

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, факультету)

Кафедра комп'ютерних систем та мереж

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавра

(ступінь вищої освіти)

на тему Розробка системи керування синтезатором мови

Виконав: студент(ка) 4 курсу, групи КНТ-527

Спеціальності 123 «Комп'ютерна інженерія»
(код і найменування спеціальності)

Освітня програма (спеціалізація)
«Комп'ютерна інженерія»

Дмитренко Катерина Олександрівна
(прізвище та ініціали)

Керівник Грушко С.С.
(прізвище та ініціали)

Рецензент Куляба-Харитоновна Т.І.
(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування вищого навчального закладу)

Інститут, факультет інформатики та радіоелектроніки, комп'ютерних наук і технологій
 Кафедра «Комп'ютерні системи та мережі»
 Ступінь вищої освіти (освітній ступінь) бакалаврський
 Спеціальність 123 Комп'ютерна інженерія
 (код і найменування)
 Освітня програма (спеціалізація) Комп'ютерна інженерія
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри Кудерметов Р.К.

“ ” 2021 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТА

Дмитренко Катарині Олександровні

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка системи керування синтезатором мови

керівник проекту (роботи) Грушко Світлана Сергіївна, к. т. н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від “ 17 березня 2021 року № 81 ”

2. Строк подання студентом проекту (роботи) 11 травня 2021 року

3. Вихідні дані до проекту (роботи) пристрій з підтримкою та аккаунт Telegram, контент для синтезу мови, робоча станція (ПК)

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) Аналіз технологій комп'ютерного синтезування голосу

2) Розробка користувацького інтерфейсу системи управління синтезатором мови

3) Архітектура і реалізація системи управління синтезатором мови з використанням символів емої

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

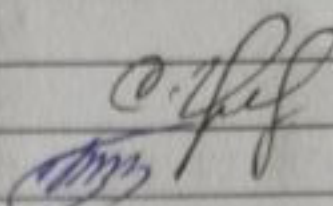
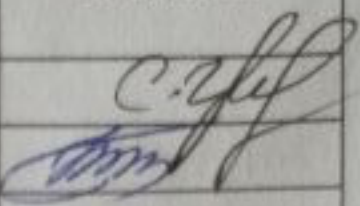
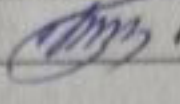
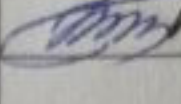
Технологічні підходи в комп'ютерному синтезі голосу;

Статистика використання емої;

Архітектура системи керування синтезатором мови;

Робота системи керування синтезатором мови

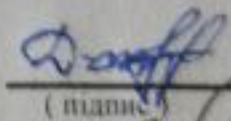
6. Консультанти розділів проекту (роботи)

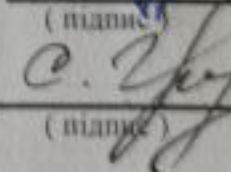
Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-3	Грушко С.С., к. т. н., доцент		
Нормоконтроль	Щербак Н.В., ст.викл. каф. КСМ		

7. Дата видачі завдання 01.03.2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз технічного завдання. Огляд сучасних рішень	01.03.2021 р.	
2	Розробка структури системи	15.03.2021 р.	
3	Розробка алгоритму роботи системи	20.03.2021 р.	
4	Реалізація проекту	01.04.2021 р.	
5	Дослідження роботи системи	10.04.2021 р.	
6	Оформлення отриманих результатів у ПЗ	15.04.2021 р.	
7	Оформлення графічного матеріалу	20.04.2021 р.	
8	Оформлення допоміжного матеріалу	01.05.2021 р.	

Студент  К.О. Дмитренко
(підпис) (ініціали та прізвище)

Керівник проекту (роботи)  С.С. Грушко
(підпис) (ініціали та прізвище)

РЕФЕРАТ

ПЗ: 62 с., 17 рис., 32 джерела, 2 додатки

СИНТЕЗ МОВИ, ТЕЛЕГРАМ, SSML, ІНТОНАЦІЙНИЙ КОНТУР

Об'єкт розробки – система керування синтезатором мови.

Метою магістерської роботи є розробка системи передачі голосових інтонацій в синтезовану комп'ютерну мову відповідно до авторського змісту тексту.

У пояснювальній записці досліджено різноманітні способи синтезування мови, розглянуто інструменти для зміни інтонацій в мові, що генерується, проведено огляд найбільших сервісів по синтезуванню мови. Досліджено і вивчено вплив символів емоції на соціум і сприйняття людиною емоцій, які ці символи передають. В роботі реалізовано систему керування синтезатором мови з урахуванням емоцій.

ЗМІСТ

Скорочення та умовні позначки	7
Вступ.....	8
1 Аналіз технологій комп'ютерного синтезування голосу	9
1.1 Сучасні тенденції розвитку технології синтезування мови.....	9
1.2 Визначення інтонаційного контуру	14
1.2.1 Генерація контуру F0 методом ресинтезу	15
1.2.2 Формування контура F0 для довільного речення.....	17
1.2.3 Генерація тонального контуру в системах інженерного типу	18
1.2.4 Генерація тонального контуру на основі лінгвістичних моделей інтонації	20
1.3 Технологічні підходи в комп'ютерному синтезі голосу	22
1.3.1 Text-to-Speech	22
1.3.2 Конкатенативний TTS	23
1.3.3 Синтез за правилами.....	24
1.3.4 Параметричний синтез	25
1.3.5 Гібридні (Deep Learning) підходи.....	26
1.4 Методи управління інтонаціями штучного голосу	30
2 Розробка користувацького інтерфейсу системи управління синтезатором мови	32
2.1 Обґрунтування пропонованого підходу управління голосовими інтонаціями комп'ютерного голосу	32
2.2 Емодзі	35
2.3 Створення бота в Телеграм	38
3 Архітектура і реалізація системи управління синтезатором мови з використанням символів емоції	44
Висновки	56
Перелік джерел посилання.....	57

Додаток А.....	61
Додаток В.....	60

Перелік графічних матеріалів:

Плакат 1 – Технологічні підходи в комп'ютерному синтезі голосу;

Плакат 2 – Статистика використання емоїї;

Плакат 3 – Архітектура системи керування синтезатором мови;

Плакат 4 – Робота системи керування синтезатором мови.

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

TTS	- Text-to-Speech (технологія перетворення тексту в мову)
DNN	- Deep Neural Network
IoT	- Інтернет речей
SSML	- Speech Synthesis Markup Language або
API	- Application Programming Interface
HTML	- Hyper Text Markup Language

ВСТУП

У той час як завдання розпізнавання мови дуже складне і вирішено лише частково, завдання синтезу мови набагато простіше (хоча і там є чимало проблем, що чекають свого рішення).

Технології синтезу мови застосовуються в багатьох ситуаціях, наприклад, в метро при оголошенні зупинок.

Власники мобільних телефонів можуть спілкуватися з автоматичною сервісною службою для визначення залишку коштів на рахунку, перемикання тарифних планів, підключення або відключення послуг тощо. Сервісна служба спілкується голосом із застосуванням технологій синтезу мови.

Випущено чимало дитячих іграшок, що «говорять» людським голосом. У цих іграшках також застосовуються най просодії, стресу стіші синтезатори мови або цифрові магнітофони.

Синтезатори мови застосовуються в різних голосових системах попередження, що встановлюються в автомобілях і літаках. Такі системи дозволяють привернути увагу людини до виникнення тієї чи іншої критичної ситуації, не відволікаючи його від процесу керування автомобілем, літаком чи іншим аналогічним засобом.

Також розроблено чимало комп'ютерних програм, здатних читати голосом вміст текстових файлів або текст, розташований на панелі програм. Ці системи можуть виявитися корисними тим, у кого ослаблений або повністю відсутній зір.

Додання емоційної окраски до синтезованої речі робить сприйняття інформації більш комфортною.

1 АНАЛІЗ ТЕХНОЛОГІЙ КОМП'ЮТЕРНОГО СИНТЕЗУВАННЯ ГОЛОСУ

1.1 Сучасні тенденції розвитку технології синтезування мови

Темп життя сучасної людини прискорюється з кожним роком. Люди знаходять все менше і менше часу для споживання вдумливого і серйозного контенту. Популярність гіпер-казуальних мобільних ігор тому приклад. У наш час, у читачів є доступ до більшої кількості історій, ніж будь-коли раніше. І проблема стоїть не в тому, щоб знайти цікаві статті, а в тому, щоб знайти час, щоб прочитати їх. Згодом, видавці новин можуть привернути увагу своїх читачів в середньому лише на одну хвилину. З іншого боку, зручність і мобільність аудіо-видань, пропонованих разом з новинними статтями, дають читачам новин абсолютно новий спосіб зануритися в історії, які їм важливі, а час залучення говорить сама за себе.

Завдяки аналітиці сервісу SpeechKit [1] ми можемо дізнатися, наскільки аудіо-підкасти збільшують залучення користувачів. Середній час, витрачений на прослуховування статті у новинних видавців SpeechKit, становить близько 3 хвилин 44 секунд на основі одного мільйона аудіо-новин. Це майже 300% збільшення часу взаємодії в порівнянні з традиційними статтями.

Аудіо-підкасти. Вони вже займають більшу частину в житті американців. Згідно зі статистикою, близько 70% жителів США (Сполучених Штатів Америки) знають що таке підкасти, а 30% жителів слухають підкасти на регулярній основі [2]. Успіх підкастів можна пов'язати з кількома факторами:

- розвиток мобільного інтернету і смартфонів. З поширенням мобільного інтернету і смартфонів, людям стало легше отримувати і ділитися контентом;
- втома від великої кількості контенту і необхідність його систематизації. До появи підкастів люди дивилися підсумкові передачі на телебаченні або читали аналітичні статті, так і зараз підкасти точно так само дозволяють отримати нову інформацію, розкласти по поличках новини та події;
- немає необхідності в концентраціях. Великим плюсом підкастів є схожість їх з радіо - можливість слухати, поки робиш будь-які справи: за кермом,

на пробіжці, коли гуляєш з собакою, миєш посуд, прибирати і так далі. Тільки на відміну від радіо, підкасти дають можливість слухачеві контролювати контент, а також дають свободу вибору - що слухати і коли слухати;

– емоційний зв'язок між автором і слухачем. Слухачі помічають риси характеру ведучого підкастів, вони починають співпереживати їм або сміятися разом з ними. Тим самим ведучі викликають у слухачів емоційний відгук і прихильність.

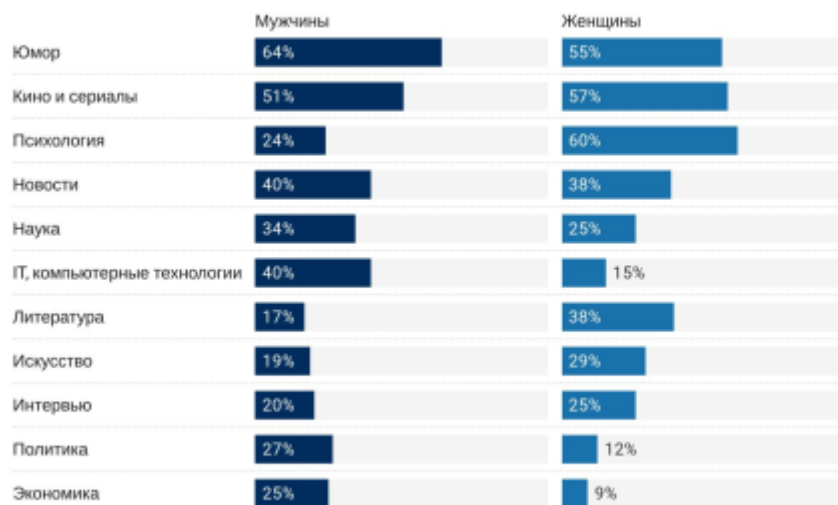
Згідно з дослідженням компанії Tiburon Research [3], кожен четвертий користувач інтернету слухає підкасти раз на місяць і частіше. Аудиторія по статі зміщена в бік чоловіків, але не набагато. За віком слухають підкасти переважно молодь і люди середнього віку, до 44 років. Підкасти слухають в основному люди із середнім доходом або вище.

Слухачі підкастів цікавляться новинами в більшій мірі, ніж ті, хто не слухає. 63% аудиторії слухають одноразово тільки один подкаст, 23% слухають від двох до трьох підкастів, 14% слухають чотири підкасти і більш. Жінки в основному цікавляться впорядкування файлів про психологію, новинними передачами, кіно і серіалами, літературою. Чоловікам цікаві гумористичні, новинні підкасти, підкасти про кіно. А також їм більшою мірою цікаві підкасти про економіку, політику, ІТ і комп'ютерні технології, ніж жінкам.

Люди найчастіше слухають підкасти в дорозі - коли йдуть на роботу або навчання, або коли просто гуляють. Підкасти супроводжують справи по дому у 59% жіночої аудиторії. Також люди часто слухають підкасти, коли займаються хобі. Третина чоловічої аудиторії слухає підкасти під час роботи. Оптимальна довжина подкасту на думку аудиторії - 24 хв.

Темы подкастов. Различия по полу

Подкасты какой тематики наиболее интересны для вас?



N=300, слушали подкасты хотя бы раз за последний месяц.
Источник: TIBURON Research • Создано с помощью Datawrapper

Рисунок 1.1 – Статистика розподілу тем підкастів в залежності від статі

Люди слухають підкасти на смартфонах, планшетах, на комп'ютерах, через розумну колонку і на Smart TV. 70% аудиторії слухає підкасти на смартфонах Android. Вони використовують програми, Google Podcasts, Spotify. На смартфони Apple припадає 18% слухачів. Вони переважно використовують додатки Apple Підкасти і Google Podcasts.

На думку дослідників, у аудиторії підкастів є потенціал для зростання. По-перше, це люди, які знають про існування підкастів, але не слухали їх, або слухають рідше ніж раз на місяць. По-друге, люди, які не знають про підкасти. Ну і по-третє, люди у віці понад 55 років.

Популярні бар'єри до підкастів - немає часу їх слухати, незручно сприймати інформацію на слух і немає цікавих для слухачів тем. Менш популярні причини - важко вибрати підкаст для прослуховування і незрозуміло яким чином слухати підкасти.

