ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут економіки, управління, права та інформаційних технологій
Кафедра інформаційних систем та технологій

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавр

на тему: **«Генерація аудіо з текстових даних на основі технологій штучного інтелекту»**

Виконав: здобувач вищої освіти
за освітньою програмою
Інформаційні управляючі системи
спеціальності 126 Інформаційні
системи та технології
ступеня вищої освіти бакалавр
групи 126ІСТ_бд_2022
Пивовар Богдан Олександрович
Керівник: Слюсарь Ігор Іванович
Рецензент: Муравльов Володимир
В’ячеславович

Полтава – 2026 року

ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут економіки, управління, права та інформаційних технологій
Кафедра інформаційних систем та технологій

Освітня програма Інформаційні управляючі системи
Спеціальність 126 Інформаційні системи та технології
Рівень вищої освіти перший (бакалаврський)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Юрій УТКІН
«10» червня 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Пивовара Богдана Олександровича

1. Тема кваліфікаційної роботи:
«Генерація аудіо з текстових даних на основі технологій штучного інтелекту»,
Керівник роботи: к. т. н., доцент, доцент кафедри інформаційних систем та технологій Слюсарь Ігор Іванович.
Затверджено наказом закладу вищої освіти від «10» квітня 2026 року № 383-ст

2. Строк подання здобувачем вищої освіти роботи «08» червня 2026 року.

3. Вихідні дані до роботи: технології TTS, моделі TTS з підтримкою україномовного контенту (ElevenLabs, Google Cloud Text-to-Speech, Supertonic 3, StyleTTS2, Piper TTS, Balacoon TTS, Ukrainian TTS, Narakeet), системи клонування голосу, Google Colab, бібліотеки Python

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
Розділ 1 Аналіз особливостей застосування сучасних систем генерація аудіо з текстових даних
Розділ 2. Розробка моделі локальної системи для генерації аудіо з текстових даних
Розділ 3. Рекомендації щодо практичної реалізації локальної системи для генерації аудіо з текстових даних

5. Перелік графічного матеріалу: схеми, рисунки, діаграми за темою та об’єктом дослідження

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання отримав
Оцінювання економічної ефективності результатів дослідження	Дивнич О. Д., к. е. н., доцент, доцент кафедри економіки та публічного управління	01.04.2026 р.	29.05.2026 р.

7. Дата видачі завдання «10» червня 2025 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Вибір і затвердження теми роботи	02.06.2025 р. – 04.06.2025	
2	Складання і затвердження розгорнутого плану та завдання на кваліфікаційну роботу	05.06.2025 р. – 10.06.2025 р.	
3	Опрацювання джерел інформації	11.06.2025 р. – 15.06.2025 р.	
4	Збір, вивчення і обробка інформації, необхідної для виконання роботи	16.06.2025 р. – 20.06.2025 р.	
5	Виконання теоретико-методологічного розділу роботи	08.09.2025 р. – 01.11.2025 р.	
6	Виконання дослідницько-аналітичного розділу роботи	19.01.2026 р. – 14.02.2026 р.	
7	Виконання проєктно-рекомендаційного розділу роботи	16.02.2026 р. – 31.03.2026 р.	
8	Оцінювання економічної ефективності результатів дослідження	01.04.2026 р. – 29.05.2026 р.	
9	Оформлення тексту роботи	01.06.2026 р. – 06.06.2026р.	
10	Попередній захист роботи на кафедрі	08.06.2026 р.	
11	Доопрацювання роботи з урахуванням зауважень і пропозицій	09.06.2026 р. – 10.06.2026 р.	
12	Нормоконтроль	11.06.2026 р. – 15.06.2026 р.	
13	Захист кваліфікаційної роботи	16.06.2026 р. - 18.06.2026 р.	

Здобувач вищої освіти **Богдан ПИВОВАР**

Керівник роботи **Ігор СЛЮСАРЬ**

ЗМІСТ

ВСТУП

Актуальність теми кваліфікаційної роботи підтверджується необхідністю застосування генерації аудіо з текстових даних (Text-to-Speech, TTS) для озвучення текстів, створення голосових асистентів, підтримки користувачів, навчальних матеріалів та ін. Підтримка української мови є важливою умовою практичного використання TTS в україномовному середовищі. Для забезпечення конфіденційності та безпеки інформації зараз впроваджуються локальні інтелектуальні агенти. Застосування у них TTS в якості голосового модуля дозволяє системі на основі штучного інтелекту (Artificial Intelligence, AI) відповідати користувачу не лише текстом, а й аудіо. В той же час, якість озвучення залежить не лише від самої моделі, а й від правильної обробки тексту, наголосів, чисел, скорочень і власних назв. Тому для вибору TTS-рішення необхідно враховувати природність звучання, стабільність вимови та можливість інтеграції в ІС. Все це свідчить про актуальність теми роботи.

Метою кваліфікаційної роботи є підвищення ефективності локальних AI-агентів за рахунок використання TTS з підтримкою україномовного контенту.

Завданнями кваліфікаційної роботи є:

– обґрунтувати вибір інструментарію для реалізації TTS з підтримкою україномовного контенту;

– розробити модель локального рішення TTS з підтримкою україномовного контенту;

– сформувати рекомендацій щодо практичної реалізації модель локального рішення TTS з підтримкою україномовного контенту;

– виконати економічне обґрунтування прийнятих рішень.

Об’єктом дослідження є процес перетворення текстової інформації у мовленнєве аудіо за допомогою сучасних AI-моделей синтезу мовлення.

Предметом дослідження є моделі та програмні засоби побудови локального TTS-рішення для генерації аудіо з тексту.

Методами дослідження є аналітичний – для вивчення сучасних TTS з підтримкою української мови та економічного обґрунтування; порівняльний аналіз – для зіставлення локальних рішень TTS-систем і визначення їх переваг і обмежень; моделювання – для розробки моделі локальної TTS-системи, її основних компонентів, функцій і взаємозв'язків між ними; експериментальний – для практичної перевірки роботи моделі Supertonic 3 у середовищі Google Colab.

Інформаційна база кваліфікаційної роботи сформована з Інтернет-ресурсів, що місять інформацію про технології генерації аудіо з текстових даних, сучасні моделі TTS з підтримкою україномовного контенту, системи клонування голосу, локальні програмні рішення TTS.

Практична значущість роботи полягає у розробці моделі локальної системи для генерації аудіо з текстових даних з підтримкою україномовного контенту; формуванні рекомендацій щодо практичної реалізації локальної системи для генерації аудіо з текстових даних з підтримкою україномовного контенту. Вони можуть бути використані для подальших досліджень за даною тематикою та при проектуванні локальних AI-агентів

Апробація результатів відбувалася в рамках XXI щорічної студентської наукової конференції «Сучасні інформаційні технології та інноваційні методики в економіці, менеджменті та бізнесі» Полтавського державного аграрного університету (22 квітня 2026 р., м. Полтава).

За результатами досліджень здійснено публікацію тез доповідей.

Структура кваліфікаційної роботи логічно пов'язана з завданнями досліджень і містить вступ, три розділи основної частини, висновки, список використаних джерел, додатки. Загальний обсяг пояснювальної записки кваліфікаційної роботи складає 63 сторінки формату А4. Вона містить 13 рисунків і 4 таблиці.

РОЗДІЛ 1
АНАЛІЗ ОСОБЛИВОСТЕЙ ЗАСТОСУВАННЯ СУЧАСНИХ СИСТЕМ ГЕНЕРАЦІЯ АУДІО З ТЕКСТОВИХ ДАНИХ

1.1 Галузі застосування систем генерація аудіо з текст і клонування голосу

Технології TTS сьогодні активно використовуються в багатьох сферах, де потрібно автоматично перетворювати текстову інформацію в аудіо (рис. 1.1). Їх розвиток пов'язаний із поширенням AI, нейронних мереж і генеративних моделей [1].

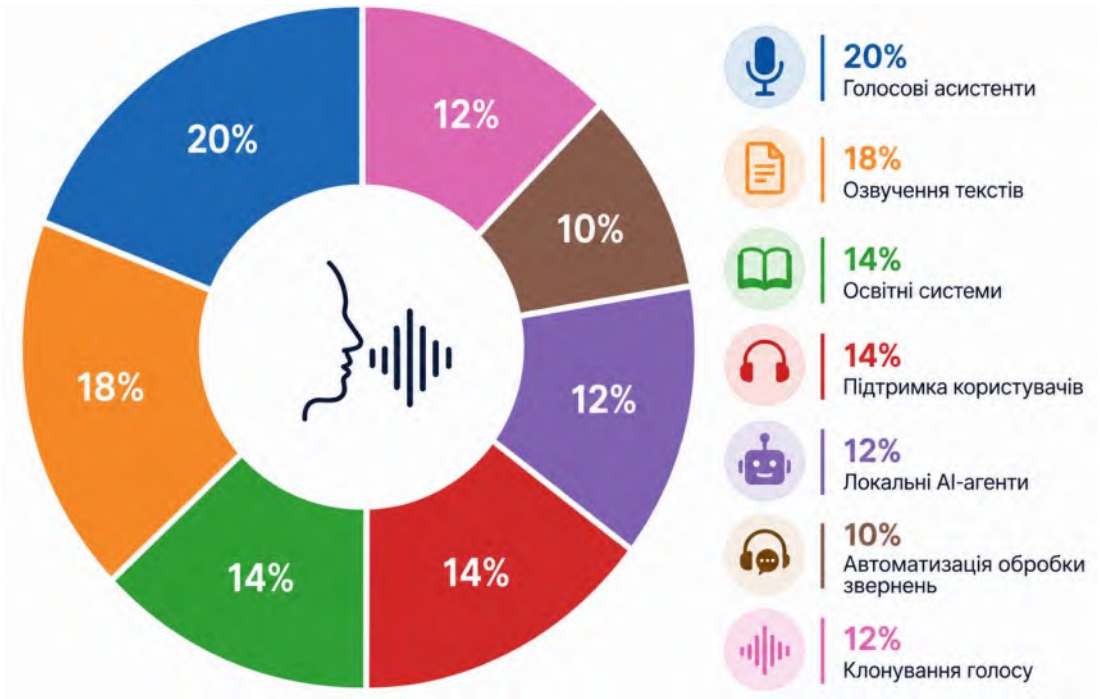

Рисунок 1.1 – Застосування TTS і клонування голосу

Вони дають змогу створювати більш природне, зрозуміле та наближене до людського мовлення аудіо. Якщо раніше синтезоване мовлення часто звучало механічно, то сучасні AI-рішення можуть передавати інтонацію, темп, паузи, емоційне забарвлення та навіть індивідуальні особливості голосу.

Однією з галузей застосування TTS-систем є голосові асистенти. Вони використовуються у смартфонах, розумних колонках, автомобільних мультимедійних системах, побутових пристроях та програмних помічниках. Завдяки TTS користувач може отримувати відповідь не лише у вигляді тексту, а й у формі голосового повідомлення. Це робить взаємодію з інформаційною системою більш природною та зручною, особливо у випадках, коли користувач не може або не хоче читати текст з екрана.

Важливим напрямом є озвучення текстових матеріалів. TTS-системи можуть застосовуватися для створення аудіоверсій статей, книг, інструкцій, новин, навчальних матеріалів, повідомлень або технічної документації. Це особливо корисно для користувачів із порушеннями зору або кращим сприйняттям інформації на слух. Крім того, автоматичне озвучення дозволяє значно скоротити час і витрати порівняно із залученням дикторів або студій звукозапису.

Окреме місце займають TTS в освітніх системах. Вони можуть використовуватися для озвучення навчальних курсів, електронних підручників, тестових завдань, пояснень до лабораторних робіт або інтерактивних тренажерів. Також TTS може бути корисним під час вивчення іноземних мов (дає змогу прослуховувати правильну вимову слів і речень).

Ще одним практичним напрямом є підтримка користувачів. У службах підтримки, контакт-центрах і сервісних платформах TTS-системи можуть застосовуватися для автоматичного озвучення відповідей, повідомлень про статус звернення, інструкцій або результатів обробки запиту. Наприклад, користувач може залишити звернення, а система після його обробки автоматично сформує голосову відповідь. Це зменшує навантаження на операторів і прискорює комунікацію з клієнтами.

Особливо перспективним є використання TTS у складі локальних AI-агентів. У такому випадку система може працювати без передавання даних до хмарних сервісів, що є важливим для конфіденційної інформації. Локальний AI-агент може приймати текстові дані, аналізувати їх за допомогою мовної

моделі, формувати відповідь і озвучувати її за допомогою TTS-модуля. Такий підхід може бути корисним у навчальних закладах, медичних установах, державних організаціях, підприємствах та інших сферах, де важливо зберігати контроль над даними.

TTS-системи також можуть застосовуватися для автоматизації обробки звернень. Наприклад, у системі підтримки користувачів текстові повідомлення, заявки або результати аналізу можуть автоматично перетворюватися в аудіо. Це дозволяє створювати голосові сповіщення, аудіозвіти або інтерактивні відповіді. У поєднанні з ASR-системами, які перетворюють мовлення в текст, TTS може бути частиною повного циклу голосової взаємодії: користувач говорить, система розпізнає мовлення, аналізує запит і відповідає синтезованим голосом.

Окремим напрямом розвитку є клонування голосу [2]. Клонування голосу передбачає створення синтезованого мовлення, яке імітує голос конкретного користувача. Для цього система використовує зразок голосу, аналізує його тембр, інтонацію, манеру вимови та інші акустичні особливості, після чого може генерувати нове мовлення заданим голосом. Така технологія може застосовуватися для персоналізованих голосових асистентів, дубляжу, озвучення навчальних матеріалів, створення аудіоконтенту або відновлення голосу для людей, які втратили можливість говорити. Разом із тим клонування голосу потребує обережного використання. Оскільки голос є індивідуальною характеристикою людини, його копіювання повинно виконуватися лише за згодою власника голосу. Також необхідно враховувати ризики несанкціонованого використання, створення фальшивих аудіозаписів або введення людей в оману. Тому під час розроблення таких систем важливо передбачати етичні обмеження, захист персональних даних і чітке маркування синтезованого аудіо.

Отже, TTS-системи мають широкий спектр практичного застосування: від голосових асистентів та озвучення текстів до локальних AI-агентів і систем клонування голосу. Найбільшу цінність вони мають у тих випадках,

коли потрібно швидко, автоматично та з мінімальними витратами створити мовленнєве аудіо з текстових даних. У межах цієї кваліфікаційної роботи особливу увагу доцільно приділити локальним TTS-рішенням, оскільки вони забезпечують більшу автономність, конфіденційність і можливість практичного використання в AI-системах. Це відповідає загальному напряму роботи, де передбачено аналіз TTS, акцент на локальні рішення, галузі їх застосування та задачі клонування голосу.

1.2 Підтримка української мови

Синтез мовлення українською мовою є одним напрямів розвитку сучасних AI-систем, оскільки дає змогу перетворювати текстову інформацію на природне усне мовлення. Для україномовного користувача такі рішення можуть застосовуватися в електронному навчанні, озвученні відео, аудіокнигах, голосових асистентах, системах доступності, навігаційних сервісах, чат-ботах, інформаційних системах підприємств та AI-агентах. На відміну від англійської мови, для якої існує значна кількість якісних голосів, україномовний TTS тривалий час мав обмежену кількість рішень. Зараз ситуація покращилася: поява нейромережевих голосів українською мовою у великих комерційних платформах, відкриті моделі, експериментальні проєкти та спеціалізовані системи для синтезу українського мовлення.

