Хмельницький національний університет
Факультет програмування
та комп'ютерних і телекомунікаційних систем
Кафедра інженерії програмного забезпечення

# ДИПЛОМНА РОБОТА

Технологія розробки програмної системи для озвучення тексту
голосом людини на основі машинного навчання

Назва теми

Рівень вищої освіти Другий (магістерський)

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

Шифр ДРІПЗ.150181.01.07.ПЗ

Виконав студент 2 курсу група ІПЗм-19-1 _____ В. В. Савінський
Підпис Ініціали, прізвище

Керівник д-р фіз.-мат. наук, професор _____ Л. П. Бедратюк
Науковий ступінь, звання Підпис Ініціали, прізвище

Нормоконтролер канд. техн. наук, доцент _____ Г. І. Радельчук
Підпис Ініціали, прізвище

**До захисту допускаю:**
Завідувач кафедри інженерії
програмного забезпечення _____ Л. П. Бедратюк
Підпис Ініціали, прізвище

7 грудня 2020 р.

Хмельницький 2020

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Програмування та комп'ютерних і телекомунікаційних систем
Кафедра Інженерії програмного забезпечення
Рівень вищої освіти Другий (магістерський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Л. П. Бедратюк
01 09 2020 р.

# ЗАВДАННЯ
# НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ)

Савінському Владиславу В'ячеславовичу
Прізвище, ім'я, по батькові студента

1. Тема проєкту (роботи) Технологія розробки програмної системи для озвучення тексту голосом людини на основі машинного навчання

Керівник проєкту (роботи) Бедратюк Леонід Петрович, д-р фіз.-мат. наук, професор
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 01.09.2020 р. № 118
2. Строк подання студентом проєкту (роботи) на кафедру 01.12.2020 р.
3. Вихідні дані до проєкту (роботи) Матеріали переддипломної практики
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Створення архітектури нейронної мережі і необхідних модулів з обробки, побудова архітектури соціальної платформи, проведення збору аудіоданих та створення набору даних українською мовою, реалізація програм для збору даних, тренування, оцінка та оптимізація роботи нейронної мережі
5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)
Презентаційні матеріали (слайди)

6. Консультанти розділів дипломного проєкту (роботи)

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийн |
| | | | |
| | | | |

7. Дата видачі завдання «_01_» __вересня__ 2020 р.

## КАЛЕНДАРНИЙ ПЛАН

| Назва етапів (розділів) дипломного проєкту (роботи) | Строк виконання етапів проєкту (роботи) | Примітка |
|---|---|---|
| 1 Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження, визначення структури дипломної роботи | 01.09 - 05.09.2020 | виконано |
| 2 Робота над розділом 1 дипломної роботи - вивчення літературних джерел; аналіз відомих моделей, методів та засобів за темою роботи, висновки до розділу та постановка задачі | 06.09 - 15.09.2020 | виконано |
| 3 Робота над розділом 2 дипломної роботи - розробка методів і моделей синтезу голосу людини; висновки до розділу; робота над науковими статтями | 16.09 - 20.09.2020 | виконано |
| 4 Робота над розділом 3 дипломної роботи - розробка алгоритмів та технологій, проектування ПЗ для вирішення поставленої задачі; висновки до розділу | 21.09 - 14.10.2020 | виконано |
| 5 Робота над розділом 4 дипломної роботи - програмна реалізація спроектованих рішень, результати експериментів, їх аналіз; висновки до розділу | 15.10 - 01.11.2020 | виконано |
| 6 Узгодження постановки задачі, отриманих результатів та висновків, оформлення пояснювальної записки згідно вимог стандартів | 02.11 - 09.11.2020 | виконано |
| 7 Попередній захист дипломної роботи | 10.11.2020 | виконано |
| 8 Перевірка роботи на наявність плагіату нормоконтроль; брошурування пояснювальної записки; підготовка супровідних документів | 11.11 - 05.12.2020 | виконано |
| 9 Підготовка до захисту дипломної роботи | 05.12 - 08.12.2020 | виконано |
| 10 Захист дипломної роботи | 09.12.2020 | виконано |

Студент _____ В.В. Савінський
Підпис Ініціали, прізвище

Керівник проекту (роботи) _____ Л.П. Бедратюк
Підпис Ініціали, прізвище

# РЕФЕРАТ

Тема дипломної роботи: «Технологія розробки програмної системи для озвучення тексту голосом людини на основі машинного навчання».

Автор проекту: Савінський Владислав В’ячеславович.

Керівник проекту: Бедратюк Леонід Петрович.

Пояснювальна записка: 85 с., 41 рис., 4 дод., 25 джерел.

ТЕХНОЛОГІЯ, СИНТЕЗ ГОЛОСУ, СОЦІАЛЬНА ПЛАТФОРМА, ЗБІР ДАНИХ, МАШИННЕ НАВЧАННЯ.

Об’єктом дослідження є технології для озвучення тексту.

Предметом дослідження є нейронні мережі для озвучення тексту.

Мета роботи – розробка архітектури та тренування глибокої нейронної мережі для синтезу голосу українською мовою та розробка платформи для збору, створення і розмітки звукових наборів даних для навчання нейронної мережі.

У дипломній роботі проаналізовані техніки для озвучення тексту голосом людини за допомогою нейронних мереж і поточний стан наборів аудіоданих. Уточнено гіперпараметри моделі на нових наборах даних для української корпусу. Удосконалений метод накопичення і збагачення звукових наборів даних шляхом розробки соціальної платформи зі спеціальним інтерфейсом для колективного роботи над створенням наборів аудіоданих. Створено розширений український набір аудіоданих та модель для синтезу українського голосу. Для розробки програмної системи використано мову програмування Python 3, Javascript та модуль PyTorch з веб-сервером Flask. Реалізована програмна система може застововуватись для автоматичного озвучення книг, надання підтримки людям із вадами зору, автоматизованого озвучення оголошень та в навігаційних системах.

# ABSTRACT

Thesis: Machine learning text to speech software system technology with a human voice

Authour of the project: Savinskyi Vladyslav Viacheslavovych.

Project Manager: Bedratyuk Leonid Petrovich.

Explanatory note: 85 pages, 41 pictures, 4 additions., 25 sources.

TECHNOLOGY, VOICE SYNTHESIS, SOCIAL PLATFORM, DATA COLLECTION, MACHINE LEARNING.

The object of research is the technology for sounding the text.

The subject of research is neural networks for sounding text.

The thesis analyzes the techniques for sounding the text with human voice using neural networks and the current state of audio data sets. The hyper parameters of the model are specified on new data sets for ukrainian corpus. Improved accumulation method for sound data sets via making social platform with special interface for collective voiceover, marking text and making sound data sets. An extended Ukrainian audio data set and model for synthesizing ukrainian voice has been created. To develop the software system, the programming languages Python 3, Javascript and PyTorch module with the Flask web server were used. Developed software system can be used for automatic makeing audiobooks, helping people with eyes defects, automated sounding of announcements at railway stations and in navigation systems.

# ЗМІСТ

# ВСТУП

Спілкування за допомогою мови є основним способом комунікації між людьми. Проте, не завжди при такому способі передачі інформації необхідно, щоб однією із комунікуючих сторін була саме людина. Сучасні технології дозволяють звільнити людину від таких рутинних та монотонних задач. Системи синтезу мовлення різної якості використовуються давно і вони вже є звичайною частиною нашого повсякденного життя: підтримка людей із вадами зору, автоматизація оголошень на залізничних станціях, в навігаційних системах, автоматизовану робота з клієнтами по телефону, чат-боти на сайтах компаній. Дану технологію використовують компанії в магазинах як аудіореклами. Також на на платформі Youtube присутні рекламні вставки від п’яти до десяти секунд у вигляді невеликого рекламного відео в різним місцях основного відео. Голосові рекламні вставки на радіо створюються за допомогою синтезованого голосу людей. Озвучення голосу також використовується з метою відпочинку і в індустрії розваг. Широко відомі голосові помічники Siri та Alisa для смартфонів. Компанія Hanson Robotics в 2015 році розробила робота Софію, який візуально схожий на людину, може говорити і вести простий діалог із співбесідником.

Завданням озвучення тексту людським голосом є швидке донесення необхідної інформації слухачу без залучення іншої людини. Завдяки широкому колу додатків, озвучення тексту привертає увагу розробників і дослідників. Синтез мовлення для озвучення тексту є частиною дисципліни під назвою обробка природної мови (NLP), яка знаходиться на перетині областей штучного інтелекту та лінгвістики. Останнім часом алгоритми синтезу мовлення суттєво покращили свою роботу в зв’язку з використанням нейронних мереж. Так, Алекса від компанії Amazon – це система з нейронний мереж, яка може якісно синтезувати мовлення.

Сучасні нейронні мережі, які використовуються для синтезу голосу, видають якісний результат, який кращий за традиційні методи розробки, що були популярні у 80-ті і 90-ті роки. Це зв’язано з тим, що нейронні мережі краще моделюють параметри, які потрібні для синтезу голосу, а також набагато зросла потужність

комп’ютерів. Розробники систем синтезу мовлення, повинні використовувати якісні і помічені вхідні дані для тренування мереж. Такі дані є цінною інформацією і не завжди є відкритими чи доступними для розробників. В результаті, сама можливість лише почати тренувати нейронні мережі для синтезу голосу вже потребує доступу до великого кола помічених даних. Наприклад, проект LibreVox – це збірка аудіокниг, де можна знайти багато книг в аудіо-форматі, але вони не мають розмітки початку і кінця текстових блоків у вигляді речень чи фраз.

Іншою проблемою, є те, що на сьогоднішній день більшість систем синтезу мовлення створені лише для англійської мови. Розробка системи для озвучення тексту українською мовою потребує великих масивів українських даних. Доцільно використати для цього доступні аудіо ресурси для їх помітки в паралельному режимі за допомогою спільноти розробників та волонтерів. Завдяки цьому можна створювати локальні набори даних з відкритим доступом. Подібні дослідження поєднують розробку технології для озвучення тексту і створення соціальної платформи для організації помічених наборі аудіоданих.

Отже, для задачі синтезу голосу створення та розмітка наборів голосових даних українською мовою є однією з найважливішою складовою частиною в розробці системи синтезу мовлення.

Метою даної роботи є розробка архітектури та тренування глибокої нейронної мережі для синтезу голосу українською мовою голосу та розробка платформи для збору, створення і розмітки звукових наборів даних для навчання нейронної мережі.

Для досягнення мети дослідження поставлено наступні завдання:

– створення архітектури нейронної мережі і необхідних модулів з обробки звукових даних;

– побудова архітектури соціальної платформи, проведення збору аудіоданих та створення набору даних українською мовою, реалізація програм для збору даних;

– тренування, оцінка та оптимізація роботи нейронної мережі.

Об’єкт дослідження – технології для озвучення тексту.

Предмет дослідження – нейронні мережі для озвучення тексту

Гіпотеза дослідження полягає в тому, що систему озвучення тексту для конкретного голосу можна успішно розробити при достатній кількість помічених даних за допомогою глибинних нейронних мереж і техніці «навчання зі вчителем».

Практична значимість дослідження полягає в створенні офлайн системи для озвучення тексту голосом, яку можна вчити на власних аудіоданих та використовувати програму для колективного створення помічених даних з будь-яких звукових джерел. Якісні звукові дані грають важливу роль в процесі навчання глибинних нейронних мереж для озвучення тексту, які самі виділяють необхідні ключові ознаки зі слів і фраз у вхідних даних і реагують на їх наявність під час тестування.

Теоретична значимість роботи полягає в аналізі параметрів і аналізі архітектури багатьох нейронних мереж; вивченні впливу різних гіперпараметрів на рівень подібності синтезованого голосу на людський. Досліджено різні джерела наборів даних, виявлено їхні переваги і недоліки.

Отримані результати дозволяють зробити висновок, що озвучувати тексти можна за допомогою невеликих нейронних мереж, але для їх роботи потрібні якісні аудіодані. Для реалізації запропонованих моделей розроблено веб-сайт, який складається з трьох модулів для різних даних обробки аудіоданих, що дозволяють організовувати, помічати і створювати наборів аудіоданих. На основі зібраних даних створений український набір аудіоданих з помітками і розроблено end2end модель на базі глибинних нейронних мереж для озвучення тексту українською мовою.

На етапі підготовки даних були розроблені:

– дві програми для маркування та озвучення тексту (створення набору аудіоданих);

– соціальна платформа для об’єднання людей в процесі маркування текстів та їх озвучення;

– клієнтський додаток для серверу.

Під час виконання завдань дослідження були застосовані:

– методи обробки природної мови;

– математична статистики;

– нейронні мережі;

– методи цифрової обробки даних;

– методи оптимізації машинного навчання.

Наукова новизна отриманих результатів:

– уточнено гіперпараметри моделі [22] на нових наборах даних для української корпусу; аналіз гіперпараметрів моделі дозволив зробити висновок, що найбільший вплив на тренування має динамічна зміна параметру learning rate (за допомогою learning rate decay) під час навчання, що дозволяє плавно проводити процес fine-tune (налагодження) моделі;

– удосконалено метод накопичення і збагачення звукових наборів даних шляхом розробки соціальної платформи зі спеціальним інтерфейсом для колективного озвучення і поміток текстів та створення звукових наборів даних;

– вперше створено розширений український набір аудіоданих;

– створено модель для синтезу українського голосу (STTSv2020.v1).

У дипломній роботі проаналізовані техніки для озвучення тексту голосом людини за допомогою нейронних мереж і поточний стан наборів аудіоданих. В результаті, розроблений розширений набір даних українською мовою і модель для озвучення текст. Синтезований голос можна коригувати після проведення тренування мережі у межах декількох годин.

Тема роботи є актуальною і потребує подальшого розвитку, а саме:

– моделювання тонких емоцій, персональних властивостей голосу;

– дослідження моделей для синтезу довгих послідовностей;

– синтез людського співу, музики;

– дослідження генерації кількох голосів однією моделлю, активне моделювання ритму голосу;

– вдосконалення нейронних вокодерів.

Результати дослідження у вигляді наборів даних доступні для інших розробників. Програмна реалізація даного дослідження може бути використана для створення набору помічених аудіоданих на будь-якій мові, для тренування і оцінки моделей синтезу голосу. Створена платформа може озвучувати цілі текстові блоки,

що можна застосувати для створення аудіокнижок. Книги не є обов’язковою вхідною інформацією, тому синтезований голос може бути сформований з вхідних текстових повідомлень чату соціальної мережі, або використовуватись для озвучення фраз персонажів комп'ютерних ігор.

За результатами магістерської роботи опубліковані тези для наукової конференції «Актуальні проблеми комп’ютерних наук»:

Савінський. В. Social platform for making labeled audio datasets for speech synthesis of human voice // Зб. наук. пр. наукової конференції «АПКН-2020». – Хмельницький ХНУ. – 2020. – С. 261-264.

# 1 ХАРАКТЕРИСТИКА ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

## 1.1 Аналіз предметної області і виявлення наявних проблем

Синтез голосу активно досліджують в останні десятиліття. Процес переведення тексту в мову (TTS) є частиною області обробки і синтезу мовлення, який озвучує написаний людський текст. Дана технологія має зрозумілу проблематику і результати, що дозволяє використовувати її разом з іншими технологіями. Так, автоматичне розпізнаванням мовлення, тобто обернена задача синтезу голосу, і машинний переклад текст разом з технікою синтезу голосу дозволяє людям спілкуватися на різних мовах і розуміти один одного. Тому, синтез мовлення – це важлива частина в обробка природної мови, яка потрібна для комунікації між людьми.

Сьогодні існує багато методів для спілкування: текстові повідомлення, аудіоповідомлення, відео-чати. Така можливість стала доступна завдяки розвитку інформаційних технологій та мережі Інтернет.

Аналіз зовнішніх процесів великих технологічних компаній по розробці програмного забезпечення (ПЗ) дозволяє зробити висновок, що ці компанії зацікавлені в тому, щоб люди продовжували використовувати їхні технології, тому що це економічно вигідно для компаній. Також слід зауважити, що такі компанії володіють великим обсягом накопичених даними про своїх користувачів. Завдяки цьому вони можуть робити аналіз на великих даних за допомогою алгоритмів машинного навчання, які набирають популярність в останні роки.

Великі дані ще не мають такої цінності, як помічені дані, тобто це дані, що мають смислову помітку біля кожного файлу, який може бути, наприклад, зображенням чи текстом. Процес створення помічених даних часто є тривалим процесом і потребує уважності від людини. Помічені дані потрібні для проведення «навчання зі вчителем». Також існує «навчання без вчителя», де дані для обох напрямків алгоритмів потрібно збирати з датчиків, якими є будь-які електронні пристрої з записом аудіо, відео чи інший даних.

Чим більше сервіс спілкування буде подібним на реальну форму спілкування, тим більше його будуть використовувати і цінувати, а також збільшиться залучення користувачів, що дасть змогу отримувати актуальні дані для підтримки й покращення роботи бізнесу. Перед розробниками соціальних платформ стоїть ряд задач, вирішення яких збільшить прив’язаність користувачів. Однією з таких задач є синтез людського голосу.

Існує багато методів і архітектур систем для синтезу голосу. Сьогодні можна почути високоякісну синтетичну мову, яка конкурує з вокалом людини. Актуальність задачі синтезу голосу із застосуванням сучасних інформаційних методів на базі нейронних мереж зумовлена тим, що в ІТ стали популярні методи вирішення прикладних задач за допомогою нейронних мереж, тому що вони видають досить якісний результат, що особливо помітно у сфері розпізнавання, сегментації і генерації зображень. Синтез голосу широко використовується у машино-людських взаємодіях, в число яких входять:

– віртуальні голосові помічники, такі як Аліса від Яндекс, Сірі від Apple, Cortana від Microsoft;
– кол-центри;
– роботи;
– комп’ютерні ігри;
– програми охорони здоров’я;
– центри зайнятості.

Популярность синтезу голосу має декілька причин. Перша причина полягає в тому, що спосіб передавати людині інформацію звуком ефективніший у порівнянні з читанням. В цьому контексті слово «ефективність» застосовується до процесу слухання, яке не потребує від людини стільки фокусу уваги, скільки потрібно в момент читання, яке вимагає від людини дивитись на текст. Читати складно під час ходьби або під час рухів головою, проте слухати музику або аудіокнигу в навушниках під час подорожі на велосипеді, або під час спортивного заняття є для людей звичайною задачею. Друга причина полягає у можливості чути людський

голос, що робить систему синтезу людино-орієнтованою, тому що людині приємно слухати людський голос, особливо це голос знайомої людини.

Голосові повідомлення мають більшу інформаційну насиченість порівняно з текстовими повідомленнями, а також швидкість створення голосового повідомлення в декілька разів більша за швидкість набіру тексту на клавіатурі. З іншого боку, голосові повідомлення мають мати в собі інформацію, яка є паузами з тишею. Така інформація необхідна в мовленні людей, тому що людина не може сприймати інформацію з такою ж швидкістю як машина, що робить тишу невід'ємним компонентом комунікації людей, як пробіли у тексті.

Людина слухає людей, коли вони зробили запис свого голосу в формі аудіоповідомлення або відео. Також є прямий ефір, де люди напряму говорять до інших людей, але це потребує створення «каналу зв'язку» між людьми, без якого вони не зможуть спілкуватись. Сучасні технології дозволяють синтезувати голос, але він обмежується одним або декількома голосам. В даному контексті синтез голосу дозволяє озвучувати книги, які раніше не були озвучені, а також слухати повідомлення в соціальних мережах в формі аудіоповідомлень. В цій ситуації синтезований голос має бути схожий на голос людини, яка прислала повідомлення, а це вимагає створення голосової моделі співрозмовника. В результаті, для реалізації цієї задачі потрібний засіб запису голосу співрозмовника для створення голосової моделі, мова про яку піде у другому розділі, а також сам синтезатор голосу.

Зазвичай засобом запису голосу є диктофон або комп'ютер з відповідною програмою для зйомки. Також є сайти, на яких люди можуть записувати свій голос. Завдяки таким програмам створюється набори даних, де є голоси і фоновий шум. В таких умовах не існує спільної цілі, що об'єднувала б працю багатьох людей. В результаті, немає цілісної платформи з орієнтацією за створення набору аудіоданих.

Провівши аналіз поточного стану в даній галузі, можна сказати наступне:

– аудіокомунікацію між людьми можна покращити за допомогою синтезу голосу;

– розвиток збору наборів аудіоданих можливий при наявності платформи з чіткою ціллю і простим способом залучення користувачів.

## 1.2 Аналіз методів машинного навчання

Машинне навчання – це підрозділ штучного інтелекту, ціль якого зводиться до створення математичних моделей, в яких закладається контрольні параметри, завдяки яким модель можна вчити і покращувати на основі певних даних (розмічених і нерозмічених).

Так, алгоритми на базі машинного навчання мають прикладні застосунки:

– Google Duplex, що є прикладом асистента, який може сам зробити замовлення в магазинах голосом власника;

– машина з автопілотом, де використовується комп’ютерний зір на базі машинного навчання для задачі знаходження і розпізнавання об’єктів;

– підказки в листах на пошті Gmail;

– мовна моделі GPT-n для генерації тексту.

Звісно, штучний інтелект повинен мати алгоритми, які відповідають за мораль і цінності для того, щоб могти правильно вирішити відому етичну проблему вагонетки. Ієрархію машинного навчання по відношенню до штучного інтелекту зображено на рисунку 1.1.

Завдяки розважальній індустрії, що в більшій мірі становить собою ігрова індустрія, яка дала великий потік фінансів виробникам ігрових відеокарт і іншого технічного обладнання, на глобальних ринках стали доступні відеокарти, що потребують відповідних процесорів для розкриття своїх потужностей, які стають доступнішими для більшості.

В сучасних умовах вже розроблено широкий спектр математичних інструментів для проведення аналітичних робіт, які пов’язані з побудовою моделей. Машинне навчання – це наука, яка об’єднує в собі декілька математичних дисциплін, а саме:

– математичний аналіз;

– теорія управління;

– статистика;

– лінійна алгебра.


Рисунок 1.1 – Ієрархія напрямків штучного інтелекту

В свою чергу, машинне навчання, як дисципліна, вирішує декілька типів технічних та гуманітарних задач. За типом навчання виділяються декілька видів навчання:

– навчання зі вчителем (Supervised learning);
– навчання з підкріпленням (Reinforcement learning);
– спонтанне навчання (Unsupervised learning);
– напівавтоматичне навчання (Semi-supervised learning).

Навчання зі вчителем є найпопулярнішим варіантом машинного навчання, де систему вивчають за схемі «об’єкт – реакція» для одержання моделі реакції для певного об’єкта.

До навчання зі вчителем відносяться задачі класифікації, регресії, сегментації. До спонтанного навчання відносяться задачі кластеризації, виявлення аномалії, зниження розмірності. Напівавтоматичне навчання – це різновидність навчання з учителем, де у процесі тренування використовуються велику кількість непомічені дані і невелику кількість помічених даних. До напівавтоматизованого навчання відносяться задача генерації зображень, генерація природної мови та рекомендації. Навчання з підкріпленням – це складніший різновид навчання зі вчителем, бо в ньому ускладнене прогнозування системи, що спричинене можливою можливу довгою затримкою зворотнього зв’язку середовища. Завдання цієї галузі машинного навчання є настільки широкі, що їх вивчають у теорії керування, дослідженні операцій, теорії інформації.

В останній роки замітний потужний розвиток галузі машинного навчання, де застосовуються нейронні мережі. Нейронні мережі дозволяються створити модель, яка зможе знаходити взаємозв’язки, які потребували багато часу і ресурсів якби їх виконувала людина. Наприклад, задача сортування фотографій за кількістю людей на ній потребує багато часу від однієї людини. Також є задачі, де необхідна висока швидкість реагування. Так, під час руху автономного автомобіля для безпечної роботи потрібно виявляти об’єкти на дорозі і розпізнавати на них. Сьогодні нейронні мережі можуть вирішувати такі складні задачі, тому що вони автоматично будують характеристики і ознаки, які необхідні для виявлення потрібний об'єктів. Це, з одного боку, дозволяє створювати практичні додатки, а з іншого боку, такі рішення як нейронні мережі подібні до «чорного ящика», який дійсно може виконувати порівняно складні задачі і одночасно мати складний механізм налагодження. Це означає, що для створення розумної системи потрібне уміле об’єднання та гнучка архітектура системи, де результати від різних методів будуть взаємодіяти один з одним для загальної цілі.

## 1.3 Аналіз наявних методів для озвучення тексту

Існують різними способи синтезу голосу, які часто базуються на різних способах побудови звуку. Як тільки нейронні мережі почали ставати більш популярним і більш доступним інструментом для програмістів, їх почали застосовувати до задачі синтезу голосу. Після цього, методи синтезу голосу стали розділяти на дві категорії:

– донейромережеві;
– нейромережеві;

Обидві категорії мають суттєву різницю між собою як в архітектурі, так і в типі вихідних даних. Задача синтезу мовлення є складною з різних причин. Запис мови в повній відповідності з їх звучанням неможливо виконати звичайним орфографічним записом. В орфографічному записі відсутня повна відповідність між буквами і звуками, що вимовляє людина. Людина говорить не так як пишеться текст. Звуків вимови більше ніж букв алфавіту. Щоб краще наблизити записи мовлення використовує фонетичну транскрипцію, при цьому виникають питання правильної розстановки наголосів і проблема зняття омографів.

До транскрипції входить:

– завдання токенізації і нормалізації тексту;
– розстановки наголосів;
– визначення тривалості пауз;
– побудова фонетичної транскрипції.

Після транскрипції вхідний текст переводиться у зручну форму для подальшого синтезу мовлення.

### 1.3.1 Донейромережеві методи синтезу голосу

Список донейромережевий методів включає в себе:

– unit selection [2];

– статистичний і параметричний метод [3, 4, 5];
– синтез формантів [6].

Метод unit selection оперує базою записаних фонем (звуків і слів) для поєднання їх в один аудіофайл. Для цього в момент створення даних кожній фонемі дається один або багато поміток: фон, діфон, напівфон, склад, звук, морфема, буква, слово, фраза, речення. Цей процес складний і довгий, тому для цієї задачі використовують готову програму для поділ слів на відповідні частини, яку часто можна бачити наукових колах по обробці людської мови і голосу. Для проведення такої автоматичної розмітки використовуються спеціальний розпізнавач мови, який видає акустичні параметри (позиція, довжина, висота звуку). На основі цих параметрів побудується індекс одиниць в базі даних. Також використовуються дерева ухвалених рішень для вибору найкращої одиниці з бази даних, а також методи динамічного програмування для мінімізації цільової функції вартості зв’язку між окремими частинами. В такому методі присутня мала кількість цифрової обробки звуку, що є незаперечною перевагою цієї технології в швидкості роботи і впливає на якість вихідного звуку. Недоліком такого методу є обмежений стиль голосу та великий розмір бази даних для синтезу, який потребує багато часу на створення лише однієї нової моделі голосу, що робить цей метод дорогим в підтримці.

Статистичний метод [3] синтезу голосу базується на математичній і фізичній моделі синтезу звуку. Автори цього методу аналізували анатомію людини і вирішили не записувати звуки вимови, а виділити тільки деякі величини для синтезу, які були основані на механізмі відтворення звуку в гортані і зв’язках людини. Природа цих параметрів основа на source-filter моделі говоріння, в якій органи в момент мовлення змінюють свої фізичні властивості, що показано на рисунку 1.2, де видно залежність резонансу величин в аудіоспектрі від вимови різних звуків. Цей метод більш універсальний, ніж конкатенативний синтез, що може досягти досить природного синтезованого мовлення, але підхід за своєю суттю обмежений властивостями мовного корпусу і довжиною записаних у базі звуків частин, що використовується для процесу вибору одиниці і динамічного підбору даних.

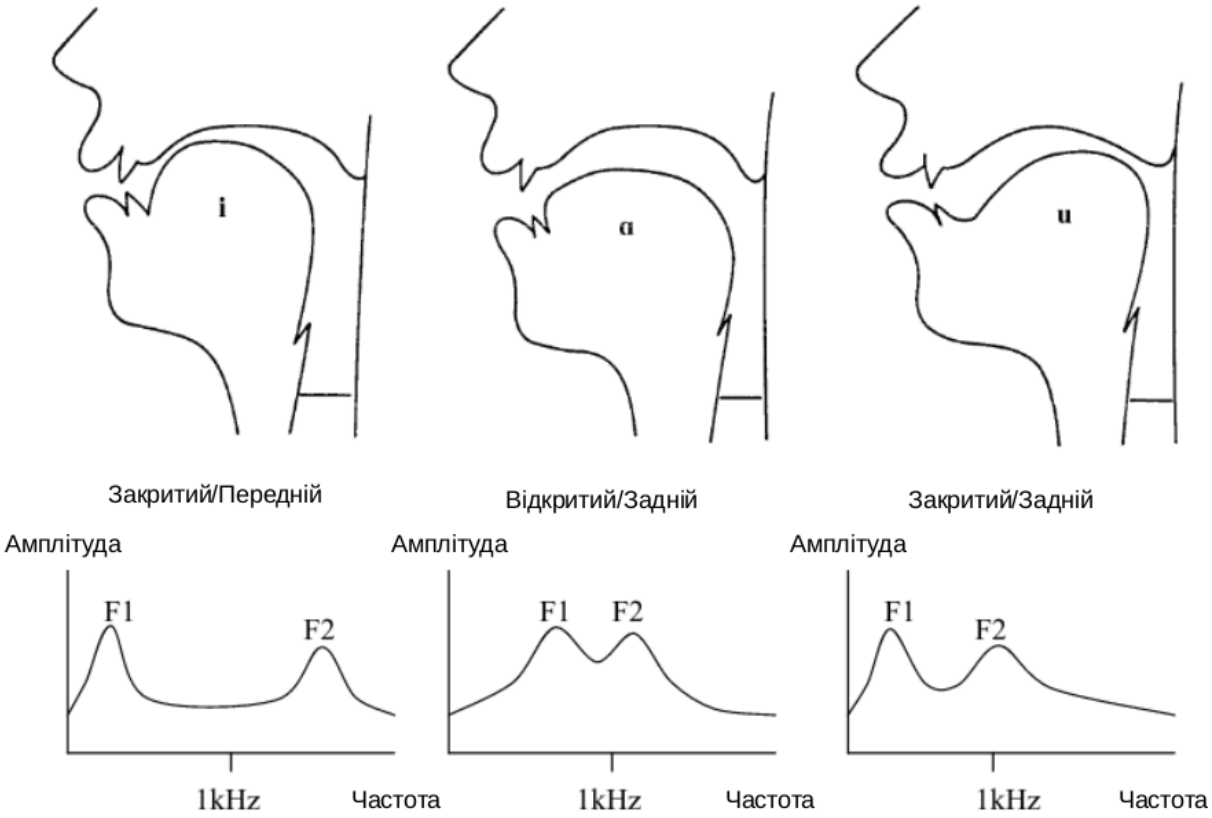



Рисунок 1.2 – Частотна реакція для різних форм голосових шляхів

Спочатку з вхідного тексту будуються контрольні параметри, які потрібні в звуковому фільтрі для видачі звуку. На рисунку 1.3 зображені чотири формати, які перемножуються в вхідним джерелом звуку (спадаючим фоном) і вихідний звуковий спектр звучить як потрібний звук. Декілька акустичних параметрів моделюються за допомогою стохастичної генеративної моделі часових рядів. До акустичних параметрів відноситься:

– фундаментальна частота F0;
– рівень голосу (для формування голосних);
– рівень шуму (для відтворення приголосних).

Всередині такого методу використовуються HMM-моделі (приховані марківські моделі) чи глибинні нейронні мережі, які передбачають контрольні параметри. Далі метод використовує вокодер для озвучення проміжних сигналів. Кожний з цих проміжних сигналів змінюється в часі і створює звукові хвилі, які сприймаються як голос.

