Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

РОЗРОБКА ІНСТРУМЕНТУ ДЛЯ СТВОРЕННЯ КОНТЕНТУ У ВИГЛЯДІ БОТУ, ЩО СИНХРОНІЗУЄ АУДІОЗАПИС З ГУБАМИ НА ВІДЕО

Текстова частина до курсової роботи
за спеціальністю „Інженерія програмного забезпечення” 121

Керівник курсової роботи
с.в. _Борозенний С.О.__
(прізвище та ініціали)
_____
(підпис)
“____” _____ 2021 р.

Виконав студент _____
___Посвистак А.І._____
(прізвище та ініціали)
“____” _____ 2021 р.

Київ 2021

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ
Зав.кафедри мультимедійних систем,
доцент, к.ф-м.н.
_____О. П. Жежерун (підпис)
„____”_____2021 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студенту Посвистак Анастасії Ігорівні факультету інформатики 4-го курсу

ТЕМА Розробка інструменту для створення контенту у вигляді боту, що синхронізує аудіозапис з губами на відео

Зміст ТЧ до курсової роботи:

Індивідуальне завдання
Вступ
1 Синхронізація відео: історія та напрямки використання
2 Нейронна мережа Wav2Lip
3 Метод
Висновки
Список літератури
Додатки (за необхідністю)

Дата видачі „___” _____ 2021 р. Борозенний С.О. _____
(підпис)
Завдання отримав _____ (підпис)

Тема: Розробка інструменту для створення контенту у вигляді боту, що синхронізує аудіозапис з губами на відео

Календарний план виконання роботи:

№ п/п	Назва етапу дипломного проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу	01.11.2020	
2.	Аналіз матеріалів за темою	14.01.2021	
3.	Розробка та програмування програми	14.02.2021	
4.	Написання текстової частини до курсової роботи	20.03.2021	
5.	Коригування виконаної роботи	07.04.2020	
6.	Створення слайдів для доповіді та написання доповіді.	09.04.2020	
7.	Остаточне оформлення роботи та слайдів	11.04.2020	
8.	Захист курсової роботи	19.04.2020	

Посвистак А. І. _____

Борозенний С. О. _____

“_____”_____

Зміст

Анотація

Робота присвячена розробці інструмента, що стане в нагоді тим, хто хоче створювати власний контент, який знадобиться в робочих потребах, а саме в сфері маркетингу та реклами, а також у буденному житті для створення розважальних відео. Це бот, який синхронізує аудіо доріжку із запису з губами на відео.

Ключові слова: синхронізація губ, розмовляючі обличчя на відео, генерація відео.

Вступ

Постановка завдання

Керуючись актуальністю проблеми на даний час, за мету роботи була поставлена розробка та реалізація боту, який стане в нагоді тим, хто хоче створювати розважальний відео-контент. Це знадобиться як для використання в роботі, наприклад, у сфері маркетингу, так і в особистих потребах, зважаючи на те, який вплив мають соціальні мережі на буденне життя.

Основна частина роботи складається з 3 розділів.

Перший розділ присвячено історії та розвитку синтетичного контенту, а також місця синхронізації губ у сім’ї синтетичного контенту.

Другий розділ містить інформацію про модель, яка використовується для розробки інструменту для створення контенту, а саме – боту.

У третьому розділі розглядаються методи, інструменти, представлені в курсовій роботі, а також результати.

Основна частина

Розділ 1 : Синхронізація відео: історія та напрямки використання

1. 1. Що таке Lip Sync

Lip Sync (скорочено від lip synchronization) — синхронізація губ — це технологія в області комп’ютерного зору, яка синхронізує записаний окремо аудіо запис із відео записом. Проте насправді ідея lip sync, або звукового дубляжу, виникла практично відразу з появою звукового кіно. Це допомагало зберегти технічну складність і візуальний ефект виступу й при цьому не жертвувати його якістю.

У сучасних реаліях технологію lip sync використовують у розважальних потребах, проте вона має набагато ширший потенціал: відтворення навчальних курсів різними мовами світу, перегляд фільмів та серіалів рідною мовою без дублювання, що робить сам процес набагато приємнішим та зручнішим, трансляції різних світових прес-конференцій та публічних звернень, які зазвичай перекладені в прямому ефірі, проте губи адресата не синхронізовані з перекладною мовою, а також автоматична анімація губ CGI(англ. computer-generated imagery, букв. «Зображення, згенеровані комп'ютером») персонажів фільмів, керуючись записаним аудіо акторів — це може заощадити кілька годин ручної праці під час створення анімаційних фільмів та насиченого розмовного ігрового вмісту.

Рис. 1 Сфери використання технології lip sync

Ефективне спілкування через мовні бар'єри завжди було головним прагненням людей у всьому світі, тому що це допомагає поширювати інформацію набагато більшій кількості людей, кому вона справді потрібна. Майже весь якісний навчальний контент у сучасному світі створюється англійською мовою, що скорочує значний відсоток переглядів через брак знань. Наукові, технічні та правові терміни, основа всієї інформації, не завжди може бути зрозумілою, дивлячись відео із субтитрами. Проте найширше застосування даної технології можна буде помітити, а зараз усе частіше

з’являються готові продуктові рішення, у сфері маркетингу та реклами. Світові компанії не стоять осторонь, а навпаки активно використовують технологію в рекламних цілях, тим самим економлять свій бюджетом і охоплюють увагу більшої кількості аудиторій: є можливість не запрошувати зірок, а самому створити ексклюзивного персонажа. Також перед демонстрацією реклами в різних країнах її дублюють національні диктори, що не кращим чином позначається на впізнаваності образу. Тому, наприклад, у проекті Malaria No More[8] англійський футболіст Девід Бекхем «заговорив» на дев'яти мовах в соціальному відеоролику про небезпеку малярії завдяки використанню цієї технології. Успіх кампанії є прекрасним прикладом того, як синтетичний контент може впливати на людей у всьому світі, щоб змусити їх почути про важливе та вжити значущі дії у своєму житті.

