ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка моделі клієнт-серверної взаємодії з веб-додатками на базі голосового та жестового контролю з використанням full-stack технологій»

на здобуття освітнього ступеня магістра
зі спеціальності 121 Інженерія програмного забезпечення
(код, найменування спеціальності)
освітньо-професійної програми «Інженерія програмного забезпечення»
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Богдан МАКСИМ'ЮК
(підпис)

Виконав: здобувач вищої освіти група ПДМ-62
Богдан МАКСИМ'ЮК

Керівник: Андрій БОНДАРЧУК
д.т.н., професор

Рецензент:
науковий ступінь, вчене звання
_____
Ім'я, ПРІЗВИЩЕ

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Магістр

Спеціальність 121Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедрою
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
«_____» _____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

_____Максим'юку Богдану Богдановичу_____

1. Тема кваліфікаційної роботи: Розробка моделі клієнт-серверної взаємодії з веб-додатками на базі голосового та жестового контролю з використанням full-stack технологій

керівник кваліфікаційної роботи Андрій БОНДАРЧУК д.т.н., професор,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «19» 10.2023р. №145 .

2. Строк подання кваліфікаційної роботи «29» грудня 2023р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, протоколи передачі даних, моделі машинного навчання та клієнт-серверної взаємодії.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Дослідження принципів побудови клієнт-серверної архітектури та протоколів передачі даних

2. Аналіз технологій машинного навчання та можливості застосування для розпізнавання жестової мови.
3. Розробка моделі клієнт-серверної взаємодії з елементом розпізнавання жестів.

5. Перелік графічного матеріалу: *презентація*
   1. Схема моделі клієнт-серверної взаємодії.
   2. Схема взаємодії під час навчання моделі.
   3. Помилки навчання та перевірки.
   4. Графік точності моделі.
   5. Аналоги.

6. Дата видачі завдання «19» жовтня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	19.10-05.11.23	
2	Вивчення матеріалів для можливих моделей клієнт-серверної архітектури та протоколів передачі даних	06.11-12.11.23	
3	Дослідження технологій машинного навчання для розпізнавання жестової мови	13.11-19.11.23	
4	Підготовка тренувальних даних для навчання моделі	20.11-26.11.23	
5	Тренування та оцінка моделі з розпізнавання жестової мови	27.11-03.12.23	
6	Розробка РОС додатку з використанням запропонованої моделі клієнт-серверної архітектури	04.12-10.12.23	
7	Оформлення роботи: вступ, висновки, реферат	11.12-20.12.23	
8	Розробка демонстраційних матеріалів	21.12-29.12.23	

Здобувач вищої освіти _____ Богдан МАКСИМ'ЮК
(підпис)

Керівник
кваліфікаційної роботи _____ Андрій БОНДАРЧУК
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 63 стор., 3 табл., 29 рис., 24 джерела.

Мета роботи – покращення точності та доступності взаємодії користувачів з веб-додатками на базі жестового контролю.

Об'єкт дослідження – процес взаємодії клієнта з сервером та розпізнавання жестів.

Предмет дослідження – протоколи передачі даних між клієнтом та сервером, моделі та алгоритми інтерпретації відеоданих

Короткий зміст роботи: У роботі проведено дослідження архітектури клієнт-серверної взаємодії та протоколи передачі даних. Проаналізовано основні технології машинного навчання для інтерпретації жестової мови. Натреновано модель для інтерпретації жестів. Розроблено модель клієнт-серверної взаємодії з розпізнаванням жестового контролю за типом товстого клієнта.

КЛЮЧОВІ СЛОВА: КЛІЄНТ-СЕРВЕРНА ВЗАЄМОДІЯ, МАШИННЕ НАВЧАННЯ, ЖЕСТОВИЙ КОНТРОЛЬ.

ABSTRACT

Text part of the master's qualification work:
63 pages, 29 pictures, 3 tables, 24 sources.

The purpose of the work is to enhance the accuracy and accessibility of user interaction with web applications based gesture controls.

Object of research – the process of client-server interaction and gesture recognition.

Subject of research – data transmission protocols between the client and server, models, and algorithms for video data interpretation.

Summary of the work: the study investigated the architecture of client-server interaction and data transmission protocols. The primary machine learning technologies for interpreting gesture language were analyzed. A model was trained for gesture interpretation. A client-server interaction model was developed, incorporating gesture control akin to a thick client.

KEYWORDS: CLIENT-SERVER INTERACTION, MACHINE LEARNING, GESTURE CONTROL

ЗМІСТ

ВСТУП

Розробка систем відео-зв'язку розпочалася задовго до активного розвитку електронно-обчислювальної техніки, і світ постійно шукав способи передачі зображення разом зі звуком під час телефонних розмов. Однак стаціонарні телефони з дисплеями та відеокамерами не мали великого комерційного успіху, оскільки були громіздкими і дорогими. Найбільше досягнення прийшло на початку 1990-х років, коли були розроблені комп'ютерні мережі та інтернет-протоколи, що зробили можливим відео-зв'язок на базі персонального комп'ютера.

Сьогодні складно уявити світ без відео-зв'язку і таких додатків, як Skype, Viber або Zoom. Бізнес використовує відео-конференції, великі компанії проводять зустрічі своїх команд, розподілених по різних офісах, та співбесіди з іноземними фахівцями. Спілкування на відстані за допомогою відео стало таким самим же звичайним і незамінним, як Інтернет і смартфони.

Однак є люди, які фізично обмежені у можливостях такого спілкування, які дала нам технологія, а точніше - у спілкуванні загалом. Це люди з повним або частковим порушенням слуху, вони не можуть розмовляти телефоном або використовувати голосовими повідомленнями. Їх можна замінити мовою жестів, яка вимагає, щоб співрозмовники дивилися один на одного.

Система відео-зв'язку з можливістю розпізнавання жестової мови є дійсно важливою у житті людей із порушеннями слуху та у житті оточуючих. Таким чином, можна не тільки забезпечити спілкування двох людей, що слабо чують, на відстані в тих випадках, коли тексту недостатньо, а й спілкування людини, яка знає мову жестів, і людини, яка не володіє необхідними навичками. Іншими словами, система відео-зв'язку може слугувати перекладачем з мови жестів в текстовий або аудіоформат.

На сьогодні методи машинного навчання визнані найефективнішими методами для вирішення цих завдань. Вони виникли як результат поєднання методів оптимізації, статистики та класичних математичних принципів. Ця галузь

має свої унікальні риси: при дослідженні та розробці алгоритмів, які покращуються з кожною епохою навчання, машинне навчання дозволяє робити передбачення та приймати рішення, не покладаючись на статичні програми з фіксованими інструкціями. Застосування штучного інтелекту допомагає розширити можливості людини та обробляти великі обсяги даних і виявляти закономірності з результатів їхнього аналізу.

Наразі методи машинного навчання зарекомендували себе як найефективніший спосіб вирішення таких завдань. Виникнувши на перетині статистики, класичних математичних дисциплін та методів оптимізації воно має свої особливості. Досліджуючи вивчення та побудову алгоритмів, результати яких розвиваються з кількістю епох вивчення даних, машинне навчання дає можливість робити прогнози та приймати рішення без використання класичних програм, які слідують статичним інструкціям. Штучний інтелект може доповнити можливості людини, пропускаючи величезний обсяг даних і виявляючи закономірності на основі результатів аналізу цих даних.

1 ТЕОРЕТИЧНІ ОСНОВИ КЛІЄНТ-СЕРВЕРНОЇ ВЗАЄМОДІЇ З ВЕБ-ДОДАТКАМИ

1.1 Клієнт-серверна архітектура та веб-додаток

Архітектура інформаційної системи - це специфікація, яка детально описує взаємодію набору стандартів програмного та апаратного забезпечення для формування IT-системи або платформи. Іншими словами, комп'ютерна архітектура – це спосіб побудови комп'ютерної системи та технології, які з нею сумісні.

Клієнт-серверна архітектура визначає взаємовідносини між виробником та споживачем у мережевій системі. У цій архітектурі сервер діє як постачальник послуг, а клієнт є отримувачем цих послуг. Сервер забезпечує високоефективні обчислювальні ресурси, що включають доступ до додатків, зберігання та обмін даними, а також можливість використання обчислювальної потужності.

Сервер може бути як фізичним комп'ютером, так і програмою, що надає певні послуги для інших програм та обробляє отримані від клієнтів повідомлення.

Клієнт – це кінцевий користувач, який отримує доступ до цих послуг у системі клієнт-сервер. Це може бути комп'ютер або система, до яких здійснюється доступ через мережу. Спочатку термін «клієнт» застосовувався до пристроїв, які не мали можливості запускати власні програми та отримували доступ до віддаленого комп'ютера через мережу.

Залежно від доступних параметрів прийнято виділяти наступні типи веб-клієнтів [1]:

- за використанням протоколу передачі даних за допомогою HTTP, протоколу WAP;
- за типом використовуваного додатку - веб-браузер або інша клієнтська програма;
- за типом рендерингу (візуалізації) даних;

- обробка логіки роботи додатку.

Особливу увагу слід приділити методу відтворення даних. Зазвичай розрізняють візуалізацію на стороні клієнта та на стороні сервера, і ці характеристики часто називають як «товстий клієнт - тонкий сервер» або «тонкий клієнт - товстий сервер» відповідно.

Оскільки клієнт-серверна архітектура не є строго типізованою структурою, ролі клієнта та сервера можуть бути розподілені відповідно до обраної концепції стека розробки програмного забезпечення, зокрема мови програмування та фрейм-ворку. Тому актуальним стає питання розподілу ролей між компонентами веб-додатку.

Підхід «товстий клієнт - тонкий сервер» спрямований на розподіл візуалізації даних і логіки обробки конкретної програми, в основному розгорнутої на стороні клієнта. Сучасні інтерпретовані інструменти мови програмування, такі як JavaScript і похідні фреймворки, можуть передавати виконання більшості бізнес-логіки програми безпосередньо в браузер.

Під час візуалізації на стороні клієнта JavaScript, який працює у веб-браузері користувача, відповідає за запит даних із сервера та взаємодію з веб-сторінкою. Наприклад, якщо користувач вводить у форму недійсне значення, клієнтського коду буде достатньо, щоб оновити сторінку з повідомленням про помилку без створення нового запиту до сервера, що працює без перезавантаження сторінки.

Натомість архітектура «тонкий клієнт-товстий сервер» є класичним підходом до розробки веб-сервісів. Наявність «товстого сервера» означає, що логіка, обробка даних та інших процеси веб-додатку реалізуються безпосередньо на стороні сервера. У випадку серверного рендерингу клієнт надсилає запит безпосередньо на сервер для кожної веб-сторінки або після того, як користувач взаємодіє з її компонентами. Таким чином, якщо у форму на веб-ресурсі будуть введені некоректні дані, клієнтський код запросить у сервера нову сторінку.

1.2 Голосове керування та управління жестами

Звук - це коливальний рух частинок середовища, що поширюється хвилями через середовище, наприклад, рідину, тверде тіло або газ, і сприймається слуховим апаратом. Звуком також називають вібрації, що сприймаються сенсорно-слуховою системою людей і тварин [2]. В даному випадку мова йде про обурення, що поширюються з певною частотою через одне з перерахованих вище середовищ.