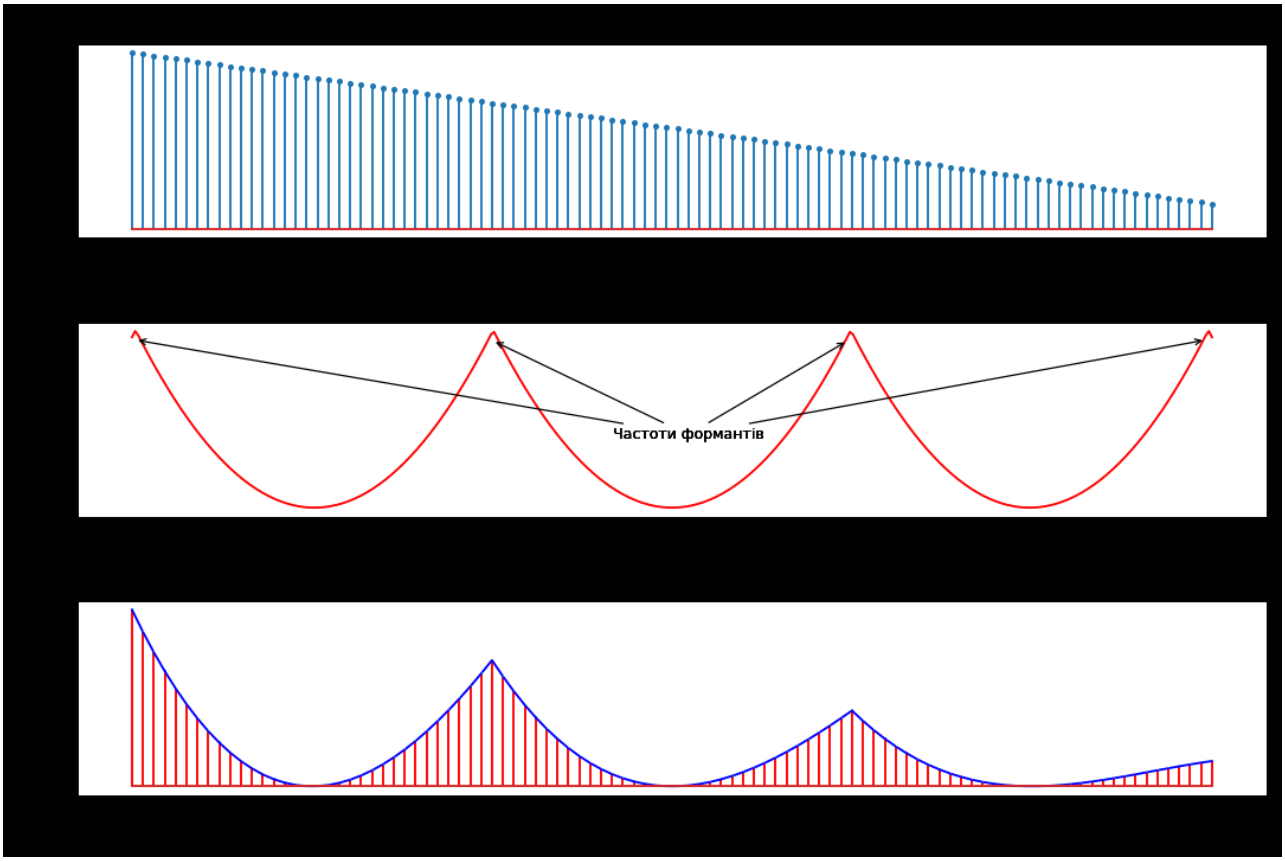



Рисунок 1.3 – Схема відтворення звуку з комбінацією фону і контрольні параметрів по source-filter моделі

Процес перетворення хвилі в звук з фонем уявлення називається вокодингом. Алгоритми для вокодинга називають вокодерами. Вокодери синтезують голос на базі вхідного стисненого сигналу у великих аудіоспектрах. Вокодер не обов’язково повинен бути нейромережевим для синтезу голосу.

Переваги методу:

– малий розмір;
– автоматична побудова голосу;
– мала мовна залежність;
– гнучкість / керованість.

Завдяки гнучкості цього методу можна впливати на стиль вихідного сигналу і робити вимову речень чи окремих слів веселою, сумною, серйозною. Недоліком методу є необхідність у вородері, що деградує загальну якість голосу, але в останні роки стали популярні такі нейромережеві вокодери, як WaveNet [7], які майже не

відрізняються від справжнього голосу, але вони дорогі у використанні. Без якісних нейромережевих вокодерів синтезований голос стає «металевим», схожим на робота. Цей метод синтезу голосу став популярним завдяки своїм перевагам і часто використовувався в ігрових консолях Sega.

В порівнянні з методом unit selection, статистичний метод складніший в реалізації, але він дозволяє створювати загальні рішення без необхідності мати записані і поділені частинки мови в будь-якому фонетичному контексті. З точки зору реалізації, unit selection синтез вимагає велику базу даних для покриття варіацій голосу. На відміну від цього, статистичний параметричний синтез дозволяє комбінувати та адаптувати моделі і, отже, не вимагає варіантів будь-яких можливих комбінацій контекстів.

Синтез формантів – це структура з фільтрів, які реагують на резонанси в голосовому сигналі. Формант – пік частотної характеристики безперешкодного голосового тракту. Один з простих резонаторів, що складають складну резонансну систему голосових шляхів. Формантний фільтр працює завдяки комбінації полосових і фазових фільтрів. Кількість фільтрів в схемі визначається порядком самого форманту. В дослідженні [6] застосовують даний метод для маніпуляції та контролю формантів у синтезі мовлення на основі HMM моделей.

### 1.3.2 Нейромережеві методи синтезу голосу

Конкатанативний синтез є найпоширенішою технологією синтезу голосу, але останнім часом він поступово заміщується більш досконалим нейромережевим параметричним синтезом мови, в якому мова генерується безпосередньо з моделі голосу диктора.

До наявних методів синтезу голосу, які використовують нейронні мережі належать наступні методи:

– WaveNet (2016);
– Fast WaveNet (2016);
– Deep Voice (2017);

– Tacotron (2017);
– Deep Voice 2 (2017);
– TTS + ASR(Speech Chain) (2017);
– Deep Voice 3 (2017);
– Tacotron 2 (2017);
– GST-Tacotron (2018).

У 2016 році була запропонована нова парадигма для синтезу голосу, а саме група Google Deepmind випустила WaveNet [7] – генеративну модель для аудіосинтезу. Вона показала значну перевагу над конкатанативним синтезом у плані натуральності та якості вихідного звуку. Це повністю згорткова глибинна нейронна мережа, що зображена на рисунку 1.4. В цьому дослідженні була запропонована causal модель і розширені згортки (dilated convolutions), що показано на рисунку 1.5. Загалом, мережу WaveNet сприймають як вокодер та використовуються разом з іншими мережами для синтезу голосу.

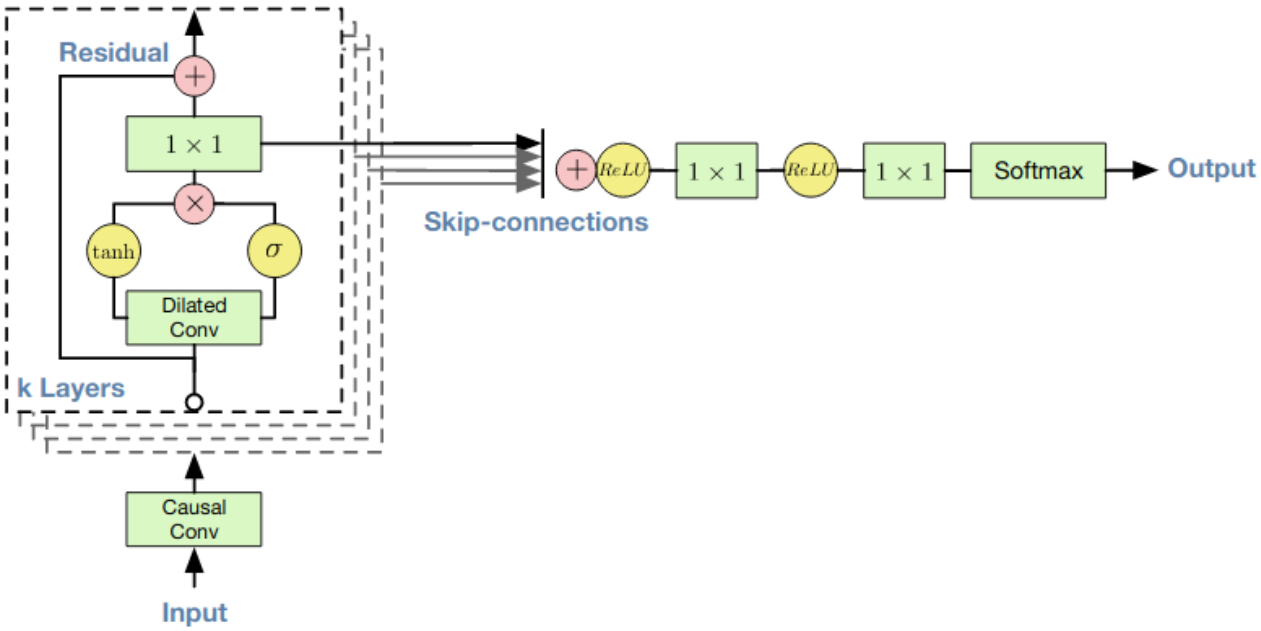



Рисунок 1.4 – Огляд залишкового блоку та всієї архітектури WaveNet [7]

Це дослідження зробило революцію в області синтезу голосу, тому що якість цього вокодера майже не відрізняється від людського голосу. Розробники стали будувати статистичні моделі замість HMM-моделей.

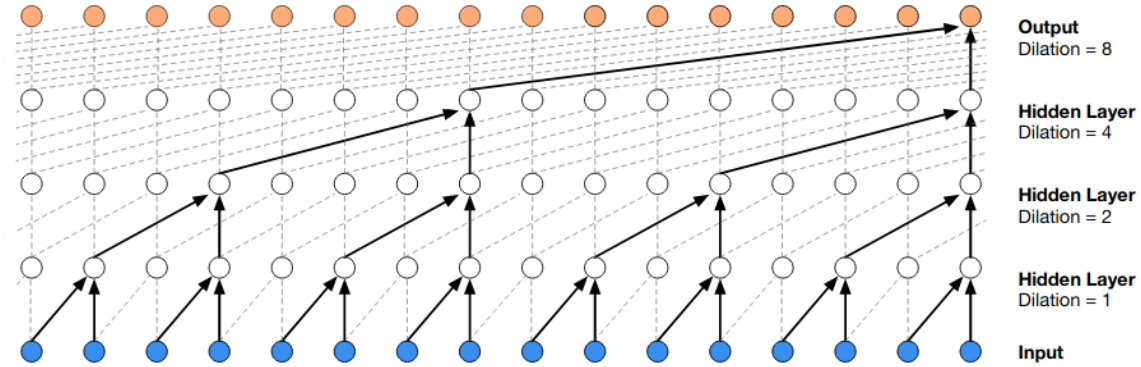



Рисунок 1.5 – Схема dilated causal згорткових шарів [7]

Властивості моделі:

– проста моделі для реалізації;

– генерування аудіосигналів без посередників і обробників звуку (більше 16000 семплів на секунду);

– softmax шар моделює згорткове розподілення скрізь кожний вхідний аудіосемпл для мережі;

– модель може бути розширена до наявності декількох спікерів всередені однієї архітектури;

– оцінка MOS сягає 4,21 для англійського набору даних.

До недоліків моделі відноситься:

– відсутність повністю end-to-end архітектури і вимога до підготовки текстових ознак;

– потреба у великих ресурсів для обчислення сигналів;

– довгий час тренування.

Після цього дослідження, нейромережеві технології синтезу мовлення почали активно розвиватися. У листопаді 2016 року Іллінойський університет та компанія IBM розробили модель fast wavenet [8]. Ця модель має механізм кешування, щоб виділити надлишкові згорткові шари за допомогою допомогою кеша в середині моделі WaveNet. Завдяки кешуванню швидкість моделі значно збільшилась.

В лютому 2017 компанія Baidu презентувала модель deep voice [9], що складається з декількох самостійно навчених моделей, об’єднаних у послідовну

чергу. Вона складається з окремо навчених моделей Grapheme-to-Phoneme і Segmentation, які генерують ознаки і набори даних для тренування аудіосинтезу, передбачення довжини аудіочастинок і передбачення фундаментальної частоти FO(Fundamental Frequency). Властивості моделі:

– згорткова модель перетворення графем у фонему (кодувальник – це Bi-GRU розміром 1024x3, декодер – це GRU розміром 1024x3);

– модель сегментації для визначення меж фонеми (згортка, GRU, згортка);

– модель прогнозування тривалості фонеми (FC-256x2, GRU розміром 128x2, FC);

– модель прогнозування основний частот (спільна модель з вищезазначеними);

– модель синтезу звуку (різновид WaveNet);

– працює швидше реального часу (до 400 разів швидший як на процесорі, так і на графічному процесорі порівняно з Fast WaveNet).

До недоліків цієї моделі відносяться:

– вона є не повністю end-to-end системою (Deep Voice покладає головну роботу на повністю end-to-end голосовий синтез);

– оцінка MOS сягає 2,67 для англійського набору даних.

У березні 2017 компанія Google випустила end-to-end модель синтезу голосу tacotron [10] – це sequence-to-sequence модель, яка складається з кодера і декодера з механізмом уваги. Він потребує лише пару (текст, звук) і може тренуватись з випадковим початковим станом. Його оцінка MOS сягає 3,82 для англійського набору даних. Tacotron відрізняється тим, що він показує деякі нові підходи в області синтезу мовлення. Кодер на початку це шар з character embeddings, що подається на вхід у мережу PreNet. Також у моделі є модуль CBHG(1d convolition back, highway network, bidirectional GRU), який спочатку був розроблений для завдання нейромашинного перекладу. Далі в ньому є механізм уваги, який застосовується до кожного декодера та має критично впливає на якість

синтезованого голосу. Загальна архітектура tacotron наведена на рисунку 1.6. Для озвучення вихідної спектограми використаний алгоритм Griffin-Lim.

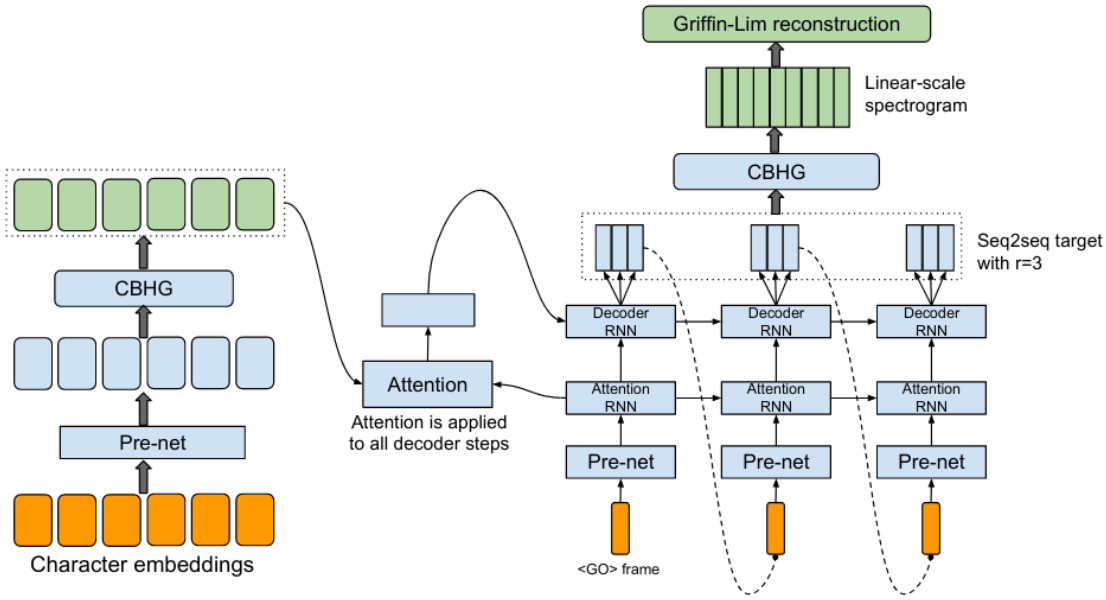



Рисунок 1.6 – Архітектура моделі Tacotron [10]

Властивості даної моделі:

– повністю end-to-end система, яка навчається на парах (текст, аудіо) з випадковим початковим станом;

– оцінка MOS сягає 3,82 для англійського набору даних;

– можливість прогнозування лінійних спектрограм.

До недоліків моделі відноситься потреба у великій кількість ресурсів і часу тренування мережі.

У 2017 компанія Baidu випустила оновлену версію моделі deep voice під назвою deep voice 2 [11]. Архітектура моделі майже співпадає з оригіналом. Ключовим моментом було введення ознак для декількох спікерів (multi-speaker feature embeddings). Це дало можливість генерувати сотні унікальних голосів. Всі моделі оснащені ознаками спікерів (speaker embeddings). Модель сегментації – це згорткова RNN мережа. Структура deep voice 2 має три частини:

– модель сегментації фонем;

– модель тривалості;

– модель частотності.

Ключовою властивістю моделі цій моделі була наявність обробки тренувальних даних пачками на рівні згортки шарів і додання залишкових з'єднань. Голосова модель базується на архітектурі WaveNet. Ця модель має MOS-оцінку 3,53.

Інститут науки і техніки Японія запропонував [12] використовувати багатомовну TTS модель разом з розпізнавання мови для реалізації «мовного ланцюга». Механізм мовного ланцюга інтегрує автоматичний модуль розпізнавання мови (ASR) разом з синтезом тексту в мову в єдиний цикл під час навчання мережі. В їхній попередній роботі був застосований механізм мовного ланцюга як система напівнатурального навчання. Це забезпечує здатність ASR та TTS допомагати один одному коли вони отримують неспарені дані, і дозволяють їм зробити висновок про відсутність пари та оптимізувати модель із втратою при реконструкції. В даному дослідженні використана загальноприйнята метрика продуктивності системи розпізнавання мови або машинного перекладу CER(character error rate). У ході проведення експериментів з примусовим навчанням з вчителем та вибіркою від Gumbel-Softmax, був покращений показник ASR на 11% відносного зниження CER порівняно з їх базовим рівнем.

Властивості цієї моделі є наступними:

– TTS модель навчається спільно з моделлю розпізнавання ASR;

– тренування нейронної моделі проводиться у два етапи: під наглядом та без нагляду;

– модель TTS схожа на tacotron, а ASR – це модель кодера-декодера багатошарової мережі LSTMNet.

У жовтні 2017 року Baidu випустила модель deep voice 3 [13], архітектура якої зображена на рисунку 1.7. Тут багатомовна система повністю відрізняється від попередніх версій deep voice, тому що вона має модель тривалості та сегментації. Одна модель генерує спектрограму для декодування в аудіохвилі за допомогою wavenet мережі. Ознаки для навчання внутрішньої мережі можуть мати інші ознаки, які може генерувати WaveNet, а саме: mel-band спектограми, linear-scalelog спектограми магнитуди, фундаментальна частота, ознака «spectral envelope». Замість WaveNet можна використані інших модулі.

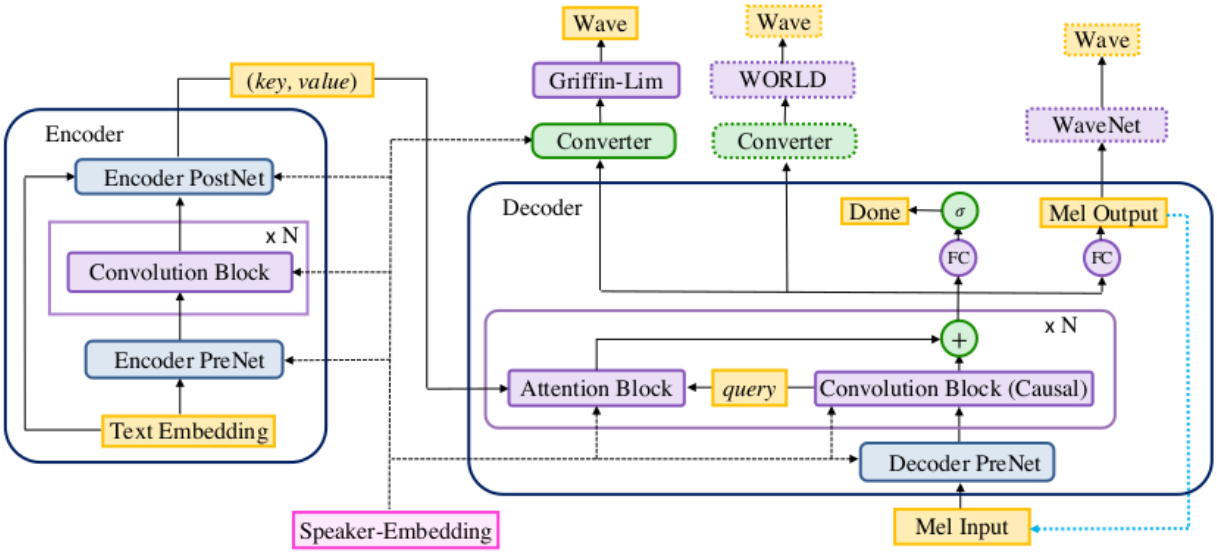



Рисунок 1.7 – Архітектура моделі Deep Voice 3 з використанням залишкового згорткового шару для кодування тексту для декодера на основі методу адитивної уваги [13]

Властивості моделі:

– повністю згортковий sequence-to-sequence архітектура кодера і декодера з монотонним механізмом уваги;

– перетворює введений текст у спектрограми (або інші акустичні ознаки);

– застосовує механізм уваги для впровадження монотонного вирівнювання звука відносно букв;

– потребує в 10 разів менше часу для тренування і дає якісний результат після 500 ітерацій (порівняно з моделлю Tacotron, який потребує більше двох мільйонів ітерацій);

– MOS оцінка 3,78 (з WaveNet), такий же бал для Tacotron (з Wavenet), 2,74 для Deep Voice 2 (з WaveNet).

У 2017 році Google випустила tacotron 2 [14]. Ця модель побудована з модифікованим кодером, де є три згорткових шари і двонаправлені LSTM блоки, які призначені для кодування (character embeddings). Адитивна увага була замінена механізмом «location sensetive attention». Загальна архітектура модулів глибинної нейронної системи tacotron 2 наведена на рисунку 1.8.

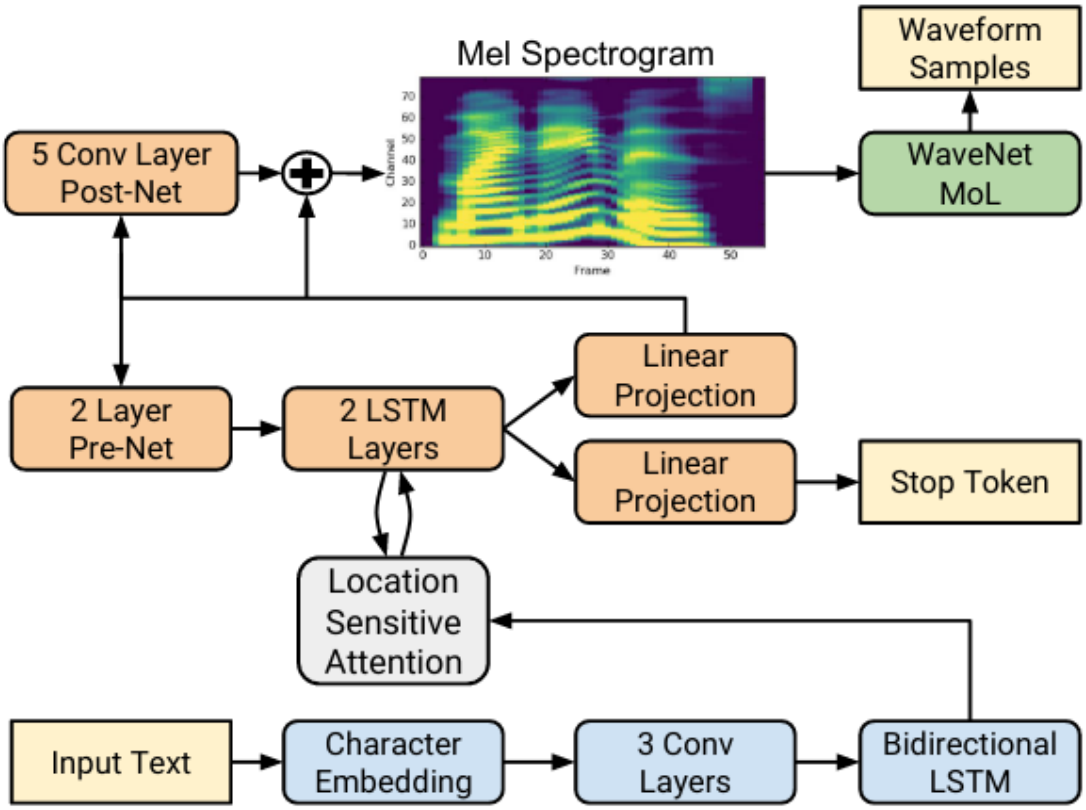



Рисунок 1.8 – Архітектура моделі tacotron 2 з модифікованим кодером [14]

Це дослідження показало, як можна використати Wavenet MOL (Mixture of Logistics), щоб регулювати прогноз спектрограми та генерувати голосові аудіосигнали.

Властивості моделі:

– має прості модулі і має зрозумілу архітектуру, на відміну від оригінального Tacotron;

– переводить символи в мел-спектрограму;

– модифікований WaveNet безпосередньо синтезує звукові форми із спектрограм без необхідності генерувати детальні лінгвістичні ознаки, тривалість фонеми та інші особливості;

– можливість використовувати попередньо підготовлену модель WaveNet, що значно впливає на час тренування;

– оцінка MOS досягає 4,526 для англійського набору даних.

В березні 2018 Google AI запропонувала нову text-to-speech модель з наявністю стилю голосу під GST-Tacotron [15], що зображена на рисунку 1.9 у вигляді діаграми. Глобальні стильові токени відповідають за швидкість і стиль звучання, тобто вони є «банком ознак» (bank of embeddings), які генеруються на базі tacotron моделі. В момент тренування кодер перетворює вхідні звукові хвилі в вектор великої фіксованої розмірності. Далі цей вектор подається на вхід до шару нейронної мережі, яка проводить перенесення стилю голосу, яка, за допомогою механізму уваги, обраховує спеціальні токени зі стильовими ознаками. Далі ці токени використовуються як частина вхідних даних для tacotron кодера.

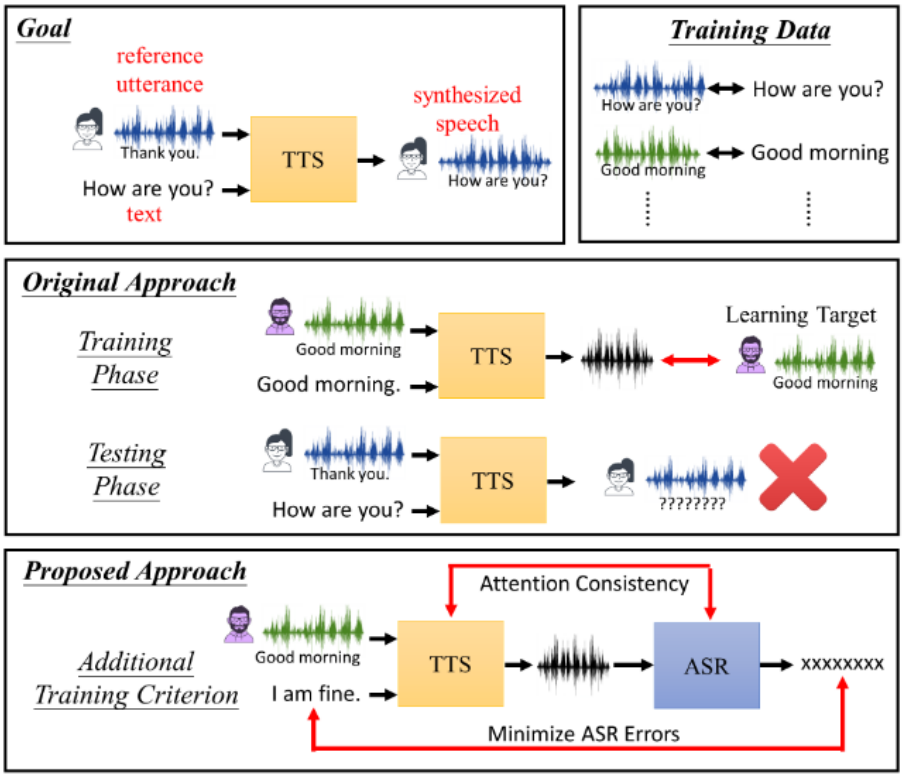



Рисунок 1.9 – Архітектура моделі GST-Tacotron [15]

В момент перевірки синтезу голосу є вибори:

– засновано на моделі tacotron з певними змінами;

– вхідне аудіо може бути використане для створення стильових токенів;

– навчені стильові перетворення можуть використовуватись для прямого контролю стилю синтезованого голосу;

Властивості моделі:

– засновано на моделі tacotron з певними змінами;

– мережа вивчає 10 стильових токенів;

– вхідний кодер є серію двовимірних згорток і 128 GRU шарів;

– використання шару стильових токенів і механізм уваги з токена розмірністю 256.

Структура розглянутих методів зображена на рисунку 1.10.

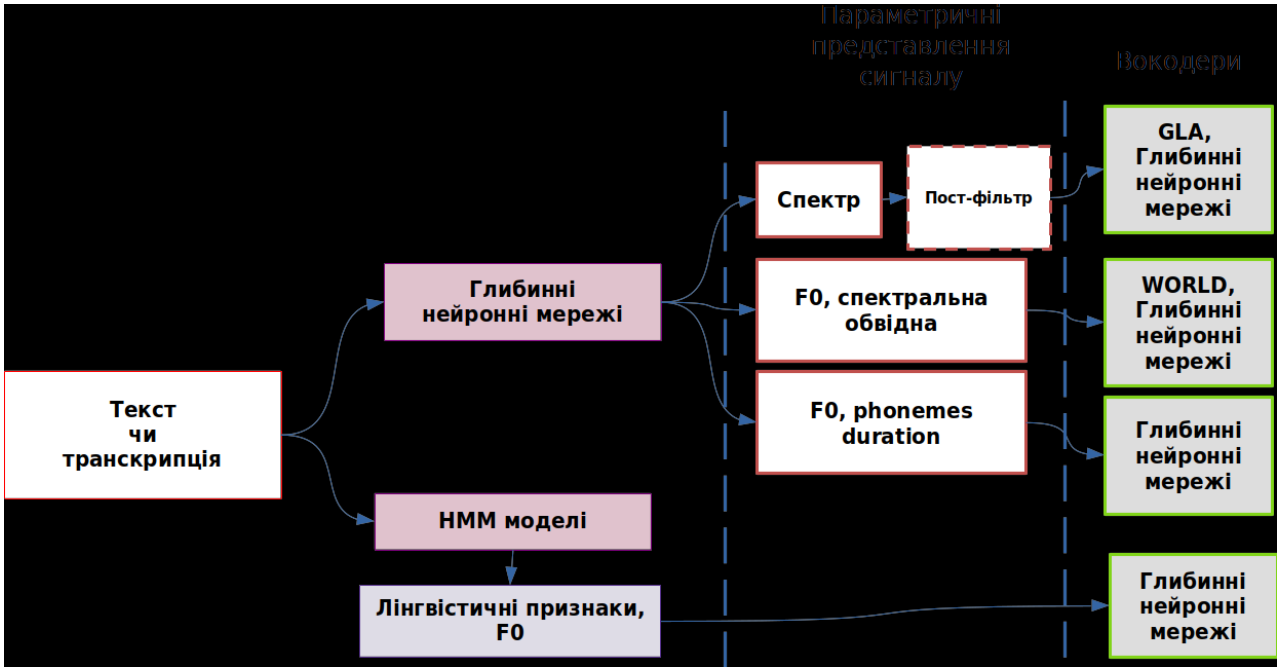



Рисунок 1.10 – Різні варіанти систем для синтезу мовлення за допомогою нейронних мереж

## 1.4 Аналіз наявних методів збору аудіодатасетів

Підготовка набору даних для навчання нейронної мережі є одним з важливих кроків у процесі навчання та підтримки роботи нейронної мережі. Поточні загальнодоступні набори даних використовуються для навчання глибинних нейронних мереж, оцінки результатів та розуміння оцінки інших моделей. Крім того, залежно від типу наборів даних, сам набір даних може бути корисним не тільки для спеціалістів глибинного навчання, а й для доменних експертів та інших

людей. Наприклад, тлумачний словник – це словник, який надає трактування слова, інформацію про вимову, граматику і етимологія, що є корисним для всіх, але, з іншого боку, він може бути основою для набору даних, який може містити категорії слів у ієрархія, що реалізовано в дослідження WordNet [16], де зібрані слова в різні категорії, які мають власну ієрархію.

Звукових наборів даних набагато менше, аніж візуальних наборів даних, бо сьогоднішня складність програмного забезпечення для процесу маркування кожного зображення є великою. Робота з аудіо схожа на роботу зі зображеннями, але популярність візуальних датасетів, таких як ImageNet, пояснюється наступним:

– швидкий процес створення одного зображення;
– швидка оцінка якості даних і виявлення помилок;
– можливості переміщення та маніпулювання наявними даними;
– програмне забезпечення для модифікації вже існуючих даних.

З аудіофайлами робота проходить інакше, тому що:

– звук не має візуальної форми, яка покаже помилки у зрозумілій формі для людини;
– відсутність способу зрозуміти весь звук відразу, тобто людині важно виразити зміст аудіофайлу при одному візуальному представленні.

Відсутність способу зрозуміти весь звук змушує слухати весь запис. У цьому сенсі аудіозапис схожий на відео по своїй формі сприйняття інформації або на зображення, яке виглядає як одна довга стрічка. В порівняння з фотографією, процес створення одного аудіозапису з відповідним йому текстом не є легкою задачею. На це є декілька причин, а саме:

– відсутність легкого у використанні програмного забезпечення для роботи з аудіо і текстом одночасно;
– відсутності шуму;
– складність озвучення довгий частин тексту.

Складність процесу створення аудіо пов’язана з тим, що якщо людина хоче зробити десятисекундний аудіозапис і в процесі його створення робить помилку на

восьмій секунді, то потрібно переробити всю частину. Цей процес може бути довгим і не завжди приємним для людей. Інші способи передбачають наявність знання програмного забезпечення. Завдяки популярності мобільних телефонів і сучасним технологія передачі зв'язку телефон людей має досить непоганий мікрофон, якість якого достатня для запису голосу і використання аудіофайлу для озвучення.

Відсутності шуму на задньому плані під час запису голосу не завжди є позитивною рисою в наборі даних. У залежності від цілей розробника і дослідження, в якісному наборі даних присутній файл з заднім фоном, який розробник може використати для власної фільтрації і обробки звуку перед використанням у тренуванні. Дані про шум використовують [17] для тренування нейронної мережі по розпізнаванню і зменшенню шуму, де приклад шуму потрібен під час верифікації звуку цільовою функцією.

