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

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

магістра
(назва освітньо-кваліфікаційного рівня)

студента *Загинайло Євгенія Олександровича*
(ПІБ)

академічної групи *121м-21-1*
(шифр)

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

освітньої програми *«Інженерія програмного забезпечення»*
(назва освітньої програми)

на тему: *Розробка програмного забезпечення для дослідження методик синтезу мовлення та створення моделі трансляції тексту до мов*

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
розділів кваліфікаційної роботи				
спеціальний	*Проф. Алексєєв М.О.*			

Рецензент				

Нормоконтролер	*Проф. Лактіонов І.С.*			

Дніпро
2022

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

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

_____ М.О. Алексєєв
(підпис) (прізвище, ініціали)

« » _____ 20 22 року

ЗАВДАННЯ
на виконання кваліфікаційної роботи магістра

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

студенту *121м-21-1* *Загинайло Євгенія Олександровича*
(група) (прізвище та ініціали)

Тема кваліфікаційної роботи *Дослідження методик синтезу мовлення для створення моделі трансляції тексту до мов*

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора НТУ «Дніпровська політехніка» від 24.10.2022р. №1169-с

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень – процес синтезу мовлення з використанням нейронної мережі.

Предмет досліджень – методи синтезу мовлення та обробки сигналів.

Мета НДР – створення моделі синтезу мовлення за умови проведення дослідження методів синтезу та обробки сигналів.

Вихідні данні для проведення роботи – теоретичні та практичні навички отримані в ході дослідження, аудіо та текстова інформація з набора даних української мови.

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Наукова новизна даної роботи полягає в створенні моделі трансляції тексту до мовлення, що базується на графемах української мови, з використанням структури акустичної моделі Glow-TTS.

Практична цінність результатів полягає в створенні програмного забезпечення моделі трансляції з використанням структури швидкої акустичної моделі Glow-TTS

для можливості подальшого покращення даної моделі та створення нових моделей використовуючи накопичені знання.

4 ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Результати досліджень мають бути подані у вигляді, що дозволяє побачити та оцінити якість удосконаленої моделі трансляції до українського мовлення.

5 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	**Строки виконання робіт (початок – кінець)**
Аналіз теми і постановка задачі	10.09.2022 – 30.09.2022
Аналіз методів трансляції тексту до мовлення та обробки сигналів	01.10.2022 – 31.10.2022
Навчання моделі трансляції тексту до українського мовлення та проведення аналізу результатів	01.11.2022 – 30.11.2022

Завдання видав _____ *Алексєєв М.О.*
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ *Загинайло Є.О.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *10.09.2022 р.*

Термін подання кваліфікаційної роботи до ЕК: *12.12.2022 р.*

РЕФЕРАТ

Пояснювальна записка: 73 сторінки, 30 рисунків, 5 таблиць, 2 додатка, 40 джерел.

Об’єкт дослідження: процес синтезу мовлення з використанням нейронної мережі.

Мета магістерської роботи: створення моделі синтезу мовлення за умови проведення дослідження методів синтезу та обробки сигналів.

Методи дослідження: для вирішення поставлених задач використані: нейронні мережі, хмарні технології, функціональне програмування, моделі трансляції тексту, методи обробки сигналів та Unix операційні системи.

Наукова новизна даної роботи полягає в створенні моделі трансляції тексту до мовлення, що базується на графемах української мови, з використанням структури акустичної моделі Glow-TTS.

Практична цінність результатів полягає в створенні програмного забезпечення моделі трансляції з використанням структури швидкої акустичної моделі Glow-TTS для можливості подальшого покращення даної моделі та створення нових моделей використовуючи накопичені знання.

Область застосування: Результат кваліфікаційної роботи може застосовуватись у сферах, які так чи інакше потребують роботи зі звуком та промовляння повідомлень або для подальшого дослідження з урахуванням напрацювань отриманих в цій роботі.

Значення роботи та висновки: досліджено методики синтезу мовлення та обробки сигналів та створена модель трансляції тексту до мовлення, яка була модифікована можливістю навчання моделей мовлення української мови, що дає змогу користуватись отриманим продуктом або використовувати його з метою створення нових застосунків на базі наявних.

Прогнози щодо розвитку досліджень: дослідити методи покращення синтезу альтернативними реалізаціями вокодерів, оптимізації параметрів конфігурації моделі та моделі в цілому.

Список ключових слів: FFT, STFT, нейронна мережа, рекурентна нейронна мережа, Glow-TTS, модель синтезу мовлення.

ABSTRACT

Explanatory note: 73 pages, 30 figures, 5 tables, 2 applications, 40 sources.

Object of research: the process of speech synthesis by neural networks utilization.

The purpose of the master's thesis: creating a model of speech synthesis under the condition of researching synthesis methods and signal processing.

Research methods: neural networks, cloud technologies, functional programming, text translation models, signal processing methods and Unix operating systems are used to solve the problems.

Originality of research lies in the creation of a text-to-speech translation model based on the graphemes of the Ukrainian language, using the structure of the Glow-TTS acoustic model.

The practical value of the results lies in the creation of a software of a broadcast model using the structure of the Glow-TTS fast acoustic model for the possibility of further improving this model and creating new models using the accumulated knowledge.

Scope of application: The result of the qualification work can be applied in areas that in one way or another require work with sound and speaking messages or for further research taking into account the results obtained in this work.

The value of this work and conclusions: methods of speech synthesis and signal processing were researched and a model of text-to-speech translation was created, which was modified by the possibility of teaching speech models of the Ukrainian language, which makes it possible to use the obtained product or to use it to create new applications based on the existing one.

Research forecast and development: investigate the ways of improving the synthesis by alternative implementations of vocoders, optimizing the configuration parameters of the model and the model as a whole.

In the "Economics" section, the complexity of developing a text-to-speech translator was calculated and a market study was conducted with the calculation of the economic efficiency of using this application in the organization.

Key words: FFT, SFFT, neural network, recurrent neural network, Glow-TTS, speech synthesis model.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

FFT – Fast Fourier Transform

STFT – Short Fast Fourier Transform

TTS – Text to speech

RNN – Recurrent neural network

CNN – Convolutional neural network

ЗМІСТ

ВСТУП

Наразі синтез мовлення є дуже актуальною темою для досліджень. Синтез людського мовлення може використовуватись для покращення роботи систем оповіщення, формування аудіо книжок, допомоги людям з різними вадами здоров'я або використання в будь-яких інших сферах які потребують постійного відтворення змінної кількості та наповнення текстової інформації.

В наш час найбільш популярним є створення нейронних мереж різних типів для кодування текстової інформації у звукове представлення.

Поруч з предковічним акустичним, або артикуляційним типом трансляції існують такі типи як: статистично параметричний, конкатенативний та різні типи трансляції з використанням глибокого навчання.

Не зважаючи на те, що наразі метод трансляції за допомогою нейронної мережі вважається найбільш популярним та наближеним до натурального мовлення – глибоке навчання завжди вимагає велику кількість часу та обсяг даних для отримання необхідної інформації про особливості мовлення. До того ж частіше за все в моделях трансляції заснованих на нейронних мережах потрібно застосовувати більше однієї нейронної мережі що значно збільшує обсяг роботи яку потрібно виконати для досягнення результату.

Об'єкт досліджень: процес синтезу мовлення з використанням нейронної мережі.

Предмет досліджень: методи синтезу мовлення та обробки сигналів.

Мета дослідження: створення моделі синтезу мовлення за умови проведення дослідження методів синтезу та обробки сигналів.

Методи дослідження: для вирішення поставлених задач використані: нейронні мережі, хмарні технології, функціональне програмування, моделі трансляції тексту, методи обробки сигналів та Unix операційні системи.

Наукова новизна даної роботи постає в розробці моделі трансляції тексту до мовлення, яка базується на графемах української мови, з використанням структури акустичної моделі Glow-TTS.

Практичне значення: Результат створення моделі трансляції з використанням структури швидкої акустичної моделі Glow-TTS дає можливість використання даної моделі, можливості її покрашення в майбутньому та створення нових моделей використовуючи накопичені знання.

Особистий внесок автора полягає в розробці моделі Glow-TTS з модифікацією можливістю навчання моделі на наборах даних української мови з попереднім дослідженням теоретичної частини та систематизації отриманих знань.

Структура та обсяг дипломної роботи: Кваліфікаційна робота складається з трьох розділів та висновків. Містить 73 сторінки тексту, 30 рисунків, 40 використаних джерел та 2 додатки.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Історія предметної галузі

Протягом останніх двох сотень років людство працює над створенням пристрою або застосунку, який мав би змогу імітувати людське мовлення. З часом отримані знання накопичуються формуючи цілі наукові напрямки, які після достатнього дослідження здатні породити речі, які раніше могли здаватись неймовірними.

Історія створення сучасних синтезаторів, або трансляторів мовлення починається з роботи вченого Крістіана Готліба, який отримав премію на конкурсі оголошеному академією наук за створену модель людського тракту який міг відтворювати прості голосні. Згодом, в 1769 році, була презентована акустично механічна мовна машина Вольфганг фон Кемпелена[1]. Основними складовими його машини були прилад для створення тиску, який слугував в якості легенів, металева пластина для створення вібрацій та шкіряна трубка в якості людського тракту. Через декілька років в машину були додані людські губи та язик що дало смогу почати створювати, наряду з голосними, приголосні звуки. Дана машина була створена не з силікону, але викарбувана з дерева. В 1791 році Кемпелен видав книжку в якій були описані усі його напрацювання в даній сфері та його спостереження в цілому.

Згодом, а саме в 1863 році була написана стаття Германом Фон Гельмгольцем про створення розумної машини синтезу мовлення, яка базувалась на електричних камертонах.

Наступним етапом в пізнанні творення людського мовлення було дослідження описані в роботі С. Штумпфа, О.Д. Рассела та Р. Педжета в якій вирішили розглянути структуру спектральних сигналів. Пристрій дослідження дав С. Штумпфу змогу синтезувати гарні голосні, що стало гарним прикладом у

формуванні звуків мовлення. О.Д. Рассел описав дослідження базовані на рентгенівських зображеннях людських артикуляторів, що дало змогу спростувати хибну фізіологію дослідників минулого та поклало новий фундамент в дослідженні мовлення.

Першим показом створення мовленнєвого синтезатора була робота Т. Сейновського. Під час демонстрації був представлений процес навчання мережі заснований на зворотному поширенні коригувальних правил. Та продемонстровано його перетворення з дитячого лепетання на звичайне доросле мовлення.

Навіть в наш час сфера синтезу мовлення з використанням глибинних нейронних мереж не є повністю вивченою, а саме тому дана тема є однією з найперспективніших.

1.2. Актуальні сфери використання транслятора мовлення

В час технологічної революції здатність створення нового продукту, який може бути корисним для великої кількості людей є головною ціллю багатьох компаній та корпорацій. Особливо корисним зазвичай можна вважати продукти, які використовуються людьми протягом великої кількості часу та ефективність яких є на стільки великою, що людина не може уявити собі життя без нього.

Головними користувачами, життя яких зазнає найбільших змін – це люди з вродженими або набутими дефектами мовлення та зору [2]. Наприклад є багато людей, які можуть отримувати інформацію з навколишнього середовища використовуючи слух, але мають ті або інші проблеми з зором які унеможливлюють або неймовірно погіршують можливість отримувати зорову інформацію. Навпаки, є люди, які можуть повноцінно читати або частково отримувати інформацію за допомогою зору, але абсолютно не мають можливості щось почути. Для кожного з таких людей створення системи трансляції тексту до мовлення або навпаки мовлення до тексту можливо було б одним з

найважливіших покращень життя. Не зважаючи на те що в наші часи системи трансляції тексту є одним з найважливіших пріоритетів великих корпорацій на кшталт Google, або Microsoft якість створених рішень може бути покращена.

Наступним найперспективнішим рішенням яке могло б допомогти окремим людям з обмеженими можливостями – це аудіо інтерфейс. Даний продукт дав би можливість людям з захворюваннями опорно-рухового апарату повноцінно користуватись пристроями, які оснащені даною функцією. Пристрій міг би забезпечувати повноцінний діалог з користувачем та виконувати команди та відтворювати звукову інформацію згідно з інструкціями користувача абсолютно без фізичної взаємодії з ним.

Також модель трансляції тексту до мовлення може бути ефективно застосована у сфері навчання [3]. Основним плюсом синтезатора є те, що комп’ютер обладнаний ним можна вважати вчителем, при заздалегідь заданій навчальній програмі. Це означає що цей вчитель може виконувати свою роботу без вихідних, цілий день протягом року. Синтез та розпізнавання також не рідко використовуються в випадках допомоги в вимовлені окремих слів або в перевірці правильного промовляння. Прикладом цього може слугувати Google перекладач в якому можна відтворити написане слово або поле з голосовим вводом на головній сторінці компанії.

Одним з інших прикладів використання синтезатора мовлення можна привести телефонний автовідповідач, який розміщується в багатьох компаніях з ціллю розвантажити вхідну телефонну лінію. Тобто цей автовідповідач буде намагатись відповісти на поставлене питання користувача не з’єднуючись з оператором, що потенціально дасть можливість залучати меншу кількість операторів при сталій кількості дзвінків.

1.3. Аналіз методологічних підходів

В даному розділі будуть розглянуті та проаналізовані різні типи та реалізації створення моделей трансляції тексту до мовлення та підходи обробки сигналів.

1.3.1. Метод артикуляційного синтезу

Метод артикуляційного синтезу – метод який базується на моделюванні людського голосового тракту (голосового апарату) та процесу артикуляції, який забезпечується завдяки відтворення змодельованих звуків язика та губ [4]. Даний вид синтезу є одним з найстаріших та базується на фізіології людини. Саме через це успіхи в синтезу мовлення цим методом безпосередньо залежали від рівня знань людства про процес звукоутворення людиною, які прогресували з часом сильно змінюючи в кращу сторону сферу синтезу.

Головною задачею моделі голосового тракту є генерація мовлення виходячи з усіх необхідних геометричних параметрів та позиції органів голосового тракту:

- губ
- язика
- піднебіння
- велуму (ділянки між піднебінням та язичком)
- горла
- гортані

Форма, положення та рух при синтезі генеруються відповідно до функцій часу кожного параметру голосового тракту який визначений в моделі. Основними параметрами голосового тракту, які використовуються в моделі є положення:

- щелепи
- верхньої губи

- нижньої губи
- кожної частини язика (кінчик та тіло)
- велуму
- гортані

Моделі голосового тракту можна поділити на статистичні, біомеханічні та геометричні.

Статистичні моделі базуються на великому обсягу даних руху голосового тракту які були виміряні такими методами як: МРТ, ЕМА або рентген. Артикуляційна інформація (точки які відображають позицію органу голосового тракту або в цілому форму усіх органів голосового тракту) отримується в кожну елементарну одиницю часу.

Біомеханічна модель націлена на моделювання фізіології усіх органів голосового тракту та їх нервово-м’язового контролю. Даний вид моделей зазвичай ґрунтується на фізіологічних знаннях про структуру м’язів, тканини, хрящів та кісткову структуру голосового апарату.

В геометричній моделі позиція та форма органів голосового апарату зазвичай розраховується з використанням заздалегідь визначених параметрів голосового тракту. Основа для визначення набору параметрів залежить від припущень щодо різноманітних конфігурацій які даний вид моделі намагається наблизити. Даний тип моделей є досить гнучким, що дає змогу легкої адаптації до артикуляційних апаратів різних спікерів. Іншими словами даний вид моделі може бути адаптований до мовця будь-якої статі, віку або з іншими параметрами які впливають на зміну параметрів голосового апарату.

Даний метод синтезу мовлення виділяється великою ефективністю, гнучкістю та якістю синтезованого мовлення, але основними недоліками його є неймовірна складність в порівнянні з іншими методами. Для того, щоб зробити додаток, що б правильно синтезував, треба бути дуже добре обізнаним як

мінімум у сфері анатомії, фізіології та декількох сферах програмування і добре орієнтуватись в тому як проходять процеси синтезу звуків у людини.

1.3.2. Метод форматного синтезу

Форматний синтез це один з видів параметричного синтезу. Даний вид синтезу як і усі варіанти синтезів починається з нормалізації тексту (трансформування тексту в канонічну форму для покращення якості синтезу). Прикладом нормалізації тексту може слугувати такий текст як «221₴» - такий текст буде нормалізований до виду «двісті двадцять одна гривня», або інший приклад – римські літери «vi» буде нормалізовано до «шість» та інше. Даний етап дозволяє позбутись таких можливих проблем в тексті[5] як:

- лого графічні знаки, такі як (&, @, #) будуть перетворені на і, в, «шарп» або «номер» відповідно
- абревіатури (м. – метр, мм – міліметр, кг – кілограм та інші)
- акроніми (НТУ ДП – «ентеудепе», ЮМЗ – «юемзе», ЖД – «жеде»)

Далі в основному йде крок перетворення отриманого нормалізованого тексту до фонем[6], що будуть використані як найменші складові синтезованих слів. Тривалість фонем залежить від швидкості мовлення та від місця даної фонеми в слові або реченні. Це також залежить від синтаксичних і семантичних міркувань. Українська мова налічує в собі 38 основні фонеми (6 голосних та 32 приголосні).

Таблиця 1.1

Голосні фонеми

1	[i]
2	[ɪ]

Таблиця 1.2

Приголосні фонеми

1	[m]
2	[b]

17	[tʲ]
18	[d͡zʲ]

3	[ɛ]
4	[ɑ]
5	[ɔ]
6	[u]

3	[p]
4	[f]
5	[w]
6	[n]
7	[d]
8	[t]
9	[d͡z]
10	[t͡s]
11	[z]
12	[s]
13	[l]
14	[r]
15	[nʲ]
16	[dʲ]

19	[t͡sʲ]
20	[zʲ]
21	[sʲ]
22	[lʲ]
23	[rʲ]
24	[d͡ʒ]
25	[t͡ʃ]
26	[ʒ]
27	[ʃ]
28	[j]
29	[g]
30	[k]
31	[x]
32	[ɦ]

Наступним кроком після визначення тривалості та інтонації для кожної фонеми буде визначення акустичних параметрів (формантів, пропускних можливостей, амплітуд) та їх взаємодією між фонемами [7].

Синтез тексту полягає не лише в об’єднанні різних параметрів, характерних для кожної фонеми. Деякі значення залежать від позиції слова, складу та навіть від того якою частиною мови вони є. Більш того вони залежать від семантики та емоційного стану. Виходячи з усіх цих складових – просте об’єднання фонем не може дати ідеального результату.