Слуховий апарат людини здатний сприймати такі коливання у невеликому діапазоні. Більшості тварин сприймають звукові коливання в набагато ширшому діапазоні частот. Цей термін також загалом визначає процес поширення коливань у середовищі з різними фізичними властивостями, де сила, яка намагається відновити вихідне положення збуджених частинок до стану спокою, є сила пружності.

Хвильові коливання, що характеризуються звуком, є об'єктивно реальними і існують незалежно від їх сприйняття будь-яким живим організмом. Вивченням законів сприйняття, генерації та поширення звукових коливань у різних середовищах займається галузь наукових знань, яка називається акустикою.

Неймовірна кількість природних явищ супроводжуються специфічними звуками, які розпізнаються і сприймаються слуховими органами, допомагають у спілкуванні та орієнтації у просторі [3]. Через особливості сприйняття звукових коливань вухом різні звуки можна поділити на: гармонійні приємні звуки, наприклад, спів птахів, звуки ліри та інші музичні звуки, та звуки з певним значенням у спектрі, що зазвичай дратує або небажані звуки, під якими зазвичай розуміють шум

Шум або акустичний шум - це вібрації часточок навколишнього середовища, що сприймається слуховою системою як небажані сигнали. З акустичної точки зору: шум – це нестійкі або випадкові акустичні коливання, що характеризуються випадковими змінами амплітуди та частоти [4].

Шум класифікується відповідно до його джерела[5]:

- аеродинамічний, що виникає в газоподібному середовищі;
- гідродинамічний, що виникає в рідкому середовищі;
- електромагнітний, що виникає внаслідок впливу змінних магнітних сил, які викликають небажане збудження електромеханічних елементів обладнання;
- механічний, що виникає в результаті ударів у місцях з'єднання деталей і вібрацій поверхонь обладнання, машин або конструкцій;

Шум також класифікується за частотними діапазонами:

- низькі частоти - менше 400 Гц;
- середні частоти - більше 400 та менше 1000 Гц;
- високі частоти - вище 1000 Гц.

У вузькоспеціалізованих галузях, таких як електроніка та акустика, існує поняття кольору шуму, згідно з яким шумовим сигналам присвоюється певний колір відповідно до їхніх статистичних і спектральних характеристик.

Однією з таких характеристик є спектральна щільність, яка характеризує розподіл потужності за частотою. В акустиці прийнято поділяти спектр шуму на такі кольори: білий, рожевий, червоний (коричневий) і сірий шум. Іноді виділяють також інші типи [5].

Білий шум - це сигнал, який містить усі частоти в спектрі з однаковою інтенсивністю в заданому діапазоні частот. У сукупності це означає, що амплітуда сигналу є постійною на всьому спектрі частот, незалежно від їх частоти (рис. 1.1). Цей тип шуму зазвичай моделюється як сукупність випадкових значень, що мають однаковий рівень енергії на всьому діапазоні частот.

В контексті звуку, білий шум має специфічні властивості, коли всі звуки відтворюються з однаковою інтенсивністю на всіх частотах, надаючи специфічний "рівномірний" звуковий спектр.Свою назву він отримав від білого світла, що включає електромагнітні хвилі частот всього видимого діапазону електромагнітного випромінювання [6].

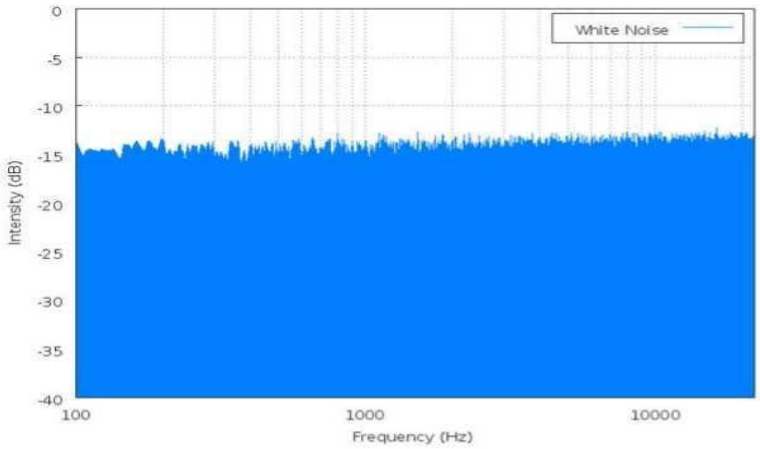


Рис. 1.1 Спектр білого шуму [6]

Рожевий шум - це тип шуму, де інтенсивність шуму на кожній октаві частот зменшується пропорційно частоті (рис. 1.2). Це означає, що вищі частоти в рожевому шумі мають меншу інтенсивність, порівняно з нижніми частотами. Назва "рожевий" походить від аналогії з колірною спектральною плотністю, де кожен октавний діапазон має однакову енергію.

У порівнянні з білим шумом, де кожна частота має однакову інтенсивність, рожевий шум характеризується таким розподілом енергії, що кожна октава має однакову енергію в спектрі частот, що робить його природнішим для сприйняття людським вухом. Іноді рожевим шумом називають будь-який шум, спектральна щільність якого зменшується зі зменшенням частоти [7].

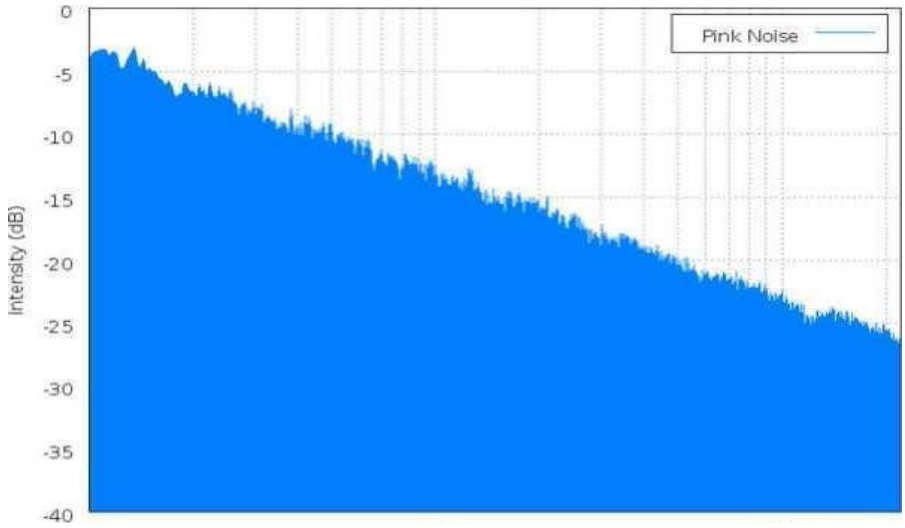


Рис. 1.2 Спектр рожевого шуму

Червоний шум - це тип шуму в сигналах, де інтенсивність шуму зменшується пропорційно частоті, внаслідок чого високі частоти мають значно меншу інтенсивність, ніж низькі. Назва "червоний" відноситься до спектрального розподілу інтенсивності, який може нагадувати зміщену в бік низьких частот частину спектра колірної палітри від червоного до фіолетового.

Шум також відомий як броунівський шум, оскільки його зміна в аудіо-сигнал від одного моменту до іншого є випадковим (рис. 1.3) [8].

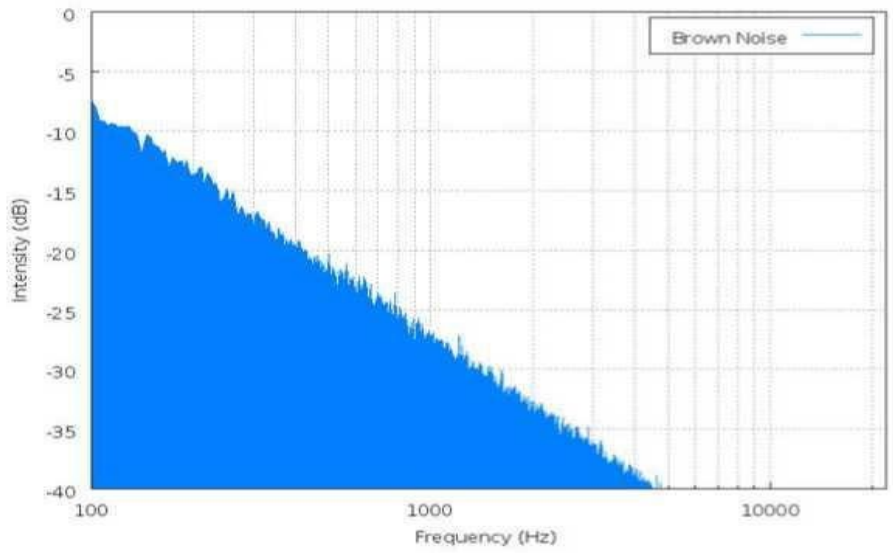


Рис. 1.3 Спектр червоного шуму

Сірий шум - це тип шуму, у якому енергія шуму розподілена рівномірно по всьому діапазону частот. Він характеризується тим, що кожна окрема частота має однакову енергію на одиничний інтервал частот. У порівнянні з іншими типами шуму, які можуть мати нерівномірний розподіл енергії по частотах, сірий шум вважається шумом з рівномірним спектром. Він відповідає акустичній кривій постійної гучності на всіх частотах, тобто для людського вуха він має однакову гучність на всіх частотах (рис. 1.4) [9].

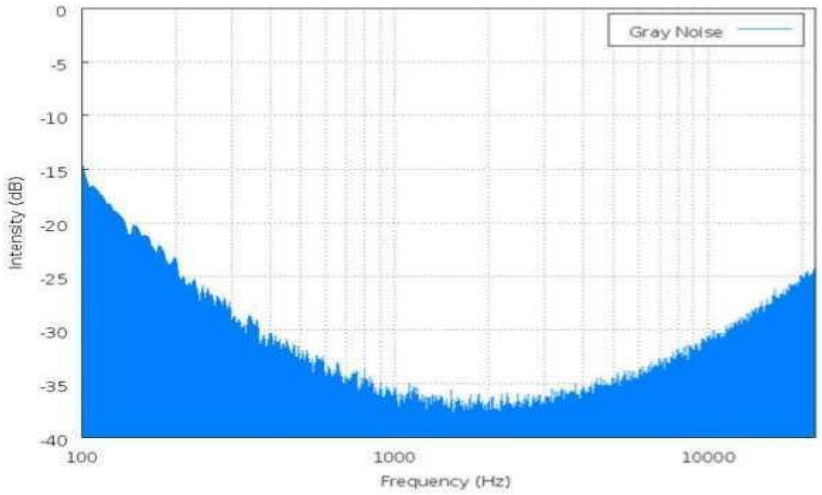


Рис. 1.4 Спектр сірого шуму

Також на розпізнавання звукових даних впливають такі явища, як: реверберація (зазвичай утворюється у великих приміщеннях і що характеризується багаторазовим відображенням звуків від стін), дифракція (викликана накладенням звукових хвиль одна на іншу при проходженні перешкод), рефракція та інші явища, що мають менший вплив на розпізнавання звуків у галузі розроблюваного методу через умови та середовища, в якому вони відбуваються.

Незважаючи на те, що розпізнавання звуків і людської мови все більше інтегруються в наше повсякденне життя, вони також розвиваються, що полегшити: спілкування за допомогою голосового введення, пошук інформації на основі звукових моделей, керування широким спектром різних пристроїв. На жаль, правильне розпізнавання людської мови є не тільки дуже ресурсномістким завданням, а й вимагає досить чіткого і чистого людського голосу.