Великі технологічні компанії, як Amazon, Google, Baidu, мають свої закриті бази даних і набори даних, які вони створили за часи свого існування чи придбали за власні кошти. Доступ до таких таких наборів даних є в робітників цих компаній, і вони користуються цими даними для створення продуктів, у тому числі і для задачі синтезу голосу.

Окрім закритих баз даних, також існують відкриті джерела. LibriVox – це вільний сайт аудіокнижок у відкритому доступі. Згідно зі статистикою Librivox, у ньому налічується понад 14000 робіт, що каталогізуються, і понад 1500 не англійських робіт. Всі ці твори є у формі колекції wav файлів без асоційованого тексту для кожного файлу. LibriVox реєструє лише матеріали, які перебувають у відкритому доступі в Сполучених Штатах, і всі книги LibriVox випускаються суспільному надбанню. Через обмеження авторських прав LibriVox видає записи лише обмеженої кількості сучасних книг.

LibriTTS [18] – це мовний корпус, призначений для використання тексту в синтезі мовлення. Він походить від оригінальних аудіоматеріалів та текстових матеріалів корпусу LibriSpeech, який використовувався для навчання та оцінки систем автоматичного розпізнавання мови. Новий корпус успадковує бажані властивості корпусу LibriSpeech, вирішуючи при цьому ряд питань, які роблять

LibriSpeech менш ідеальним для роботи з перетворення тексту в мовлення. Випущений корпус складається з 585 годин мовних даних із частотою дискретизації 24 кГц від 2456 динаміків та відповідних текстів.

LJ Speech [19] – це набір даних у відкритому доступі, що складається з 13100 коротких аудіокліпів одного оратора, що читає уривки з семи науково-популярних книг. Для кожного кліпу надається транскрипція. Тривалість кліпів варіюється від однієї до 10 секунд, а загальна тривалість становить приблизно 24 години. Тексти були опубліковані між 1884 і 1964 роками та перебувають у відкритому доступі. Аудіоверсія була записана у 2016-2017 роках і є загальнодоступним ресурсом.

CSS10 [20] – це колекція наборів даних мовлення з одним оратором для 10 мов світу. Він складається з коротких аудіокліпів з аудіокнижок LibriVox та їх вирівняних текстів.

Описані набори аудіодатасетів у більшості випадків побудовані на існуючих аудіокнижках з ручним вирівнюванням тексту для аудіофайлів. Набори аудіоданих, як правило, мають формат пар (wavfile, text) або триплетів (wavfile, text, transcription).

Якщо спікер хоче зробити аудіоверсію свого повідомлення, то потрібно створити інтерфейс аудіоредактора, де спікер може керувати свою аудіоверсію, вирівнювати текст, автоматично розділяти текст на речення, редагувати текст повідомлення, перезаписувати окремі частини послідовностей та надавати детальну інформацію про себе:

– стать;
– вік;
– географічне розташування.

Інформація про спікера дозволяє отримати детальну картину його голосу. Набори даних з такими даними будуть корисні спікерам, дослідникам машинного навчання і глибинних нейронних мереж в області синтезу голосу, а також гостям-користувачам. На рисунку 1.11 показано, як користувач може взаємовигідно користуватись платформою.

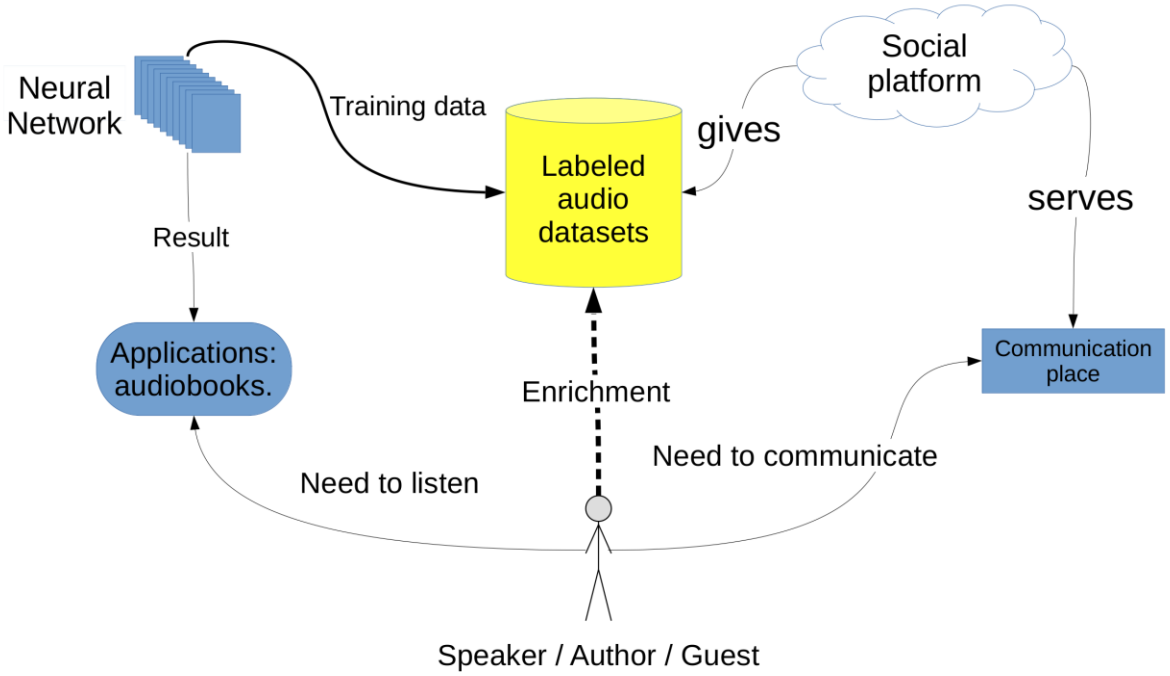



Рисунок 1.11 – Діаграма використання для користувача

Таким чином, можна створити певні локальні набори аудіоданих, які можуть містити таку інформацію, як загальний акцент або діалект якоїсь мови. Додаток, який може використовувати такі набори даних, може побудувати моделі для синтезу мовлення з конкретного регіону країни, налаштувати вік чи стать оратора. Набори даних, зібрані на цій платформі, можуть бути використані не тільки в проектах по синтезу голосу, але і в задачах розпізнавання звукових сигналів, розпізнавання мови, наскрізного розпізнавання мови, перевірки спікера, ідентифікації спікера, виділення шуму, зменьшення шуму в наявному звуці, синтез штучних даних з необіхними шумами, що необхідні в навчанні декількох мереж в режимі «вчитель і студент» та іншими суміжними галузями, що пов’язаних з цим навчання.

Після проведення опису існуючих платформ можна сказати, що існує необхідність в організації наборів аудіоданих і побудові соціальної платформи для створення позначених аудіоданих з простим інтерфейсом і доступом для використання. Платформа має буди доступна користувачам і чітко появнювати та показувати мету проєкту. На платформі повинна бути швидка реєстрація, після якої користувач отримує доступ до панелі пошуку та озвучення текстів з можливістю маркування тексту.

## 1.5 Висновки і постановка задачі

У розділі розглядається задача синтезу голосу людини на базі її власного голосу зі збереженням її мовної стилістики і відтворенням голосу, подібного до людського. В цьому розділі сформульована актуальність задачі синтезу голосу, описані сучасні методи синтезу людського голосу, описані сучасний стан датасетів для синтезу голосу і роботи в аудіоданими та методи збору аудіодатасетів, поставлені основні задачі дослідження.

Актуальність задачі синтезу голосу і створення платформи для збагачення аудіодатасетів полягає у наступному:

– рості кількісті наукових статей з тематики синтезу голосу в останні роки розвитку машинного навчання;

– потребі створювати і ефективніше тренувати нейронні мережі для синтезу звуку;

– відсутність програмного забезпечення для озвучення текстів українською мовою і потреба у розвитку вільного ПЗ;

– необхідності організації інформації і створенні платформи для науковців у сфері машинного навчання;

– необхідності сприйняття текстової інформаційний в звуковій форматі;

– необхідності створення незалежний програмних рішень, які не використовують ресурси великих технічних компаній.

До описаних сучасних методів в області синтезу голосу відносяться:

– конкатенативні методи;

– статистичний і параметричний підхід;

– HMM-метод;

– нейромежеві моделі, а саме spectogram predictor і vocoder.

Сучасний стан аудіодатасетів здатен побудувати загальну систему синтезу голосу, але така система буде «середнім» людським голосом, що є добрим в загальній задачі синтезу голосу, але не в задачі конкретного одного голосу. Для синтезу конкретного голосу людини не підійдуть голоси інших людей, але такі дані

все ж таки мають цінність в процесі створення системи синтезу, хоча вони не мають тих локальних і індивідуальних властивостей, що є в цільовому голосі людини.

До основних задач, які потрібно вирішити під час розробки системи синтезу людського голосу, відносяться:

– створення архітектури нейронної мережі для синтезу голосу;
– створення набору даних;
– проведення тренування.

Після проведення аналізу наявних публікацій і дослідження предметної області, цільова система синтезу голосу має мати наступні можливості:

– робота в офлайн режимі;
– наявність базового голосу, для звичайного синтезу голосу;
– наявність базового голосу подальшого тренування з метою переносу стилю голосу на базову модель голосу;
– наявність платформи, яка буде інструментом для створення аудіодатасетів і спілкування між користувачами.

До функціоналу платформу по створенню набору аудіоданих відноситься:

– реєстрація і аутентифікація користувачів;
– можливості передавати текстові повідомлення між користувачами;
– можливості передавати голосові повідомлення між користувачами;
– створення постів для озвучення;
– можливості залишити коментар до посту;
– індивідуальне озвучення постів;
– колективне озвучення постів;
– функція пошуку постів;
– функція формування наборів аудіоданих з фільтрацією по характеристикам постів.

Об’єктом дослідження є технології для озвучення тексту. Предмет дослідження – нейронні мережі для озвучення тексту.

Метою дослідження є розробка архітектури та тренування глибокої нейронної мережі для синтезу голосу українською мовою та розробка платформи

для збору, створення і розмітки звукових наборів даних для навчання глибинної нейронної мережі.

Завданнями дослідження є:

– створення архітектури нейронної мережі і побудова архітектури соціальної платформи для проведення збору аудіоданих та створення набору даних українською мовою

– створення і розмітка звукових наборів даних для навчання глибинної нейронної мережіж

– тренування, оцінка та оптимізація роботи нейронної мережі.

# 2 РОЗРОБКА МЕТОДІВ І МОДЕЛЕЙ СИНТЕЗУ ГОЛОСУ ЛЮДИНИ НА БАЗІ АУДІОДАНИХ

## 2.1 Математичний апарат для робота з аудіо

Для того, щоб зробити синтез голосу, необхідно мати уявлення про звук і процес мовлення людини. Звук – це різновид форми коливань. Коливання є процес, який локалізується у деякій ділянці простору протягом тривалого часу. Хвиля – це коливний безперервний процес переходу з однієї ділянки простору до іншої. Механічна хвиля – це поширення деформацій пружних середовищ. У широкому розумінні звукові хвилі є механічними хвилями (тобто хвилі в пружних середовищах). У вузькому значенні звук – це пружна хвиля, дія якої створює у людини слухові відчуття. Звичайна людина сприймає звук, частота якого коливається від 20 Гц до 20 кГц. Інфразвук – це більш низькі частоти, ультразвук – більш високі частоти.

Глибиною кодування звуку називають кількість біт, що відводиться на один звуковий сигнал. Сучасні звукові карти забезпечують 16-, 32- і 64-бітну глибину кодування звуку. Отже, хвиля – це процес поширення коливань у просторі.

Як і будь-яка інша форма коливання, справжній звук – це неперервний сигнал. Амплітудна спектрограма – це графік зміни амплітуди звуку в різних момент часу, як показано на рисунках 2.1 і 2.2, де зображений амплітудний спектр функції синуса з частотою 440 Гц. Функція синусу має періодичні неперервні коливання.

Комп’ютер працює з дискретними даними, тому неперервний звук записують певну кількість разів в секунду, завдяки чому зі звуком можна оперувати як з масивом чисел, де індекс масиву означає час запису, а значення – висоту коливання (амплітуду). Кількість коливань хвиль в секунду називається частотою звуку. Хвилі з різною частотою сприймаються органами слуху як звук різної висоти. Хвилі з малою частотою сприймаються як низькі, басові звуки, а хвилі з великою частотою – як високі. Частота звуку вимірюється в Гц. 1 Гц – це одне коливання в секунду; 1000Гц = 1 кГц. Діапазон звуку, який може сприймати людський орган слуху, називають звуковим діапазоном.

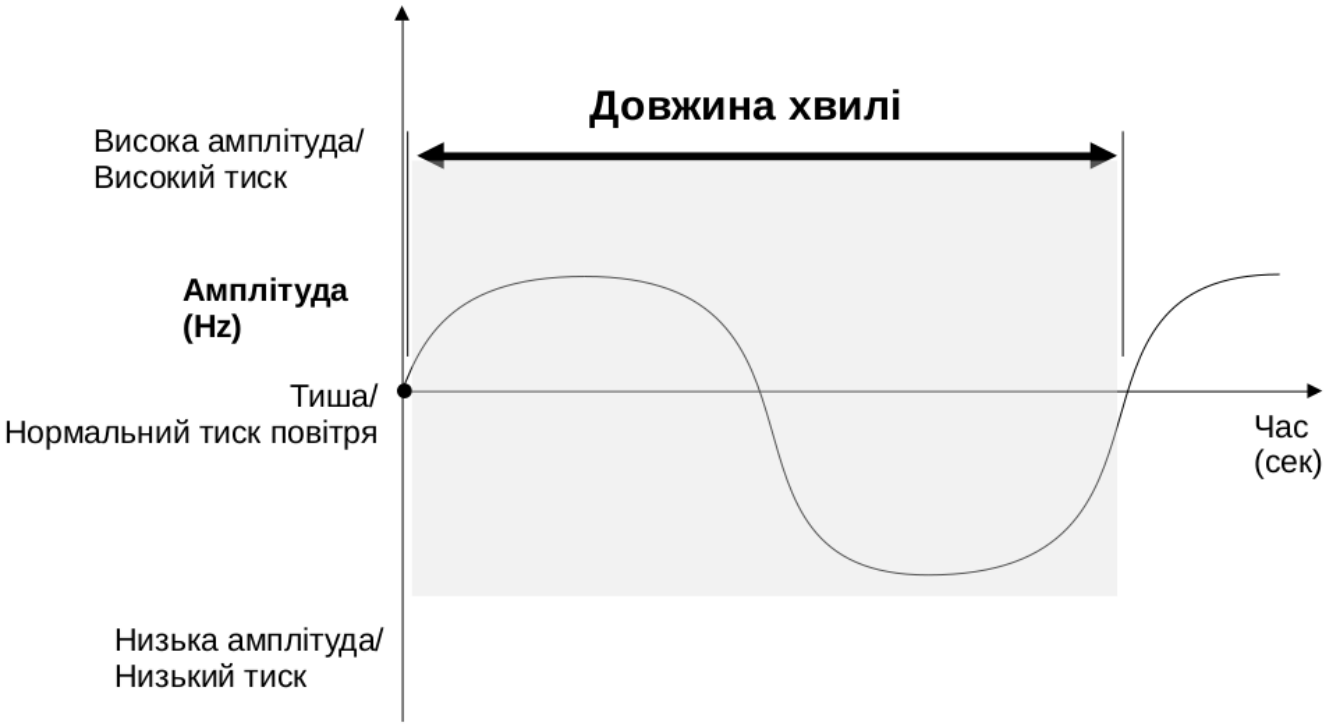



Рисунок 2.1 – Структура звукової хвилі

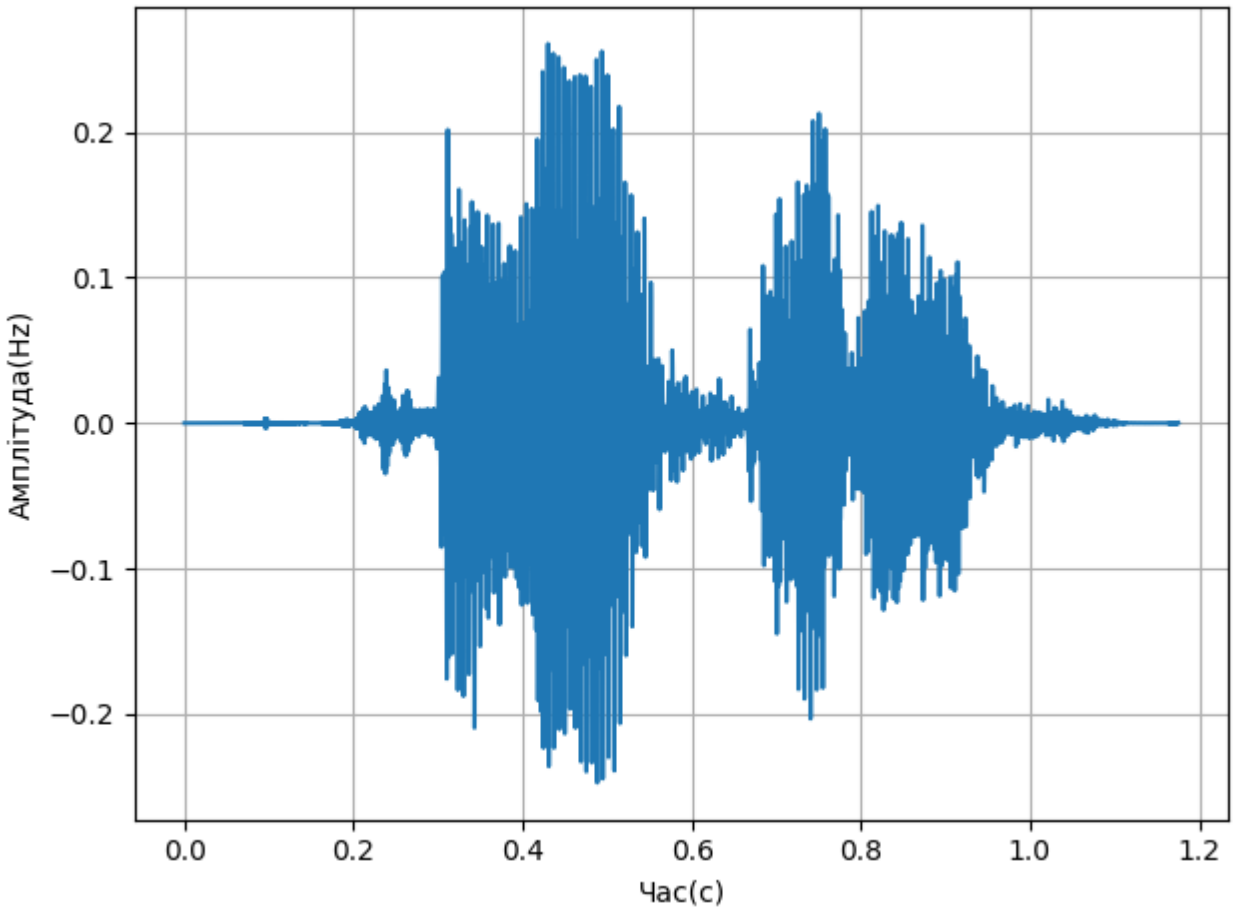



Рисунок 2.2 – Амплітудний графік звуку

В процесі мовлення задіяні багато органів. В першу чергу – це голосові зв’язки, які коливаються на унікальному для кожної людини діапазоні частот, який називають частотний тон. Особливості будови голосових зв’язок і визначають унікальні характеристики голосу. Далі, в результаті роботи артикуляційних органів, а це, в першу чергу, язик, губи і зуби, коливання голосових зв’язок утворюється в приголосні і голосні звуки. Складний природний механізм виникнення голосу, що показаний на рисунку 2.3, можна представити у вигляді простої математичної моделі з цифровим фільтром, який генерує на виході потрібні звуки мови в залежності від типу вхідного сигналу, який може бути періодичним сигналом або білим шумом. Коефіцієнти фільтра періодично перебудовуються відображаючи зміни положення артикуляційних органів.

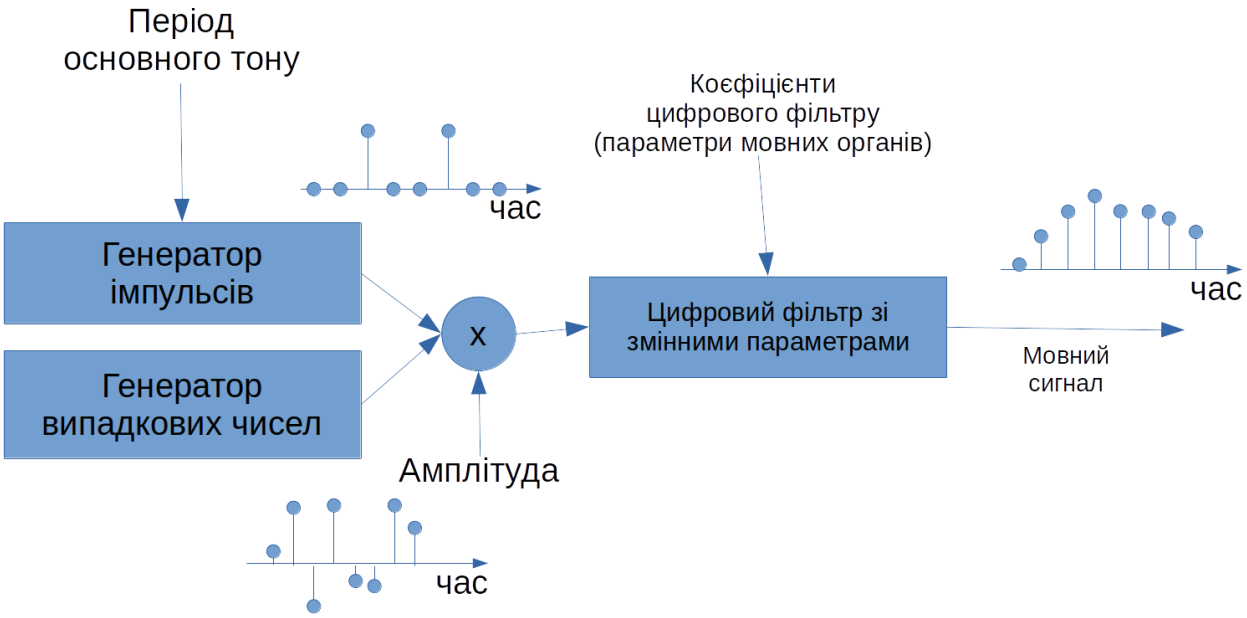



Рисунок 2.3 – Цифрова модель створення мовлення

Будь-який звук можна розкласти на прості синусоїди за допомогою перетворення Фур’є. Дискретне перетворення Фур’є (ДПФ) широко використовується для цифрової фільтрації та спектрально-кореляційного аналізу сигналів. Форма запису звуку в вигляді масиву чисел підходить для домену часу, а за допомогою ДПФ можна побачити звук в домені частоти, що схематично

зображено на рисунку 2.4 і 2.5. На рисунку 2.6 представлені графіки частот під час вимови букви «а» в різних доменах представлення.

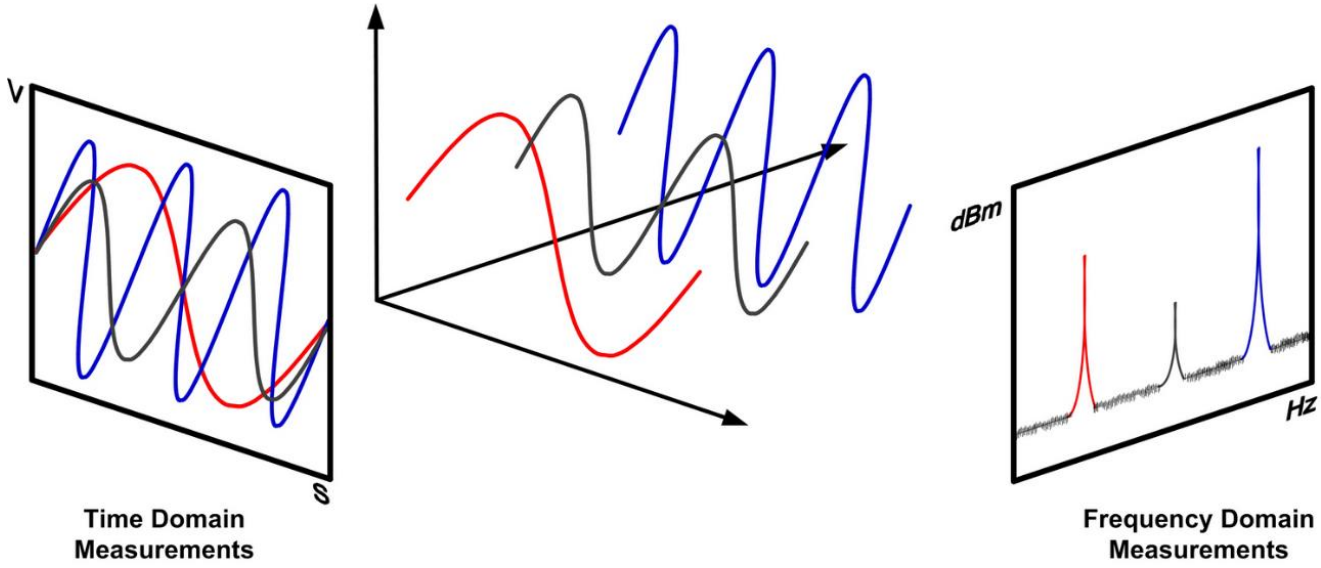



Рисунок 2.4 – Вигляд сигналу в домені часу і частоти [21]

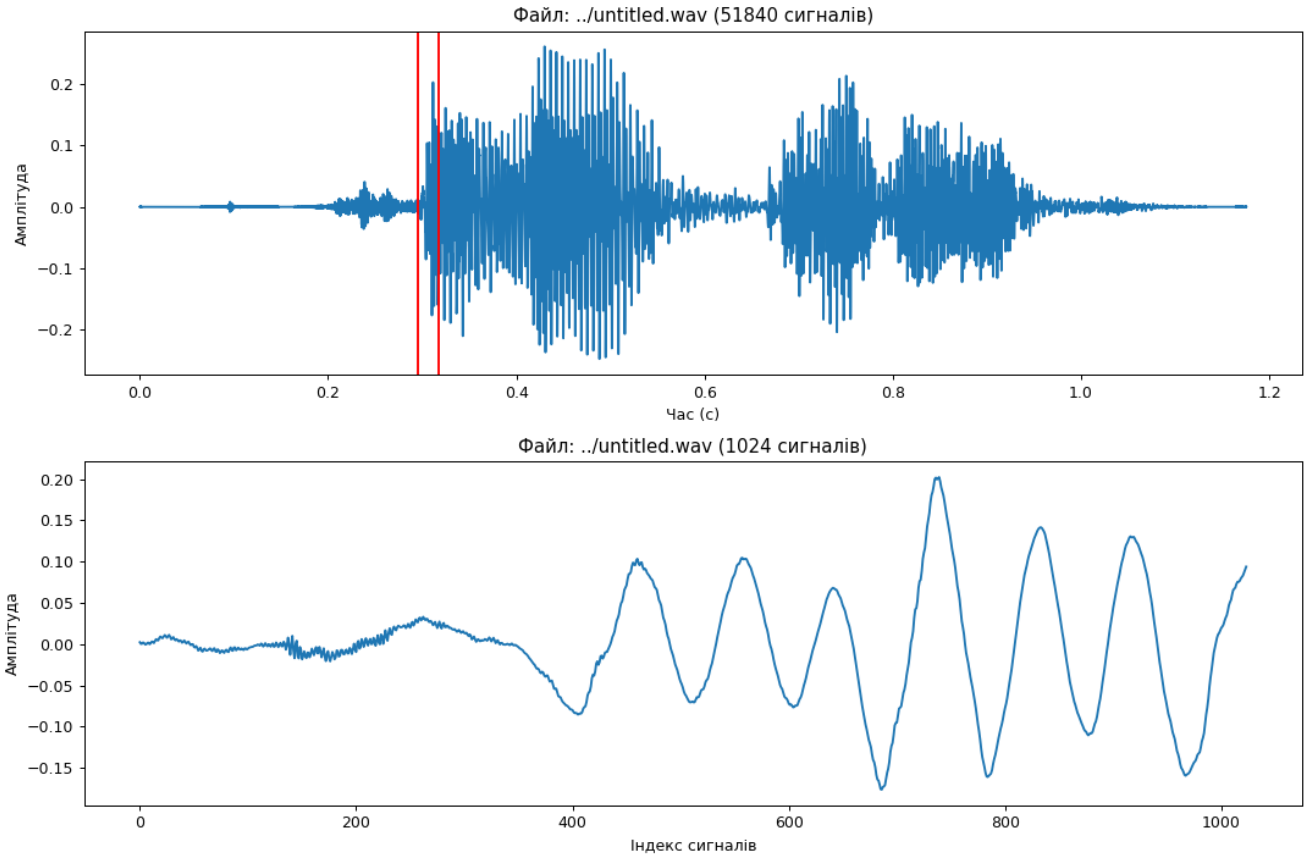



Рисунок 2.5 – Приклад амплітудної спектрограми

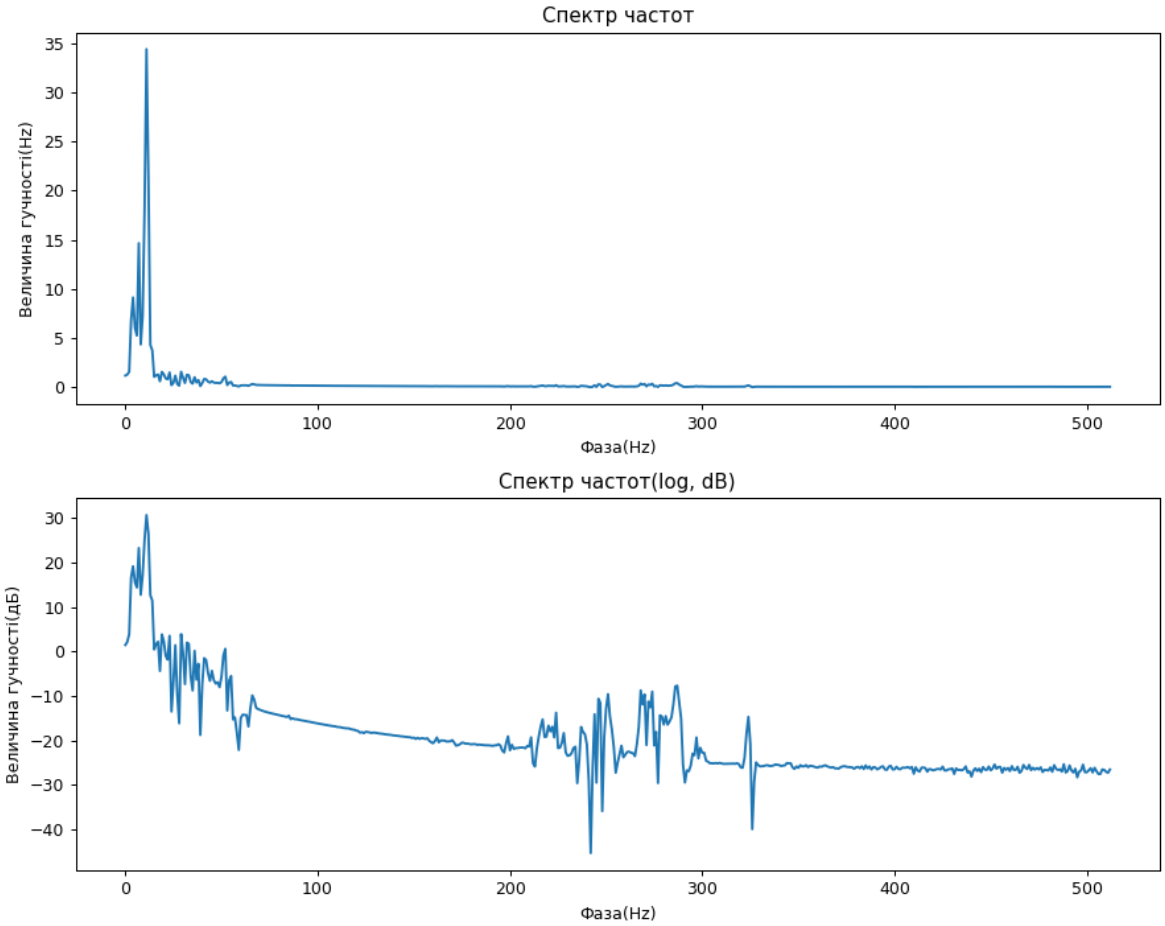



Рисунок 2.6 – Спектр частот для букви «а»

Після збереження звуку в дискретній формі з нього потрібно отримати характеристики про голос. Звук у формі значень амплітуд в різний момент часу важливий для аналізу, оскільки він несе інформацію про гучність звуку в момент часу. Для виділення характеристик з аудіосигналу часто аналізують частоти, які присутні в аудіосигналі. Їх можна одержати за допомогою дискретного перетворення Фур'є. Цей алгоритм перетворює послідовність комплексних чисел = $_0, _1, \dots, _{-1}$ в послідовність = $_0, _1, \dots, _{-1}$ за формулою (2.1).