До номенклатури наявних рішень для україномовного синтезу мовлення можна віднести універсальні TTS-платформи, спеціалізовані сервіси озвучення, бібліотеки для програмної інтеграції, відкриті нейромережеві моделі та системи екранного доступу. Серед найбільш відомих рішень, які підтримують українську мову, можна виділити ElevenLabs [3], Google Cloud Text-to-Speech [4], Narakeet [5], Microsoft Azure Speech [6], gTTS [7], RHVoice [8], Piper [9], Balacoon TTS [10], Ukrainian TTS на основі ESPnet [11], а також окремі експериментальні моделі на базі VITS:

StyleTTS2 [12], Coqui TTS [13] або інших фреймворків синтезу мовлення. Одним із найпомітніших класів рішень є великі універсальні нейромережеві TTS-системи, які мають підтримку української мови у вигляді готових голосів. Наприклад, Google Cloud Text-to-Speech має для української мови голоси стандартного, WaveNet- і Chirp3-HD-рівня, тобто підтримує як базовий, так і більш природний нейромережевий синтез мовлення. В офіційному переліку Google для uk-UA наведено значну кількість голосів Chirp3-HD, а також голоси uk-UA-Standard-B і uk-UA-Wavenet-B. Це свідчить про те, що українська мова вже представлена не лише мінімальною підтримкою, а й сучаснішими моделями синтезу [14].

Схожі можливості має Microsoft Azure Speech. Для української мови в Azure доступні нейронні голоси uk-UA-PolinaNeural та uk-UA-OstapNeural, тобто жіночий і чоловічий варіанти синтезованого мовлення. Такі голоси можна використовувати для озвучення текстів, побудови голосових інтерфейсів, навчальних матеріалів, інформаційних повідомлень або інтеграції у програмні системи. Крім того, Azure підтримує для української мови функції, пов’язані з віземами, що може бути корисним для синхронізації мовлення з анімованими аватарами або мультимедійними інтерфейсами [15].

Окрему групу становлять сучасні генеративні сервіси синтезу мовлення, орієнтовані на природність звучання, емоційність і створення голосового контенту. До таких рішень можна віднести ElevenLabs, Cartesia, Respeecher та інші подібні платформи. Наприклад, ElevenLabs декларує підтримку українського TTS і позиціонує його як засіб для створення природного, емоційного та чіткого мовлення українською мовою [3]. Такі системи зазвичай краще підходять для озвучення відео, презентацій, навчального контенту, рекламних матеріалів та аудіокниг, оскільки вони забезпечують вищу природність інтонації порівняно з класичними синтезаторами. Водночас їхнім обмеженням може бути менший контроль над внутрішньою роботою моделі, залежність якості від конкретного голосу, можливі помилки у вимові складних українських слів, власних назв,

абревіатур або змішаних українсько-англійських фраз. До практичних інструментів швидкого озвучення також належать сервіси на кшталт Narakeet [5] і gTTS [7]. Narakeet пропонує український text-to-speech як готовий інструмент для перетворення тексту на аудіофайл і підтримує велику кількість мов. gTTS є простим програмним засобом, який часто використовується в навчальних або демонстраційних Python-проєктах для швидкого синтезу мовлення. Його перевагою є простота використання, але недоліком є обмежений контроль над голосом, інтонацією, темпом, стилем мовлення та якістю вимови. Тому gTTS доцільно розглядати як інструмент для простих демонстрацій або прототипів, але не як оптимальне рішення для професійного озвучення.

Важливе місце серед україномовних TTS-рішень займають відкриті та дослідницькі проєкти. Наприклад, Ukrainian TTS від robinhad реалізує синтез українського мовлення з використанням ESPnet і має відкриту демонстрацію на Hugging Face [16]. Такі проєкти мають велике значення для розвитку україномовної інфраструктури, оскільки вони дають можливість дослідникам і розробникам експериментувати з моделями, навчати власні голоси, аналізувати якість синтезу та адаптувати систему до конкретних задач. Їхня якість може бути достатньою для навчальних, наукових або експериментальних цілей, однак у деяких випадках вони можуть поступатися великим комерційним моделям за стабільністю, інтонаційною виразністю та якістю вимови довгих складних речень.

Серед відкритих або доступних для інтеграції рішень варто також згадати Piper. Це швидка система нейромережевого синтезу мовлення, для якої на Hugging Face доступний набір голосів piper-voices; у картці моделі зазначено, що набір містить голоси для Piper і охоплює 35 мов [9]. Для української мови в екосистемі Piper використовувалися голоси, підготовлені на основі україномовних датасетів. Перевагою таких рішень є висока швидкість роботи, можливість інтеграції в прикладні системи та придатність до експериментів. Водночас для української мови існували зауваження щодо

якості вимови, зокрема через помилки фонетичної обробки й некоректне трактування окремих українських літер або звуків. Це означає, що перед практичним використанням таких моделей необхідно тестувати вимову на реальних текстах: новинах, зверненнях користувачів, технічних інструкціях, числах, датах, абревіатурах та власних назвах [17].

Окремо слід відзначити Balacoon TTS, де було реалізовано нейромережевий синтез українського мовлення [10]. У матеріалах Balacoon зазначено, що український синтез доступний у вигляді Python-пакетів, Docker-контейнера, а також може працювати в режимі реального часу навіть на малопотужних пристроях, зокрема Raspberry Pi. Це робить такі рішення цікавими для практичного використання в голосових помічниках, локальних інформаційних системах, робототехніці, освітніх пристроях та AI-агентах. Їхньою перевагою є поєднання швидкості, доступності й орієнтації саме на українську мову. Проте якість конкретного результату залежить від голосу, навчального корпусу, попередньої обробки тексту та правильності наголосів.

Для задач доступності та озвучення інтерфейсів важливим рішенням є RHVoice. Це вільний багатомовний синтезатор мовлення, який має українські голоси, зокрема Anatol, Natalia, Marianna та Volodymyr [18]. RHVoice орієнтований насамперед на використання зі скрінрідерами, програмами читання тексту та допоміжними технологіями для людей із порушеннями зору. Його перевагами є зрозумілість мовлення, невисокі вимоги до ресурсів і придатність для тривалого читання тексту. Водночас природність такого синтезу зазвичай нижча, ніж у сучасних генеративних нейромережевих голосів. Тому RHVoice доцільно розглядати як практичне рішення для доступності, навігації та озвучення інтерфейсів, але не як найкращий варіант для емоційного або медійного озвучення.

Якість українського TTS залежить не тільки від самої AI-моделі, а і від лінгвістичної обробки тексту. Українська мова має низку особливостей, які ускладнюють синтез мовлення: рухомий наголос, відмінювання, чергування звуків, наявність апострофа, літери «ґ», складні власні назви, абревіатури,

числівники, дати, скорочення та англомовні технічні терміни. Наприклад, помилка в наголосі може зробити слово неприродним або навіть змінити його значення. Тому якісний TTS для української мови повинен не лише озвучувати символи, а й правильно виконувати нормалізацію тексту, визначати наголос, обробляти числа, одиниці вимірювання, скорочення та змішані мовні фрагменти.

За природністю звучання найкращі результати зазвичай демонструють сучасні нейромережеві та генеративні TTS-системи. Вони здатні формувати плавнішу інтонацію, природні паузи, емоційніше мовлення та менш «роботизований» голос. Такі рішення придатні для озвучення навчальних відео, презентацій, рекламних матеріалів, подкастів, аудіокниг і голосових інтерфейсів. Проте навіть вони можуть помилятися у вимові рідковживаних слів, прізвищ, географічних назв, технічних термінів або фраз із неправильним пунктуаційним оформленням. Тому для отримання якісного результату важливо готувати текст перед синтезом: розставляти розділові знаки, розшифровувати скорочення, перевіряти числа та за потреби вручну коригувати слова з неоднозначним наголосом.

Менш природні, але стабільні та швидкі синтезатори, такі як RHVoice або деякі легкі TTS-двигуни, краще підходять для функціональних задач: читання повідомлень, озвучення інтерфейсів, навігації, систем доступності, технічних сповіщень. Їхня основна перевага полягає не в емоційності, а в зрозумілості, швидкості та надійності. Водночас для медійного контенту, де важливі інтонація, тембр і природність, доцільніше використовувати сучасні нейромережеві голоси.

Отже, сучасна номенклатура TTS-рішень з підтримкою української мови є досить різноманітною. Вона охоплює як великі універсальні платформи з готовими нейромережевими голосами, так і спеціалізовані відкриті проєкти, бібліотеки для програмної інтеграції, системи доступності та експериментальні моделі. Найбільш придатними для практичного використання є рішення, які забезпечують достатню природність мовлення,

стабільну вимову українських слів, можливість налаштування параметрів синтезу та зручну інтеграцію в інформаційні системи. Водночас жодне рішення не є універсальним: вибір конкретного TTS-інструменту залежить від задачі, вимог до якості звучання, швидкодії, контрольованості голосу, обробки української орфоепії та умов подальшого використання.

Хмарні сервіси зазвичай забезпечують вищу природність звучання, більшу кількість голосів, API для інтеграції та зручний вебінтерфейс. Тобто, перевагою хмарних TTS-сервісів є простота використання: користувачу достатньо ввести текст, вибрати голос і отримати аудіофайл або підключити API до власного програмного продукту. Такі сервіси часто мають більш природну інтонацію, стабільнішу вимову та більшу кількість готових голосів. Водночас їхнім обмеженням є залежність від інтернет-з’єднання, наявність тарифних планів, обмеження на кількість символів або хвилин синтезу, а також передавання текстових даних на зовнішні сервери. Для проєктів, де обробляється конфіденційна інформація, це може бути суттєвим недоліком.

1.3 Локальні програмні рішення

Локальні TTS-рішення є важливим напрямом розвитку систем синтезу мовлення, оскільки дають змогу виконувати перетворення текстових даних в аудіо без постійного звернення до зовнішніх хмарних сервісів. У такому випадку програмне забезпечення, мовна модель і процес генерації мовлення розміщуються безпосередньо на комп’ютері користувача, локальному сервері або edge-пристрої. Це особливо актуально для систем, які працюють із персональними, службовими або конфіденційними текстовими даними.

Основною перевагою локального запуску TTS-систем є підвищення рівня конфіденційності. У хмарних сервісах текст, який потрібно озвучити, передається на зовнішній сервер, де відбувається його обробка. У випадку локального TTS-рішення текстові дані залишаються в межах локальної

інфраструктури, що зменшує ризики витоку інформації та спрощує дотримання вимог до захисту даних. Це важливо для освітніх, медичних, корпоративних, державних та інших інформаційних систем, де зміст повідомлень може містити чутливу інформацію.

Ще однією перевагою є незалежність від хмарних платформ, тарифних обмежень, API-ключів і стабільності інтернет-з'єднання. Локальна TTS-система може працювати автономно після встановлення необхідних моделей і програмних залежностей. Це дозволяє використовувати синтез мовлення в умовах обмеженого або нестабільного доступу до мережі, а також у випадках, коли використання зовнішніх сервісів є небажаним з організаційних або безпекових причин. Крім того, локальне розгортання дає більший контроль над версіями моделей, налаштуваннями швидкості мовлення, вибором голосу, форматом аудіофайлів та інтеграцією з іншими компонентами системи.

Особливе значення локальні TTS-рішення мають у складі локальних AI-агентів. Такий агент може приймати текстовий запит або генерувати відповідь за допомогою мовної моделі, після чого локальна TTS-система озвучує цю відповідь без звернення до хмарного сервісу. У результаті формується автономний голосовий інтерфейс, який може працювати в локальному середовищі. Наприклад, локальний AI-агент може використовуватися для озвучення відповідей у системі підтримки користувачів, навчальному застосунку, інформаційному кіоску, голосовому помічнику або системі доступності для людей з порушеннями зору.

Серед локальних TTS-рішень, які можуть бути використані для україномовного синтезу мовлення, доцільно розглянути Piper TTS, Coqui TTS з українською VITS-моделлю, StyleTTS2 Ukrainian, Supertonic 3 та robinhad/ukrainian-tts.

Piper TTS є легким локальним нейромережевим синтезатором мовлення, орієнтованим на швидку генерацію аудіо без використання хмарних сервісів. Його перевагою є невисокі вимоги до апаратних ресурсів, що робить Piper придатним для запуску на персональних комп'ютерах,

локальних серверах і малопотужних пристроях. Для української мови в офіційному списку голосів Piper зазначені голоси lada та ukrainian_tts для мовного коду uk_UA, при цьому ukrainian_tts поданий у якості medium. Це дозволяє розглядати Piper як один із найпростіших практичних варіантів локального україномовного TTS [19].

Coqui TTS з українською VITS-моделлю можна розглядати як приклад використання TTS-фреймворку для запуску конкретної нейромережевої архітектури. У цьому випадку Coqui TTS виступає програмним інструментом, який забезпечує запуск моделі, а VITS є архітектурою нейромережевого синтезу мовлення. Для української мови доступна модель tts_models/uk/mai/vits, навчена на наборі даних MAI з частотою дискретизації 22050 Гц. Таке рішення є корисним для бакалаврської роботи, оскільки дозволяє не лише реалізувати генерацію мовлення, а й пояснити принцип роботи сучасної нейромережевої TTS-архітектури [20].

StyleTTS2 Ukrainian є більш сучасним підходом до україномовного синтезу мовлення. Це рішення орієнтоване саме на український text-to-speech і дає змогу генерувати мовлення з українського тексту. На сторінці проєкту зазначено, що користувач може вводити український текст, додавати позначки наголосу та отримувати озвучення вибраним голосом. Крім того, у профілі розробника вказано, що StyleTTS2 навчався на українському багатоспікерному наборі даних. Таке рішення є перспективним для отримання природнішого звучання, однак його практичне розгортання може бути складнішим порівняно з Piper TTS або готовими легкими ONNX-рішеннями [21]. Supertonic 3 є сучасною багатомовною TTS-системою для локального запуску. Її важливою особливістю є використання ONNX Runtime, що дає змогу виконувати синтез мовлення без звернення до хмарних API. У документації Supertonic зазначено, що система працює повністю на пристрої користувача, не потребує хмарних викликів під час синтезу та підтримує 31 мову, зокрема українську з мовним кодом uk. Також підкреслюється, що модель орієнтована на локальний запуск на desktop, mobile, browser, edge-

пристроях і навіть Raspberry Pi. Завдяки цьому Supertonic 3 є дуже доречним рішенням для побудови локального AI-агента, якому потрібна автономна генерація голосу [22]. Спеціалізованим україномовним TTS-рішенням на основі ESPNET є robinhad/ukrainian-tts [23]. На відміну від багатомовних систем, цей проєкт безпосередньо орієнтований на синтез українського мовлення. Серед його переваг зазначені повністю офлайн-режим роботи, наявність кількох голосів, автоматичне визначення наголосів, керування швидкістю мовлення та підтримка роботи на Windows, macOS і Linux. Це робить robinhad/ukrainian-tts особливо корисним для практичної частини роботи, де потрібно показати саме україномовне локальне рішення, а не лише загальну багатомовну модель TTS. Отже, локальні TTS-рішення мають значні переваги для задач генерації аудіо з текстових даних. Вони забезпечують конфіденційність, автономність, незалежність від хмарних сервісів і можливість інтеграції з локальними AI-агентами. Для практичної реалізації найбільш простими варіантами є Piper TTS та robinhad/ukrainian-tts, оскільки вони мають готову підтримку української мови й можуть працювати локально. Coqui TTS з українською VITS-моделлю доцільно використовувати для демонстрації нейромережевого підходу до синтезу мовлення. StyleTTS2 Ukrainian можна розглядати як перспективну модель для якіснішого україномовного синтезу, а Supertonic 3 як сучасне локальне багатомовне рішення, придатне для інтеграції в автономні голосові AI-системи.