Поєднання вищезгаданих явищ і шуму, навіть кожного окремо, значно ускладнює завдання розпізнавання людської мови. Звідси випливає висновок, що: «розробка методу розпізнавання голосових команд з використанням штучних нейронних мереж з шумоподавленням» актуальна на даний момент.

Рис. 1.5 Системи розпізнавання мови

Усі сучасні описи мовлення певною мірою імовірнісні. Це означає, не існує чітких інтервалів між одиницями або між словами. Системи розпізнавання мови можна розподілити на кілька класів, залежно від того, які послідовності слів вони можуть аналізувати:

1. Окремі слова: ці системи фокусуються на вимові окремих слів, і кожне слово вимовляється окремо, з відокремленою тишею навколо нього. Це зручно для використання у ситуаціях, коли потрібно надати команду, що складається з одного слова.

2. Зв’язна мова: подібно до попереднього типу, ці системи дозволяють послідовне вимовляння фраз з мінімальними паузами між ними.

3. Безперервна мова: ця форма розпізнавання наближена до природної мови, де користувач може вимовляти слова без відчутних пауз.

4. Суцільна мова (спонтанна мова): ці системи здатні розпізнавати природну мову людини. Це одна з найбільш складних задач, оскільки вимагає розпізнавання інтонацій, пауз, тонування іншими словами, врахування контексту і т. д.

Розмір словника має велике значення для роботи системи розпізнавання мови. Словниковий запас - це набір слів, на які система може опиратися. Коли кількість слів невелика, розпізнавання стає простішим у порівнянні з великим словниковим запасом. Залежно від розміру словника розрізняють такі типи:

- Малий словник (розпізнається від 1 до 100 слів або словосполучень)
- Середній словник (розпізнається від 101 до 1000 слів або словосполучень)
- Великий словник (розпізнається від 1001 до 10000 слів або словосполучень)
- Дуже великий словник (розпізнається понад 10 000 слів або словосполучень)

Кожна людина має свої особливості та властивості голосу. Виходячи з цього, можна виділити два типи систем:

1. Дикторозалежні моделі. Система що враховує унікальні характеристики вимови кожного окремого диктора під час процесу розпізнавання мовлення. Ці моделі навчаються відокремлювати і розпізнавати голосові команди чи тексти залежно від особливостей вимови та акценту конкретних людей. Вона навчається на конкретному користувачеві, перш ніж може розпізнавати сказане. Така система працює добре, коли до системи звертається лише один користувач.

Системи, залежні від ораторів, зазвичай здатні розпізнавати мову в різних контекстах (слова, фрази). Однак незалежні від оратора системи можуть розпізнавати мову у різних користувачів, обмежуючи мовний контекст.

2. Диктороне залежні моделі. Програмне забезпечення, яке не потребує навчання і не залежить від мовця, використовується в автоматизованих телефонних інтерфейсах. Такими системи можуть користуватися різні особи без навчання, щоб розпізнавати мовленнєві шаблони кожної людини.

У контексті комп'ютерного зору та розпізнавання образів, жест - це форма невербальної комунікації, в якій рухи людського тіла або частин тіла використовуються для передачі інформації. Жести включають в себе рух однієї або кількох частин людського тіла, можуть бути емоційно-навантаженими і служать інформаційним засобом взаємодії.

Жести рук - це аспект мови тіла, який передається за допомогою центру долоні, положення пальців і форми руки. Жести рук можна поділити на статичні та динамічні. Статичні жести відноситься до постійної форми руки, тоді як

динамічні жести складаються з серії рухів рук, наприклад, махання. У жестах присутні різноманітні рухи рук, наприклад, рукостискання відрізняються від людини до людини і можуть змінюватися в залежності від часу і місця. Основна відмінність між позами і жестами полягає в тому, що пози фокусуються на формі рук, тоді як жести - на русі руки. Основні підходи до дослідження жестів рук можна класифікувати залежно від датчика: рукавички чи зображення [6].

Класифікація жестів враховує спосіб обробки інформації. Статичні жести, які вимагають точної позиції, аналізуються після введення даних. У випадку динамічних жестів обробка відбувається в реальному часі під час виконання дії. Процес спрощеного розпізнавання рухів має такі основні етапи (рис. 1.6):

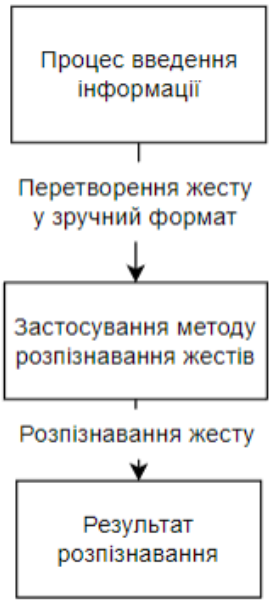


Рис. 1.6 Спрощена структура системи розпізнавання жестів руки

Точка входу - це пристрої, які використовуються для введення інформації в систему. Вона трансформує вхідну інформацію у цифрову форму і передає його методу розпізнавання, який порівнює отриману інформацію з шаблонами, які є у базі даних. В результаті метод повертає розпізнаний жест, який можна використовувати в подальших процесах залежно від завдання. Наприклад, сенсор Kinect використовується для взаємодії з ігровою консоллю Xbox 360 за допомогою словесних команд, положень тіла та жестів рук [7]. У цьому підході

жести можуть використовуватися для управління відеогрою і виконання відповідних рухів персонажа на основі вхідної інформації.

Жести можна розглядати як рухи та дії певних частини людського тіла, які мають чітке значення для передачі інформації або вираження емоцій. Це означає, що жести є явними символом і існують на тому ж рівні, що й інші засоби комунікації.

Розпізнавання жестів має широкий спектр застосувань:

- віртуальна реальність - жестове управління для віртуальної реальності є ключовим в зануренні (стан в якому людина втрачає самоусвідомлення), що може забезпечити максимальну реалістичність;
- робототехніка - для того, щоб відтворити щось близьке до людської поведінки, необхідно ретельно проаналізувати існуючі жести, щоб запрограмувати роботів на поведінку, подібну до людського тіла, яке здатне відтворювати надточні операції;
- інтерфейсні рішення - можливість натуральної взаємодії з додатками виключаючи використання механічного контакту або пристроїв є зрозумілим та зручним способом інтеракції та може постати альтернативою поточним пристроям введення;
- розробка ігор - методи керування та взаємодії з ігровими системами, яким потрібні акселерометри, камери та інші пристрої для детекції рухів та уникнення натискання кнопок, необхідних для традиційних контролерів відеоігор;
- мова жестів - так як це основна комунікативна одиниця для людей з порушеннями мовлення та слуху.

Поточне дослідження ставить за мету покращення існуючих методів розпізнавання жестових мов та створення моделі клієнт-серверної взаємодії.

Дизлексія різного ступеня прояву є проблемою для людей. Особи з такою вадою використовують багато різних взаємодії, але найважливішим для них є використання мови жестів.

Моделі додатків для розпізнавання мови жестів для людей з вадами слуху є

дуже важливими, оскільки такі додатки дозволяють зменшити розрив в спілкуванні для людей з вадами.

Основною метою цього дослідження є розробка системи на основі зорових можливостей, для вилучення рухів мови жестів із відеопослідовностей за допомогою попередньо навченої моделі та подальшої клієнт-серверної комунікації. Система на основі зору була обрана тому, що вона надає більш простий, зрозумілий та основне доступний спосіб спілкування між людьми та комп'ютерами.

Мова жестів дозволяє взаємодіяти глухим та німим людям. Спільноти використовують жести для свого спілкування, коли передача голосу неможлива або коли важко писати, але можна бачити. У такі моменти мова жестів є єдиним способом обміну інформацією між людьми. Зазвичай люди використовують мову жестів, коли не хочуть говорити, але для глухих та німих людей жестова мова є єдиним засобом спілкування. Мова жестів так само важлива, як і усна.

У світі стільки мов жестів, скільки загальних мов, але, на жаль, не існує універсальної мови, і кожен регіон має свої особливості. Наприклад, американська жестова мова, українська жестова мова тощо. На рисунку 1.7 зображено алфавіт української жестової мови.

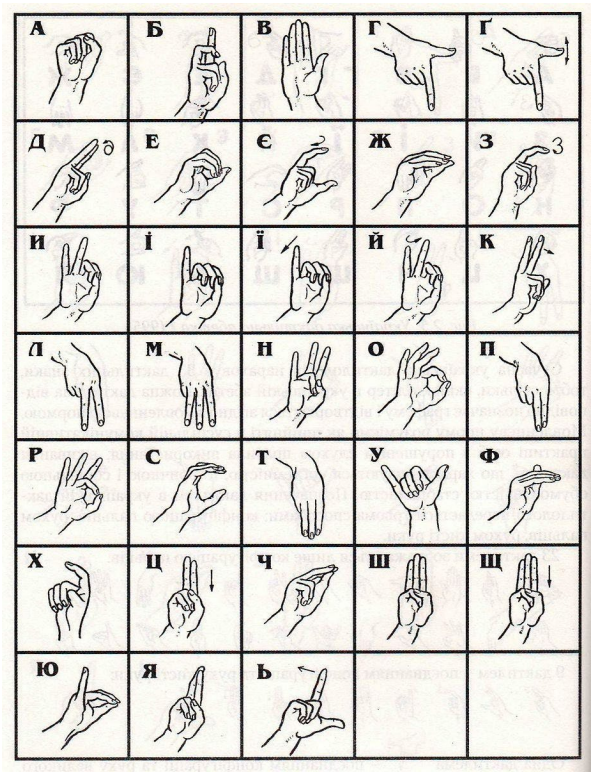


Рис. 1.7 Алфавіт української жестової мови

Мова жестів може використовуватися за допомогою жестів однією або двома руками, у поєднанні з мімікою, формами або рухами рота та губ, а також у поєднанні з положенням тіла. Ці мови в основному використовуються в культурах глухих і слабочуючих людей для спілкування. Використання жестової мови неглухих людьми менш поширене, але досить поширене, оскільки часто доводиться спілкуватися з глухими людьми за допомогою мови жестів.

Однією з важливих неправильних думок про мови жестів є те, що вони якимось чином залежать або походять від розмовної мови (звук і письмо), що ці мови були винайдені чуючими людьми, але насправді це не так. Крім того, мову жестів часто вважають дактилологією, фактично використовується в мові жестів для вимови імен, географічних назв, а також конкретних термінів у розмовній мові, або артикуляційними жестами, які використовуються чуючими для передачі інформації за допомогою жестів, які граматично ідентичні усній мові. Насправді мови жестів практично повністю незалежні від вербальних мов і продовжують розвиватися: з'являються нові жести, зникають старі - і більшість з них мало пов'язана з розвитком вербальної мови. Кількість мов жестів в країні не пов'язана

з кількістю в ній усних мов. Навіть у країні з кількома розмовними мовами, може існувати спільна мова жестів, а в деяких країнах, навпаки.

Лінгвістичні підтверджено, що жестові мови насправді є складними та багатими, аналогічно звуковим мовам. І це все попри поширений міф про їхню несправжність. Професійні лінгвістичні дослідження показали, що жестові мови мають усі ключові компоненти, що роблять їх повноцінними мовами.