Інтонація та довжина вимовляння фонеми є двома важливими складовими просодії саме через це для кожної фонеми розраховується значення інтонації та тривалості. Розрахунок залежить не тільки від слова, фрази, речення, але й від частини мови кожного слова яке було знайдене у внутрішній таблиці пошуку службових слів.

Основною ціллю роботи форматного синтезу є створення синтезованого мовлення використовуючи адитивний синтез та акустичну модель. Для досягнення поставленого результату проводиться зміна параметрів частоти, інтонації та навіть рівня шуму.

Не зважаючи на можливу низьку якість (відносно натуральної людської мови) даний метод синтезу є достатньо швидким в порівнянні з іншими представленими методами та може бути надійно використаний у випадках, коли за ціль поставлене формування зрозумілої якості мовлення у відносно короткий термін.

1.3.3. Метод конкатенативного синтезу

Конкатенативний метод синтезу мовлення є достатньо перспективним, але при тому ж відносно простим методом синтезу мовлення. В реалізаціях даного методу використовуються великі бази даних звуків, або навіть в окремих випадках складів, слів або речень [8].

Принципом даного алгоритму являється конкатенація заздалегідь збережених частин у слова, речення і параграфи. Збережені звуки сегментують на «одиниці» та використовують Unit Selection Algorithm або алгоритм вибору одиниць для вибору найбільш придатних частин [9]. Послідовність одиниць, які якомога краще відповідають звуку або фразі яку потрібно синтезувати називається цільовою послідовністю.

Конкатенативний синтез може бути більше або менше залежним від даних [10]. Тобто замість строгого задання правил формування, створених завдяки кропіткому міркуванню (як, наприклад у підході заснованому на правилах[11]) правила формуються виходячи з наявних даних. Перевагою даного методу є те, що завдяки великої бази даних звуків можна достатньо просто підібрати правильну звукову одиницю.

В конкатенативному синтезі можна виділити декілька основних способів формування вихідного мовлення:

- Синтез інструментів високого рівня
- Повторний синтез аудіо
- Вільний синтез

У випадку синтезу інструментів високого рівня конкатенативний синтез може усвідомлювати не тільки цільові звукові одиниці, але й контекст бази даних. Можливість відбору одиниць з відповідних контекстів покращує якість синтезу та дає можливість створювати переходи природного звучання. Інформацію, пов’язану з вихідними звуками, можна використати для відбору цільових одиниць, що забезпечує гарний контроль синтезу, де дрібні деталі, яких бракує цільовій специфікації, заповнюються одиницями з бази даних.

Повторний синтез аудіо працює шляхом відбору звуку або фрази з бази звуків та повторного синтезу його з такою ж висотою, амплітудою та тембром використовуючи цільові одиниці з бази даних.

Вільний синтез з різних звукових баз даних дає мовленнєвому синтезатору ефективний контроль результату завдяки сприйнятно-значущим дескрипторам та дозволяє переглядати та досліджувати корпус звуків.

Конкатенативний синтез можна порівняти зі складанням мозаїки. Це складний метод, який потребує взаємодії багатьох різних концепцій, тому багато роботи лише над одним аспектом буде лише одним кроком для формування цілого.

Опис роботи даного методу може бути складним, але часто даний метод використовується у достатньо тривіальних випадках. Приклад використання конкатенативного методу – система оповіщення на підприємствах. Заздалегідь записані частини речень з відповідним контекстом пов’язаним з виробництвом

використовуються в синтезі формуючи достатньо якісний результат що спонукає до подальшого використання даного методу.

1.3.4. Метод синтезу заснований на нейронних мережах.

Метод, який використовує нейронні мережі для синтезу людського мовлення заведено називати мовленнєвим синтезатором глибинного навчання. Глибинне навчання – техніка машинного навчання, яка базується на навчанні машини робити звичні речі для людини, але незвичайні для машини [12]. Даній техніці машинного навчання притаманні великий розмір даних для навчання та нейронні мережі з великою кількістю шарів.

Мовленнєвий синтезатор використовує нейронну мережу глибинного навчання задля генерації штучного мовлення використовуючи вхідний текст або значення спектрограм [13]. Даний вид нейронних мереж навчається з використанням великої кількості наборів даних. Вхідними даними для такого виду моделі є записані аудіо файли голосу людини, якого в рамках навчання називають «мовець» та текст, що дає розуміння що говориться в кожному з аудіо файлів [14].

Базові реалізації мовленнєвого синтезатора виконувались з використанням рекурентних прихованих нейронних шарів [15], але в наш час розробники намагаються уникати даного типу шарів через їх велику зв’язність та залежність від результату минулого розрахунку. Зі збільшенням розмірів нейронної мережі час, який потрібно використати для навчання рекурентного шару суттєво збільшується, що унеможливлює подальше дослідження на машині із посереднім рівнем обчислювальної потужності [16].

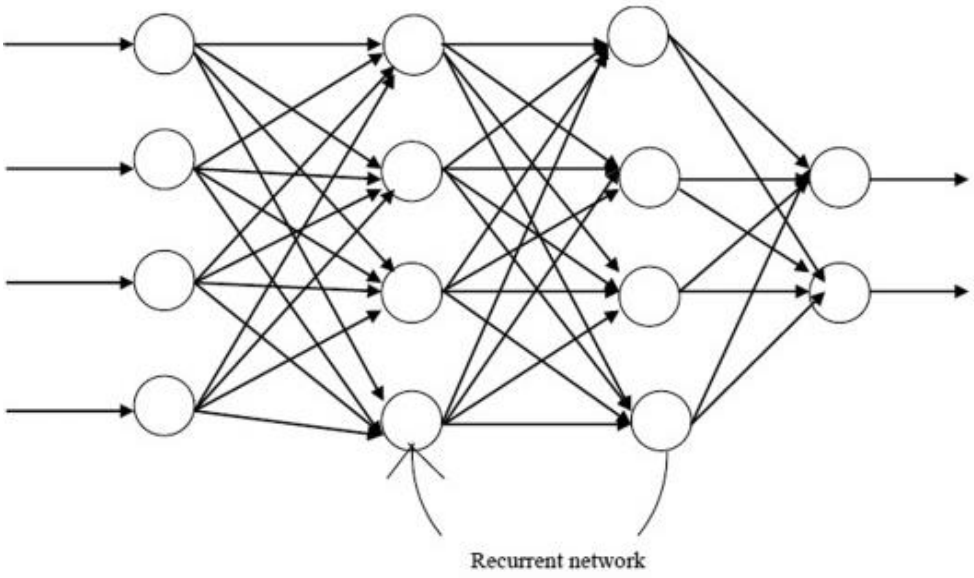

Рис. 1.1 Мережа з рекурентними шарами.

В якості аналога рекурентним шарам нейронної мережі наразі використовуються згорткові шари, які не мають залежності від розрахунку минулого кроку і тому використовуються в версіях нейронних мереж, які намагаються балансувати між якістю та швидкістю синтезу мовлення.

1.3.5. Перетворення Фур’є.

Швидке перетворення Фур’є – один з найважливіших методів вимірювання в науці про аудіо та акустичні вимірювання[17]. Його ціллю є перетворення сигналу в окремі спектральні компоненти тим самим передавши інформацію про частотну характеристику окремого сигналу. Тобто іншими словами дане перетворення робить розбиття сигналу на альтернативне представлення, яке характеризується функціями косинуса або синуса різних частот. Таке перетворення доводить, що будь-яку хвилю можна представити у вигляді суми синусоїд.

Швидке перетворення використовується для аналізу несправностей, контролю якості та перевірки умов машин або систем [18]. На рисунку 1.2 поданий приклад роботи даного алгоритму, де зліва зображений сигнал, час

відтворення якого складає 16 секунд та справа відображений графік його спектру частот.

Для просторового представлення процесу перетворення була наведена ілюстрація зображена на рисунку 1.3 [19]. Ліва частина рисунка містить графік сигналу в якому відбувається накладання синусоїд різної частоти, посередині безпосередньо спрощений набір синусоїд, які формують сигнал та справа — графік частотного спектра.

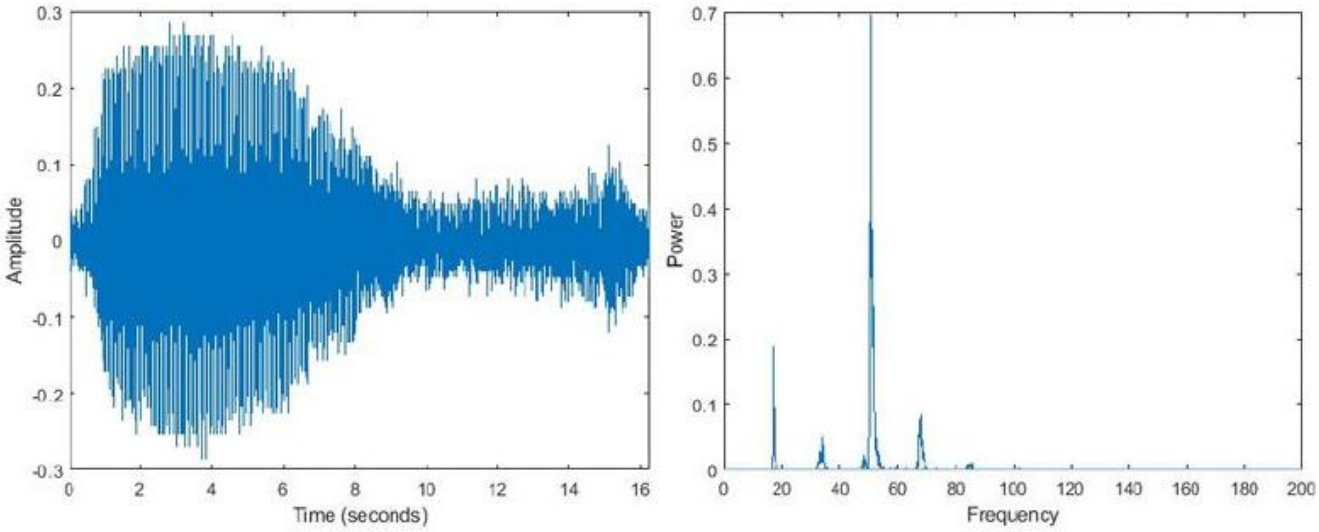

Рис. 1.2. Приклад перетворення сигналу на спектр частот

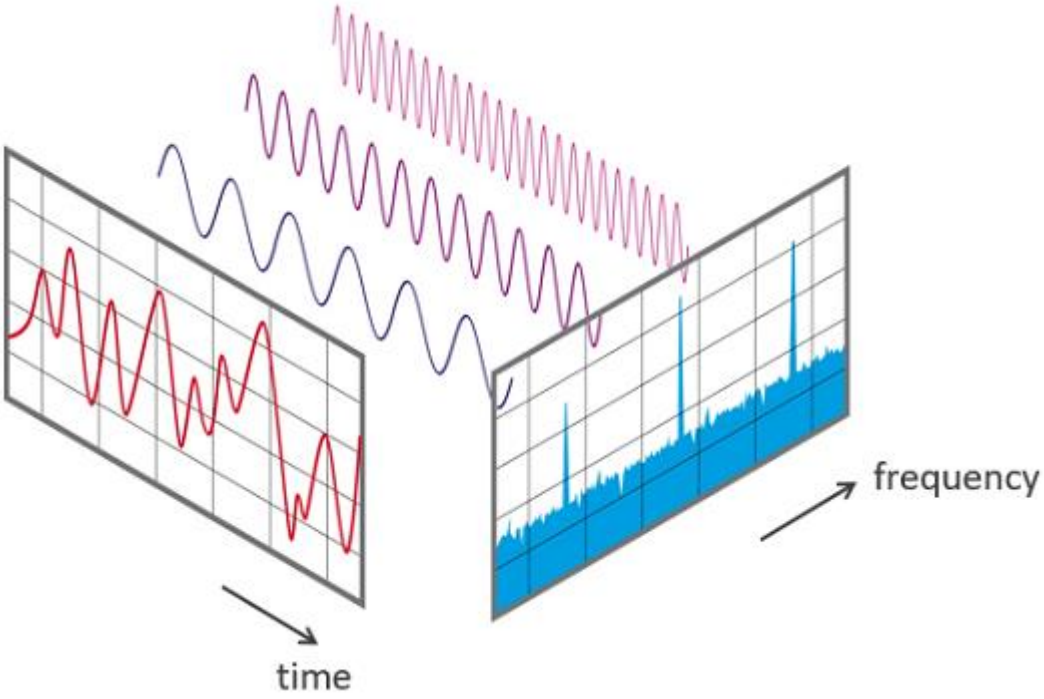

Рис. 1.3. Детальне представлення процесу перетворення FFT.

1.3.6. Короткочасне перетворення Фур’є

Short Fast Fourier transform – це алгоритм, який працює по схожому принципу зі звичайним Fast Fourier transform, але у ситуації безперервного часу, функція, яку потрібно перетворити, множиться на «віконну» функцію, яка є відмінною від нуля, але лише протягом визначеного проміжку часу.

Береться перетворення Фур’є (одновимірна функція) вихідного сигналу на проміжку, потім вікно «ковзає» вздовж вісі часу до кінця, що призводить до двовимірного представлення сигналу. Даний тип перетворення призводить до формування спектрограми, яка зображена на рисунку 1.4. На рисунку 1.5 наведений приклад звичайного відображення віконної функції на графіку сигналу.

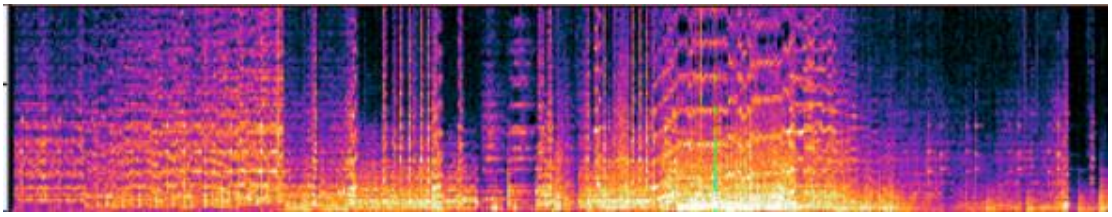

Рис. 1.4. Приклад спектрограми.

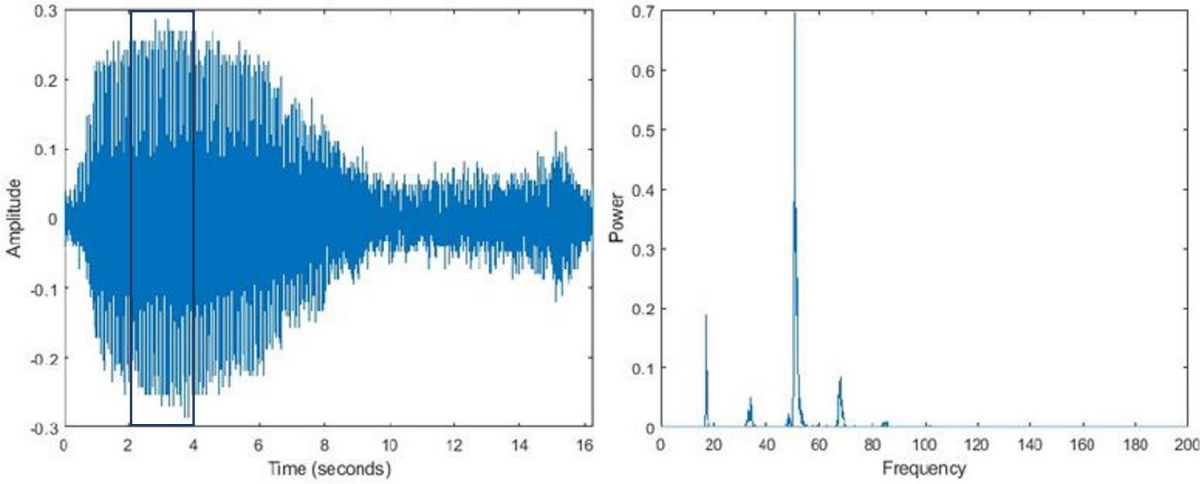

Рис. 1.5. Приклад швидкого перетворення Фур’є з віконною функцією довжиною 2 секунди

1.3.7. Wavelet перетворення

Розвиток даного перетворення і сфери його застосування є активною сферою досліджень. Дане перетворення запропонували геофізик Ж. Морле та фізик-теоретик А. Гроссманом [20]. Разом з колегою Ю. Мейером вони розробили основу для математичного представлення вейвлетів. На цьому етапі, етапі створення, вейвлети були повністю у сфері математики і більше зосереджувались на теоретичному представлені ніж на використанні у практиці [21]. Згодом, дослідники Добеші та Маллат змінили це визначивши зв’язок між вейвлетами та цифровою обробкою сигналів.

Наразі вейвлети використовуються в багатьох сферах, таких як:

- компресія даних

- обробка зображень
- спектральній оцінці

Вейвлет перетворення дуже схоже на перетворення Фур'є, але навіть більше на короткочасне швидке (Short Fast Fourier Transform) зі зміненою функцією перевірки якості. Головною відмінністю є те, що перетворення Фур'є розкладає сигнал у функції синусів та косинусів, вейвлет перетворення ж навпаки використовує функції, які локалізуються в дійсному та Фур'є просторі.

Вейвлет перетворення може бути представлене такою формулою[22]:

$$F(a,b) = \int_{-\infty}^{\infty} f(x)\,\psi^{*}_{(a,b)}(x)dx \qquad (1.1)$$

Незалежність від базисних функцій перетворення Фур'є призводить до отримання результатів, які описуються виключно в області частот. Опис сигналу як функції від часу або області частот є не дуже зручним варіантом, а в більшості ситуацій навіть є критичною проблемою в обробці сигналів. Людський слух покладається на часовий та частотний параметри одночасно для розпізнання та опису звуків.

Описана проблема наводить нас на думку, що перетворення Фур'є має багато недоліків для використання в стандартному випадку. У випадку аналізу стаціонарних сигналів [23] (сигналів, значення частот яких не змінюється з часом), перетворення Фур'є є ідеальним інструментом, але не зовсім підходить для обробки змінних сигналів. Це означає що для аналізу не стаціонарних сигналів [24] потрібна функція, яка може перетворювати сигнал не тільки в частотну, а в частотно-часову область.