А що ж щодо ринку підкастів в Сполучених Штатах Америки? Згідно з дослідженням компанії Podcast Insights [4], на даний момент існує понад мільйон різних підкастів. Було випущено більше 30 мільйонів епізодів підкастів за весь час.

- 50% домоведення в США любителі підкастів. Це більше 60 мільйонів будинків по всій країні. [Nielsen Q3 2017 Podcast Insights] [5];
- 75% населення США знайомі з терміном «підкастинг» - в порівнянні з 70% в 2019 році. [Infinite Dial 2020] [6];
- 55% (155 мільйонів) населення США слухали хоча б один подкаст в 2020 році - в порівнянні з 51% в 2019 році (Infinite Dial 20) [6];
- 24% (68 мільйонів) слухають підкасти щотижня - у порівнянні з 22% в 2019 році (Infinite Dial 20) [6];
- 16 мільйонів чоловік в США є «палкими шанувальниками підкастів» (Nielsen Q1 2018) [7];
- 51% слухачів підкастів - чоловіки - таке ж співвідношення статі, як і у населення США (Infinite Dial 20) [6];
- 45% щомісячних слухачів підкастів мають дохід домогосподарства більше \$ 75 тис. Проти 35% для всього населення [4];
- 27% слухачів підкастів в США мають вищу освіту - проти 19% для всього населення в США [4];
- 63% слухачів підкастів - білі - проти 58% живуть в США (Infinite Dial 20) [6];
- вік слухачів щомісячних підкастів в порівнянні з населенням США (Infinite Dial 20): [6] 12-34: 48% (проти 37%), 35-54: 32% (проти 40%), 55+: 20% (проти 23%).

Тепер розглянемо статистику про те що, де, як і коли люди слухають підкасти:

- 49% прослуховування підкастів робиться вдома, в порівнянні з 51% в 2017 році [4];
- 22% слухають під час водіння (в автомобілі), так само, як 2017 [4];
- слухачі підкастів підписуються в середньому на 6 передач за останній тиждень (Infinite Dial 20) [6];

- слухачі підкастів слухають в середньому 7 різних шоу в тиждень, в порівнянні з 5 у 2017 році [4];
- 80% слухають подкаст повністю або більшу частину кожного епізоду, в порівнянні з 86% в 2017 році [4];
- прослуховування підкастів на комп'ютері зросла з 29% до 31% (прослуховування смартфонів зменшилася з 71% до 69% відповідно) [4];
- 19% слухачів прослуховують підкасти на збільшеній швидкості [4];
- 65% щомісячних слухачів підкастів слухали менше 3 років [4];
- комедія - найпопулярніший жанр підкастингу, за яким слідують освіта і новини [4].

Але запис подкасту це досить складний і трудомісткий процес. Для цього потрібно відповідне обладнання, людина з гарною дикцією і час на запис. Набагато легше нам було б генерувати підкасти за допомогою комп'ютерного синтезу мови. Таким чином можна зробити висновок, що актуальність синтезу мови в нинішній час дуже висока.

У той же час, переклад тексту в аудіо-підкасти вимагає вирішення проблеми - передачі голосових інтонацій відповідно до авторського змісту тексту. Крім цього, монотонний комп'ютерний голос досить дратівливий. Весь текст має одну і ту ж інтонацію по ходу всього повідомлення. Таким чином, це і близько не наближається до якості голосу, яку нам пропонують аудіо-підкасти. Тому для повноцінного сприйняття користувачами аудіо-інформації, необхідно надавати емоційне забарвлення повідомленням і передавати його при генерації аудіозаписів.

Сервіс SpeechKit дозволяє моментально додавати аудіо версію тексту на будь-якому сайті. Це дозволяє читачам споживати більше контенту на ходу, збільшує конверсію і розширює аудиторію блогів і новинних сайтів. Сервіс підтримує безліч різних мов, кілька синтезаторів (Google WaveNet, Amazon Polly, Microsoft Cognitive Services), а також надає статистику по прослуховуванням - кількість прослуховувань, середня довжина прослуховування, тип пристрою, на якому аудіо було прослухано, кількість кліків на посилання в плеєрі і так далі. Даний сервіс вже зарекомендував себе на ринку як надійний, ним користуються

новинні сайти і блоги з усього світу - Forbes, SingularityHub, the Canary, BusinessLive, La Prensa, Jyllands-Posten і інші.

1.2 Визначення інтонаційного контуру

Базові інтонаційні моделі, з обмеженого набору яких надходять створювачі синтезаторів, реалізуються практично на безконечній множині речень. Навіть в мовах, де тональний параметр не використовується для створення лексичних протипоставлень, реалізація базових моделей в конкретних пропозиціях може залежати від таких фонових властивостей, як довгі речення, кількість, місце розташування та ступінь вираженості словесних ударених, число слогів у виконаних словах, структура слогів і навіть їх звукова композиція. У результаті у різних реченнях контур F_0 (контур основної частоти голосів) може мати всю різноманітну та складну формулу: інтонаційно мотивовані зміни тону (підйоми та падіння) можуть створюватися рівними (платообразними) ділянками; у контурі можуть бути присутніми "дирки" та місцеві падіння, освічені глухотою / дзвінкою умовою; контур у цілому може бути розміщений у різних областях голосового діапазону говорячого; параметри тонального простору, що займається контуром (його робоча зона), можуть змінюватися від початку до кінця речень, наприклад, контур може одночасно знижуватися і зужуватися і т. д.

Відтворення подібних поверхневих ефектів при синтезуванні речей, з однієї сторони, необхідне, так як від цього сильно залежить естетичність кінцевого результату, а з іншого - представляє значущі труднощі. Це заставляє розробників або створювати самих, або шукати в лінгвістичній фонетиці які-небудь інтонаційно-просодичні моделі, які можна було використовувати на основі автоматичного породження тональних контурів. Елементи модельних представлень містять навіть у простіших системах, які забезпечують лише просодичний ресинтез.

1.2.1 Генерація контуру F0 методом ресинтезу

У системах, заснованих на просодичному ресинтезі, в пам'яті системи зберігаються детальні кількісні дані про контури основної частоти, інтенсивності і тривалості для деякого фіксованого набору фраз, отримані в результаті вимірювання їх природних вимов. Наприклад, контур основної частоти може бути збережений у вигляді послідовності чисел, що представляють результат поперіодного вимірювання звукової хвилі на вокальних відрізках фрази, або ж як послідовність значень, виміряних через невеликі часові інтервали (наприклад, кожні 10 мс) по контуру F0, отриманого за допомогою автоматичних методів акустичного аналізу мови. Потім ці дані відтворюються без зміни при генерації синтетичних відрізків, які не виходять, як правило, за межі того набору фраз, для яких в системі є готові просодичні зразки. Незважаючи на очевидні обмеження, описані системи знаходять своє застосування. Зокрема, вони виявляються корисними при тестуванні якості синтезаторів в озвучуванні сегментного складу мовних відрізків, тощо. Допомагають оцінити ступінь природності синтезованої мови, що складається з штучних звуків і природної просодії. В цьому випадку синтезуватися може будь-який мовний відрізок, проте для отримання просодичних даних для ресинтезу він повинен бути спочатку виголошений людиною, тобто стати відомим синтезуючій системі. Для розуміння закономірностей просодичного оформлення мовних відрізків подібний ресинтез не представляє особливого інтересу.

Ресинтез відомих просодичних зразків використовується також в системах, заснованих на так званих методах стилізації тонального контуру - акустичних або перцептивних. Мета акустичної стилізації полягає в тому, щоб скоротити детальну інформацію, яка міститься в контурах F0 природних фраз шляхом автоматичного виділення деякого набору опорних (цільових) точок, що апроксимують контур в цілому. Стилізація може бути широкою або вузькою, в залежності від дозволеної максимальної щільності опорних точок.

Вузький різновид стилізації часто реалізується у вигляді вибору трьох точок контуру на відрізку кожного окремого гласного фрази - початкової, екстремальної

(або серединної) і кінцевої. Опорні точки при апроксимації контуру з'єднуються прямими лініями.

При широкій стилізації в якості опорних точок часто вибираються локальні екстремуми контуру (піки і западини). Переходи між ними інтерполюються або прямими лініями, або більш складними функціями. При такому підході в якості особливих характеристик контура можуть використовуватися також прямі, що відображають динаміку зміни загального тонального простору контуру в часі. Лінія, що з'єднує локальні максимуми кривої F_0 , утворює верхню межу цього простору (topline). Нижня межа (baseline) задається локальними мінімумами. Спадний характер обох ліній відображає загальне зміщення контуру F_0 вниз, яке часто спостерігається при проголошенні оповідних пропозицій у багатьох мовах і називається деклінації.

Перцептивна стилізація відрізняється від чисто акустичної тим, що при виборі способу апроксимації кривої F_0 враховуються дані сприйняття. Найбільш відомим прикладом застосування методу перцептивної стилізації є модель, що розробляється з 1960 р в Інституті перцептивних досліджень (IPO) в Голландії. Вихідний контур F_0 спочатку апроксимується вручну послідовністю прямих відрізків (тональних сегментів), що не співвідносяться якимось спеціальним чином з сегментної основою аналізованої фрази. Потім фраза з апроксимувати контуром ресинтезується, далі за допомогою повторних ресінтезацій знаходиться така апроксимація контуру, яка містить мінімальну кількість тональних сегментів і на слух не відрізняється від вихідного контуру. Примітно, що в експертних експериментах з перцептивною стилізацією було виявлено, що модифікації кривої F_0 на ділянках глухих і дзвінких приголосних і суміжних з ними голосних (так звані мікропросодії) практично не впливають на сприйняття тонального контуру фрази. "Голландський" метод апроксимації контуру F_0 можна розглядати як широкий різновид перцептивної стилізації.

У деяких публікаціях описані методи автоматичної перцептивної стилізації, засновані на підходах, відмінних від голландського методу. Іноді приймається, що мінімальним носієм тональних відмінностей є склад. Сприйняття тони в рамках

складу залежить не тільки від F0, а й від інших фонетичних характеристик (тривалості, інтенсивності, звуковий структури і т. п.). На думку зазначених авторів, перцептивна стилізація тонального контуру фрази повинна являти собою послідовність тонованих складів, а тональний контур складу слід інтерпретувати з урахуванням впливу всіх акустичних чинників на сприйняття висоти тону, а також з урахуванням відомих психоакустичних даних (абсолютних і відносних слухових порогів оцінки тональних змін). Описаний метод є прикладом вузького різновиду перцептивної стилізації, він був реалізований в автоматичному режимі і інтенсивно тестувався на матеріалі французької мови. Судячи з наявних в літературі оцінками, всі методи стилізації контурів F0 дозволяють ресинтезувати мову високої якості. При створенні систем TTS отримання якісного тонального ресинтезу за допомогою тих чи інших автоматичних методів не є метою розробок, однак виконує важливу підготовчу функцію. По-перше, будь-який метод стилізації (за умови високої якості ресинтезу) дозволяє отримати таке уявлення спостережуваного контуру F0, яке звільнене від непотрібних акустичних деталей і параметризовані, тобто містить кількісну специфікацію кінцевого числа опорних тональних елементів (точок або відрізків), з допомогою яких апроксимується контур. По-друге, вибір опорних елементів стилізації часто відображає теоретичні уявлення (або допущення) дослідників про те, що являє собою глибинна інтонаційна характеристика речення, яка виходить (або може бути отримана) на виході лінгвістичного блоку підготовки тексту до озвучення. В цьому випадку ресинтез на основі обраного методу стилізації дозволяє дати попередню оцінку складності параметричного просодичного інтерфейсу і активно використовується для поточного налагодження правил генерації тонального контуру. У той же час ясно, що ресинтез сам по собі не може забезпечити породження тонального оформлення довільного речення.

1.2.2 Формування контура F0 для довільного речення

У конкретних системах автоматичного синтезу мови зміст і складність просодичних правил, що породжують тональний контур речення по його інтонаційному опису, залежить як від практичних можливостей лінгвістичного блоку системи, так і від того, що розуміється під інтонаційною структурою

пропозиції. Мінімальна інтонаційно значима інформація включає: вказівку на комунікативний тип пропозиції (sentence mode), інтонаційне членування і розташування акцентованих (або просто лексично ударних) складів в межах кожної інтонаційної групи. В рамках цього загального мінімуму вимоги до програми діляться на дві великі групи залежно від того, чи використовується в них власне інтонаційна транскрипція, що базується на деякому фіксованому наборі інтонаційних одиниць - загальних моделей або більше елементарних просодичних елементів, що входять в інтонаційну систему синтезованої мови. Умовно системи синтезу, в яких інтонаційна транскрипція на вході просодичних правил в явному вигляді не використовується, можуть розглядатися як реалізації інженерного підходу, на відміну від систем, що спираються на транскрипцію. Останні системи називаються лінгвістично (фонологічно) орієнтованими. Розглянемо основні особливості цих підходів.

