми, частота основного тону яких значно відрізняється. Проте може мати місце легке зашумлення. Алгоритм нормалізації енергії ERN дозволяє вирівняти гучність на протязі усього висловлювання. Алгоритм PSOLA та ERN можна віднести до надійних алгоритмів просодичної модифікаіції. Вони забеспечують перцептивно значиме зниження емоційності висловлювання, мають зрозумілий принцип роботи та очікуваний вплив на мовний сигнал при дотриманні певного діапазону змін.

Реалізоване програмне забезпечення та отримані вихідні дані дозволяють оцінити практичну цінність та результати дослідження.

У результаті проведеного дослідження було спроектовано алгоритм покращення якості розпізнавання мовного сигналу на основі методів просодичної модифікації – алгоритмів PSOLA та ERN.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Automatic speech recognition and speech variability: A review / M. Benzeghiba, R. De Mori, O. Deroo, S. Dupont, T. Erbes, D. Jouvet, L. Fissore, P. Laface, A. Mertins, C. Ris, R. Rose, V. Tyagi, C. Wellekens // Speech Communication, Elsevier. – 2007. – №49. – P. 763-786.

2. V.V.R. Vegesna. Prosody modification for speech recognition in emotionally mismatched conditions / V.V.R. Vegesna, K. Gurugubelli, A.K. Vuppala // International Journal of Speech Technology, 3. – 2018. – P. 521-532.

3. Harpreet Kaur. Prosody Modification of its Output Speech Signal / Harpreet Kaur, Parminder Singh // International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE). – 2014. – №5. – P. 1056–1059.

4. Krothapalli Sreenivasa Rao. Real Time Prosody Modification / Krothapalli Sreenivasa Rao // Journal of Signal and Information Processing. – 2010. – №1. – P. 50-62.

5. Fast Prosody Modification using Instants of Significant Excitation / S. R. M. Prasanna, D. Govind, K. S. Rao, B. Yegnanarayana // Speech Prosody. – 2010.

6. Synthesis of Emotional Speech by Prosody Modification of Vowel Segments of Neutral Speech / Md Shah Fahad, Shreya Singh, Shruti Gupta, Akshay Deepak and Abhinav // Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE). – 2019. – P. 49-54

7. D. Joshi. Speech Emotion Recognition: A Review / D. Joshi, M. B. Zalte // IOSR Journal of Electronics and Communication Engineering (IOSR – JECE). – 2013. – №4. – P. 34–37.

8. End-to-End Deep Neural Network for Automatic Speech Recognition / William Song, Jim Cai // Stanford University. – 2015.

9. Yonatan Belinkov. Analyzing Hidden Representations in End-to-End Automatic Speech Recognition Systems / Yonatan Belinkov and James Glass //

Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017). – 2017.

10. Speech Recognition and Deep Learning / Adam Coats, Vinay Rao // [Електронний ресурс] : [Інтернет-портал]. – Електронні дані. – Режим доступу:https://www.youtube.com/watch?v=9dXiAecyJrY&feature=youtu.be&t=13874 – Bay Area Deep Learning School Day 2 at CEMEX auditorium, Stanford.

11. Graves. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks / Graves, Alex // Proceedings of the 23rd international conference on Machine learning. ACM. – 2006. – P. 369–376.

12. Saurabh Padmawar. Classification Of Speech Using Mfcc And Power Spectrum / Saurabh Padmawar, P.S. Deshpande // International Journal of Engineering Research and Applications (IJERA). – 2013. – №1. – P. 1451–1454.

13. K. R. Anne. Acoustic Modeling for Emotion Recognition / K. R. Anne, S. Kuchibhotla, H. D. Vankayalapati. – Springer : Springer Briefs in Electrical and Computer Engineering, 2015. – 76 p.

14. K. Sreenivasa Rao. Robust Emotion Recognition using Spectral and Prosodic Features / K. Sreenivasa Rao, Shashidhar G. Koolagudi. – Springer : Springer Briefs in Electrical and Computer Engineering, 2013. – 118 p.