1. 2. Місце lip sync у сім'ї синтетичного контенту

Фейкове відео (далі — deepfake) — це синтез слів «глибинне навчання» (англ. deep learning) і «підробка» (англ. fake). Термін уперше був використаний користувачем платформи Reddit[6]: він створив місце на сайті, де інші користувачі ділилися відео, які використовували технологію зміни облич з відкритим кодом. З тих пір значення цього терміну розширилося, включивши в себе програми для розробки синтетичного контенту, що існували до сторінки Reddit, та нові, наприклад, StyleGAN — нейронна мережа, яка генерує фотографії неіснуючих людей ThisPersonDoesNotExist[7]. Це методика зображення людини, що базується на штучному інтелекті. Вона використовується для поєднання і накладення існуючих зображень та відео на вихідні зображення або відеоролики. Технологія дозволяє створювати відео, якого насправді не було, "змусити" людину говорити те, чого вона ніколи не

говорила. З роками доступ до технології створення діпфейків став простішим, оскільки з’явилося багато програм та веб-сайтів, які дозволяють користувачам створювати власний контент. Сам термін deepfake не завжди має позитивне сприйняття через використання його в цілях обманути людей, але хоча інструменти для заміни обличчя можуть бути переконливими, вони відносно сирі та зазвичай залишають цифрові або візуальні артефакти, які комп’ютер може виявити.

З іншого боку, технології синхронізації губ є більш тонкими, тому їх важче помітити. Вони маніпулюють набагато меншою частиною зображення, а потім синтезують рухи губ, які точно відповідають тому, як насправді рухався би рот людини, якби вона сказала конкретні слова.

Проте існує ряд потенційно вигідних випадків їхніх використань для різних підприємств, зокрема додатків у маркетингу та рекламі, які вже використовуються відомими брендами.

Перші наукові роботи з фейковими відео з’явилися у 1997 році у галузі комп’ютерних наук, підгалузь — комп’ютерний зір. Це була програма Video Rewrite[2], яка використовувала наявні кадри з людиною та автоматично створювала нові, де людина нібито промовляє заготовлений текст, якого на оригінальному відео немає(див. рис. 2). Цей прийом корисний при дублюванні фільмів, наприклад, де послідовність фільмів може бути змінена, щоб синхронізувати рухи акторів губами з новим звуковим супроводом. Video Rewrite — це перша система анімації для обличчя, яка автоматизує всі завдання маркування та складання, необхідні для повторної синхронізації існуючих кадрів на новий саундтрек.

Рис. 2 Результати системи Video Rewrite

Їхній підхід був визначний в автоматизації всіх ключових компонентів: відстеження обличчя, виявлення фонем, та загальна композиція. Також вони отримали переконливі результати для кількох коротких послідовностей. Проте загальність методу на практиці була обмежена через недостатню кількість довідкових даних про фонеми та візем — зображення обличчя людей при вимові певної фонеми.

Face2Face[14], проект, опублікований в 2016 році, модифікує відео з обличчям людини, щоб зобразити, як вони імітують міміку іншої людини в режимі реального часу(див. рис. 3). Проект славиться тим, що вони розробили перший метод реконструкції міміки в режимі реального часу за допомогою камери, яка не фіксує глибину, що робить можливим виконання техніки за допомогою звичайних споживчих камер.

Рис. 3 Результати роботи Face2Face[14]

Один із найвідоміших сучасних проектів був опублікований у 2017 році та називався Synthesizing Obama(букв. "Синтезуючи Обаму")[3](див. рис. 4). Програма модифікує наявні відеокадри колишнього президента США Барака Обами таким чином, що зображує як він вимовляє слова, які містяться в окремій звуковій доріжці. Основна відмінність цієї програми від інших наявних — це фотореалістична техніка для синхронізації губ з аудіо. Враховуючи форму рота в кожний момент часу, високоякісну текстуру рота синтезується та складається її з правильним 3D-співпаданням пози, щоб змінити те, що Барак Обама говорить у цільовому відео, на відповідність вхідній аудіо доріжці. [3]

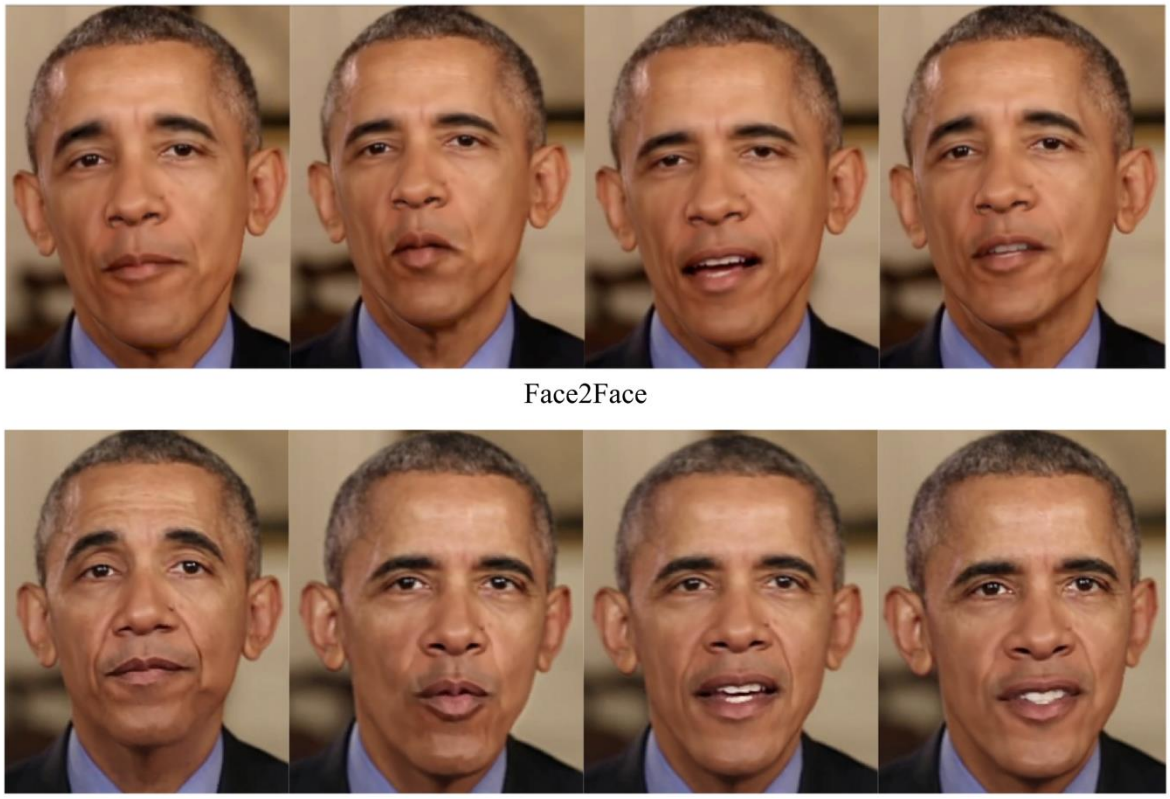

Рис. 4 Порівняння результатів Face2Face з Synthesizing Obama
на однаковому відео[3]

У 2020 році на світ з’явився продукт, але справжню популярність він здобув на початку 2021. Це додаток Wombo.AI[10], заснований на технології діпфейків та створений виключно в розважальних цілях: він "змушує" людину на фотографії "заспівати". Програма була розроблена тільки для фотографій, зроблених на фронтальну камеру, проте користувачі усвідомили комічний потенціал у завантаженні фотографій відомих людей та "змушували" їх під різні пісні – програма налічує близько 20 треків.