1.4 Системи клонування голосу

Системи клонування голосу є одним із сучасних напрямів розвитку технологій синтезу мовлення на основі AI (рис. 1.2). Їх основне призначення полягає у створенні штучного голосу, який максимально наближений до голосу конкретної людини. На відміну від звичайних TTS-систем, які використовують заздалегідь підготовлені стандартні голоси, технології

клонування голосу дають змогу відтворювати індивідуальні особливості мовлення: тембр, інтонацію, швидкість, емоційне забарвлення та характерну манеру вимови.

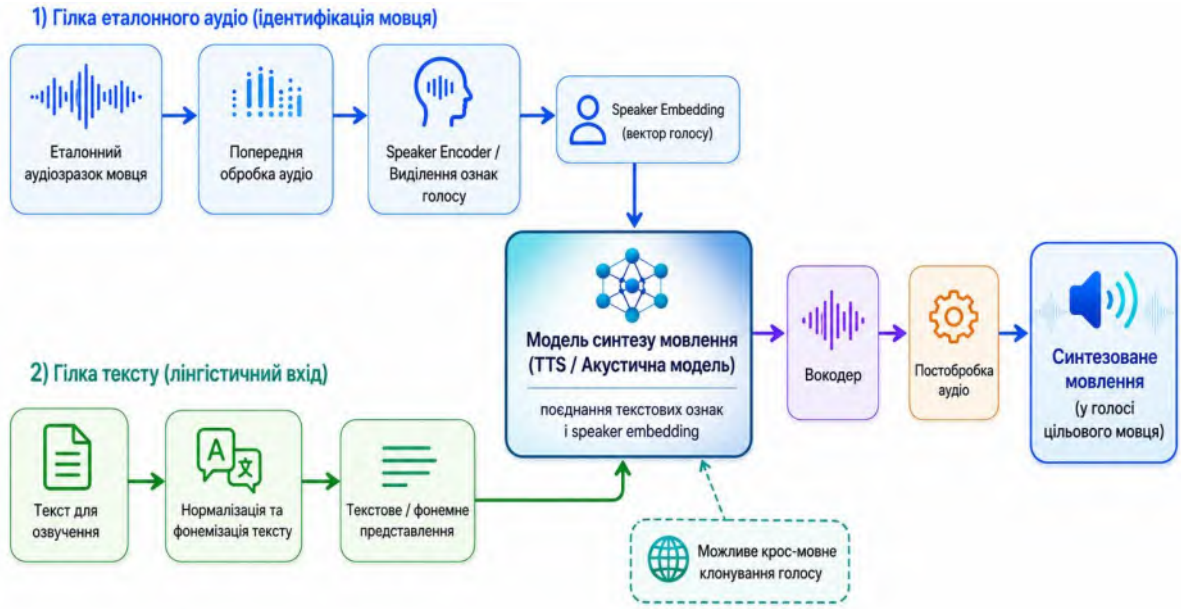

Рисунок 1.2 – Архітектура системи клонування голосу

Принцип роботи таких систем зазвичай передбачає використання зразка мовлення людини. На основі короткого або тривалого аудіофрагмента модель аналізує голосові характеристики диктора та формує його голосове представлення. Після цього система може озвучувати новий текст голосом, подібним до голосу вихідного мовця. У сучасних рішеннях для цього застосовуються нейронні мережі, моделі глибокого навчання, механізми перетворення голосу та генеративні підходи.

Системи клонування голосу можуть використовуватися у сфері озвучення відео, аудіокниг, комп’ютерних ігор, віртуальних асистентів, локалізації контенту та створення персоналізованих голосових інтерфейсів. Їх застосування дає змогу скоротити час на запис мовлення, зменшити потребу в повторному залученні диктора та швидко створювати аудіоматеріали для різних цифрових продуктів. Особливо корисними такі системи є тоді, коли потрібно зберегти однаковий стиль мовлення, тембр і манеру подачі

інформації в багатьох аудіофрагментах. Також вони мають значення для людей, які втратили можливість говорити, оскільки дають змогу відновити або зберегти індивідуальний голос для подальшого використання в допоміжних комунікаційних системах. У цьому випадку клонований голос може виконувати не лише технічну, а й соціальну функцію, адже власний голос є важливою частиною особистості людини. Водночас технологія клонування голосу має певні ризики. Зокрема, її можна використати для синтезу підроблених аудіозаписів, шахрайства або поширення дезінформації. Тому застосування таких систем повинно здійснюватися з дотриманням етичних норм, прав людини на власний голос та вимог інформаційної безпеки. Важливими умовами є отримання згоди на використання голосових даних, обмеження несанкціонованого доступу до моделей і контроль за створеними аудіофайлами.

Отже, клонування голосу є перспективною, але водночас відповідальною технологією, яка потребує обережного та контрольованого використання.

РОЗДІЛ 2
РОЗРОБКА МОДЕЛІ ЛОКАЛЬНОЇ СИСТЕМИ ДЛЯ ГЕНЕРАЦІЇ АУДІО З ТЕКСТОВИХ ДАНИХ

2.1 Поняття та принцип роботи системи генерація аудіо з текст

Основною метою TTS-систем є створення аудіо, яке за звучанням максимально наближене до природного людського мовлення. Такі системи використовуються у голосових асистентах, навігаційних системах, навчальних застосунках, автоматизованих інформаційних сервісах, системах доступності для осіб з порушеннями зору, локальних AI-агентах та інших програмних рішеннях, де необхідно озвучувати текстову інформацію.

У загальному вигляді роботу TTS-системи можна поділити на кілька основних етапів: обробку тексту, мовне моделювання, генерацію акустичних ознак та формування аудіосигналу (рис. 2.1).

Рисунок 2.1 – Етапи роботи TTS

На першому етапі система отримує вхідний текст і виконує його попередню обробку. До цього процесу належать очищення тексту, розбиття його на речення, нормалізація чисел, дат, скорочень, абревіатур та спеціальних символів. Наприклад, запис «2026 р.» має бути правильно інтерпретований як «дві тисячі двадцять шостий рік», а скорочення «км» – як «кілометрів» або «кілометри» залежно від контексту. Для української мови важливими є правильне визначення наголосів, відмінків, граматичних форм і

вимови окремих слів. Після попередньої обробки текст перетворюється у форму, зручну для подальшого аналізу. У багатьох TTS-системах використовується фонемне представлення, коли слова подаються не лише як послідовність літер, а як послідовність звуків. Це дає змогу точніше моделювати вимову, особливо для мов зі складними правилами читання або змінним наголосом. На цьому етапі система фактично визначає, як саме повинен звучати текст, тобто які фонеми, паузи, інтонаційні акценти та ритмічні особливості мають бути використані у TTS.

Наступним етапом є мовне моделювання, під час якого система аналізує структуру тексту та прогнозує просодичні характеристики мовлення. До них належать інтонація, темп, тривалість звуків, паузи між словами, емоційне забарвлення та логічні наголоси. Саме цей етап значною мірою впливає на природність звучання синтезованого мовлення. Якщо система неправильно визначає місця пауз або інтонаційні підйоми й спади, мовлення може звучати механічно, неприродно або важко сприйматися слухачем.

Після цього виконується генерація акустичних ознак. У традиційних нейромережевих TTS-системах модель не завжди одразу створює готовий аудіосигнал. Спочатку вона може формувати проміжне акустичне представлення, наприклад мел-спектрограму. Мел-спектрограма є візуально-числовим описом звукового сигналу, у якому відображено зміну частотних характеристик мовлення в часі. Вона містить інформацію про те, з якою інтенсивністю та в якій часовій послідовності мають бути згенеровані звуки.

Останнім етапом є формування аудіосигналу, тобто перетворення акустичних ознак у звукову хвилю. Для цього використовуються спеціальні модулі, які називають вокодерами. Вокодер приймає акустичне представлення, наприклад мел-спектрограму (рис. 2.2), і генерує аудіофайл у вигляді мовленнєвого сигналу. У сучасних системах для цього часто застосовуються нейромережеві вокодери, які забезпечують вищу якість звучання порівняно з класичними методами. Від якості вокодера залежить чіткість, природність, відсутність шумів і загальне сприйняття

синтезованого голосу. Сучасні TTS-системи значною мірою базуються на методах глибокого навчання. На відміну від ранніх підходів, які використовували попередньо записані фрагменти мовлення або статистичні моделі, нейромережеві системи здатні гнучкіше моделювати зв'язок між текстом і мовленням. Вони краще передають інтонацію, ритм, плавність звучання та індивідуальні особливості голосу.

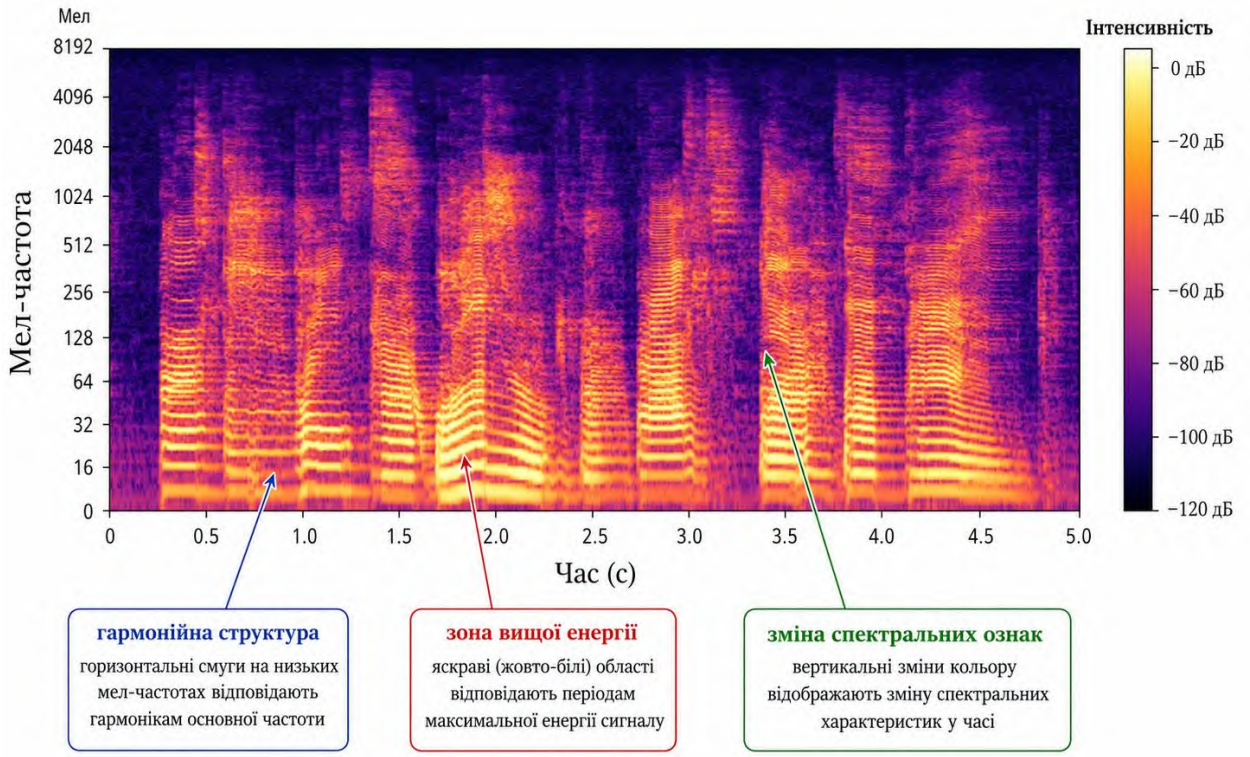

Рисунок 2.2 – Приклад мел-спектрограми

Завдяки цьому синтезоване мовлення поступово наближається до природного людського голосу. Одним із важливих напрямів розвитку TTS є використання трансформерних архітектур. Трансформери добре працюють із послідовностями даних і здатні враховувати контекст у межах усього речення або навіть більшого фрагмента тексту.

У TTS-системах такі архітектури можуть використовуватися для перетворення текстової послідовності в акустичні ознаки. Перевагою трансформерних моделей є здатність ефективно моделювати довгі залежності між словами, інтонаційними конструкціями та фонетичними особливостями.

Це дозволяє отримувати більш плавне та природне мовлення, особливо у складних реченнях.

Іншим важливим класом сучасних TTS-рішень є нейромережева архітектура для наскрізного синтезу мовлення з тексту, яка поєднує варіаційне моделювання, змагальне навчання та генерацію аудіосигналу в єдиній системі (Variational Inference with adversarial learning for end-to-end Text-to-Speech, VITS) – рис. 2.3. Різниця з підходами, де акустична модель і вокодер працюють як окремі незалежні блоки, в тому, що VITS-подібні рішення прагнуть об’єднати процес перетворення тексту в мовлення в одну узгоджену архітектуру. Такі моделі можуть забезпечувати достатньо природне звучання, підтримувати локальний запуск і використовуватися в практичних TTS-системах.

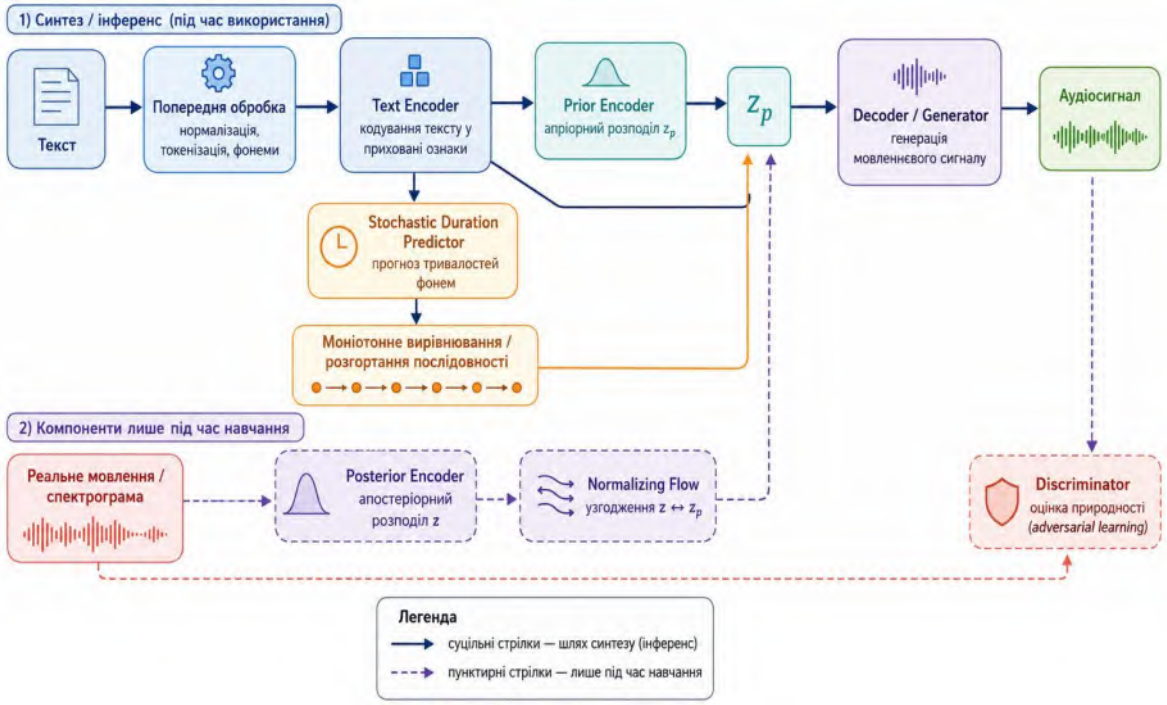

Рисунок 2.3 – Спрощена схема роботи VITS

Окрему групу становлять сучасні генеративні моделі синтезу мовлення. Вони орієнтовані на створення більш виразного, емоційного та індивідуалізованого голосу. Такі моделі можуть враховувати стиль мовлення, тембр, інтонацію, швидкість, паузи та інші характеристики (рис. 2.4). У

деяких системах також реалізується можливість вибору голосу, стилю або навіть клонування голосу за коротким аудіозразком.