Дана ціль може бути досягнута з використанням Short Fast Fourier Transform, який буде вираховувати частотну область на певній ділянці часу, що зробить можливим розраховувати частотний спектр для малих проміжків часу

[25] та в перспективі сформувати велику спектрограму залежності частоти від гучності сигналу для усіх відрізків часу отриманих діленням віконною функцією.

Якщо в STFT результат перетворення можна сприймати як результат виконання перетворення Фур'є віконного сигналу, то також його можна представити у вигляді декомпозиції сигналу в віконному базисі функцій. Базисні функції – набір функцій, при об'єднанні в якості зваженої суми яких можуть бути використані для конструювання наданого сигналу. У випадку STFT базисними функціями є синусоїди, у випадку з Wavelet transform базисними функціями є вейвлети. Приклад вейвлетів, які можуть бути використані наведено на рисунку 1.6.

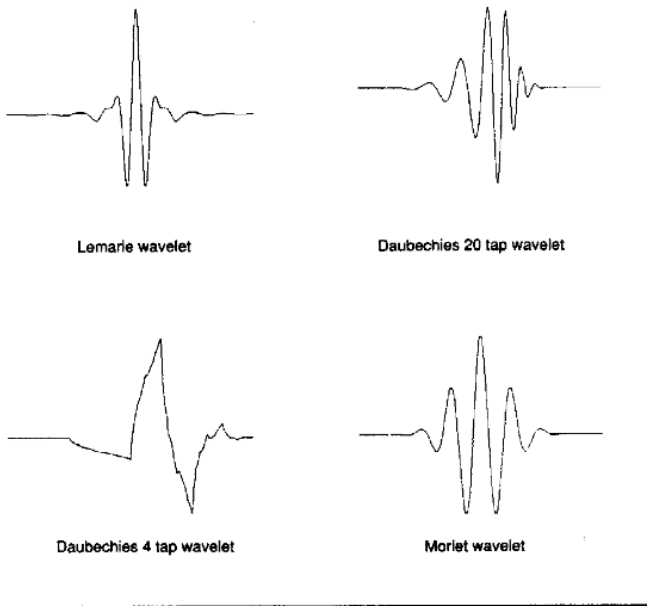

Рис. 1.6. Приклад вейвлетів

У випадку з Wavelet transform функція частоти ω замінюється коефіцієнтом масштабу a та τ замінюється коефіцієнтом який позначає подовженість вейвлету у часі – b. Приклад зміни вейвлетів згідно зі зміною коефіцієнтів наведено на рисунку 1.7.

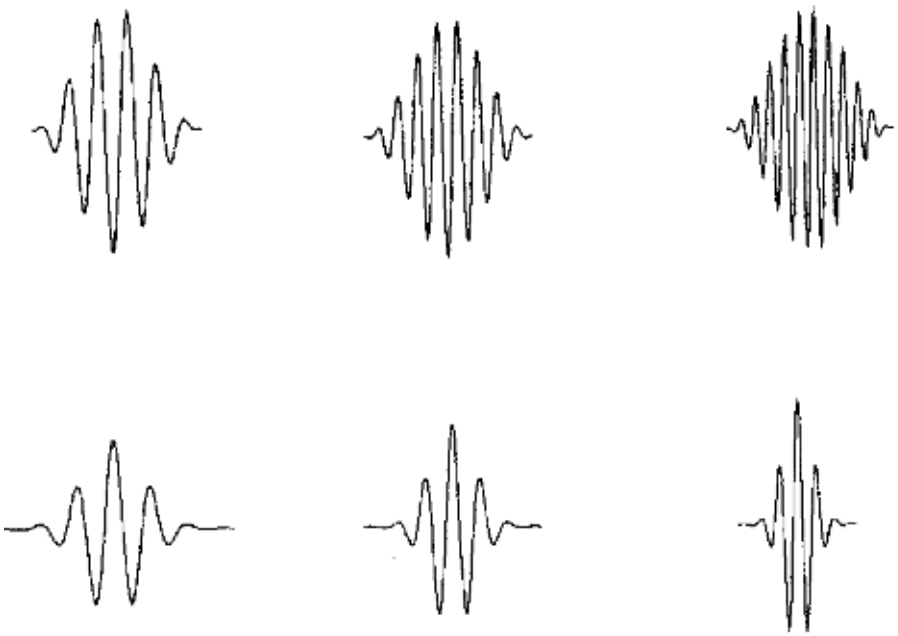

Рис. 1.7. Приклад впливу коефіцієнтів на вейвлет

Даний тип перетворення є дуже популярним і потенційно найбільш ефективним в частотно-часовому представленні сигналу, але досі є достатньо не популяризованим серед основних підходів до обробки мовлення у порівнянні зі спектрограмами [26], які отримуються завдяки Short Fast Fourier Transform.

1.4. Постановка задачі

В результаті розгляду різних підходів до синтезу мовлення та різних методів обробки сигналів було обрано до використання метод синтезу заснований на глибинних мережах та короткочасне швидке перетворення Фур’є як метод обробки сигналу.

У даній кваліфікаційній роботі потрібно розв'язати задачі:

1. Обрати модель синтезу мовлення.
2. Удосконалити модель можливістю навчання на українському наборі даних.

3. Навчити модель синтезу штучного мовлення зі збереженням характеристик голосу мовця.
4. Провести оцінку результатів навчання.
5. Візуалізувати результати у вигляді графіків.

1.5. Висновки

Були розглянуті різні підходи до синтезу мовлення: артикуляційний, форматний, конкатенативний синтез та синтез заснований на нейронних мережах. Також були розглянуті та порівняні різні перетворення в обробці сигналів: перетворення Фур’є, короткочасне перетворення Фур’є та вейвлет перетворення. Завдяки аналізу цих підходів, було поставлено задачу кваліфікаційної роботи.

РОЗДІЛ 2

ВИЗНАЧЕННЯ МОДЕЛІ СИНТЕЗУ МОВЛЕННЯ ТА МОДИФІКАЦІЯ

2.1. Перегляд аналогів обраної моделі

В обраному методі синтезу мовлення на базі глибинної мережі виділяють деякі основоположні моделі синтезу, а саме:

- Tacotron
- Fast-Speech
- Glow-TTS

В кожній з цих моделей є різні версії з модифікацією параметрів характерних для ситуацій для яких модель була модифікована. В деяких випадках це може бути якась характерна проблема, яка виходить зі структури моделі, але в багатьох реалізаціях, версії які є відмінними від канонічної різняться способом навчання (частіше за все це спрощення самого процесу навчання) та збільшенням якості звуку.

Tacotron був представлений компанією Google в березні 2017 року як наскрізна система. Даний тип моделі використовує архітектуру кодувальника-декодера та механізм уваги [27]. Ціллю кодувальника в даній моделі є отримання представлення тексту для подальшого навчання, а декодера в розшифровування отриманого результату спектрограми. В якості декодера зазвичай використовують Griffin-Lim алгоритм. Оцінкою MOS для Tacotron на англійському наборі даних було значення 3.82.

Нова версія Tacotron додала багато покращень в первісну модель, а саме [28]:

- покращений механізм уваги
- замість Griffin-Lim алгоритму використовується WaveNet

- замість лінійчастих спектрограм почали використовувати мел-спектрограми

Нова версія Tacotron має оцінку MOS 4.53 на тому ж наборі даних.

Не так давно був представлений підхід створення моделей синтезу мовлення з використанням перетворювачів. Раніше вони використовувались повсюдно в NLP (Natural Language Processing), тому в даній сфері їх поява була передбачена. Використання мовних синтезаторів на базі перетворювачів було націлено на розв'язання таких проблем як

- довгий період навчання та формування мовлення
- важкість навчання через велику зв’язність рекурентних нейронних мереж

Незабаром була представлена модель Fast-Speech, яка була виконана на базі перетворювачів та дала можливість прискорити існуючі моделі у декілька десятків разів [29]. Основними плюсами архітектури нової моделі були:

- паралельна генерація мел-спектрограм
- збільшена точність визначення залежності конкретної фонеми від її спектрограми
- доданий регулятор швидкості голосу, який дав можливість змінювати довжини окремих фонем.

2.2. Використання Griffin-Lim алгоритму в якості вокодеру

Griffin-Lim алгоритм – це метод фазової реконструкції, який заснований на надлишковості перетворення Фур’є [30].

Алгоритм відтворює спектрограму комплексних значень, використовуючи формулу:

$$\boldsymbol{X}^{[\boldsymbol{m+1}]} = \boldsymbol{P_C}(\boldsymbol{P_A}(\boldsymbol{X}^{[\boldsymbol{m}]})) \quad (2.1)$$

Де змінні це:

- $\boldsymbol{X}$ – спектрограма з комплексними значеннями, яка оновлюється ітеративно,
- $\boldsymbol{P_A}$ – проекція на множину $\boldsymbol{A}$,
- m – індекс поточної ітерації,
- $\boldsymbol{A}$ – множина спектрограм

А $\boldsymbol{P_C}$ розраховується за формулою [31]:

$$\boldsymbol{P_C}(\boldsymbol{X}) = \boldsymbol{G}\boldsymbol{G}^*\boldsymbol{X} \quad (2.2)$$

Де $\boldsymbol{G}$ – це значення Short Fast Fourier Transform, а $\boldsymbol{G}^*$ - це псевдо обернене значення STFT.

2.3. Математичне представлення перетворення Фур'є.

Перетворення Фур'є має вигляд:

$$f(\xi) = \int_{-\infty}^{\infty} f(x) \cdot e^{-i2\pi\xi x} dx \quad (2.3)$$

Де ξ - це значення частоти.

Інтегрування для всіх значень ξ дає можливим створення функції частотного спектра.

Для дійсної функції виконується властивість симетрії, тобто $y(-\xi) = y^*(\xi)$. В якій $y^*(\xi)$ - це комплексне сполучення $y^{(\xi)}$. (Комплексне число реальна частина якого дорівнює початковій функції, а уявна частина дорівнює за модулем, але має зворотний знак).

Виходячи з цього можна представити формулу 1 як систему:

$$f(\xi) = \begin{cases} \int_{-\infty}^{\infty} f(x) e^{-i2\pi\xi x}\, dx, & \xi \geq 0 \\ \hat{f}(|\xi|), & \xi < 0 \end{cases} \tag{2.4}$$

У випадках, коли ми маємо діло зі скінченною кількістю x, перетворення Фур'є можна представити у вигляді суми:

$$f(\xi) = \sum_{n=0}^{N-1} f(n) e^{-i2\pi\xi x} \tag{2.5}$$

Де N – кількість елементів вектора x.

2.4. Математичне представлення короткочасного перетворення Фур'є.

В якості основного алгоритма перетворення звукової хвилі на спектр частот в рамках роботи використовується короткочасне перетворення Фур'є. Даний алгоритм виступає модифікацією звичайного алгоритму Фур'є.

Короткочасне перетворення Фур'є має вигляд:

$$F(\tau, \xi) = \int_{-\infty}^{\infty} f(x) * w(x - \tau) e^{-i2\pi\xi x} dx \tag{2.6}$$

Відмінність даної формули від формули 2.1 полягає в присутності «віконної» функції w. Дана функція дає можливість отримувати значення частотного спектра на певному часовому проміжку. Здатність отримувати частотний спектр на заданому проміжку робить можливим боротьбу з принципом невизначеності Гейзенберга.

Принцип Гейзенберга виражається в даній формулі:

$$\Delta x \Delta p \geq \frac{h}{4\pi} \tag{2.7}$$

Де Δx – не визначеність в позиції, Δp – не визначеність моментуму, а h – планкова константа (6.626×10^{-34}).

В загальному випадку, принцип невизначеності Гейзенберга криється в неможливості виміряти з гарною точністю дві характеристики об’єкта, який рухається в часі. В оригінальному формулюванні йдеться про позицію об’єкта у просторі та його швидкість, але в ситуації з обробкою сигналів невизначеність полягає у не можливості визначення частоти за одиницю часу. (Частотна характеристика унеможливлює отримання інформації про її зміну з часом).

Для того, щоб уникнути цієї проблеми – використовується короткочасне перетворення Фур’є, яке вимірює частоти на певному проміжку. В машинному навчанні заведено називати частини, на які розбивається звукова хвиля – batch, а даний процес – batching. Для процесу батчингу використовується заздалегідь визначена константа на величину якої розбивається звукова хвиля. Уявімо що частота дискретизації дорівнює 44100, та довжина батча 32. Це означає, що для 1 секунди звукової хвилі ми будемо мати 1378 батча для обробки.

2.5. Glow-TTS як модель для синтезу

Glow-TTS – це одна з новіших моделей, яка є паралельною моделлю синтезу мовлення та зроблена для максимізації ефективності навчання та синтезу мовлення.

Паралельна модель декодування – це така модель, яка проводить декодування моделі послідовності до послідовності (sequence to sequence) у паралельному режимі.

Модель послідовність до послідовністю (sequence to sequence) – це особливий клас архітектури рекурентної нейронної мережі, які використовуються для вирішення задач в обробці природньої мови (natural language processing) [32].

Рекурентна нейронна мережа – це такий вид нейронної мережі, який містить прихований стан h та не обов'язкове вихідне значення y, яке працює на послідовності змінної довжини $\boldsymbol{x} = (\boldsymbol{x_1}, \dots, \boldsymbol{x_T})$. На кожному окремому проміжку часу t прихований стан $\boldsymbol{h_{\langle t \rangle}}$ рекурентної нейронної мережі розраховується за формулою 2.8:

$$h_{\langle t \rangle} = f\left(h_{\langle t-1 \rangle}, x_t\right), \tag{2.8}$$

Де $\boldsymbol{f}$, це не лінійна функція активації, яка може бути простою як функція сигмоїди або функцією набагато складнішої структури.

Приклад діаграм навчання та синтезу наведено на рисунку 2.1 та 2.2 відповідно.

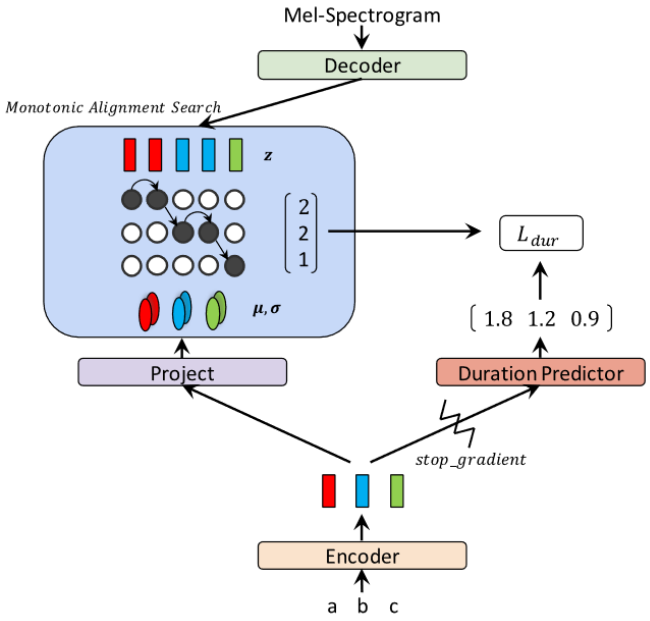

Рис. 2.1. Діаграма навчання моделі Glow-TTS [33]

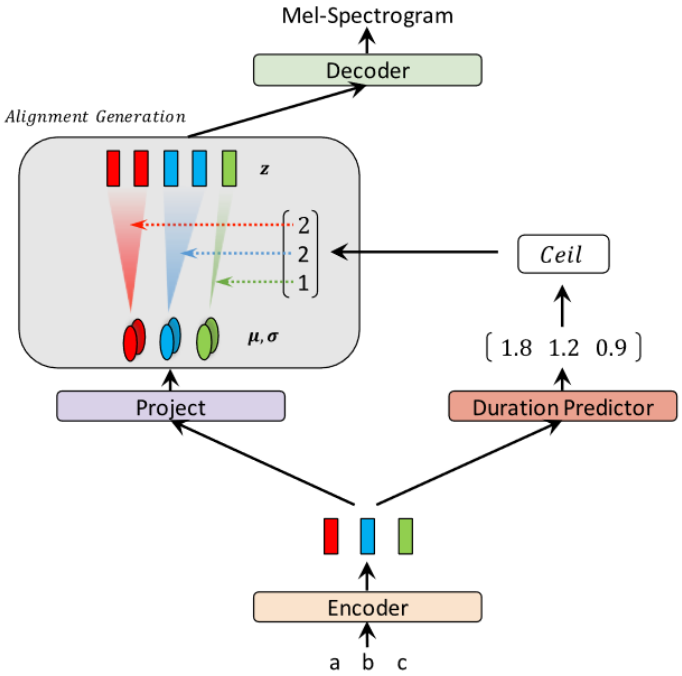

Рис. 2.2. Діаграма синтезу мовлення Glow-TTS [33]

Однією з основних частин Glow-TTS є декодер. Його задачею є перетворення мел-спектрограми на її латентне представлення під час навчання та вміти перетворювати попередній розподіл у розподіл мел-спектрограми для паралельного декодування.

Маючи мел-спектрограму $\boldsymbol{x}$, Glow-TTS перетворює її на латентну змінну z (змінна, яка виводиться за допомогою мат. моделі) використовуючи декодування основаному на потоку $\boldsymbol{f_{dec} : x \rightarrow z}$ без використання будь-якої текстової інформації. Змінна z відповідає певному ізотропному(однаковий у будь-якому напрямку) розподілу Гауса.

Далі, текстовий кодувальник $\boldsymbol{f_{enc}}$ перетворює базовий стан тексту $\boldsymbol{c}$ до високорівневого представлення тексту $\boldsymbol{h}$ та відображає $\boldsymbol{h}$ в статистиці, $\boldsymbol{\mu}$ та $\boldsymbol{\sigma}$ Гаусівського розподілу**.** Таким чином, кожна текстова послідовність має відповідний розподіл та кожна частина латентної змінної $\boldsymbol{z_j}$ відповідає одному з цих розподілів, який був передбачений умовами кодувальника.[33]

2.5.1. Ефективність Glow-TTS

Для того щоб зробити висновок яку модель покращувати та навчати синтезувати українське мовлення було проведено дослідження щодо MOS та часу формування вихідного результату.

MOS – це середнє значення оцінки, яке набуло дуже великої популярності в якості індикатора сприйняття якості медіа[34]. Середня оцінка має значення від 1 до 5, що означає жахливу та ідеальну якість відповідно. MOS розраховується як середнє арифметичне одиничних суб’єктивних оцінок за формулою:

$$MOS = \frac{\sum_{n=1}^{N} R_n}{N}, \tag{2.9}$$

Де $\boldsymbol{R_n}$ – це окрема оцінка n-ного медіа.