Розділ 2 : Нейронна мережа Wav2Lip

2. 1. Порівняння Wav2Lip з LipGAN

Wav2Lip [1] - це перша незалежна від контексту модель, яка генерує відео з максимально точною синхронізацією губ, що відповідає реальним синхронізованим відео. Оцінки людей показують, що згенеровані відеоролики Wav2Lip надають перевагу над існуючими методами та несинхронізованими версіями більше 90% часу.

Wav2Lip — це покращена версія моделі LipGAN(див. рис. 5)[4]. Головною відмінністю між ними є покращений та надійніший дискримінатор — нейронна мережа - класифікатор, яка має завдання визначити, чи згенеровані рухи губ відповідають аудіо доріжці для певного кадру. Було виявлено, що дискримінатор LipGAN є лише приблизно на 56% точним під час виявлення несинхронізованих пар аудіо-губ на тестовому наборі LRS2[9]. Дискримінатор із синхронізації губ готується заздалегідь та не проходить навчання з часом, тому його точність становить 91% на тих самих тестових даних. Виділяють[1] дві основні причини такої різниці: по-перше, дискримінатор LipGAN використовує один кадр для перевірки синхронізації губ. По-друге, створені зображення під час тренувань містять багато артефактів через великі масштаби та варіації пози. Навчання дискримінатора в налаштуваннях GAN на шумно створених зображеннях, як це було зроблено в LipGAN, призводить до того, що дискримінатор фокусується на візуальних артефактах, а не на відповідності аудіо з губами. Саме це випливає у значне падіння точності виявлення несинхронізації.

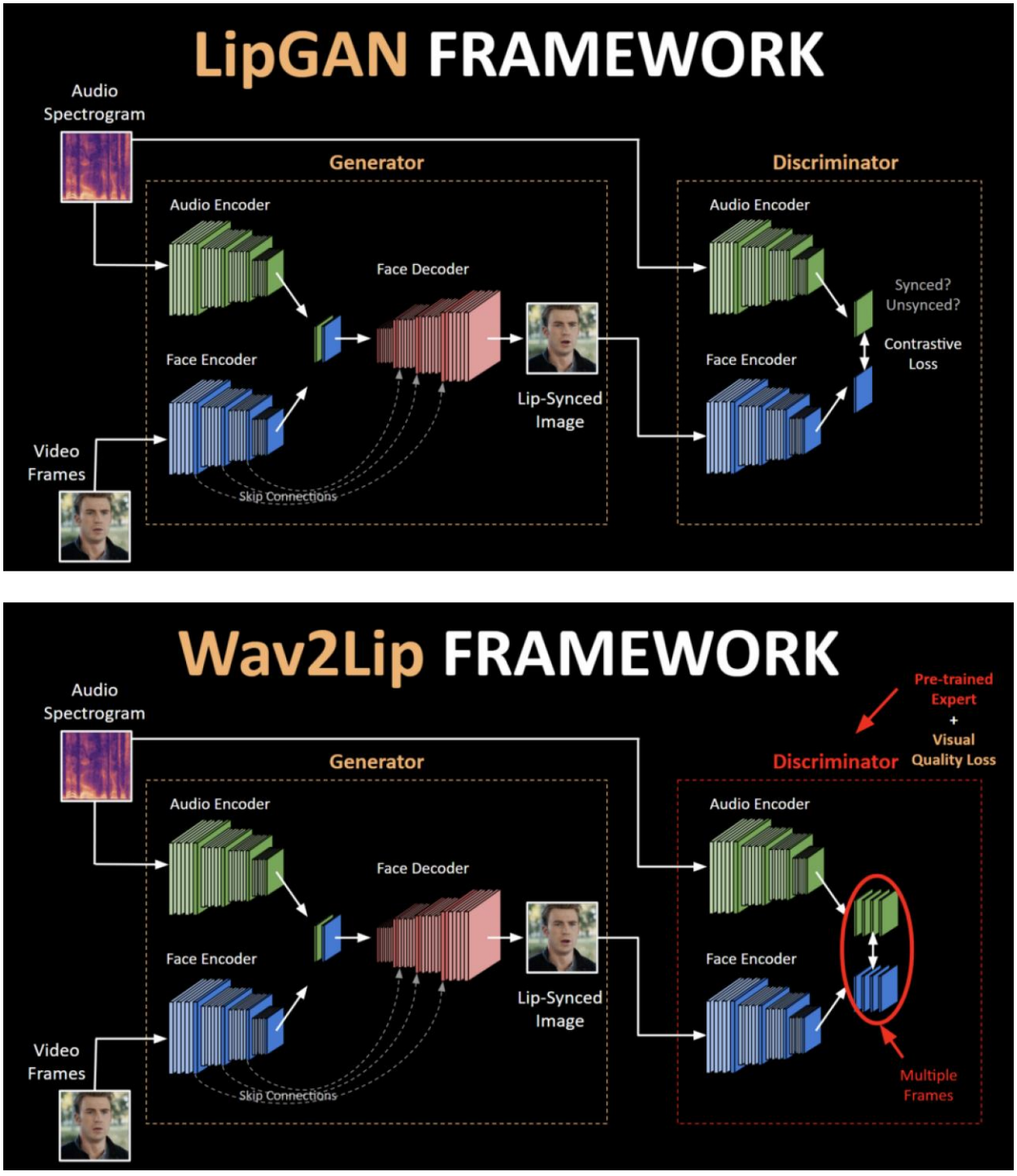

Рис. 5. Порівняння LipGAN з Wav2Lip

Той факт, що дискримінатор не навчається паралельно з генератором суперечить початковій концепції GAN, але дає кращі результати. GAN(англ. Generative Adversarial Network, букв. "Генеративна змагальна мережа") працюють, змушуючи дві нейронні мережі конкурувати між собою: генератор, який генерує фальшиві зображення, в даному випадку, фальшиві синхронізації губ, і експертний дискримінатор, який намагається визначити, чи генерується синхронізація губ реальною чи фальшивою. Дискримінатор намагається визначити фейкове відео, в нашому випадку, фейкову синхронізацію губ, порівнюючи реальне зображення із несправжнім lip sync. Цикл зворотного зв'язку нейронної мережі, який намагається обдурити дискримінатор, а дискримінатор, навпаки, намагаючись виявити підробку, з часом покращує GAN.