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{i2\pi}{N}kn} = \sum_{n=0}^{N-1} x_n \cdot \left[\cos\left(\frac{2\pi}{N}kn\right) - i \cdot \sin\left(\frac{2\pi}{N}kn\right)\right], \quad (2.1)$$

де X – це множина комплексних чисел;

1 – це кількість аудіосигналів, а останній вираз використовує формулу Ейлера.

Якщо об’єднати цих два спектри в один, то можна отримати спектрограму, яка зображена на рисунку 2.7 (де вісь Ox – це час, Oy – це частоти, а інтенсивність кольору – це міра кількісті частоти на даному проміжку часу).

Спектрограма на рисунку 2.7 створена за допомогою фреймів зі звуків, оскільки інформація про частоти від всього проміжку аудіосигналу дає частоти для всього проміжку. Таке перетворення можна виконати із застосуванням вікна hamming. Вікно hamming застосовується для накладання фільтру на звук для зменьшення різних змін в звуці.

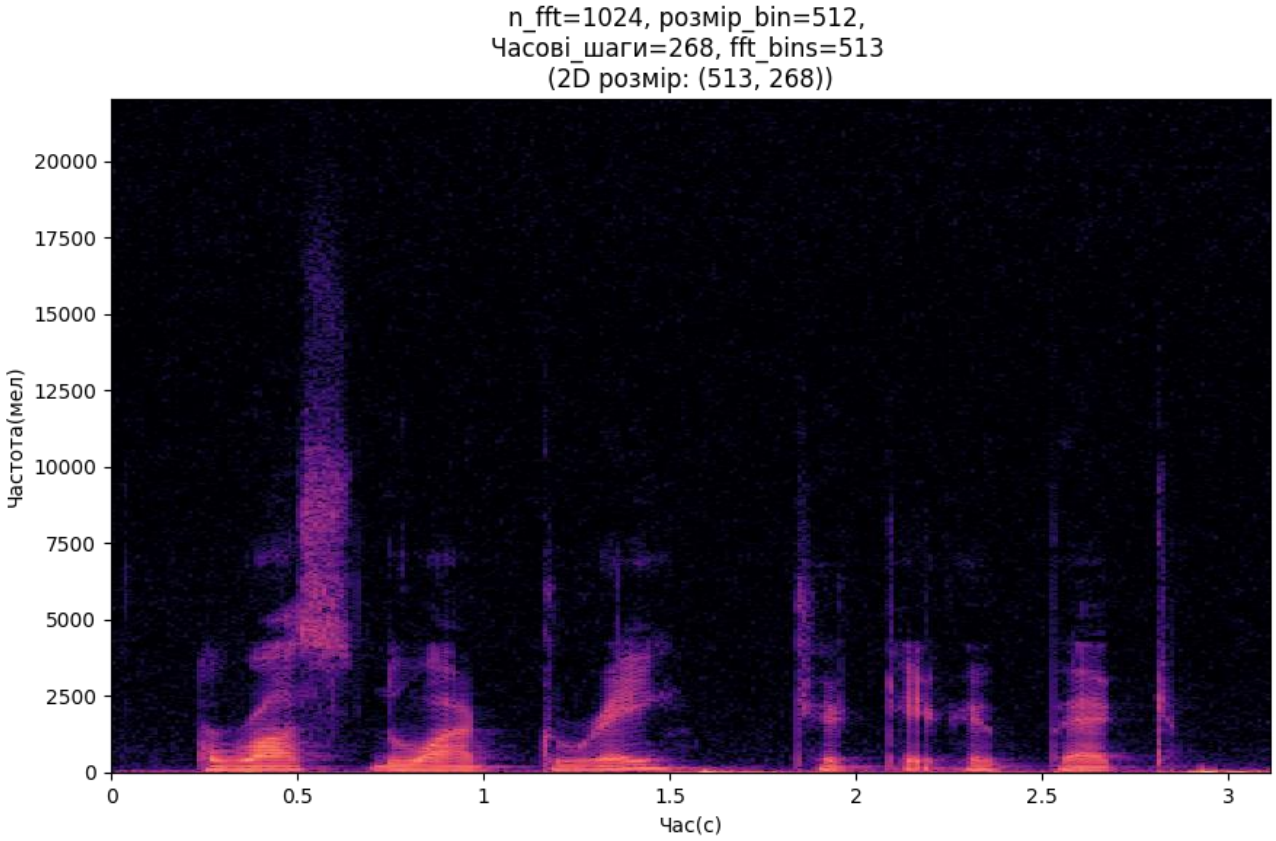



Рисунок 2.7 – Нормалізована спектрограма звук

В результаті отримується звук з плавно спадаючим звуком на початку і в кінці, що схематично зображено на рисунку 2.8, де застосовується вікно довжиною 1000 семплів до звуку, щоб створити нульові значення на кінцях семплу.

За формулую (2.2) визначається вікно hamming для обробки звуку, де дві константи відповідають на параметри вікна.

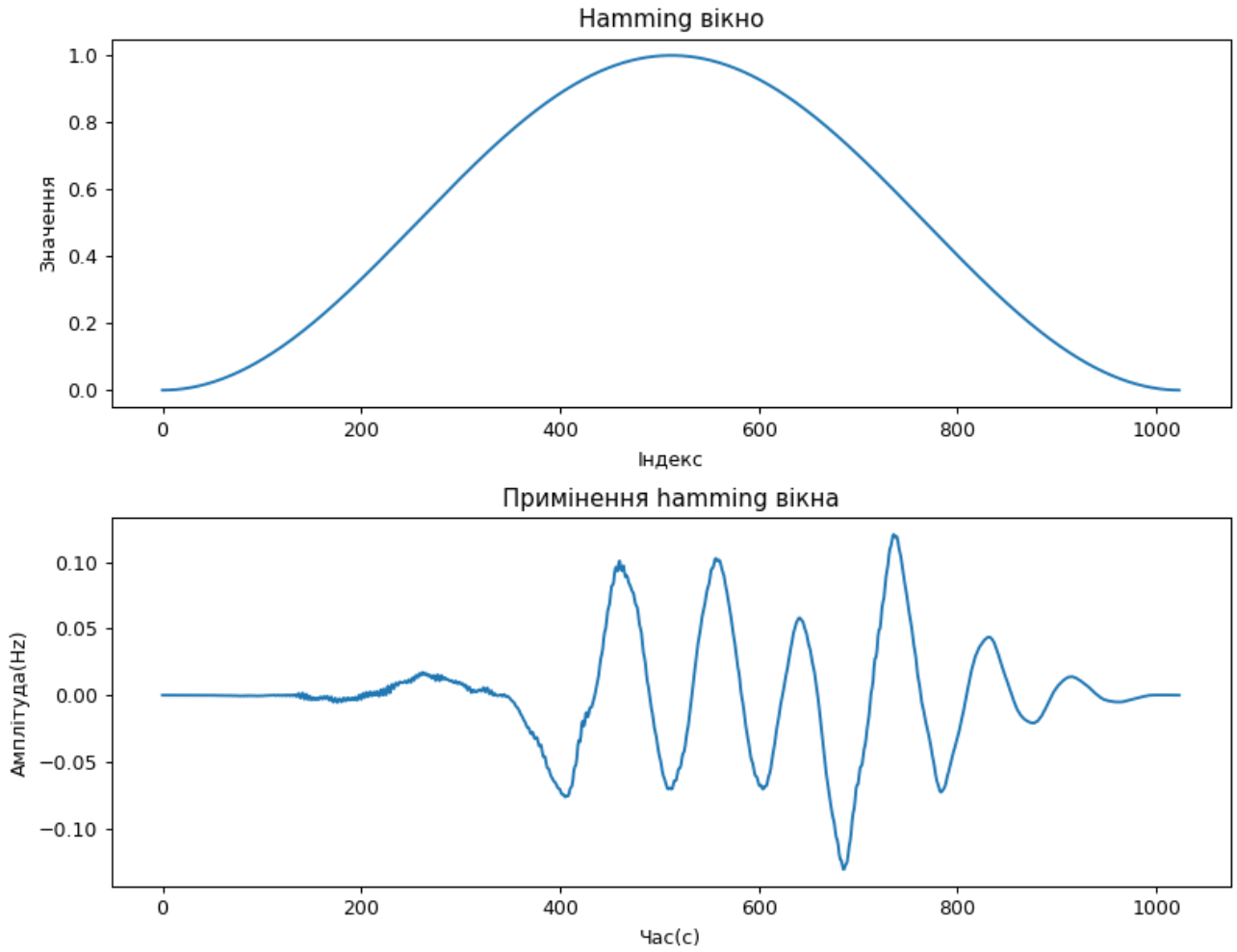



Рисунок 2.8 – Вікно hamming і звук з hamming вікном

$$H(\theta) = 0.54 + 0.46cos\left(n\frac{2*\pi}{N}\right), \tag{2.2}$$

де n – це амплітуда від 0 до 1;

1 – довжина аудіосигналу;

\+ – аудіосигнал з єфектом hamming вікна.

Після проведення ДПФ комплексні числа зручно перевести в дійсні, за допомогою формули (2.3), яка є періодограмою, що є оцінкою спектральної щільності потужності, заснованою на обчисленні квадрата модуля перетворення Фур'є послідовності даних. Ця операція потрібна для переведення списку комплексних чисел до одновимірного списку магнитуд сигналу, яку можна подати на вхід нейронній мережі.

$$P_i(k) = \frac{1}{N}|S_i(k)|^2, \tag{2.3}$$

де N – кількість аудіосигналів в фреймі;

$S_i(k)$ – значення сигнал k зі спектограми;

Як було сказано раніше, людина чує різницю в низьких частотах звуку краще, ніж у високих частотах, але ця різниця не лінійна, а логарифмічна, бо чим вище по логарифмічній шкалі частота звук, тим гірше людина може розпізнати зміни в ньому. Для переведення нормалізованої лінійної форми [22] спектрограми у логарифмічну використовуються формула (2.4)

$$m = 2595log_{10}(1 + \frac{f}{700}), \tag{2.4}$$

де f – вхідний аудіосигнал у вигляді спектограми;

P – вихідна мел-спектограми.

MFCC – це коефіцієнти, які в сукупності складають MFC. Вони засновані на нелінійному спектр-в-спектр перетворенні. Різниця між кепструмом і кепструмом з мел-частотами полягає в тому, що в MFC смуги частот однаково розташовані за шкалою розплаву, що наближає реакцію слухової системи людини краще, ніж лінійна форма частот, що використовуються в звичайній кепструмі. Це скручування частоти може забезпечити краще представлення звуку при стисненні звуку. MFCC коефіцієнти отримують наступним чином:

– віконне перетворення Фур’є;

– відображення і перетворення потужності спектру в мел-спектрограму з банком мел-фільтрів;

– застосування логарифму для кожної потужності мел-частоти;

– дискретне косинусне перетворення списку мел-частот як для сигналу;

– виділення амплітуд результуючого спектру як коефіцієнтів MFCC.

Мел-спектрограма в різних діапазонах частот має різну кількість інформації. Це важливо в процесі вилучання властивостей голосу зі звуку. На рисунку 2.9 зображена мел-спектограма звуку під час вимови букви «а».

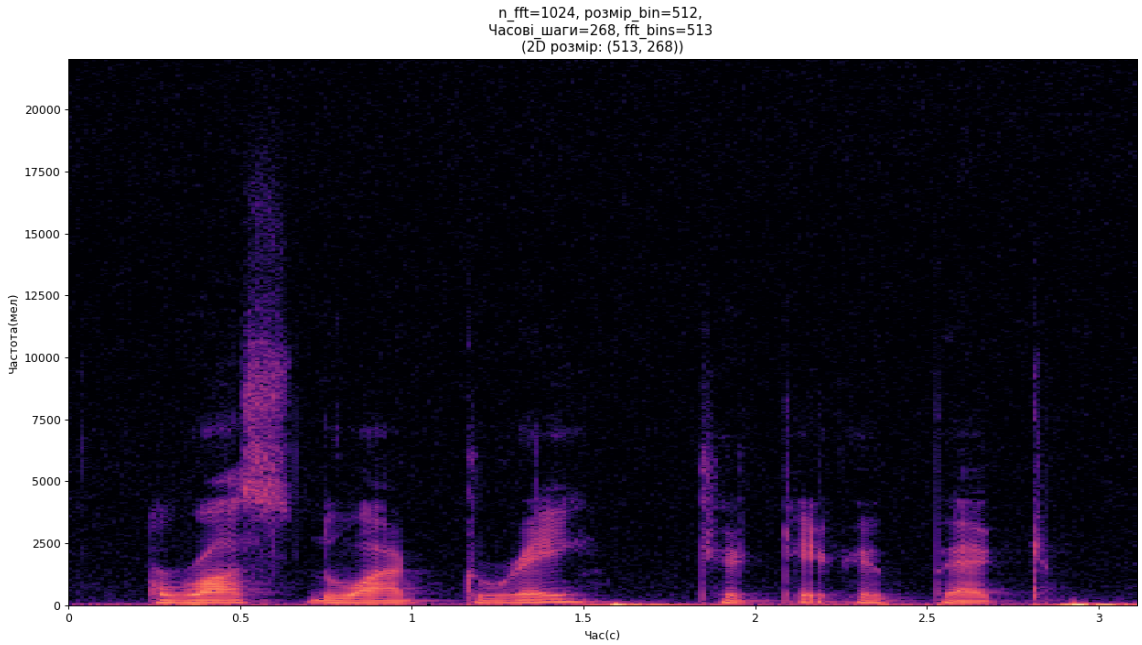



Рисунок 2.9 – Мел-спектрограма для букви «а»

На рисунку 2.10 зображені різні діапазони частот, де в нижніх частотах закодовано більше інформації, ніж в високих, що свідчить про необхідність перетворювати спектр в спектрограму без високих частоти.

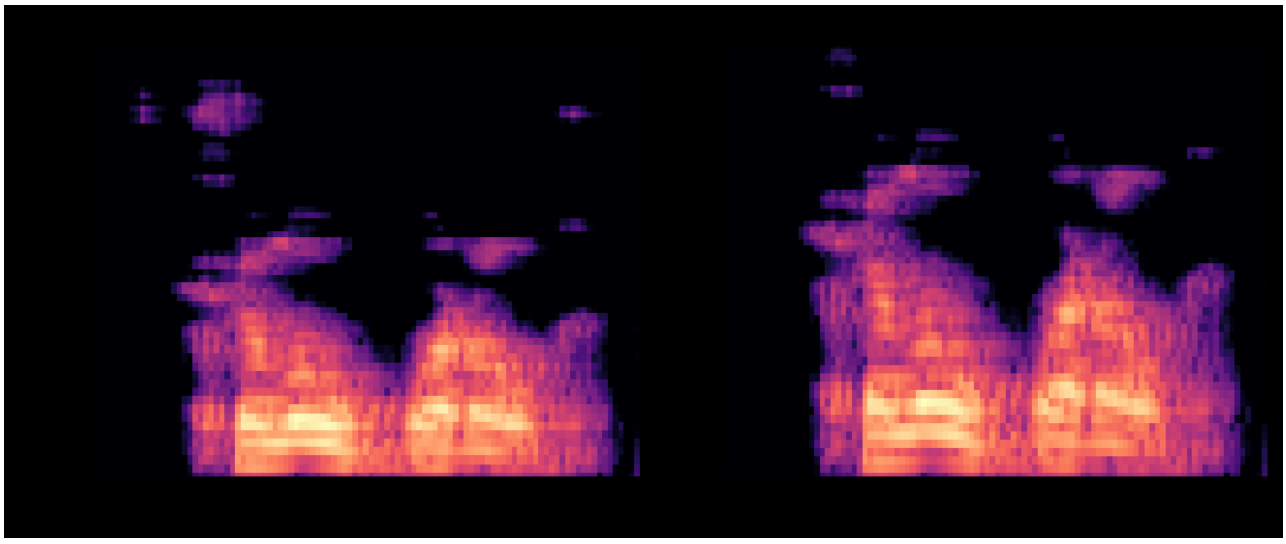


Рисунок 2.10 – Мел-спектрограма в різних діапазонах

У процесі синтезу мовлення на базі глибинних нейромереж передбачається спочатку спектрограма або мел-спектрограма за допомогою seq2seq моделі, що далі подається на глибинного нейронного вокодер для отримання синтезованої вихідної форми сигналу.

Сигнали, що повторюються через фіксовані проміжки часи, називають періодичними. Для прикладу, якщо вдарити гітару, то вона вібрує та від неї чути звук. Якщо коливання амплітуди однієї струни записати на аудіорекордер та побудувати амплітудний графік, що буде видно хвилю, як показано на рисунку 2.11, де сигнал є періодичним і має синусоїду форму. Частота цього сигналу становить близько 439 Гц, що трохи нижче 440 Гц, що є стандартним кроком настройки для оркестрової музики.

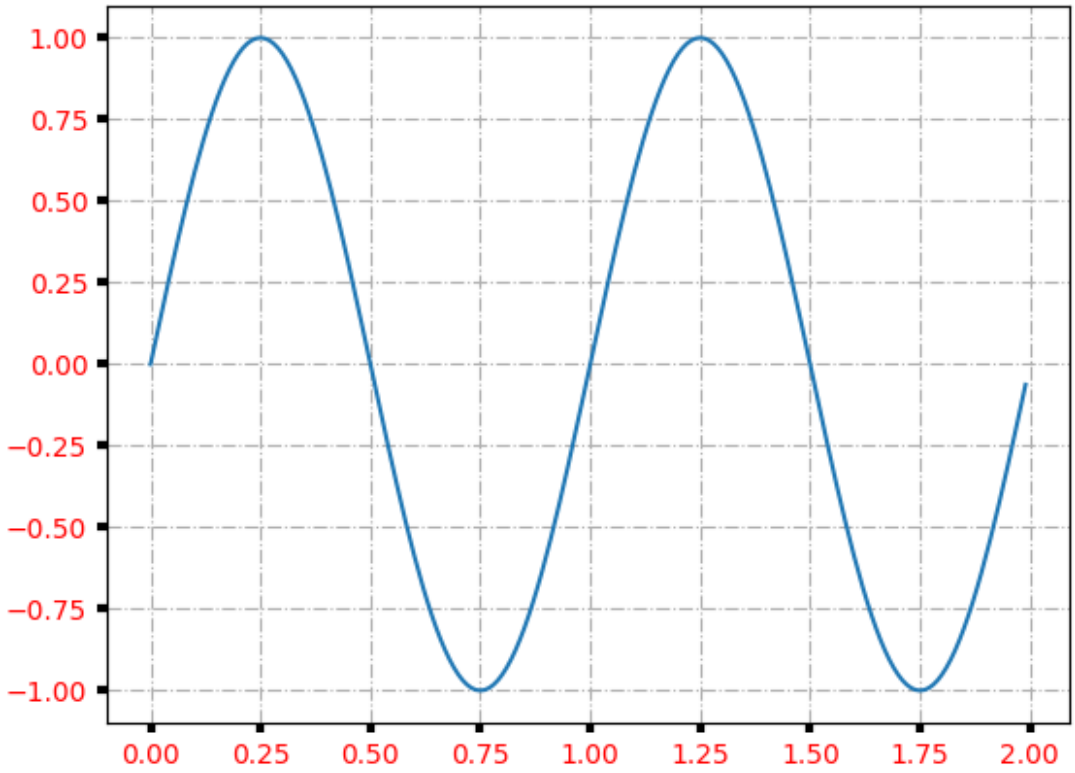



Рисунок 2.11 – Періодична хвиля

Процес перекладу звукових сигналів від неперервної форми подання до дискретної форми називають оцифровуванням.

При кодуванні звуку важливою характеристикою є частота дискретизації звуку – це кількість вимірювань рівнів сигналу за секунду:

– один вимір в секунду відповідає частоті 1 Гц;

– 1000 вимірювань в секунду відповідає частоті 1 кГц.

Кількість вимірювань гучності звуку за одну секунду є частотою дискретизації звуку. Кількість вимірювань може лежати в діапазоні від 8 кГц до 48 кГц. Якість записів з різними частотами можна оцінити як якість звука радіотрансляції та якість звучання музичних носіїв.

Чим більшими є частота і глибина дискретизації звуку, тим більшою є якість звучання оцифрованого звуку, тому що в такому звуці зберігається більше інформації про коливань звуку. Для людського слуху діапазон від 16 кГц до 24 кГц для розмови і співу сприймається досить добре.

На рисунку 2.12 показано перетворення звукового сигналу в дискретний сигнал: а – це звуковий сигнал на вході АЦП; б – дискретний сигнал на виході АЦП в комплексних чисел.

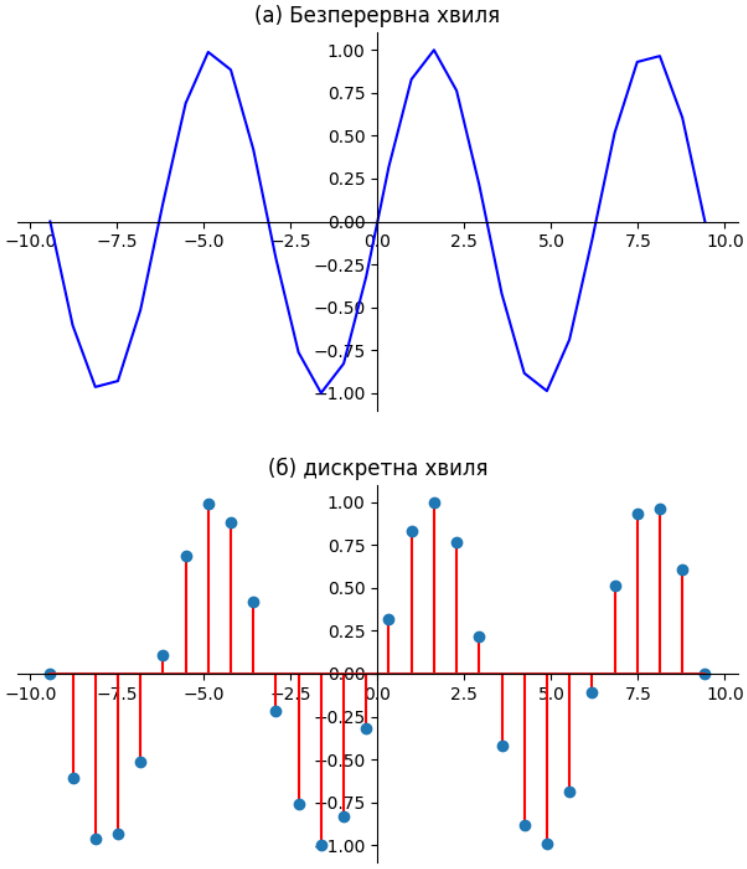



Рисунок 2.12 – Неперервна і дискретна хвилі

Найнижче якість оцифрованого звуку, відповідне якості телефонного зв’язку, виходить при частоті дискретизації 8000 раз в секунду, глибині дискретизації вісім бітів і записи однієї звукової доріжки (режим «моно»). Найвища якість оцифрованого звуку, досягається при частоті дискретизації 48 000 раз в секунду, або навіть більшу кількість разів, глибині дискретизації 16 бітів і записи двох звукових доріжок, що називають режимом «стерео». Оцінити інформаційний обсяг моноаудіофайла V можна наступним чином: V = N*f*k, де N – загальна тривалість звучання (секунд), f – частота дискретизації (Гц), k – глибина кодування (біт). Наприклад, при тривалості звучання в п’ять хвилин і середній якості звуку (8 біт, 48 кГц): V = 5*60*48000*8 біт = 115200000 біт = 14400000 байт = 14062,5 Кбайт = 13,7329 Мбайт.

При кодуванні стереозвуку процес дискретизації проводиться окремо і незалежно для лівого і правого каналів, що, відповідно, збільшує обсяг звукового файлу в два рази в порівнянні з монозвуком. Наприклад, оцінимо інформаційний обсяг цифрового файлу з тривалістю звучання в одну секунду при середній якості звуку (16 бітів, 24000 вимірювань в секунду). Для цього глибину кодування необхідно помножити на кількість вимірювань в одну секунду і помножити на два (стереозвук): V = 16 біт*24000*2 = 768000 біт = 96000 байт = 93,75 Кбайт.

Існують різні методи кодування звукової інформації двійковим кодом, серед яких можна виділити два основних напрямки: метод FM і метод Wave-Table. Метод FM (Frequency Modulation) заснований на тому, що теоретично будь-який складний звук можна розкласти на послідовність найпростіших гармонійних сигналів різних частот, кожен з яких являє собою правильну синусоїду і може бути описаний кодом. Розкладання звукових сигналів в гармонійні ряди і подання у вигляді дискретних цифрових сигналів виконують спеціальні пристрої – аналогово-цифрові перетворювачі (АЦП). Також є слова, які мають вимовляються і пишуться по-різному, тому в дискретному представленні вони можуть мати велику довжину і звучати від 200 до 1000 мілісекунд. В результаті, відношення одного токен-слова до хвилі звуку може мати велике значення.

Проблема знаходження позиції слова у дискретному представленні звуку полягає в тому, що невідомо де і як розміщюється слово в звуковому представленні. Людина може зробити паузи і змінювати інтонацію в своїх мові, і цим задача знаходження слів в звуковому представлення голосу стає більш складною. Також під час запису голосу можуть записуватись інші звуки, і такі звуки будуть шумом. Проблема отримання узагальненого представлення про синтез голосу з набому даних полягає в тому, що система синтезу має мати змогу отримати загальне представлення про текст і відповідні до слів звуки, які потрібно синтезувати. Система повинна розуміти, з одної сторони, це синтезувати правильні слова, а з іншої, як синтезувати паузи між словами. Паузу на тексті представляє знак пробілу та знаки, що не є цифрами, буквами і лапками.

Фазову реконструкцію можна провести алгоритмом Гріффіна-Ліма (GLA). Він заснований на надмірності швидкого перетворення Фур'є та сприяє узгодженню спектрограми шляхом ітерації скрізь дві проекцій, де спектрограма вважається є послідовною, коли зберігається її залежність між частками проекцій внаслідок надмірності STFT. GLA базується лише на ідеї узгодженості та не враховує жодних попередніх знань про цільовий сигнал.

## 2.2 Проектування нейронної мережі для синтезу голосу

### 2.2.1 Нейронні мережі як метод машинного навчання

Нейронна мережа прямого зв'язку складається з L шарів, де перший шар ( 1,2,.. ) застосовує нелінійне перетворення H для вхідних даних. H – параметр афінного перетворення, який має параметри у вигляді матриці з вагами $W_H^l$ і вектором зсуву з наступною функцією активації f. Функція активації f має бути диференційованою для здійснення алгоритму зворотнього поширення помилки. Кожен шар приймає вхідний вектор ( ) і генерує вектор ( ), як це видно з формули (2.5). Дана послідовна зв'язка називається повністю з'єднаний шаром.

$$y^{(l)} = f\left(H\left(x^{(l)}, W_H^{(l)}, b_H^{(l)}\right)\right) = f\left(W_H^l x^{(l)} + b_H^l\right), \quad (2.5)$$

Згорткові нейронні мережі (CNN) застосовують набір дискретних згорткових операції за допомогою яких вони можуть вивчити вхідні дані, які згорнуті за допомогою віконних функцій. Ці віконні функції називаються ядрами.

Згортки можна застосовувати для матриць будь-яких розмірів. Нехай × – це вхідні данні і × – це ядро. Операція згортки I і K записується по формулі (2.6).

$$C(i,j) = (I * K)(i,j) = \sum_{m=1}^{M}\sum_{n=1}^{N} I(i+m, j+n)K(m,n), 1 \le i \le A, 1 \le j \le B, \quad (2.6)$$

де – це оператор згортки.

В методах машинного навчання використовуються різні методи оптимізації для покращення процесу навчання нейронної мережі на різних рівнях.

Техніка Dropout – це алгоритмічний підхід, який призначений протидіяти проблемі overfitting в глибинних нейромережах мереж. Dropout призначений для поєднання різних шарів моделей в один механізм. Також цей метод використовується для багатьох архітектура, бо його досиль легко реалізувати. Ансамблі поєднують багато відокремлених мереж, усереднюючи їх результати. Це впливає на час тренування, який може зменшуватись при правильному підході. Головний принцип – випадково скидати мережеві одиниці під час навчання, і заважати їм занадто сильно адаптуватись. За допомогою тимчасового видалення блоку та всіх його вхідних та вихідних зв'язках формуються нові з'єднання в мережі. У найпростішій формі кожний нейрон тренується за допомогою фіксованої ймовірності p, де 0 < p < 1.

Нехай 1D згортка визначається як ← ( ). Параметр stride рівний двійці. Нехай оператор визначається як композиція шарів, запис ( ) = ( ( ( ))) та ( )2( ) = ( ) визначає запис мережі.

5H/8 – це функція активації, що виражається в формулою (2.7). За шарами згортки іноді слідує Highway [23] мережа, що виражається формулою (2.8),

$$ReLU(x) = max(x, 0), \tag{2.7}$$

$$Highway(X; L) = \sigma(H_1) \ominus H_2 + (1 - \sigma(H_1)) \ominus X, \tag{2.8}$$

де H1 і H2 – це матриці правильної розмірності, які є вихідним шаром з L в формі [H1, H2] = L(X);

оператор – це поелементне множення;

– це поелементне застосування функції sigmoid.

Формула (2.9) визначає HC як одновимірну згортку на базі Highway згорти.

$$HC_{k*b}^{d \leftarrow d}(X) := Highway(X; C_{k*b}^{2d \leftarrow d}), \tag{2.9}$$

де C – це згорткових шар;

O – вхідний тензор;

N G– розмір ядра і коефіцієнт розширення відповідно.

### 2.2.2 Архітектура нейронної системи для задачі озвучення тексту

В дослідженні використана архітектура, яка базується на дослідження [24]. Синтез голосу на базі нейронної мережі складається з чотирьох блоків, що наведені на рисунку 2.13. Архітектура складається з двох систем. Перша система вчить, як перекласти текстові дані в мел-спектрограми на основі чотирьох модулів, а саме Text-encoder, механізм уваги, Audio-decoder, Audio-encooder, а друга система перетворює мел-спектрограми в повний STFT спектограму за допомогою однієї глибинної мережі SSRN.

Як згадувалось раніше, аудіосигнал можна перевести в комплексну спектрограму завдяки алгоритму віконного перетворення Фур’є STFT і оберненому алгоритму STFT.

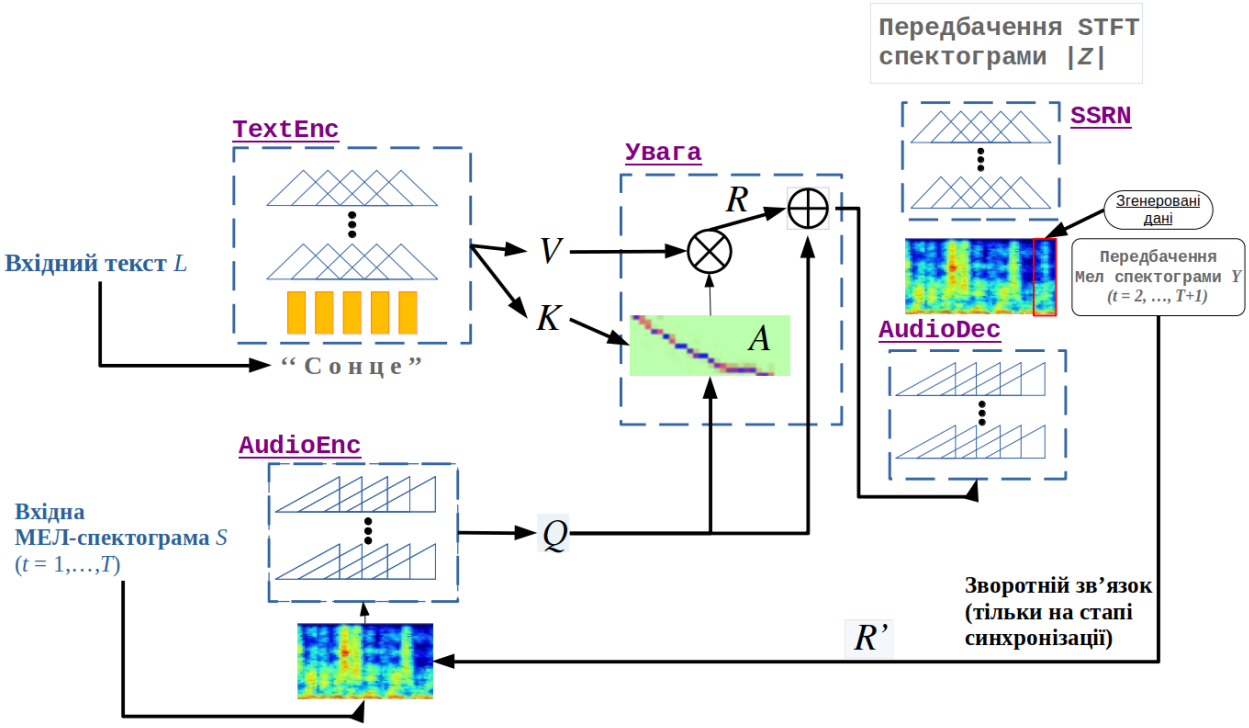



Рисунок 2.13 – Архітектура цільової моделі

Формули (2.10), (2.11), (2.12) і (2.13) визначають шари нейронної мережі Text Encoder, Audio Encoder, Audio Dencoder, Spectrogram Super-resolution Network відповідно.

$$TextEnd(L) := \left(HC_{1*1}^{2d\leftarrow 2d}\right)^2 \triangleleft \left(HC_{3*1}^{2d\leftarrow 2d}\right)^2 \triangleleft \left(HC_{3*27}^{2d\leftarrow 2d} \triangleleft HC_{3*9}^{2d\leftarrow 2d} \triangleleft HC_{3*3}^{2d\leftarrow 2d} \triangleleft HC_{3*1}^{2d\leftarrow 2d}\triangleleft^2 \triangleleft C_{1*1}^{2d\leftarrow 2d} \triangleleft ReLU \triangleleft C_{1*1}^{2d\leftarrow e} \triangleleft CharEmbed^{e-dim}(L)\right), \quad (2.10)$$

$$AudioEnd(S) := \left(HC_{3*3}^{d\leftarrow d}\right)^2 \triangleleft \left(HC_{3*27}^{d\leftarrow d} \triangleleft HC_{3*9}^{d\leftarrow d} \triangleleft HC_{3*3}^{d\leftarrow d} \triangleleft HC_{3*1}^{d\leftarrow d^2} \triangleleft C_{1*1}^{d\leftarrow d} \triangleleft ReLU \triangleleft C_{1*1}^{d\leftarrow d} \triangleleft ReLU \triangleleft C_{1*1}^{d\leftarrow F}(S)\right), \quad (2.11)$$

$$AudioDec(R') := \sigma \triangleleft C_{1*1}^{F\leftarrow d} \triangleleft \left(ReLU \triangleleft C_{1*1}^{d\leftarrow d}\right)^3 \triangleleft \left(HC_{3*1}^{d\leftarrow d}\right)^2 \triangleleft \left(HC_{3*27}^{d\leftarrow d} \triangleleft HC_{3*9}^{d\leftarrow d} \triangleleft HC_{3*3}^{d\leftarrow d} \triangleleft HC_{3*1}^{d\leftarrow d}\triangleleft\right) \triangleleft C_{1*1}^{d\leftarrow 2d}(R')\right), \quad (2.12)$$

$$SSRN(Y) := \sigma \triangleleft C_{1*1}^{F'\leftarrow F'} \triangleleft \left(ReLU \triangleleft C_{1*1}^{F'\leftarrow F'}\right)^2 \triangleleft C_{1*1}^{F'\leftarrow 2c} \triangleleft \left(HC_{3*1}^{2c\leftarrow 2c}\right)^2 \triangleleft C_{1*1}^{2c\leftarrow c} \triangleleft \left(HC_{3*3}^{c\leftarrow c} \triangleleft HC_{3*1}^{c\leftarrow c} \triangleleft D_{2*1}^{c\leftarrow c}\triangleleft\right)^2 \triangleleft \left(HC_{3*3}^{c\leftarrow c} \triangleleft HC_{3*1}^{c\leftarrow c}\triangleleft\right) \triangleleft C_{1*1}^{c\leftarrow F}(Y)\right), \quad (2.13)$$

де C, H, HC – це згортки з розділу 2.2.1;

– це шар перетворення символів в числа;

Техніка Dropout і Layer normalization застосовується для всіх шарів нейромереж, де є highway блоки.