З лінгвістичної точки зору, жестові мови такі ж багаті та складні, як і будь-яка звукова мова, незважаючи на загальне ставлення до них як до несправжніх мов. Дослідження, проведені професійними лінгвістами, показали, що жестові мови мають усі елементи, що характеризують їх як автентичні мови.

Отже, висновок полягає в тому, що розмовна жестова мова та вербальна мова з жестовим супроводом - це дві різні системи. Перша використовується у неформальному спілкуванні, коли жести є основним способом передачі повідомлення, а друга - це використання жестів для підтримки словесної мови в офіційному оточенні, коли слова використовуються основним чином. У цій роботі проводитиметься розпізнавання жестового супроводу вербальної мови, коли жести і слова використовуються разом для передачі повідомлення, проте буде звертатися увага на розпізнавання жестів у відповідь на окремі слова. Окрема увага буде звернена на способи комунікації між клієнтом та сервером для покращення загальної моделі.

1.3 Огляд існуючих рішень: SignAll та Leap Motion

SignAll - це продукт, який використовує технологію автоматизованого перекладу ASL для забезпечення зв’язку між людьми з нормальною та порушеною функцією слуху. Система встановлюється статично в офісах, центрах обслуговування клієнтів, у залах засідань тощо.

Особа з проблемами слуху одягає пару рукавичок та знаків, кольори на рукавичках допомагають технології розрізнити пальці. Людина, що слухає, використовує голос, мова підбирається автоматичною системою розпізнавання

мовлення.

SignAll - це група захоплених своєю справою розробників та дизайнерів, які займаються пошуком інноваційних рішень, що уможливлюють спонтанну комунікацію між глухими та чуючими людьми. Це призводить до збільшення можливостей для зв'язку, особистого та професійного розвитку, безперешкодної інтеграції між глухими та чуючими.

SignAll була заснована Золтом Роботкою та Яношем Ровнєм і є спільнотою Dolphio, провідної компанії з розробки програмного забезпечення в Будапешті, Угорщина.

Переклад мови жестів довгий час вважався найскладнішим завданням комп'ютерного зору, але коли до команди приєднався перший глухий співробітник (угорсько-перський програмний інженер), вони надихнулися на створення автоматизованого перекладу жестової мови. Те, що починалося як технічний виклик і проєкт, перетворилося на офіційну компанію. Розвиток технологій у поєднанні з інноваціями кількох математиків та інженерів, SignAll зміг підготувати концепцію, яка б ознаменувала стартову лінію в ряді успіхів.

Команда ретельно відібрала найкращих лінгвістів, які спеціалізуються на розмовній та жестовій мовах. Вони також визнали необхідність постійного вкладу з боку глухих громадян.

Хоча у світі існує багато різних жестових мов - американська (ASL) є найбільш поширена і підтримувана у світі.

Характеристики:

- спілкування можливе для людей без порушень слуху та з ними;
- робота з розмовною англійською мовою;
- навчальний компонент для навчання мови жестів
- переклад на англійську (розпізнавання включаючи не ручні маркери);
- наявність чату з відображенням розмови для мовця та слухача;

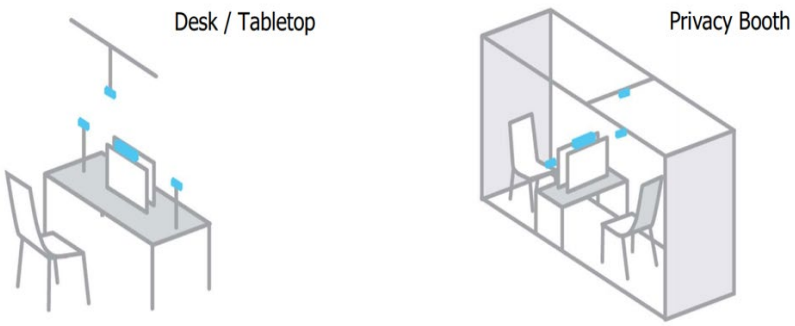


Рис. 1.8 Конфігурація

Система:

- камери та кріплення для них
- освітлення
- персональний комп’ютер та монітор
- планшет/ сенсорний екран
- спеціальні рукавички

Leap Motion - це пристрій, який використовує інфрачервоні датчики для виявлення рухів рук у просторі. Він був розроблений компанією Leap Motion, яка була придбана компанією Ultraleap у 2019 році.

Сам пристрій складається з двох камер та кількох інфрачервоних світлодіодів. Вони відстежують інфрачервоне світло з довжиною хвилі 850 нанометрів, яка знаходиться за межами видимого світлового спектру. Світлодіоди пульсують синхронно з частотою кадрів камери, що значно знижує споживання енергії та підвищує інтенсивність.

Ширококутні лінзи використовуються для створення великої зони взаємодії для виявлення рук користувача. Leap Motion Controller має зону взаємодії, яка розширюється від 10 см до 60 см і більше, простягаючись від пристрою в типовому полі огляду 140x120°. Модуль камери Stereo IR 170 має ще більшу зону взаємодії, що розширюється від 10 см до 75 см і більше, з типовим полем огляду 170x170° (мінімум 160x160°).

Вона має форму перевернутої піраміди для Leap Motion Controller та

перевернутої конусоподібної форми для Stereo IR 170. Це досягається перетином полів огляду бінокулярних камер.

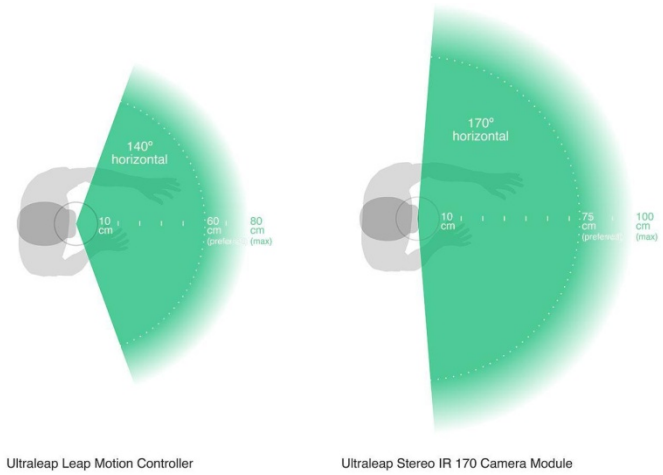


Рис. 1.9 Горизонтальна зона взаємодії

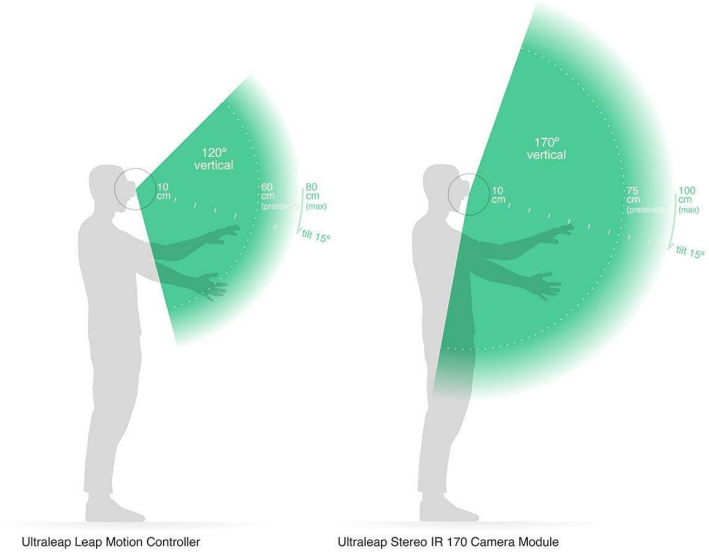


Рис. 1.10 Вертикальна зона взаємодії

Дані мають форму однотонових стереозображень інфрачервоного спектру. Також вони розділені на ліву та праву камери. Зазвичай ми побачимо лише об'єкти, безпосередньо освітлені світлодіодами пристрою. Однак лампи розжарювання, галогенові лампи та денне світло також освітлюватимуть сцену в інфрачервоному спектрі. Також можна помітити, що деякі речі, наприклад бавовняні сорочки, можуть здаватися білими, навіть якщо вони темні у видимому

спектрі.

Що важливо - платформа відстеження рук не генерує карту глибини - натомість вона застосовує до початкових даних датчиків алгоритми і обробляє зображення. Після компенсації фонових об’єктів, освітлення, тощо - генерується 3D-зображення того, що "бачить" пристрій. Одна з ключових переваг на даний момент - покращена обробка інтенцій двох рук одночасно.

В кінці сервіс видає результат, виражений у вигляді серії кадрів або знімків, що містять усі дані відстеження - до транспортного протоколу. Через цей протокол сервіс спілкується з панеллю контролю, вбудованими та клієнтськими веб-бібліотеками. Клієнтська бібліотека організовує дані в структуру об'єктно-орієнтованого API, керує історією кадрів і надає допоміжні функції та класи [26].

1.4 Технології клієнт-серверної взаємодії

З розвитком інформаційно-комунікаційних технологій збільшується складність інформаційних систем та обсяг даних у них. Кожна прикладна програма відображає частину реального світу і містить його формалізований опис у вигляді даних. Великі обсяги даних зберігаються окремо від коду програми, що виконується, і організовані у вигляді баз даних. Для роботи з даними використовуються спеціальні програмні пакети та системи управління базами даних (СУБД).

Переважним інструментом обробки даних сьогодні є клієнт-серверна архітектура, яка дозволяє одній прикладній програмі взаємодіяти з іншими подібними програмами в мережі, обмінюючись даними через сервер баз даних.

Під клієнт-серверною технологією мається на увазі такий спосіб взаємодії програмних компонентів, у якому вони утворюють єдину систему. Як випливає з назви, існує клієнтський процес, який потребує певних ресурсів, і серверний процес, який ці ресурси надає. Їм не обов'язково перебувати на одному комп'ютері. Сервер зазвичай розташовується на одному вузлі локальної мережі, а клієнти – на інших вузлах.

У випадку з базами даних клієнт керує інтерфейсом користувача і робочою логікою, діє як робочої станції. Клієнт приймає запит від користувача, перевіряє синтаксис та формує запит до бази даних SQL або іншій мові бази даних, відповідно до логіки клієнтської програми. Потім він надсилає повідомлення на сервер, чекає відповіді та форматує отримані дані для подання користувачеві. Сервер приймає та обробляє запити до бази даних, після чого надсилає отримані результати назад клієнту. Така обробка включає перевірку облікових даних клієнта, забезпечення вимог цілісності, а також запит та оновлення даних. Крім того, підтримується управління паралелізмом та відновленням.

Клієнт-серверна архітектура має низку переваг:

- підвищується загальна продуктивність системи: оскільки клієнти та сервер знаходяться на різних машинах, їх процесори паралельно запускають різні процеси.
- коли один сервер відмовляє, інші клієнти можуть продовжувати працювати з іншими доступними серверами, що забезпечує більшу надійність системи;
- і навпаки - ця архітектура легко масштабується. Кілька клієнтів може звертатися до одного сервера, що дозволяє обслуговувати багато користувачів одночасно;
- витрати зв'язок скорочуються. Частина операцій виконують клієнтські комп'ютери, а через мережу надсилаються лише запити до баз даних, що дозволяє істотно скоротити обсяг даних, що надсилаються через мережу;
- знижується вартість обладнання; досить потужний комп'ютер з великим пристроєм, що запам'ятовує, потрібен тільки для сервера - для зберігання і управління базою даних;
- підвищується рівень узгодженості даних, оскільки кожній клієнтській програмі не доведеться виконувати власну перевірку цілісності.