Нейронна мережа, здатна обдурити дискримінатора, виграє. І дискримінатор, і генеративна змагальна мережа навчаються одночасно. Ідея полягає в тому, що з часом і дискримінатор, і генератор стануть кращими у своїй роботі.

2. 2. Як працює Wav2Lip

2. 2. 1. Архітектура дискримінатора

Як було сказано раніше, для того, щоб модель надала більш точні результати, необхідно використовувати заздалегідь підготовленого експерта з розпізнавання синхронізації губ, дискримінатора, що точно визначає синхронізацію на реальних відеодоріжках. Існує мережа, яка була використана для виправлення помилок синхронізації губ для створення великих наборів даних синхронізації губ, це SyncNet[11].

У моделі Wav2Lip було взято за основу цю мережу, проте внесено певні зміни для підготовки експертного дискримінатора синхронізації губ. По-перше, замість подавання зображень у сірих відтінках, об'єднаних за каналами, як у оригінальній моделі, було вирішено подавати кольорові зображення. По-друге, модель Wav2Lip значно глибшаи. По-третє, використовується інша функція втрат(loss function): подібність косинусів з бінарними перехресними ентропійними втратами. Тобто, обчислюється точковий добуток між активованим ReLU вбудованим відео та мовленням v, s, щоб отримати одне значення між [0, 1] для кожного зразка, яке вказує на ймовірність синхронізації вхідної аудіо-відео пари:

$$P_{sync} = \frac{v \cdot s}{\max(||v||_2 \cdot ||s||_2,\ \epsilon)} \quad (1)$$

Експертний дискримінатор тренується на наборі даних LRS2, що налічує близько 29 годин даних(відео), з кількістю фреймів, що подаються на вхід, 64, з $T_v =$ кадрів, використовуючи оптимізатор Адама [12] з початковою швидкістю навчання $1e^{-3}$. Після введених покращень та змін, експертний

дискримінатор для синхронізації губ має точність приблизно на 91% на тестовому наборі LRS2, тоді як дискримінатор, який використовується в LipGAN, на 56% точний на тому самому тестовому наборі.

2. 2. 2. Архітектура генератора

Отримавши точний дискримінатор, його можна використовувати для того, щоб натренувати генератор(див. рис. 2), перемогти його й позбутися неточних формувань під час навчання.

Почнемо з опису архітектури генератора. Тут використовується подібна до LipGAN архітектура. Головна відмінність між ними, як уже було написано, — це не паралельне навчання дискримінатора з генератором, а окреме. Генератор G містить три блоки: (i) модель-ідентичності, (ii) модель-мовлення й (iii) декодер обличчя. Модель-ідентичності — це стек залишкових згорткових шарів, які кодують випадковий опорний кадр R, об'єднаний з P-позицією (цільова грань із замаскованою нижньою половиною) вздовж осі каналу. Кодер мови також являє собою стек 2D-звивин для кодування вхідного сегмента мовлення S, який потім об'єднується з поданням обличчя. Декодер також являє собою набір звивинних шарів, поряд із звивками для транспортування. Генератор навчений мінімізувати втрати на реконструкцію L1 між сформованими кадрами L_g та справжніми L_G:

$$L_{recon} = \frac{1}{N} \sum_{i=1}^{N} ||L_g - L_G||_1$$

Так як дискримінатор навчався, отримувавши на вхід $T_v = 5$ суміжних кадрів, і обробляв їх, то генератор має так само працювати: генерувати всі

також потрібен буде генератор G, щоб генерувати всі $T_v = 5$ кадри. Оскільки наш генератор обробляє кожен кадр незалежно, ми складаємо часові кроки вздовж розміру партії, подаючи опорні кадри, щоб отримати вхідну форму (), де N, H, W - партія – розмір, висота та ширина відповідно. Під час подачі сформованих кадрів експертному дискримінатору, часові кроки об'єднуються уздовж виміру каналу, як це було зроблено також під час навчання дискримінатора. Отриманою вхідною формою для експертного дискримінатора є (), де лише нижня половина сформованої поверхні використовується для дискримінації. Генератор також навчений мінімізувати "експертну втрату синхронізації" від експертного дискримінатора:

де обчислюється відповідно до рівняння 1.

Слід зауважити, що ваги експертного дискримінатора залишаються замороженими під час навчання генератора. Ця сильна дискримінація, заснована виключно на концепції синхронізації губ, отриманої з реальних відео, змушує генератор також досягти реалістичної синхронізації губ, щоб мінімізувати втрати при синхронізації губ.

2. 2. 3. Генерування фотореалістичних облич

Результати показали, що використання готового й натренованого дискримінатора для синхронізації губ «змушує» генератор створювати точні форми губ. Однак іноді в результаті деякі області бувають трохи розмитими або містять незначні артефакти. Щоб зменшити цю незначну втрату якості, додатково разом із генератором навчається простий візуальний дискримінатор якості в налаштуваннях GAN. Таким чином, існує два дискримінатори: один для точної синхронізації, а інший — для кращої візуальної якості. Розпізнавач синхронізації губ не навчений у налаштуваннях GAN тому, як уже було

написано раніше, що дискримінатор, який був навчений окремо, а не паралельно з генератором, дає результат з точністю в 91%, а не 56%, як у випадку з LipGAN[4]. З іншого боку, оскільки візуальний дискримінатор якості не виконує жодних перевірок синхронізації губ і лише «штрафує» генератор за фальшиві генерації облич, він тренується на створених обличчях.