Text Encoder кодує вхідне речення = [ 1, … , ] , яка складається з *N* символів в дві матриці , × , що визначає мережу ( , ) = ( ). Нейронна глибинна мережа Audio Encoder кодує мел-спектрограму (= 1: ,1: ) × аудіофайлу з довжиною *T* в матрицю × , що визначає мережу = ( 1: ,1: ). Матриця уваги A × визначає міру впіввідношення *n*-го символ в послідовності на *t*-й тимчасовій рамці 1: ,1: за формулую (2.14). Якщо 1, то це означає, що модуль розглядає *n*-й символ послідовності у часовому інтервал *t* та він буде сприймат або + 1 символи навколо них у наступній часовій рамці *t* + 1. Незважаючи на це, очікується, що дані кодуються в n-му стовпці матриці V. Таким чином, клітинка матриці уваги × , декодована до наступних кадрів 1: ,2: +1, отримується так: = ( , , ) = .

$$A = softman_{x-axis}(K^T Q/\sqrt{(}d)), \tag{2.14}$$

де softmax – нормована експоненційна функція.

Далі результуюча матриця R поєднується із закодовиним звуком Q, як ′ = [ , ]. Матриця ′ 2 × декодується в грубу мел-спектограму як 1: ,2: +1 = ( ′). Результат 1: ,2: +1 порівнюється з аудіофайлом 1: ,2: +1 завдяки цільовій функції ( 1: ,2: +1| 1: ,2: +1), де помилка враховується в процесі зворотнього поширення. Функція помилки – це сума втрати L1 та бінарна дивергенції за формулую (2.15).

$$\begin{aligned} D_{bin}(Y|S) &= E_{ft}\left[-S_{ft}log(T_{ft}) - (1-S_{ft})log(1-Y_{ft})\right] \\ &= E_{ft}\left[-S_{ft}Y_{ft} + log(1+exp\widehat{Y}_{ft})\right], \end{aligned} \tag{2.15}$$

де ̂ = ( ).

Далі синтезується повна спектограма | | ′×4 з мел-спектограми *Y*. Збільшення розмірності частоти *F* до ′ досить просте. Для цього збільшується

канали згортки *1D*. Збільшення розмірності в часовому напрямоку робиться іншим чином, але завдяки подвійному застосуванням зворотньої згортки з параметром stride = 2 можна в чотири рази збільшити довжину послідовності з $T$ до $4T = {}'$.

## 2.3 Проектування соціальної платформи

Опишемо роботу і склад соціальної платформи, яку потрібно розробити. Соціальна платформа складається з таких класів, як:

– користувач;
– роль користувача;
– оголошення;
– коментар до оголошення;
– аудіооголошення;
– аудіосесій.

Саму платформу може використовувати будь-яка людина, але платформа має певні ролі, а саме:

– гість (людина, яка ще не зареєструвалась на платформі);
– звичайний користувач (людина, яка зареєструвалась на платформі і має на ній базові права);
– спікер (людина, яка зареєструвалась на платформі і має хоча б одне оголошення, яке повністю озвучене ним);
– модератор (людина, яка має права на редагування коментарів і оголошень користувачів);
– адміністратор (людина, яка має повні права і може змінювати будь-які дані на платформі).

В базові права користувача входить:

– можливість коментувати свої оголошення;
– можливість коментувати оголошення інших користувачів;
– писати коментарі під оголошеннями;

– озвучувати свої оголошення;

– озвучувати оголошення інших користувачів.

На платформі потрібен модератор для цензури, бо цільова соціально-орієнтована платформа має різні можливості для зловживання її сервісом, а саме:

– спам;

– завантаження вірусів чи шкідливого коду в текст оголошення;

– надання недостовірних даних сервісу;

– нецензурна лексика в адрес користувача;

– публікація заборонених матеріалів.

Адміністратор виступає в ролі розробника платформи з можливістю змінювати дані на платформі (назви заголовків, списків, контактної інформації) і правами модератора.

Спікер – це найголовна роль користувача на платформі, оскільки саме цей користувач озвучує тексти оголошень. Роль спікера створена для фільтрації користувачів та швидкого пошуку.

Якщо спікер хоче створити оголошення, то він має зробити наступні дії:

– написати текст оголошення у вигляді unicode тексту;

– перейти на власну сторінки на платформі;

– опублікувати текстом, вказавши додаткові деталі оголошення.

Якщо спікер хоче озвучити оголошення, то він має зробити наступні дії:

– обрати оголошення;

– створити аудіосесію;

– перейти в редактор;

– провести озвучення.

Користувача може стати спікером в декількох випадках:

– попросити адміністратора видати роль спікера;

– повністю озвучити своє оголошення, яке має більше 10 блоків;

– повністю озвучити оголошення іншого користувача, яке має більше 10 блоків тексту.

Редактор – це та частина платформи, де користувачі озвучують текст. Цей редактор має мати зрозумілий інтерфейс для маніпуляції записаного звуку і відповідному йому тексту.

Редактор повинен мати наступні функції:

– озвучити речення;
– почути речення;
– видалити речення;
– експорт всіх озвучених речень в набір даних;
– експорт вибраних речень в набір даних;
– пошук по записаним реченнями.

Як було згадано раніше, щоб отримати профіль користувача потрібно зареєструватись. Реєстрація не є обов’язковою на платформі, тому гість може залишити коментар під оголошенням і автором цього коментаря буде спеціальний анонімний користувач. Під час реєстрації користувач вказує про дані двох типів: обов’язкові і вибіркові. До обов’язкових полів відносяться:

– ім’я;
– логін;
– пароль;
– пошта.

До вибіркових полів відносяться:

– геолокація (дане поля є текстом, а не фіксованим вибором, тому в ньому можна не вказувати точне місце знаходження, а лише приблизне);
– вік.

Оголошення – це текст, який завантажується користувачем у вигляді блоків unicode тексту. Для озвучення оголошення його потрібно розділити на блоки. Кожний блок створюється шляхом розділення цілого тексту на окремі речення.

Неформатований текст оголошення є важливою частиною оголошення, бо на базі цього тексту воно буде пропонувати автоматичне розділення тексту на окремі блоки для озвучення. Якщо видані блоки будуть некоректно розділені, то користувачу потрібно буде розділити текст власноруч. Якщо користувач не хоче

автоматично розділяти текст, то він може зразу перейти до ручного розділення. Під час поділу текст кожний окремий блок має складатись з одного речення тексту. Така рекомендація дозволяє створювати речення, які мають нормальну довжину (від семи до 10 слів), що дозволяє їх озвучувати і не створювати помилки під час озвучення. Якщо блок має два і більше речення в собі, то це збільшує шанс помилкового озвучення блока і негативно впливає на загальний процес озвучення спікера.

Будь-який користувач може коментувати свої оголошення і оголошення інших користувачів. Під час створення озвучення користувач одночасно створює аудіосесію. Аудіосесія є зв’язкою між оголошеннями і файлами озвучення, на основі якою можна контролювати і організовувати різні озвучення одного користувача. Наприклад, якщо оголошення – це популярна книга серед користувачів, то вона може мати аудіоверсію в декількох різних людей. За одну сесію можна озвучити тільки одну книгу. Діаграма класів для бази даних наведена на рисунку 2.14.

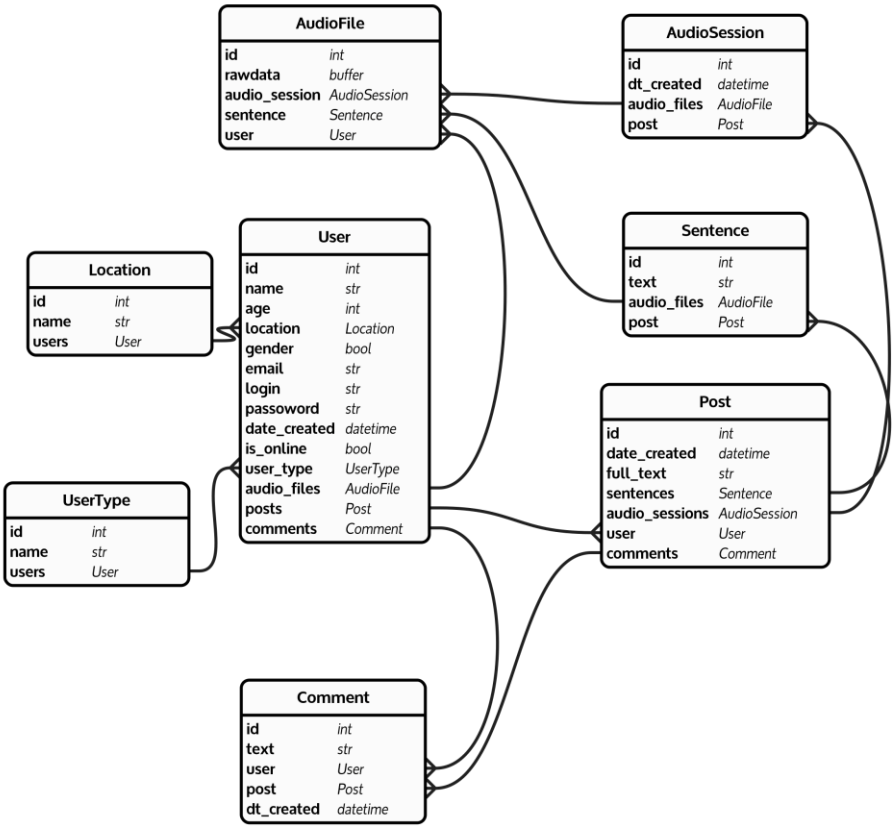



Рисунок 2.14 – Діаграма класів для бази даних веб-сервера

# 3 ПРОЕКТУВАННЯ КАРКАСУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## 3.1 Розробка моделі для синтезу голосу

Для побудови моделей нейронних мереж використана бібліотека torch. Окремі мережі реалізовані в якості чотирьох функцій і однієї функції уваги: textencoder, audioenc, attention, audiodec, SSRN.

Гіперпараметри моделі для обробка сигналів:

– sample rate = 22050, дискретизація звуку;
– n_fft = 2048, кількість fft блоків;
– n_mels = 80, розмір мел-банку для стиснення аудіо;
– preemphasis = .95, коефіцієнт обробки;
– ref db = 20, нормалізований звук.

Гіперпараметри моделі для моделей:

– dropout rate = 0.1, імовірність dropout rate;
– e = 128, розмір embedding шару;
– d = 256, кількість приховані шарів мережі Text2Mel;
– c = 512, кількість приховані шарів мережі SSRN;
– vocab, корпус моделі (літери англійської алфавіту без цифр);
– max N = 200-10, максимальне число символів;
– max T = 200+10, максимальне число мел-кадрів.

Гіперпараметри для тренування:

– number iterations = 2500000, максимальна кількість ітерацій;
– learning rate = 0.001, початковий крок навчання;
– batch = 32, розмір batch.

## 3.2 Побудова нейронних мереж

Вхідні дані проходять етап нормалізації, де з тексту виділяється всі літери, яких немає в корпусі. Далі проходить навчання мереж в два етапи, де спочатку

тренується модуль з мережею text-encoder і мережею SSRN. Вхідні дані потрібно подавати в стилі набору даних LJSpeech, а саме один файл-таблиця, де перша колонка відповідає назві файлу без його розширення, друга колонка – це текст, третя колонка – нормалізований текст. Аудіофайли бажано зберігати в форматі .wav. Перед тренуванням аудіофайли перетворюються в спеціальних вектор, який потрібний у вхідному кроці тренування.

Обробкою аудіо перед створення спеціального вектора є послідовне застосування алгоритмів: pre emphasis, linear scale, mel spectrogram, hertz to decibel, normalize, transpose.

В програмі розроблені дві системи нейромереж і декілька модулів нейромереж. Тренувальний цикл для навчання модулів зображений на рисунку 3.1. Text-to-Mel і SSRN – це системи нейромереж, а text-encoder, audio-encoder, audio-decoder, ssrn – це нейромережеві модулі, структура який зображена на рисунках 3.2-3.5.

```
for loop_cnt in range(int(Hyper.num_batches / batch_maker.num_batches() + 0.5)):
    print("loop", loop_cnt)
    bar = PrettyBar(batch_maker.num_batches())
    bar.set_description("training...")
    loss_str0 = MovingAverage()
    loss_str1 = MovingAverage()

    for bi in bar:
        batch = batch_maker.next_batch()
        #low res | high res
        mels,      mags = torch.FloatTensor(batch["mels"]).to(device), \
                          torch.FloatTensor(batch["mags"]).to(device)
        #forward
        mag_logits, mag_pred = graph(mels)
        #loss
        loss_mags = criterion_mags(mag_pred, mags)
        loss_bd2 = criterion_bd2(mag_logits, mags)
        loss = loss_mags + loss_bd2

        #backward
        graph.zero_grad(); optimizer.zero_grad(); loss.backward()
        #clip grad
        nn.utils.clip_grad_value_(graph.parameters(), 1)
        optimizer.step()

        #log
        loss_str0.add(loss_mags.cpu().data.mean())
        loss_str1.add(loss_bd2.cpu().data.mean())
        lossplot_mags.add(loss_str0(), global_step)
        lossplot_bd2.add(loss_str1(), global_step)
        bar.set_description("gs: {}, mags: {}, bd2: {}".format(global_step,
                            loss_str0(), loss_str1()))
```

Рисунок 3.1 – Тренувальний цикл

```
TextEncoder(
  (seq_): Sequential(
    (char-embed): CharEmbed(75, 128, padding_idx=0)
    (conv_0): MaskedConv1d(128, 512, kernel_size=(1,), stride=(1,))
    (relu_0): ReLU()
    (drop_0): Dropout(p=0.05, inplace=False)
    (conv_1): MaskedConv1d(512, 512, kernel_size=(1,), stride=(1,))
    (drop_1): Dropout(p=0.05, inplace=False)
    (highway-conv_2): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_2): Dropout(p=0.05, inplace=False)
    (highway-conv_3): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(3,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_3): Dropout(p=0.05, inplace=False)
    (highway-conv_4): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(9,), dilation=(9,)
      (sigmoid_): Sigmoid()
    )
    (drop_4): Dropout(p=0.05, inplace=False)
    (highway-conv_5): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(27,), dilation=(27,)
      (sigmoid_): Sigmoid()
    )
    (drop_5): Dropout(p=0.05, inplace=False)
    (highway-conv_6): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_6): Dropout(p=0.05, inplace=False)
    (highway-conv_7): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(3,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_7): Dropout(p=0.05, inplace=False)
    (highway-conv_8): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(9,), dilation=(9,)
      (sigmoid_): Sigmoid()
    )
    (drop_8): Dropout(p=0.05, inplace=False)
    (highway-conv_9): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(27,), dilation=(27,)
      (sigmoid_): Sigmoid()
    )
    (drop_9): Dropout(p=0.05, inplace=False)
    (highway-conv_10): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_10): Dropout(p=0.05, inplace=False)
    (highway-conv_11): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_11): Dropout(p=0.05, inplace=False)
    (highway-conv_12): HighwayConv1d(
      512, 1024, kernel_size=(1,), stride=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_12): Dropout(p=0.05, inplace=False)
    (highway-conv_13): HighwayConv1d(
      512, 1024, kernel_size=(1,), stride=(1,)
      (sigmoid_): Sigmoid()
    )
  )
)
```

Рисунок 3.2 – Архітектура модуля text-encoder

```
AudioEncoder(
  (seq_): Sequential(
    (conv_0): MaskedConv1d(80, 256, kernel_size=(1,), stride=(1,))
    (relu_0): ReLU()
    (drop_0): Dropout(p=0.05, inplace=False)
    (conv_1): MaskedConv1d(256, 256, kernel_size=(1,), stride=(1,))
    (relu_1): ReLU()
    (drop_1): Dropout(p=0.05, inplace=False)
    (relu_2): MaskedConv1d(256, 256, kernel_size=(1,), stride=(1,))
    (drop_2): Dropout(p=0.05, inplace=False)
    (highway-conv_3): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(2,)
      (sigmoid_): Sigmoid()
    )
    (drop_3): Dropout(p=0.05, inplace=False)
    (highway-conv_4): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(6,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_4): Dropout(p=0.05, inplace=False)
    (highway-conv_5): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(18,), dilation=(9,)
      (sigmoid_): Sigmoid()
    )
    (drop_5): Dropout(p=0.05, inplace=False)
    (highway-conv_6): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(54,), dilation=(27,)
      (sigmoid_): Sigmoid()
    )
    (drop_6): Dropout(p=0.05, inplace=False)
    (highway-conv_7): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(2,)
      (sigmoid_): Sigmoid()
    )
    (drop_7): Dropout(p=0.05, inplace=False)
    (highway-conv_8): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(6,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_8): Dropout(p=0.05, inplace=False)
    (highway-conv_9): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(18,), dilation=(9,)
      (sigmoid_): Sigmoid()
    )
    (drop_9): Dropout(p=0.05, inplace=False)
    (highway-conv_10): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(54,), dilation=(27,)
      (sigmoid_): Sigmoid()
    )
    (drop_10): Dropout(p=0.05, inplace=False)
    (highway-conv_11): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(6,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_11): Dropout(p=0.05, inplace=False)
    (highway-conv_12): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(6,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
  )
)
```

Рисунок 3.3 – Архітектура модуля audio-encoder

```
AudioDecoder(
  (seq_): Sequential(
    (conv_0): MaskedConv1d(512, 256, kernel_size=(1,), stride=(1,))
    (drop_0): Dropout(p=0.05, inplace=False)
    (highway-conv_1): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(2,)
      (sigmoid_): Sigmoid()
    )
    (drop_1): Dropout(p=0.05, inplace=False)
    (highway-conv_2): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(6,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_2): Dropout(p=0.05, inplace=False)
    (highway-conv_3): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(18,), dilation=(9,)
      (sigmoid_): Sigmoid()
    )
    (drop_3): Dropout(p=0.05, inplace=False)
    (highway-conv_4): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(54,), dilation=(27,)
      (sigmoid_): Sigmoid()
    )
    (drop_4): Dropout(p=0.05, inplace=False)
    (highway-conv_5): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(2,)
      (sigmoid_): Sigmoid()
    )
    (drop_5): Dropout(p=0.05, inplace=False)
    (highway-conv_6): HighwayConv1d(
      256, 512, kernel_size=(3,), stride=(1,), padding=(2,)
      (sigmoid_): Sigmoid()
    )
    (drop_6): Dropout(p=0.05, inplace=False)
    (conv_7): MaskedConv1d(256, 256, kernel_size=(1,), stride=(1,))
    (relu_7): ReLU()
    (drop_7): Dropout(p=0.05, inplace=False)
    (conv_8): MaskedConv1d(256, 256, kernel_size=(1,), stride=(1,))
    (relu_8): ReLU()
    (drop_8): Dropout(p=0.05, inplace=False)
    (conv_9): MaskedConv1d(256, 256, kernel_size=(1,), stride=(1,))
    (relu_9): ReLU()
    (drop_9): Dropout(p=0.05, inplace=False)
    (conv_10): MaskedConv1d(256, 80, kernel_size=(1,), stride=(1,))
  )
)
```

Рисунок 3.4 – Архітектура модуля audio-decoder

```
SuperRes(
  (seq_): Sequential(
    (conv_0): MaskedConv1d(80, 512, kernel_size=(1,), stride=(1,))
    (drop_0): Dropout(p=0.05, inplace=False)
    (highway-conv_1): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_1): Dropout(p=0.05, inplace=False)
    (highway-conv_2): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(3,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_2): Dropout(p=0.05, inplace=False)
    (deconv_3): Deconv1d(512, 512, kernel_size=(2,), stride=(2,))
    (drop_3): Dropout(p=0.05, inplace=False)
    (highway-conv_4): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_4): Dropout(p=0.05, inplace=False)
    (highway-conv_5): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(3,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_5): Dropout(p=0.05, inplace=False)
    (deconv_6): Deconv1d(512, 512, kernel_size=(2,), stride=(2,))
    (drop_6): Dropout(p=0.05, inplace=False)
    (highway-conv_7): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_7): Dropout(p=0.05, inplace=False)
    (highway-conv_8): HighwayConv1d(
      512, 1024, kernel_size=(3,), stride=(1,), padding=(3,), dilation=(3,)
      (sigmoid_): Sigmoid()
    )
    (drop_8): Dropout(p=0.05, inplace=False)
    (conv_9): MaskedConv1d(512, 1024, kernel_size=(1,), stride=(1,))
    (drop_9): Dropout(p=0.05, inplace=False)
    (highway-conv_10): HighwayConv1d(
      1024, 2048, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_10): Dropout(p=0.05, inplace=False)
    (highway-conv_11): HighwayConv1d(
      1024, 2048, kernel_size=(3,), stride=(1,), padding=(1,)
      (sigmoid_): Sigmoid()
    )
    (drop_11): Dropout(p=0.05, inplace=False)
    (conv_12): MaskedConv1d(1024, 1025, kernel_size=(1,), stride=(1,))
    (drop_12): Dropout(p=0.05, inplace=False)
    (conv_13): MaskedConv1d(1025, 1025, kernel_size=(1,), stride=(1,))
    (relu_13): ReLU()
    (drop_13): Dropout(p=0.05, inplace=False)
    (conv_14): MaskedConv1d(1025, 1025, kernel_size=(1,), stride=(1,))
    (relu_14): ReLU()
    (drop_14): Dropout(p=0.05, inplace=False)
    (conv_15): MaskedConv1d(1025, 1025, kernel_size=(1,), stride=(1,))
  )
  (sigmoid_): Sigmoid()
)
```

Рисунок 3.5 – Архітектура модуля SSRN

## 3.3 Підготовка вхідних даних

Для створення моделей потрібно визначити алфавіт мови і відповідний набір даних. Щоб створити базову модель для англійської моделі використаний набір даних LJSpeech. Оскільки набір даних англомовний, то для нього необхідні англійські речення.

Перед створення коректної моделі для певної люди був створений власний набір даних зі 178 речень, тривалістю від 0,7 до 9,7 секунд. Для створення англійської і української моделі побудований набір даних, який склад з 200 речень. Речення озвученні автором роботи і створені на базі спеціального корпусу. Корпус складається з власних речень в яких присутні 500 найчастіших слів кожної мови.

## 3.4 Розробка соціальної платформи

Вся соціальна платформа складається з окремих модулів. Загальні функціональні модулі соціальної платформи зображені на рисунку 3.5.

За допомогою веб-сервера користувачі повноцінно взаємодіють з іншими користувачами через оголошення і коментарі за допомогою клієнтської програми і публічного API, який потрібен для розробників програмного забезпечення, які бажають використовувати протоколи спілкування з сервером. Для озвучення і розмітки тексту користувачу надається спеціальний сайт.

Вихідний код цього сайту можна завантажити та запустити локально для вільного використання. Для покращення інтерфейсу взаємодії з користувачами мережі Інтернет, потрібно мати веб-версію інструментів маркування. За допомогою двох програм користувач може розміщувати дані тексту і аудіо. В першій програмі можна озвучити речення, а друга програма вирівнює речення до звуку. Другу програму можна паралельно використовувати під час маркування з іншими користувачами, що є головною перевагою цього методу, оскільки швидче маркувати існуючий звук, ніж записувати новий.

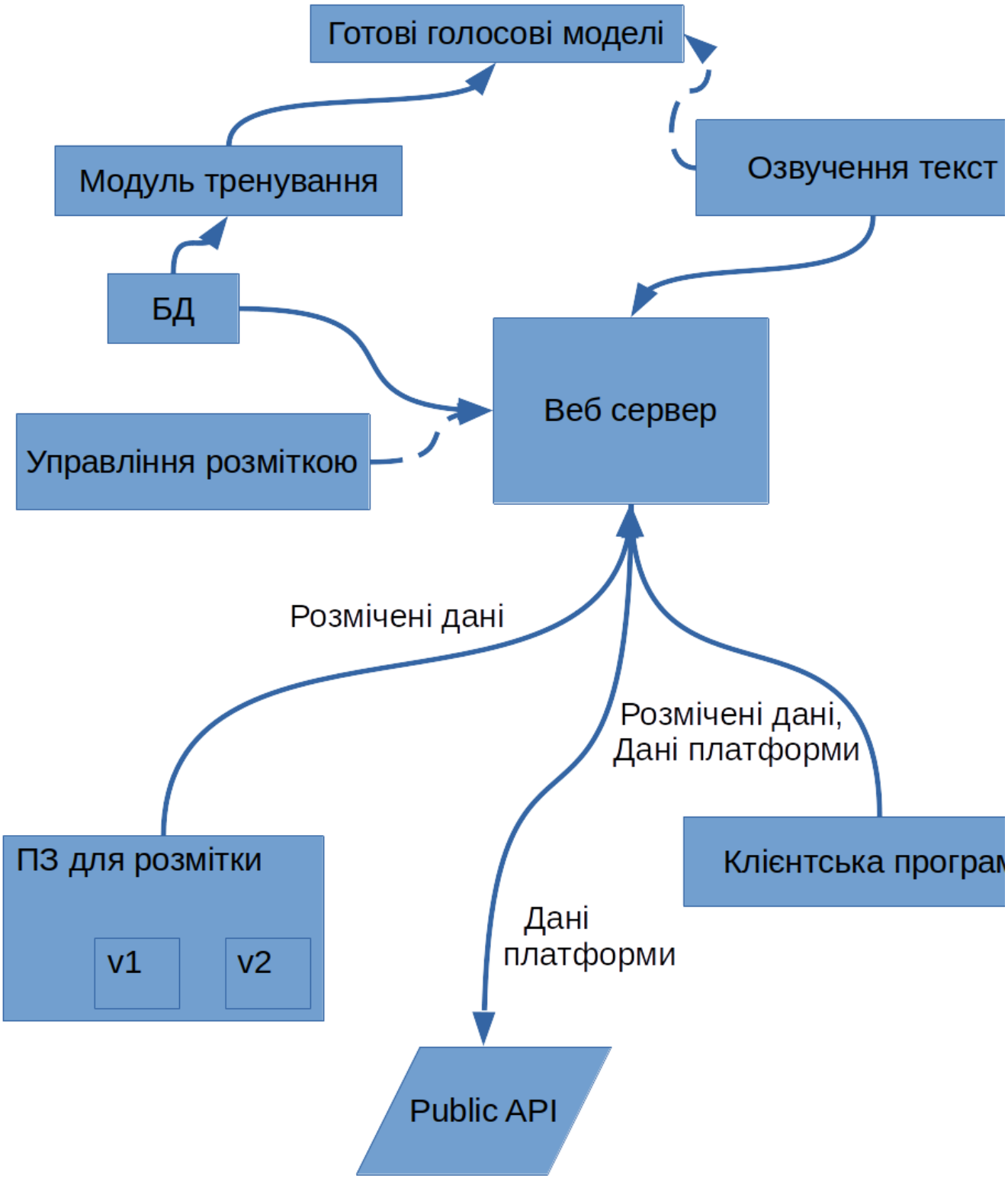



Рисунок 3.5 – Модулі соціальної платформи

## 3.5 Створення архітектури веб-серверу

Архітектура веб-серверу потрібна для відокремлення робіт по озвученню текстів і менеджменту веб-запитів.

Побудова платформи для спілкування потребує таких компонентів, як:

– база даних;
– вузол контролера;
– шаблони відображення;
– тимчасове сховище.

Тимчасове сховище може бути модулем кешування, що використовується для швидкого пошуку або видачі нещодавно отриманої інформації. Шаблони відображення потрібні для гнучкої зміни зовнішнього вигляду веб-сторінок, як-то інтерфейс маркування або озвучення тексту. Вузол контролера потрібний для видачі потрібних аудіофайлів з текстом і наборів даних, наданням інструментів для створення наборів даних, реєстрації оброблених даних і аутентифікації та авторизації користувачів, проведення озвучення тексту. Усі описані модулі зображені на рисунку 3.6.

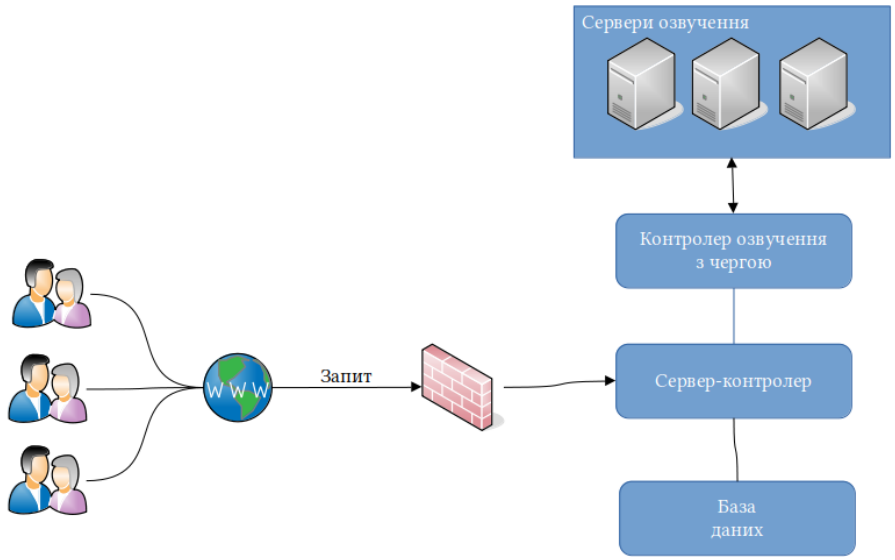



Рисунок 3.6 – Архітектура моделі веб-серверу

# 4 РОЗРОБКА І ТЕСТУВАННЯ ПРОГРАМНОГО КОМПЛЕКСУ

## 4.1 Вибір інструментів для реалізації нейронної мережі і платформи

Для виконання проекту обрана мова програмування Python 3 – динамічно-типізована мова програмування з підтримкою об’єктно-орієнтованої парадигми. Дана мова є популярної серед розробників і дослідників нейронних мереж, машинного навчання, математичного моделювання. Основна мета цього розділу полягає в описі тестування ПЗ і описі модулів, які обрані для розробники ПЗ.

Під час розробки ПЗ були проаналізовані і обрані бібліотеки для розробки нейронної мережі, обробки звуку, побудови серверу, створення бази даних, створення графічного інтерфейсу.

Для створення нейромереж мовою Python існують декілька бібліотек, а саме:

– pytorch;
– tensorflow;
– keras;

Модуль tensorflow – це розробка компанії Google представляє собою програмний модуль для задач машинного навчання, серед яких є нейронні мережі. Розробники TensorFlow створили бібліотеку з декількома рівнями абстракції, які можна вибрати під свої потреби. Це можливо завдяки інтерфейсу Keras, який є надбудовою над tensorflow і його можна використовувати для оперування різними рівнями розробки мереж.