Генеративні підходи є перспективними для створення природних голосових інтерфейсів, персоналізованих AI-асистентів, озвучення навчальних матеріалів, відео та інтерактивних застосунків. У контексті локальних AI-агентів TTS-система виконує роль вихідного мовленнєвого модуля. Наприклад, локальна мовна модель може сформувати текстову відповідь користувачу, після чого TTS-модель перетворює її на аудіо. Завдяки цьому AI-агент може не лише обробляти текстові запити, а й відповідати голосом. Якщо всі компоненти працюють локально, така система забезпечує автономність, незалежність від хмарних сервісів і вищий рівень конфіденційності.

Рисунок 2.4 – Спрощена схема генеративної моделі синтезу мовлення

Отже, TTS-система є складною багатокомпонентною технологією, яка поєднує обробку природної мови, фонетичний аналіз, нейромережеве моделювання та генерацію аудіосигналу. Її робота охоплює шлях від вхідного тексту до готового звукового файлу. Сучасні нейромережеві підходи, зокрема трансформерні архітектури, VITS-подібні рішення та генеративні моделі,

значно підвищили якість синтезу мовлення й зробили можливим створення природніших, гнучкіших та більш автономних TTS-систем.

2.2 Структура моделі

Локальна TTS-система призначена для перетворення текстових даних у мовленнєвий аудіосигнал без використання зовнішніх хмарних сервісів. На відміну від on-line-платформ синтезу мовлення, у локальному варіанті всі основні етапи обробки виконуються безпосередньо на локальному сервері або іншому обчислювальному пристрої. Такий підхід є особливо доцільним у випадках, коли потрібно забезпечити автономність роботи, конфіденційність текстових даних і можливість інтеграції TTS-модуля до локального AI-агента.

У загальному вигляді структура локальної TTS-системи складається з кількох основних компонентів: модуля введення тексту, блоку попередньої обробки, TTS-моделі, модуля генерації аудіосигналу, блоку збереження результату та компонента подальшого використання згенерованого аудіофайлу (рис. 2.5). Кожен із цих елементів виконує окрему функцію, але разом вони формують єдиний процес перетворення тексту в мовлення. Першим етапом роботи системи є отримання вхідного тексту. Джерелом текстових даних може бути користувацьке введення, текстовий файл, повідомлення з інформаційної системи, відповідь мовної моделі або результат роботи іншого програмного модуля. Наприклад, у локальному AI-агенті текст може формуватися автоматично як відповідь на запит користувача. Після цього цей текст передається до TTS-модуля для озвучення. Наступним етапом є попередня обробка тексту. Вона необхідна для того, щоб підготувати текст до коректного синтезу мовлення. На цьому етапі можуть виконуватися нормалізація символів, приведення чисел і дат до словесної форми, обробка скорочень, розбиття тексту на речення, токенізація та, за потреби, фонемізація. Для української мови особливе значення має правильна обробка

наголосів, відмінків, абревіатур і спеціальних позначень. Якість попередньої обробки безпосередньо впливає на природність звучання, оскільки помилки на цьому етапі можуть призвести до неправильної вимови окремих слів або неприродних пауз.

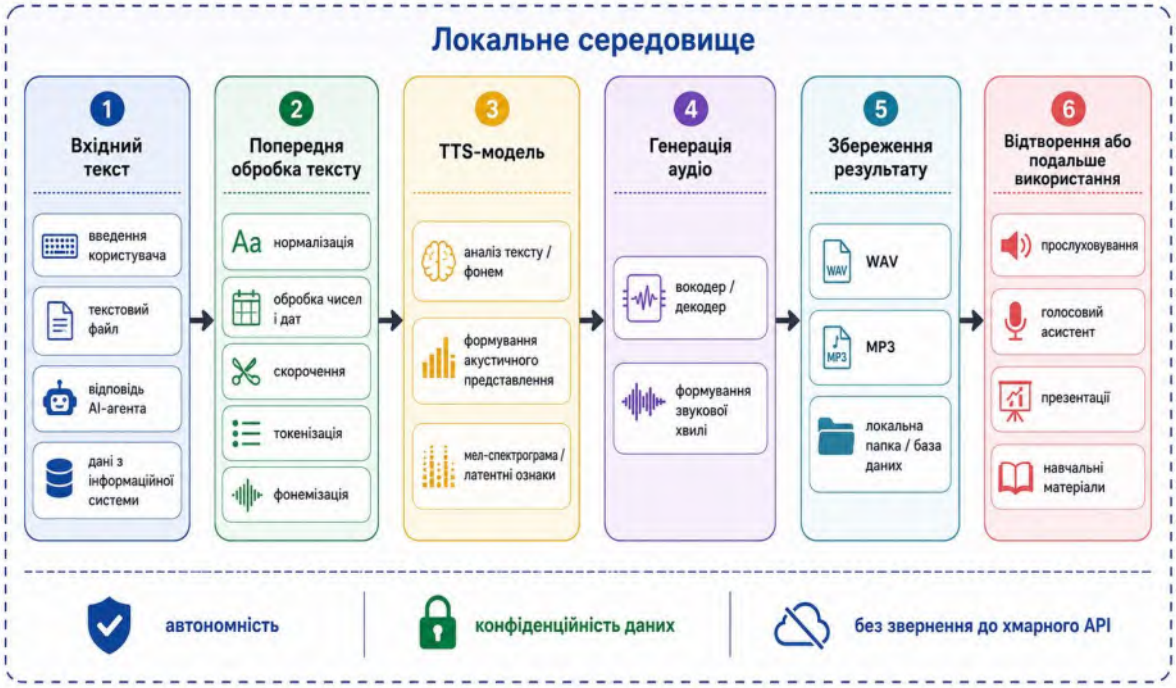

Рисунок 2.5 – Структура моделі локальної системи TTS

Після підготовки текст передається до TTS-моделі. Саме цей компонент є центральною частиною системи. TTS-модель аналізує текстове або фонемне представлення та формує акустичне представлення майбутнього мовлення. Залежно від обраної технології це може бути мел-спектрограма, латентні акустичні ознаки або інший внутрішній опис мовленнєвого сигналу. У сучасних локальних рішеннях для цього можуть використовуватися нейромережеві моделі, зокрема VITS-подібні архітектури, StyleTTS2, Piper TTS, Supertonic 3 або інші TTS-системи, здатні працювати без звернення до хмарного API. Далі виконується генерація аудіосигналу. На цьому етапі акустичні ознаки перетворюються у звукову хвилю. У багатьох TTS-системах цю функцію виконує вокодер або декодер, який формує готовий мовленнєвий сигнал. Результатом роботи є аудіодані, які можуть бути відтворені одразу або

збережені у файл. Зазвичай для збереження використовуються формати WAV або MP3. Формат WAV є зручним для подальшої технічної обробки та аналізу, оскільки зберігає звук без стиснення, тоді як MP3 доцільний у випадках, коли важливий менший розмір файлу.

Після генерації аудіо система виконує збереження результату. Згенерований файл може отримувати автоматичну назву, зберігатися у визначеній папці або передаватися до іншого модуля програми. Наприклад, у навчальному застосунку аудіофайл може зберігатися разом із відповідним текстом, у голосовому асистенті – відтворюватися одразу після генерації, а в системі підтримки користувачів – додаватися до історії звернення. Важливо, що при локальному підході і текст, і створений аудіофайл залишаються в межах локального середовища.

Останнім етапом є подальше використання згенерованого аудіофайлу. Він може бути прослуханий користувачем, доданий до відео, використаний у презентації, навчальному матеріалі, голосовому повідомленні або інтегрований у роботу локального AI-агента. У разі побудови голосового AI-асистента такий файл може відтворюватися автоматично як відповідь системи на запит користувача. Таким чином, TTS-модуль виконує роль вихідного голосового інтерфейсу, який перетворює текстову відповідь системи на природніше для сприйняття мовлення.

Загальну архітектуру локальної TTS-системи можна подати у вигляді такої послідовності: вхідний текст → попередня обробка → TTS-модель → генерація аудіо → збереження аудіофайлу → відтворення або подальше використання. Перевагою такої архітектури є її модульність. Кожен компонент системи може бути замінений або вдосконалений незалежно від інших. Наприклад, можна змінити TTS-модель, додати кращий модуль обробки українського тексту, реалізувати вибір голосу або змінити формат збереження аудіо. Це поступово розширює функціональність системи без повної перебудови її архітектури. Розглянута структура локальної системи TTS є зручною для практичної реалізації, оскільки може бути використана як

окремий інструмент для генерації аудіо або як складова частина локального AI-агента, що працює автономно та не передає дані до зовнішніх сервісів.

2.3 Програмне рішення Supertonic 3

SupertonicTTS можна розглядати як один із перспективних інструментів для побудови локальної системи синтезу мовлення, оскільки його основна ідея полягає у виконанні перетворення тексту в аудіо без звернення до хмарних сервісів. Згідно [24], SupertonicTTS описується як легка on-device TTS-система, що працює через ONNX Runtime і призначена для локального запуску на персональних комп’ютерах, мобільних пристроях, у браузері та на edge-пристроях [25]. Такий підхід відповідає концепції локальної TTS-системи, що є важливим для конфіденційності та автономності роботи програмного рішення.

У межах локальної TTS-системи SupertonicTTS може виконувати роль основного модуля генерації мовлення. На вхід до такого модуля подається попередньо підготовлений текст, після чого модель формує аудіосигнал і зберігає його у вигляді звукового файлу, наприклад WAV. У практичному сценарії це дозволяє використовувати SupertonicTTS для озвучення повідомлень локального AI-агента, генерації голосових відповідей, створення навчальних або інформаційних аудіоматеріалів, а також для експериментів із багатомовним синтезом мовлення. Supertonic 3 підтримує 31 мову, серед яких у списку зазначена й українська мова, що робить його потенційно придатним для україномовних застосувань [26]. Незважаючи на значні технічні досягнення, більшість сучасних систем синтезу мови, як і раніше, вимагають великої кількості параметрів, масивних навчальних даних «мова-текст» та складного конвеєра. До його складу входить модуль перетворення графем у фонеми (G2P), вирівнювач тексту та мови або попередньо навчені моделі для отримання текстових і мовних характеристик. Ці фактори в сукупності

призводять до збільшення обчислювальних витрат під час навчання та виведення, а також створюють складні взаємозалежності між компонентами системи. Враховуючи ці проблеми, перспективним напрямом досліджень у галузі синтезу мовлення є розробка більш оптимізованого конвеєра, який знижує архітектурну складність та накладні витрати на навчання, зберігаючи при цьому конкурентоспроможну продуктивність. Підхід, запропонований у [24], спрямований на підвищення як ефективності, так і масштабованості відповідно до останніх тенденцій у LLM, які фокусуються на економічно ефективному навчанні та оптимізованому висновку [27, 28]. Для ефективного вирішення цих проблем за умови забезпечення конкурентоспроможної продуктивності пропонується нову систему синтезу мови, а саме SupertonicTTS. Ця система складається з трьох модулів: (1) автокодувальника мови, який кодує аудіо в безперервне латентне уявлення; тривалість синтезованої мови. Зокрема, автокодувальник мовлення навчається стискати аудіо в латентний простір і точно відновлювати його. Модуль перетворення тексту в латентне уявлення вчиться оцінювати умовний ймовірнісний шлях для генерації латентів з шуму на основі тексту та еталонної мови. На відміну від традиційних моделей тривалості на рівні фонем, предиктор тривалості оптимізовано для оцінки загальної тривалості мовлення. Під час виведення модуль перетворення тексту в латентне уявлення генерує латенти з прогнозованою тривалістю висловлювання на основі вхідного тексту та еталонного мовлення, які потім декодуються автокодувальником мови для отримання аудіо з частотою 44,1 кГц. Наш підхід заснований на останніх досягненнях у галузі генеративних моделей, зокрема на моделях прихованої дифузії (LDM) [29-32]. При цьому використовується кілька методів для значного підвищення масштабованості та ефективності. По-перше, проектується латентний простір з напрочуд низькою розмірністю і стискаємо латенти вздовж тимчасової осі перед передачею їх модулю перетворення тексту в латенти. Ця стратегія значно знижує обчислювальні витрати наступних процесів. По-друге, вводиться розширення пакета з поділом

контексту для прискорення збіжності функції втрат, пропонуючи переваги, аналогічні до збільшення розміру пакета, але з відносно меншими обчислювальними витратами. Запропоноване розширення пакета ефективно сприяє точному навчанню вирівнювання тексту та мови. По-третє, широко використовуються блоки ConvNeXt [33-35] у всіх модулях для забезпечення легковажної та ефективної архітектури. Крім того, спрощується конвеєр TTS, використовуючи механізми перехресної уваги для вирівнювання тексту та мови, аналогічні [34, 36], та використовуючи як вхідні дані необроблений текст на рівні символів. Таке проектне рішення усуває необхідність зовнішнього вирівнювача та моделі перетворення графем на фонеми (G2P), які часто стають вузькими місцями при масштабуванні на нові області даних або мови. Крім того, утримуємось від включення зовнішніх попередньо навчених моделей, тим самим зменшуючи архітектурні залежності та складність. SupertonicTTS побудований на основі латентної дифузії, яка продемонструвала передові результати у різних завданнях генерації [30, 31, 37]. Зокрема, навчання SupertonicTTS поділено на три етапи (рис. 2.6): спочатку навчається автокодувальник мови для відображення вхідного аудіо в низькорозмірний латентний простір та відновлення вихідного аудіо. Потім модуль перетворення тексту в латентний простір навчається генерувати латентні уявлення, які точно відображають характеристики мовлення як вхідного тексту, так і еталонної мови. Нарешті, оптимізується провісник тривалості з метою оцінки загальної тривалості промови з урахуванням вхідного тексту та еталонної промови. Для забезпечення швидкої обробки та архітектурної ефективності SupertonicTTS в основному використовує блоки ConvNeXt [38], які нещодавно показали конкурентоспроможні результати у генерації мови [34, 38, 39]. Весь конвеєр додатково оптимізований за рахунок виключення зовнішніх попередньо навчених моделей, вирівнювачів тексту та мовлення та модулів G2P.