Розділ 3 : Метод

3. 1. Інструменти та бібліотеки, використані в роботі

Мова написання роботи — Python, так як це одна з мов програмування, що дозволяє зручно працювати з нейронними мережами, а також розробляти телеграм ботів. Більшість моделей та алгоритмів підтримуються цією мовою, що є гарною перевагою у використанні. Проекти машинного навчання відрізняються від традиційних програмних проектів. Відмінності полягають у наборі технологій, навичках, необхідних для проекту на основі штучного інтелекту. Для реалізації цих прагнень до штучного інтелекту слід використовувати мову програмування, яка є стабільною, гнучкою та має доступні інструменти. Python пропонує все це.

Так як зараз увесь світ поступово переходить у дистанційний режим, проводячи вільний час у соціальних мережах і месенджерах, то бот — це одне з найкращих рішень для створення корисних інструментів. Вони легко поширюються, а також не треба вигадувати чогось нового, тому що боти створюються в месенджерах, які вже мають певну популярність. Для створення боту я обрала такий месенджер, як Telegram[17], що наразі є одним з найпопулярніших месенджерів серед молоді.

Перш ніж починати розробку, бота необхідно зареєструвати та отримати його унікальний id, який є одночасно і токеном. Для цього в Telegram існує спеціальний бот — @BotFather — створений для реєстрації, налаштувань та управління telegram-ботами(див рис. 6). За його допомогою можна створити необмежену кількість ботів.

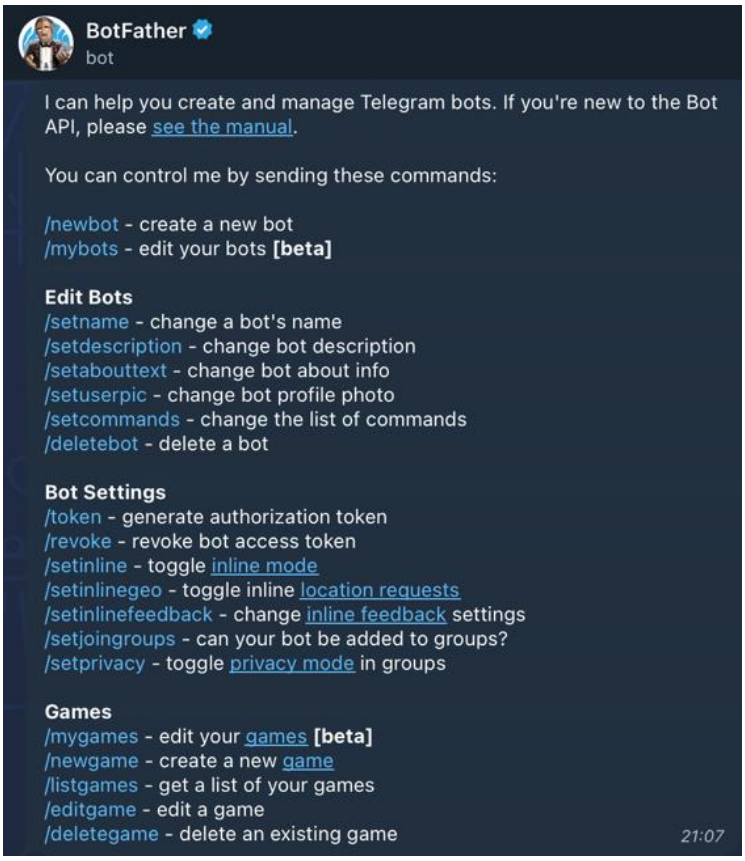

Рис. 6 Функціонал, наявний в боті BotFather

Для розробки самого бота я використовувала бібліотеку PyTelegramBotApi[16], так як вона містить усі необхідні функції для обробки відео та аудіо, що є головними об’єктами в моїй роботі.

Програма була розгорнута на хмарі з метою зручності в користуванні. Слід розглянути поняття хмарних обчислень.

Хмарні обчислення — це надання обчислювальних служб(серверів, аналітики, баз даних і програмного забезпечення) через Інтернет(«хмару»). Більшість служб хмарних обчислень поділяються на три загальні групи:

- Інфраструктура як послуга (IaaS), наприклад, AWS. Основна група хмарних обчислювальних служб. У схемі IaaS ви орендуєте IT-інфраструктуру (сервери, віртуальні машини, сховище, мережі та

операційні системи) у хмарного постачальника з системою оплати в міру використання;

- Платформа як послуга (PaaS), наприклад, AWS Elastic Beanstalk, Google App Engine, Heroku. Платформа як послуга відноситься до хмарних обчислювальних служб, які постачають середовище, доступне за потребою, для розробки, тестування, доставки додатків та управління ними. PaaS спрощує розробникам завдання швидкого створення веб-додатків або мобільних додатків без необхідності мати справу з базовою інфраструктурою серверів, сховища, мережі та баз даних, необхідних для розробки(див. рис. 7);
- Програмне забезпечення як послуга (SaaS), наприклад, Microsoft Office 365. Це метод доставки додатків через Інтернет за запитом та зазвичай на основі передплати. У схемі SaaS хмарні постачальники розміщують програмне забезпечення та базову інфраструктуру й керують ними, а також займаються всім обслуговуванням, включаючи оновлення програмного забезпечення та установку налаштувань безпеки. Користувачі підключаються до додатка через Інтернет, зазвичай за допомогою веб-браузера на своєму телефоні, планшеті або ПК.

Існує декілька причин, чому варто використовувати саме хмарні сервіси, а не звичайні веб-хостинги. Наприклад, необмеженість у послугах — завжди можна розширити список, якщо щось знадобиться додатково, усе контролюється провайдером.