1.2.3 Генерація тонального контуру в системах інженерного типу

В цю групу, перш за все, потрапляють додатки, які спираються на вузьку акустичну стилізацію тональних контурів. Алгоритми автоматичного отримання контурів F0 (pitch extraction) і автоматичної сегментації мовного сигналу створюють можливість побудови великих, просодичних орієнтованих баз даних, в яких фіксуються частотні значення опорних точок контуру F0 для кожного голосного або окремого складу в складі речення. Відповідність між мінімальною інтонаційно значущою інформацією, яка дається для кожного речення в базі даних, і тональними параметрами голосних або складів (з урахуванням великого набору поверхневих фонетичних змінних - типу складу, його положення в слові і інтонаційної групі і т.п.) встановлюється за допомогою статистичних класифікаційних методів або методів, що застосовуються в системах розпізнавання мови, зокрема нейроподібних мереж. Після такого попереднього аналізу або навчання реальний синтез довільного речення виходить шляхом конкатенації тональних складових контурів, вибраних з бази з урахуванням як інтонаційних ознак, так і поверхневих фонетичних чинників, що впливають на акустичну реалізацію складового контуру F0. Неважко бачити, що просодичні тональні

правила замінюються в системах описаного типу зберіганням великого інвентарю тональних складових контурів, які конкатенуються "склеюються", утворюючи складний тональний контур речення. За наявними у пресі відгуками, синтез на основі вузької акустичної стилізації і тональної конкатенації забезпечує дуже високу природність синтезованої мови. Розробки в цьому напрямку почалися порівняно недавно, їх технологічність, значна частка автоматизації підготовчої роботи залучають дослідників, що займаються мовними технологіями, і фахівці прогнозують бурхливе зростання відповідних додатків. У той же час з лінгвістичної точки зору подібні системи мало цікаві: фактично в них можна вбачати уявлення про інтонації як про деякий акустичний гештальт, який розгортається у вигляді складної тональної схеми на 44 складові ланцюжку речення. Однак можливо, що деякі мовні одиниці, ритуальні або несучі сильне емоційне забарвлення, дійсно запам'ятовуються і використовуються в мові, забезпечені подібними "гештальтними схемами-мелодіями", що знаходяться за межами власної інтонаційної системи мови. Безумовний інтерес для лінгвістично орієнтованих досліджень інтонації представляє комп'ютерний інструментарій, який використовується при створенні систем послогового конкатенативного тонального синтезу. Крім додатків, заснованих на конкатенації складових тональних контурів, до систем інженерного типу відноситься і ряд розробок, які насправді займають проміжне положення між чистою тональною конкатенацією і лінгвістично орієнтованими моделями тонального синтезу. У додатках такого типу найбільш часто використовується артикуляційно-акустична модель тонального контуру (productionoriented model), запропонована відомим японським фахівцем в області мовних технологій Х. Фуджісакі [12]. Основне припущення цієї моделі полягає в тому, що тональний контур, безперервний за своєю природою, є насправді реалізацією локальних фізіологічних подій, які здійснюються різними ларінгальними механізмами. Розрізняються два типи подій - фразові і акцентні тональні команди, які моделюються відповідно імпульсною і ступінчастою функціями. Крім цього, вводиться один глобальний параметр, який фіксує нижню межу робочої області голосового діапазону, на неї накладаються фразові і акцентні

команди. Локальні компоненти моделі описуються кількома параметрами, які задають відносну амплітуду тонального зміни і часові моменти реалізації команд (таймування) щодо меж фрази (для фразових імпульсів) і кордонів акцентованого слова для акцентних. Результируючий тональний контур виходить шляхом складання всіх компонентів, що мають, як впливає зі сказаного вище, різні області реалізації в часі. У зв'язку з цим модель Фуджісакі часто відносять до суперпозиційної фонетичної моделі інтонації. При створенні системи синтезу для конкретної мови використовуються просодичні бази даних, де кожне речення містить, принаймні, мінімальну інтонаційну інформацію. При аналізі корпусу речень фразові команди співвідносяться з межами інтонаційного членування, а акцентні - з акцентованими складами. Амплітудні і часові параметри апроксимуючих функцій підбираються по базі даних за допомогою статистичних методів. Модель тестувалася в системах синтезу для вельми різних мов: японської, англійської, китайської, німецької та ряду інших.

1.2.4 Генерація тонального контуру на основі лінгвістичних моделей інтонації

В лінгвістично орієнтованих системах тонального синтезу контур F_0 розглядається як акустична маніфестація інтонаційної структури речення, яка може бути представлена у вигляді певної конфігурації абстрактних інтонаційних елементів, які повинні фіксуватися в вихідній транскрипції лінгвістичного блоку синтезатора. У розробці таких систем активну участь беруть лінгвісти. Відповідно до теоретичних напрямків, існуючими в західній інтонології, можна виділити два типи моделей, які не тільки знаходять застосування в системах синтезу мови по тексту, але і завдяки цьому активно розвиваються. Це так звані суперпозиційні (layered components) і лінійні або послідовні (tone sequences) моделі. Обидва типи моделей виходять з уявлення про комбінаторної природі інтонації: інтонаційна структура речення розглядається як конструкція, що складається з декількох функціонально самостійних тональних елементів. Обидва типи моделей визнають існування і лінгвістичну значимість локальних тональних об'єктів, що мають фіксовану часову прив'язку в реченні, і глобальні тональні ознаки, які

характеризують тональний простір, в рамках якого реалізується контур в цілому. Однак функціональна інтерпретація локальних і глобальних тональних елементів і їх взаємодію в реченні трактується в цих моделях по-різному. У суперпозиційній моделі інтонаційна структура пропозиції розглядається як ієрархічна просодична структура, яка визначається в кожній точці речення одночасно трьома тональними об'єктами, кожен з яких має свою сферу реалізації. Тональні складові описуються наступним чином: виділяються глобальні тональні ознаки, що характеризують тональний простір, в якому реалізується речення в цілому, глобальні тональні ознаки простору, займаного послідовними інтонаційними групами в реченні, і тональні фігури, які реалізуються на складових, які називаються акцентними групами. Інтонаційні контури основних комунікативних типів речень відрізняються тільки глобальною тональною ознакою, що відображає частотне зміщення тонального контуру в часі (його піднаглядним корелятом служить лінія деклінації, що з'єднує акцентовані склади в реченні). Так, розповідні речення мають найбільш різкий нахил деклінаційної лінії, а загальне питання характеризується відсутністю нахилу (плоскою лінією деклінації). Реалізаційною базою лінгвістичних моделей суперпозиційного типу є описана вище модель Фуджісакі.

Абстрактні тональні репрезентації, які умовно можна розглядати як маршрут або схему руху в цільовому тональному просторі, перетворюються в спостережувані контури F0 за допомогою просодичних правил двох типів: тонального шкалювання і таймування. Правила тонального шкалювання визначають для абстрактних цільових тонів конкретні значення F0, які вважаються залежними від двох факторів: ступеня вибраного складу, який несе тон, і тональної специфікації попереднього тону. Таким чином, частотна специфікація послідовності тонів здійснюється строго зліва направо (звідси назва "лінійна" модель). Правила таймування задають з урахуванням різних поверхневих фонетичних чинників координати часових точок, в яких повинна досягатися тональна мета. Крім просодичних правил, використовуються адаптуючі функції, за допомогою яких в контурі F0 цільові тональні точки з'єднуються тональними переходами і контур в цілому згладжується. Глобальні тенденції, які

спостерігаються в контурах F0, в крайніх варіантах лінійної моделі описуються також виключно локально. Наприклад, деклінації вважається поверхневим результатом локальної взаємодії певних суміжних тонів (аналогічно downstep в африканських мовах), а не глобальною тональною ознакою, що поширюється на всю інтонаційну групу. Локальна інтерпретація глобальних тенденцій є найбільш дискусійною стороною строго лінійних моделей і причиною побудови різних гібридних моделей, автори яких вводять в лінійну модель і глобальні тональні ознаки. В цілому, треба сказати, що на Заході, особливо в США, лінійна модель Пьерхумберт отримала дуже великий резонанс як в фонологічних дослідженнях, так і прикладних розробках. Ця модель в адаптованому вигляді використовувалася в системах синтезу для англійської, німецької, китайської, японської та шведської мов. При створенні додатків все просодичні правила і адаптуючі функції настроюються автоматично за допомогою великих анотованих баз даних. Для інтонаційної анотації мовних корпусів була створена широко відома просодична транскрипційна система ToBI (Англ. Tones and Break Indices). У той же час не можна не відзначити, що лінгвістичний (функціональний) потенціал лінійної моделі навіть для англійської мови в повній мірі не перевірений і не використовується в системах синтезу, так як до сих пір не сформульовані правила вибору тонів, що утворюють тональний компонент інтонаційної структури пропозиції.

1.3 Технологічні підходи в комп'ютерному синтезі голосу

1.3.1 Text-to-Speech

Text-to-Speech (TTS) або синтез мови, у вузькому сенсі - це формування мовного сигналу за друкованим текстом або штучне виробництво людської мови в цифровому форматі. Синтез мови застосовується в багатьох сферах нашого життя - в інформаційно-довідкових системах, в оголошеннях про відправлення поїздів і тому подібного, у голосових помічників і в загальному в акустичних діалогах людини з комп'ютером.

В даний час метою TTS є не просто змусити машини говорити, а зробити так, щоб вони звучали як люди різного віку і статі. У перспективі ми зможемо слухати озвучені машиною аудіокниги і новини по телевізору або спілкуватися з помічниками, не помічаючи різниці між ними і справжніми людьми. Існують кілька підходів конверсії TTS - параметричний синтез, конкатенативний синтез, синтез за правилами, гібридний підхід із застосуванням Deep Neural Network (DNN).

Порівнюємо дані підходи і дізнаємося, який підходить нам більше. Для цього спочатку розберемося з критеріями порівняння. Як правило, якість синтезаторів системи TTS оцінюється з різних сторін, включаючи розбірливість, природність і переваги синтетичної мови [8], а також фактори людського сприйняття, такі як зрозумілість [9]:

- чіткість - якість згенерованого звуку або кожного слова, виробленого в реченні;
- природність - якість мовлення, що генерується з точки зору її часової структури, вимови і передачі емоцій. Перевага - вибір слухача; на переваги і природність впливають система TTS, якість сигналу і голоси, окремо і в комбінації;
- зрозумілість - ступінь розуміння отриманих повідомлень.

Розробки в області комп'ютерних наук і штучного інтелекту впливають на підходи до синтезу мови, які розвивалися роками у відповідь на останні тенденції і нові можливості в зборі та обробці даних. Протягом довгого часу двома основними методами перетворення тексту в мову є конкатенативний TTS і параметричний TTS. Революція глибокого навчання додала нову перспективу до проблеми синтезу мови, змістивши акцент з мовних функцій, розроблених людиною, на повністю машинно-отримані параметри [10,11].

1.3.2 Конкатенативний TTS

Конкатенативний TTS спирається на високоякісні записи аудіокліпів, які об'єднуються для формування мови. На першому етапі записуються голосові актори, що вимовляють діапазон мовних одиниць, від цілих речень до складів, які додатково маркуються і сегментуються лінгвістичними одиницями від телефонів до фраз і речень, що утворюють величезну базу даних. Під час синтезу мови

механізм перетворення тексту в мову шукає в такій базі даних мовні блоки, які відповідають вхідному тексту, об'єднує їх разом і створює аудіофайл.

До плюсів такого підходу можна навести:

- високу якість звуку з точки зору чіткості;
- можливість збереження оригінального акторського голосу.

А до мінусів:

- такі системи дуже трудомісткі, оскільки вимагають величезних баз даних і жорсткого кодування комбінації для формування цих слів;
- результуюча мова може звучати менш природно і беземоційно, тому що практично неможливо отримати аудіозаписи всіх можливих слів, вимовлених у всіх можливих комбінаціях емоцій, просодії, стресу і т. д.

Як приклад конкатенативного TTS можна привести синтез співаючого голосу. Маючи можливість записати певного співака, такі системи здатні зберегти спадщину, відновивши записи зірок минулих днів, як робить Asapela Group, а також змусити вашого улюбленого співака виконати іншу пісню на свій смак, як в Vocaloid.

1.3.3 Синтез за правилами

Метод синтезу за правилами створює мовні сегменти, генеруючи штучні сигнали на основі набору певних правил, що імітують формантну структуру та інші спектральні властивості природної мови. Синтезоване мовлення проводиться з використанням адитивної синтезу і акустичної моделі. У акустичної моделі використовуються такі параметри, як озвучування, основна частота, рівні шуму і т.д., які змінювалися в часі. Системи, засновані на формантах, можуть керувати всіма аспектами вихідної мови, створюючи різноманітні емоції і різні тональні голоси за допомогою деяких методів просодичного і інтонаційного моделювання.

Плюсами такого підходу є:

- добре поставлене, зрозуміле синтезоване мовлення навіть на високих швидкостях, відсутність акустичних глюків;
- менше залежить від вихідних мовних даних для виробництва вихідний мови;

- добре підходить для вбудованих систем, де пам'ять і потужність мікропроцесора обмежені.

Мінусами ж даного підходу є:

- низька природність - техніка створює штучну, що лунає роботом мову, яка далека від природної мови людини;
- складно розробити правила, які визначають синхронізацію джерела і динамічні значення всіх параметрів фільтра навіть для простих слів.

Одним з найбільш відомих прикладів техніки синтезу за правилами є *espeak-ng*, багатомовна система синтезу мовлення з відкритим вихідним кодом, заснована на синтезаторі Клаттен. Ця система включена в якості синтезатора мови за замовчуванням в програму читання з екрану з відкритим вихідним кодом NVDA для Windows, Android, Ubuntu і інших дистрибутивів Linux. Більш того, його попередник *eSpeak* використовувався Google Translate для 27 мов в 2010 році.

1.3.4 Параметричний синтез

Щоб усунути обмеження конкатенативного TTS, був розроблений більш статистичний метод. Ідея, що лежить в основі цього, полягає в тому, що якщо ми зможемо наблизитися до параметрів, з яких складається мова, ми зможемо навчити модель для генерації всіх видів мовлення. Параметричний метод об'єднує параметри, включаючи основну частоту, амплітудний спектр і т.д., і обробляє їх для генерації мови. На першому етапі текст обробляється для отримання мовних особливостей, таких як фонема або тривалість. Другий крок вимагає вилучення функцій вокодера, такі як кепстра, спектрограма, основна частота і т. д., які представляють собою деяку невід'ємну характеристику людської мови і використовуються при обробці звуку. Ці функції розроблені вручну і, поряд з лінгвістичними функціями, включені в математичну модель, яка називається вокодером. При генерації форми хвилі вокодер перетворює характеристики і оцінює параметри мови, такі як фаза, швидкість мовлення, інтонація та інші. У техніці використовуються приховані напівмарковські моделі

- переходи між станами все ще існують, і модель на цьому рівні є марковською, але явною моделлю тривалості в кожному стані не є марковська.

Плюси:

- збільшена природність звуку. На жаль, технологія створення емоційних голосів ще не вдосконалена, але це те, на що здатний параметричний TTS. Він має великий потенціал в області емоційних голосів;

- гнучкість - легше змінити висоту тону для емоційних змін або використовувати адаптацію MLLR для зміни характеристик голосу;

- більш низькі витрати на розробку: для цього потрібно всього 2-3 години запису голосу актора, що сприяє меншій кількості записів, меншій базі даних і меншій обробці даних.