Результати були взяті з моделей, які вчились на одному мовці жіночої статі набору даних LJSpeech. Кількість аудіо треків дорівнює 13100 з сумарною довжиною близько до доби (24 години). Набір даних рандомізовано ділився на 3

частини: для навчання, перевірки (validation крок) та тестування, що є стандартним підходом до навчання. Значення MOS наведено в таблиці 2.1.

Таблиця 2.1

Оцінки MOS [34]

Метод	**Середнє значення оцінки**
Оригінал	$\mathbf{4.54 \pm 0.06}$
Оригінал (Mel + WaveGlow)	$\mathbf{4.19 \pm 0.07}$
Tacotron2 (Mel + WaveGlow)	$\mathbf{3.88 \pm 0.08}$
Glow-TTS (Mel + WaveGlow)	$\mathbf{4.01 \pm 0.08}$

В таблиці 2.1 можна побачити що оцінка моделі Glow-TTS краще за оцінку Tacotron2, не звертаючи увагу на можливу дельту, але все ж гірша за значення оригіналу, навіть при кращій дельті, який отримали завдяки оцінки результату конструювання його мел-спектрограм з використанням WaveGlow.

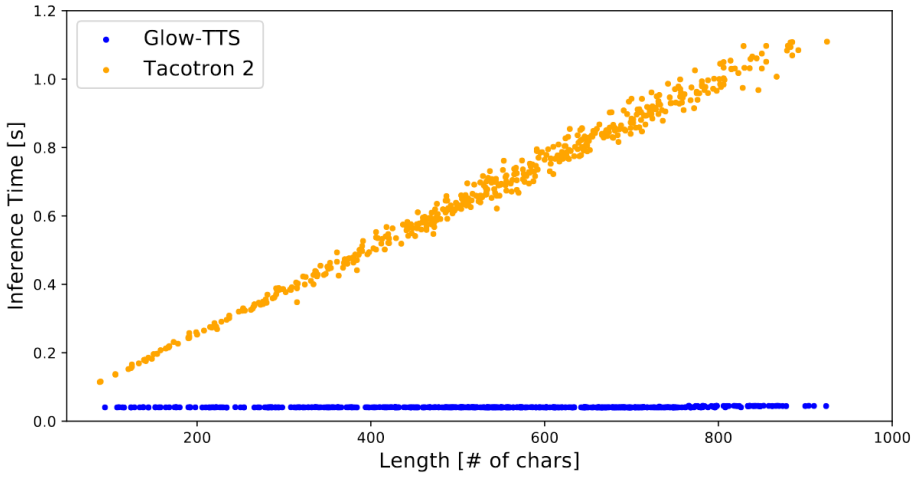

Рис. 2.3. Залежність часу генерації мовлення від довжини тексту [33]

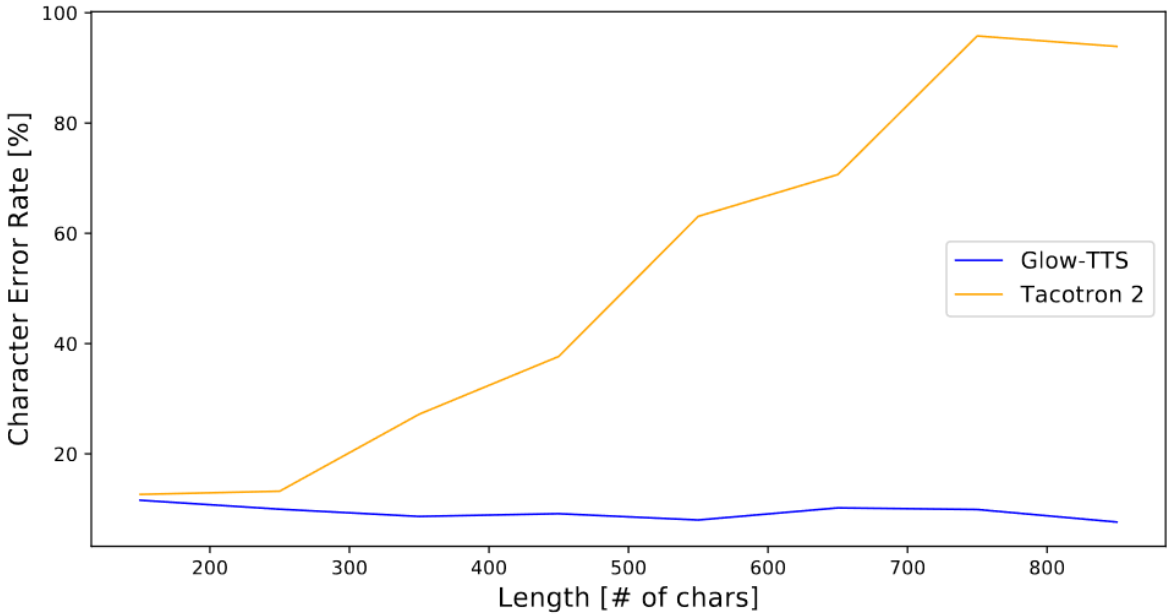

Рис. 2.4. Залежність CER від довжини слова [35]

На рисунку 2.3 наведений графік залежності часу генерації мовлення від довжини слова для якого утворюється мел-спектрограма. Як можна побачити у випадку Tacotron2 кількість часу, потрібного для генерації мовлення лінійно зростає в той час, як значення часу, для моделі Glow-TTS, потрібного для генерації спектрограми, при збільшені кількості символів не змінюється.

На рисунку 2.4 наведена залежність CER від довжини слова. CER моделі Tacotron 2 починає стрімко зростати, коли довжина вхідного тексту перевищує 260, а модель Glow-TTS навпаки спадає з самого початку вимірювання майже на усьому проміжку функції.

CER (частота помилкових символів) – це метрика, яка зазвичай використовується у автоматичних системах розпізнавання, або синтезу мовлення.

Дана метрика передає процентну кількість хибно передбачених літер, це означає що в ефективних моделях дана метрика повинна прагнути до нуля, що вважається ідеальною ситуацією [35]. Метрика розраховується за наступною формулою:

$$CER = \frac{S + D + I}{N} \quad (2.10)$$

Де змінні це:

- $\boldsymbol{S}$ – кількість замінених символів,
- D – кількість видалених символів,
- I – кількість доданих символів,
- C – кількість правильних символів,
- N – загальна кількість символів, яку можна уявити як ($\boldsymbol{N = S + D + C}$).

2.5.2. Модифікація Glow-TTS

Базова реалізація моделі Glow-TTS не підтримує українську мову. Зазвичай такі моделі існують у відриві від мов, або з класичною реалізацією використовуючи англійську мову яка відрізняється від мов світу.

Для того щоб зробити можливим навчання моделі Glow-TTS використовуючи графеми мови потрібно було додати підтримку українських літер, тобто змінити базову конфігурацію моделі, в яку додати словник потрібних літер в правильному кодуванні. До того ж потрібно змінити базовий форматер для можливості підтримки довільної структури набору даних та переписати текстовий очищувач, який буде правильно робити нормалізацію тексту, видаляти зайві символи та робити класичне форматування тексту яке характерне для набору даних української мови.

2.6. Висновки

В цьому розділі були розглянуті моделі синтезу мовлення, які базуються на глибинних нейронних мережах, такі як: Tacotron2, Fast-Speech та Glow-TTS та виділена модель для подальшого дослідження та навчання (Glow-TTS).

Також була розглянута ефективність обраної моделі порівняно з популярною реалізацією моделі трансляції до мовлення та запропонована модифікація моделі для додання можливості синтезу українського мовлення.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ МОДЕЛІ GLOW-TTS ТА ЕКСПЕРИМЕНТИ З МОДЕЛЛЮ СИНТЕЗУ

3.1. Деталі реалізації мовленнєвого синтезатора.

Основна ідея навчання нейронної мережі полягає у тому щоб дати їй можливість визначити необхідні значення ваг у кожного зв’язку між нейронами на кожному шару створеної моделі.

Головною ідеєю в створенні моделі трансляції тексту до українського мовлення є створення:

- Селектору мовних властивостей
- Акустичної моделі.
- Вокодеру.

Припустимо що фільтрація необхідної інформації в наборі даних, форматер вхідного тексту та інші частини коду були включені до одної з вище вказаних частин реалізації. Припущення робиться через те що форматування тексту може бути проведено неопосередковано під час селекції мовних властивостей, або вже при зчитуванні даних для навчання акустичної моделі.

Селектор мовних властивостей працює з текстом, який отримується з набору даних. Ця частина синтезатора мовлення зазвичай відповідає за правильну фільтрацію та форматування тексту, виокремлення фонем з кожної одиниці речення набору даних.

Акустична модель відповідає за правильне формування спектрограми використовуючи текст, який був оброблений селектором мовних властивостей. В акустичній моделі використовують один з алгоритмів для перетворення сигналу в частотний спектр для подальшого навчання виходячи зі значень

певних частот на проміжку спектра. Зазвичай в цій ситуації використовується короткочасне перетворення Фур’є.

При дослідженні даної теми було визначено що в якості аналогу даного алгоритму також може використовуватись Wavelet Transform.

Вокодер, глобально, це категорія інструментів, яка допомагає в кодуванні мовлення та створювався для того щоб допомагати в синтезі людської мови [36]. В моделях синтезу людського мовлення він використовується для перетворення спектрограм на звукові хвилі. Зазвичай в якості вокодеру виступає окрема нейронна мережа, яка вчиться, на тому ж наборі даних, що й акустична модель. Результат перетворення передається у модуль відображення або зберігається в файл.

Також існує шлях, який полегшує навчання моделей синтезу. Основною ідеєю якого є спрощення вхідних даних акустичної моделі, тобто використання графем замість фонем. Також полегшенням може слугувати використовування Griffin-Lim алгоритму в якості вокодеру. Даний алгоритм базується на надмірності короткочасного перетворення Фур’є та робить можливим перетворення спектрограм на звукову хвилю без попереднього навчання нейронної мережі.

3.2. Google Colab як платформа для виконання коду

Першою проблемою, яка постає перед будь-якою людиною, яка хоче почати навчати нейронну мережу – це необхідність мати достатньо високопродуктивну систему, яка зможе відносно в короткі терміни продемонструвати результат навчання.

Теоретично, нейронні мережі можуть навчатись як на GPU так і на CPU, але під час дослідження було виявлено на прикладі платформи Google Colab, що ефективність центрального процесору поступається ефективності графічного процесору майже в 80 разів.

Дослідження проведені на ноутбуці Omen 15 з графічним процесором на базі мобільної відеокарти NVIDIA Geforce GTX 1660 Ti та центральним процесором AMD Ryzen 7 4800H показали що ефективність відеокарти більше за ефективність процесору в 3-4 рази.

Не дивлячись на те, що експеримент проходив на безкоштовній підписці Google Colab навчання на базі графічного процесору проходило швидше ніж на GTX 1660 Ti в 2-3 рази, це дало розуміння, що 1660 Ti у сфері навчання нейронних мереж є не продуктивною ідеєю і тому для дослідження була використана підписка Google Colab Pro для навчання.

Характеристики усіх наявних підписок платформи Google Colab наведено в таблиці 3.1.

Таблиця 3.1

Підписки Google Colab

	Google Colab Basic	Google Colab Pro	Google Colab Pro+
Ціна	0$	9.99$	49.99$
Обчислювальні одиниці	0	100	500
GPU	**Tesla K80**	Tesla P100	Tesla P100
RAM	**12Gb**	32Gb	52Gb

Як можна зробити висновок з підписок Google Colab, зображених в таблиці 4 – близько 12 гігабайт оперативної пам’яті – мінімальний поріг необхідний для навчання. До того ж в базовій версії використовується відеокарта NVIDIA Tesla K80, яка є достатньо продуктивною картою.

Щодо проблем з якими прийдеться стикнутись розробнику на даній платформі – це не можливість навчання більше 12 годин на день (в базовій версії підписки), швидкість витрачання обчислювальних одиниць в середньому 8 од/год, що виснажує баланс підписки Google Colab Pro повністю вже через 12.5

годин, що автоматично переключає користувача на графічний процесор базового рівня. Іншими словами, 1 година виконання коду коштує 80 центів на платформі Colab. На рисунку 3.1 був наведений програмний інтерфейс розглянутої платформи.

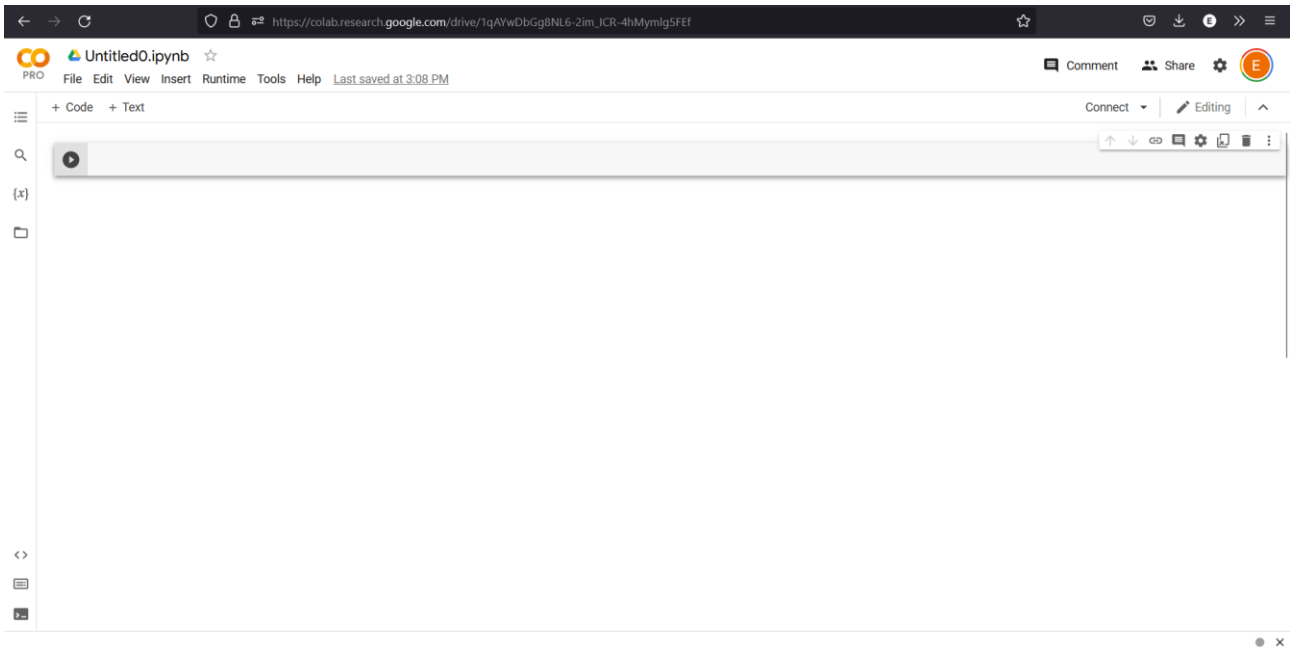

Рис. 3.1. Інтерфейс Colab в базовому проекті.

Інтернет платформа Colab використовує структуру файлів та файловий формат, який був розроблений у проекті з відкритим вихідним кодом – Project Jupyter [37].

Даний формат документів був спочатку популярним у сферах, де потрібно було поєднувати виконуваний код із додаванням висновків або примітками. Основною сферою використання, на початку, було обчислювальне математичне програмне забезпечення, прикладами яких можуть слугувати Maple або Mathematica.

До розробки даного проєкту користувачі повинні були мати справу з багатьма файлами, які викликались в терміналі, що робило розробку достатньо проблематичною, бо такий підхід до розробки міг бути практичним тільки в

малих проектах, які не потребували логічного винесення частини коду в окремий блок.

Використання ж Project Jupyter дало ж змогу розробникам виділяти окремі блоки коду та виконувати їх в одному скоупі та робити опис окремих блоків в тому ж файлі без необхідності виносити пояснення в коментарі, що неабияк покращило якість коду.

3.3. Keras як бібліотека для розробки з глибинних нейронних мереж

Keras – бібліотека, яка надає надзвичайно потужні складові абстракцій для побудови нейронних глибинних мереж [38].

Складові бібліотеки Keras створюються з використанням Theano та Tensorflow. Бібліотека підтримує обчислення як на центральному процесорі так і на графічному і дає можливість швидко приступати до розробки прототипів майбутньої мережі.

Головними етапами в створенні моделі глибинного навчання є[39]:

- Визначення моделі, тобто створення базової Sequential model та додання до неї шарів характерних до вирішуємої задачі.
- Компіляція моделі, визначення функції втрат та оптимізатору перед безпосереднім викликом методу compile
- Виконання процесу «фіттінгу» даними, які були попередньо оброблені. Даний процес запускається викликом методу fit()
- Виконання передбачень. Використання навченої моделі для створення передбачень базуючись на нових даних, викликаючи фукнцію evaluate() або predict()

Якщо узагальнити усі кроки створення моделі використовуючи Keras, то це виконання наступних дій:

1. Завантаження даних до програми.
2. Форматування даних.

3. Визначення моделі.
4. Компіляція моделі.
5. «Фіттінг» моделі.
6. Виконання навчання.
7. Передбачення моделлю.
8. Збереження навченої моделі.

Після виконання усіх вище описаних дій та проходження усіх поколінь на шостому етапі зі збереженням навченої моделі можна вважати модель навченою.

Tensorflow – це система машинного навчання, яка працює з великім обсягом інформації та в гетерогенних середовищах.

Гетерогенне середовище – це складна система, яка будується завдяки принципу взаємодії різноманітних програмних та апаратних платформ.

TensorFlow підтримує створення різноманітних додатків, але особливо націлений на навчання та виведення за допомогою глибинних нейронних мереж. Він використовується як платформа для досліджень і розгортання систем машинного навчання в сферах [40]:

- Розпізнання мовлення
- Комп’ютерний зір
- Робототехніка
- Пошук інформації
- Обробка природньої мови

Також, в кваліфікаційній роботи був використаний набір інструментів Coqui-TTS, з відкритим вихідним кодом для роботи з глибинним навчанням у сфері синтезу людського мовлення.

3.4. Підготовка набору даних

Для створення моделі трансляції тексту до мовлення був використаний набір даних, який має назву «Ukrainian Open Speech To Text Dataset».