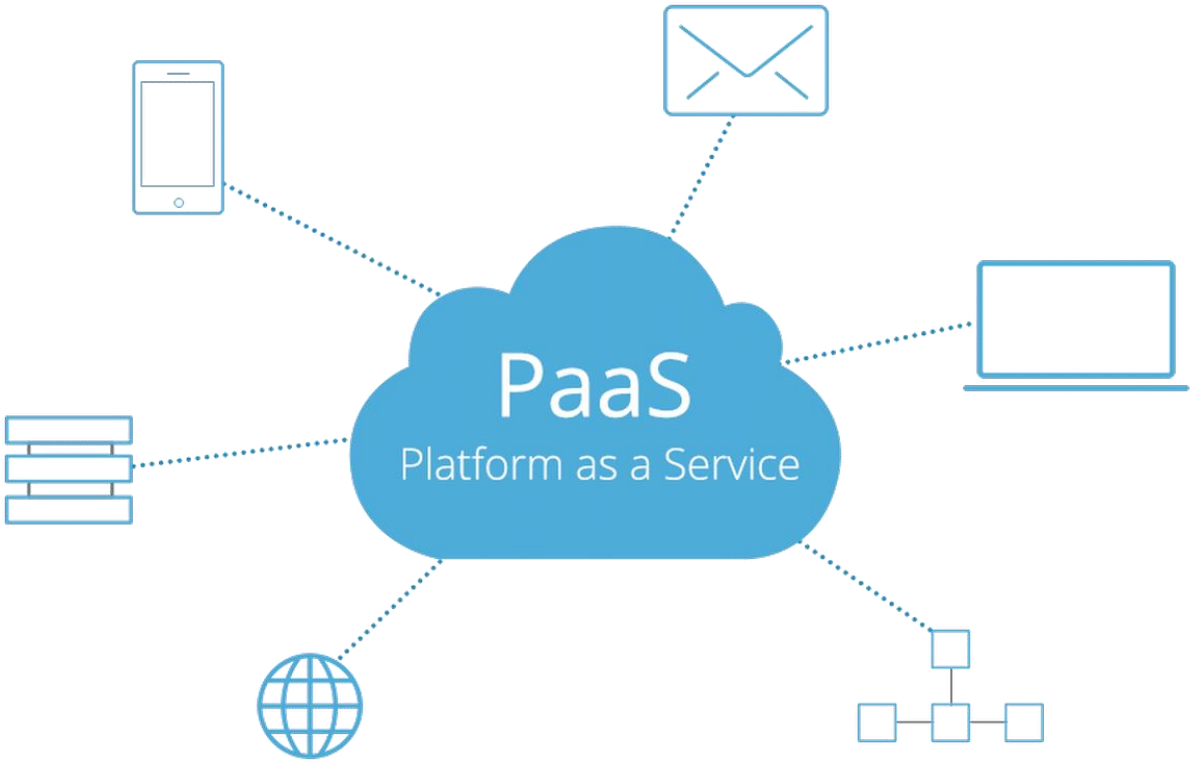

Рис. 7. Платформа як послуга[13]

Для розгортання своєї програми я обрала PaaS-платформу FloydHub[18], яку ще розробники називають Heroku[21] for Data Science. Це хмарна платформа для запуску коду машинного навчання, платформа керує середовищем, щоб можна було зосередитися на проекті. Мій вибір пав на FloydHub, тому що це одне з кращих рішень для розробки на GPU. Платформа - це абстракція контейнерів Docker[19], що працюють на екземплярах Amazon EC2[20], яка надає простий API для виконання більшості завдань, пов'язаних із розробкою моделей машинного навчання. Подібно до Heroku, де розробники завантажують код, а решту залишають PaaS, FloydHub очікує від користувачів завантаження наборів даних, коду для навчання моделі та коду для запуску

навченої моделі. Користувачі можуть швидко розпочати навчальну роботу із заздалегідь визначеними налаштуваннями конфігурації або вибрати надання файлу YAML із власною конфігурацією. FloydHub візьме на себе відповідальність за створення власного середовища та запуск коду в ньому.

Давайте розглянемо основні компоненти FloydHub. Проекти FloydHub виступають межею для всіх активів, що належать до конкретного проекту. Він буде містити код, експерименти з версіями, вихідні файли, журнали та повну історію роботи. Він також може містити блокноти Jupyter[22], створені в Workspaces. Першим кроком до використання FloydHub є створення проекту. Після створення проекту очевидним наступним кроком є початок навчальної роботи.

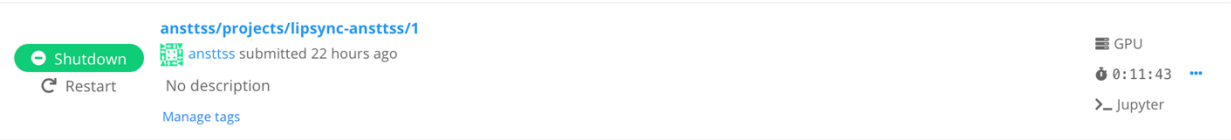

Рис. 8 Приклад екземпляру «роботи»

FloydHub має простий та інтуїтивно зрозумілий робочий процес для започаткування навчальної роботи. Очікується, що розробники пишуть і тестують код Python на своїй локальній машині перед створенням завдання. Коли настає час запустити навчальну роботу в масштабі, вони просто вибирають заздалегідь визначене середовище, таке як TensorFlow і PyTorch, а також тип екземпляра на основі CPU або GPU. Вони також можуть вказувати на розташування набору даних, який уже завантажено. Якщо завдання потребує спеціальних пакетів та залежностей, їх можна додати до файлу з назвою floyd_requirements.txt, який відповідає тому ж формату, що і Python's requirements.txt. Це дозволяє користувачам точно визначити версії та пакети, необхідні для роботи. Коли Jupyter Notebook став стандартним IDE для

обробки даних, FloydHub додав підтримку для них через Workspaces. Окрім запуску блокнота, Workspace майже схожий на віртуальну машину, де користувачі можуть отримати доступ до оболонки. Файли з локальних машин можна безпосередньо завантажити в робочу область, щоб отримати доступ із блокнота Jupyter.

3. 2. Огляд методів. Результати

Бот містить команди /help та /start, які відповідають користувачу однаковим чином, а саме — закликають його завантажити своє відео. Варто зазначити, що усі обробники виконуються в тому порядку, в якому вони були оголошені. Щоб запустити бота, треба додати стрічку bot.polling().

handling start and help commands

```
@bot.message_handler(commands=['start', 'help'])
def start_message(message):
    chat_id = message.chat.id
    response_text = "Hello, I am a LipSync bot. " + " I can make anyone say anything!"
    continue_text = "Send me a short video with a person. The face must be clearly visible"

    if os.path.exists("id" + str(chat_id)):
        shutil.rmtree("id" + str(chat_id))

    if not os.path.exists("id" + str(chat_id)):
        os.makedirs("id" + str(chat_id))

    bot.send_message(message.from_user.id, response_text)
    bot.send_message(message.from_user.id, continue_text)
```

Рис. 9 Обробка команд /help та /start

Рис. 10 Команда /start

Рис. 11 Команда /help

Після того, як людина завантажила відео, на якому чітко зображене обличчя людини, бот відправляє користувачу запит про аудіо(див. рис. 12).