15. Udo Zölzer. DAFX: Digital Audio Effects, 2nd Edition / Udo Zölzer (Editor). – John Wiley & Sons, 2011. – 624 p.

16. Energy Normalization in Automatic Speech Recognition / N. Jakovljevic, M. Janev, D. Pekar, D. Miskovic // Proceedings of the Text, Speech and Dialogue: 11th International Conference (TSD). – 2008. – P. 341 – 347.

17. Hung – Shin Lee. A Log – energy Scaling Normalization Scheme for Robust Speech Recognition / Hung – Shin Lee, Hung – Bin Chen, Berlin Chen // National Taiwan Normal University. – 2007.

18. Weizhong Zhu. Log – energy dynamic range normalization for robust speech recognition / Weizhong Zhu and Douglas O'Shaughnessy // IEEE International

Conference on Transactions on Acoustics, Speech, and Signal Processing (ICASSP). – 2005. – P. 245–248.

19. Chaudhary R. Network Service Chaining in Fog and Cloud Computing for the 5G Environment: Data Management and Security Challenges / R. Chaudhary, N.Kumar, S. Zeadally // IEEE Communications Magazine.- vol. 55, no. 11.- November, 2017.- P. 114-122.

20. Frahim J. Securing the Internet of Things: A Proposed Framework / J. Frahim // Cisco White Paper.- 2015.

21. Aujla GS Data Offloading in 5G-Enabled Software-Defined Vehicular Networks: A Stackelberg Game-Based Approach / GS Aujla // IEEE Commun. Mag.- vol. 55, no. 7.- July 2017.

22. Peng M. Energy-Efficient Resource Assignment and Power Allocation in Heterogeneous Cloud Radio Access Networks / M. Peng // IEEE Transactions on Vehicular Technology.- vol. 64, no. 11.- Nov. 2015.- P. 5275-5287.

23. Gonzales D. Cloud-trust - A Security Assessment Model for Infrastructure as a Service (Iaas) Clouds / D. Gonzales // IEEE Transactions on Cloud Compu-ting.- vol. 5, no. 3.- July-Sept. 1, 2017.- P. 523-536.

24. John W. Research Directions in Network Service Chaining / W. John // 2013 IEEE SDN for Future Networks and Services.- Nov. 2013.- p. 1-7.