Автокодувальник мови перетворює вхідний аудіосигнал у приховані вистави за допомогою прихованого кодувальника і відновлює аудіосигнал з

цих подань за допомогою прихованого декодера. Надалі використовуємо крейд-спектрограму як вхідні ознаки для прихованого кодувальника замість необробленого аудіосигналу. Наші попередні експерименти показують, що такий підхід прискорює збіжність функції втрат під час навчання порівняно з використанням необробленого аудіосигналу. Прихований простір спроектований таким чином, щоб бути безперервним та значно меншою розмірністю, ніж кількість каналів крейди-спектрограми.

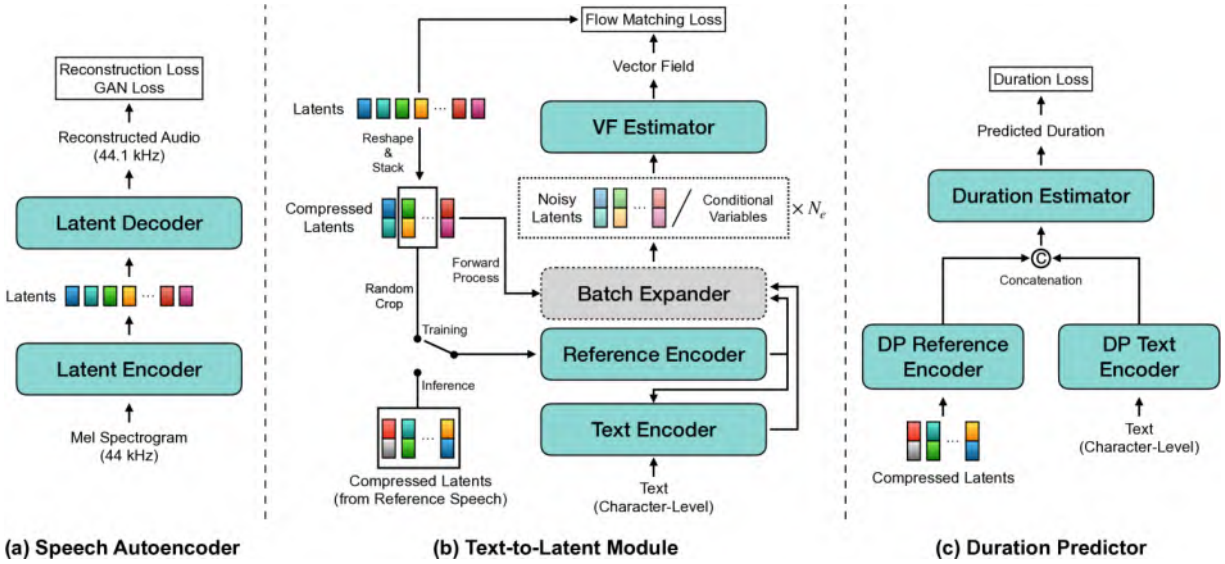

Рисунок 2.6 – Загальна архітектура SupertonicTTS [24]

Вхідно-вихідна структура автокодувальника мови відповідає структурі звичайних нейронних вокодерів. Тому автокодувальник мови можна інтерпретувати як нейронний вокодер із компактним прихованим простором у центрі. Латентний кодувальник побудований на основі архітектури Vocos, яка здебільшого складається з блоків ConvNeXt для підвищення обчислювальної ефективності [33, 34]. Щоб адаптувати архітектуру для латентного кодування, видаляємо вихідну головку Фур'є і вводимо лінійний шар безпосередньо перед остаточною нормалізацією. Цей лінійний шар проектує приховані уявлення у простір меншої розмірності. Хоча латентний кодувальник не використовується під час виведення TTS, його ефективна конструкція використовується для забезпечення швидкого латентного кодування протягом

усього навчання модуля перетворення тексту на латентне представлення. Аналогічно, латентний декодер слідує архітектурі Vocos з декількома ключовими модифікаціями. По-перше, адаптований шар пошарової згортки в блоках ConvNeXt, щоб він був причинним та розширеним. Введення причинних шарів дозволяє латентному декодер працювати в потоковому режимі. Крім того, замість використання головки Фур'є вводено два лінійні шари з активацією PReLU [40]. Підсумковий вихід у часовій області виходить шляхом згладжування вихідних даних на рівні кадрів. Цей підхід натхнен WaveNeXt [35], але використовується більш високу приховану розмірність з нелінійністю для підвищення репрезентативної здатності.

Аналогічно сучасним нейронним вокодерам [41, 42], автокодувальник мовлення навчається в рамках GAN [43], використовуючи комбінацію втрат реконструкції, втрат змагального типу та втрат зіставлення ознак. Втрати реконструкції визначаються як спектральна втрата, що обчислюється в галузі крейди-спектрограми. Для змагального навчання використовуються як багато-періодні дискримінатори (MPD) [42], так і багато-дозволені дискримінатори (MRD) [44] для підвищення якості сприйняття. Крім того, для мінімізації відстаней між дискримінаторними ознаками реальних та згенерованих зразків застосовується функція втрат зіставлення ознак, що додатково стабілізує процес навчання за допомогою змагальних моделей.

Модуль перетворення тексту в латентне уявлення генерує латентне уявлення, яке відображає основні характеристики мовлення як з вхідного тексту, так і з еталонної мови, дотримуючись структури LDM [30, 31, 37]. Шум поступово уточнюється до структурованого уявлення з допомогою залежить від часу векторного поля, створюваного структурою зіставлення потоків [45]. Модуль перетворення тексту в латентне уявлення оцінює це векторне поле на основі тексту, еталонної мови та часу, забезпечуючи, щоб кінцеве уявлення зберігало відповідні атрибути мови.

Модуль перетворення тексту на латентні сигнали не використовує зовнішні попередньо навчені моделі, модулі G2P або алгоритми

вирівнювання тексту на мову. Зокрема, він використовує текст на рівні символів як вхідні дані та застосовує механізми перехресної уваги для вирівнювання тексту та мови в рамках оптимізованої архітектури. Для ефективного навчання та збіжності використано дві методи: тимчасове стиснення латентних сигналів та розширення пакета з використанням контекстного обміну.

При цьому моделюються тимчасово стислі латенти, які перетворюються та розташовуються вздовж розмірності каналу. Цей має переваги порівняно з використанням вихідних латентних представлень. По-перше, він полегшує завдання вирівнювання тексту та мови через зменшення загальної кількості мовних кадрів. По-друге, це знижує обчислювальні витрати, що особливо вигідно для шарів, які покладаються на обчислювально-інтенсивні послідовні операції, такі як механізм уваги [46]. Нарешті, вся тимчасова інформація зберігається у перетвореному поданні, що дозволяє ідеально відновити дані, просто звернувши процес стиснення.

Далі використовується розширення пакета з поділом контексту підвищення ефективності навчання умовних генеративних моделей, заснованих на алгоритмах дифузії чи зіставлення потоків [45, 47, 48]. У традиційних схемах навчання ці генеративні алгоритми використовують вибірку шуму та крок дифузії для обурення вхідної змінної (прямий процес) та оптимізують моделі для придушення шуму за допомогою наданих умовних змінних (зворотний процес). На відміну від цього, пропоноване розширення пакета використовує вибірку декількох шумових екземплярів для отримання кількох обурених змінних вхідних на різних кроках дифузії. При цьому відповідні умовні змінні повторно використовуються шляхом дублювання 1. Ця стратегія імітує ефект збільшення розміру пакета, але обчислювально більш ефективною, оскільки умовні змінні використовуються спільно для кількох обурених вхідних даних (тобто поділ контексту).

Еталонний кодувальник приймає на вхід тимчасово стислі латенти. Ці латенти виходять шляхом обрізання частини навчальних даних під час

навчання та витягуються з еталонної мови під час виведення. Потім він обробляє латенти за допомогою декількох блоків ConvNeXt і генерує еталонні уявлення ключа та значення через два шари уваги, аналогічно блоку тембрових токенів NANSY++ [49]. Зверніть увагу, що еталонний ключ і значення є векторами фіксованого розміру, що не залежать від довжини вхідних даних.

Текстовий кодувальник обробляє вхідні дані на рівні символів за допомогою блоків ConvNeXt та шарів самоуваги. Ця архітектура розроблена для ефективного захоплення як локальних, і далеких залежностей у тексті, забезпечуючи обчислювальну ефективність. Вихідні дані із шарів самоуваги додатково уточнюються за допомогою векторів еталонного ключа та значення через два шари перехресної уваги, створюючи адаптивні до диктора текстові уявлення. Оцінка векторного поля в основному складається з блоків ConvNeXt, а також блоків тимчасової обробки, обробки тексту та обробки еталонного значення. Для підвищення виразності моделі деякі блоки ConvNeXt включають розширені згорткові шари. Тимчасова обробка досягається шляхом глобального додавання тимчасового вбудовування до вхідної послідовності. Як обробка тексту, і обробка еталонного значення використовують механізм перехресного уваги, де умовні змінні служать ключами і значеннями.

Модуль перетворення тексту на латенти оптимізується з використанням алгоритму зіставлення потоків [45]. Під час навчання як вхідні дані для еталонного кодувальника використовується випадково вирізаний сегмент стиснутих латентів. Щоб запобігти витоку інформації, застосовується маска до відповідного сегменту при обчисленні функції втрат зіставлення потоків, аналогічно попереднім роботам [36].

На етапі виведення запропонована структура вимагає синтезу загальної довжини прихованих уявлень. У цьому контексті очікується, що SupertonicTTS буде стійким до помилок в оцінці тривалості порівняно з іншими моделями TTS, які покладаються на інформацію про тривалість на

рівні фонем [50]. З урахуванням цього використано блок прогнозу тривалості лише на рівні висловлювання із простою, але ефективної архітектурою. Зокрема, текстове вбудовування на рівні висловлювання та еталонне вбудовування витягуються з текстового введення та еталонного мовлення відповідно з використанням блоків ConvNext та шарів уваги. Ці вбудовування поєднуються і перетворюються на скалярне значення, що представляє загальну тривалість мови, за допомогою лінійних шарів. Цей блок прогнозу тривалості навчається з використанням відстані між істинною та прогнозованою тривалістю. Для навчання автокодувальника мови були об'єднані загальнодоступні набори даних з нашою внутрішньою базою даних, в результаті чого розробники отримали загалом 11167 годин аудіозаписів приблизно від 14000 дикторів. Для навчання модуля перетворення тексту в латентну пам'ять та провісник тривалості вибиралися чотири англійські набори даних: LJSpeech, VCTK, Hi-Fi TTS і LibriTTS. У сукупності ці набори даних містять 945 годин високоякісного мовлення від 2576 англомовних. Усі аудіофайли були передискретизовані до цільової частоти дискретизації 44,1кГц, якщо їхня вихідна частота дискретизації відрізнялася. Далі було оптимізовано автокодувальник мови протягом 1,5 млн ітерацій, використовуючи оптимізатор AdamW зі швидкістю навчання 2×10^{-4} та розміром пакета 128.

Однією з головних переваг SupertonicTTS є орієнтація на швидке локальне виконання. Для бакалаврської роботи це важливо, оскільки локальна TTS-система повинна не лише генерувати якісне мовлення, а й бути достатньо простою для встановлення, тестування та демонстрації. Наявність Python SDK спрощує інтеграцію Supertonic TTS у програмний прототип: розробник може встановити бібліотеку, завантажити модельні файли та виконати синтез мовлення без створення складної серверної інфраструктури. Крім того, використання ONNX Runtime робить систему більш портативною, тобто потенційно придатною для запуску на різних апаратних платформах. Ще однією перевагою є компактність моделі. У матеріалах до Supertonic 3

зазначено, що публічні ONNX-активи мають близько 99 млн параметрів, що є значно меншим розміром порівняно з багатьма відкритими TTS-системами класу 0,7-2 млрд параметрів. Для локального застосування це має практичне значення, оскільки менша модель швидше завантажується, потребує менше пам’яті та краще підходить для комп’ютерів без потужного графічного процесора. Це особливо актуально для експериментальних систем, у яких важливо перевірити працездатність технології на доступному обладнанні.

У контексті конфіденційності Supertonic TTS також має суттєву перевагу. Оскільки синтез мовлення може виконуватися локально, текстові дані не потрібно передавати до сторонніх API. Це важливо у випадках, коли система працює з персональними повідомленнями, службовими документами, навчальними матеріалами або зверненнями користувачів. Локальне виконання зменшує залежність від інтернет-з’єднання, підвищує контроль над даними та дає змогу використовувати TTS-модуль у закритому середовищі, наприклад у локальній мережі організації або на окремому робочому комп’ютері.

2.4 Визначення особливостей Supertonic 3 від попередніх версій

Supertonic 3 відрізняється від попередніх версій насамперед ширшою мовною підтримкою, кращою стабільністю читання тексту та більшою придатністю до локального використання табл. 2.1. Supertonic 2 уже був орієнтований на локальний TTS: модель працювала через ONNX Runtime, не потребувала хмарного API та підтримувала 5 мов – англійську, корейську, іспанську, португальську та французьку. Також у матеріалах Supertonic 2 зазначалося, що модель має близько 66 млн параметрів і зберігає високу швидкість інференсу. Supertonic 3 є наступним етапом розвитку цієї системи. Основне покращення – розширення мовної підтримки з 5 до 31 мови. У списку підтримуваних мов Supertonic 3 зазначена українська мова.

Також Supertonic 3 має кращу стабільність озвучення тексту. В офіційному репозиторії зазначено, що порівняно із Supertonic 2 нова версія зменшує кількість помилок типу repeat failures і skip failures, тобто ситуацій, коли модель повторює фрагмент тексту або пропускає окремі слова чи частини речення.