Далі розвиток дворівневої клієнт-серверної архітектури передбачає розподіл функцій клієнта ще на два рівні. У трирівневій клієнт-серверній архітектурі «тонкий» клієнт управляє лише інтерфейсом користувача, а середній рівень

обробки даних відповідає за решту клієнтської програми. Третій рівень - це сервер бази даних. Такий підхід виявився більш ефективним у вебі, де звичайний веб-браузер може виступати в якості клієнта.

У ході розвитку клієнт-серверної технології змінювалися, як і змінювалися методи їх реалізації. Далі розглянуться способи організації доступу до даних та особливості в контексті клієнт-серверної технології

Архітектура клієнт-сервер дворівневого типу (див. Рис. 1.10) передбачає використання спеціалізованого програмного забезпечення на виділеному сервері, яке є сервером баз даних, наприклад, PostgreSQL або MySQL, операційна система якого керує цим процесом. Система управління базами даних (СУБД) розділена на дві частини: клієнтську та серверну. Основу сервера баз даних становить мова запитів - SQL. Запити SQL, відправлені клієнтом на сервер баз даних, виконують пошук та відбір необхідної інформації безпосередньо на сервері. Після цього обрані дані транспортуються по мережі до клієнта (див. Рис. 1.11). Таким чином, обсяг переданих даних по мережі значно зменшується.

Цей принцип широко використовується в базах даних, а також у інших галузях застосування комп’ютерних технологій. Розглянемо деякі з найвідоміших:

1. Електронна пошта. Тут поштова програма (Outlook, Email тощо) виступає як клієнт, який підключається до поштового сервера, на якому знаходиться електронна скринька. Поштова програма «спілкується» з сервером (надсилає запити на отримання особистих повідомлень, запити на видалення листа тощо) за стандартними протоколами найпоширеніші POP3 та SMTP.

Рис. 1.11 Дворівнева архітектура «клієнт-сервер»

2. WWW – найвідоміший сервіс Інтернету, завдяки якому став надзвичайно популярним серед користувачів. У цьому випадку під час перегляду веб-сайтів використовується технологія клієнт-сервер. Веб-браузер діє як клієнтська програма, яка надсилає запити на сервер, на якому розміщено веб-сторінки, використовуючи протокол передачі даних (HTTP). Протокол HTTP також широко використовується в «тонких» клієнтах, які розглянемо пізніше.

Практично всі послуги, які надає Інтернет, засновані на цій технології, оскільки при отриманні довільних даних з мережі програма користувача посилає запит, що містить певні інструкції, на віддалений сервер. Наприклад, видалення листа - при керуванні поштовою скринькою, відправка вмісту певного файлу клієнту - при роботі з FTP, запит інформації про користувача - при спілкуванні через ICQ. Подібні приклади можна навести практично для більшості глобальних мережевих сервісів.

Виділяють такі особливості використання дворівневої архітектури:

- можливість використання різних інструментів, які надає СУБД через пряме підключення до сервера бази даних;
- навантаження на мережу суттєво знижується, бо надсилаючи запити та обробляючи їх на сервері, клієнт обмінюється із сервером лише необхідними даними;

- складніше масштабувати, ніж трирівневу архітектуру, оскільки доводиться використовувати різні клієнти для різних операційних систем;

- потребує додаткових витрат на облаштування та встановлення клієнтської частини;

- при модифікації серверної частини, додавання нових функцій, зміні базової СУБД, як правило, клієнтську частину необхідно замінити або оновити.

Трирівнева клієнт-серверна архітектура функціонує в мережах Інтранет та Інтернет. Клієнтська частина («тонкий клієнт»), з якою працює користувач, - це веб-браузер або клієнтська програма, яка взаємодіє з веб-сервісами. Вся програмна логіка передається на сервер додатків, який забезпечує формування запитів до бази даних, які потім передаються на виконання на сервер бази даних. Сервером програм може бути веб-сервер або спеціалізована серверна програма (наприклад, Oracle Forms Server). Схема такої конструкції представлена на рисунку 1.12.

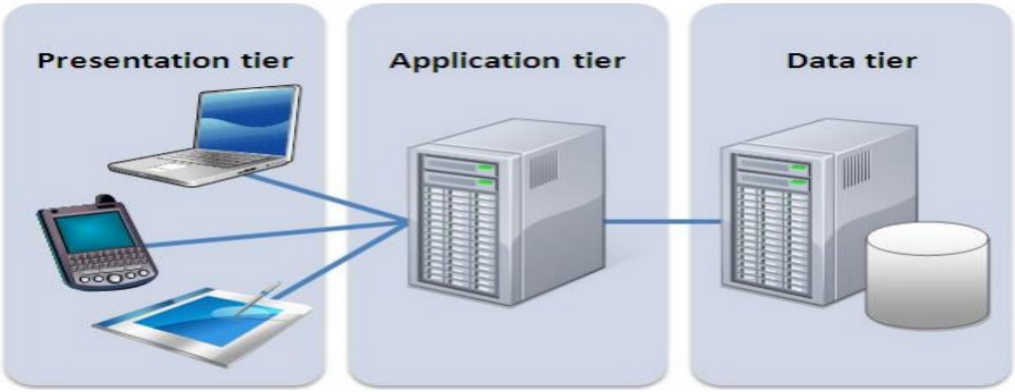


Рис. 1.12 Схема трирівневої архітектури «клієнт-сервер»

Така трирівнева архітектура клієнт-сервер є основою для розробки веб-сайтів і порталів, які використовуються для зберігання даних. Наприклад, численні інтернет-магазини, пошуково-довідкові сервери, системи Інтернет-телефонії та обміну повідомленнями в режимі реального часу, системи передачі відео в Інтернеті тощо. Для програмних засобів, що працюють на основі протоколу HTTP, характерним є те, що користувачі працюють з ними за

допомогою звичайних браузерів, такі як Microsoft Internet Explorer. У той же час вони доступі, як і звичайні статичні HTML-сторінки, через URL-адресу.

Однак ці програмні інструменти зовсім не схожі на статичні сторінки HTML. Через веб-сервер вони можуть отримати доступ до різних об'єктів, сервісів та систем, наприклад баз даних. Таким чином у відповідь на запит, введений користувачем у вікно браузера, веб-сервер може створити звіт і відобразити його у цьому вікні. Для отримання результатів веб-сервер формує запит до бази даних [10].

Важливо, що додатки на основі веб-технологій могли працювати не тільки в Інтернеті, а й у локальних мережах. Використання браузера комп'ютера користувача як основного засобу доступу до баз даних значно полегшує обслуговування великих локальних мереж. При цьому спрощується не тільки процедура встановлення програмного забезпечення на мережевих робочих станціях, але й спрощується підтримка баз даних та інших систем, що працюють централізовано на спеціально виділених серверах.

2 ПРОЄКТУВАННЯ МОДЕЛІ КЛІЄНТ-СЕРВЕРНОЇ ВЗАЄМОДІЇ З ВЕБ-ДОДАТКАМИ НА ОСНОВІ ГОЛОСОВОГО І ЖЕСТОВОГО КЕРУВАННЯ

2.1 Загальна структура систем

Проєктування системи відео-зв'язку з урахуванням комбінації нейронних мереж різної архітектури, де модель повністю управляється даними без необхідності вручну розробляти шаблони або правила. Оскільки здатність до узагальнення є мірою продуктивності нейронної мережі, високої продуктивності в розпізнавання образів можна досягти, об’єднавши кілька моделей. Крім того, оскільки інтерпретація жестів проходить, коли відео потрапляє в канал обробки, необхідно знайти найбільш оптимальний спосіб передачі потоку відео.

Розроблена система включає наступні складові:

- модуль введення даних
- модуль обробки даних - поєднання нейромережевих моделей різних архітектур;
- процесор набору даних;
- модуль передачі відео;
- модуль серверної обробки даних.

Вхідні дані для проєктованої системи є набір відеоданих X. Кожне відео відповідає одному елементу з набору класів Y. У цьому контексті кожен конкретний елемент є лінгвістичним представленням конкретного жесту. Хай окрема вхідна послідовність $X = (x_1, \dots, x_N)$ та вихідна послідовність $Y = (y_1, \dots, y_N)$ є вектором фіксованої довжини N. Крім того, Y безпосередньо залежить від вхідної відео послідовності X. Тобто існує невідома цільова залежність – зіставлення $X \to Y$ значення яких відомі тільки на елементах кінцевої навчальної вибірки $X^N = \{(x_1, y_1), \dots, (x_N, y_N)\}$ маючи вхідну послідовність та послідовність

вербальних уявлень жестів, наша модель має на меті класифікувати будь-який об'єкт x ∈ *X*.Для вирішення цієї проблеми було вирішено використовувати штучні нейромережі. Мережі прямого поширення є простим засобом функціональної апроксимації і можуть бути використані для розв’язання задач класифікації. Їх ефективність досить висока, оскільки вони фактично генерують велику кількість регресійних моделей (що можуть бути використані для вирішення завдань статистичної класифікаці). Однак будь-який метод, заснований на нейронних мережах, не може генерувати класифікатори необхідної якості, якщо система не має достатньо повного набору прикладів для завдання, яке вона має вирішувати [6].

До теперішнього часу розроблено багато різноманітних типів нейромереж і саме тому рішення про вибір тієї чи іншої реалізації NN зазвичай приймається на основі експертних оцінок або емпіричних експериментів, які однозначно зможуть вказати найбільш ефективну модель архітектури. В обох випадках є недоліки - вимога наявності експертних знань для оцінки, що не завжди можливо, а емпіричні експерименти через величезну різноманітність NN можуть призвести до значних ресурсних та тимчасових витрат. Отже, раціональним рішенням є використання архітектури нейромережі, яка є проста у реалізації і водночас добре відома та вивчена.

Відео складно класифікувати, оскільки воно містить як часові, так і просторові характеристики. Тобто кожен кадр відео містить як інформацію, важливу для самого кадру, так і контекст цього кадру щодо попередніх кадрів в часі. Тому, за прикладом [7] для класифікації просторових та тимчасових ознак було вирішено використати комплекс нейронних мереж, що складається з двох різних моделей. Згорткова нейронна мережа (ЗНМ), використовується для класифікації просторових ознак, а рекурентна нейронна мережа (РНМ), вибирається для класифікації тимчасових ознак. Навчання ЗНМ проводиться на кадрах, отриманих з послідовності відео навчальної вибірки. Далі на основі навченої моделі ЗНМ прогнозуються окремі кадри для отримання послідовності вихідних даних шару вибірки для кожного відео. На наступному етапі

послідовність шару субдискретизації подається на вхід РНМ для навчання тимчасових ознак. Схема навчання моделі представлена на рисунку 2.1.