Мінуси:

- більш низька якість звуку з точки зору чіткості - є багато артефактів, що призводять до приглушеної мови, з постійно гуде, гучним звуком;

- голос може здаватися роботизованим - в TTS, заснованому на статистичній моделі, приглушений звук робить голос стабільним, але неприродним і роботизованим.

Незважаючи на появу параметричних TTS ще в 1990-х роках, популярними вони стали тільки в 2007 році з появою Festival Speech Synthesis System з Університету Единбурга і Університету Карнегі. Festvox є прикладом таких движків, що лежать в основі систем синтезу мови, таких як FreeTTS.

1.3.5 Гібридні (Deep Learning) підходи

Підхід, заснований на DNN (Deep Neural Network), являє собою ще один варіант підходів статистичного синтезу, який використовується для подолання неефективності дерев рішень, використовуваних в прихованій марковській моделі, для моделювання складних тематичних залежностей. Крок вперед і подальший прорив дозволили машинам проектувати елементи без втручання людини. Функції, розроблені людьми, засновані на нашому розумінні мови, але не обов'язково, що цей підхід правильний. У методах DNN зв'язок між вхідними текстами і їх акустичними реалізаціями моделюється самою DNN. Акустичні особливості створюються з використанням згладжування генерації параметра максимальної правдоподібності. Функції, отримані за допомогою Deep Learning, не читаються

людиною, але вони багато читались комп'ютером і представляють дані, необхідні для моделі.

Плюси:

- величезне поліпшення як з точки зору зрозумілості, так і природності;
- не вимагає великої попередньої обробки людиною і розробки функцій.

Мінуси:

- методи синтезу мови Deep Learning все ще вимагають досліджень.

Синтез за допомогою Deep Learning зараз домінує в області генерації аудіо мови, будучи ядром практично всіх успішних систем TTS, таких як WaveNet, Nuance TTS або SampleRNN.

Nuance TTS і Sample RNN - дві системи, які покладаються на рекурентні нейронні мережі. SampleRNN, наприклад, використовує ієрархію рекурентних рівнів, які мають різні тактові частоти для обробки аудіо. Ієрархія форм з декількома RNN, де верхній рівень приймає великі порції вхідних даних, обробляє їх і передає їх на нижній рівень, який обробляє менші порції і т.д. І потім нижній рівень генерує одну вибірку. Така архітектура дає набагато менш зрозумілі результати, але працює швидко.

WaveNet, будучи ядром Google Cloud Text-to-Speech, являє собою згорткову нейронну мережу, яка приймає оцифрований необроблений звуковий сигнал в якості вхідного сигналу, який потім проходить через ці шари згортки і видає зразок сигналу. Незважаючи на те, що WaveNet близький до досконалості за своєю зрозумілістю і природністю, він є неприйнятно повільним (на створення 1 секунди звуку йде близько 4 хвилин).

Нарешті, нова хвиля наскрізного навчання принесла модель Google Tacotron, яка вчиться синтезувати мову безпосередньо з (текстових, аудіо) пар. Він приймає символи тексту в якості вхідних даних і передає їх через різні підмодулі нейронної мережі і генерує спектрограму аудіо.

На ринку представлено багато різних синтезаторів тексту, як від одиничних розробників, так і від таких великих гравців. Розглянемо сервіси по перетворенню

тексту в мову від Amazon, Microsoft і Google. Кожен з цих технологічних гігантів має свої синтезатори і голоси, які вони надають користувачам через своє API.

Amazon Polly - це хмарний сервіс, який перетворює текст в реальну мову. Amazon Polly підтримує кілька мов і включає в себе різноманітні реалістичні голоси. Оплата в Amazon Polly відбувається тільки за синтезований текст. Можна кешувати і відтворювати сгенеровану мову Amazon Polly без додаткових витрат.

Крім того, Amazon Polly включає в себе кілька голосів Neural Text-to-Speech (NTTS), пропонуючи тим самим природний Text-to-Speech голос. Технологія Neural TTS також підтримує стиль говорити мовою Newscaster, яка адаптована до сценаріїв використання новин.

Поширені випадки використання Amazon Polly: мобільні додатки, такі як програми для читання новин, ігри, платформи електронного навчання, додатки для спеціальних можливостей для людей з порушеннями зору і швидко зростаючий сегмент Інтернету речей (IoT).

Ось деякі з переваг використання Amazon Polly:

- висока якість - Amazon Polly пропонує як нову нейронну TTS, так і кращу в своєму класі стандартну технологію TTS для синтезу чудової природної мови з високою точністю вимови (включаючи аббревіатури, розширення аббревіатур, інтерпретацію дати / часу і усунення неоднозначності географічних даних);
- низька затримка - Amazon Polly забезпечує швидкі відповіді, що робить його життєздатним варіантом для випадків використання з низькою затримкою, таких як діалогові системи;
- підтримка великого асортименту мов і голосів - Amazon Polly підтримує десятки голосів на різних мовах, пропонуючи варіанти чоловічого і жіночого голосу для більшості мов. В даний час Neural TTS підтримує три голоси британської англійської та вісім голосів американської англійської. Це число буде продовжувати рости в міру того, як ми будемо додавати більше нейронних голосів. Американські англомовні голоси Метью і Джоанна також можуть використовувати

стиль мови Neural Newscaster, подібний до того, що ви можете почути від професійного ведучого новин.

- рентабельність - модель оплати за використання Amazon Polly означає, що витрати на настройку відсутні. Можна почати з малого і збільшувати масштаб по мірі зростання нашого застосування.

- хмарне рішення - рішення TTS на пристрої вимагають значних обчислювальних ресурсів, зокрема, потужності процесора, оперативної пам'яті і дискового простору. Це може привести до більш високих витрат на розробку і більшого енергоспоживання на таких пристроях, як планшети, смартфони і т.д. Навпаки, перетворення TTS, що виконується в хмарі AWS, значно знижує вимоги до локальних ресурсів. Це дозволяє підтримувати всі доступні мови і голоси з максимально можливою якістю. Більш того, оновлення та покращення голосів моментально доступні для всіх користувачів без необхідності оновлення ПЗ на пристроях.

Також, що дуже важливо для нашої роботи, синтезатор Amazon Polly підтримує деякі модифікації для своїх голосів, такі як шепіт і режим ведучого новин. Застосовуються ці модифікації за допомогою кастомних SSML тегів.

Google Cloud Text-to-Speech перетворить текст в людську мову в більш ніж 180 голосів на більш ніж 30 мовах. Він застосовує потужні нейронні мережі Google для надання високоякісного звуку.

Відділ DeepMind провів інноваційне дослідження моделей машинного навчання, щоб генерувати мову, яка імітує людські голоси і звучить природніше, скорочуючи розрив з людськими характеристиками на 70%.

Cloud Text-to-Speech пропонує ексклюзивний доступ до більш ніж 90 голосам WaveNet і буде продовжувати додавати більше з часом.

Також даний сервіс підтримує будь-який додаток або пристрій, який може відправляти запит REST або gRPC, включаючи телефони, ПК, планшети й пристрої IoT (наприклад, автомобілі, телевізори, динаміки).

Microsoft Cognitive Services Text to Speech - ще один сервіс для перетворення тексту в мову. У його арсеналі понад 75 голосів на 45-ти мовах, також є і нейронні

голосу. У його переваги можна записати асинхронну генерацію довгих аудіофайлів, використання SSML для позначення конкретного голосу конкретного тексту. Тобто ми можемо за один запит в API отримати аудіофайл з декількома голосами, на відміну від конкурентних сервісів від Google і Amazon. Також Microsoft Cognitive Services Text to Speech підтримує кілька модифікацій для своїх голосів, в числі яких веселий голос і режим ведучого новин. Застосовуються ці модифікації за допомогою кастомних SSML тегів.

1.4 Методи управління інтонаціями штучного голосу

Для управління інтонаціями голосів, що генеруються, ми будемо використовувати розмітку SSML [12] і зокрема її тег `prosody`. Вона підтримується більшістю сервісів генерації комп'ютерних голосів, але не всі сервіси підтримують тег `prosody`, наприклад її не підтримує Yandex.Speechkit.

SSML є частиною більшого набору специфікацій розмітки для голосових браузерів, розроблених в консорціумі W3C. Він призначений для надання багатій мови розмітки на основі XML для допомоги в створенні штучної мови в Web та інших додатках. Основна роль мови розмітки полягає в тому, щоб надати авторам синтезованого контенту стандартний спосіб управління такими аспектами мовного виведення, як вимова, гучність, висота, швидкість і т.д. на різних платформах з підтримкою синтезу.

Передбачуване використання SSML полягає в поліпшенні якості синтезованого контенту. Різні елементи розмітки впливають на різні етапи процесу синтезу. Розмітка може бути створена або автоматично, наприклад, за допомогою XSLT або CSS3 з документа XHTML, або вручну людиною. Розмітка може бути присутнім в повному документі SSML або як частина фрагмента, вбудованого в іншу мову, хоча взаємодія з іншими мовами не вказується як частина самого SSML. Велика частина розмітки, включеної в SSML, підходить для використання

більшістю розробників контенту; проте деякі розширені функції, такі як фонема і просодія (наприклад, для проектування мовного контуру) можуть зажадати спеціальних знань.

Елемент просодії дозволяє контролювати висоту звуку, швидкість мови і гучність мовного виведення. Теоретично, за допомогою цих параметрів ми зможемо надати голосу, що генериується інтонацію. Розглянемо деякі його атрибути:

- pitch - основний тон для генерованого голосу. Хоча точне значення «Основного тону» буде варіюватися в залежності від синтезатора, збільшення / зменшення цього значення зазвичай збільшує / зменшує приблизний тон голосу на виході. Можна вибрати зі значень число, за яким слід «Hz», відносно зміна або «x-low», «low», «medium», «high», «x-high» або «default». Мітки від «x-low» до «x-high» представляють собою послідовність монотонно неубутних рівнів висоти тону;

- rate - зміна швидкості мови для вмісту тексту. Можна вибрати зі значень не негативний відсоток або «x-slow», «slow», «medium», «fast», «x-fast» або «default». Мітки від «x-slow» до «x-fast» представляють собою послідовність монотонно неубутних швидкостей мови. Коли значення є невід'ємним відсотком він діє як множник значення за замовчуванням. Наприклад, значення 100% означає відсутність змін в швидкості мови, значення 200% означає збільшення швидкості мови в два рази відносно швидкості за замовчуванням, а значення 50% означає швидкість мови, рівну половині швидкості мови за замовчуванням. Швидкість за замовчуванням для голосу залежить від мови та діалекту, а також від індивідуальних характеристик голосу. Швидкість за замовчуванням для голосу повинна бути такою, щоб вона сприймалася як нормальна швидкість мови для голосу при читанні тексту вголос. Оскільки голоси залежать від синтезатора, швидкість за замовчуванням також буде залежати від нього.

- volume - гучність для міститься тексту. Допустимі значення - число, перед яким стоїть «+» або «-», за яким відразу слід «dB»; або «silent», «x-soft», «soft», «medium», «loud», «x-loud» або «default». За замовчуванням це + 0.0dB. Завдання

значення «silent» означає вказівку мінус нескінченність децибел (dB). Мітки від «silent» до «x-loud» представляють собою послідовність монотонно неубутних рівнів гучності.

2 РОЗРОБКА КОРИСТУВАЛЬНИЦЬКОГО ІНТЕРФЕЙСУ СИСТЕМИ УПРАВЛІННЯ СИНТЕЗАТОРОМ МОВИ

2.1 Обґрунтування пропонованого підходу управління голосовими інтонаціями комп'ютерного голосу

При виборі методу вирішення цього завдання виходимо з того, що додаткова зручність для прослуховування аудіо для користувача не повинна створювати значних незручностей на стороні творців контенту. В результаті проведеного дослідження, був зроблений висновок, що найзручнішим способом емоційного забарвлення мови для творців контенту буде використання звичних сценаріїв передачі інтонацій, характерних для месенджерів та соціальних мереж - за допомогою символів Емої.



Рисунок 2.1 – Статистика використання символів Емої

Ряд досліджень був присвячений вивченню взаємодії людей за допомогою Емої [14,15,16]. Опис соціальних і культурних значень, які Емої принесли в наше життя, дозволили зрозуміти які емоції ховаються за тими чи іншими символами. Якщо в кінці речення стоїть символ "особи, що плаче" (😓), значить автор хотів передати емоцію смутку в цьому повідомленні. А символ "червоної особи" (😡) передає злість або обурення автора. За допомогою величезної кількості найрізноманітніших емоційних символів Емої ми можемо з великою ймовірністю підібрати саме ту емоцію, яка потрібна нам в реченні для передачі правильного емоційного забарвлення.

Але як же нам змінювати інтонацію голосу? Майже всі великі сервіси генерації голоси підтримують мову розмітки синтезу речі (Speech Synthesis Markup Language або SSML). Більшість з популярних сервісів підтримують універсальний тег `prosody`. Він дозволяє контролювати висоту тону, швидкість мови і гучність. За допомогою цих трьох параметрів, а також спеціальних тегів синтезаторів, ми постараємося відтворити різні емоційні забарвлення. Для цього нам необхідно перетворити інтонації в поєднання певного тону, швидкості і гучності мовлення. Наприклад, параметри для розчарованого і сумного відтінку: `volume = "+ 40dB"`, `rate = "slow"` і `pitch = "+ 30%"`.

Исходный текст	Полученный SSML
Привет. Как дела? А у меня сегодня потерялся телефон 😓	<pre> <prosody><p><s>Привет. Как дела?</s></s><prosody volume="+40dB" rate="slow" pitch="+30%">А у меня сегодня потерялся телефон</prosody></s></p></prosody> </pre>

Рисунок 2.2 – Приклад перетворення тексту в SSML код

Для наочності роботи нашого інтерфейсу, ми повинні якось його реалізувати. Було вирішено написати програму, яка буде перетворювати повідомлення в телеграм каналах в їх аудіоверсії.