Додаток А

Лістинг програми

```
#Join the geojson file with covid_df
df_final = geojson.merge(covid_df, left_on="coty_code", right_on="fips_code", how="outer")
df_final = df_final[~df_final['geometry'].isna()]

us_map = folium.Map(location=[40, -96], zoom_start=4,tiles='openstreetmap')
us_map

#Create the choropleth map add it to the base map
custom_scale = (df_final['new_cases_7'].quantile((0,0.2,0.4,0.6,0.8,1))).tolist()
folium.Choropleth(
        geo_data=r'...\georef-united-states-of-america-county.geojson',
        data=df_final,
        columns=['fips_code', 'new_cases_7days'],  #Here we tell folium to get the county fips and
plot new_cases_7days metric for each county
        key_on='feature.properties.coty_code', #Here we grab the geometries/county boundaries
from the geojson file using the key 'coty_code' which is the same as county fips
        threshold_scale=custom_scale, #use the custom scale we created for legend
        fill_color='YlOrRd',
        nan_fill_color="White", #Use white color if there is no data available for the county
        fill_opacity=0.7,
        line_opacity=0.2,
        legend_name='New Cases Past 7 Days (Per 100K Population) ', #title of the legend
        highlight=True,
        line_color='black').add_to(us_map)

us_map

#Add Customized Tooltips to the map
folium.features.GeoJson(
                data=df_final,
                name='New Cases Past 7 days (Per 100K Population)',
                smooth_factor=2,
```

```
            style_function=lambda x: {'color':'black','fillColor':'transparent','weight':0.5},
            tooltip=folium.features.GeoJsonTooltip(
                fields=['report_date',
                        'county_name',
                        'state_name',
                        'new_cases_7days',
                        'pct_positive_7days'
                       ],
                aliases=["Report Date:",
                         "County Name:",
                         "State Name:",
                         "New Cases Past 7 days<br>(Per 100K Population):",
                         "Percent of Positive Cases<br>Past 7days:"
                        ],
                localize=True,
                sticky=False,
                labels=True,
                style="""
                    background-color: #F0EFEF;
                    border: 2px solid black;
                    border-radius: 3px;
                    box-shadow: 3px;
                """,
                max_width=800,),
                    highlight_function=lambda x: {'weight':3,'fillColor':'grey'},
                ).add_to(us_map)

us_map

#Create two FeatureGroup layers
us_map = folium.Map(location=[40, -96], zoom_start=4,tiles=None,overlay=False)
fg1 = folium.FeatureGroup(name='New Covid-19 Cases Past 7
Days',overlay=False).add_to(us_map)
```

```
fg2 = folium.FeatureGroup(name='Percent of Positive Cases Past 7
Days',overlay=False).add_to(us_map)

#Add the first choropleth map layer to fg1
custom_scale1 = (df_final['new_cases_7days'].quantile((0,0.2,0.4,0.6,0.8,1))).tolist()
New_cases=folium.Choropleth(
        geo_data=r'...\georef-united-states-of-america-county.geojson',
        data=df_final,
        columns=['fips_code', 'new_cases_7days'],
        key_on='feature.properties.coty_code',
        threshold_scale=custom_scale1, #use the custom scale we created for legend
        fill_color='YlOrRd',
        nan_fill_color="White", #Use white color if there is no data available for the county
        fill_opacity=0.7,
        line_opacity=0.2,
        legend_name='New Cases Past 7 Days (Per 100K Population) ',
        highlight=True,
        overlay=False,
        line_color='black').geojson.add_to(fg1)

#Add customized tooltips to the map
folium.features.GeoJson(
                data=df_final,
                name='New Cases Past 7 days (Per 100K Population)',
                smooth_factor=2,
                style_function=lambda x: {'color':'black','fillColor':'transparent','weight':0.5},
                tooltip=folium.features.GeoJsonTooltip(
                    fields=['report_date',
                            'county_name',
                            'state_name',
                            'new_cases_7days',
                            'pct_positive_7days',
                           ],
```

```
            aliases=["Report Date:",
                    "County Name:",
                    "State Name:",
                    "New Cases Past 7 days<br>(Per 100K Population):",
                    "Percent of Positive Cases<br>Past 7days:",
                    ],
            localize=True,
            sticky=False,
            labels=True,
            style="""
               background-color: #F0EFEF;
               border: 2px solid black;
               border-radius: 3px;
               box-shadow: 3px;
            """,
            max_width=800,),
               highlight_function=lambda x: {'weight':3,'fillColor':'grey'},
            ).add_to(New_cases)

#Add layer control to the map
folium.TileLayer('cartodbdark_matter',overlay=True,name="View in Dark Mode").add_to(us_map)
folium.TileLayer('cartodbpositron',overlay=True,name="Viw in Light Mode").add_to(us_map)
folium.LayerControl(collapsed=False).add_to(us_map)
us_map
us_map.save("index.html") #save to a file
```

Додаток Б

ВІДГУК

керівника економічного розділу

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Факультет інформаційних технологій
Кафедра програмного забезпечення комп'ютерних систем

ВІДГУК

Керівника економічної частини Професора Вагонової О.Г.
(прізвище, ім'я, по батькові, вчене звання)

на магістерську роботу

Студента ІІ курсу групи 121м-20-1 Тарана Данила Григоровича
(прізвище, ім'я, по батькові)

На тему: Розробка програмного забезпечення просодичної модифікації мовного сигналу

.

«___»_____20__ р. _____
(підпис)

Додаток Д

ПЕРЕЛІК ДОКУМЕНТІВ НА ОПТИЧНОМУ НОСІЇ

Ім’я файла	**Опис**
Пояснювальні документи	
ТАРАН.doc	Пояснювальна записка кваліфікаційної роботи. Документ Word.
ТАРАН.pdf	Пояснювальна записка кваліфікаційної роботи в форматі PDF
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація.ppt	Презентація роботи