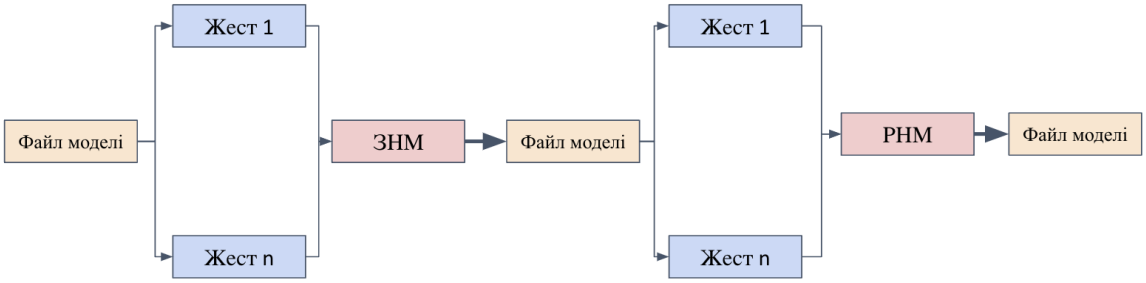


Рис. 2.1 Схема взаємодії під час навчання моделі

2.2. Класифікація просторових об'єктів та символів часу

CNN або згорткова нейронна мережа (рис. 2.2) є однією з найпоширеніших категорій нейронних мереж, особливо для об'ємних даних, таких як зображення та відео. Згорткова нейронна мережа та її різновиди також вважаються найкращими алгоритмами для пошуку об'єктів на сцені з точки зору точності та швидкості. З 2012 року ці нейромережі посідали перше місце у відомому міжнародному конкурсі з розпізнавання зображень ImageNet Large Scale Visual Recognition Challenge [8].

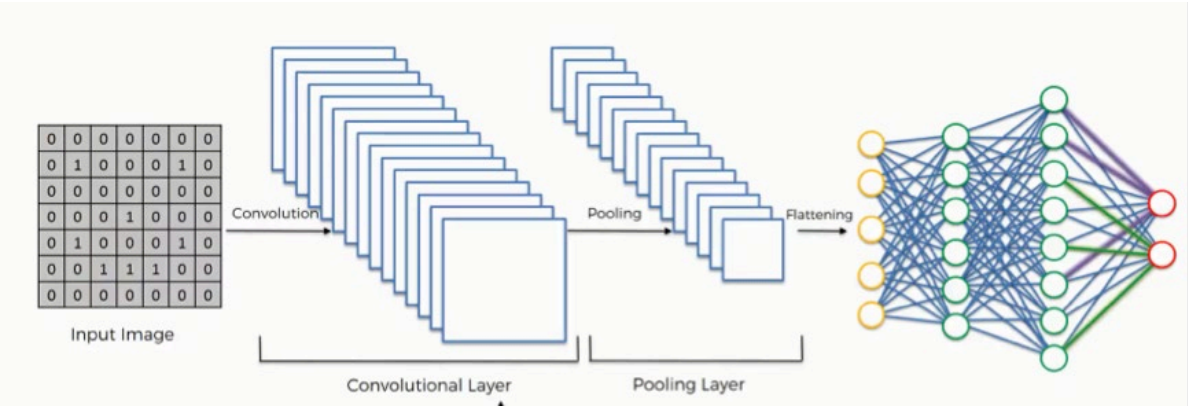


Рис. 2.2 Модель згорткової нейронної мережі

ЗНМ працює подібно до стандартних нейронних мереж. Ключова відмінність полягає в тому, що кожна одиниця в шарі ЗНМ є двовимірним (або більше) фільтром, який згортається на вході цього шару. Це важливо, коли потрібно навчитися закономірностям з багатовимірних вхідних даних, таких як зображення або відео, оскільки фільтри ЗНМ мають подібну (але меншу) просторову форму до вхідних даних, використовуючи спільне використання параметрів, щоб значно зменшити кількість змінних, що підлягають навчанню.

На сьогоднішній день архітектура ЗНМ є досить складна, містить десятки внутрішніх шарів та мільйонів параметрів, що робить такі архітектури дуже складними для візуалізації, вони займають сотні тисяч байт дискового простору і вимагають великих часових та апаратних можливостей для їх навчання. Тому в цій роботі використовується архітектура ЗНМ - Inception-V3, розроблена компанією Google [10]. Архітектура цієї нейронної мережі зображена на рисунку 2.3.

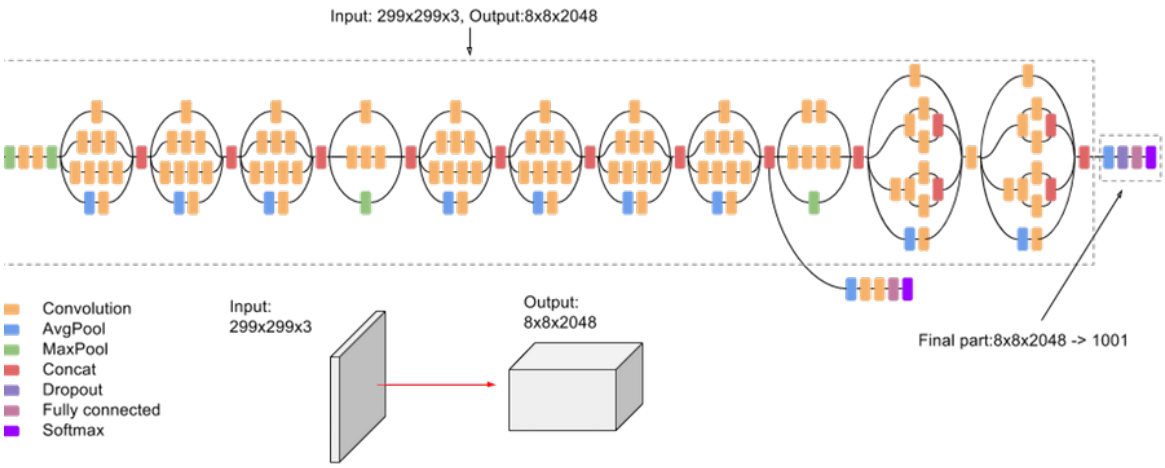


Рис. 2.3 Архітектура ЗНМ Inception-V3

Для реалізації цієї нейромережі використовується бібліотека Tensor Flow. Inception-V3 - це велика модель класифікації зображень з безліччю параметрів, здатна розрізняти різні типи зображень. Для досягнення результатів достатньо використовувати метод “Transfer Learning”, тобто перенавчити останній шар моделі завдяки новим образам та класам. Завдяки цьому можна скористатися

перевагами тижнів попереднього навчання на мільйонному датасеті ImageNet не навчаючи власну складну ЗНМ з нуля, що дозволяє навчати модель на меншому наборі даних, покращуючи узагальнюючі можливості та заощаджувати час і апаратні ресурси.

Трансферне навчання - це здатність системи розпізнавати та застосовувати знання та навички, набуті в минулих завданнях, до поточних завдань та даних. Специфікація рівнів, на яких застосовується “Transfer Learning”.

Як згадувалося вище, для класифікації часових ознак використовуються RNN або рекурентні нейронні мережі (РНМ). На відміну від прямих нейронних мереж, існують звані рекурентні нейронні мережі. У перших інформація передається мережею нейронів прямолінійно від шару до шару, тоді як у РНМ нейрони здатні обмінюватися інформацією між собою. Для прикладу, крім нового фрагмента вхідних даних, нейрон отримує ще додаткову інформацію з попереднього стану мережі. Таким способом, мережа можу «запам’ятовувати». Це докорінно змінює принципи її роботи та дає можливість аналізувати ряди даних, де важливим є знання порядку значень.

Якщо прямі нейронні мережі можна назвати «простими» функціями, то рекурентні нейронні мережі майже напевно можна назвати програмами. Дійсно, пам'ять РНМ є повною за Аланом Тьюрингом при правильному виборі ваги нейронна мережа може успішно симулювати функціонування комп'ютерних додатків.

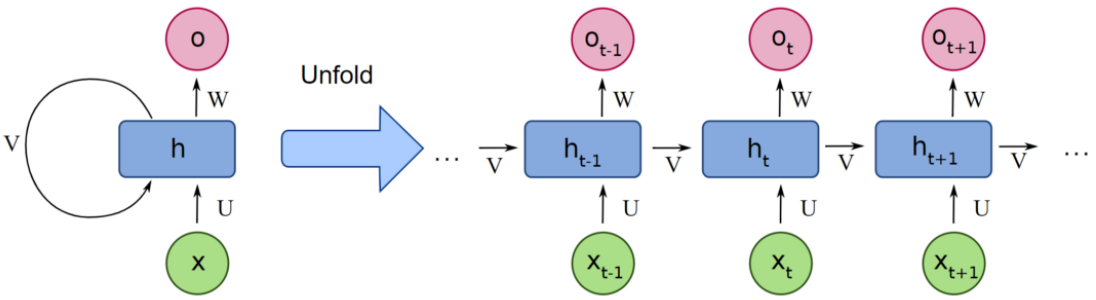


Рис. 2.4 Схема 1-но шарової РНМ

З кожним циклом роботи у внутрішній шар нейронів надходить вхідний датасет X та інформація попереднього стану внутрішнього шару A, з яких формує відповідь h. Ця модель описується таким чином[11]:

$h(t)$ стан прихованого шару для входу $h(t)$ ($h(0) = 0$):

$$h(t) = f(V \cdot x(t) + U \cdot h(t-1) + b_h), \quad (2.1)$$

де $h(t)$ - вхідний вектор номер t;

U - матриця ваг вхідного шару;

b_h - вектор зміщень прихованого шару;

V - матриця ваг вихідного шару;

f - функція активації шару.

$y(t)$- вихід мережі для входу $h(t)$:

$$y(t) = f(W \cdot h(t) + b_y), \quad (2.2)$$

де W – матриця ваг зворотних зв’язків прихованого шару;

b_y - вектор зміщень вихідного шару;

f – функція активації шару.

Однією з важливих можливостей ідей РНМ є здатність пов'язати інформацію минулу з теперішніми завданням. Наприклад, знання попередніх кадрів відео може допомогти зрозуміти поточний кадр.

На практиці, РНМ не зовсім вдається відслідковувати довгострокові зв'язки, хоча їхні нейрони мають добру «короткотривалу пам'ять». Нейронні мережі з довгою короткочасною пам'яттю або скорочено LSTM-мережі, позбавлені цього недоліку.

LSTM-мережі – це штучні нейронні мережі, які містить цілі обчислювальні блоки в прихованих шарах замість звичайних нейронів (рис. 2.6). Мережевий блок можна описати як вузол «прийняття рішень», який може зберігати значення з будь-якого періоду в минулому.