Таблиця 2.1 – Порівняння Supertonic 3 з попередньою моделлю

Критерій	Supertonic 2	Supertonic 3
Кількість мов	5 мов: English, Korean, Spanish, Portuguese, French	31 мова, зокрема Ukrainian / українська
Стабільність синтезу	Можливі повтори, пропуски слів або нестабільне читання складних фрагментів	Зменшено repeat і skip failures, тобто повтори та пропуски під час озвучення
Схожість голосу	Базова якість для підтримуваних мов	Покращена speaker similarity для спільного набору мов
Локальний запуск	ONNX Runtime, локальний запуск, без хмарного API	Зберігає сумісний ONNX-інтерфейс, тому інтеграції з v2 легше перенести на v3
Розмір моделі	близько 66 млн параметрів	близько 99 млн параметрів у публічних ONNX-активах

Це особливо важливо для довших текстів, наприклад навчальних матеріалів, повідомлень AI-агента або автоматичного озвучення документів. Ще одна відмінність полягає в покращенні speaker similarity – схожості синтезованого мовлення з цільовим стилем або голосом у межах мов, які підтримувалися і раніше. Тобто Supertonic 3 має не лише більше мов, а й кращу якість відтворення голосових характеристик у порівнянні з Supertonic 2. З погляду локального використання Supertonic 3 залишається зручним для інтеграції, оскільки зберігає сумісний із Supertonic 2 публічний ONNX-інтерфейс. Це означає, що програмні рішення, створені для Supertonic 2, потенційно можна адаптувати до Supertonic 3 без повної переробки логіки інференсу. Водночас Supertonic TTS має й певні обмеження. По-перше, хоча підтримка української мови заявлена, якість вимови, інтонаційна природність, правильність наголосів і стабільність роботи з довгими україномовними текстами потребують окремого практичного тестування. Для бакалаврської роботи доцільно не лише вказати на підтримку української

мови, а й провести експериментальну перевірку: синтезувати декілька фрагментів тексту різної складності, оцінити чіткість вимови, природність звучання та наявність помилок у відтворенні слів. По-друге, локальне використання не означає повної відсутності технічних вимог. Навіть компактна TTS-модель потребує ОЗП, обчислювальних ресурсів і коректно налаштованого середовища. На слабких пристроях швидкість генерації може бути нижчою, а перший запуск може вимагати завантаження модельних файлів з Hugging Face. Тому SupertonicTTS доцільно розглядати не як універсальне готове рішення для всіх сценаріїв, а як зручний інструмент для локального та експериментального синтезу мовлення. По-третє, слід враховувати ліцензійні та функціональні обмеження. Код Supertonic поширюється окремо від модельних файлів, а модель Supertonic 3 на Hugging Face має власну ліцензію OpenRAIL. Це означає, що перед використанням у комерційних або публічних продуктах необхідно перевіряти умови ліцензування. Також не всі можливості, пов’язані з користувацькими голосами або стилями мовлення, можуть бути однаково доступними у відкритому варіанті. Supertonic 3 можна розглядати як сучасну легку TTS-модель, орієнтовану на локальний або on-device синтез мовлення. Модель підтримує використання голосових стилів, а створення власного стилю голосу можливе через Supertonic Voice Builder, який формує voice-style JSON або embedding на основі референсного аудіо. Водночас Supertonic 3 не слід описувати як повністю відкрите zero-shot рішення для клонування голосу, оскільки створення власного голосового стилю винесене в окремий інструмент Voice Builder.

РОЗДІЛ 3
РЕКОМЕНДАЦІЇ ЩОДО ПРАКТИЧНОЇ РЕАЛІЗАЦІЇ ЛОКАЛЬНОЇ СИСТЕМИ ДЛЯ ГЕНЕРАЦІЇ АУДІО З ТЕКСТОВИХ ДАНИХ

3.1 Експериментальна реалізація моделі у Google Colab

У практичній частині реалізовано приклад синтезу мовлення з текстових даних з використанням моделі Supertonic 3 (Додаток А). Реалізація виконувалася у середовищі Google Colab. Воно вдало підходить для експериментальної реалізації моделі TTS на основі Supertonic 3, оскільки воно дає змогу швидко розгорнути Python-середовище без попереднього налаштування локальної операційної системи. Це особливо зручно на етапі первинного тестування TTS-моделі, коли необхідно перевірити працездатність обраного рішення, якість синтезованого мовлення, підтримку української мови та можливість збереження результату у вигляді аудіофайлу.

На початковому етапі роботи було виконано підготовку програмного середовища. Для цього встановлювалася бібліотека Supertonic, яка містить необхідні засоби для роботи з TTS-моделлю. Після встановлення здійснювалося підключення основних програмних модулів, необхідних для синтезу мовлення, відтворення аудіо безпосередньо в середовищі Colab, збереження аудіофайлу та його подальшого завантаження на комп’ютер.

Після підготовки середовища створювався об’єкт TTS-системи. Під час його ініціалізації було використано параметр автоматичного завантаження модельних файлів. Це означає, що необхідні компоненти моделі завантажуються автоматично під час першого запуску. Такий підхід спрощує практичну реалізацію, оскільки користувачу не потрібно вручну шукати, завантажувати та підключати окремі файли моделі.

Наступним етапом було формування вхідного тексту, який необхідно перетворити на мовлення. У межах практичного прикладу було використано українськомовний текст, пов’язаний із тематикою роботи. Він описував

можливості систем синтезу мовлення на основі AI та їхнє значення для локальної обробки даних. Використання саме української мови є важливим для перевірки можливостей моделі для україномовного синтезу мовлення.

Після задання тексту було обрано голосовий стиль. У Supertonic 3 передбачено декілька варіантів голосів, що дозволяє змінювати характер звучання синтезованого мовлення. У реалізованому прикладі було використано один із жіночих голосових стилів. Це дало змогу оцінити якість генерації мовлення та природність звучання на прикладі конкретного голосового профілю.

Основним етапом практичної реалізації став процес синтезу мовлення. На цьому етапі вхідний текст передавався до TTS-моделі, після чого модель формувала аудіосигнал. Для коректної генерації було вказано, що синтез виконується українською мовою. Також було задано параметри, які впливають на швидкість та якість синтезу.

Одним із важливих параметрів є кількість кроків генерації. Вона визначає, наскільки детально модель формує аудіосигнал. Зменшення кількості кроків може прискорити синтез, однак у деяких випадках це може впливати на якість звучання. У практичній реалізації було обрано збалансоване значення, яке дозволяє отримати прийнятну якість аудіо без надмірного збільшення часу обробки.

Також у програмі було задано швидкість мовлення. Значення швидкості відповідало нормальному темпу, тобто синтезоване мовлення не було штучно прискорене або сповільнене. Це важливо для отримання природного звучання, яке можна використовувати для демонстрації роботи TTS-системи.

Оскільки вхідний текст може бути досить довгим, у реалізації було передбачено його поділ на окремі фрагменти. Це дозволяє стабільніше обробляти великі тексти та зменшує ймовірність помилок під час генерації аудіо. Між окремими фрагментами синтезованого мовлення було додано короткі паузи. Завдяки цьому звучання стає більш природним, а перехід між частинами тексту – менш різким. Після завершення синтезу отриманий

аудіосигнал було збережено у файл формату WAV. Цей формат є зручним для практичної роботи, оскільки широко підтримується різними програмами для відтворення та обробки аудіо. Крім того, WAV дозволяє зберігати звук без втрати якості, що є важливим для подальшого аналізу результатів синтезу.

Для перевірки результату було виконано зчитування збереженого аудіофайлу та визначення його реальної тривалості. Такий підхід дозволяє переконатися, що файл було сформовано коректно, а також отримати кількісну характеристику результату роботи програми. У результаті виконання практичного прикладу було сформовано аудіофайл «long_text_uk.wav» тривалістю приблизно 20,92 с.

Додатково в середовищі Google Colab було реалізовано можливість прослуховування згенерованого аудіо безпосередньо в ноутбуці. Це є зручним для швидкої перевірки якості синтезу без необхідності відкривати файл у сторонніх програмах. Після прослуховування результат можна завантажити на комп’ютер користувача для подальшого використання, збереження або аналізу. Таким чином, розроблений програмний приклад демонструє повний цикл роботи TTS-системи: підготовку середовища, ініціалізацію моделі Supertonic 3, задання вхідного тексту, вибір голосового стилю, синтез мовлення українською мовою, збереження результату у WAV-файл (рис. 3.1), перевірку тривалості аудіо, прослуховування та завантаження готового файлу.

Отримана реалізація може розглядатися як базовий приклад практичного застосування технологій штучного інтелекту для генерації аудіо з текстових даних. Запропонована реалізація підтверджує можливість використання Supertonic 3 як інструменту для побудови локальної або експериментальної TTS-системи. Вона демонструє, що сучасні моделі синтезу мовлення можуть бути застосовані для автоматичного перетворення тексту на аудіо, створення голосових повідомлень, озвучення інформаційних матеріалів та подальшої інтеграції в AI-агенти або інші програмні системи. Для забезпечення відтворюваності експериментального середовища локальну TTS-систему на базі Supertonic 3 доцільно запускати у Docker-контейнері.

Такий підхід дозволяє ізолювати залежності Python, ONNX Runtime та допоміжні бібліотеки від основної ОС.

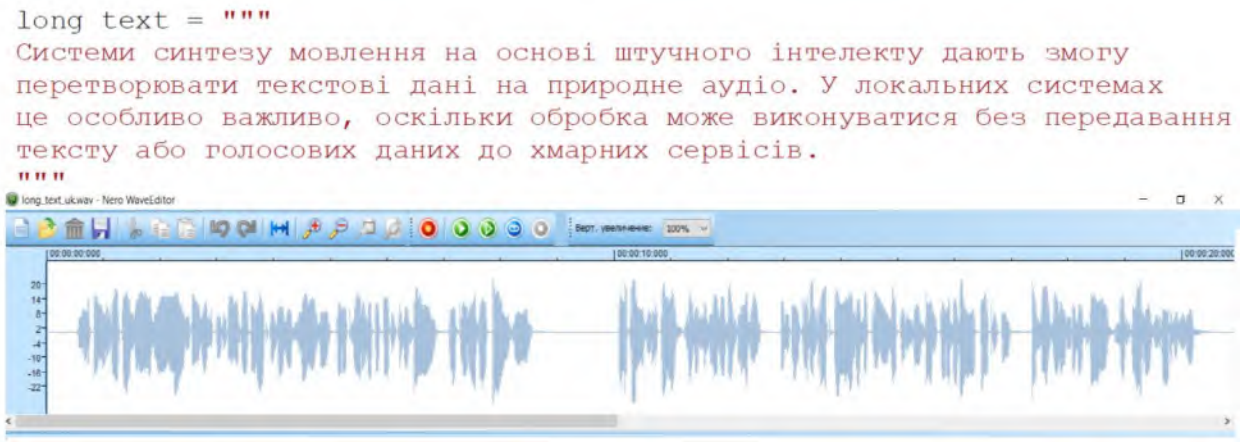

Рисунок 3.1 – Текст і WAV-файл «long_text_uk.wav»

Після першого завантаження модельні файли можуть зберігатися у Docker volume, що забезпечує повторний запуск системи без необхідності повторного завантаження моделі. У результаті формується портативне середовище, що придатне для експериментальної перевірки TTS-рішення.

Docker забезпечує ізоляцію середовища виконання, що є важливою перевагою під час експериментальної та практичної реалізації TTS-системи. У контейнері можна зафіксувати необхідну версію Python, встановити потрібні бібліотеки, залежності для роботи з аудіофайлами та саму TTS-модель. Завдяки цьому система стає більш відтворюваною: її можна запустити на різних комп’ютерах без необхідності вручну налаштовувати кожну залежність окремо. Це особливо важливо для бакалаврської роботи, оскільки дає змогу показати не лише факт запуску моделі, а й структурований підхід до побудови програмного середовища.

У режимі локального HTTP-сервера TTS-модель не запускається як одноразовий скрипт, що просто генерує аудіофайл, а працює постійно та очікує вхідні запити. Користувач або інша програма передає на сервер текст, параметри мови, голосу та швидкості мовлення, після чого сервер виконує синтез і повертає результат у вигляді аудіоданих або збереженого WAV-файлу.

Така архітектура є зручною для подальшої інтеграції, оскільки будь-який застосунок, який уміє надсилати HTTP-запити, може використовувати локальний TTS-сервіс без прямої роботи з внутрішнім кодом моделі.

Використання HTTP-сервера також дає змогу логічно розділити систему на окремі функціональні частини. Наприклад, один модуль може відповідати за отримання або формування тексту, інший — за його попередню обробку, а Docker-контейнер із TTS-сервером — лише за генерацію мовлення. Такий підхід відповідає принципам модульної архітектури, де кожен компонент виконує свою окрему задачу. У майбутньому це дозволяє легше замінити модель, змінити інтерфейс користувача або додати нові функції без повної перебудови всієї системи.

Важливою перевагою Docker-рішення є можливість локального виконання синтезу мовлення. Після встановлення програмних залежностей і завантаження необхідних модельних файлів сам процес генерації аудіо може виконуватися без використання хмарних API. Це є суттєвим фактором для систем, де потрібно зберігати контроль над текстовими даними та результатами синтезу. Наприклад, у задачах озвучування внутрішніх документів, повідомлень користувачів або службових текстів локальна обробка зменшує ризики передавання інформації стороннім сервісам.

Перший запуск системи може потребувати доступу до Інтернет, оскільки модельні файли та залежності зазвичай завантажуються з віддалених репозиторіїв. Однак після початкового налаштування їх можна зберігати у кеші або окремому томі Docker. Завдяки цьому повторні запуски контейнера стають швидшими й не потребують повторного завантаження всіх ресурсів. Такий механізм підвищує стабільність роботи системи та робить її зручнішою для багаторазового експериментального використання.

У практичній реалізації локальний HTTP-сервер на базі Docker може використовуватися як проміжний шар між користувачем і моделлю синтезу мовлення. Наприклад, користувач вводить текст у вебінтерфейсі, після чого цей текст передається на локальний сервер, де виконується синтез.

Згенерований аудіофайл повертається назад і може бути прослуханий, збережений або використаний в іншому програмному процесі. Така схема наближена до реальних умов застосування TTS-систем, оскільки модель не ізольована від інших компонентів, а працює як сервіс, готовий до інтеграції.

Окремо слід зазначити, що Docker як локальний HTTP-сервер є зручним рішенням для побудови локальних AI-агентів. У такій системі мовна модель або інший інтелектуальний компонент може формувати текстову відповідь, а TTS-сервер перетворює її на голосове повідомлення. Це дозволяє реалізувати голосовий інтерфейс для локального асистента без обов'язкового використання зовнішніх сервісів синтезу мовлення. У результаті формується більш автономна система, у якій основні етапи обробки даних виконуються в межах локального середовища.

До переваг такого підходу можна віднести відтворюваність середовища, ізоляцію залежностей, можливість локального запуску, зручність інтеграції через HTTP-запити та придатність для подальшого масштабування. Водночас існують і певні обмеження. Користувач повинен мати базові навички роботи з Docker, правильно налаштувати порти, каталоги для збереження результатів і кеш модельних файлів. Крім того, продуктивність системи залежить від характеристик комп'ютера, на якому виконується контейнер, зокрема від потужності процесора, обсягу оперативної пам'яті та наявності апаратного прискорення.

Таким чином, використання Docker у режимі локального HTTP-сервера є доцільним способом практичної реалізації TTS-системи. На відміну від запуску окремого скрипта, серверний підхід дозволяє перетворити модель синтезу мовлення на повноцінний локальний сервіс, який може взаємодіяти з іншими програмними компонентами. Для бакалаврської роботи це рішення є показовим, оскільки демонструє не лише запуск моделі Supertonic 3, а й можливість її використання в архітектурі локальної програмної системи або AI-агента.

3.2 Клонування голосу за допомогою Supertonic Voice Builder

За результатами експериментальної перевірки встановлено, що Supertonic 3 коректно працює з українським текстом і містить набір вбудованих україномовних голосів, зокрема п’ять чоловічих та п’ять жіночих дикторів. Це дає підстави розглядати Supertonic 3 як практично придатний інструмент для реалізації україномовної TTS-системи. Наявність кількох голосових профілів дозволяє генерувати мовлення з різними тембральними характеристиками без необхідності додаткового навчання моделі. Водночас використання Supertonic 3 для клонування конкретного голосу потребує застосування Supertonic Voice Builder, який створює окремий voice-style JSON на основі еталонного аудіозразка мовця [51].

Supertonic Voice Builder – це додатковий інструмент екосистеми Supertonic, призначений для створення власного голосового стилю на основі еталонного аудіозразка (рис. 3.2).