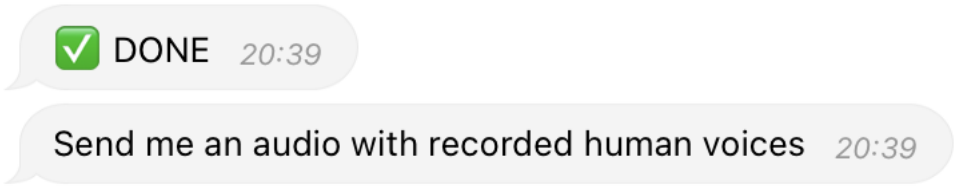

Рис. 12 Повідомлення бота після того, як користувач завантажив відео

Обробник повідомлень — це функція, яка прикрашена декоратором message_handler екземпляра TeleBot. Обробники повідомлень складаються з одного або декількох фільтрів. Кожен фільтр повертає значення True для певного повідомлення, щоб обробник повідомлень отримав право обробляти це повідомлення. Наприклад, обробник повідомлень відео оголошується наступним чином (за умови, що бот є екземпляром TeleBot): @bot.message_handler(content_types="video"):

def video_handler(message):

bot.send_message(message, “VIDEO”)

Назви функцій не мають обмежень. Функція повинна приймати щонайменше одного аргументу, який буде повідомленням, що функція повинна обробляти.

Проте content_type(фільтр) може бути різного типу, наприклад: текст, аудіо, документи, фото, наклейка, відеоповідомлення, голосове повідомлення, місце розташування, контакт, закріплене повідомлення, новий член групи і тд.

Також TeleBot підтримує такі фільтри, як regexp, commands та func, тобто регулярний вираз у вигляді рядка, список рядків та функцію відповідно.

Для обробки аудіо, що надсилають користувачі, я використовую фільтр content_type типу “audio”. Після того, як користувач відправив своє аудіо, його файли відправляються в модель та через деякий час надсилається результат.

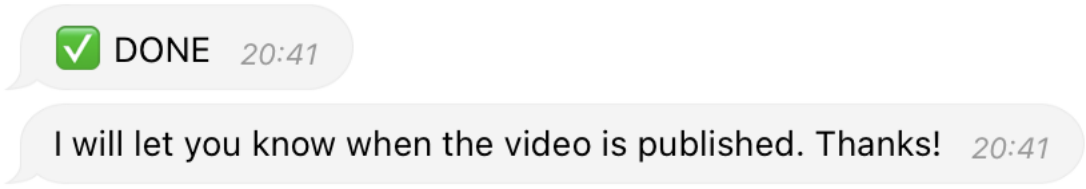

Рис. 13 Повідомлення бота після того, як користувач завантажив аудіо

handling video, audio and text message types

```python
@bot.message_handler(content_types=['video'])
def video_handler(message):
    chat_id = message.chat.id
    raw_video = message.video.file_id
    path = "id" + str(chat_id) + '/' + raw_video + ".mp4"
    file_info = bot.get_file(raw_video)
    downloaded_file = bot.download_file(file_info.file_path)
    with open(path, 'wb') as new_file_video:
        new_file_video.write(downloaded_file)
    response_text = "✅ DONE"
    continue_text = "Send me an audio with recorded human voices"
    bot.send_message(message.from_user.id, response_text)
    bot.send_message(message.from_user.id, continue_text)


@bot.message_handler(content_types="audio")
def audio_handler(message):
    chat_id = message.chat.id
    CHECKPOINT_PATH = 'wav2lip_gan.pth'
    VIDEO = ''
    AUDIO = ''
    OUTPUT = 'results/result.mp4'

    raw_audio = message.audio.file_id
    path = "id" + str(chat_id) + '/' + raw_audio + ".wav"
    file_info = bot.get_file(raw_audio)
    downloaded_file = bot.download_file(file_info.file_path)
    with open(path, 'wb') as new_file_audio:
        new_file_audio.write(downloaded_file)

    response_text = "✅ DONE"
    continue_text = "I will let you know when the video is published. Thanks!"

    for file in os.listdir("id" + str(chat_id) + '/'):
        if fnmatch.fnmatch(file, '*.wav'):
            AUDIO = file
        elif fnmatch.fnmatch(file, '*.mp4'):
            VIDEO = file
    bot.send_message(message.from_user.id, response_text)
    bot.send_message(message.from_user.id, continue_text)


    if os.path.exists("id" + str(chat_id)):
        if len(os.listdir("id" + str(chat_id))) == 2:
            bot.send_message(message.from_user.id, "Result")
            main(video_path=os.path.join("id"+str(chat_id), VIDEO), audio_path=os.path.join("id"+str(chat_
            bot.send_video(message.from_user.id, data=open(OUTPUT, 'rb'), supports_streaming=True)

```

Рис. 14 Обробка повідомлень типу аудіо та відео

Варто зауважити, що якщо відео та аудіо будуть різної довжини, то результуюче відео може зациклитися. Наприклад, якщо відео буде 10 секунд, а аудіо — 25 секунд, то результат, який вийде, буде тривалістю 25 секунд: відео повториться 2.5 рази – збільшиться кількість фреймів. А якщо навпаки, відео буде довше, аніж аудіо, то воно обріжеться.