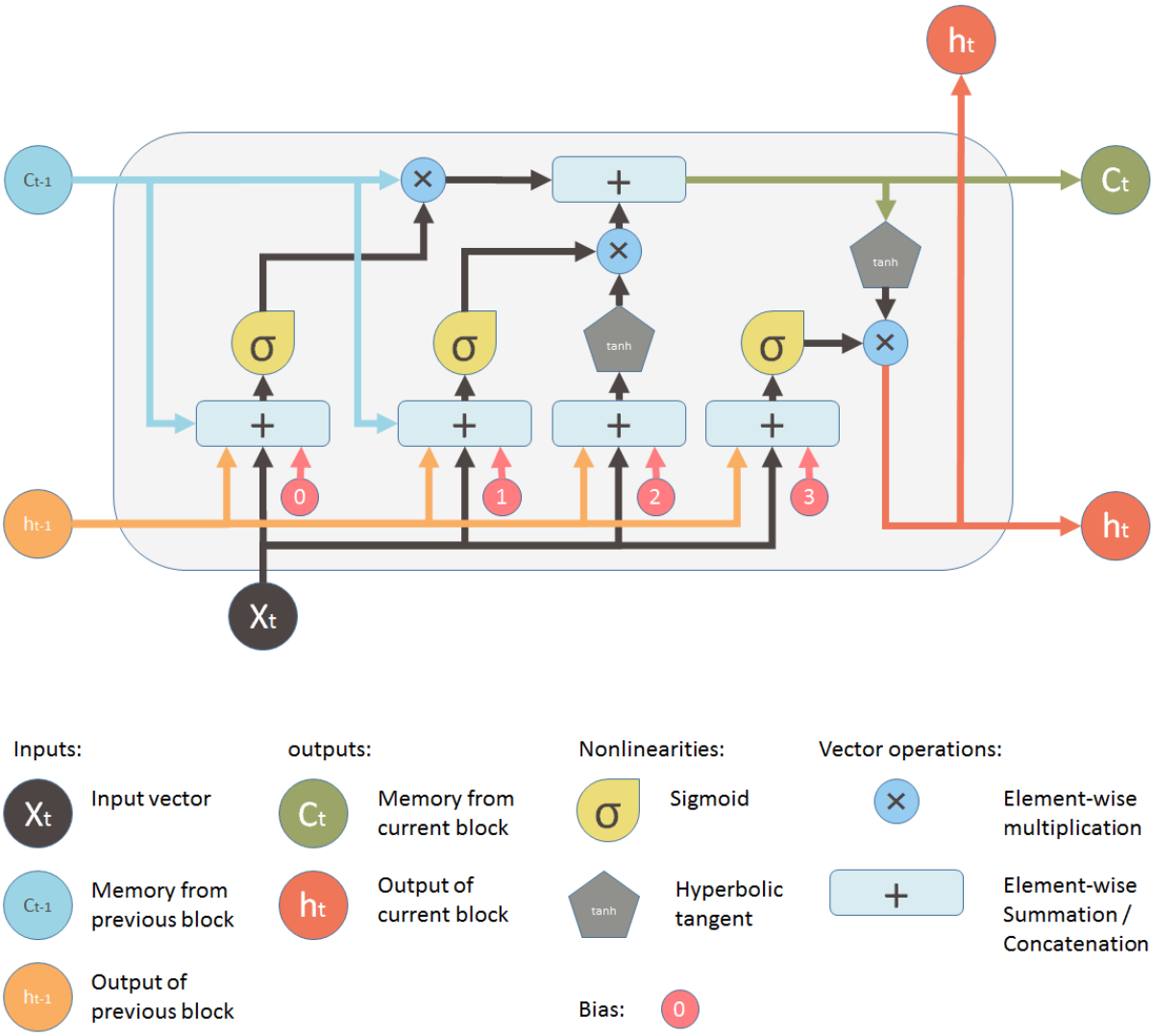


Рис. 2.5 Структура мережевого блоку LSTM

Таким чином, було створено модель РНМ на основі LSTM. Перший шар є вхідним і використовується для передачі вхідних даних для наступних шарів. Розмір цього шару визначається розміром наданих вхідних даних. Наша модель являє собою широку мережу з 256 однорівневих блоків LSTM. За цим шаром слідує повнозв'язковий шар. Нейрони повнозв'язномого шару пов'язані із кожним нейроном попереднього шару. Його наповнює кількість нейронів що дорівнює кількості класів і. В кінці застосовується регресія (лінійна чи логістична) до вхідного датасету. В якості алгоритму оптимізації використовується adam - адаптивна оцінка моменту, яка оптимізує стохастично градієнтний спуск для мінімізації заданої функції втрат «categorical_crossentropy» (для обрахунку помилок).

2.3 Підбір навчальної вибірки та навчання нейронної системи

Машинне навчання сильно залежить від даних. Це найважливіший аспект, який уможливлює навчання за допомогою алгоритму і пояснює, чому машинне навчання стало настільки популярним в останні роки. Але незважаючи на терабайти інформації та наукових знань, якщо введені дані неможливо зрозуміти, алгоритм не буде ефективним. Справа в тому, що всі набори даних недосконалі, дані часто неточні, безладні та часто містять втрачені та помилкові значення. Тому підготовка даних є важливим етапом у процесі проєктування нейромережі. Підготовка даних - сукупність процесів, які допомагають створити набір даних, який найкраще підходить для машинного навчання.

Відбір даних має три основні цілі:

1. обмежити розмір датасету і прискорити процес навчання моделі;
2. усунути шум із даних і покращити здатність моделі робити прогнози;
3. сприяти інтерпретації даних людиною [13].

Зазвичай, під час вибору датасету ми спочатку отримуємо деяке поліпшення точності прогнозування після видалення деяких даних, але потім, у міру видалення більшої кількості даних, точність повільно, але неухильно знижується. Потім, після перевищення певної межі, точність прогнозування починає безперервно знижуватися, і якщо хочемо йти далі шляхом стиснення, повинні вибрати точку відповідно до цілі.

Умови, яким повинні відповідати дані згідно до завдання цього проєкту:

- вибірка повинна містити більше 1900 значень;
- на відео присутні жести ASL;
- дані повинні бути відправлені у форматі відео;
- один жест - одне відео;
- жести повинні виконуватися людиною (а не комп'ютером);
- відео мають максимально близьку тривалість.

В якості тренувального набору даних для даної роботи був обраний датасет, що складається з коротких відеороликів тривалістю 1-2 секунди, де описані слова

американською мовою жестів (ASL) [14]. Було обрано жести з-поміж найбільш поширених у словнику ASL, включаючи як дієслова, так і іменники. Цей набір містить 60 слів, кожне слово 50 відео. Загалом вибірка містить 3000 відео-файлів. У таблиці 2.1 наведено слова, що відповідають жесту, їхні унікальні ідентифікатори та напрямок руки, що представляє жест - H, R - права рука, B - обидві руки.

Таблиця 2.1

Подання тренувальної вибірки

ID	**Слово**	**H**	**ID**	**Слово**	**H**	**ID**	**Слово**	**H**	**ID**	**Слово**	**H**
1	Give	B	16	Chewing-gum	R	31	Breakfast	B	46	Born	R
2	Realize	R	17	Candy	R	32	Birthday	R	47	Drawer	R
3	Run	B	18	Barbecue	B	33	Mock	B	48	Away	R
4	Copy	B	19	Rice	B	34	Where	R	49	Man	R
5	Buy	R	20	Trap	B	35	Last name	R	50	Son	R
6	Bathe	B	21	Deaf	R	36	Country	R	51	Enemy	R
7	Dance	B	22	Perfume	R	37	Uruguay	R	52	Women	R
8	Help	B	23	Patience	R	38	Argentina	R	53	Find	R
9	Catch	B	24	Name	R	39	Food	R	54	Colors	R
10	To land	B	25	None	R	40	Water	R	55	Light-blue	R
11	Appear	B	26	Ship	R	41	Milk	R	56	Bright	R
12	Shut down	R	27	Music	B	42	Sweet milk	R	57	Yellow	R
13	Thanks	B	28	Coin	B	43	Bitter	R	58	Green	R
14	Accept	B	29	Map	B	44	Skimmer	R	59	Red	R
15	Yogurt	B	30	Hungry	R	45	Call	R	60	Opaque	R

Обґрунтуванням вибору саме цього набору даних є відсутність аналогічних даних для українсько, великий обсяг даних у цьому наборі даних. Кожний

відтворювач показує жест. Що важливо - руки повинні контрастно виділятися, що дозволить видалити лишні шуми для полегшення навчання нейромережі.

У 1986 Раммельхарт і Хінтон запропонували алгоритм зворотного поширення помилки на навчання багаторівневої мережі [15]. Численні публікації з промислового використання багатошарових мереж навчених цим алгоритмом довели його ефективність практично.

Основною ідеєю зворотного поширення помилки є шлях отримання оцінки помилки для прихованих нейронів. Невідомі помилки з нейронів прихованих шарів спричиняють відомі помилки нейронів вихідного шару. Що сильніший синаптичний зв'язок між вихідним нейроном та нейроном прихованого шару, тим більший вплив помилки прихованого шару. Як наслідок, оцінити похибку елементів прихованих шарів можна як виважену суму помилок наступних шарів. У процесі навчання інформація поширюється від внутрішніх шарів до зовнішніх, а вага помилки, зробленої мережею, - у зворотному напрямку.

Детально цей алгоритм описано у таблиці 2.2. Припустимо, що мережа має один прихований шар щоб спростити позначення.. Матриця ваг для входів прихованого шару позначена *W*, а матриця ваг, що з'єднує прихований та вихідний шари позначимо *V*. Для індексів будемо використовувати такі позначення: *i - для* входів, індекс *j - для* елементів прихованого шару, а *k* - для виходів.

Хай нейромережа навчається вибіркою (X_a, Y_a), $a = 1..p$. Активність нейронів позначати малими літерами у з відповідними індексами, а малими літерами *x* - сумарні зважені входи нейронів.

Таблиця 2.2

Розрахунки при алгоритму зворотного розповсюдження помилки

Крок 1	Початкові значення ваги всіх нейронів всіх шарів *V (t = 0)* і *W (t = 0)* задаються випадковими числами.
Крок 2	Вхідне зображення *Xa* подається в мережу, в результаті формується вихідне зображення y^1 Y_a. Водночас нейрони послідовно функціонують по шарах за такими правилами: прихований шар: $x_j = \sum_i \quad W_{ij}X_i^a;\ \ y_j = f(x_j)$ (2.3) вихідний шар: $x_k = \sum_j \quad V_{jk}y_j;\ \ y_k = f(x_k)$ (2.4) Тут $f(x)$ - сигмоїдальна функція, визначена формулою 3.1.
Крок 3	Обраховується квадратична помилка мережі для поточного вхідного зображення: $E = \frac{1}{2\sum_k \ (y_k - Y_k^a)^2}$(2,5) Ця функціональність потребує мінімізації. Класичний метод градієнтної оптимізації ітеративно виконує уточнення аргументу: $V_{jk}(t+1) = V_{jk}(t) - h\frac{\partial E}{\partial V_{jk}}$(2,6) Функція помилок не залежить від ваги V_{jk} тому використаємо формулами неявного диференціювання комплексної функції: $\frac{\partial \mathrm{E}}{\partial \mathrm{y_k}} = \delta_\mathrm{k} = (\mathrm{y_k} - \mathrm{Y_k^a})$(2,7) $\frac{\partial \mathrm{E}}{\partial \mathrm{x_k}} = \frac{\partial \mathrm{E}}{\partial \mathrm{y_k}} \cdot \frac{\partial \mathrm{y_k}}{\partial \mathrm{x_k}} = \delta_\mathrm{k} \cdot \mathrm{y_k}(1 - \mathrm{y_k})$(2,8) $\frac{\partial \mathrm{E}}{\partial \mathrm{V_{jk}}} = \frac{\partial \mathrm{E}}{\partial \mathrm{y_k}} \cdot \frac{\partial \mathrm{y_k}}{\partial \mathrm{x_k}} \cdot \frac{\partial \mathrm{x_k}}{\partial \mathrm{V_{jk}}} = \delta_\mathrm{k} \cdot \mathrm{y_k}(1 - \mathrm{y_k}) * \mathrm{y_j}$(2,9) Тут враховано корисну властивість сигмоїдальної функції *f(x):* її похідна виражається тільки через значення самої функції: $f'(x) = f(1 - f)$(2.10) - Таким чином, отримані всі необхідні значення визначення ваг початкового шару *V.*