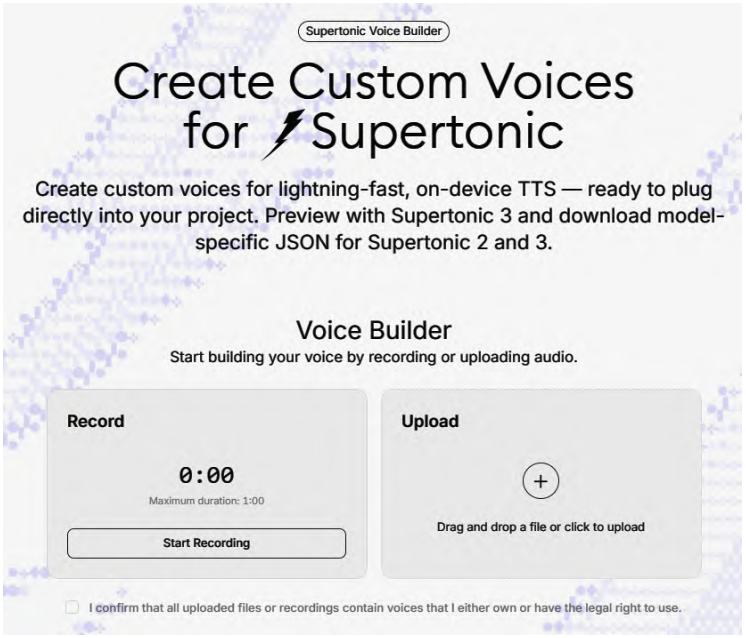

Рисунок 3.2 – Supertonic Voice Builder [51]

Якщо базова модель Supertonic 3 може працювати з готовими фіксованими голосами, то Voice Builder дає можливість сформувати окремий

голосовий профіль, який надалі використовується під час синтезу мовлення. На офіційній сторінці Supertonic 3 зазначено, що для створення власного voice-style JSON із reference audio використовується саме Supertonic Voice Builder, а отримані стилі можуть містити embeddings для Supertonic 2 і Supertonic 3. Принцип роботи Supertonic Voice Builder полягає в тому, що користувач записує або завантажує короткий аудіофрагмент із голосом цільового мовця. Цей аудіозразок використовується як еталон для визначення індивідуальних характеристик голосу: тембру, загального звучання, особливостей подачі та манери мовлення. У вебінтерфейсі Voice Builder передбачено можливість записати голос безпосередньо або завантажити готовий аудіофайл; тривалість запису обмежується однією хвилиною. Після цього система формує голосовий стиль, який можна попередньо прослухати за допомогою Supertonic 3 та експортувати у форматі JSON [52].

Отриманий JSON-файл фактично виконує роль голосового профілю. Він не є звичайним аудіозаписом, а містить параметризоване представлення голосу, яке модель Supertonic 3 може використовувати під час генерації мовлення. Завдяки цьому під час синтезу тексту модель отримує не лише текстовий вхід, а й інформацію про бажаний голосовий стиль. У результаті згенероване мовлення має відтворювати не стандартний голос із набору пресетів, а голосові характеристики, наближені до еталонного мовця. У практичній реалізації робота з Supertonic Voice Builder може складатися з кількох етапів. Спочатку готується якісний аудіозразок диктора: бажано без фонового шуму, музики, сторонніх голосів і сильного реверберaційного ефекту. Далі цей аудіозразок завантажується у Voice Builder або записується через мікрофон. Після обробки система створює voice-style JSON, який надалі підключається до Supertonic 3 як користувацький голосовий стиль. На наступному етапі у модель подається текст для озвучення, а модель генерує аудіофайл, використовуючи створений голосовий профіль.

Важливо зазначити, що Supertonic Voice Builder не є повністю локальним модулем у тому самому сенсі, що й запуск Supertonic 3 на

власному комп'ютері. Якщо сама модель Supertonic 3 орієнтована на локальний або on-device синтез і може працювати без хмарного API, то створення власного голосового стилю виконується через окремий сервіс Voice Builder. В офіційному репозиторії Supertonic вказано, що open-weight пакет зосереджений на локальному TTS із фіксованими голосами й не містить офіційного voice-cloning pipeline; для перенесення власного голосу в локальне розгортання пропонується використовувати Voice Builder, який перетворює короткий reference recording у JSON-файли для Supertonic 3.

3.3 Порівняння Supertonic 3 з іншими моделями

TTS Supertonic 3 залишається в межах конкурентного діапазону WER/CER порівняно з більшими відкритими моделями TTS, такими як VoxCPM2 (рис. 3.3), зберігаючи водночас легкий шлях розгортання на пристрої (мови, що позначені «*», використовують CER, а інші – WER).

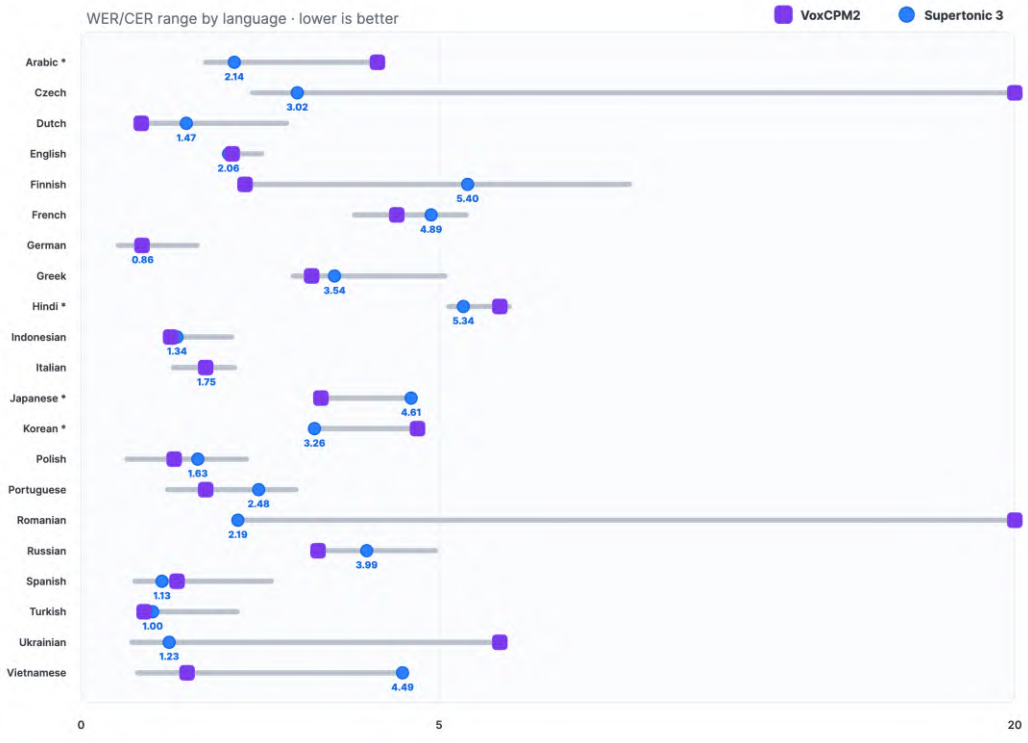

Рисунок 3.3 – Діапазон WER/CER за мовами (менше значення – краще) [52]

У табл. 3.1 наведено результати порівняння продуктивності Supertonic з сучасними TTS без попереднього навчання. Вданому випадку, розглядались базові моделі: VALL-E [53], VoiceBox [54], Mega-TTS 2 [55], CLaM-TTS [56] та DiTTo-TTS [36].

Таблиця 3.1 – Продуктивність моделей TTS

Модель	WER ↓	CER ↓	SIM ↑	Час виводу, с ↓	Датасет, год	Кіль-ть параметрів
GT	2,18	0,60	0,6770	-	-	-
VALL-E	5,9	-	-	~6,4	60,000	410M
VoiceBox	1,9	-	0,662	~6,2	60,000	371M
Mega-TTS 2	2,46	-	0,898	3,02 (V100)	60,000	473M
CLaM-TTS	5,11	2,87	0,495	4,15 (A100)	55,000	>1.3B
DiTTo-TTS	2,56	0,89	0,627	1,62 (A100)	55,000	970M
Supertonic	2,64	0,83	0,472	0,23 (RTX 4090)	915	44M

Ці моделі демонструють гарні результати синтезу мови навіть у сценаріях із нульовою кількістю прикладів. Для оцінки Supertonic використана схема: для кожного висловлювання було згенеровано 5 зразків з тестової підмножини LibriSpeech, використовуючи 3-секундну еталонну промову з іншого висловлювання того ж диктора, з 32 кроками виводу і CER, щоб оцінити більш тонкі деталі вимови у синтезованій мові. Для порівняння подібності дикторів з Mega-TTS 2 також використаний WavLM Base+6. Час обробки виміряно шляхом усереднення часу обробки 100 поколінь 10-секундного мовлення, виконаного на RTX 4090.

За показником CER модель Supertonic досягла результату 0,83 – перевищивши DiTTo-TTS (0,89). Тобто, Supertonic може генерувати мову з мінімальною кількістю пропусків, повторень або помилок вимови. Що стосується подібності мовців, виміряної за допомогою WavLM-TDNN, SupertonicTTS набрав 0,472, що нижче, ніж у VoiceBox (0,662), CLaM-TTS (0,495) та DiTTo-TTS (0,627). Ця різниця пов'язана з тим, щщо Supertonic не використовує текстову інформацію як еталон. Це потенційно впливає на продуктивність при нульовому навчанні. Однак Supertonic досягла 0,915 при оцінці з WavLM Base+, перевершивши Mega-TTS 2 (теж не використовує

текст як еталонну мову). Цих результатів було досягнуто при значно меншій кількісті параметрів (рис. 3.4) і розміру набору даних TTS менше 2 % порівняно з базовими моделями. Нарешті, Supertonic продемонструвала гарну швидкість виводу – всього 0,23 секунд для генерації 10-секундного зразка мови. Цей результат значно швидше, ніж у DiTTo-TTS, (1,62 секунди). Загалом Supertonic демонструє конкурентоспроможну продуктивність з ефективністю в архітектурі, та у навчанні. Supertonic 3 працює швидко на процесорі, навіть порівняно з більшими базовими моделями, виміряними на GPU A100 [57], та використовує значно менше пам'яті (рис. 3.5).

Рисунок 3.4 – Розміри моделей TTS

Рисунок 3.5 – RTF та пікове використання пам'яті на вибраному пристрої (CPU і GPU)

Модель не потребує GPU, що значно спрощує локальне, браузерне та периферійне розгортання. Порівняно з Piper TTS, Supertonic 3 є цікавим саме як об’єкт дослідження, оскільки це новіше багатомовне рішення, орієнтоване на локальний інференс і сучасні сценарії використання. Вибір зумовлений тим, що Supertonic 3 поєднує підтримку локального запуску, багатомовність, наявність української мови, порівняно сучасну архітектуру та орієнтацію на стабільну генерацію мовлення (табл. 3.2).

Таблиця 3.2 – Узагальнене порівняння Piper TTS і Supertonic 3

Критерій	Piper TTS	Supertonic 3
Хмарний API	Не є обов’язковим	Не потрібний для синтезу
Підтримка української мови	Є українські голоси, але якість залежить від конкретної моделі	Українська входить до переліку підтримуваних мов
Стабільність довгого тексту	Залежить від голосу та фонемізації	Заявлено зменшення повторів і пропусків тексту
Експериментальна цінність	Добре підходить як базове локальне рішення	Більш доцільна для дослідження багатомовного TTS

Piper TTS можна вважати перевіреним і практичним інструментом, але його якість значною мірою залежить від конкретної голосової моделі, її рівня якості та особливостей фонемізації. Для української мови це може бути додатковим обмеженням, оскільки у відкритих обговореннях Piper раніше порушувалася проблема некоректної вимови в українському голосі Lada, пов’язана з особливостями eSpeak-ng. Це не означає, що Piper непридатний для української мови, однак свідчить про необхідність додаткової перевірки вимови та якості конкретних голосів.

3.4 Економічне обґрунтування результатів

Економічне обґрунтування результатів роботи полягає в оцінюванні доцільності практичного використання локальної системи для генерації аудіо з текстових даних. У межах цієї роботи розглядається варіант, коли система

розгортається на вже наявному комп'ютері або локальному сервері. Тому витрати на придбання апаратної частини в розрахунках не враховуються. Основними витратами в такому випадку є витрати на налаштування ПЗ, підготовку TTS-моделі до роботи, тестування, супровід системи та споживання електроенергії. Запропонована локальна система має економічну перевагу порівняно з використанням хмарних TTS-сервісів, оскільки після початкового налаштування вона не потребує постійної оплати за кожну хвилину синтезованого мовлення. Для проведення орієнтовного економічного розрахунку приймемо кілька вихідних даних (табл. 3.3).

Таблиця 3.3 – Прийняті вихідні дані

Показник	Позначення	Значення
Трудомісткість налаштування локальної TTS-системи	$T_{нал}$	30 год
Вартість однієї години роботи фахівця	$S_{год}$	200 грн/год
Вартість придбання комп'ютера або сервера	$C_{обл}$	0 грн
Витрати на ПЗ	$C_{ПЗ}$	0 грн
Середня додаткова потужність під час роботи системи	P	0,08 кВт
Орієнтовний час роботи системи на місяць	t	50 год/міс
Тариф на електроенергію	$C_{кВт}$	5 грн/кВт·год
Час супроводу системи на місяць	$T_{супр}$	2 год/міс
Плановий обсяг генерації аудіо	$V_{міс}$	1000 хв/міс
Умовна вартість хмарної генерації аудіо	$C_{тариф}$	2,50 грн/хв

Вартість налаштування локальної TTS-системи визначається за формулою:

$$C_{нал} = T_{нал} \cdot S_{год} = 30 \cdot 200 = 6000 \text{ грн}, \quad (3.1)$$

де $C_{нал}$ – вартість налаштування системи.

Початкові витрати на впровадження системи можна визначити за виразом:

$$K_0 = C_{нал} + C_{обл} + C_{пз} = 6000 \text{ грн}, \quad (3.2)$$

де K_0 – початкові витрати на впровадження системи.

Витрати на електроенергію під час експлуатації дорівнюють:

$$C_{ел} = P \cdot t \cdot C_{кВт} = 0,08 \cdot 50 \cdot 5 = 20 \text{ грн/міс}, \quad (3.3)$$

де $C_{ел}$ – витрати на електроенергію.

Витрати на щомісячний супровід системи визначаються за формулою:

$$C_{супр} = T_{супр} \cdot S_{год} = 2 \cdot 200 = 400 \text{ грн/міс}, \quad (3.4)$$

де $C_{супр}$ – витрати на супровід системи.

Загальні щомісячні витрати на експлуатацію локальної системи визначаються за формулою:

$$C_{лок} = C_{ел} + C_{супр} = 20 + 400 = 420 \text{ грн/міс} \quad (3.5)$$

де $C_{лок}$ – щомісячні витрати на роботу локальної TTS-системи.

Собівартість генерації однієї хвилини аудіо визначається за формулою:

$$C_{1хв} = \frac{C_{лок}}{V_{міс}} = \frac{420}{1000} = 0,42 \text{ грн/хв}, \quad (3.6)$$

де $C_{1хв}$ – собівартість однієї хвилини синтезованого аудіо.

Для порівняння розрахуємо умовні витрати на використання хмарного TTS-сервісу:

$$C_{хмара} = V_{міс} \cdot C_{тариф} = 1000 \cdot 2,5 = 2500 \text{ грн/міс}, \quad (3.7)$$

де $C_{хмара}$ – витрати на хмарну генерацію аудіо.