Одним з останніх популярних проектів в сфері голосової генерації є Частушки Бот. З його допомогою користувачі можуть згенерувати свої частівки - для цього необхідно ввести свій текст, розставити коми, наголоси, а бот підставляє музику і зачитує його у вигляді частівки. Даний бот з'явився влітку 2018 року, після чого його стали використовувати і в месенджері, і в інших соцмережах. Працює він на основі системи Yandex.Speechkit, але голос під мелодії не підлаштовується. Музика при цьому одна на вибір - класична пісня під частівки. У своїй роботі будемо використовувати близький підхід - отримувати текст користувача, парсити його і видавати аудіофайл на виході, який відправиться в телеграм канал.

Telegram - кроссплатформне додаток для передачі повідомлень і медіафайлів різних форматів. Воно має понад 400 мільйонів активних користувачів на 2020 рік, а кожен день аудиторія збільшується більш ніж на півтора мільйона людей [17]. Одна з функцій месенджера - можливість створення публічних каналів.

Канали - це інструмент для трансляції публічних повідомлень великій аудиторії. Вони пропонують унікальну можливість безпосередньо зв'язатися з людьми, відправляючи повідомлення на свої телефони з кожним постом. Оновлення в каналах приходять на телефон як приватні повідомлення, а зворотний зв'язок відсутній - тут немає ні коментарів, ні лайків. Відсутність зворотного зв'язку не дає автору скотитися в популізм, залишаючись самим собою. Тим самим між автором і читачем відбувається більш інтимна взаємодія, ніж на інших майданчиках з блогами.

Також, досить близька реалізація нашого підходу існує в інстаграмі і твіттері - це хештеги, додані до постів. Користувачі самі наносять розмітки у вигляді хештегів в свої повідомлення, щоб збільшити їх конверсію, кількість кліків. У розроблюваному додатку пропонується, щоб користувачі також наносили розмітку у вигляді символів емоїї, а на виході отримували аудіофайли зі згенерованою промовою. Хештег - одна з найпопулярніших функцій інстаграма, статистика за

кількістю хештегів представлена нижче на скріншоті завдяки сервісу Top-Hashtags.com.

1	#love	1.823B
2	#instagood	1.143B
3	#fashion	806.8M
4	#photooftheday	791.5M
5	#beautiful	657.4M
6	#art	643.2M
7	#photography	575.4M
8	#happy	575.4M
9	#picoftheday	568.8M
10	#cute	564.9M
11	#follow	557.4M
12	#tbt	534.5M
13	#followme	526.4M

Рисунок 2.3 – Статистика використання хештегів в соціальній мережі Instagram

2.2 Емодзі

Емодзі (від яп. 絵 - картинка і 文字 - знак, символ) - мова ідеограм і смайликів, який використовується в електронних повідомленнях та веб-сторінках, а також самі піктограми. Ця графічна мова, де замість слів використовуються поєднання картинок, з'явилася в Японії і поширилася по всьому світу. Спочатку емодзі виглядали як мова піктограм, де символи використовувалися так само, як і в ASCII-Емотіконах, але мали більш широкий діапазон вживання, а самі іконки були стандартизовані в клавіатурі, яка стала доступна на мобільних пристроях. Деякі

символи емодзі мають специфічне значення в японській культурі: наприклад, бізнесмен, що кланяється, іорітен, біла квітка, що позначає «блискуче виконане домашнє завдання», або група символів для позначення популярної їжі: рамен, данго, онігирі і суші.

На відміну від емодзі, емотикони виникли як засіб зображення емоцій існуючими типографськими засобами з ініціативи користувачів. Тим не менш, деякі емодзі, що позначають особи, відносять до емотиконів

Перші емодзі створив в 1998 або 1999 році Сігетака Куріта - член групи, яка працювала над створенням платформи мобільного Інтернету i-mode, яка була запущена в мережі оператора NTT DoCoMo. Перший набір з 172 символів емодзі розміром 12×12 пікселів був розроблений в складі функцій обміну повідомленнями для i-mode з метою полегшення електронного спілкування і став характерною особливістю, що відрізняє цю платформу від інших служб [2].

Однак в 1997 році Ніколя Лауфрані [3], звернувши увагу на зростання популярності ASCII-емотикони в мобільних технологіях, почав експериментувати з анімованими смайликами [4], з метою створення барвистих значків, які відповідали б існували спочатку ASCII-емотиконів, що складається з простих знаків пунктуації, щоб удосконалити їх для більш інтерактивного використання в цифровому спілкуванні. З них Лауфрані створив перші графічні емотикони і склав онлайн-словник емотикони [5], розбитий на окремі категорії: «Класичні», «Емоції», «Прапори», «Свята», «Розваги», «Спорт», «Погода», «Тварини», «Їжа», «Національності», «Професії», «Планети», «Зодіак», «Немовлята»; вперше ці зображення були зареєстровані в 1997 році в Бюро реєстрації авторських прав США, згодом ці значки були розміщені у вигляді файлів .gif в Інтернеті в 1998 році і стали першими в історії графічними Емотикони, застосовуваними в технологіях [6]. У 2000 році «Каталог емотикони», створений Лауфрані, став доступний в Інтернеті для скачування користувачами на стільникові телефони через сайт smileydictionary.com, на якому було зібрано понад 1000 графічних емотикони-смайликів і їх ascii-версій. Цей же каталог згодом, в 2002 році, був опублікований в книзі, що вийшла у видавництві Marabout під назвою «Dico Smileys» [7]. У 2001

році компанія Smiley початку ліцензувати права на використання графічних емотикони Лауфрані при скачуванні емотикони на стільникові телефони різними телекомунікаційними компаніями, в числі яких були Nokia, Motorola, Samsung, SFR (vodafone) і Sky Telemedia.

Три провідні оператори зв'язку в Японії (NTT docomo, SoftBank Mobile і au) вводили свої варіанти клавіатури емодзі.

Хоча спочатку емодзі були доступні тільки в Японії, деякі їх набори були включені в Юнікод, що дозволило використовувати їх і в інших країнах. В результаті в деяких смартфонах з операційною системою Windows Phone і в серії iPhone від Apple був дозволений доступ до емодзі. Вони також з'явилися в поштовому сервісі Gmail з квітня 2009 року [8]. Операційна система для комп'ютерів виробництва Apple Mac OS X підтримує емодзі, починаючи з версії Mac OS X 10.7 зі шрифтом Apple Color Emoji [9]. В даний час багато популярні додатки для миттєвого обміну повідомленнями, такі як WhatsApp, Telegram, Hangouts, Discord, Skype і ВКонтакте [10], також дозволяють використовувати набори емодзі.

Android-пристрої підтримують емодзі по-різному в залежності від версії операційної системи. Google додав вбудовану підтримку емодзі до клавіатури Google в листопаді 2013 року в усі версії Android, починаючи з Android 4.4 [11]. Деякі додатки для обміну повідомленнями, створені для мобільних операційних систем Android [12] сторонніми розробниками, також мають модулі, що дозволяють використовувати емодзі.

У системах Windows емодзі підтримуються шрифтом Segoe UI Emoji. Починаючи з версії Windows 8.1 підтримуються і кольорові емодзі [13].

Існують вільні безкоштовні шрифти з повною підтримкою емодзі - наприклад, Symbola [14].

Відображення емодзі на різних пристроях відбувається по-різному, що часом викликає непорозуміння між користувачами [15].

На таких операційних системах, як Windows 2000, Windows XP, Windows Vista і Windows 7, деякі емодзі не показуються коректно - буде показаний тільки невеликий квадратик.

2.3 Створення бота в Телеграм

Перш ніж починати розробку, бота необхідно зареєструвати і отримати його унікальний id, який є одночасно і токеном. Для цього в Telegram існує спеціальний бот - @BotFather.

Пишемо йому / start і отримуємо список всіх його команд.

Перша і головна - / newbot - відправляємо йому і бот просить придумати ім'я нашого нового боту. Єдине обмеження на ім'я - воно повинно закінчуватися на «bot». У разі успіху BotFather повертає токен бота і посилання для швидкого додавання бота в контакти, інакше доведеться поламати голову над ім'ям.

Для початку роботи цього вже достатньо. Особливо педантичні можуть вже тут привласнити боту аватар, опис та вітальне повідомлення.

Не забудьте перевірити отриманий токен за допомогою посилання api.telegram.org/bot<TOKEN>/getMe, кажуть, не завжди працює з першого разу.

Створювати бота будемо на Python3, проте завдяки адекватності цієї мови алгоритми легко переносяться на будь-яку іншу.

Telegram дозволяє не робити вивантаження повідомлень вручну, а поставити webHook, і тоді вони самі будуть надсилати кожне повідомлення. Для Python, щоб не морочитися з cgi і потоками, зручно використовувати який-небудь реактор, тому для реалізації оберемо tornado.web. (Для GAE зручно використовувати зв'язку Python2 + Flask)

```

URL = "https://api.telegram.org/bot%s/" % BOT_TOKEN
MyURL = "https://example.com/hook"

api = requests.Session()
application = tornado.web.Application([
    (r"/", Handler),
])

if __name__ == '__main__':
    signal.signal(signal.SIGTERM, signal_term_handler)
    try:
        set_hook = api.get(URL + "setWebhook?url=%s" % MyURL)
        if set_hook.status_code != 200:
            logging.error("Can't set hook: %s. Quit." % set_hook.text)
            exit(1)
        application.listen(8888)
        tornado.ioloop.IOLoop.current().start()
    except KeyboardInterrupt:
        signal_term_handler(signal.SIGTERM, None)

```

Рисунок 2.1. – Каркас боту

Тут при запуску бота встановлюється вебхук на адресу і відловлюється сигнал виходу, щоб повернути поведінку з ручним вивантаженням подій.

Додаток торнадо для обробки запитів приймає клас `tornado.web.RequestHandler`, в якому і буде логіка бота.

```

class Handler(tornado.web.RequestHandler):
    def post(self):
        try:
            logging.debug("Got request: %s" % self.request.body)
            update = tornado.escape.json_decode(self.request.body)
            message = update['message']
            text = message.get('text')
            if text:
                logging.info("MESSAGE\t%s\t%s" % (message['chat']['id'],
text))

                if text[0] == '/':
                    command, *arguments = text.split(" ", 1)
                    response = CMD.get(command, not_found)(arguments, me
ssage)

                    logging.info("REPLY\t%s\t%s" % (message['chat']['i
d'], response))

                    send_reply(response)
        except Exception as e:
            logging.warning(str(e))

```

Рисунок 2.2 – Логіка бота

Тут CMD - словник доступних команд, а send_reply - функція відправки відповіді, яка на вхід приймає вже сформований об'єкт Message.

Власне, її код досить простий (рис. 2.3).

```

def send_reply(response):
    if 'text' in response:
        api.post(URL + "sendMessage", data=response)

```

Рисунок 2.3 – Логіка бота (продовження)

Тепер, коли вся логіка бота описана можна почати придумувати йому команди.

Насамперед, необхідно дотримати угоду Telegram і навчити бота двом командам: / start і / help:

```
def help_message(arguments, message):
    response = {'chat_id': message['chat']['id']}
    result = ["Hey, %s!" % message["from"].get("first_name"),
              "\rI can accept only these commands:"]
    for command in CMD:
        result.append(command)
    response['text'] = "\n\t".join(result)
    return response
```

Рисунок 2.4 – Структура message

Структура message ['from'] - це об'єкт типу User, вона надає боту інформацію як id користувача, так і його ім'я. Для відповідей ж корисніше використовувати message ['chat'] ['id'] - в разі особистого спілкування там буде User, а в разі чату - id чату. В іншому випадку можна отримати ситуацію, коли користувач пише в чат, а бот відповідає в личку.

Команда / start без параметрів призначена для виведення інформації про боті, а з параметрами - для ідентифікації. Корисно її використовувати для дій, які потребують авторизації.

Після цього можна додати якусь свою команду, наприклад, / base64.

```
def base64_decode(arguments, message):
    response = {'chat_id': message['chat']['id']}
    try:
        response['text'] = b64decode(" ".join(arguments).encode("utf8"))
    except:
        response['text'] = "Can't decode it"
    finally:
        return response
```

Рисунок 2.5 – Додавання своєї команди

Як можна було помітити, Telegram надсилає повідомлення цілком, а не розбите, і обмеження на те, що команди починаються зі слешу - тільки для зручності мобільних користувачів.

Щоб бот отримував всі повідомлення в групах пишемо @BotFather команду /setprivacy і вимикаємо приватність.

Для початку в Handler додаємо обробник.

```
if text[0] == '/':
    ...
else:
    response = CMD("<speech>")(message)
    logging.info("REPLY\t%s\t%s" % (message['chat']['id'], response))
    send_reply(response)
```

Рисунок 2.6 – Обробник в Handler

А потім в список команд додаємо псевдо-мову.

```
RESPONSES = {
    "Hello": ["Hi there!", "Hi!", "Welcome!", "Hello, {name}!"],
    "Hi there": ["Hello!", "Hello, {name}!", "Hi!", "Welcome!"],
    "Hi!": ["Hi there!", "Hello, {name}!", "Welcome!", "Hello!"],
    "Welcome": ["Hi there!", "Hi!", "Hello!", "Hello, {name}!"],
}

def human_response(message):
    leven = fuzzywuzzy.process.extract(message.get("text", ""), RESPONSES.keys(), limit=1)[0]
    response = {'chat_id': message['chat']['id']}
    if leven[1] < 75:
        response['text'] = "I can not understand you"
    else:
        response['text'] = random.choice(RESPONSES.get(leven[0])).format_map(
            {'name': message["from"].get("first_name", "")}
        )
    return response
```

Рисунок 2.7 – Додавання псевдо-мови в команди

Тут емпірична константа 75 відносно непогано відображає ймовірність того, що користувач все-таки хотів сказати. А format_map - зручна для однакового опису

рядків як вимагають підстановки, так і без неї. Тепер бот буде відповідати на вітання і іноді навіть звертатися по імені.

Боти, як і будь-який нормальний користувач Telegram, можуть не тільки писати повідомлення, а й ділитися картинками, музикою, стікерами.