Як можна зрозуміти з назви, даний датасет був створенний для того щоб дати можливість створювати моделі трансляції мовлення до тексту, але при правильному перетворенні він може бути використаний в якості набору даних для навчання моделей трансляції з тексту до мовлення.

В даному наборі даних міститься одразу велика кількість різних аудіо файлів від багатьох мовців різного віку, статі та різної якості запису, саме тому було винайдено найбільший набір даних який був озвучений одним мовцем та виокремлений к загального набору. Цей дата сет був набором мовлення з української біблії, автора якого на жаль знайти не вдалось.

3.5. Тренування моделі Glow-TTS

Для тренування моделі на першому етапі був використаний Google Colab, який було розглянуто раніше, але після не тривалого проміжку часу модель, яка навчалась була перенесена на локальний комп’ютер з встановленою NVIDIA RTX 3060.

На рисунках 3.2-3.5 продемонстрована частина селектору мовних властивостей, який був реалізований з використанням платформи Google Colab.

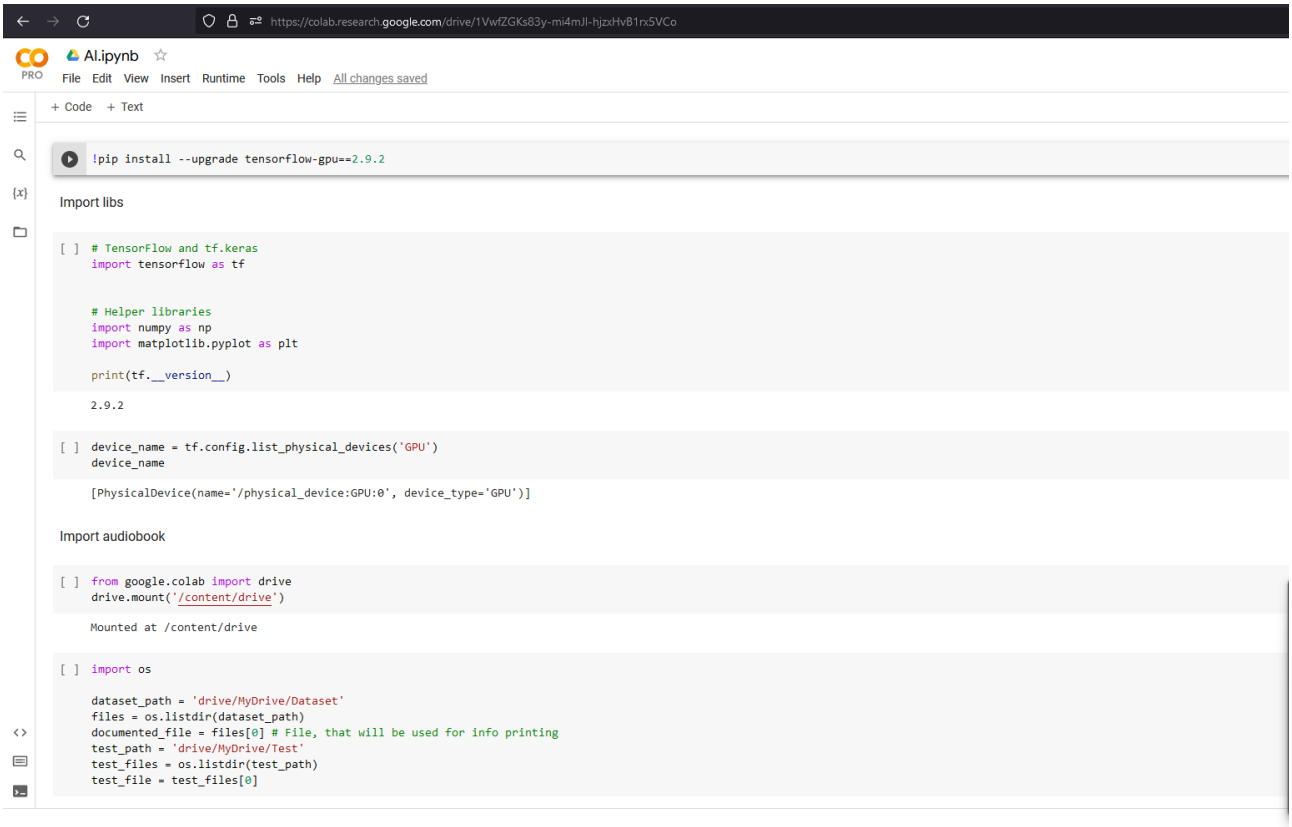

Рис. 3.2. Проєкт у Google Colab

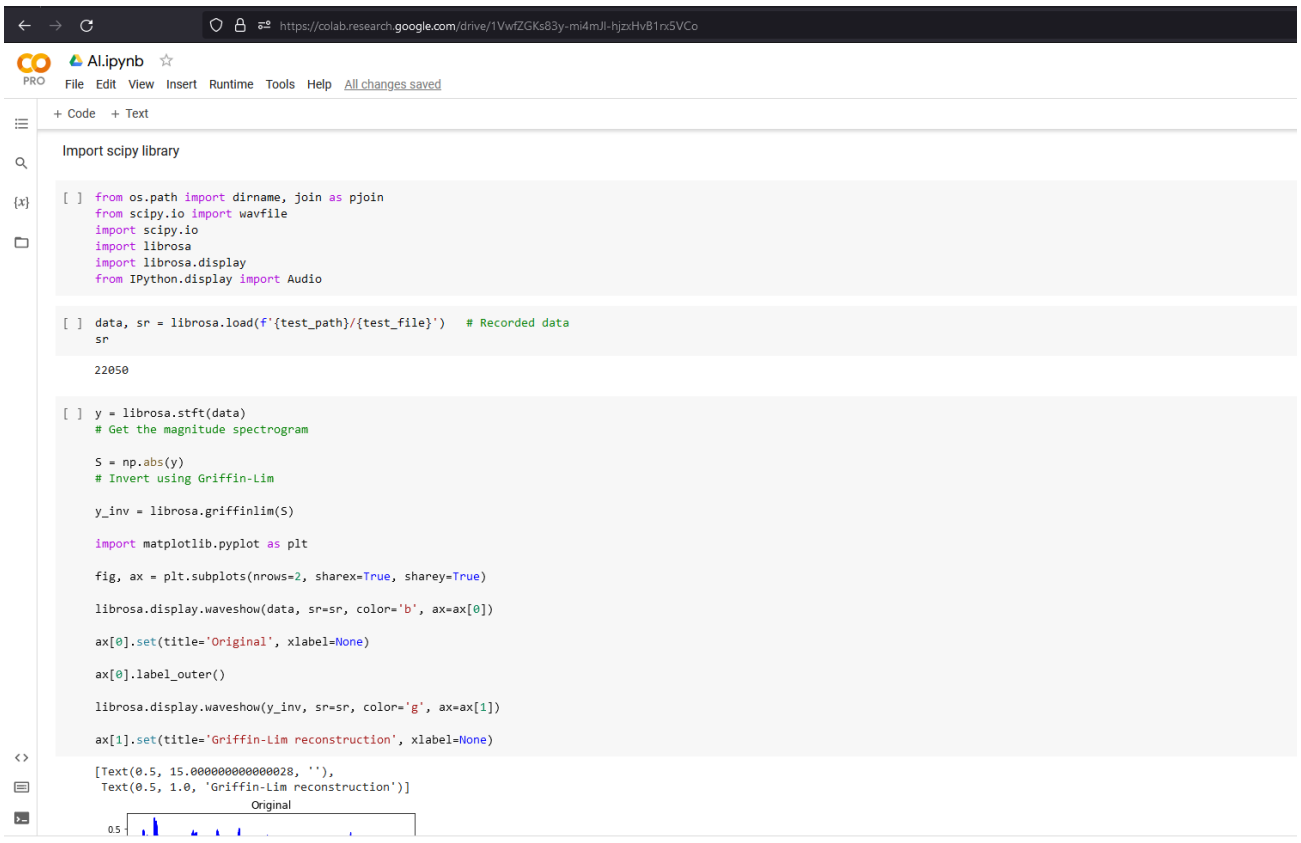

Рис. 3.3 Проєкт у Google Colab

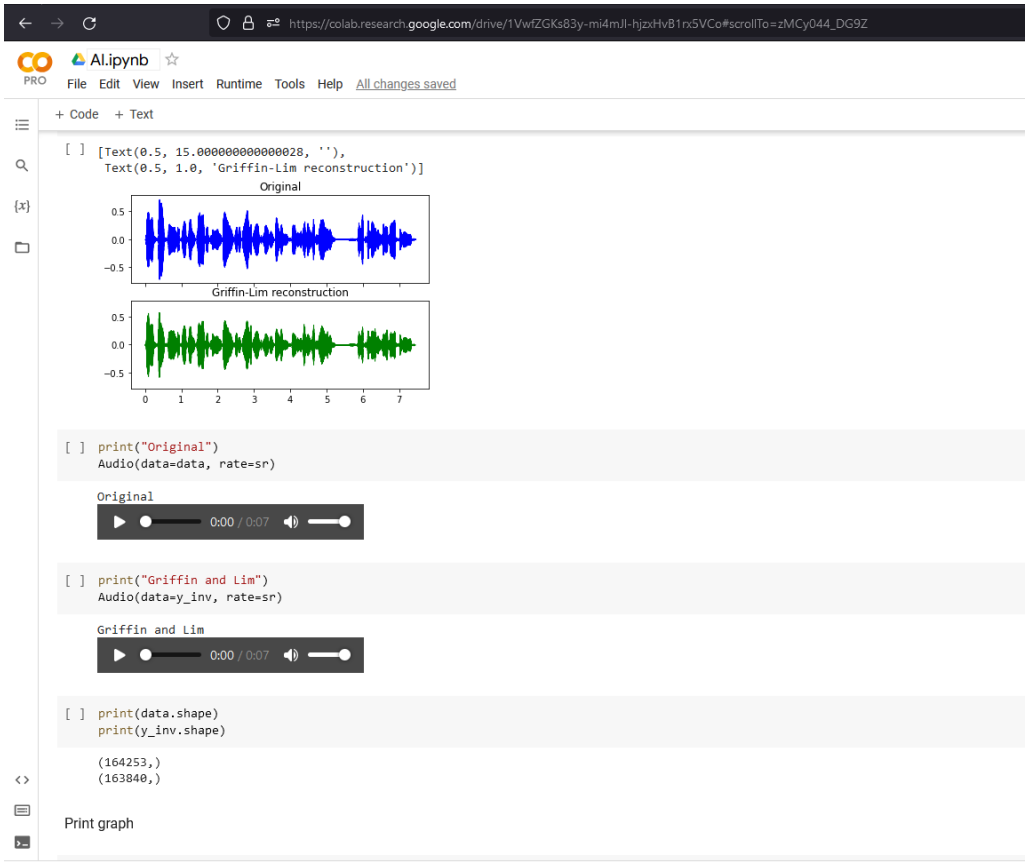

Рис. 3.4. Проєкт у Google Colab

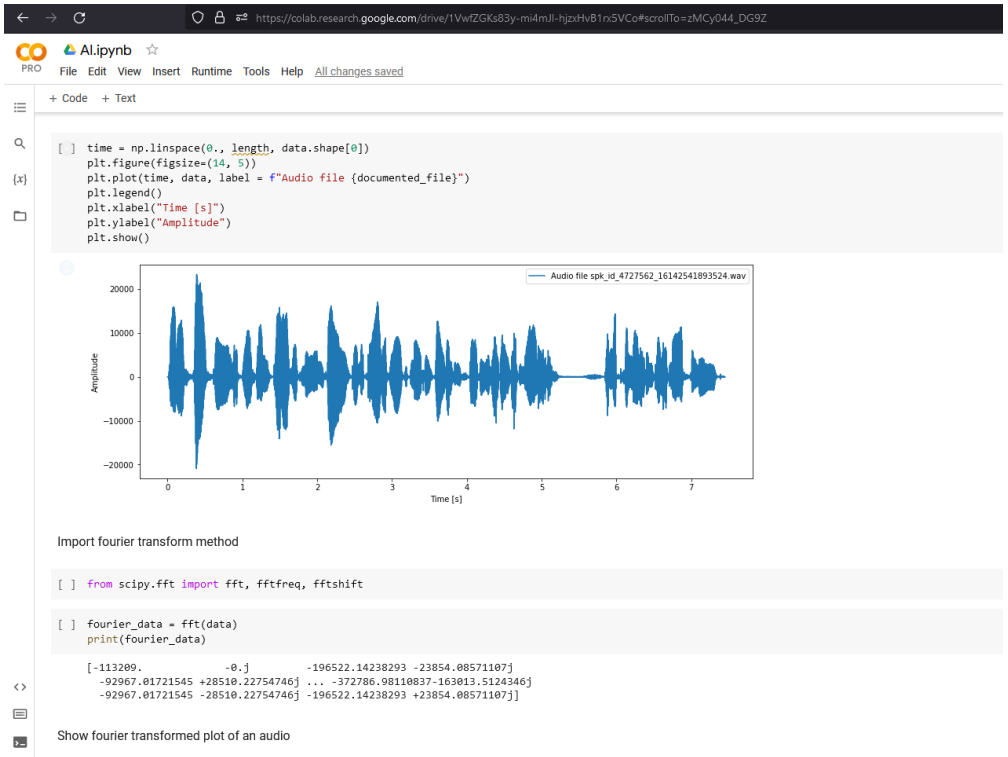

Рис. 3.5. Проєкт у Google Colab

3.6. Результати навчання моделі трансляції тексту до мовлення

На рисунках 3.6-3.15 наведені приклади Ground truth спектрограм, графіків механізмів уваги та передбачувань на різних етапах формування моделі.