В бібліотеці можна створювати власні цільові функції, оптимізатори, функції активації тощо. Для проектів великого розміру, яким необхідні специфічні нейронні мережі, існує спеціальна API стратегія для розподіленого самого навчання на багатьох машинах, які можуть мати різні технічні характеристики, що не потребують зміни конфігурації визначеної моделі.

Перший стабільний випуск бібліотеки був в 2015 році. Бібліотека має вільну ліцензію Apache 2.0, надає готові методи і алгоритми для обчислення. Нейронні мережі і операції з ними реалізовані різнотипними тензорами. Операції можуть

бути складними, що є функціями з багатьма параметрами, так і простими, як додавання чи множення.

PyTorch – це багатоплатформена модель для розробки і тренування нейронних мереж, яка створена командую FAIR(Facebook AI Research lab) у 2016 році мовою програмування CUDA, C++, Python. Вона має ліценцію BSD, що дозволяє вільно її використовувати у будь-яких проектах. Сама бібліотека є розвитком проекту Torch, який стартував у 2002 році, а також інтеграцією з бібліотекою Caffe. Бібліотека оперує тензорами, надає простий API для створення своїх низькорівневих функцій і механізм зворотніх викликів, які потрібні при тренуванні і оцінці моделей під час навчання.

Keras – це модуль з API для глибокого навчання, який надає зручний інтерфейс для TensorFlow. Основна ідея модуля полягає в швидкій побудові моделей і можливості виконувати експерименти з ними.

Окрім головного API для нейронних мереж, Keras також має велику екосистему інструментів для розробника. В розробці нейромереж ці інструменти охоплюють широкий спектр задач, а також переводять кожний крок робочого процесу в процесі візуальне представлення за допомогою TensorBoard.

У бібліотеці keras входять інструменти:

– Keras Tuner;
– AutoKeras;
– Хмара TensorFlow;
– TensorFlow.js;
– TensorFlow Lite.

Keras Tuner – система для масштабованої оптимізації гіперпараметрів, що допомагає знайти потрібні параметри для моделі завдяки зручному інтерфейсу. Система AutoML під назвою AutoKeras створена на Keras. Вона розроблена лабораторією DATA і робить машинне навчання доступним широкому колу людей. Набір утиліт TensorFlow Cloud від Google допомагають легко запустити масштабні задачі для Keras на GCP. Для веб-розробників існує javascript бібліотека під назвою TensorFlow.js, яка здатна запускати збережені і натренована моделі TensorFlow у

браузері чи на сервері. Для вбудованих, великих і малих систем (до яких відносяться мобільні телефони) існує TensorFlow Lite, який підходить для багатьох пристроїв, де наявна підтримка моделей на Keras.

Для задачі побудови нейронної мережі в даному дослідженні використана бібліотека torch, тому що вона має достатній рівень документації для реалізації потрібної моделі, а також доступна на різних мовах програмування, до яких входять мови lua, C++, Go.

Для створення веб-додатків на Python існують бібліотеки, а саме:

– django (для створення модульних додатків і серверів);
– flask (для створення мікросервісів).

Обидва фреймворки мають схожий інтерфейс для створення сайтів.

Django – це потужна бібліотека для створення веб-сайтів і сервісів. Ключові особливості Django:

– відкритий програмний код;
– відкрита ліцензія BSD 3 версії;
– 15 років зрілості;
– автономний веб-сервер для розробки та тестування;
– система шаблонів для генерації сайтів;
– фреймворк кешування;
– система інтернаціоналізації;
– підтримка модульних тестів;
– розширювана система аутентифікації;
– підтримки від незалежного фонду;
– своя модель для взаємодії з базою даних.

Django має складну файлову структуру, яку потрібно пам’ятати і змінювати в процесі розробки сайтів і API.

Flask – це мікрофреймворк, оскільки для його роботи не потрібні складні модулі чи певні інструменти. Особливості Flask:

– 10 років підтримки;
– ліцензія BSD;

– наявність двох систем шаблонування: Werkzeug і Jinja;
– окремий сервер для розробки;
– підтримка модульного тестування для веб API;
– наявність багатьох доповнень для створення RESTful серверу.

Головна особливість Django у розмірі модулів, які надає співтовариство. Flask легкий у використанні, має розроблену просту якісну структуру. Головна особливість Flask у модульності і гнучкості, а також у малому розміру бібліотеки, завдяки чому її легше підтримувати і розуміти каркас додатку.

Для задачі побудови веб-серверу в даному дослідженні обрана бібліотека Flask і мова шаблонування Jinja2, тому що цей інструмент має багато прикладів використання і зрозумілу документацію.

Для збереження і контролю інформації на соціальній платформі потрібна база даних. Існують різні типи баз даних: реляційні, графові, на основі документів, на основі ключів.

Сьогодні доступними є наступні популярні СУБД:

– MySQL;
– PostgreSQL;
– Microsoft SQL Server.

Microsoft SQL Server не підходить для розробки проекту через власницьку ліцензію для розробки ПЗ.

Реляційні бази даних MySQL і PostgreSQL добре задокументовані технології, а мова запитів SQL достатньо зріла. Обидві СУБД продаються та підтримуються низкою усталених корпорацій. Стандарти SQL чітко визначені та загальновизнані, а сама мова SQL має різні діалекти.

В даній роботі обрана реляційна база даних MySQL, бо вона доступна на всіх операційних системах, має ліценцію GPLv2 і екосистему інструментів для розробки, як-то редактор схеми БД в додатку phpmyadmin.

Для створення клієнтського додатку потрібно створити dekstop застосунок. На мові Python доступні декілька бібліотек для цього, а саме:

– QT;

– TKinter;
– WxWidgets;
– GTK;
– Kivy.

Qt була розробленою компанією Trolltech, яка створювала бібліотеку мовою C++. Особливості бібліотеки:

– швидка робота;
– редактор графічного інтерфейсу;
– багатоплатформність;
– одна з найкращих технічних документацій в світі ПЗ;
– доступність на різних мовах програмування.

Бібліотека tkinter побудована на мові python, але вона контролює іншу мову Tk, яка створює графічні інтерфейси. Цей модуль не має власного графічного інтерфейсу. Модуль має малу документацію, що погано впливає на процес побудови програм. Особливості модуля tkinter:

– багатоплатформність на різних версіях мови Python;
– відома в світі Linux і UNIX-подібних систем;
– вільне програмне забезпечення, випущене за ліцензією Python.

Ця бібліотека має звичайну для GUI-бібліотек структуру додатку та віджетів для розширення, але мала документація робить її складною під час розробки.

Для клієнтського додатку обрана бібліотека Qt, бо в неї найкраща документація, широкий вибір віджетів і прикладів їх використання, а також висока швидкість роботи бібліотеки. Оскільки бібліотека написана мовою C++, то для розробки на python використані прив’язки на бібліотеці PySide2 з відкритою ліцензією.

Для створення додатків для маркування і озвучення тексту використані веб-технології HTML, CSS, javascript. Вони потрібні для розробки платформи і для системи збору даних у вигляді сайту.

Отже, визначено технології для розробки ПЗ, а саме:

– для створення соціальної платформи обрана бібліотека Flask, мова для шаблонів Jinja 2, бібліотека каскадних стилів Bootstrap 4;

– для створення клієнтської програми обрана бібліотека Qt;

– для створення програми для маркування даних використані веб-технології і мови HTML, CSS і javascript.

## 4.2 Реалізація програм маркування і API веб-серверу

На рисунку 4.1 зображена файлова структура проекту, яка надає API для користувачів. В даній структурі є RESTful сервер і дві веб-сторінки з інтерфейсом для створення наборів даних. В створеному ПЗ икористаний архітектурний шаблон MVC для розділення шаблонів HTML, контролерів і моделей бази даних.

```
tree
.
├── auth.py
├── db.py
├── __init__.py
├── main.py
├── schuma.sql
├── templates
│   ├── base.html
│   ├── home.html
│   ├── index.html
│   ├── page-marker-1.html
│   └── page-marker-2.html
└── tests
    ├── test_audio1.py
    ├── test_auth.py
    ├── test_db.py
    ├── text_audio2.py
    ├── text_submit_marker1.py
    └── text_submit_marker2.py

2 directories, 16 files
```

Рисунок 4.1 – Файлова структура для веб проекту

На основі сутностей в базі даних створений окремий endpoint (вузол зв’язку), який виступає контролером. Кожний з вузлів надає CRUD для сутностей оголошення, користувач, коментар.

Програма маркування розроблена як одна веб-сторінка. Програму можна використовувати локально. Інтерфейс програм для маркування і озвучення текстів зображений на рисунках 4.2 і 4.3.

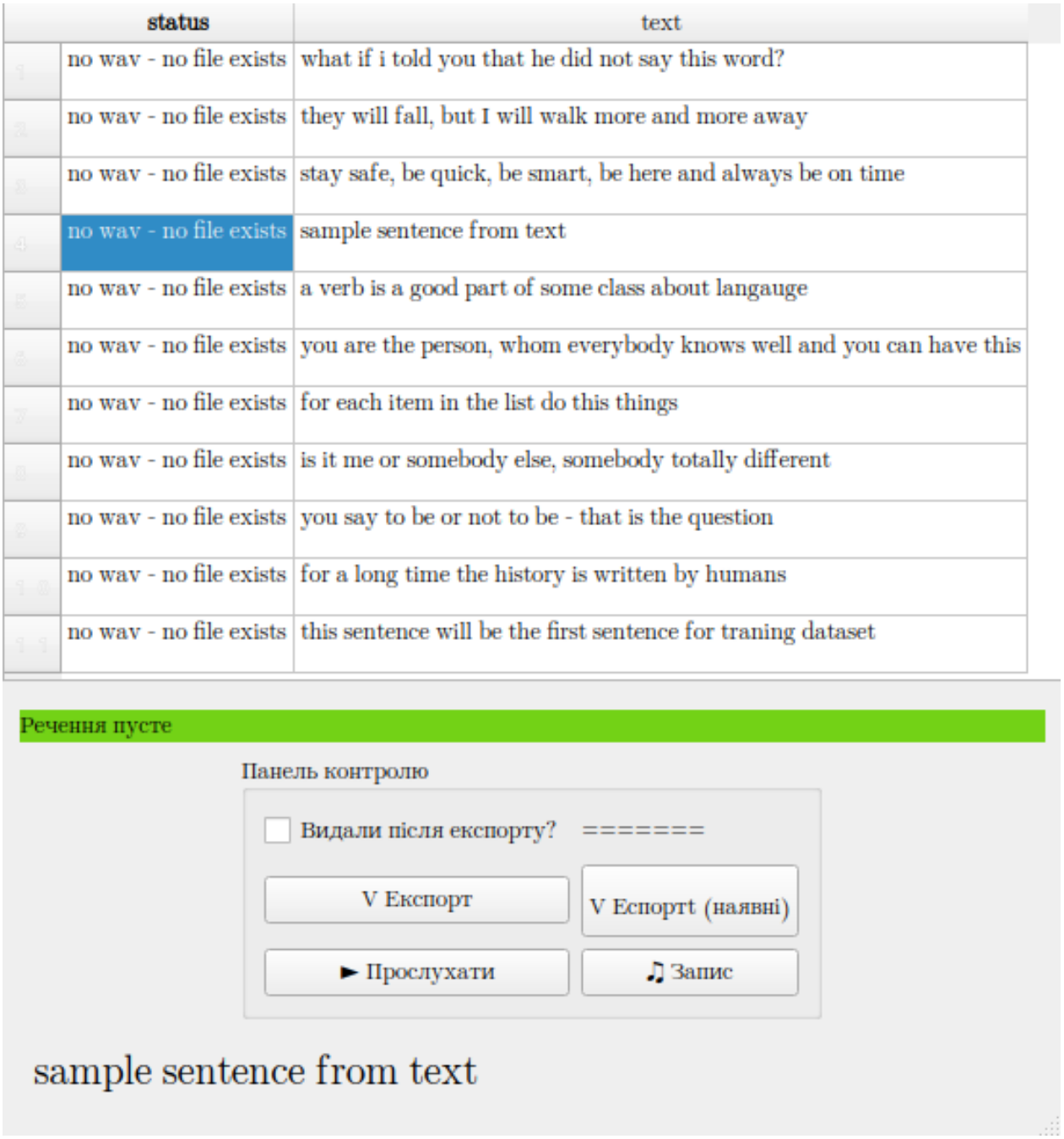



Рисунок 4.2 – Графічний інтерфейс для програми маркування речень через озвучення

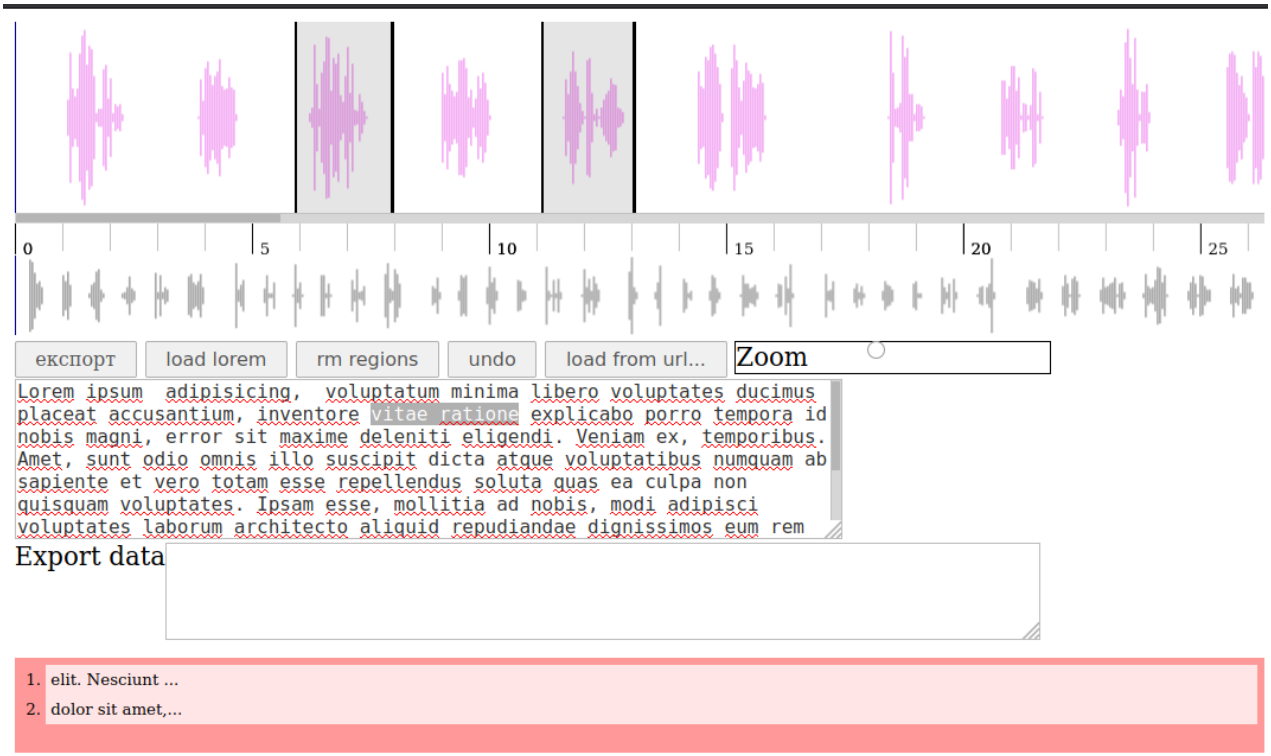



Рисунок 4.3 – Графічний інтерфейс для програми маркування речень через розмітку тексту

Реєстрацію на сервісі можна виконати двома способами:

– надання логіна і паролю;

– отримання автоматичного логіну і паролю;

Автоматично створені дані можна змінити за допомогою спеціального запиту на сервер. Користувачі отримують ключ сесії після проходження аутентифікації, який використовується протягом одного дня. Ключ повинен оновлюватись, що відслідковується в клієнтському додатку. Якщо користувач виходить з додатку, то на сервер відправляється спеціальний запит, який видаляє ключ сесії. Додаток автоматично оновлює ключ за одну годину до його знищення. Для взаємодії з API створена діаграма послідовності, яка зображена на рисунку 4.4, де зображений протокол спілкування через API.

Відповідь від серверу завжди поступає у форматі json, окрім випадків з інструментами для маркування сайтів. Для показу станів додатку на стороні клієнта використовуються HTTP-коди запитів. Гості, які не використовують клієнтський додаток, мають доступ до платформи через відправлення GET і POST запитів

серверу. Завдяки модулям всередині Flask, сервер реалізований за допомогою класів, в яких відповідні методи get, post, put і patch відображаються на http-запити, що створює зручний і швидкий спосіб розробки сервісу. Динамічна мова python дозволяє отримувати інформацію про всі об’єкти, які створені користувачем і інтерпретатором, в тому числі назви методів класу. Flask використовує цю характеристику мови для автоматичної побудови шляхів сайту.

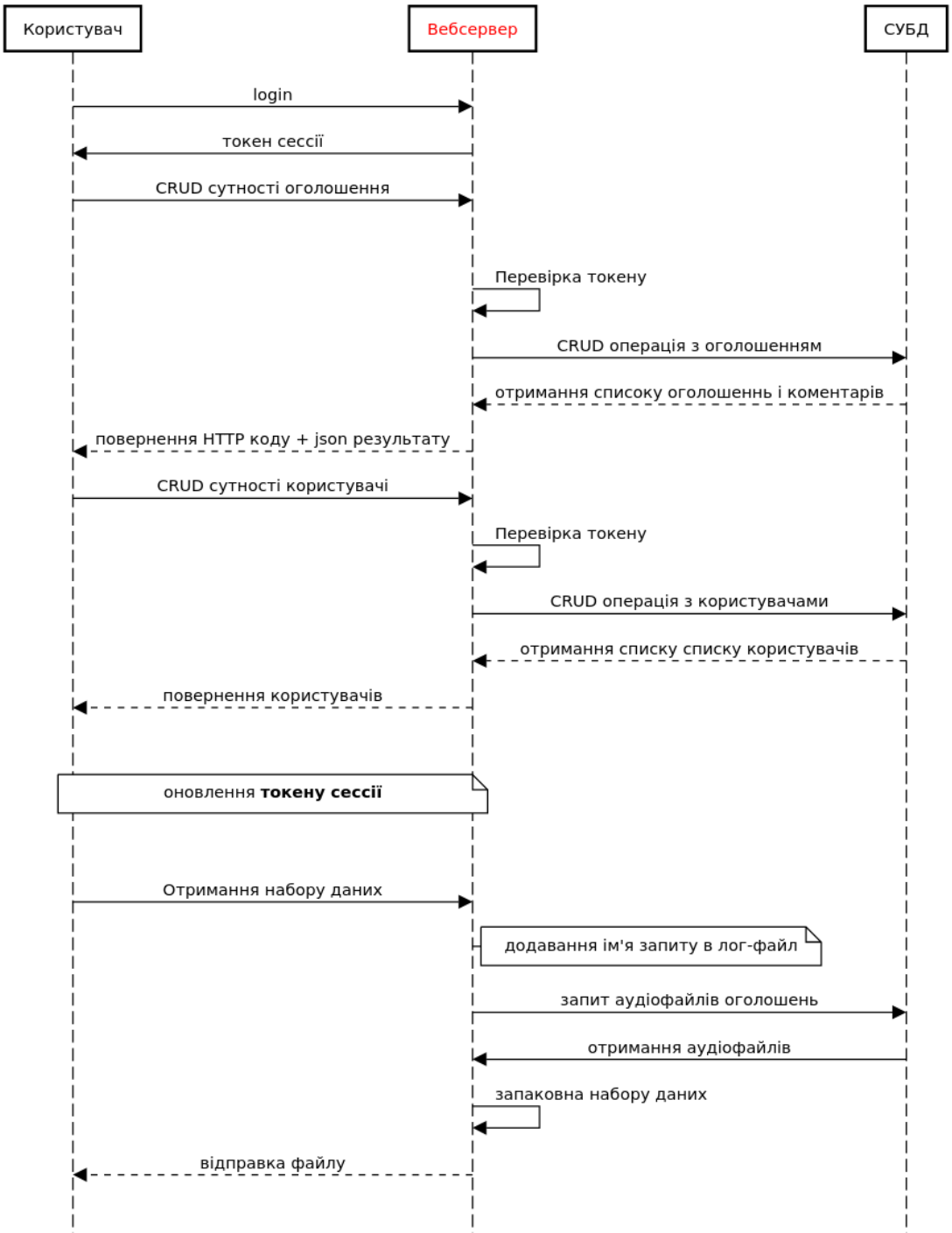



Рисунок 4.4 – Діаграма послідовності

## 4.2 Тренування нейронної мережі

Тренування базової англійського моделі зайняло два дні: один день (15 годин) на тренування першого блоку TextEnc і один день (20 годин) на SSRN блок. Далі TextEnc і SSRN блоки використані як transfer learning моделі.

На основі дослідів виявлено, що подальше тренування всієї моделі під окрему людину може займати від 40 хвилин до двох годин. TextEnc блок потребує від 2000 до 15000 кроків тренування, що займає від трьох до 45 хвилин. SSRN блок потребує від 2000 до 15000 кроків тренування, що займає від 20 до 30 хвилин.

На рисунках 4.5-4.9 зображені графіки механізму уваги, де якість знаходження букв виділена різними кольорами. Світлий колір означає вдале знаходження букви, а темний колір означає, що мережа не змогла передбачити потрібний символ. Завдяки такому графіку можна відслідковувати якість моделі та зберігати під час навчання ті моделі, які мають найкращий механізм уваги. Як видно з графіків, вже на перших кроках мережа може передбачити базові звуки, але, як показали досліди, для отримання якісного голосу потрібно від 50000 до 150000 кроків тренування.

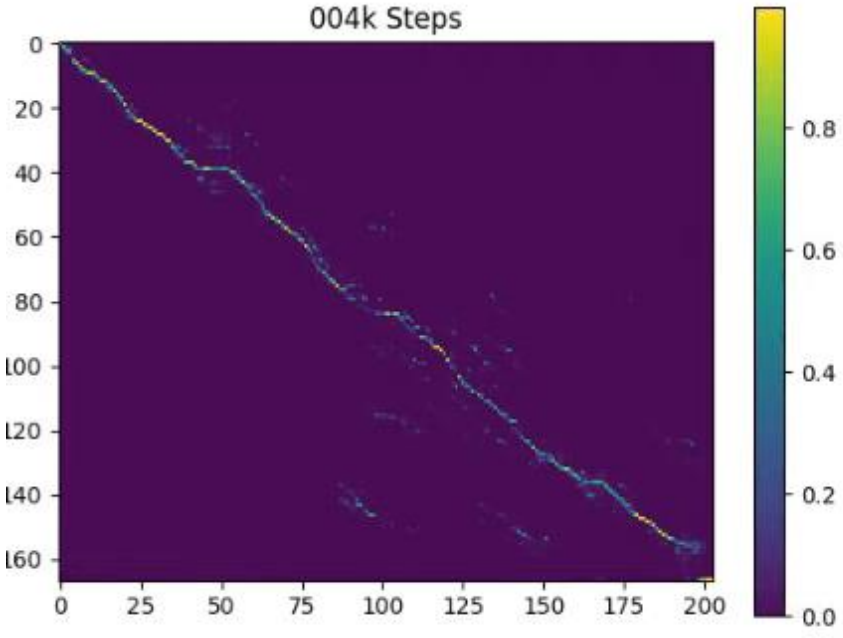



Рисунок 4.5 – Графік уваги на четвертому кроці тренування

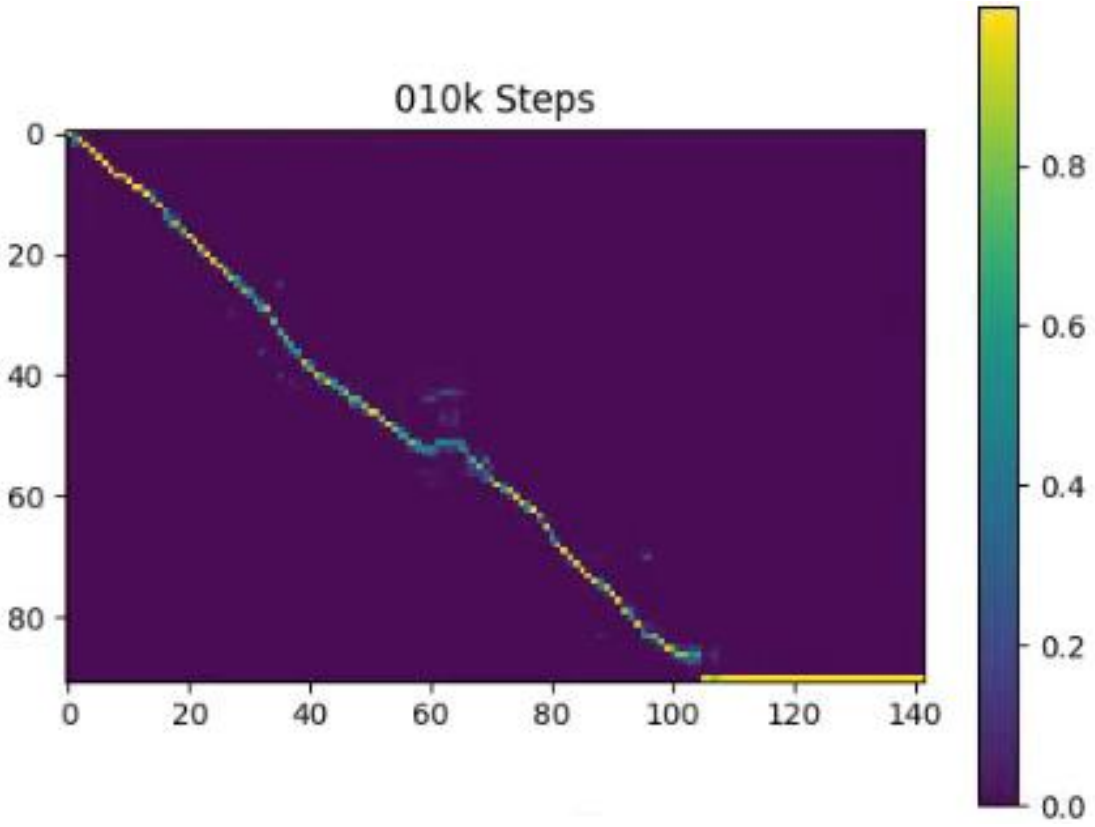



Рисунок 4.6 – Графік уваги на 10-му кроці тренування

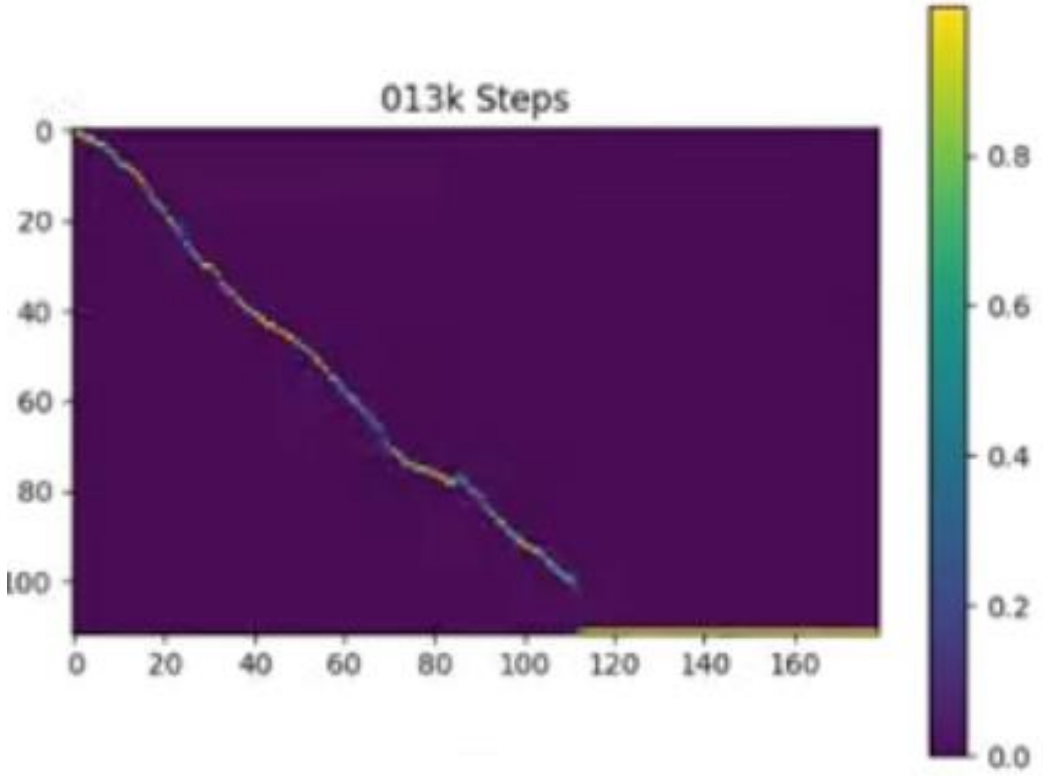



Рисунок 4.7 – Графік уваги на 13-му кроці тренування

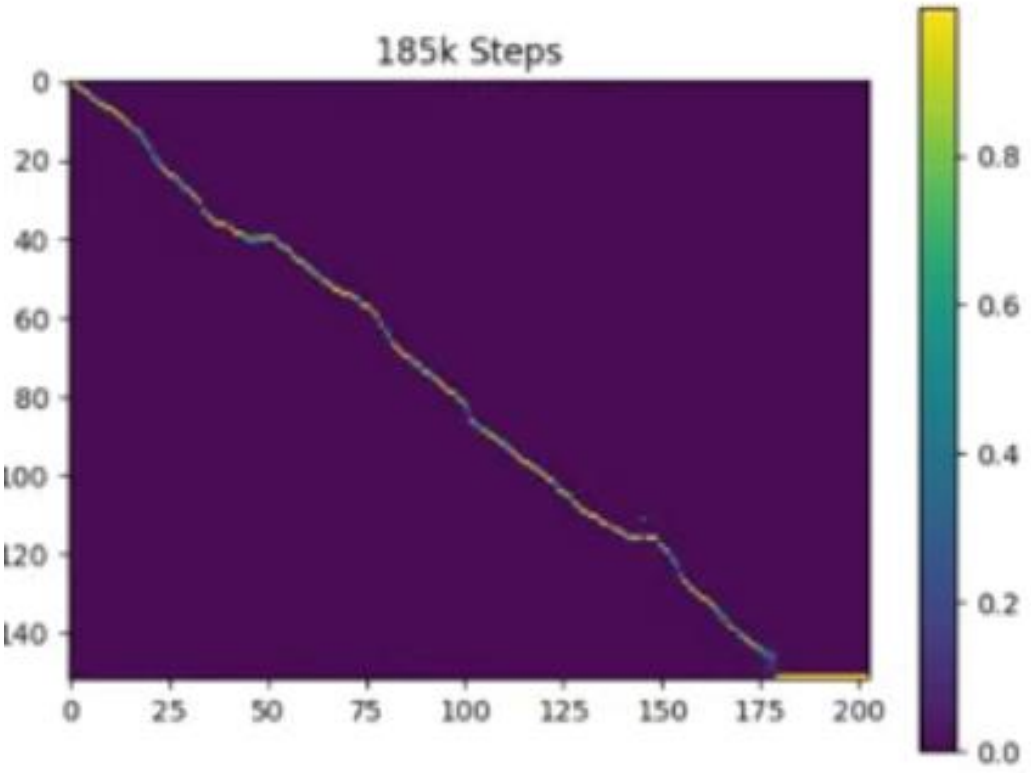



Рисунок 4.8 – Графік уваги на 185000 кроці тренування

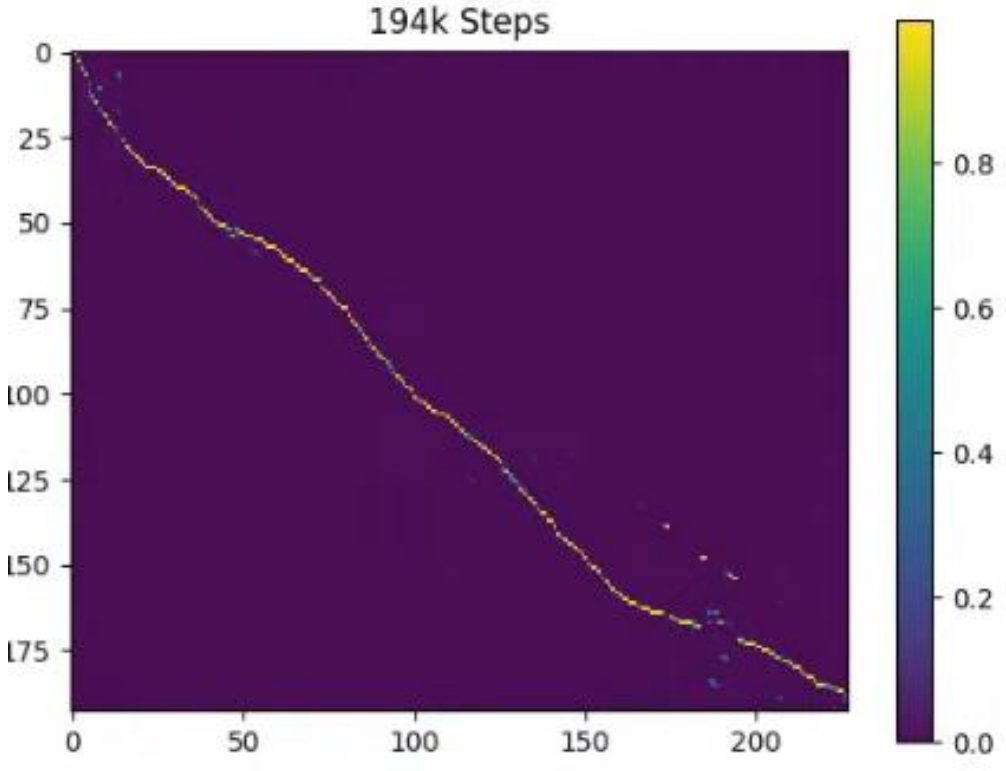



Рисунок 4.9 – Графік уваги на 194000 кроці тренування

## 4.3 Результати і оцінка якості синтезу нейронної мережі

Для оцінки якості синтезу голосу застосовується метод опитування людей. Така оцінка називається mean opinion score (MOS). Було опитано 20 осіб. Оцінка MOS сягає 2,9 для української мови і 3.1 для англійської мови. Голос схожий на людський і вимовляє слова в різних контекстах по-різному, ставить правильні наголоси в словах і фразах, а також робить паузи між слова. Згідно отриманих результатів модель з нейронними мережами дозволила створювати і озвучувати тексти потрібним голосом. Процес отримання власної моделі голосу приблизно займає до п’яти годин, якщо одночасно враховувати час завантаження даних і тренування. В результаті, розроблена система виконує всі задачі, які визначені в першому розділу роботи.