Для прикладу розширимо словник RESPONSES.

```
RESPONSES["What time is it?"] = ["<at_sticker>", "{date} UTC"]
```

Рисунок 2.8 – Словник RESPONSES.

І будемо відловлювати текст <at_sticker>.

```
if response['text'] == "<at_sticker>":
    response['sticker'] = "BQADAgADeAcAA1Ox9wOjY2jpAAHq9DUC"
    del response['text']
```

Рисунок 2.9 – Відловлення тексту <at_sticker>

Видно, що тепер структура Message вже не містить текст, тому необхідно модифікувати send_reply.

```
def send_reply(response):
    if 'sticker' in response:
        api.post(URL + "sendSticker", data=response)
    elif 'text' in response:
        api.post(URL + "sendMessage", data=response)
```

Рисунок 2.10 – Модифікування send_reply

Завдяки зручності API і швидкому старту боти Telegram можуть стати гарною платформою для автоматизації своїх дій, налаштування повідомлень, створення вікторин і task-based змагань (CTF, DozoR та інш.).

На жаль, на даний момент існує обмеження на використання webHook - він працює тільки по https і тільки з дійсним сертифікатом.

На щастя, Telegram також вміє працювати і по ручному оновленню, тому не змінюючи коду можна створити ще одну службу Puller, яка буде викачувати їх і слати на локальну адресу.

```
while True:
    r = requests.get(URL + "?offset=%s" % (last + 1))
    if r.status_code == 200:
        for message in r.json()["result"]:
            last = int(message["update_id"])
            requests.post("http://localhost:8888/",
                           data=json.dumps(message),
                           headers={'Content-type': 'application/json',
                                   'Accept': 'text/plain'})
    else:
        logging.warning("FAIL " + r.text)
        time.sleep(3)
```

Рисунок 2.11 – Створення служби Puller

3 АРХІТЕКТУРА І РЕАЛІЗАЦІЯ СИСТЕМИ УПРАВЛІННЯ СИНТЕЗАТОРОМ МОВИ З ВИКОРИСТАННЯМ СИМВОЛІВ ЕМОЛІ

Для початку ми повинні скласти архітектуру нашого застосування. Для того, щоб згенерувати аудіо версію повідомлення в каналі нам необхідно:

- отримати повідомлення з телеграм каналу;
- перетворити повідомлення в SSML розмітку. Впровадити потрібні інтонації в залежності від наявності символів Емоїї;
- згенерувати аудіофайл з SSML за допомогою синтезатора;
- завантажити аудіофайл в телеграм канал.

Отримати повідомлення з телеграм каналу можна за допомогою будь-якого пристрою з клієнтом Telegram. Для цього заходимо на потрібний нам канал, вибираємо потрібне нам повідомлення і додаємо його в інтерфейс нашого застосування.

Перетворити повідомлення в SSML розмітку ми зможемо за допомогою вбудованих методів Ruby і використання шаблонів. Але тут ми зустрічаємося з двома проблемами. По-перше, кожна людина по різному використовує символи Емоїї. Хтось вставляє емоїї в кінець речення, хтось в початок. У кого-то емоїї надає забарвлення фразі до нього, у кого-то після.

Було проаналізовано найпопулярніші повідомлення в соцмережах, які містять символи емоїї. Більшість користувачів вставляють символи емоїї в кінці речення, надаючи емоційне забарвлення фразі, яка стоїть до емоїї. Також ми знайшли кілька публікацій, що описують гайдлайни до написання повідомлень з емоїї [19,20]. У реалізації алгоритму для складання SSML будемо відштовхуватися від цих спостережень. Другою проблемою є розбивка тексту на фрази, до яких застосовуватимуться емоційні забарвлення від символів емоїї. Спочатку була ідея просто розділяти текст на фрази по знаків пунктуації в кінці речень. Але тут виявилось одразу кілька недоліків - такі скорочення, як "тощо, тощо" та інші, сприймалися як кінець речення; речення з кількома знаками пунктуації в кінці, наприклад "?!?", втрачали частину сенсу і неправильно поділяли речення. Потім було використано бібліотеку `nlp-pure` [21] для поділу тексту, і вона вирішила більшу частину проблем.

Для генерації аудіофайлу з SSML будемо використовувати сервіс, описаний в попередньому параграфі - Amazon Polly. Для цього скористаємося офіційним SDK AWS для Ruby.

Ну і для того, щоб завантажити аудіофайл в канал нам також потрібно пристрій з клієнтом Telegram. Також користувач повинен мати права на відправлення повідомлень в цей канал. Ми прикріплюємо наш аудіофайл до повідомлення і відправляємо, все просто.

Або ж не просто? Для виконання першого і останнього пункту потрібно мати окремий пристрій і вручну здійснювати різні дії. Невже не можна якось автоматизувати і спростити їх? В цьому допоможуть боти в Телеграма та Telegram Bot API (Application Programming Interface).

Боти - спеціальні акаунти в месенджері Telegram. Вони автоматично обробляють і відправляють повідомлення. З ними можна взаємодіяти за допомогою повідомлень в звичайних або групових чатах. Логіка ботів контролюється за допомогою HTTPS (HyperText Transfer Protocol Secure) запитів до API (Application Programming Interface) для пошукових роботів.

Що ж можуть боти? Ось кілька прикладів їх використання:

- інтеграції з іншими сервісами - наприклад бот може відправляти повідомлення про якусь подію (розумний будинок);
- інструменти і утиліти - прогноз погоди, переклад текстів, створення опитувань і багато іншого, на що здатні боти;
- одно- і розраховані на багато користувачів ігри - користувачі можуть грати в HTML5-ігри (HyperText Markup Language) в групах або приватних чатах.

Було вирішено написати свого Telegram бота, який автоматично буде забирати кожне повідомлення з телеграм каналу, перетворювати його в аудіофайл, і потім відправляти його назад в канал.

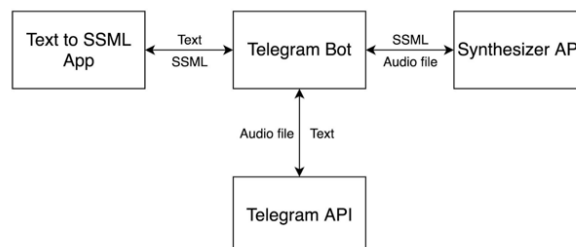


Рисунок 3.1 – Архітектура системи генерації аудіо-подкастів

Важливим етапом є вибір бібліотеки для роботи з API Telegram. Були розглянуті дві найпопулярніші бібліотеки: telegram-bot і telegram-bot-ruby. Було обрано telegram-bot-ruby [22] з кількох причин:

- telegram-bot-ruby є легковагою обгорткою для API. У той же час telegram-bot вдає із себе повноцінний фреймворк для створення телеграм-ботів. Можливості telegram-bot надлишкові для нашого застосунку;
- telegram-bot-ruby надає можливість використовувати проксі сервер для країн, в яких Telegram офіційно заблокований.

Тепер потрібно розібратися, як використовувати Telegram Bot API для вирішення першого і третього пункту. Необхідно отримувати повідомлення з каналу і відправляти аудіофайли в нього. Для отримання повідомлень використовується метод бібліотеки telegram-bot-ruby bot.listen. Він в свою чергу використовує метод setWebhook Telegram API для того, щоб встановити вебхук для нашого бота.

Webhook (також званий веб-зворотним викликом або HTTP push API) це спосіб для додатка надавати іншим програмам будь-яку інформацію в режимі реального часу. Webhook відправляє дані в інші програми, як тільки відбулася якась дія, тобто ви отримуєте дані негайно, на відміну від типових API, де вам потрібно дуже часто запитувати дані, щоб отримувати їх в реальному часі.

Щоб відправляти аудіофайли в телеграм канал, в Telegram API існують два методи - sendAudio і sendVoice. sendAudio відправляє аудіофайл так, щоб він запускався у вбудованому музичному плеєрі Телеграма. SendVoice ж маскує аудіофайл під аудіоповідомлення. Ми будемо використовувати sendAudio, так як тут потрібен формат аудіо mp3, на відміну від ogg. А також функція аудіоплеєра поліпшить досвід використання згенерованих аудіопідкаст.

По ходу розробки знайшлося ще кілька проблем. Одна з них - необхідність місця, для зберігання аудіофайлів до завантаження їх в телеграм канал. Через особливості платформи Heroku, а саме через ефемерність файлової системи, ми не можемо зберігати згенерований аудіофайл на одному диску, де і знаходиться бот.

Для вирішення цієї проблеми, було вирішено використовувати Amazon S3 для зберігання аудіофайлів там.

Друга проблема полягає в тому, що для кожної мови необхідно використовувати свій голос в синтезаторі. Немає універсального голосу, щоб можна було з його допомогою згенерувати мову на будь-якій мові. Значить для кожної мови необхідно підібрати певний голос. Було складено список, де визначено для кожної мови свій голос:

- Lotte – Dutch;
- Maxim – Russian;
- Salli – English;
- Marlene – German;
- Giorgio – Italian;
- Inês – Portuguese;
- Zeina – Arabic;
- Zhiyu – Chinese;
- Astrid – Swedish;
- Mia – Spanish;
- Carmen – Romanian;
- Ewa – Polish;
- Filiz – Turkish;
- Céline – French;
- Mads – Danish;
- Mizuki – Japanese;
- Liv – Norwegian;
- Seo Yeon – Korean.

Тепер потрібно визначити, якою мовою написано повідомлення, яке прийшло боту. Після недовгого пошуку був обраний сервіс по визначенню мови DetectLanguage [23]. Вибрали ми його з кількох причин:

- наявність безкоштовного плану з обмеженням в 1000 запитів в день. Для початку розробки цього буде більш ніж достатньо.

- офіційна бібліотека для мови програмування Ruby від розробника. Немає необхідності здійснювати REST запити до API сервісу безпосередньо.

Тепер можна писати код. Почнемо з самого телеграм бота. По-перше, необхідно створити файл, в якому буде зберігатися код. Назвемо його `bot.rb`. Потім необхідно підключити бібліотеки, необхідні для роботи.

- 'telegram / bot' для роботи з Telegram Bot API;
- 'dotenv / load' для установки змінних в робочому середовищі;
- 'aws-sdk-core', 'aws-sdk-polly' і 'aws-sdk-s3' для використання сервісів Amazon S3 і Amazon Polly;
- 'detect_language' для визначення мови повідомлення.

Потім необхідно налаштувати змінні оточення для розроблюваного застосунку. Ми повинні налаштувати: API ключ, секретний ключ і назва для Amazon Polly і Amazon S3, API ключ для DetectLanguage, API ключ для Telegram Bot API. У production-оточенні їх можна налаштувати за допомогою налаштувань проекту в Heroku, в development-оточенні з цим допоможе бібліотека `dotenv`. Було створено файл `.env` в кореневій папці проекту, заповнений потрібними змінними.

Встановлюємо ключі і змінні в налаштування бібліотек.

Лістинг 3.1 - Ключі і змінні в налаштування бібліотек

```
Aws.config.update( region: 'eu-west-1',
  credentials:      Aws::Credentials.new(ENV['S3_KEY'],
ENV['S3_SECRET'])
)

DetectLanguage.configure do |config|
  config.api_key = ENV['DETECT_LANGUAGE_KEY'] end

S3_BUCKET = Aws::S3::Resource.new.bucket(ENV['S3_BUCKET'])

token = ENV['TELEGRAM_BOT_API_TOKEN']
Telegram::Bot::Client.run(token, logger: Logger.new($stderr)) do
|bot|
  ... end
```

Також необхідно ініціалізувати ще 2 змінних - словник, який містить пари ключ-значення з кодом мови і назвою мови з Amazon Polly.

Лістинг 3.2 - Словник, який містить пари ключ-значення

```
language_map = { "nl-NL": "Lotte",
"ru-RU": "Maxim",
"en-US": "Salli",
"en-GB-WLS": "Geraint",
"es-US": "Miguel",
"de-DE": "Marlene",
"it-IT": "Giorgio",
"pt-PT": "Ines",
"arb": "Zeina",
"cmn-CN": "Zhiyu",
"cy-GB": "Gwyneth",
"is-IS": "Karl",
"es-ES": "Lucia",
"sv-SE": "Astrid",
"es-MX": "Mia",
"pt-BR": "Vitoria",
"en-IN": "Raveena",
"fr-CA": "Chantal",
"en-GB": "Amy",
"en-AU": "Russell",
"ro-RO": "Carmen",
"pl-PL": "Ewa",
"tr-TR": "Filiz",
"fr-FR": "Celine",
"da-DK": "Mads",
"ja-JP": "Mizuki",
"nb-NO": "Liv",
"ko-KR": "Seoyeon"
```

```
polly = Aws::Polly::Client.new
```

Ініціалізуємо бота за допомогою бібліотеки telegram-bot-ruby.

Лістинг 3.3 – Ініціалізація бота

```
Telegram::Bot::Client.run(token, logger: Logger.new($stderr)) do
|bot|
...
End
```

Треба вказати боту, щоб він прослуховував всі канали на наявність нових повідомлень, доступ до яких він має.

Лістинг 3.4 – Вказівка боту

```
bot.listen do |message|
... end
```

Визначаємо мову повідомлення, використовуючи сервіс DetectLanguage і їх бібліотеку. Скористаємося методом `simple_detect`, який повертає тільки код мови. А потім вибираємо потрібний нам голос Amazon Polly за цим кодом.

Лістинг 3.5 – Визначення мови повідомлення

```
language = DetectLanguage.simple_detect(message.text) voice =
language_map[language]
```

Тепер потрібно згенерувати SSML код з нашого прийшов повідомлення. І після цього ми нарешті зможемо згенерувати аудіоподкаст. Для зручності коду, винесемо логіку генерації SSML в окремий клас, а в боті тільки викличемо метод з нього. Реалізацію класу опишемо нижче.

Лістинг 3.6 - Логіка генерації SSML

```
ssml_message = SSMLTranslator.new(message.text).call
response      = polly.synthesize_speech(text: ssml_message,
output_format: :mp3, voice_id: voice, text_type: "ssml", engine:
:standard)
```

У змінну `response` ми отримали наш згенерований аудіофайл в бінарному вигляді. Тепер нам необхідно згенерувати назву для нашого аудіофайли, зберегти його в тимчасовий файл і відправити в Amazon S3. Файл складемо з поточної дати, часу і перших слів повідомлення.