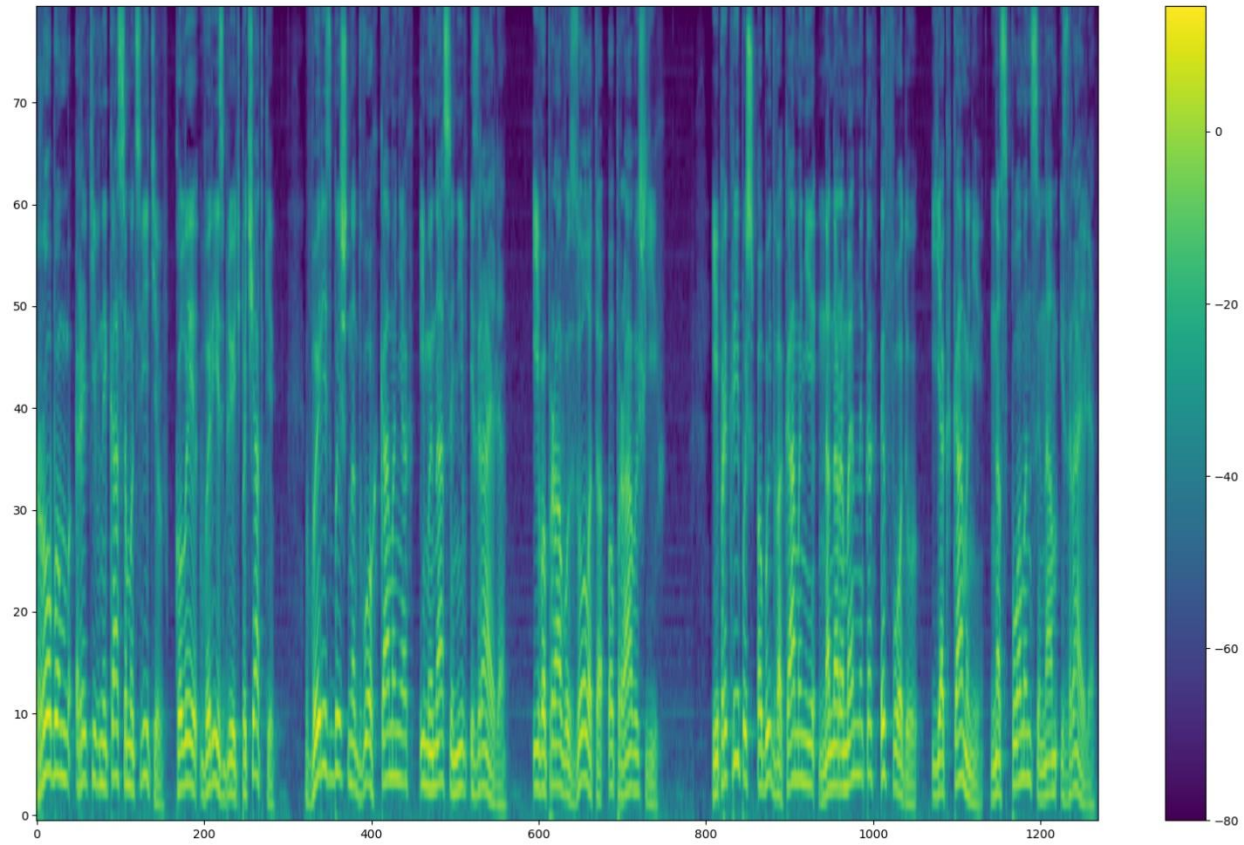

Рис 3.6. Спектрограма Ground truth на етапі перевірки

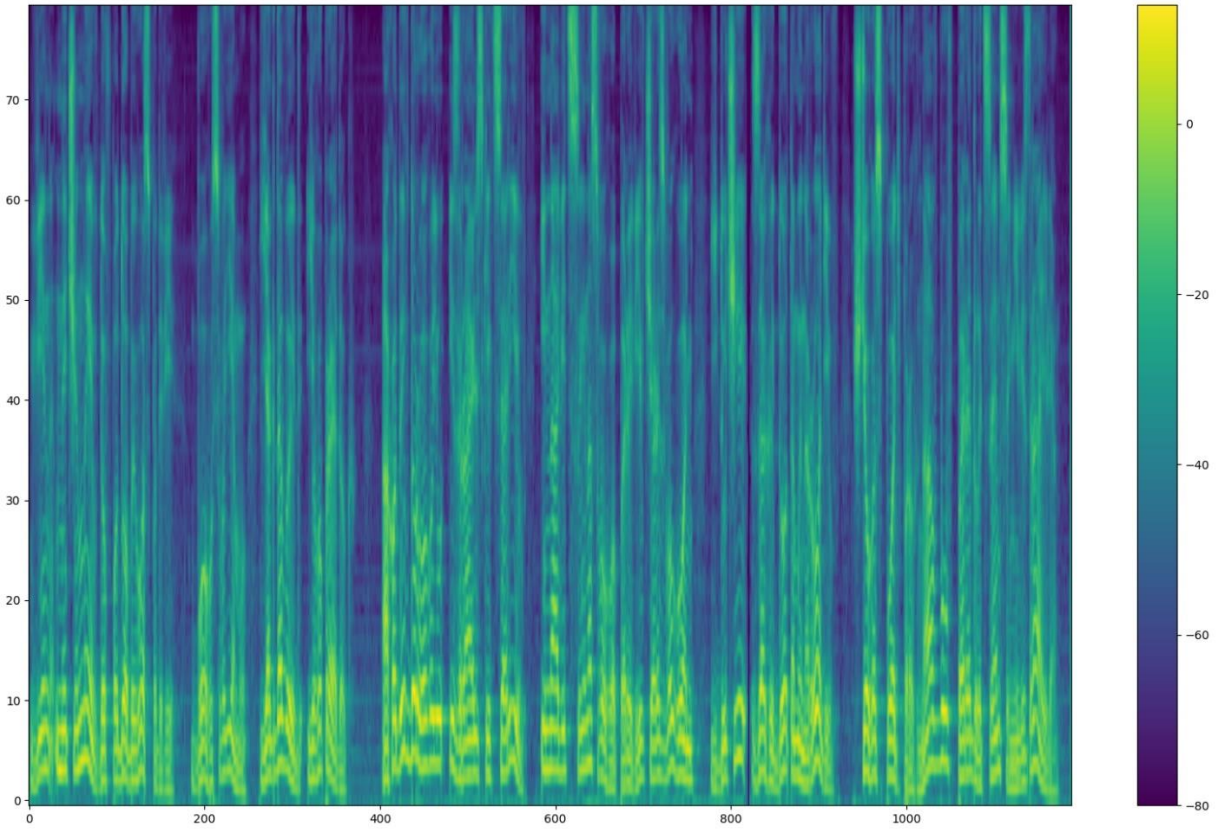

Рис. 3.7. Спектрограма Ground truth на етапі навчання

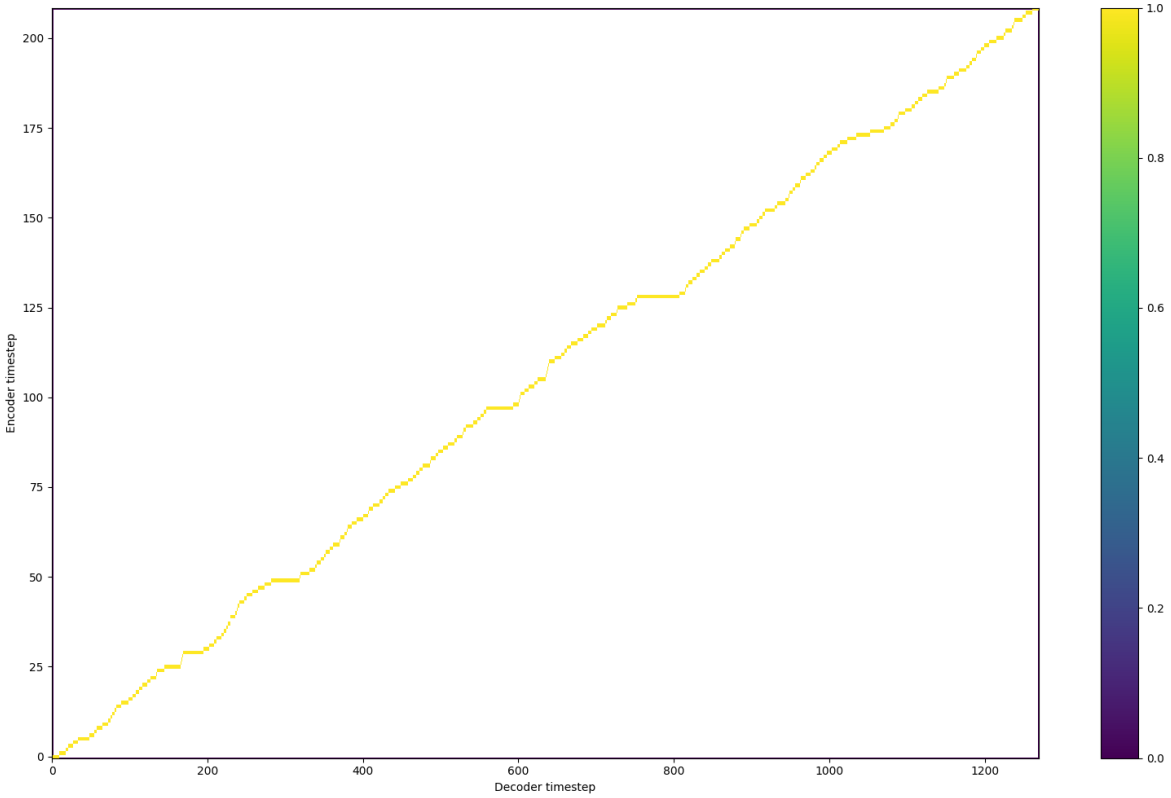

Рис. 3.8. Механізм уваги на етапі перевірки

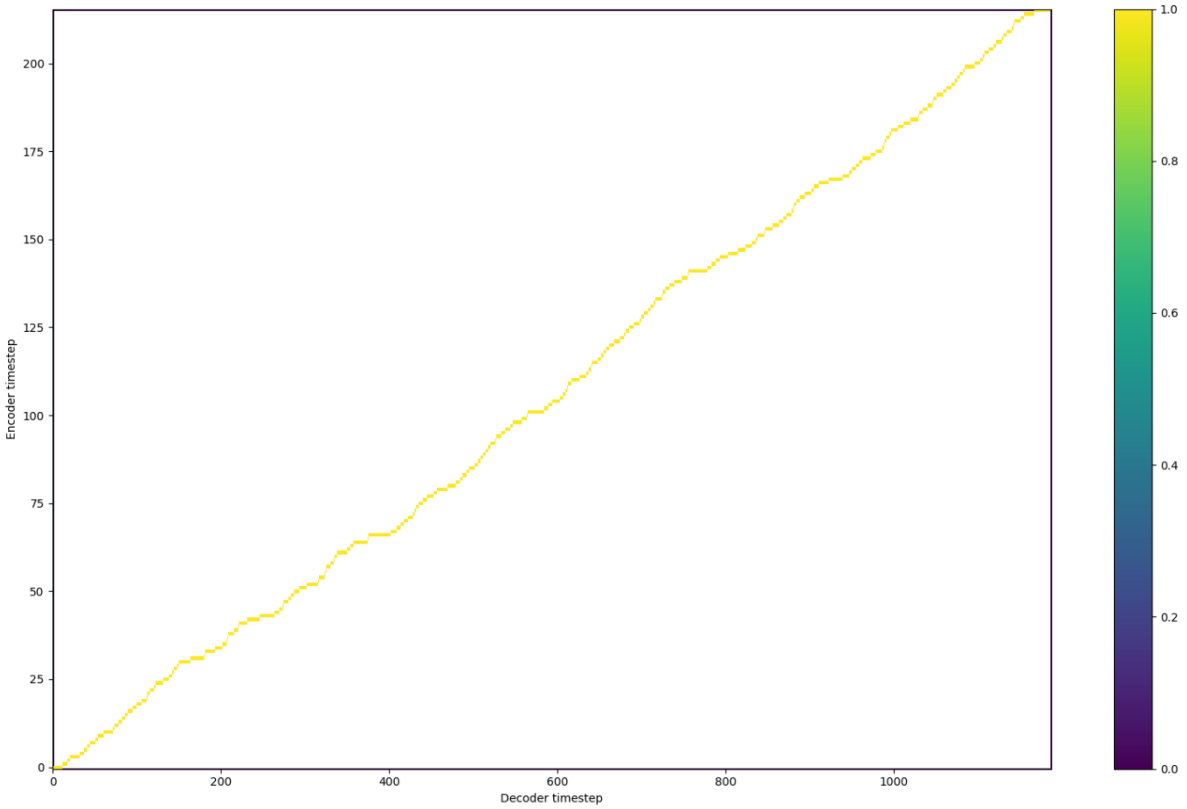

Рис. 3.9. Механізм уваги на етапі навчання

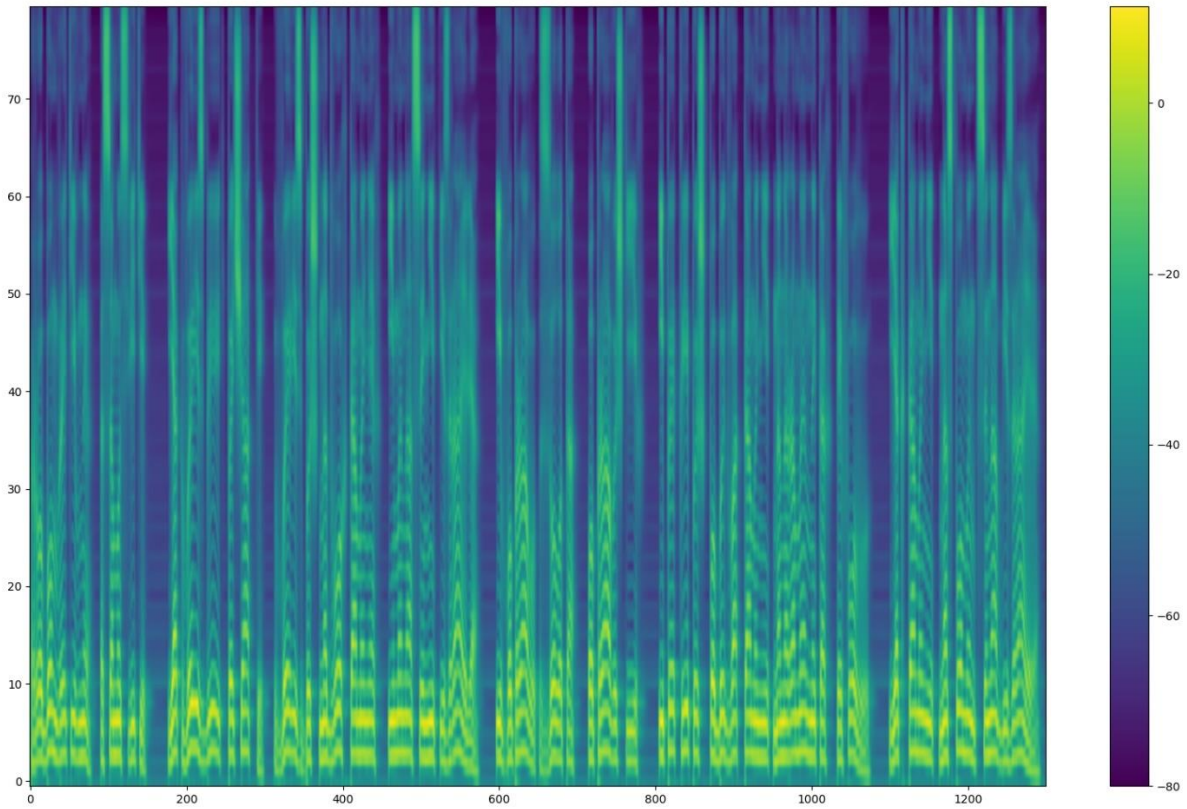

Рис. 3.10. Спектрограма передбаченого тексту на етапі перевірки

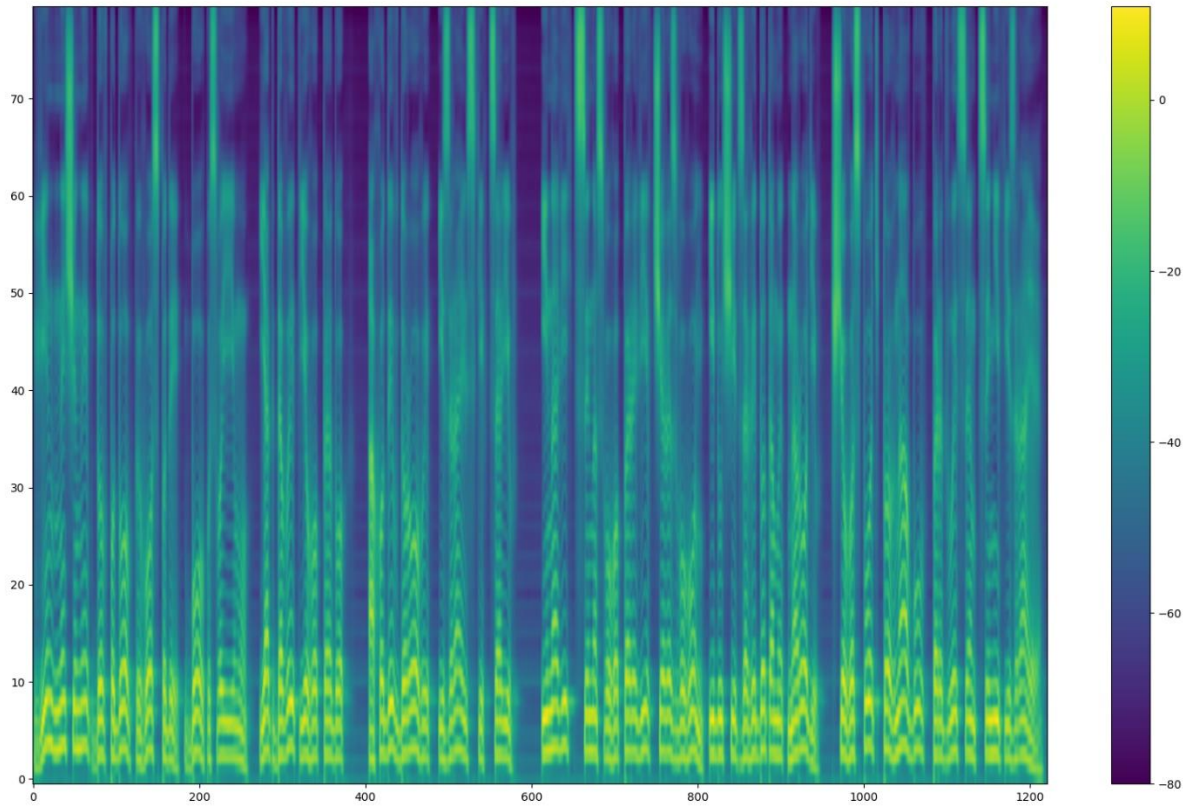

Рис. 3.11. Спектрограма передбаченого тексту на етапі навчання

На продемонстрованих рисунках можна побачити, що спектрограми передбаченого тексту і на етапі навчання і, що важливіше, на етапі перевірки є достатньо наближеними до Ground truth, що є непоганим результатом.

Щодо графіку механізму уваги, можна сказати що продемонстрований результат є не ідеальним, але в порівнянні з графіком на рисунку 3.12., який був отриманий через 400 кроків після початку навчання є досить прийнятним.

На рисунках 3.12. та 3.13. зображені графіки механізму уваги для нейронної мережі, яка пройшла 400 кроків навчання.

На прикладі даних графіків можна неозброєним оком побачити, що нейронна мережа показує досить не поганий результат на етапі навчання, але поганий на етапі перевірки. Даний висновок стає ще більш очевидним при перегляданні рисунків 3.14 та 3.15.

В той час, коли графік передбачення для окремого тексту, який береться з набору даних під час навчання є гіршим за результат представлений на рисунку

3.11, але є досить не поганим, якщо ж ми подивимось на спектрограму зображену на рисунку 3.15, то зрозуміємо що нейронна мережа майже не в стані передбачити мовлення згідно з наданим текстом, тому в ситуації з такою спектрограмою синтезованого мовлення у розробника немає іншого вибору як продовжити навчання моделі.