Економія від використання локальної системи порівняно з хмарним сервісом визначається за формулою:

$$E_{міс} = C_{хмара} - C_{лок} = 2500 - 420 = 2080 \text{ грн/міс}, \quad (3.8)$$

де $E_{міс}$ – щомісячний економічний ефект.

Строк окупності впровадження локальної системи визначається за формулою:

$$K_{ок} = \frac{K_0}{E_{міс}} = \frac{6000}{2080} \approx 2,88 \text{ міс}. \quad (3.9)$$

Отже, за прийнятих умов локальна система може окупитися приблизно за 3 місяці. Проведені розрахунки показують, що використання локальної системи для генерації аудіо з текстових даних є економічно доцільним за

умови регулярного застосування. Оскільки комп’ютер або сервер уже наявний, основні початкові витрати пов’язані не з придбанням обладнання, а з налаштуванням програмного середовища та перевіркою працездатності системи. Це суттєво зменшує строк окупності рішення. Крім економії коштів, локальна система має додаткові практичні переваги: вона зменшує залежність від зовнішніх сервісів, забезпечує контроль над текстовими даними, може працювати у внутрішній мережі організації та є зручною для інтеграції в локальні AI-агенти.

ВИСНОВКИ

Локальні TTS є мають переваги: автономність, конфіденційність, незалежність від хмарних сервісів і можливість роботи без постійного доступу до Інтернет. Якість озвучення залежить не лише від самої моделі, а й від правильної обробки тексту, наголосів, чисел, скорочень і власних назв. Тому для вибору TTS-рішення необхідно враховувати природність звучання, стабільність вимови та можливість інтеграції в майбутню систему.

Підтримка української мови є важливою умовою практичного використання TTS-систем. Для україномовного синтезу можуть використовуватися такі рішення, як Piper TTS, Coqui TTS, StyleTTS2 Ukrainian, Supertonic 3 та robinhad/ukrainian-tts. Найбільш перспективними є ті системи, які поєднують простоту локального запуску, підтримку української мови та придатність до інтеграції в AI-агента.

Робота TTS передбачає послідовне перетворення вхідного тексту на акустичне представлення, а потім на готовий звуковий файл. Сучасні нейромережеві підходи, зокрема трансформерні архітектури, VITS-подібні моделі та генеративні системи, значно підвищують природність синтезованого мовлення.

Модель Supertonic 3 можна розглядати як перспективне програмне рішення для побудови локальної системи синтезу мовлення. Її перевагами є підтримка локального запуску, використання ONNX Runtime, відносно компактна модель та можливість інтеграції через Python SDK. Застосування Supertonic Voice Builder розширює можливості Supertonic 3 за рахунок створення власного голосового стилю на основі еталонного аудіозразка. Отриманий voice-style JSON може використовуватися під час синтезу мовлення для наближення згенерованого голосу до характеристик конкретного мовця.

Експериментальна реалізація моделі Supertonic 3 у Google Colab дозволила перевірити повний цикл генерації аудіо з українськомовного

тексту. У процесі роботи було виконано підготовку середовища, ініціалізацію моделі, вибір голосового стилю, синтез мовлення, збереження WAV-файлу та перевірку його тривалості. Отриманий результат підтверджує можливість практичного використання Supertonic 3 для створення україномовного аудіо на основі текстових даних. Додатково запропоновано використання Docker і локального HTTP-сервера, що робить систему більш відтворюваною, автономною та придатною для інтеграції в локального AI-агента.

Економічне обґрунтування показало доцільність використання локальної системи генерації аудіо з текстових даних за умови її регулярного застосування. Основні витрати пов’язані з налаштуванням програмного середовища, супроводом системи та споживанням електроенергії, тоді як витрати на хмарну генерацію аудіо можуть бути значно вищими. За наведеними розрахунками локальна система може окупитися приблизно за три місяці. Крім економічної вигоди, локальний підхід забезпечує автономність, контроль над даними та можливість інтеграції в закриті інформаційні системи або локальні AI-агенти.

Таким чином, результатами роботи є модель локальної системи для генерації аудіо з текстових даних з підтримкою україномовного контенту, рекомендації щодо практичної реалізації локальної системи для генерації аудіо з текстових даних з підтримкою україномовного контенту. Вони можуть бути використані для подальших досліджень за даною тематикою та при проектуванні локальних AI-агентів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Slyusar V., Sliusar I. Leveraging pre-trained neural networks for image classification in audio signal analysis for mobile applications of home automation. *Research Tendencies and Prospect Domains for AI Development and Implementation*. River Publishers, 2024. P. 109-128.

2. Пивовар Б. Генерація аудіо з текстових даних на основі qwen3-tts-0.6b-base. *Збірник XXI щорічної студентської наукової конференції «Сучасні інформаційні технології та інноваційні методики в економіці, менеджменті та бізнесі» Полтавського державного аграрного університету* (квітень 2026 р., м. Полтава). Полтава: ПДАУ, 2026 р. С. 73, 74.

3. Create Realistic Ukrainian Text to Speech. Elevenlabs. URL: https://elevenlabs.io/text-to-speech/ukrainian (дата звернення: 05.05.2026).

4. Text-to-Speech AI. *Google*. URL: https://cloud.google.com/text-to-speech (дата звернення: 05.05.2026).

5. Ukrainian Text To Speech. Narakeet. URL: https://www.narakeet.com/languages/ukrainian-text-to-speech (дата звернення: 05.05.2026).

6. Azure Speech in Foundry Tools. Microsoft. URL: https://azure.microsoft.com/en-us/products/ai-foundry/tools/speech (дата звернення: 05.05.2026).

7. gTTS. URL: https://github.com/pndurette/gtts (дата звернення: 05.05.2026).

8. RHVoice. Installation. URL: https://rhvoice.org/installation/ (дата звернення: 05.05.2026).

9. Rhasspy/piper-voices. URL: https://huggingface.co/rhasspy/piper-voices (дата звернення: 05.05.2026).

10. Українська мова в Balacoon. URL: https://balacoon.com/blog/uk_release/ (дата звернення: 05.05.2026).

11. Українські TTS. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Українські_TTS (дата звернення: 05.05.2026).

12. StyleTTS2-AI Voice Generation System. *StyleTTS2*. URL: https://styletts2.com (дата звернення: 05.05.2026).

13. Coqui TTS & XTTS V2: Free Text to Speech. Coqui TTS. URL: https://coquitts.com (дата звернення: 05.05.2026).

14. Supported voices and languages. *Googl*e. URL: https://docs.cloud.google.com/text-to-speech/docs/list-voices-and-types (дата звернення: 05.05.2026).

15. Language and voice support for Azure Speech. *Microsoft*. URL: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/language-support?tabs=stt (дата звернення: 05.05.2026).

16. Robinhad/ukrainian-tts. URL: https://huggingface.co/spaces/robinhad/ukrainian-tts/blob/c9d5f8fac198dd23b0cc7338b546b3f98769f1ef/README.md (дата звернення: 05.05.2026).

17. Incorrect pronunciation of Ukrainian voice Lada #32. URL: https://github.com/rhasspy/piper/issues/32 (дата звернення: 05.05.2026).

18. Ukrainian voices. URL: https://rhvoice.org/uk-voices/ (дата звернення: 05.05.2026).

19. Rhasspy/piper. URL: https://github.com/rhasspy/piper/blob/master/VOICES.md (дата звернення: 05.05.2026).

20. Ukrainian Female TTS Model Vits Encoding Trained on Mai Dataset at 22050H. URL: https://aimodels.org/ai-models/text-to-speech-synthesis/ukrainian-female-tts-model-vits-encoding-trained-on-mai-dataset-at-22050hz (дата звернення: 05.05.2026).

21. StyleTTS2 ukrainian demo. URL: https://huggingface.co/spaces/patriotyk/styletts2-ukrainian (дата звернення: 05.05.2026).

22. Supertone-inc/supertonic. URL: https://github.com/supertone-inc/supertonic (дата звернення: 05.05.2026).

23. Robinhad/ukrainian-tts. URL: https://github.com/robinhad/ukrainian-tts (дата звернення: 05.05.2026).

24. Kim H., Yang J., Yu Y.SupertonicTTS: Towards Highly Scalable and Efficient Text-to-Speech System. URL: https://arxiv.org/html/2503.23108v1 (дата звернення: 05.05.2026).

25. Supertone-inc/supertonic. URL: https://github.com/supertone-inc/supertonic (дата звернення: 05.05.2026).

26. Supertone/supertonic-3. URL: https://huggingface.co/Supertone/supertonic-3 (дата звернення: 05.05.2026).

27. Dettmers T., Pagnoni A., Holtzman A., Zettlemoyer L. QLoRA: Efficient finetuning of quantized LLMs. *Advances in neural information processing systems*,

36:10088–10115, 2023. URL: https://arxiv.org/abs/2305.14314 (дата звернення: 05.05.2026).

28. Liu A., Feng B., Wang B. et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. arXiv preprint. arXiv:2405.04434, 2024. URL: https://arxiv.org/abs/2405.04434 (дата звернення: 05.05.2026).

29. Podell D., English Z., Lacey K., Blattmann A. et al. SDXL: Improving latent diffusion models for high-resolution image synthesis. *In The Twelfth International Conference on Learning Representations*, 2024. URL: https://openreview.net/forum?id=di52zR8xgf (дата звернення: 05.05.2026).

30. Rombach R., Blattmann A., Lorenz D. et al. Highresolution image synthesis with latent diffusion models. *In Proc. of the IEEE/CVF conf. on computer vision and pattern recognition*, 2022. P. 10684–10695.

31. Lovelace J., Kishore V., Wan C. et al. Latent diffusion for language generation. *Advances in Neural Inf. Processing Systems*, 2023. 36:56998–57025.

32. Lovelace J., Ray S., Kim K. et al. Simple-TTS: End-to-end text-to-speech synthesis with latent diffusion, 2024. URL: https://openreview.net/forum?id=m4mwbPjOwb (дата звернення: 05.05.2026).

33. Liu Z., Mao H., Wu C. et al. A convnet for the 2020s. *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition* (CVPR), 2022.

34. Siuzdak H. Vocos: Closing the gap between time-domain and fourier-based neural vocoders for high-quality audio synthesis. *In The Twelfth International Conference on Learning Representations*, 2024. URL: https://openreview.net/forum?id=vY9nzQmQBw (дата звернення: 05.05.2026).

35. Okamoto T., Yamashita H., Ohtani Y., Wavenext: Convnext-based fast neural vocoder without istft layer. *In 2023 IEEE Automatic Speech Recognition and Understanding Workshop* (ASRU), IEEE, 2023. P. 1–8.

36. Lee K., Kim D., Kim J. et al. DiTTo-TTS: Diffusion transformers for scalable text-to-speech without domain-specific factors. *In The Thirteenth International Conference on Learning Representations*, 2025. URL: https://openreview.net/forum?id=hQvX9MBowC (дата звернення: 05.05.2026).

37. Podell D., English Z., Lacey K. et al. SDXL: Improving latent diffusion models for high-resolution image synthesis. *In The Twelfth International Conference on Learning Representations*, 2024. URL: https://openreview.net/forum?id=di52zR8xgf (дата звернення: 05.05.2026).

38. Zhang J., Liu Z., Yan L. et al. Stage-wise Distortion-Perception Traversal in Zero-shot Inverse Problems with Diffusion Models. URL: https://arxiv.org/abs/2605.28711 (дата звернення: 05.05.2026).

39. Wang X., Jiang M., Ma Z. et al. Spark-tts: An efficient llm-based text-to-speech model with single-stream decoupled speech tokens. arXiv preprint arXiv:2503.01710, 2025. URL: https://arxiv.org/abs/2503.01710 (дата звернення: 05.05.2026).

40. He K., Zhang X., S. Ren, Sun J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *In Proceedings of the IEEE international conference on computer vision*, 2015. P. 1026–1034.

41. Lee S., Ping W., Ginsburg B. et al. BigVGAN: A universal neural vocoder with large-scale training. *In The Eleventh International Conference on Learning Representations*, 2023. URL: https://openreview.net/forum?id=iTtGCMDEzS_ (дата звернення: 05.05.2026).

42. Kong J., Kim J., Bae J. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in neural information processing systems*, 2020. 33:17022–17033.

43. Goodfellow I., Pouget-Abadie J., Mirza M. et al. Generative adversarial nets. *Advances in neural information processing systems*, 2014. 27.

44. Jang W., Lim D., Yoon J. et al. UnivNET: A neural vocoder with multi-resolution spectrogram discriminators for high-fidelity waveform generation. *In Interspeech*, 2021. URL: https://api.semanticscholar.org/CorpusID:235435945 (дата звернення: 05.05.2026).

45. Lipman Y., Chen R., Ben-Hamu H. Flow matching for generative modeling. *In The Eleventh International Conference on Learning Representations*, 2023. URL: https://openreview.net/forum?id=PqvMRDCJT9t (дата звернення: 05.05.2026).

46. Vaswani A., Shazeer N., Parmar N. et al. Attention is all you need. *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf (дата звернення: 05.05.2026).

47. Ho J., Jain A., Abbeel P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 2020. 33:6840–6851.

48. Song Y., Sohl-Dickstein J., Kingma D. et al. Score-based generative modeling through stochastic differential equations. arXiv preprint

arXiv:2011.13456, 2020. URL: https://arxiv.org/abs/2011.13456 (дата звернення: 05.05.2026).

49. Choi H., Yang J., Lee J. et al. Nansy++: Unified voice synthesis with neural analysis and synthesis. *In The Eleventh International Conference on Learning Representations*, 2023.

50. Kim J., Lee K., Chung S. CLam-TTS: Improving neural codec language model for zero-shot text-to-speech. In The Twelfth International Conference on Learning Representations, 2024. URL: https://openreview.net/forum?id=ofzeypWosV (дата звернення: 05.05.2026).

51. Pricing. URL: https://play.supertone.ai/subscription (дата звернення: 05.05.2026).

52. Create Custom Voices for Supertonic. Supertonic. URL: https://supertonic.supertone.ai/voice-builder (дата звернення: 05.05.2026).

53. Wang C., Chen S., Wu Y. et al. Neural codec language models are zero-shot text to speech synthesizers. *IEEE Transactions on Audio, Speech and Language Processing*, 2023. 33:705–718. URL: https://api.semanticscholar.org/CorpusID:255440307.

54. Le M., Vyas A., Shi B., et al. Voicebox: Text-guided multilingual universal speech generation at scale. Advances in neural information processing systems, 2024. 36 p. URL: https://arxiv.org/abs/2306.15687 (дата звернення: 05.05.2026).

55. Jiang Z., Liu J., Ren Y. et al. Mega-TTS 2: Boosting prompting mechanisms for zero-shot speech synthesis. *In The Twelfth International Conference on Learning Representations*, 2024. URL: https://openreview.net/forum?id=mvMI3N4AvD (дата звернення: 05.05.2026).

56. Kim J., Lee K., Chung S. et al. CLam-TTS: Improving neural codec language model for zero-shot text-to-speech. *In The Twelfth International Conference on Learning Representations*, 2024. URL: https://openreview.net/forum?id=ofzeypWosV (дата звернення: 05.05.2026).

57. NVIDIA A100 Tensor Core GPU. NVIDIA. URL: https://www.nvidia.com/en-us/data-center/a100 (дата звернення: 05.05.2026).