## 4.4 Інструкція користувача

Для використання програми для маркування голосу потрібно:

– вибрати програму для озвучення або маркування;
– встановити інтерпретатор мови програмування Python 3;
– встановити необхідні бібліотеки для роботи з нейронними мережами.

В окремих теках з ПЗ для створення наборів даних є файли setup.sh і setup.bat, в яких реалізовані вище перераховані пункти. Для користувачів на операційній системи Windows необхідно запустити setup.bat за допомогою подвійного кліку лівої миші на файл, а для користувачів Unix-подібних систем потрібно відкрити інтерпретатор командного рядка і в теці з програмою, виконати команду chmod +x setup.sh для надання прав запуски скрипту і виконати команду ./setup.sh для запуску процедури встановлення.

Після того, як скрипт setup завершиться, запускається основна програма за допомогою команди python3 main.py. Дана програма відкриє браузер або GUI

програму для озвучення чи маркування відповідно. В кожній з цих програм є підказки, які з’являються під час наведення курсору миші на віджет.

Апаратне забезпечення для створення наборів даних потребує мінімальних технічних характеристик, а саме:

– процесор з тактовою частотою від 500 МГц;
– оперативна пам’ять від 2048 Мб;
– наявність мережевої карти;
– відеокарта з об’ємом пам’яті від 500 Мб.

Спеціальна кваліфікація оперативного персоналу не потрібна для використання ПЗ, оскільки в ньому достатньо простий інтерфейс.

Повідомлення користувачу видаються під час відправлення і збереження даних в додатку.

# ВИСНОВКИ

Для проведення експериментів в області машинного навчання потрібні теоретичні знання, науковий метод і якісні дані. Нейронні мережі – це потужний інструмент машинного навчання, який здатний тренуватись на вхідних даних, генерувати, розпізнавати та класифікувати дані. Для створення таких математичних моделей потрібен експерт, що знає алгоритми добування даних, їх організації і властивості процесу навчання мереж.

У дослідженні за темою роботи проаналізовано методи озвучення тексту і способи збору аудіоданих за допомогою колективної роботи. На основі теоретичний даних запропонована модель, що складається з чотирьох нейронних мереж по перетворенню і синтезу голосу для озвучення тексту українського мовою. Оцінки якості навченої моделі за шкалою MOS сягає від 2.9 до 3.1, час тренування базової моделі – до трьох днів, час тренування коригування власного голосу – до трьох годин. Замість RNN шарів в нейронній мережі використані виключно CNN шари і механізм уваги для задачі синтезу голосу.

Основні задачі для вирішення зазначеної мети були виконані. До них входять:

– створення архітектури та проведення тренування глибокої нейронної мережі для синтезу голосу української мовою і побудова архітектури соціальної платформи для проведення збору аудіоданих та створення набору даних українською мовою

– створення і розмітка звукових наборів даних для навчання глибинної нейронної мережі

– тренування, оцінка та оптимізація роботи нейронної мережі.

В результаті розробки була побудована архітектури нейронної мережі для озвучення тексту на основі синтезу голосу людини. В межері були реалізовані усі вимоги, які були вказані в завданні, натренована модель для озвучення українського тексту і розроблена соціальна платформа для збору помічених набоір аудіоданих.

Для реалізації завдання проекту була використана мова програмування Python 3, бібліотека torch і модуль Flask. Спроектоване ПЗ для синтезу голосу

дозволяє використовувати штучний голос для озвучення текстів, модифікувати стандартний голос під свій голос та використовувати його як один з доступних голосів.

Після проведення дослідження і отримання практичних результатів планується застосовувати набуті результати в розробці онлайн-сервісів для комунікації на різних мовах, інтегрувати методами розпізнавання мови для створення штучний наборів даних. Відкритим залишається питання рішень, які здатні синтезувати різні голоси однією глибинної мережею або розмовляти на різних мовах за допомогою однієї моделі.

За результатами досліджень опублікована одна стаття [25].

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. McCulloch W. A logical calculus of the ideas immanent in nervous activity // The bulletin of mathematical biophysics. – 1943. – Vol. 5, No. 1. – P. 155-133.

2. Hunt A. Unit selection in a concatenative speech synthesis system using a large speech database // IEEE International Conference on Acoustics. – 1996. – P. 373-376.

3. Zen H. Statistical parametric speech synthesis // Speech communication. – 2009. – P. 1039-1064.

4. Wu Z. Investigating gated recurrent networks for speech synthesis // ICASSP. – 2016. – P. 5140-5144.

5. Zen H. Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis // ICASSP. – 2013. – P. 4470-4474.

6. Lei M. Formant-controlled HMM-based speech synthesis // Twelfth Annual Conference of the International Speech Communication Association. – 2011. – P. 24.

7. Oord A. Wavenet: A generative model for raw audio // arXiv preprint arXiv:1609.03499. – 2016.

8. Paine T. Fast wavenet generation algorithm // arXiv preprint arXiv:1611.09482.– 2016.

9. Arik S. Deep voice: Real-time neural text-to-speech // arXiv preprint arXiv:1702.07825. – 2017.

10. Wang Y. Tacotron: A fully end-to-end text-to-speech synthesis model // arXiv preprint arXiv:1703.10135. – 2017.

11. Gibiansky A. Deep voice 2: Multi-speaker neural text-to-speech // Advances in neural information processing systems. – 2017. – P. 2962-2970.

12. Ren Y. Almost unsupervised text to speech and automatic speech recognition // arXiv preprint arXiv:1905.06791. – 2019.

13. Ping W. Deep voice 3: Scaling text-to-speech with convolutional sequence learning // ICLR 2018. – 2018.

14. Shen J. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions // ICASSP. – 2018. – P. 4779-4783.

15. Wang Y. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis // arXiv preprint arXiv:1803.09017. – 2018.

16. Miller G. A. WordNet: An Electronic Lexical Database : книга / Miller G. A. – MIT press, 1998. – 55 с.

17. Xu Y. A regression approach to speech enhancement based on deep neural networks // ACM Trans. Audio, Speech and Lang. Proc. – 2014. – Vol 23, no 1. – P. 7-19.

18. Zen H. LibriTTS: A corpus derived from LibriSpeech for text-to-speech // arXiv preprint arXiv:1904.02882. – 2019.

19. Keithito.com: [Веб-сайт]. URL: https://keithito.com/LJ-Speech-Dataset/ (дата звернення: 22.11.2020).

20. Kyubyong P. CSS10: A Collection of Single Speaker Speech Datasets for 10 Languages // Proc. Interspeech 2019. – 2019. – P. 1566-1570.

21. Electronics europe News // Electronics-eetimes 2020. URL: https://cdn.cdnartwhere.eu/electronics-eetimes.com/sites/default/files/images/01-edit-photo-uploads/2011/2011-08-august/c0864-figure1.gif (дата звернення: 29.11.2020).

22. O'shaughnessy D. Speech Communications: Human And Machine (IEEE) : книга / D. O'shaughnessy. – Universities press, 1987. – 45 с.

23. Srivastava R., Rupesh K., Schmidhuber J. Training Very Deep Networks / Зб. наук. пр // Advances in Neural Information Processing Systems. California, 2015. С. 2377–2385.

24. Tachibana H. Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention // IEEE International Conference on Acoustics. – 2018. – P. 4784-478.

25. Савінський. В. Social platform for making labeled audio datasets for speech synthesis of human voice // Зб. наук. пр. наукової конференції «АПКН-2020». – Хмельницький ХНУ. – 2020. – С. 261-264.

ДОДАТОК А

(обов’язковий)

# ТЕХНІЧНЕ ЗАВДАННЯ

Мета розробки

Метою створення розробки є реалізація програми для озвучення тексту українською мовою голосом людини для персонального офлайн користування.

Предметом розробки є нейронні мережні для синтезу голосу.

Системи призначена для надання можливості озвучення текстів і повідомлень українською мовою за допомогою синтезованого голосу конкретної людини.

Опис проекту

Компанії Замовнику потрібний програмний модуль і додаток для соціальної мережі. Люди в соціальній мережі можуть обмінюватись текстовими повідомленнями, але вони не мають можливості їх прослухати. В зв'язку з цим, необхідно створити програму, яка озвучує повідомлення адресанта і адресата відповідними голосами.

Оцінка термінів

Термін виконання проекту – 3 місяці. Звіти поточного стану проекту потрібно давати кожні 3 дні.

Вимоги до мов програмування і модулів

Усі модулі повинні бути реалізовані мовою програмування Python 3.8. Для реалізації консольного інтерфейсу необхідний модуль curses, для графічної оболонки - бібліотка pyqt5.

Основні функції мають мати документацію, тобто перша стрічка функцій і класів будуть мати першою інструкцією багатомовного коментаря зі вказанням типів вхідних і вихідних даних. Для табуляції у вихідному коді використовується знак пробілу.

Графічний дизайн програми має бути у форматі .ui файлів для бібліотеки графічної оболонки, який генерується автоматично в дизайнері інтерфейсу.

Основні функції додатку повинні мати гарячі клавіши від F1 до F12 і маркерні позиції, які доступні при натисканні клавіши Alt.

Функціональні вимоги

Потрібно розробити модуль синтезу голосу для внутрішньої інтеграції у вигляді бібліотеки з API. Розробка додатку потрібна для тестування і налагодження синтезу голосу.

Модуль для інтеграції повинен мати функцію say(text, **parameters) зі списоком параметрів:

– do split sentences, функціонал розділення тексту на речення

– out file, шлях до вихідного файлу

– out dir, шлях до вихідної папки для зберігання файлу чи файлів.

Перед відправкою тексту, користувач має мати змогу прослухати текст потрібних голосом.

Опція «озвучка окремих речень». Ця функція буде озвучувати вхідний текст після його розділення на речення; функція «зберегти текст в аудіо» має зберігати текст повідомлення будь-якого зі сторін у вихідний файл в форматі wav. Також має створюватись файл, в якому є список озвучених речень і їх абсолютних шлях до аудіофайлу до кожного з цих речень. Якщо повідомлення вже були озвучені, то потрібно їх зберігати в кеш, який можна очистити по бажанню користувача.

Потрібно мати декілька способів введення тексту, а саме:

– набір на клавіатурі;

– вставка з буферу обміну;

– завантаження файлу з кодуванням unicode.

Нефункціональних вимоги

Стиль додатку має мати чорні кольори для заднього плану і білий колір для тексту, шрифт для тексту – Inconsolata. Вихідних код програми, залежності від інших модулів і модель голосу мають бути доступні користувачу для зміни, запуску і копіювання.

Весь графічний дизайн додатку має мати тільки одне головне вікно. Дизайн графічного інтерфейсу розробляється Виконавцем на його власний розсуд. Під час озвучки текст екран користувача має подавати сигнали стану озвучки, а саме «в процесі» і «озвучується».

Для роботи з програмою потрібні базові знання з комп'ютером, а для роботи клієнтського додатку потрібно:

– монітор

– доступ до мережі інтернет

– клавіатура і миш

– процесор тактовою частотою не менше 2000 МГц;

– об’єм оперативної пам’яті не менше ніж 4096 Мб

ДОДАТОК Б

(обов’язковий)

# КОМПЛЕКС РОЗРОБЛЕНИХ ДІАГРАМ

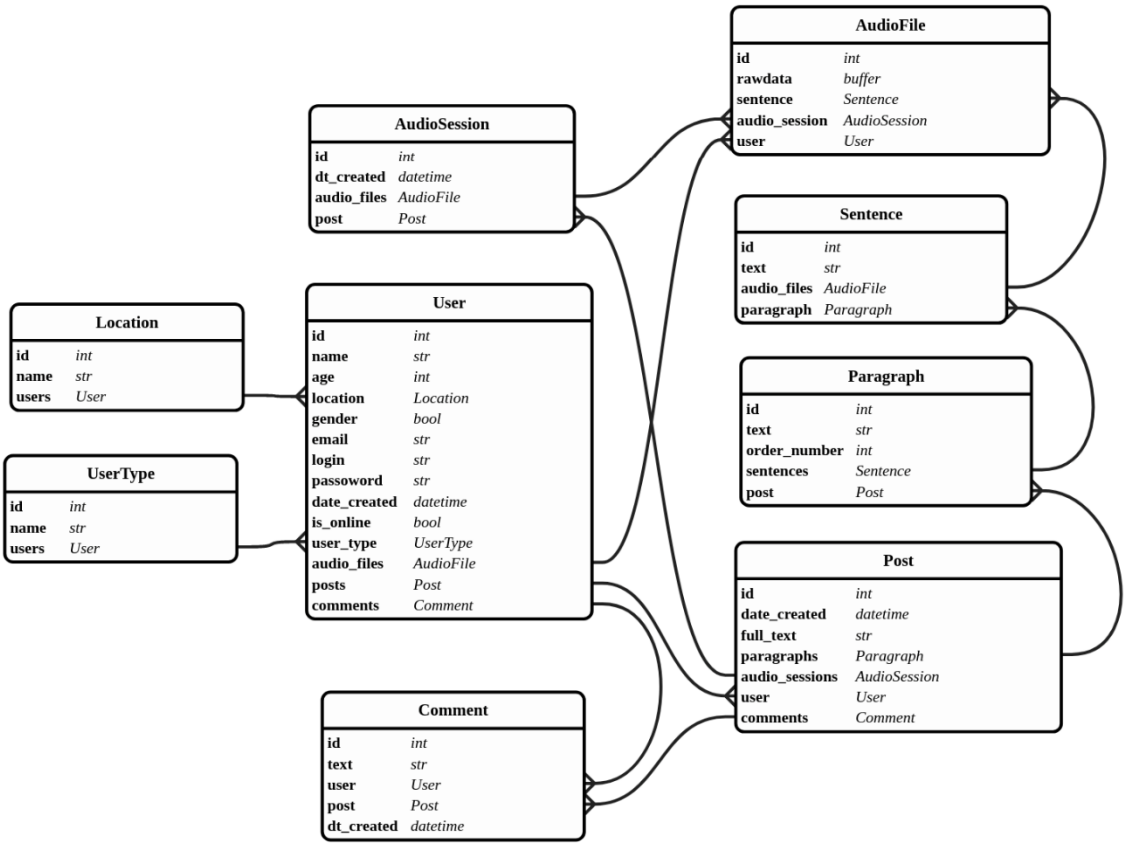



Рисунок Б.1 – База даних соціальної платформи

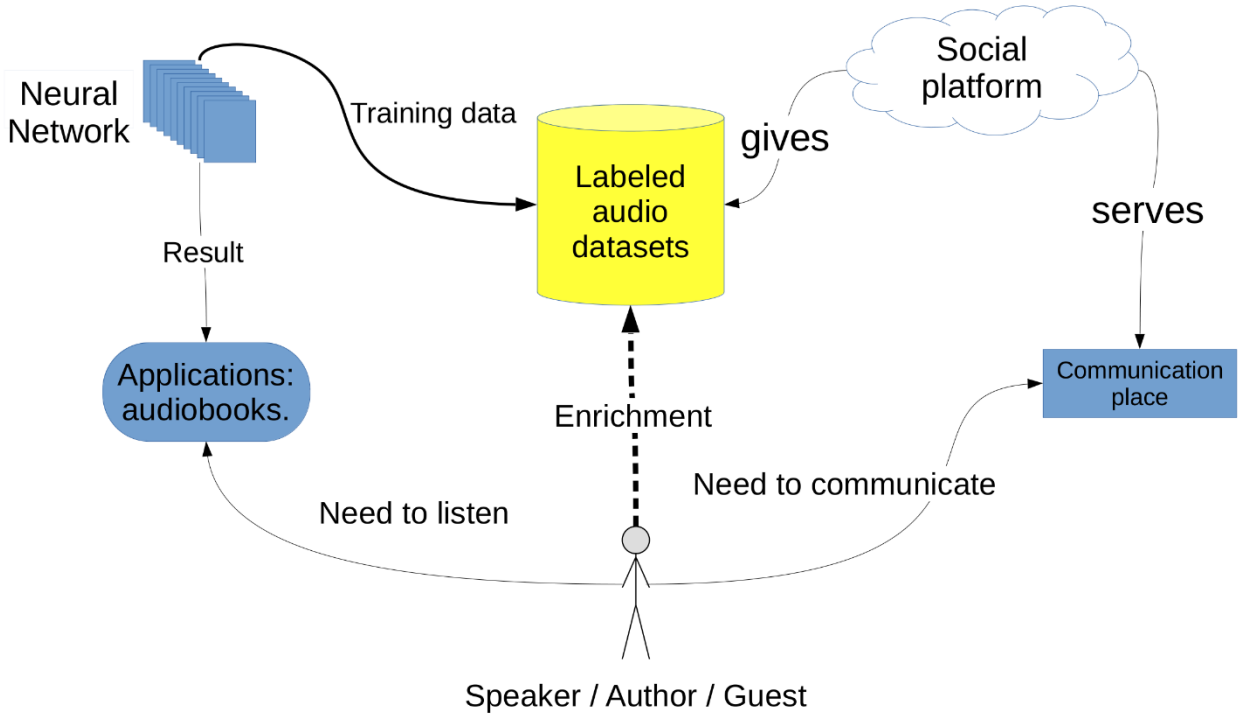



Рисунок Б.2 – Схема взаємодії користувача з соціальною платформою

ДОДАТОК В

(обов’язковий)

# ТЕЗИ НАУКОВОЇ РОБОТИ

Social platform for communication and labeled audio datasets

Савінський В.В.
Науковий керівник – д. ф.-м.н., проф, Бедратюк Леонід Петрович
Хмельницький національний університет

В даній роботі розроблена соціальна платформа для комунікації і створення аудіо датасетів для контрольованого навчання. Платформа може бути реалізована як незалежний веб-сайт або вбудована всередину існуючого проекту як окрема частина. Дана платформа корисна як для відвідувачів і учасників, так і для розробників.

The architecture of social network for communication and making audio dataset for supervised learning is developed in this work. The platform can be implemented as independent website or embedded inside of the existing project as a separate part. The platform is useful for both visitors with participants and researchers.

Finding a dataset for training neural network is one of the important steps in process of building, training and tuning a neural network.

Currently public datasets is used for training deep neural network, evaluating result and understand the score for other models. Also, depending on type of the datasets, dataset itself can be useful not only for deep learning practitioners, but also for domain experts and other public as well. For example, explanatory dictionary is a dictionary that gives additional information, like pronunciation, grammar, meaning, etymology[1], is useful for everybody, but, on other hand, it can be a basis for a dataset, that can contain a categories of words in hierarchy [2].

Audio dataset are less common compared to visual datasets due to complexity with software for labeling process of each picture. Existing

LibriVox is a free public domain audiobooks site. According to the Librivox statistics, it has more then 14000 cataloged works and more then 1500 non-english works. All those works are in form of a collection on wav files without associated text to each file. LibriVox only records material that is in the public domain in the United States, and all LibriVox books are released with a public domain dedication. Because of copyright restrictions, LibriVox produces recordings of only a limited number of contemporary books.

LibriTTS [3] is a speech corpus, designed for text-to-speech use. It is derived from the original audio and text materials of the LibriSpeech corpus, which has been used for training and evaluating automatic speech recognition systems. The new corpus inherits desired properties of the LibriSpeech corpus while addressing a number of issues which make LibriSpeech less than ideal for text-to-speech work. The released corpus consists

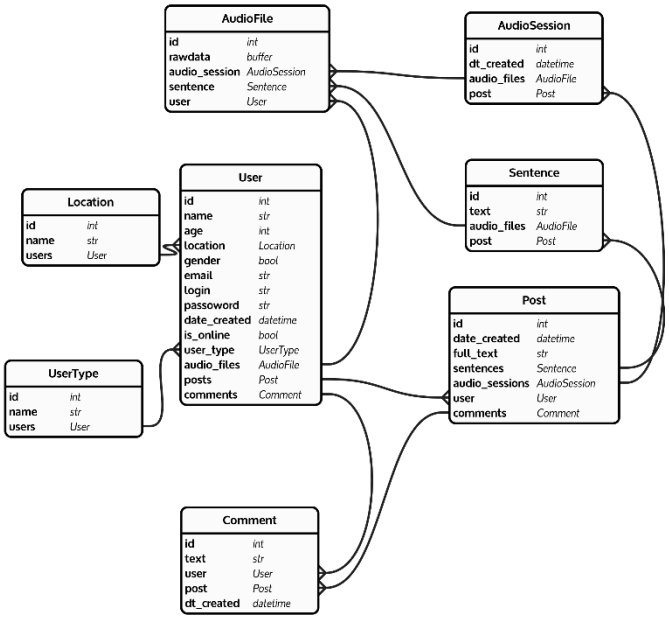
of 585 hours of speech data at 24kHz sampling rate from 2,456 speakers and the corresponding texts.

LJ Speech Dataset[4] is a public domain speech dataset consisting of 13,100 short audio clips of a single speaker reading passages from 7 non-fiction books. A transcription is provided for each clip. Clips vary in length from 1 to 10 seconds and have a total length of approximately 24 hours. The texts were published between 1884 and 1964, and are in the public domain. The audio was recorded in 2016-2017 by the LibriVox project and is also in the public domain.

CSS10[5] paper is a collection of single speaker speech datasets for ten languages. It is composed of short audio clips from LibriVox audiobooks and their aligned texts.

All described audio datasets were build upon existing audiobooks with manually alignment text for audio files. Audio datasets generally are in format of (wavfile, text) pairs or (wavfile, text, transcription) triplets.

In this work an architecture for social platform designed for text and audio blogposts is given. This social platform will be used like a regular blog, but users can attach a audio-version on their blogpost. Also other people can voice-over other users blogpost too. A blogpost is a text work, that can be a anything from list of sentences to a small novel or a book.




Picture 1 – Database schema for social platform

If a speaker wants to make an audio version of his blogpost, then a audio-editor is given to him, where he can manage his audio version, align text, automatically split text into sentences and give detail information about himself:

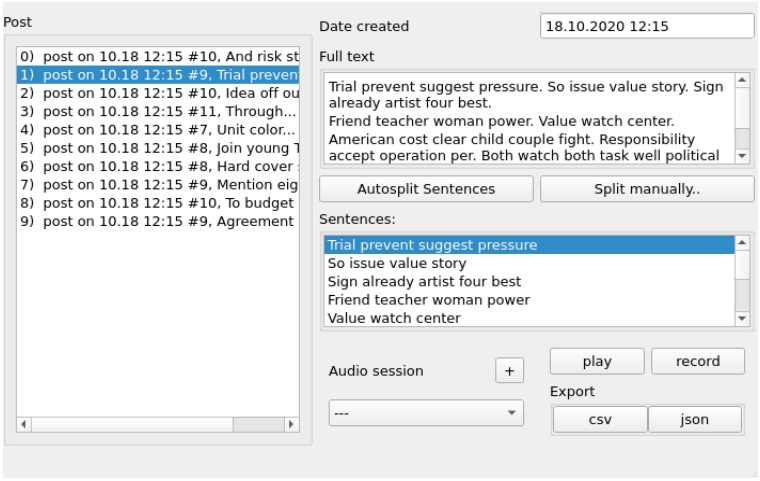
– gender
– age
– geographical location

Information about speaker allows to have a detail picture of his voice and can be used to synthesize a specific type voice.

A database schema for social platform is shown on picture 1.

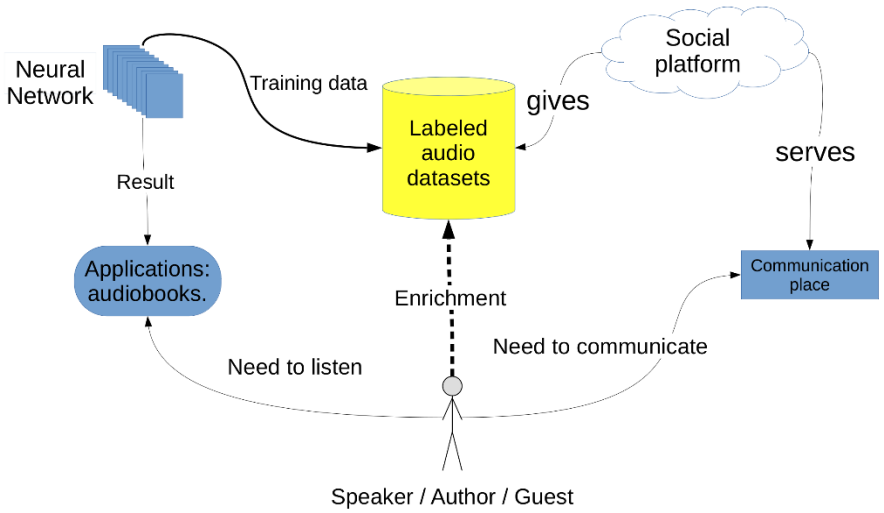
This platform is useful for both authors, speakers, deep learning researchers and guests users, which is shown on picture 2.

For making and managing audio versions of posts, a simple user interface is given on picture 2.




Picture 2 – User Interface for managing audio version of posts

The overall picture of communication between users and social platform is shown on picture 3.




Picture 3 – Interaction between users and social platform

It is possible to make a specific local audio datasets, which can contains information, like common accent or dialect of some language. Application, that can use such datasets, can build models for speech synthesis from specific region of country, tune age or gender of a speaker.

A datasets gathered from this platform can be used in projects, which are related to audio signal recognition, speech recognition, end-to-end speech recognition, text-to-speech synthesis, speaker verification, speaker identification and related supervised learning problems.

In conclusion, the architecture of social platform for communication and crowdsourcing labeled audio dataset is given. This platform can be implemented as an independent website or included in existing project as a module, if the project itself is a blogging platform or related media distributor.

Finally, a simple and effective interface for collecting, managing and building labels audio datasets is given.

Future work will involve improving the collected audio files, making models for notifying user about quality of recording, automatically clean background noise with custom parameters, adding additional valuable metadata to audio files.

Перелік посилань

1. Explanatory dictionary[Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Explanatory_dictionary

2. George A. Miller. 1995. WordNet: a lexical database for English. Commun. ACM 38, 11 (Nov. 1995), 39–41. – Режим доступу: https://doi.org/10.1145/219717.219748

3. Park, K., Mulc, T. (2019) CSS10: A Collection of Single Speaker Speech Datasets for 10 Languages. Proc. Interspeech 2019, 1566-1570, DOI: 10.21437/Interspeech.2019-1500.

4. Zen, H., Dang, V., Clark, R., Zhang, Y., Weiss, R.J., Jia, Y., Chen, Z., Wu, Y. (2019) LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech. Proc. Interspeech 2019, 1526-1530, DOI: 10.21437/Interspeech.2019-2441.

5. LJSpeech Dataset[Електронний ресурс]. – Режим доступу: https://keithito.com/LJ-Speech-Dataset/

ДОДАТОК Г

(обов’язковий)

**ПРЕЗЕНТАЦІЙНІ МАТЕРІАЛИ**

Тема:
Технологія розробки програмної системи для озвучення тексту голосом людини на основі машинного навчання

Науковий керівник:
д. ф-м.н., проф.
Бедратюк Л.П.
Виконавець:
студент 2 курсу групи ІПЗм-19-1
Савінський В.В.

01.

## Об'єкт і предмет дослідження

**Об'єкт**
Технології для озвучення тексту

**Предмет**
Нейронні мережі для озвучення тексту.

02.

# Мета дослідження і завдання дослідження

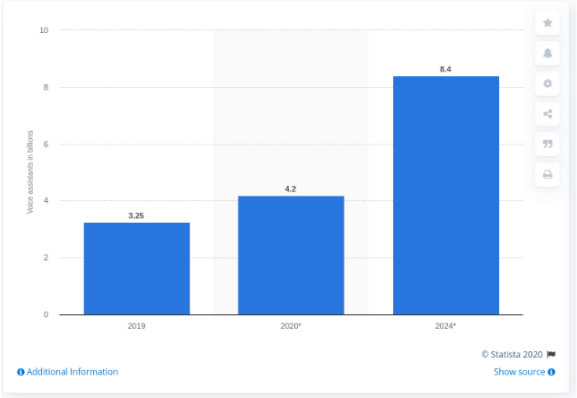
**Мета:**
1) Розробка архітектури та тренування глибокої нейронної мережі для синтезу голосу української мовою
2) Створення і розмітка звукових наборів даних для навчання нейронної мережі

**Завдання:**
На основі аналізу ML методів, нейронні мережі обрані як метод синтезу голосу. До задач входять пункти:
- побудова архітектури нейронної мережі і необхідних модулів обробки даних;
- створення архітектури соціальної платформи, організація збору аудіоданих та створення набору даних;
- тренування, оцінка та оптимізація роботи нейронної мережі.



# Актуальність теми

- Розвиток галузі і ринку продуктів персональних помічників[1]
- Розвиток інтерактивних освітніх програми
- Відсутність українського набору даних для аудіо;
- Аудіокниги у виконанні будь-якого голосу
- Колективна робота та озвучення тексту
- Великий інтерес до проблеми озвучення тексту, зросла кількість наукових статей з цієї тематики в останні роки
- Створено і натренорвано нейронні мережі для синтезу звуку
- Відсутність програмного забезпечення для озвучення текстів українською мовою

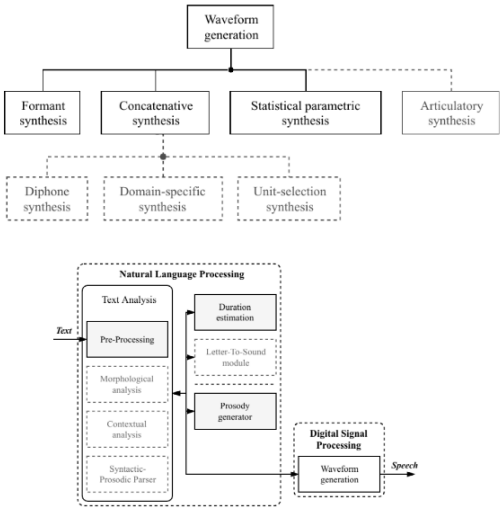


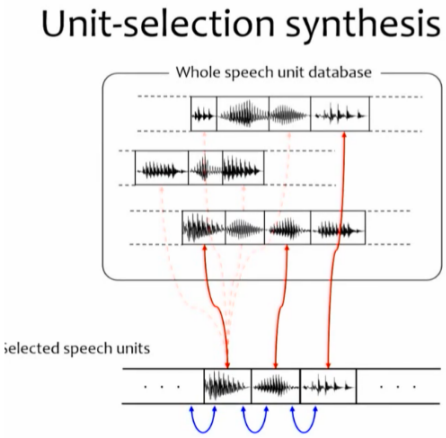

Кількість цифрових голосових помічників,
що використовуються у всьому світі з 2019 по 2024 рік

[1] - https://www.statista.com/statistics/973815/worldwide-digital-voice-assistant-in-use/

# Аналіз стану проблеми і інших рішень

Типи архітектурних рішень для задачі синтезу голосу:

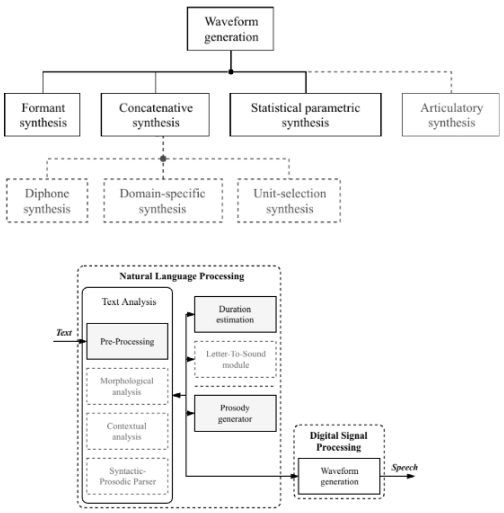


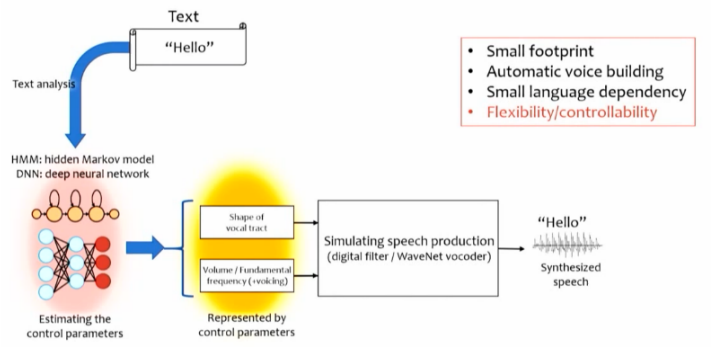






# Аналіз стану проблеми і інших рішень

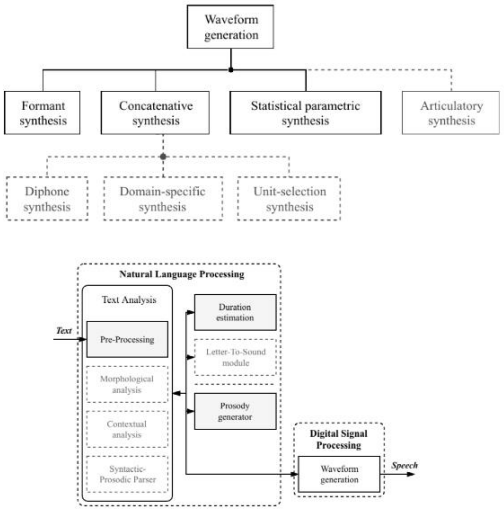
Типи архітектурних рішень для задачі синтезу голосу:

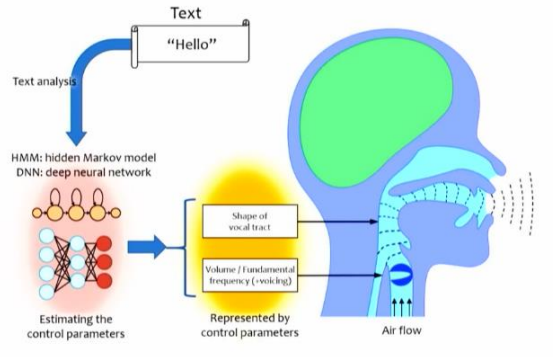


# Аналіз стану проблеми і інших рішень

Типи архітектурних рішень для задачі синтезу голосу:




06.