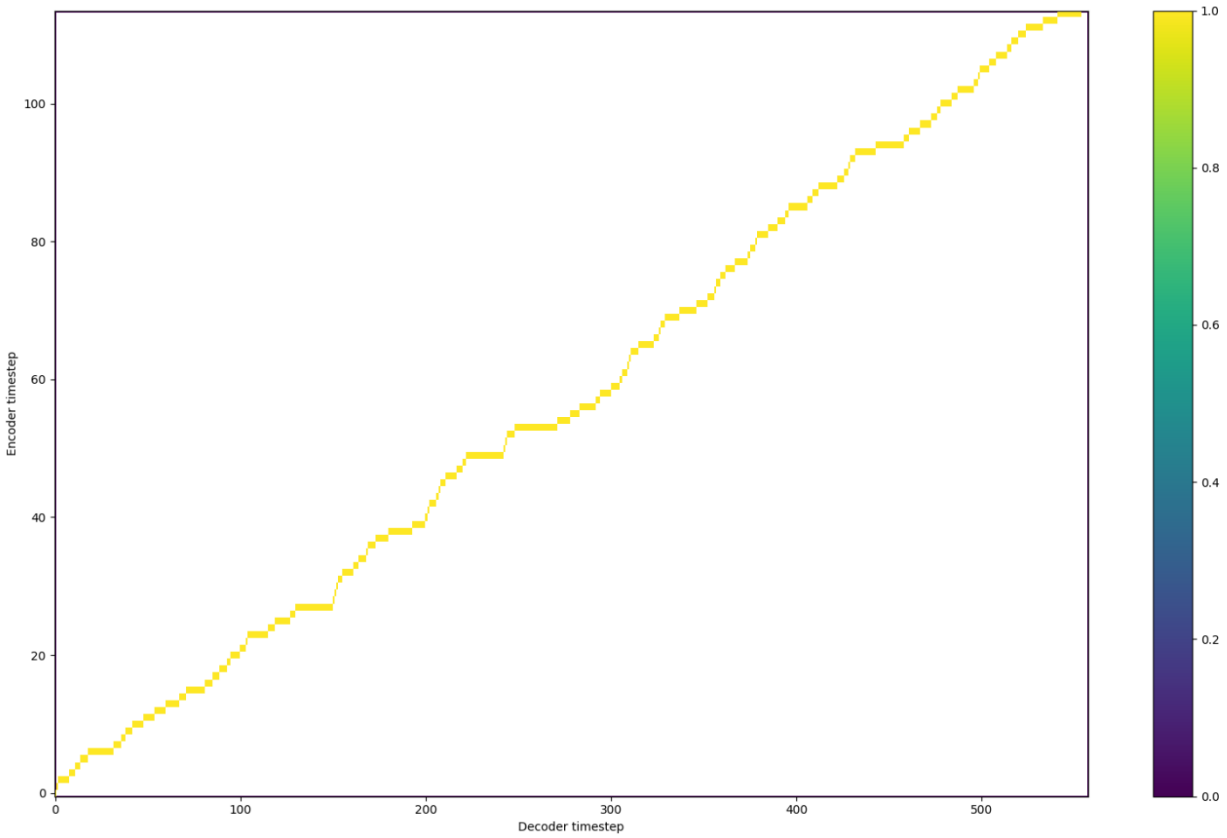

Рис. 3.12. Механізм уваги мережі через 400 кроків на етапі навчання

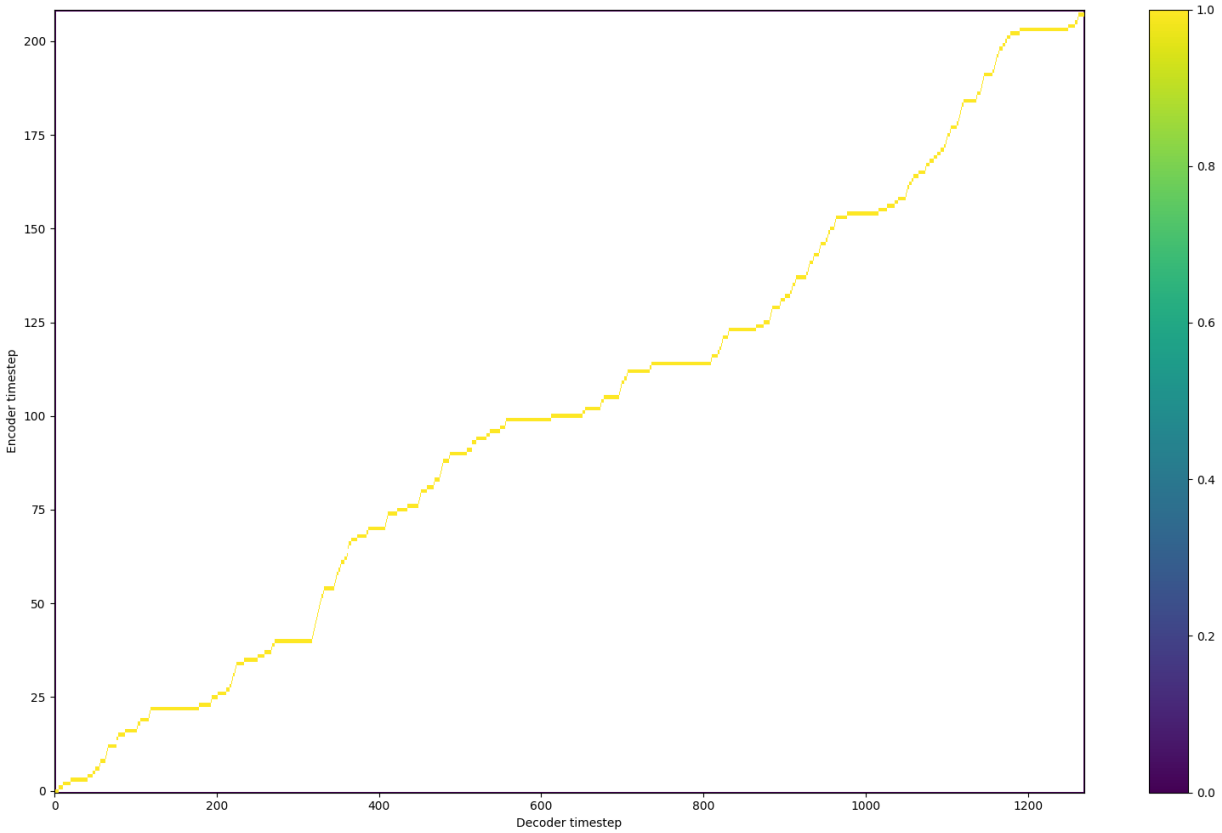

Рис. 3.13. Механізм уваги мережі через 400 кроків на етапі перевірки

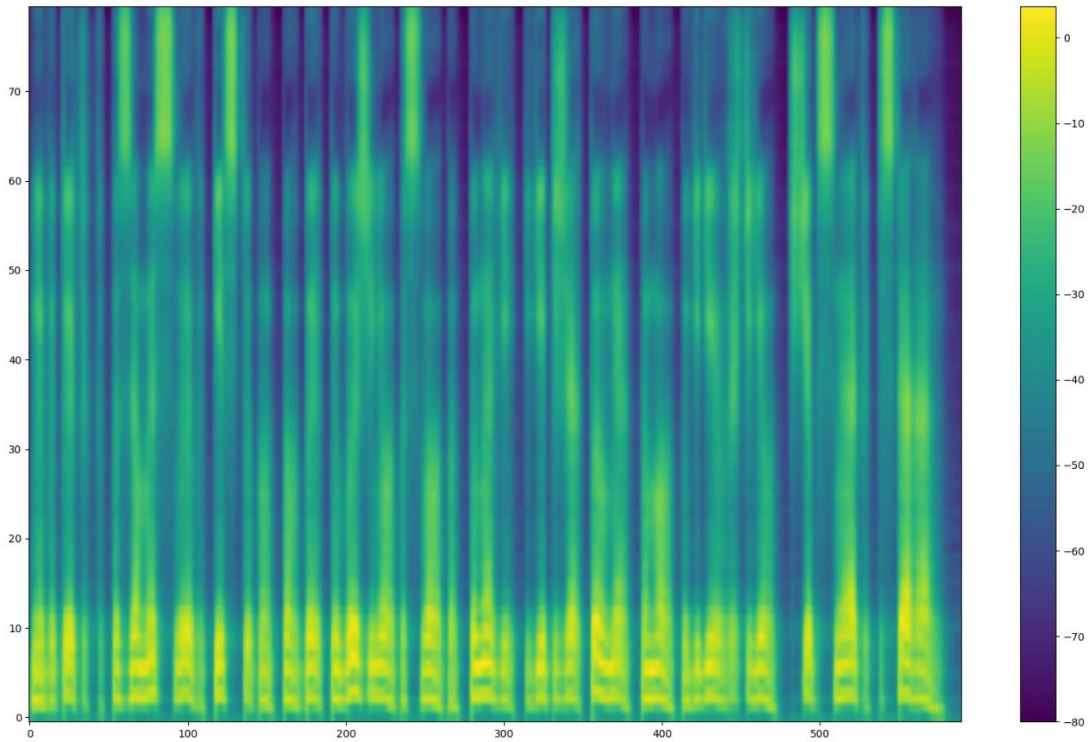

Рис. 3.14. Спектрограма передбаченого тексту мережі через 400 кроків на етапі навчання

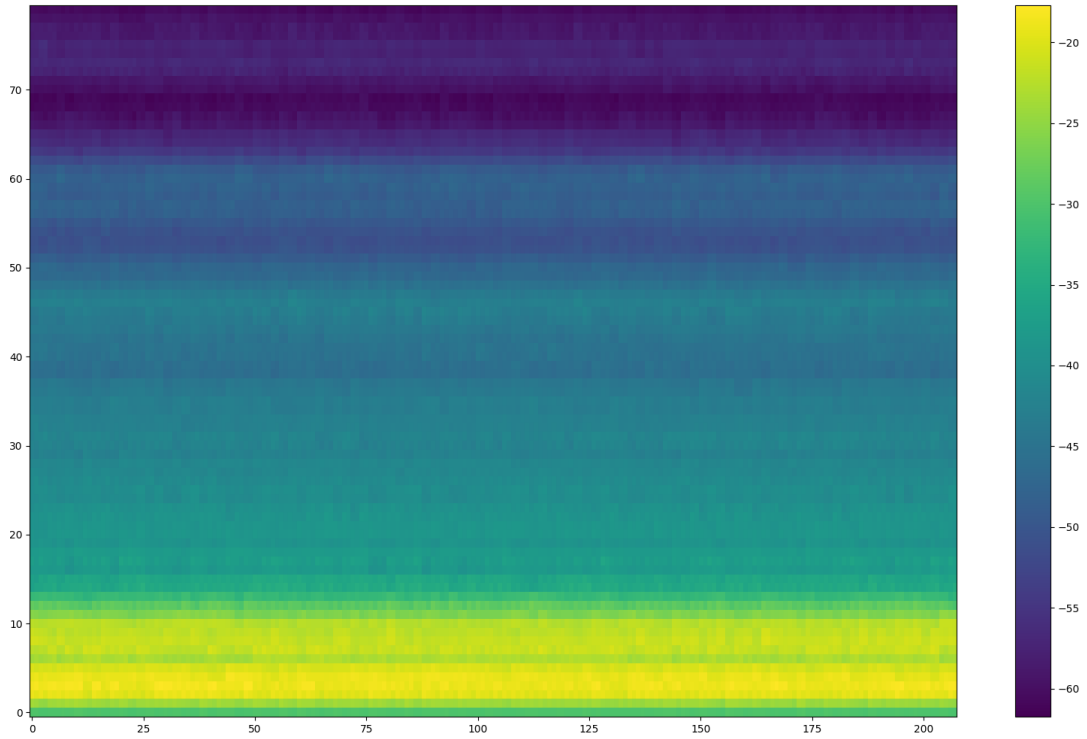

Рис. 3.15 Спектрограма передбаченого тексту мережі через 400 кроків на етапі перевірки

3.7. Порівняння результатів генерації сигналу моделлю

На рисунках 3.16–3.19 зображені різні сигнали, які були сформовані завдяки моделі синтезу тексту до мовлення.

Виходячи з результату на рисунку 3.16 можна сказати, що сигнал представляє щось на кшталт шуму з маленькою амплітудою при кількості кроків 400.

На рисунку 3.17. можна побачити, що сигнал стає більш схожим на сигнал звичайного мовлення та при прослухуванні вже стає розпізнаване синтезоване мовлення як щось, що відрізняється від шуму та віддалено схоже на мовлення людини.

На рисунку 3.18 можна побачити вже сформований сигнал людського мовлення, який при прослуховуванні дійсно дає можливість розібрати більшість слів та розуміти контекст сформованого речення.

Щодо результату на рисунку 3.19. Можна констатувати покращення якості формованого мовлення моделлю порівняно з моделлю на етапі 110 000 кроків (рисунок 3.18), але визначити дуже великий прогрес в звучанні при синтезі неозброєним вухом не виявляється можливим.

Дані результати підтверджують стандартний розподіл функції втрат при навчанні нейронної мережі. Це означає, що при подальшому навчанні нейронної мережі, зміни, які можна буде визначити в результаті синтезу, будуть менше виражені.

Рис. 3.16. Сигнал моделі, яка навчалась 400 кроків

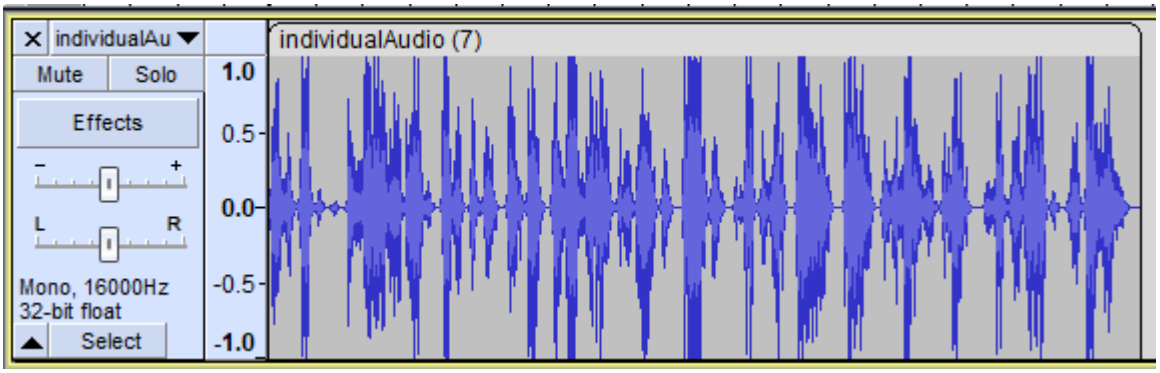

Рис. 3.17. Сигнал моделі, яка навчалась 20 000 кроків

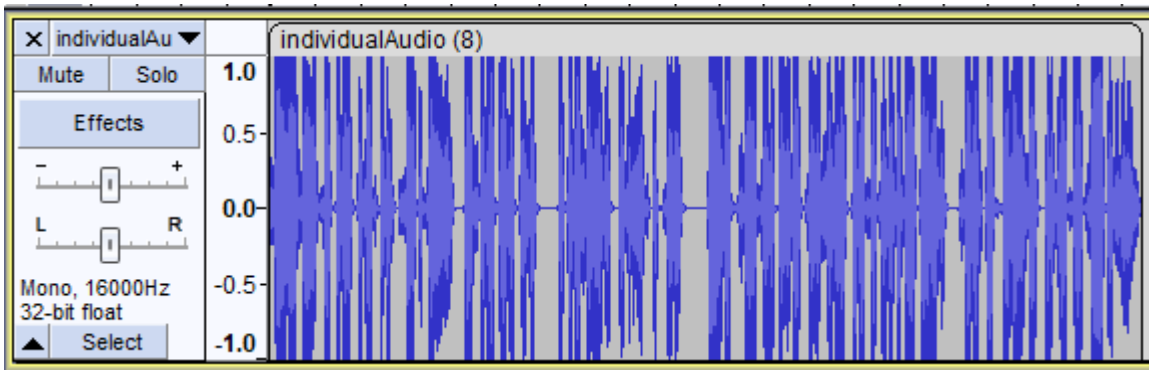

Рис. 3.18. Сигнал моделі, яка навчалась 110 000 кроків

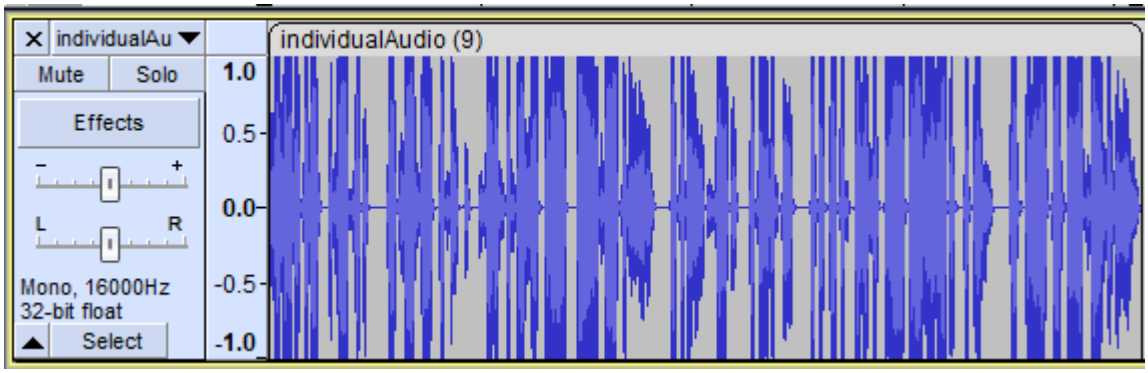

Рис. 3.19 – Сигнал моделі, яка навчалась 120 000 кроків

3.8. Висновки

Було створено модель трансляції тексту до мовлення з використанням інструментів з відкритим вихідним кодом Coqui-TTS моделі Glow-TTS. Після створення моделі були проведені експерименти над навченою моделлю на різних етапах навчання, зібрані метрики, з яких були отримані графіки механізму уваги та спектрограми утвореного мовлення під час навчання та перевірки, що дало змогу зробити висновок що обрана модель прогресує з часом в кінцевому варіанті може формувати українське мовлення, яке є дуже близьке до оригінального.

Результат дослідження є успішним, згідно з поставленою метою проєкту – «дослідження методик синтезу мовлення та обробки сигналів для створення моделі трансляції тексту до мовлення». Модель трансляції тексту до мовлення

була успішно модифікована для можливості навчання трансляції українського тексту до мовлення. В якості можливих покращень, можна запропонувати заміну графемних елементарних представлень тексту до фонемних, модифікація розглянутої моделі можливістю навчатись на наборі даних зібраних з різних мовців та модифікація вокодеру з метою покращення якості перетвореного сигналу.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи було досліджено методики синтезу мовлення та обробки сигналів та створена модель трансляції тексту до мовлення, яка була модифікована можливістю навчання моделей мовлення української мови. Проведений аналіз та оцінка результатів, графіки зі спектрограмами яких були наведені.

Для вирішення поставлених задач були використані нейронні мережі, хмарні технології, функціональне програмування, моделі трансляції тексту, методи обробки сигналів та Unix операційні системи.

Результати дослідження та проведеного аналізу можуть бути використані для подальшого дослідження в сфері трансляторів мовлення та для створення трансляторів з наведеними удосконаленнями для будь-яких сфер які потребують транслятора тексту до мовлення.

Подальші дослідження в даній сфері дозволять створювати моделі трансляції тексту до мовлення з урахуванням отриманого досвіду та з покращенням частин реалізації, наведених раніше, а саме: якості синтезу завдяки покращення вокодеру, якості навчання, завдяки заміни графем на фонеми української мови та додавання мовців різної статі та віку.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Manfred R. Schroeder. A brief history of synthetic speech [Електронний ресурс] / Manfred R. Schroeder – Режим доступу до ресурсу: https://doi.org/10.1016/0167-6393(93)90074-U.
2. How Speech Technologies Can Help People with Disabilities [Електронний ресурс] / Vlado Delić, Milan Sečujski, Nataša Vujnović Sedlar та ін.] – Режим доступу до ресурсу: https://doi.org/10.1007/978-3-319-11581-8_30.
3. Applications of Synthetic Speech [Електронний ресурс] – Режим доступу до ресурсу: http://research.spa.aalto.fi/publications/theses/lemmetty_mst/chap6.html.
4. Bernd J. Kröger. Articulatory Synthesis of Speech and Singing: State of the Art and Suggestions for Future Research [Електронний ресурс] / Bernd J. Kröger, Peter Birkholz – Режим доступу до ресурсу: https://doi.org/10.1007/978-3-642-00525-1_31.
5. Michel Divay. Text-To-Speech Formant Synthesis For French [Електронний ресурс] / Michel Divay, Ed Bruckert – Режим доступу до ресурсу: https://doi.org/10.1007/978-0-387-68439-0_13.
6. Text-to-Speech Engines Text Normalization [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/windows/win32/lwef/text-to-speech-engines-text-normalization?redirectedfrom=MSDN.
7. Jonathan Harrington. Digital Formant Synthesis [Електронний ресурс] / Jonathan Harrington, Steve Cassidy – Режим доступу до ресурсу: https://doi.org/10.1007/978-94-011-4657-9_7.
8. A Review of Deep Learning Based Speech Synthesis [Електронний ресурс] / Yishuang Ning, Sheng He, Zhiyong Wu та ін.] – Режим доступу до ресурсу: https://doi.org/10.3390/app9194050.