Лістинг 3.7 – Тимчасовий файл

```
media_tmp      = Tempfile.open("#{SecureRandom.hex}_tmp.mp3")
media_tmp.write(response.audio_stream.read)
media_name = "#{message.text.split(' ')[0..2].join('
')}-#{Time.now.utc.strftime('%F')}.mp3"

S3_BUCKET.object(media_name).put( body: media_tmp,
content_type: 'audio/mpeg'
)
```

Нам залишилося отримати пряме посилання на аудіофайл з S3 і за допомогою Telegram Bot API відправити цей файл в канал, звідки ми і отримали це повідомлення.

Лістинг 3.8 - Відправка файлу в Telegram канал

```
media_file_remote_url
S3_BUCKET.object(media_name).presigned_url(:get, expires_in: 120)

bot.api.send_audio(chat_id: "@#{message.chat.username}", audio:
media_file_remote_url)
```

Опишемо клас SSMLTranslator і його метод `generate_ssml`. На вхід він повинен отримувати текст з повідомлення, а на виході повертати рядок з SSML розміткою. Ініціалізуємо змінні, в яких будемо зберігати пари з масиву з символами емоїї одного значення і фрагментами шаблону.

```
emoji_template_list = [
  [
    ['😞', '😓', '😭', '😔', '😫', '😩', '😪', '😴'], template_hash[:sad]
  ],
  [
    ['😏', '😌'], template_hash[:whispering]
  ],
  [
    ['😱', '😨', '😧', '😦', '😤', '😡'], template_hash[:scared]
  ],
  [
    ['😊', '😄', '😁', '😃', '😅', '😆', '😇', '😈', '😉'], template_hash[:happy]
  ],
  [
    ['😡', '😠', '😤', '😥', '😦', '😧'], template_hash[:angry]
  ]
]
```

Рисунок 3.2 - Пари з масиву з символами емоїї одного значення

У `template_hash` ми будемо зберігати пари, ключем буде емоція, яка позначає сенс шаблону, а в значенні буде масив з частин самого шаблону.

Лістинг 3.9 - Template_hash

```

template_hash = { sad: [
    "<s><prosody    volume='loud'    rate='slow'    pitch='+30%>",
"</prosody></s>"
],
  whispering: [
    "<s><amazon:effect                                name='whispered'>",
"</prosody></amazon:effect></s>"
],
  scared: [
    "<s><prosody    volume='x-soft'    rate='medium'    pitch='+12%>",
"</prosody></s>"
],
  happy: [
    "<s><amazon:emotion    name='excited'    intensity='medium'>",
"</prosody></amazon:effect></s>"
],
  angry: [
    "<s><prosody    volume='x-loud'    rate='slow'    pitch='-14%>",
"</prosody></s>"
]
}

```

Потім ми повинні розбити текст на речення. Спочатку ми використовували просте регулярний вираз `/[.?!]/`, щоб розділяти текст по знаків пунктуації. Але через такого підходу виникли проблеми з деякими прикладами:

"Hello, Mr.Freeman. How are you doing?"

Якщо використовувати регулярку, то в результаті ми отримаємо поділ тексту на ["Hello, Mr.", "Freeman.", "How are you doing?"]. Ми могли б покращувати наше регулярний вираз, щоб враховувати все більше і більше кейсів, але ми вирішили скористатися бібліотекою `nlp-pure`. Вона використовує алгоритми NLP для поділу тексту на слова і пропозиції. При використанні даної бібліотеки даний приклад правильно розбивається на реченні: ["Hello, Mr.Freeman.", "How are you doing?"].

Надаємо змінної `chunks` масив з пропозиціями з тексту.

```
chunks = NlpPure::Segmenting::DefaultSentence.parse text
```

Потім ми повинні пройтися по всіх реченнях, визначити наявність емоутов в них і застосувати той чи інший емоційний окрас до них. Пройдемося по всіх пропозиціях в масиві `chunks` і складемо з них новий масив, тільки вже з потрібними SSML тегами. Для початку нам потрібно знайти символ емої в реченні і

перевірити, чи підтримуємо ми цю емоцію чи ні. Для цього розіб'ємо пропозицію на символи і співвіднесемо новий масив з масивом всіх підтримуваних символів емої. Якщо такого символу не знайшлося, тоді обернемо наші речення в SSML тег <s>, що позначає межі речення. Якщо ж такий символ є, то відправимо наші речення нову функцію `ssml_emoji`, яка буде обертати наш текст в потрібний шаблон.

В кінці ми з'єднаємо всі елементи нового масиву з реченнями, обгорнутими в SSML теги, в один рядок. І повертаємо з нашої функції цей рядок, обгорнутих в головний SSML тег <peak>.

Лістинг 3.10 - Визначення наявності емоутів

```
def generate_ssml
  chunks = NlpPure::Segmenting::DefaultSentence.parse text

  ssml_chunks = chunks.map do |chunk| splitted_sentence =
chunk.split('') emoji_in_text = (text & all_emojis).last
  return ssml_emoji(emoji_in_text, chunk) if emoji_in_text

  "<s>#{chunk}</s>" end.join

  "<peak>#{ssml_chunks}</peak>" end

  private

  def ssml_emoji(emoji, text)
    template = choose_template(emoji) template[0] + text +
template[1]
    end

  def choose_template(emoji) emoji_template_list.each do |pair|
return pair[1] if pair[0].include?(emoji) end
  end
```

Тепер потрібно протестувати нашого бота. Створюємо всі необхідні ресурси, проставляємо змінні в оточення. Запускаємо бота і намагаємося відправити повідомлення в канал.

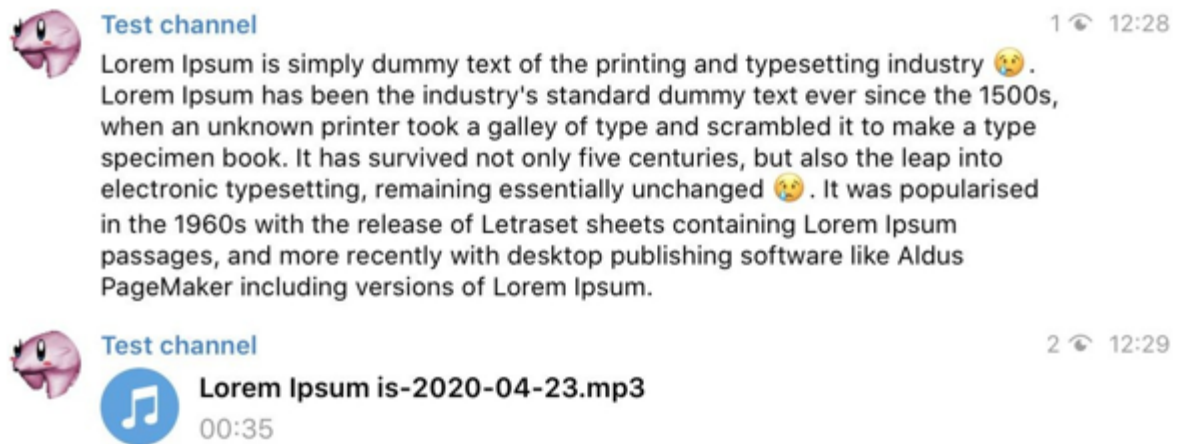


Рисунок 3.3 – Тестування бота в телеграм каналі

Як ми бачимо, бот працює як треба і генерує аудіо-подкаст з надісланого в канал повідомлення. Інтонація голосу регулюється символами емоїї в тексті.

ВИСНОВКИ

При вирішенні завдання по передачі голосових інтонацій в синтезуєму комп'ютерну мову відповідно до авторського змісту тексту, в роботі були досліджені різноманітні способи синтезування мови, розглянуті інструменти для зміни інтонацій в мові, що генерується, був проведений огляд найбільших сервісів по синтезуванню мови.

Було досліджено і вивчено вплив символів емої на соціум і сприйняття людиною емоцій, які ці символи передають. Був складений словник з угрупованням символів емої по переданим ними емоціям.

В роботі реалізовано інтерфейс перетворення тексту з символами емої в розмітку для синтезаторів, спроектовано і написано телеграм бот для генерації аудіо-підкастів з повідомлень каналу з урахуванням емоцій, переданих за допомогою емої.

Таким чином, задача вирішена в повному обсязі, було отримано спосіб передавати голосові інтонації синтезатору відповідно до задумки автора.

В майбутньому можливо удосконалити деякі аспекти інтерфейсу. Необхідні більш продуманий поділ тексту на частини, підтримка декількох символів емої в одному реченні, поділ пропозицій не розділовими знаками, а символами емої. Також можливо зробити витяг інтерфейсу з вихідного коду телеграм бота в окрему бібліотеку, яка буде по заданих параметрах (тип синтезатора, текст) генерувати SSML с емоціями. Також є ідея зробити конструктор для складання власних емоцій, де б кінцеві користувачі могли налаштовувати різні інтонації і підбирати до них символи емої самостійно.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. How audio is increasing engagement with digital news publishing, SpeechKit. URL:<https://medium.com/speechkit/how-audio-is-increasing-engagement-with-digital-news-publishing-c6b0ceb445e3> (дата звернення 13.05.2020)
2. The result of the research about podcasts, conducted by the Edison Research Company, 2019.
3. URL: <https://edisonresearch.com/infinite-dial-2019> (дата звернення 12.04.2021)
4. Исследование. Подкасты: аудитория, рейтинги подкастов и подкастеров. TIBURON Research, 19 февраля 2020.
5. URL:<https://tiburon-research.ru/cases/issledovanie-podkasty-auditoriya-reytingi-podkastov-i-podkasterov> (дата звернення 13.05.2021)
6. 2020 Podcast Stats & Facts (New Research From Apr 2020). Ross Winn, 21 апреля 2020.
7. URL:<https://www.podcastinsights.com/podcast-statistics/> (дата звернення 12.05.2021)
8. Nielsen Podcast Insights A Marketer's Guide To Podcasting, Q3 2017. URL:<https://www.nielsen.com/wp-content/uploads/sites/3/2019/04/nielsen-podcast-insights-q3-2017.pdf> (дата звернення 12.05.2021)
9. The result of the research about podcasts, conducted by the Edison Research Company, 2020.
10. URL:<https://www.edisonresearch.com/wp-content/uploads/2020/03/The-Infinite-Dial-2020-from-Edison-Research-and-Triton-Digital.pdf> (дата звернення 12.05.2021)

11. Nielsen Podcast Insights A Marketer's Guide To Podcasting, Q1 2018.
URL:<https://www.nielsen.com/wp-content/uploads/sites/3/2019/04/marketeres-gu ide-to-podcasting-march-2018.pdf>
12. Stevens C. et al. On-line experimental methods to evaluate text-to-speech (TTS) synthesis: effects of voice gender and signal quality on intelligibility, naturalness and preference. *Computer speech & language*, 2005. Т. 19. №. 2. с. 129-146.
13. Pisoni D. B., Nusbaum H. C., Greene B. G. Perception of synthetic speech generated by rule. *Proceedings of the IEEE*. 1985. Т. 73. №. 11. с. 1665-1676.
14. King S. A beginners' guide to statistical parametric speech synthesis //The Centre for Speech Technology Research, University of Edinburgh, UK. 2010.
15. Kuligowska K., Kisielewicz P., Włodarz A. Speech synthesis systems: disadvantages and limitations. *International Journal of Engineering & Technology*. 2018. Т. 7. №. 2.28. с. 234-239.
16. Taylor P., Isard A. SSML: A speech synthesis markup language. *Speech communication*. 1997. Т. 21. №. 1-2. с. 123-133.
17. Statistic on the use of the Emoji symbols in iPhone keyboard "emoji Xpress".
18. URL: <http://www.emojistats.org/> (дата звернення 12.04.2020)
19. Крылов Ю.В. Семантика эмодзи в виртуальном диалоге // Вестник Омского государственного педагогического университета. Гуманитарные исследования. 2017. №2 (15)
20. Stark L., Crawford K. The conservatism of emoji: Work, affect, and communication. *Social Media+ Society*. 2015. Т. 1. №. 2. с. 2056305115604853.
21. Ljubešić N., Fišer D. A global analysis of emoji usage. *Proceedings of the 10th Web as Corpus Workshop*. 2016. с. 82-89.
22. Statistics on the use of the Telegram messenger.
23. URL:<https://telegram.org/blog/400-million> (дата звернення 26.04.2021)
18. Statistics on the use of hashtags in Instagram social network.
URL:<https://top-hashtags.com/instagram/> (дата звернення 26.04.2021) 19.TIME Exclusive: Here Are Rules of Using Emoji You Didn't Know You Were

24. Following, Katy Steinmetz, July 17 2014
URL:<https://time.com/2993508/emoji-rules-tweets/> (дата звернення 12.05.2020)
25. The Definitive Guide to Emoji Punctuation, December 11 2019
URL:<https://blog.emojipedia.org/the-definitive-guide-to-emoji-punctuation/> (дата звернення 12.05.2021)
26. Natural language processing algorithms implemented in pure Ruby with minimal dependencies, GitHub Repository.
27. URL:<https://github.com/parhamr/nlp-pure> (дата звернення 16.05.2021)
28. Ruby wrapper for Telegram's Bot API, GitHub Repository.
29. URL: <https://github.com/atipugin/telegram-bot-ruby> (дата звернення 03.05.2020)
30. Language identification JSON webservice accepts text and returns detected language code and score.
31. URL: <http://gititis.kpfu.ru/ArthurGaliullin/VKR>
32. 25.Miller H. J. et al. “Blissfully Happy” or “Ready to Fight”: Varying Interpretations of Emoji //Tenth International AAAI Conference on Web and Social Media. 2016.