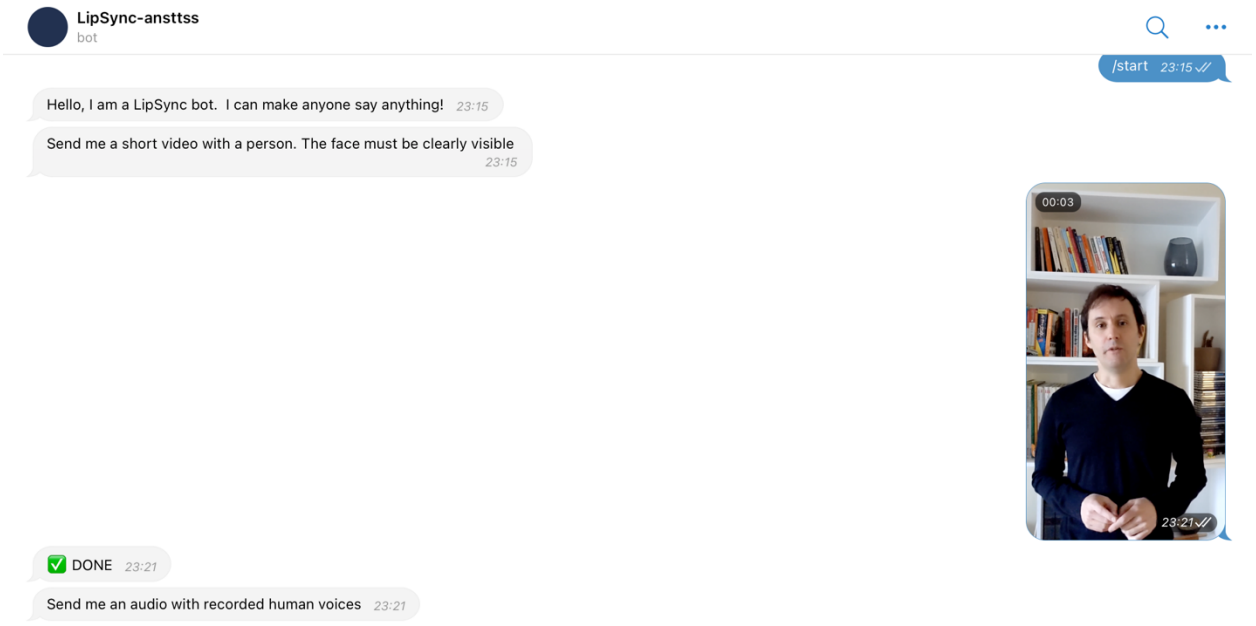

Рис. 13 Демонстрація роботи №1

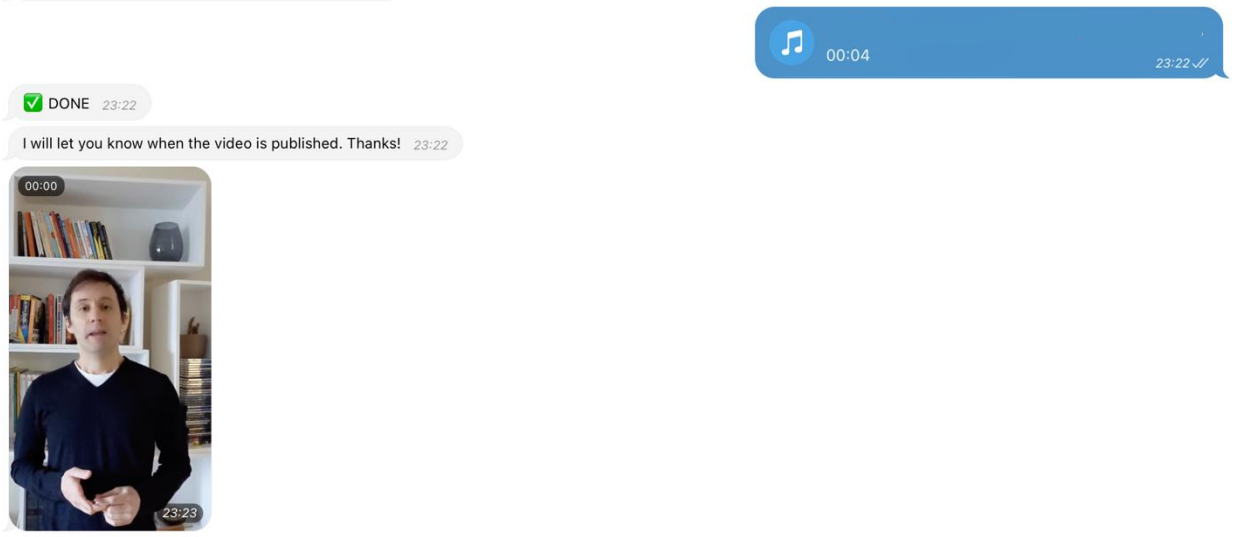

Рис. 14 Демонстрація роботи №2

Висновок

Розроблений інструмент, що був розглянутий у цій роботі, стане в нагоді людям зі сфери маркетингу, наприклад, потрібно зробити так, щоб відома людина заговорила мовою, яка необхідна для окремої цільової аудиторії. Тоді в нагоду прийде даний бот. А також тим, хто захоче синхронізувати аудіо з відео в інших цілях.

Робота написана на мові Python, а також розгорнута на хмарній платформі FloydHub, що надало можливість без проблем запустити модель і самого бота, тому що платформа надає години для використання GPU.

Список прийнятих скорочень

Lip sync — Lip synchronization — синхронізація губ

GAN — Generative Adversarial Network — генеративна змагальна мережа

PaaS — Platform as a Service — платформа як послуга

AWS — Amazon Web Services

ML — Machine Learning — машинне навчання

GPU — Graphics Processing Unit — графічний процесор

IDE — Integrated Development Environment — комплексне програмне рішення для розробки програмного забезпечення

fps — frames per second

Список використаних джерел

[1] http://cdn.iiit.ac.in/cdn/cvit.iiit.ac.in/images/Projects/Speech-to-Lip/paper.pdf

[2] https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/human/bregler-sig97.pdf

[3] https://grail.cs.washington.edu/projects/AudioToObama/siggraph17_obama.pdf

[4] https://github.com/Rudrabha/LipGAN

[5] https://medium.com/deepgamingai/deepfakes-ai-improved-lip-sync-animations-with-wav2lip-b5d4f590dcf

[6] https://reddit.com

[7] https://thispersondoesnotexist.com

[8] https://www.malarianomore.org

[9] https://www.robots.ox.ac.uk/~vgg/data/lip_reading/lrs2.html

[10] https://www.wombo.ai

[11] https://github.com/joonson/syncnet_python

[12] https://arxiv.org/abs/1412.6980

[13] https://blog.iron.io/what-is-platform-as-a-service/

[14]http://www.graphics.stanford.edu/~niessner/papers/2016/1facetoface/thies2016face.pdf

[15] https://aws.amazon.com

[16] https://github.com/eternnoir/pyTelegramBotAPI

[17] https://telegram.com

[18] https://www.floydhub.com

[19] https://www.docker.com

[20] https://aws.amazon.com/ec2

[21] https://heroku.com

[22] https://jupyter.org