9. Alan W Taylor. Automatically clustering similar units for unit selection in speech synthesis [Електронний ресурс] / Alan W Taylor, Paul A – Режим доступу до ресурсу: https://era.ed.ac.uk/handle/1842/1236.
10. A.J. Hunt. Unit selection in a concatenative speech synthesis system using a large speech database [Електронний ресурс] / A.J. Hunt, A.W. Black – Режим доступу до ресурсу: https://ieeexplore.ieee.org/abstract/document/541110.
11. Rolf Carlson Prof. Rule-Based Speech Synthesis [Електронний ресурс] / Rolf Carlson Prof., Björn Granström Prof. – Режим доступу до ресурсу: https://www.doi.org/10.1007/978-3-540-49127-9_20.
12. Yann LeCun. Deep learning [Електронний ресурс] / Yann LeCun, Yoshua Bengio, Geoffrey Hinton – Режим доступу до ресурсу: https://www.nature.com/articles/nature14539.
13. Matt Spencer. A Deep Learning Network Approach to ab initio Protein Secondary Structure Prediction [Електронний ресурс] / Matt Spencer, Jesse Eickholt, Jianlin Cheng – Режим доступу до ресурсу: https://ieeexplore.ieee.org/abstract/document/6872810.
14. Kyubyong Park. CSS10: A Collection of Single Speaker Speech Datasets for 10 Languages [Електронний ресурс] / Kyubyong Park, Thomas Mulc – Режим доступу до ресурсу: https://arxiv.org/abs/1903.11269.
15. Naihan Li. Neural Speech Synthesis with Transformer Network [Електронний ресурс] / Naihan Li, Shujie Liu, Yanqing Liu. – 2018. – Режим доступу до ресурсу: https://arxiv.org/abs/1809.08895.
16. M. Schuster. Bidirectional recurrent neural networks [Електронний ресурс] / M. Schuster, K.K. Paliwal – Режим доступу до ресурсу: https://ieeexplore.ieee.org/abstract/document/650093.
17. Fast Fourier Transformation FFT - Basics [Електронний ресурс] – Режим доступу до ресурсу: https://www.nti-audio.com/en/support/know-how/fast-fourier-transform-fft.

18. Henri J. Nussbaumer . The Fast Fourier Transform [Електронний ресурс] / Henri J. Nussbaumer – Режим доступу до ресурсу: https://doi.org/10.1007/978-3-662-00551-4_4.
19. W.T. Cochran. What Is the Fast Fourier Transform [Електронний ресурс] / W.T. Cochran, J.W. Cooley, D.L. Favin – Режим доступу до ресурсу: https://doi.org/10.1109/PROC.1967.5957.
20. P.M. Bentley. Wavelet transforms: an introduction. Electronics & Communication Engineering Journal [Електронний ресурс] / P.M. Bentley, J.T.E. McDonnell – Режим доступу до ресурсу: https://doi.org/10.1049/ecej:19940401.
21. A. Arneodo. Wavelet Transform of Multifractals [Електронний ресурс] / A. Arneodo, G. Grasseau, M. Holschneider – Режим доступу до ресурсу: https://doi.org/10.1103/PhysRevLett.61.2281.
22. Dengsheng Zhang. Wavelet Transform [Електронний ресурс] / Dengsheng Zhang – Режим доступу до ресурсу: https://doi.org/10.1007/978-3-030-17989-2_3.
23. Nathanaël Perraudin. Stationary Signal Processing on Graphs [Електронний ресурс] / Nathanaël Perraudin, Pierre Vandergheynst – Режим доступу до ресурсу: https://ieeexplore.ieee.org/abstract/document/7891646.
24. Kunihiko Kodera. A new method for the numerical analysis of non-stationary signals [Електронний ресурс] / Kunihiko Kodera, Claude De Villedary, Roger Gendrin – Режим доступу до ресурсу: https://www.sciencedirect.com/science/article/abs/pii/0031920176900443.
25. Francisco Jurado. Comparison between discrete STFT and wavelets for the analysis of power quality events [Електронний ресурс] / Francisco Jurado, José R. Saenz – Режим доступу до ресурсу: https://www.sciencedirect.com/science/article/abs/pii/S0378779602000354.
26. Chaitali Chakrabarti. Architectures for wavelet transforms: A survey [Електронний ресурс] / Chaitali Chakrabarti, Mohan Vishwanath, Robert

M. Owens – Режим доступу до ресурсу: https://link.springer.com/article/10.1007/BF00925498.

27. Yuxuan Wang. Tacotron: Towards End-to-End Speech Synthesis [Електронний ресурс] / Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton – Режим доступу до ресурсу: https://arxiv.org/abs/1703.10135.
28. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation [Електронний ресурс] / Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre та ін.] – Режим доступу до ресурсу: https://arxiv.org/abs/1406.1078.
29. Yi Ren. FastSpeech 2: Fast and High-Quality End-to-End Text to Speech [Електронний ресурс] / Yi Ren, Chenxu Hu, Xu Tan – Режим доступу до ресурсу: https://arxiv.org/abs/2006.04558.
30. Nathanaël Perraudin. A fast Griffin-Lim algorithm [Електронний ресурс] / Nathanaël Perraudin, Peter Balazs, Peter L. Søndergaard – Режим доступу до ресурсу: https://ieeexplore.ieee.org/abstract/document/6701851.
31. Yoshiki Masuyama. Deep Griffin–Lim Iteration [Електронний ресурс] / Yoshiki Masuyama, Kohei Yatabe, Yuma Koizumi – Режим доступу до ресурсу: https://ieeexplore.ieee.org/abstract/document/8682744.
32. Ilya Sutskever. Sequence to Sequence Learning with Neural Networks [Електронний ресурс] / Ilya Sutskever, Oriol Vinyals, Quoc V. Le – Режим доступу до ресурсу: https://arxiv.org/abs/1409.3215.
33. Glow-TTS: A Generative Flow for Text-to-Speech via Monotonic Alignment Search [Електронний ресурс] / Jaehyeon Kim, Sungwon Kim, Jungil Kong, Sungroh Yoon – Режим доступу до ресурсу: https://arxiv.org/pdf/2005.11129v1.pdf.
34. Robert C. Streijl. Mean opinion score (MOS) revisited: methods and applications, limitations and alternatives [Електронний ресурс] / Robert C. Streijl, Stefan Winkler, David S. Hands – Режим доступу до ресурсу: https://doi.org/10.1007/s00530-014-0446-1.

35. Char Error Rate [Електронний ресурс] – Режим доступу до ресурсу: https://torchmetrics.readthedocs.io/en/stable/text/char_error_rate.html.
36. J. L. Flanagan. Phase Vocoder [Електронний ресурс] / J. L. Flanagan, R. M. Golden – Режим доступу до ресурсу: https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1966.tb01706.x.
37. Hans Fangohr. Jupyter in Computational Science [Електронний ресурс] / Hans Fangohr, Thomas Kluyver, Massimo DiPierro – Режим доступу до ресурсу: https://doi.org/10.1109/MCSE.2021.3059494.
38. Nikhil Ketkar . Introduction to Keras [Електронний ресурс] / Nikhil Ketkar – Режим доступу до ресурсу: http://doi.org/10.1007/978-1-4842-2766-4_7.
39. Navin Kumar Manaswi. Understanding and Working with Keras [Електронний ресурс] / Navin Kumar Manaswi – Режим доступу до ресурсу: https://link.springer.com/chapter/10.1007/978-1-4842-3516-4_2.
40. Martín Abadi. TensorFlow: Learning Functions at Scale [Електронний ресурс] / Martín Abadi – Режим доступу до ресурсу: https://doi.org/10.1145/2951913.2976746.

Додаток А

КОД ПРОГРАМИ

Install Tensorflow

```
!pip install --upgrade tensorflow-gpu==2.9.2
```

Import libs

```
# TensorFlow and tf.keras
import tensorflow as tf

# Helper libraries
import numpy as np
import matplotlib.pyplot as plt

print(tf.__version__)

device_name = tf.config.list_physical_devices('GPU')
device_name
```

Import audiobook

```
from google.colab import drive
drive.mount('/content/drive')

import os

dataset_path = 'drive/MyDrive/Dataset'
files = os.listdir(dataset_path)
documented_file = files[0] # File, that will be used for info printing
test_path = 'drive/MyDrive/Test'
test_files = os.listdir(test_path)
test_file = test_files[0]
```

Import scipy library

```
from os.path import dirname, join as pjoin
from scipy.io import wavfile
import scipy.io
import librosa
import librosa.display
from IPython.display import Audio

data, sr = librosa.load(f'{test_path}/{test_file}')   # Recorded data
sr

y = librosa.stft(data)
# Get the magnitude spectrogram
```

```
S = np.abs(y)
# Invert using Griffin-Lim

y_inv = librosa.griffinlim(S)

import matplotlib.pyplot as plt

fig, ax = plt.subplots(nrows=2, sharex=True, sharey=True)

librosa.display.waveshow(data, sr=sr, color='b', ax=ax[0])

ax[0].set(title='Original', xlabel=None)

ax[0].label_outer()

librosa.display.waveshow(y_inv, sr=sr, color='g', ax=ax[1])

ax[1].set(title='Griffin-Lim reconstruction', xlabel=None)

print("Original")
Audio(data=data, rate=sr)

print("Griffin and Lim")
Audio(data=y_inv, rate=sr)

print(data.shape)
print(y_inv.shape)
```

Print graph

```
time = np.linspace(0., length, data.shape[0])
plt.figure(figsize=(14, 5))
plt.plot(time, data, label = f"Audio file {documented_file}")
plt.legend()
plt.xlabel("Time [s]")
plt.ylabel("Amplitude")
plt.show()
```

Import fourier transform method

```
from scipy.fft import fft, fftfreq, fftshift
fourier_data = fft(data)
print(fourier_data)
```

Show fourier transformed plot of an audio

```
t = np.arange(data.shape[0])
sp = fftshift(fourier_data)
freq = fftshift(fftfreq(t.shape[-1]))
plt.plot(freq, sp.real, freq, sp.imag)
plt.show()
```

Add phonemes segmentation code

Add libraries import

```
from operator import contains
from tkinter import N
from anytree import Node, RenderTree
from anytree.search import findall
```

Create tree of Ukrainian phonemes

```
# Корень
phonemes = Node('root')
consonants = 'бвгґджзклмнпрстфхцчшщ'

# 6 голосні
Node('а', parent=phonemes)
Node('е', parent=phonemes)
Node('і', parent=phonemes)
Node('у', parent=phonemes)
Node('о', parent=phonemes)
Node('и', parent=phonemes)

# 32 приголосні
Node('м', parent=phonemes)
n = Node('н', parent=phonemes)
Node('нь', parent=n)
Node('б', parent=phonemes)
d = Node('д', parent=phonemes)
dz = Node('дз', parent=d)
Node('дзь', parent=dz)
Node('дж', parent=d)
Node('дь', parent=d)
Node('г', parent=phonemes)
Node('п', parent=phonemes)
t = Node('т', parent=phonemes)
Node('ть', parent=t)
ts = Node('ц', parent=phonemes)
Node('ць', parent=ts)
ch = Node('ч', parent=phonemes)
Node('к', parent=phonemes)
Node('в', parent=phonemes)
Node('й', parent=phonemes)
Node('ґ', parent=phonemes)
z = Node('з', parent=phonemes)
Node('зь', parent=z)
j = Node('ж', parent=phonemes)
Node('ф', parent=phonemes)
s = Node('с', parent=phonemes)
Node('сь', parent=s)
sh = Node('ш', parent=phonemes)
Node('х', parent=phonemes)
l = Node('л', parent=phonemes)
Node('ль', parent=l)
```

```
r = Node('р', parent=phonemes)
Node('рь', parent=r)

# 10 здвоєних приголосних
Node('нн', parent=n)
Node('дд', parent=d)
Node('тт', parent=t)
Node('лл', parent=l)
Node('цц', parent=ts)
Node('зз', parent=z)
Node('сс', parent=s)
Node('чч', parent=ch)
Node('жж', parent=j)
Node('шш', parent=sh)

specific_letters = {
    'я': ['йа', '\'а'],
    'ю': ['йу', '\'у'],
    'є': ['йе', '\'е'],
    'ї': ['йі', '\'і'],
    'щ': ['шч', '\'ш'],
}

specific_signs = 'ь\''
```

Add segmentation method

```
def get_phonemes(word):
    word = word.lower()
    result = ''
    counter = 0

    while counter < len(word):
        nodes = findall(
            phonemes, filter_=lambda node: node.name in word[counter])

        # Звичайний випадок
        if not (len(nodes) == 0 and word[counter] in specific_letters or
 word[counter] in specific_signs):
            result += word[counter]
            counter += 1
            continue

        # Якщо я, ю, є, ї, щ на початку слова
        if counter == 0:
            result += specific_letters[word[0]][0]
        # Якщо апостроф або м'який знак перед я, ю, є, ї - йа, йу, йе, й
і
        elif word[counter] in specific_signs:
            if word[counter] == 'ь':
                result += '\''
```

```
                if counter+1 < len(word) and word[counter+1] in specific
_letters:
                        result += specific_letters[word[counter+1]][0]
                        counter += 1
            else:
                result += f'{specific_letters[word[counter+1]][0]}'
                counter += 1
        # Якшо перед приголосним
        elif consonants.find(word[counter-1]) != -1:
            result += specific_letters[word[counter]][1]
        # Щ та Ї завжди позначають 2 звуки
        elif word[counter] == 'щ' or word[counter] == 'ї':
            result += specific_letters[word[counter]][0]

        counter += 1

    return result
```

Test method on an array of words

```
words = ['Гуцульщина', 'їдьте', 'стільці', 'піднось', 'будуються', 'ґедз
ь', 'гаївка', 'в\'юн', 'від\'їжджати', 'буряк', 'бур\'ян', 'кущі',
         'Єреван', 'білявка', 'здоров\'я', 'мольєр', 'яблуко']
for word in words:
    print(get_phonemes(word))

import os

# TrainingArgs: Defines the set of arguments of the Trainer.
from trainer import Trainer, TrainerArgs

# GlowTTSConfig: all model related values for training, validating and
testing.
from TTS.tts.configs.glow_tts_config import GlowTTSConfig

# BaseDatasetConfig: defines name, formatter and path of the dataset.
from TTS.tts.configs.shared_configs import BaseDatasetConfig,
CharactersConfig
from TTS.tts.datasets import load_tts_samples
from TTS.tts.models.glow_tts import GlowTTS
from TTS.tts.utils.text.tokenizer import TTSTokenizer
from TTS.utils.audio import AudioProcessor

# use the same path as this script as training folder.
output_path = 'coqui-output-ua'

# DEFINE DATASET CONFIG
dataset_config = BaseDatasetConfig(
    name="ukraineSpeech", meta_file_train="metadata.csv", path="Diploma"
)
```

```
character_config = CharactersConfig()
character_config.pad = '_'
character_config.eos = '~'
character_config.bos = '^'
character_config.punctuations = '''!'(),-.:;? '''
character_config.characters =
'НУЇШИЧГТЬЦВЩЖКФПЕРБҐЄМЯАОСЮІЛХЗДЙнуїшичгтьцвщжкфпербґємяаосюілхздй'

# INITIALIZE THE TRAINING CONFIGURATION
# Configure the model. Every config class inherits the BaseTTSConfig.
config = GlowTTSConfig(
    characters=character_config,
    batch_size=32,
    eval_batch_size=16,
    num_loader_workers=4,
    num_eval_loader_workers=4,
    run_eval=True,
    test_delay_epochs=-1,
    epochs=200,
    use_phonemes=False,
    print_step=25,
    print_eval=False,
    mixed_precision=True,
    output_path=output_path,
    datasets=[dataset_config],
    test_sentences=[
        'Цікаво перевірити синтезовану мову',
        'Вісімнадцяте святкування Дня Незалежності України',
    ]
)
config.audio.sample_rate=16000

# INITIALIZE THE AUDIO PROCESSOR
# Audio processor is used for feature extraction and audio I/O.
# It mainly serves to the dataloader and the training loggers.
ap = AudioProcessor.init_from_config(config)

# INITIALIZE THE TOKENIZER
# Tokenizer is used to convert text to sequences of token IDs.
# If characters are not defined in the config, default characters are
passed to the config
tokenizer, config = TTSTokenizer.init_from_config(config)

# custom formatter implementation
def formatter(root_path, manifest_file, **kwargs):  # pylint:
disable=unused-argument
    """Assumes each line as ```<filename>|<transcription>```
    """
    txt_file = os.path.join(root_path, manifest_file)
    items = []
```

```
    speaker_name = "spk_id_4727562"
    with open(txt_file, "r", encoding="utf-8") as ttf:
        for line in ttf:
            cols = line.split("|")
            wav_file = os.path.join(root_path, "wavs", cols[0] + '.wav')
            text = cols[1]
            items.append({"text":text, "audio_file":wav_file,
"speaker_name":speaker_name})
    return items

# LOAD DATA SAMPLES
# Each sample is a list of ```[text, audio_file_path, speaker_name]```
# can define your custom sample loader returning the list of samples.
# Or define custom formatter and pass it to the `load_tts_samples`.
# Check `TTS.tts.datasets.load_tts_samples` for more details.
train_samples, eval_samples = load_tts_samples(dataset_config,
eval_split=True, formatter=formatter)

# INITIALIZE THE MODEL
# Models take a config object and a speaker manager as input
# Config defines the details of the model like the number of layers, the
size of the embedding, etc.
# Speaker manager is used by multi-speaker models.
model = GlowTTS(config, ap, tokenizer, speaker_manager=None)

# INITIALIZE THE TRAINER
# Trainer provides a generic API to train TTS models with all its perks
like mixed-precision training, distributed training, etc.
trainer = Trainer(
    TrainerArgs(continue_path='coqui-output-ua/run-November-12-
2022_01+02AM-0000000/'), config, output_path, model=model,
train_samples=train_samples, eval_samples=eval_samples
)

trainer.fit()
```

Додаток Б

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
Дипломна_робота(Загинайло Є.О.).docx	Пояснювальна записка роботи. Документ Word.
Дипломна_робота(Загинайло Є.О.).pdf	Пояснювальна записка роботи. Документ PDF.
Програма	
Program.zip	Архів. Містить код програми.
Презентація	
Презентація(Загинайло Є.О.).pptx	Презентація дипломної роботи.