ДОДАТОК А

ЗБЕРЕЖЕННЯ ПАР, КЛЮЧЕМ ЯКИХ Э ЕМОЦІЯ, ЯКА ПОЗНАЧАЄ СЕНС ШАБЛОНУ

Лістинг А1 - Збереження пар, ключем яких э емоція, яка позначає сенс шаблону

```

template_hash = { sad: [
    "<s><prosody    volume='loud'    rate='slow'    pitch='+30%'",
"</prosody></s>"
    ],
    whispering: [
    "<s><amazon:effect                                name='whispered'",
"</prosody></amazon:effect></s>"
    ],
    scared: [
    "<s><prosody    volume='x-soft'    rate='medium'    pitch='+12%'",
"</prosody></s>"
    ],
    happy: [
    "<s><amazon:emotion    name='excited'    intensity='medium'",
"</prosody></amazon:effect></s>"
    ],
    angry: [
    "<s><prosody    volume='x-loud'    rate='slow'    pitch='-14%'",
"</prosody></s>"
    ]
}

```

ДОДАТОК В

З'ЄДНАННЯ ВСІХ ЕЛЕМЕНТІВ НОВОГО МАСИВУ З РЕЧЕННЯМИ, ОБГОРНУТИМИ В SSML ТЕГИ

Лістинг В.1 – З'єднання всіх елементів нового масиву з реченнями, обгорнутими в SSML теги

```
def generate_ssml
  chunks = NlpPure::Segmenting::DefaultSentence.parse text

  ssml_chunks = chunks.map do |chunk| splitted_sentence =
chunk.split('') emoji_in_text = (text & all_emojis).last
  return ssml_emoji(emoji_in_text, chunk) if emoji_in_text

  "<s>#{chunk}</s>" end.join

  "<speak>#{ssml_chunks}</speak>" end

private

def ssml_emoji(emoji, text)
  template = choose_template(emoji)  template[0]  +  text  +
template[1]
  end

def choose_template(emoji) emoji_template_list.each do |pair|
  return pair[1] if pair[0].include?(emoji) end
end
```

Технологічні підходи в комп'ютерному синтезі голосу

ПАРАМЕТРИЧНИЙ СИНТЕЗ

Плюси:

- збільшена природність звуку;
- гнучкість - легше змінити висоту тону для емоційних змін або використовувати адаптацію MLLR для зміни характеристик голосу;
- більш низькі витрати на розробку: для цього потрібно всього 2-3 години запису голосу актора, що сприяє меншій кількості записів, меншій базі даних і меншій обробці даних.

Мінуси:

- більш низька якість звуку з точки зору чіткості - є багато артефактів, що призводять до приглушеної мови, з постійно гуде, гучним звуком;
- голос може здаватися роботизованим - в TTS, заснованому на статистичній моделі, приглушений звук робить голос стабільним, але неприродним і роботизованим.

СИНТЕЗ ЗА ПРАВИЛАМИ

Плюси:

- добре поставлене, зрозуміле синтезоване мовлення навіть на високих швидкостях, відсутність акустичних глоків;
- менше залежить від вихідних мовних даних для виробництва вихідний мови;
- добре підходить для вбудованих систем, де пам'ять і потужність мікропроцесора обмежені.

Мінуси:

- низька природність - техніка створює штучну, що лунає роботом мова, яка далека від природної мови людини;
- складно розробити правила, які визначають синхронізацію джерела і динамічні значення всіх параметрів фільтра навіть для простих слів.

КОНКАТЕНАТИВНИЙ СИНТЕЗ

Плюси:

- високу якість звуку з точки зору чіткості;
- можливість збереження оригінального акторського голосу.

Мінуси:

- такі системи дуже трудомісткі, оскільки вимагають величезних баз даних і жорсткого кодування комбінації для формування цих слів;
- результуюча мова може звучати менш природно і беземоційно, тому що практично неможливо отримати аудіозаписи всіх можливих слів, вимовлених у всіх можливих комбінаціях емоцій, просодії, стресу і т. д.

ГІБРИДНІ ПІДХОДИ

Плюси:

- величезне поліпшення як з точки зору зрозумілості, так і природності;
- не вимагає великої попередньої обробки людиною і розробки функцій.

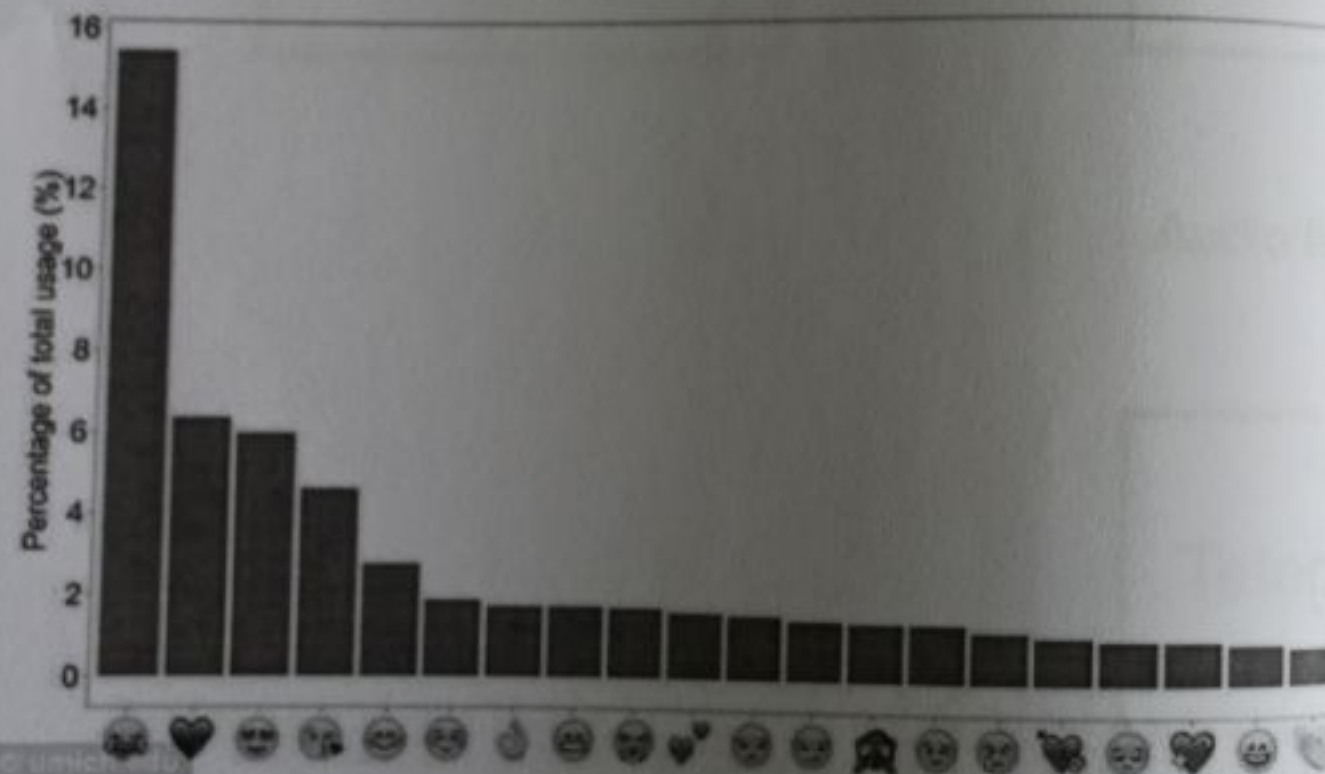
Мінуси:

- методи синтезу мови Deep Learning все ще вимагають досліджень.

					13.02070849.52730 ПЛ1		
					Технологічні підходи в комп'ютерному синтезі голосу		
Зам.	Лист	№ докум.	Підп.	Дат.	Лист	Маса	Масшт.
Розроб.		Дмитренко К.О.	Д.О.		Лист 1		Листів 4
Перев.		Грушко С.С.	С.С.	21.05			
Т.контр.							
Н.контр.		Щербак Н.В.	Н.В.				
Затв.		Кудерметов Р.К.	Р.К.				

НУ «Запорізька політехніка»
КНТ-527

Статистика використання етої

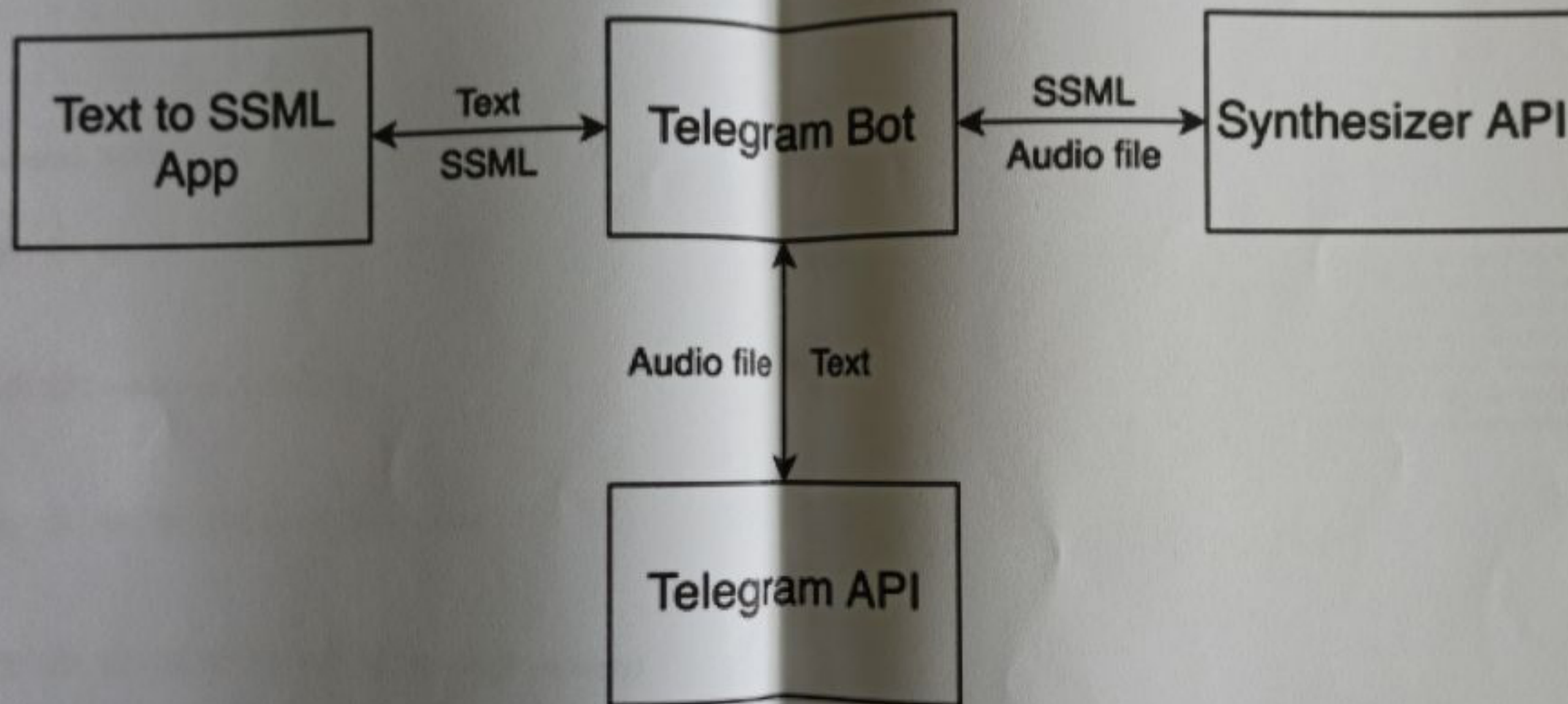


Emoji usage in text messages

	% emoji-msg	Top emoji
US	9.2	          
Brazil	5.1	          
Mexico	7.9	          
France	19.8	          
Spain	3.4	          
Turkey	5.8	          
Argentina	3.1	          
Indonesia	3.2	          
Russia	10.9	          
Colombia	3.7	          

					13.02070849.52730 ПЛ2							
					Статистика використання етожі				Лист		Маса	Масит.
Зам.	Лист	№ докум.	Підп.	Дата					Лист 2		Листів 4	
Розроб.		Дмитренко К.О.	<i>Ваша</i>									
Перев.		Грушко С.С.	<i>Ваша</i>	21.05								
Т.контр.												
Н.контр.		Щербак Н.В.	<i>Ваша</i>								НУ «Запорізька політехніка»	
Заст.		Куберметов Р.К.	<i>Ваша</i>								КНТ-527	

Архітектура системи керування синтезатором мови



					13.02070849.52730 ПЛЗ				
					Архітектура системи керування синтезатором мови	Лист	Маса	Масштаб	
Зам.	Лист	№ докум.	Підп.	Дати					
Розроб.		Дмитренко К.О.							
Перев.		Грушко С.С.		21.05					
Т.контр.						Лист 3	Листів 4		
Н.контр.		Щербак Н.В.				НУ «Запорізька політехніка»			
Затв.		Кудерметов Р.К.				КНТ-527			

Робота системи керування синтезатором мови

```
emoji_template_list = [
    [
        ['😞', '😞', '😞', '😞', '😞', '😞'], template_hash[:sad]
    ],
    [
        ['😏', '😏'], template_hash[:whispering]
    ],
    [
        ['😱', '😱', '😱', '😱', '😱'], template_hash[:scared]
    ],
    [
        ['😄', '😄', '😄', '😄', '😄', '😄', '😄', '😄', '😄'], template_hash[:happy]
    ],
    [
        ['😡', '😡', '😡', '😡', '😡'], template_hash[:angry]
    ]
]
```

Перетворення Емої в шаблон



Test channel

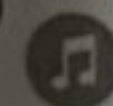
1 12:28

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.



Test channel

2 12:29



Lorem Ipsum is-2020-04-23.mp3

00:35

Тестування в Telegram

						13.02070849.52730 ПЛ4			
							Лист	Міся	Місяц
Зам.	Лист	№ докум.	Підп.	Дат.	Робота системи керування синтезатором мови				
Розроб.		Дмитренко К.О.	<i>Дмитренко</i>						
Перев.		Грушко С.С.	<i>С.С. Грушко</i>	21/05					
Т.контр.									
						Лист 4	Листів 4		
Н.контр.		Щербак Н.В.	<i>Щербак</i>		Робота системи керування синтезатором мови	НУ «Запорізька політехніка»			
Затв.		Куберняков Р.К.	<i>Куберняков</i>			КНТ-527			