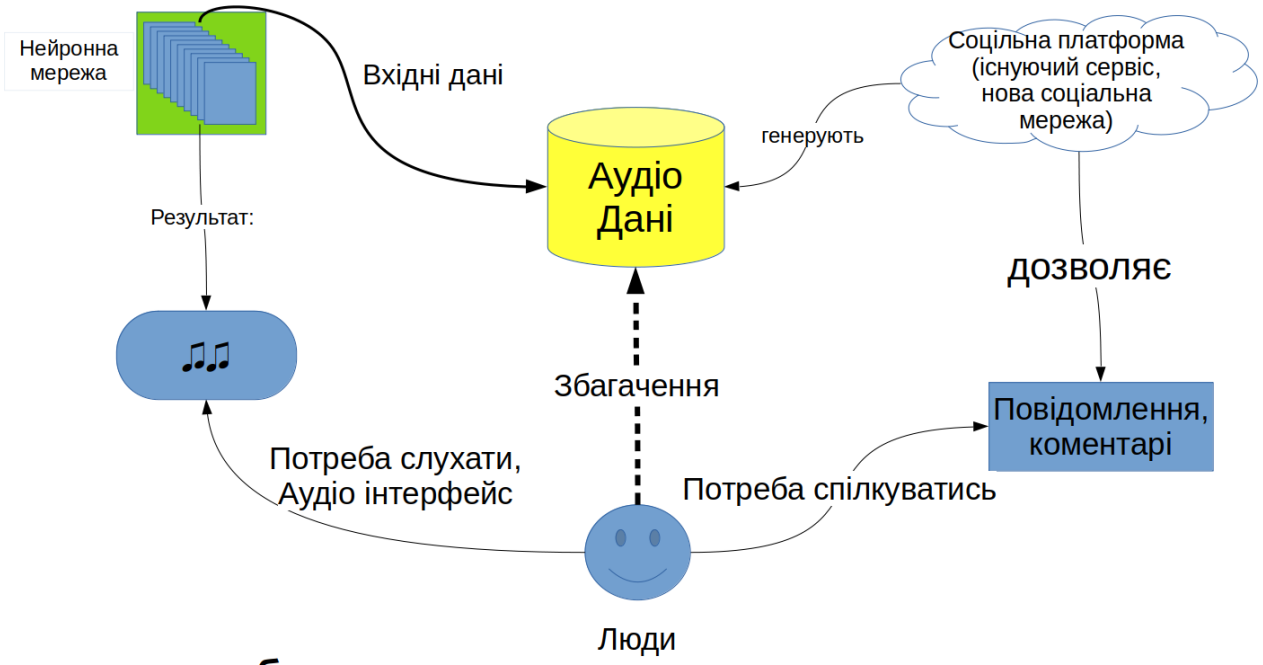
# Аналіз стану проблеми і інших рішень

Наявні методів синтезу голосу з використанням нейронних мереж:
- **WaveNet** (2016)
- Fast WaveNet (2016)
- Deep Voice (2017)
- **Tacotron** (2017)
- Deep Voice 2 (2017)
- TTS + ASR(Speech Chain) (2017)
- **Deep Voice 3** (2017)
- Tacotron 2 (2017)
- GST-Tacotron (2018)

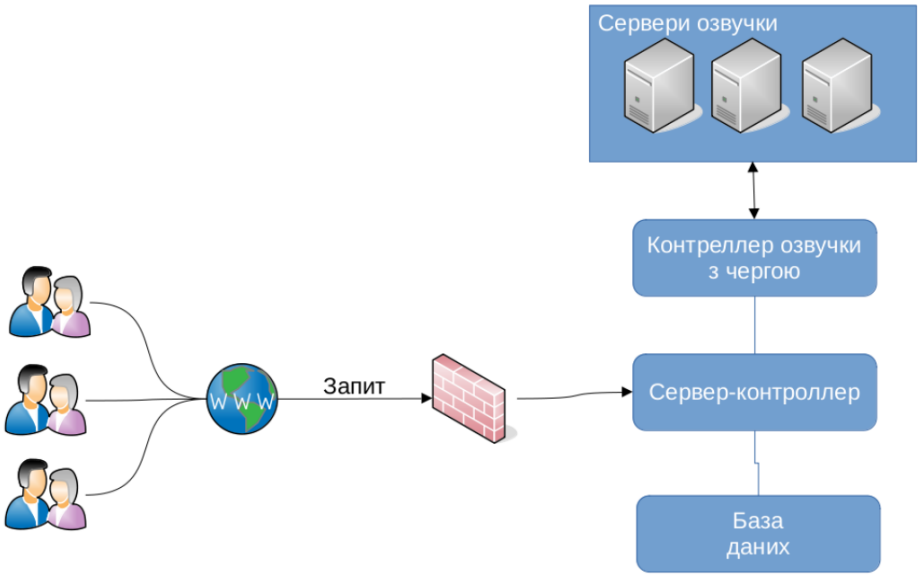
**DNN**

- Work for larger database?
- **Flat structure**
  - Easy to implement
  - Difficult to shouting troubles
- Often prior knowledge / model complexity is embedded in initialization and/or training process
- Suitable for parallel/distributed computation
- Optimization in continuous space

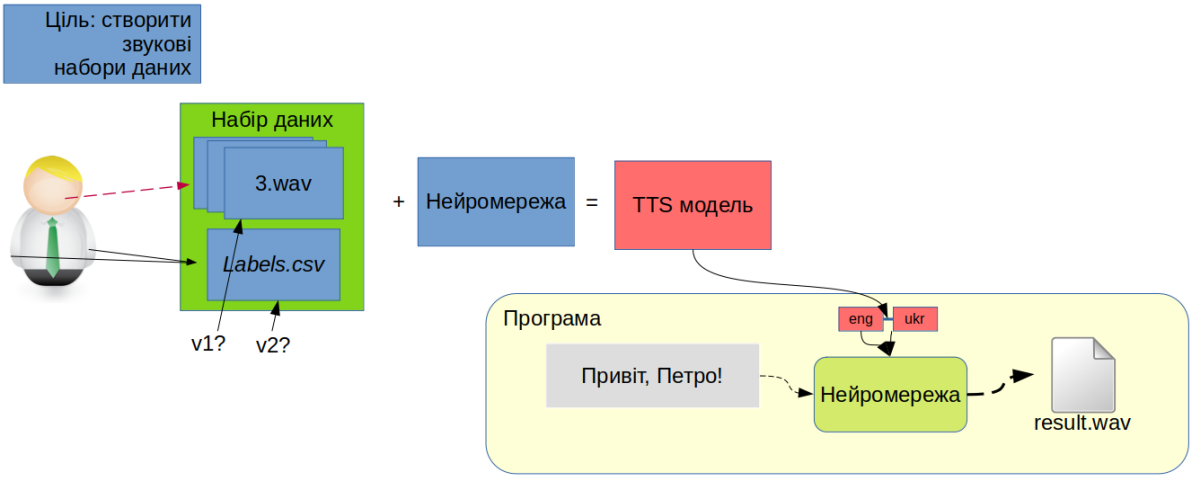
06.



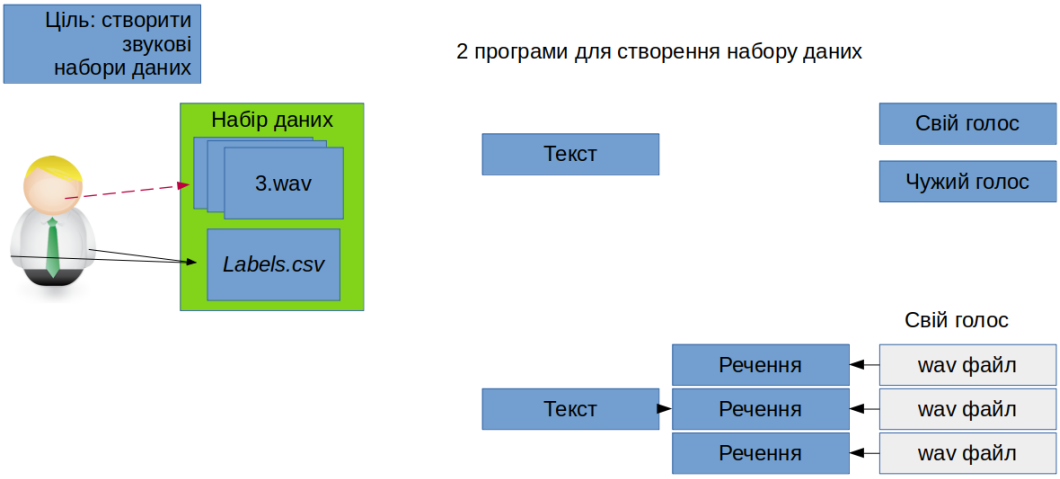

**Предметна область** 05.



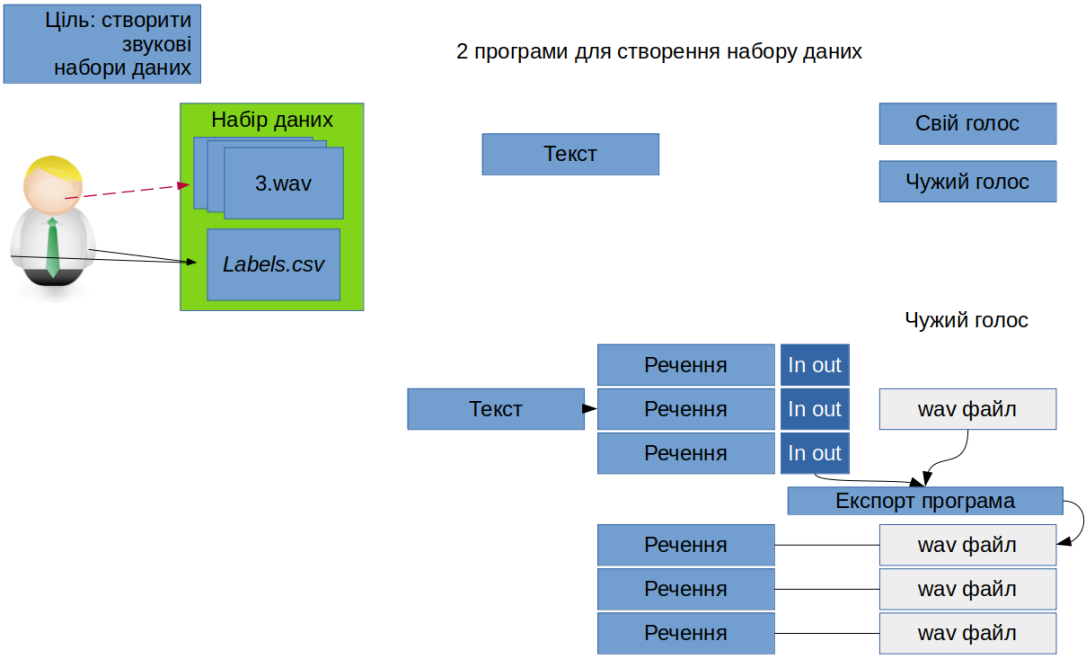

**Предметна область** 05.


Ціль: створити звукові набори даних
Набір даних
3.wav
Labels.csv
v1?
v2?
+
Нейромережа
=
TTS модель
Програма
eng
ukr
Привіт, Петро!
Нейромережа
result.wav

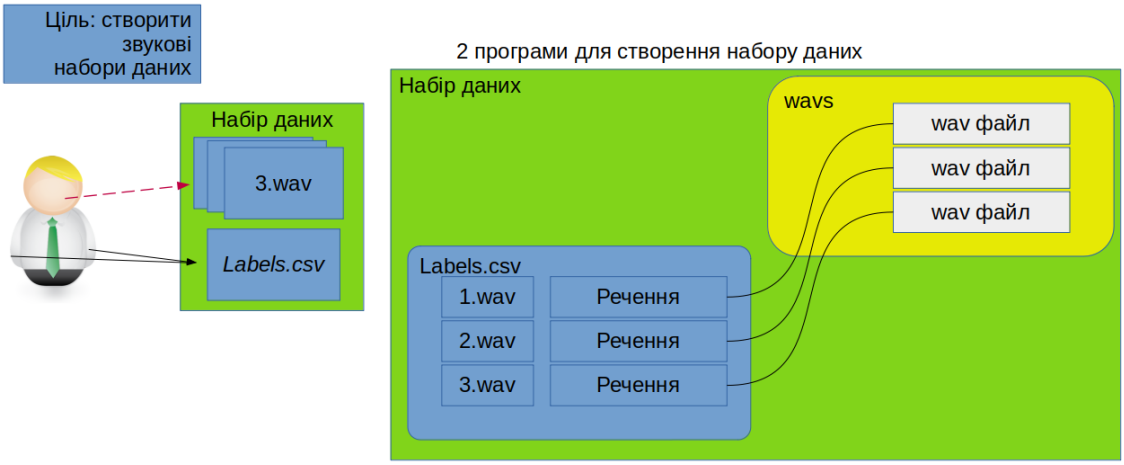
05.


Ціль: створити звукові набори даних
Набір даних
3.wav
Labels.csv
2 програми для створення набору даних
Текст
Свій голос
Чужий голос
Свій голос
Речення
wav файл
Текст
Речення
wav файл
Речення
wav файл

05.


Ціль: створити звукові набори даних
2 програми для створення набору даних
Набір даних
3.wav
Labels.csv
Текст
Свій голос
Чужий голос
Чужий голос
Речення
In out
Текст
Речення
In out
wav файл
Речення
In out
Експорт програма
Речення
wav файл
Речення
wav файл
Речення
wav файл

05.


Ціль: створити звукові набори даних
2 програми для створення набору даних
Набір даних
3.wav
Labels.csv
Набір даних
wavs
wav файл
wav файл
wav файл
Labels.csv
1.wav
Речення
2.wav
Речення
3.wav
Речення

05.

# Методи розробки

Базова архітектура нейронної мережі синтезатора в статті[1].

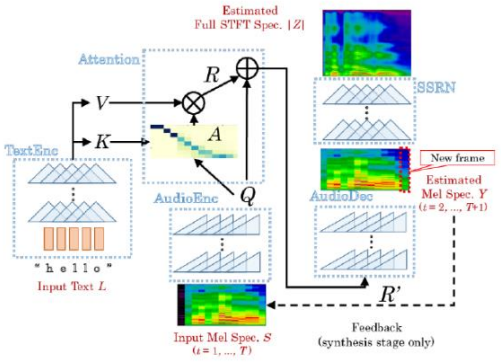



**Fig. 1**. Network architecture.

$$\mathsf{TextEnc}(L) := (\mathsf{HC}^{2d\leftarrow 2d}_{1\star 1})^2 \triangleleft (\mathsf{HC}^{2d\leftarrow 2d}_{3\star 1})^2 \triangleleft (\mathsf{HC}^{2d\leftarrow 2d}_{3\star 27} \triangleleft \mathsf{HC}^{2d\leftarrow 2d}_{3\star 9} \triangleleft \mathsf{HC}^{2d\leftarrow 2d}_{3\star 3} \triangleleft \mathsf{HC}^{2d\leftarrow 2d}_{3\star 1})^2 \triangleleft \mathsf{C}^{2d\leftarrow 2d}_{1\star 1} \triangleleft \mathsf{ReLU} \triangleleft \mathsf{C}^{2d\leftarrow e}_{1\star 1} \triangleleft \mathsf{CharEmbed}^{e\text{-dim}}(L).$$

$$\mathsf{AudioEnc}(S) := (\mathsf{HC}^{d\leftarrow d}_{3\star 3})^2 \triangleleft (\mathsf{HC}^{d\leftarrow d}_{3\star 27} \triangleleft \mathsf{HC}^{d\leftarrow d}_{3\star 9} \triangleleft \mathsf{HC}^{d\leftarrow d}_{3\star 3} \triangleleft \mathsf{HC}^{d\leftarrow d}_{3\star 1})^2 \triangleleft \mathsf{C}^{d\leftarrow d}_{1\star 1} \triangleleft \mathsf{ReLU} \triangleleft \mathsf{C}^{d\leftarrow d}_{1\star 1} \triangleleft \mathsf{ReLU} \triangleleft \mathsf{C}^{d\leftarrow F}_{1\star 1}(S).$$

$$\mathsf{AudioDec}(R') := \sigma \triangleleft \mathsf{C}^{F\leftarrow d}_{1\star 1} \triangleleft (\mathsf{ReLU} \triangleleft \mathsf{C}^{d\leftarrow d}_{1\star 1})^3 \triangleleft (\mathsf{HC}^{d\leftarrow d}_{3\star 1})^2 \triangleleft (\mathsf{HC}^{d\leftarrow d}_{3\star 27} \triangleleft \mathsf{HC}^{d\leftarrow d}_{3\star 9} \triangleleft \mathsf{HC}^{d\leftarrow d}_{3\star 3} \triangleleft \mathsf{HC}^{d\leftarrow d}_{3\star 1}) \triangleleft \mathsf{C}^{d\leftarrow 2d}_{1\star 1}(R').$$

$$\mathsf{SSRN}(Y) := \sigma \triangleleft \mathsf{C}^{F'\leftarrow F'}_{1\star 1} \triangleleft (\mathsf{ReLU} \triangleleft \mathsf{C}^{F'\leftarrow F'}_{1\star 1})^2 \triangleleft \mathsf{C}^{F'\leftarrow 2c}_{1\star 1} \triangleleft (\mathsf{HC}^{2c\leftarrow 2c}_{3\star 1})^2 \triangleleft \mathsf{C}^{2c\leftarrow c}_{1\star 1} \triangleleft (\mathsf{HC}^{c\leftarrow c}_{3\star 3} \triangleleft \mathsf{HC}^{c\leftarrow c}_{3\star 1} \triangleleft \mathsf{D}^{c\leftarrow c}_{2\star 1})^2 \triangleleft (\mathsf{HC}^{c\leftarrow c}_{3\star 3} \triangleleft \mathsf{HC}^{c\leftarrow c}_{3\star 1}) \triangleleft \mathsf{C}^{c\leftarrow F}_{1\star 1}(Y).$$

[1] - Tachibana H., Uenoyama K., Aihara S. Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention //2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). – IEEE, 2018. – С. 4784-4788.



# Методи вдосконалення

**Loss**: L1, cross entropy

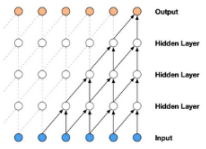
**Стохастична оптимізація**: *Adam[Kingma et al, 2014]*

**Увага**: *Адитивна увага (увага Багданау)*
звернення до різних частин вхідного вектора для фіксації довгострокових залежностей

**Регуляризація**: *Dropout [Srivastava, 2014]*
скидає одиницю (разом із сумою поєднання с шаром мережі) коли проходить навчання при вказаній ймовірності

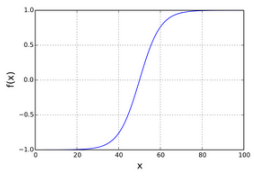
**Learning rate decay**(розклад для lr):
*Adafactor[arXiv:1804.04235]*
Розклад значень lr під час навчання

**Causal Convolution**
[Oord et al. in WaveNet]



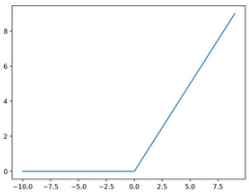

**Функції активації**

Sigmoid



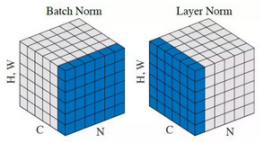

$$f(x) = \frac{1}{(1+\exp(-x))}$$

ReLU



$$f(x) = \max(0, x)$$

**Нормалізація**: *Layer Normalization*
(полегшення оптимізації, згладжуючи поверхню втрат мережі)




$$\mu^l = \frac{1}{H}\sum_{i=1}^{H} a_i^l$$

$$\sigma^l = \sqrt{\frac{1}{H}\sum_{i=1}^{H}\left(a_i^l - \mu^l\right)^2}$$

# Методи

| | |
|---|---|
| Мова програмування | Python 3 |
| Бібліотека для тренування | TensorFlow >= 1.3 |

Синтезатор

| | |
|---|---|
| Мова програмування | Python 3, HTML/CSS/Javascript |
| Фреймворки | Flask 1.1, p5.js, jq |
| СУБД | MySQL 5.7 |
| ОС | Ubuntu 20 Server |

Соціальна мережа



# Отримані результати

Отримано  end2end модель глибокої нейронної мережі  для озвучення тексту українською мовою

На етапі підготовки даних  було розроблені
- програма для маркування тексту (створення набору аудіоданих)
- програма для озвучення тексту (створення набору аудіоданих)
- соціальна платформ (RESTful API)
- клієнтських додаток для серверу

Рівень якості синтезу голосу оцінкою MOS: ≈3.0
Тренування нейронних мереж:
- базова модель голосу - від 1 до 2 днів
- персональних голос - від 30хв до 3 годин

# Наукова новизна

1. **Уточнено гіперпараметри моделі** [1] на нових наборах даних, а саме max_N = 569, max_T = 988, lr=0.002, український корпус.
Аналіз гіперпараметрів моделі дозволив зробити висновок, що найбільший вплив на тренування має динамічна зміна параметру lr (за допомогою learning rate decay) під час навчання, що дозволяє плавно проводити процес fine-tune (налагодження) моделі. На практиці було створено модель для синтезу українського голосу (STTSv2020.v1);
Порівняння показників якості розробленої нейромережевої моделі з аналогами показало схожість якості з моделлю Tacotron і задовільну якість синтезу.

[1] - Tachibana H., Uenoyama K., Aihara S. Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention //2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). – IEEE, 2018. – С. 4784-4788.



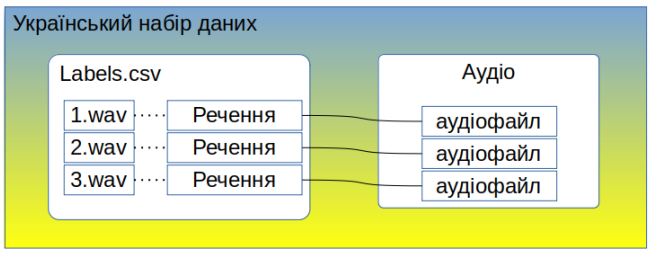
# Наукова новизна

2. Удосконалений метод накопичення і збагачення звукових наборів даних шляхом **розробки соціальної платформи зі спеціальним інтерфейсом** для колективного озвучення і поміток текстів
та створення звукових наборів даних.

*Програми data_collection:recorder, data_collection:marker, web\desktop

# Наукова новизна

3. Вперше створено розширений український набір аудіоданих




09.

# Практичне значення

- озвучення книги\повідомлень своїм *голосом* чи *голосом інших людей*
- вбудування штучного голосу в комп’ютерні ігри, де персонажі гри читають текст відомими для гравця голосами для збагачення **атмосфери гри.**
- використання згенерованого власного голосу для VR; створення почуття "внутрішнього голосу", який буде включатись під час гри гравцю в потрібний момент.
- допомога сліпим людям
- створення відео
    а) під часу монтажу сюжету
    б) під часу живої трансляції
- озвучення довільних слів

10.

# Публікації

Савінський В. В. Social platform for communication and labeled audio datasets // ЗБІРНИК НАУКОВИХ ПРАЦЬ за матеріалом XII всеукраїнської науково-практичної конференції[«Актуальні проблеми комп'ютерних наук АПКН-2020»], (Хмельницький), X жовтня - 10 листопада 2020 г.): - Хмельницкий: ХНУ, 2020. – XX-XX.

11.

# Висновки та рекомендації

***Висновки та рекомендації***

В даній роботі був
⇒проведений аналіз класичних методів і методів ML для озвучення голосу;
⇒побудована модель на базі нейромереж, яка дозволяє озвучувати тексти голосом окремих людей;
⇒розроблена архітектура соціальної платформи для колективного створення звукових наборів даних;
⇒створений український аудіо набір даних.

***Майбутнє***

- інтеграція подібних моделей для кращого синтезу голосу;
- моделювання тонких емоції, персональних властивостей голосу;
- дослідження моделей для синтезу довгих послідовностей (довгі речень);
- синтез людського співу, музики.

Дослідження генерації кількох голосів однією моделлю, активне моделювання ритму голосу, вдосконалені нейронні вокодери.

12.

Дякую за увагу, доповідь закінчено

# Anti-Plagiarism v-15.257

**Максимальне співпадіння з одним документом 1.0%**

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 10%**

| | Документ | | Сумарний збіг по Базі Даних | |
|---|---|---|---|---|
| ID: 82488<br>Назва: Технологія розробки програмної системи для озвучення тексту голосом людини на основі машинного навчання<br>Додано в БД: 2020-12-04<br>Автора: В. В. Савінський<br>Керівники: Л. П. Бедратюк<br>Консультанти:<br>Опоненти: | Символи | Лексеми | Символи | Лексеми |
| | 102818 | 971 | 2187 (2%) | 32 (3%) |

Джерело плагіату

| ID | Опис | Наявність плагіату в документі | |
|---|---|---|---|
| | | Символи | Лексеми |

UNICHECK

Ім'я користувача:
**Кафедра ІПЗ**

ID перевірки:
**1005370612**

Дата перевірки:
**04.12.2020 20:20:04 EET**

Тип перевірки:
**Doc vs Internet + Library**

Дата звіту:
**04.12.2020 20:33:34 EET**

ID користувача:
**100005589**

---

Назва документа: **DR_WORK5**

Кількість сторінок: **113** Кількість слів: **15897** Кількість символів: **119622** Розмір файлу: **11.64 MB** ID файлу: **1005663221**

---

1 слово позначене як "вилучене" та не враховується у підрахунку слів

## 4.96%
## Схожість

Найбільша схожість: **1.57%** з джерелом з Бібліотеки (ID файлу: **1005655523**)

| | | |
|---|---|---|
| 3.89% Джерела з Інтернету | 295 | Сторінка 115 |
| 1.57% Джерела з Бібліотеки | 30 | Сторінка 117 |

## 0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

## 0%
## Вилучень

Немає вилучених джерел

## Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

| | |
|---|---|
| Замінені символи | 4 |

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ КАФЕДРИ **ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ** ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Технологія розробки програмної системи для озвучення тексту голосом людини на основі машинного навчання

Автор: В. В. Савінський

Спеціальність:121 Інженерія програмного забезпечення

Освітня програма: Інженерія програмного забезпечення

Науковий керівник:д.ф.-м.н., проф. Бедратюк Л.П.

Після аналізу звіту подібності зроблено такий висновок:

| № | Висновок | Позначка про відповідність |
|---|---|---|
| 1 | Запозичення, виявлені в роботі, є законними і не є плагіатом Робота приймається до захисту. | **відповідає** |
| 2 | Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи | |
| 3 | Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат. | |
| 4 | Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту. | |
| 5 | Інше: | |

**Підтвердження**: Текст є оригінальним, виявлені запозичення не є плагіатом оскільки розміщені в розділах які не описують безпосередньо авторське дослідження, складають 4.96% та мають посилання на літературні джерела. Робота приймається до захисту.

4.11.2020

Підпис керівника _____

Підпис завідувача кафедри _____

Підпис гаранта ОП _____

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

# РЕЦЕНЗІЯ НА ДИПЛОМНУ РОБОТУ

Дипломник Савінський Владислав В'ячеславович

Тема Технологія розробки програмної системи для озвучення тексту голосом людини на основі машинного навчання

Спеціальність 121 – Інженерія програмного забезпечення

## Обсяг дипломної роботи:

Кількість листів креслень _____; кількість сторінок записки 113

1.Короткий зміст ДР та прийнятих рішень Робота складається з таких основних розділів: вступ, аналіз предметної області розробка нейронної мережі та соціальної платформи, проектування каркасу ПЗ, розробка і тестування ПЗ, висновки, перелік посилань, додатки. У першому розділі описані найважливі дослідження останніх років в області синтезу голосу; проаналізовані наявні методи збору наборів аудіоданих, де описані деталі кожної з платформи. В кінці є висновок та чітка постановка задачі, яка заключається в побудові платформи для отримання аудіоданих і побудові нейромережевої системи з її подальшим навчання на зібраних даних з платформи. У другому розділі формально описана структура глибинних нейромереж, математичний апарат для роботи з аудіо і комплекс нейромереж для задачі озвучення тексту. Наявні зрозумілі діаграми, що візуалізують і доповнюють опис. У третьому розділі гарно представлена вся система для синтезу і збору даних в детальних схемах з параметрами. У четвертому розділі магістрант описує інструменти розробки і звертає увагу на ліцензію доступних інструментів, серед яких він вирішив обрати вільне програмне забезпечення.

2. Висновок про відповідність ДР поставленому завданню Дипломна робота виконана у відповідності до завдання та в повному обсязі у встановлений термін.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: Глибина обґрунтування рішень магістраста базується проблемі з наявністю українських наборів даних для створення системи нейронних мереж для синтезу голосу в цілях озвучення тексту. У першому розділ проведений комплексний аналіз процесу сприйняття інформації в різних формах, що свідчить про наявність систематичного підходу в роботі. Другий розділ описується алгоритми алгоритми для швидкої обробки звуку, FFT, STFT і різні форми спектограми з діаграмами та поясненням, що свідчить про використання спеціалізованого програмного забезпечення; побудований каркас для бази даних соціальної платформи з описом сутностей в базі даних, що логічно зв'язано з першим розділом; викладені методи для оптимізації глибинних нейронних мереж. На основі цих формальних мат. методів, у третьому розділі зображені розвернуті структури нейромережених модулі та тренувальний цикл нейромережі, а в четвертому розділі магістрант показує результати своєї роботи, де порівнює їх з іншими дослідженнями та моделями, що описані в першому розділі.

4. Позитивні сторони роботи Факт проведення аналізу методів синтезу голосу на основі нейронних мереж. Використання сучасних інструментів для моделювання і реалізації нейронних мереж, використання сучасних веб-застосутку. Створення ПЗ, що об'єднує роботу колективу. Збагачення українського поміченого набору аудіоданих.

5. Негативні сторони роботи _____
Недоробка елементів управління зворотнього зв'язку, які потребують
залучення спеціаліста в області графічного проектування
користувацького веб-дизайну.
В цілому, недолік не зменшує позитивне враження
від роботи, яку виконано на високому рівні.

6. Оцінка графічного оформлення та пояснювальної записки роботи _____
В усіх розділах наведені чіткі рисунки та діаграми, які наочно зображують
і конкретизують ідеї магістранта. В другому розділі описується
поняття звуку, опис роботи зі звуком і методи представлення звуку в різних
форматах, які наглядно підкріплені рисунками. Робота написана в великою
кількістю списків, що придає ієрархічну модель опису питань на
початку підрозділів.

7. Відгук про роботу в цілому _____
В кожному розділі наведені висновки, де магістрант обгрунтував свій вибір
з декількох практичних сторін. Достовірність результатів показана
на графіках і діаграмах. Дана робота має тісний зв'язок з практикою, оскільки
сфера застосування створеної системи широка: від системи сповіщення
до озвучення книг.

8. Інші зауваження рекомендується зробити окремі програми у вигляді
настільної реалізації, які не потребують запуску веб-серверу і наявності
браузера для роботи

9. Оцінка дипломної роботи _____
Вважаю, що робота задовольняє вимогам, які пред'являються
до дипломних робіт і заслуговує на оцінку «відмінно»

РЕЦЕНЗЕНТ (прізвище, ім'я, по-батькові, посада, місце роботи) _____
Говорущенко Тетяна Олександрівна, доктор технічних наук, професор,
зав. кафедри комп'ютерної інженерії та системного програмування ХНУ

"2" грудня 2020 р. _____
(підпис)