Продовження таблиці 2.2

Розрахунки при алгоритму зворотного розповсюдження помилки

Крок 4	На цьому етапі розраховуються ваги прихованих шарів. $W_{ij}(t+1) = W_{ij}(t) - h\frac{\partial E}{\partial W_{ij}}$(2.11) Розрахунки похідних виробляються за тими самими формулами, крім деякого ускладнення формули похибки *dj*. $\frac{\partial \mathrm{E}}{\partial \mathrm{x_k}} = \frac{\partial \mathrm{E}}{\partial \mathrm{y_k}} \cdot \frac{\partial \mathrm{y_k}}{\partial \mathrm{x_k}} = \delta_\mathrm{k} \cdot \mathrm{y_k}(1-\mathrm{y_k})$(2.12) $\frac{\partial \mathrm{E}}{\partial \mathrm{y_j}} = \delta_\mathrm{j} = \sum_\mathrm{k} \frac{\partial \mathrm{E}}{\partial \mathrm{x_k}} \cdot \frac{\partial \mathrm{x_k}}{\partial \mathrm{y_j}} = \sum_\mathrm{k} \delta_\mathrm{k} \cdot \mathrm{y_k}(1-\mathrm{y_k}) \cdot \mathrm{V_{jk}}$(2.13) $\frac{\partial \mathrm{E}}{\partial \mathrm{w_{ij}}} = \frac{\partial \mathrm{E}}{\partial \mathrm{y_j}} \cdot \frac{\partial \mathrm{y_j}}{\partial \mathrm{x_j}} \cdot \frac{\partial \mathrm{x_j}}{\partial \mathrm{w_{ij}}} = \delta_\mathrm{j} \cdot \mathrm{y_j}(1-\mathrm{y_k}) \cdot \mathrm{X_i^a} = \left[\sum_\mathrm{k} \delta_\mathrm{k} \cdot \mathrm{y_k}(1-\mathrm{y_k}) \cdot \mathrm{V_{jk}}\right] \cdot [\mathrm{y_j}(1-\mathrm{y_j}) \cdot \mathrm{X_j^a}]$(2.14) При розрахунку dj застосовано метод зворотного розповсюдження помилки, де приватні похідні обчислюються лише для змінних у наступному шарі. Ваги нейронів у прихованому шарі змінюються за відповідними формулами. У нейронній мережі з кількома прихованими шарами ця процедура застосовується послідовно до кожного з них, починаючи з шару, що передує входу, і закінчуючи шаром, який йде після входу. При цьому формули залишаються однаковими при заміні елементів вихідного шару на елементи відповідного прихованого шару.
Крок 5	Кроки 2-4 будуть повторюватися за всіма навчальними векторами. Процес припиняється після досягнення невеликої сумарної (або ж бажаної) помилки або обмежується максимально припустимою кількістю ітерацій, як у методі навчання Розенблатта.

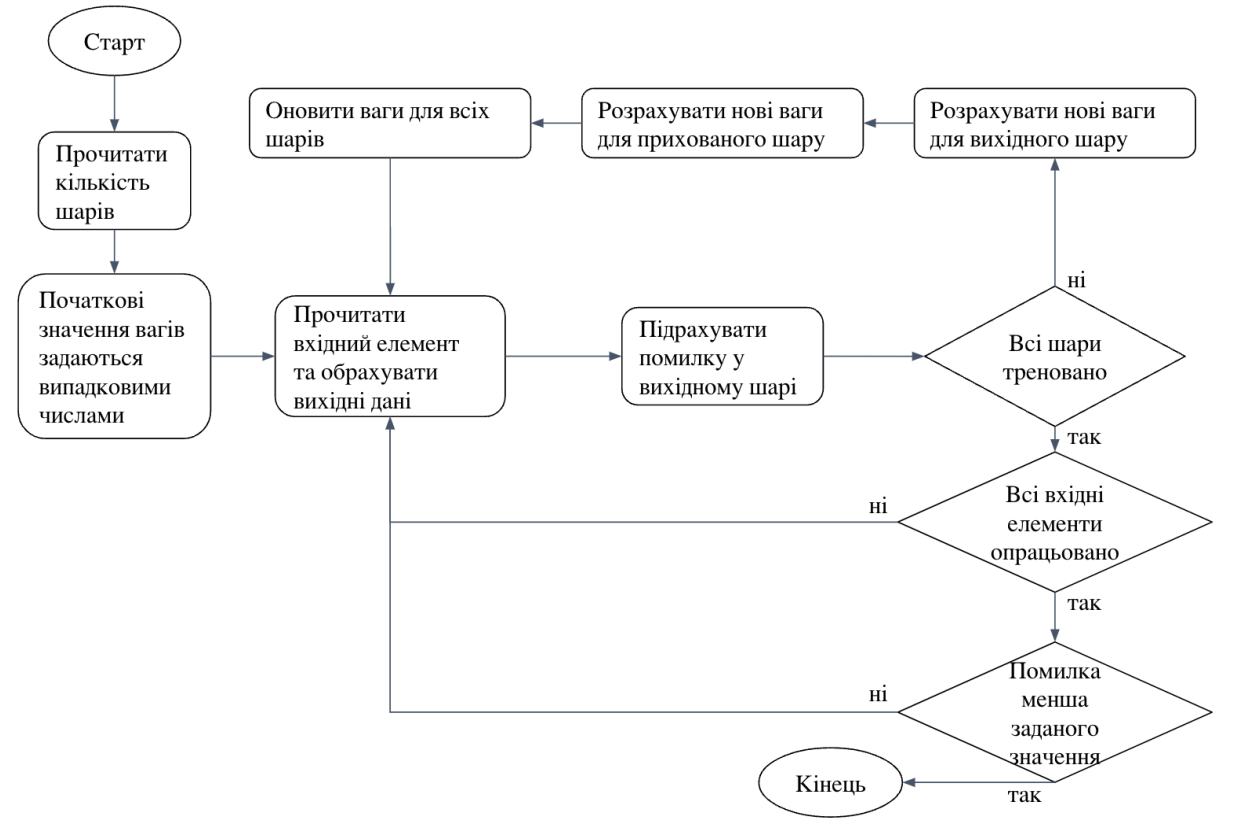


Рис. 2.6 Алгоритм зворотного розповсюдження помилки

Метод навчання передбачає оптимізацію функції помилки за допомогою градієнтного методу, що описано в кроках 3-4. Особливість зворотного поширення помилки полягає у можливості оцінки помилок для нейронів у прихованих шарах за допомогою зваженої суми помилок із наступного шару.

Змінна h означає швидкість навчання та вибирається достатньо малим для збіжності методу. З приводу конвергенції, важливо враховувати декілька моментів. По-перше, досвід показує, що метод зворотного поширення помилок виконується досить повільно. Повільна збіжність є загальною особливістю всіх градієнтних методів через те, що локальний напрямок градієнта не завжди співпадає з напрямком до мінімуму. По-друге, ваги обчислюються незалежно для кожної пари зображень у навчальній вибірці. У той же час, підвищення роботи на одній парі може погіршити роботи попередніх кадрів. У цьому значенні немає надійної гарантії конвергенції.

За даними досліджень, задля представлення довільного функціонального

відображення для навчальної вибірки достатньо лише двох шарів нейронів. Водночас практика показує, що для складних функцій можна зекономити кількість нейронів за рахунок використання більш ніж одного прихованого шару [16].

2.4 Протоколи, відеокодеки та формати передачі відео

Ключовим етапом у створенні системи відеозв'язку з розпізнаванням жестів на основі нейронних мереж є переднє оброблення всього набору даних, яке дозволить системі інтерпретувати отримані дані як відеоінформацію.

Так, відео може бути розглянуте як послідовність зображень, де кожен кадр відео є зображенням. Для аналізу відео за допомогою нейронних мереж зазвичай кожен кадр конвертується у відповідне зображення, яке подається на вхід моделі. Отже, переднє оброблення даних для розпізнавання жестів у відео може включати обробку кожного кадру як окремого зображення, де нейронна мережа аналізує кожне зображення для розпізнавання жестів чи іншої інформації. Тому для початку розділимо наш датасет по відповідних папках за допомогою bash-скрипта.

Оскільки відео мають бути представлені як зображення з відповідними словами жестів, кінцевий результат обробки попередніх даних потребує ієрархії папок, де в корені міститься ряд тек, назва кожної відповідає слову. Отже, перш за все, потрібно розділити наш набір даних за відповідні папки, для цього використали bash скрипт.

Наступним кроком буде вилучення фреймів із відеоряду кожного жесту. Кожне відео генерує 10 050 кадрів. Далі видаляється шум, тобто увесь фон і залишаються тільки контрастні руки щоб отримати більш релевантні функції із зображення. Отримане зображення перетворюється в чорно-білий формат, де зберігається лише інформація про його яскравість, щоб уникнути впливу властивостей кольору на навчання моделі. Вхідне зображення та результати його обробки відображені на рисунку 2.7.

Потокове мультимедіа - це медіа, яка постійно приймається користувачем від провайдера потокового мовлення. Ця концепція застосовується і до даних, що розповсюджується за допомогою телекомунікацій, так і до даних, що спочатку поширювалася потоком (телебачення, радіо) або ні (книги, відеокасети, аудіо компакт-диски) [17].

Рис. 2.7 Вхідне зображення та результат обробки фрейму

Під час потокової передачі відео використовуються методи стиснення та буферизації даних. Це дозволяє транслювати відео в реальному часі, передаючи дані у вигляді стиснутих пакетів. Основна перевага цього підходу полягає в тому, що користувач може почати перегляд відео без необхідності чекати завершення повного завантаження відеофайлу.

Потокова мультимедіа зазвичай здійснюється послідовно або ж в режимі реального часу.

При послідовному способі передачі відео відтворюється безпосередньо з жорсткого диска комп'ютера або сервера постачальника послуг. Зазвичай, цей метод забезпечує вищу якість зображення та звуку. Проте, його недолік полягає в тому, що перехід від одного моменту відео до іншого вимагає завантаження відповідного фрагменту. Це означає, що користувач повинен почекати

завантаження потрібної частини перед тим, як переглядати її. Для цієї трансляції використовується стандартний веб-сервер.

Режим реального часу вимагає використання потокового сервера і є найбільш підходящим для передачі тривалих відеофайлів. Користувач може обирати точку, з якої бажає почати перегляд. Цей метод потокової передачі мультимедіа широко застосовується для демонстрації захоплення екрану або трансляції відео з веб-камери.

Для досягнення поставленої в цій роботі мети найбільше підходить другий метод - потокове відео в реальному часі. Переважно цифрові відео призначені для зберігання та подальшого відтворення. Це витікає у дві основні вимоги: компактність та можливість відтворення на різних пристроях і платформах.

Більшість відео-файлів не націлені на потокову передачу. У потоковому відео відбувається поділ на невеликі фрагменти, які послідовно передаються та відтворюються по мірі їх отримання. Якщо відео потокове, воно надходить безпосередньо з камери. У протилежному випадку, воно завантажується з файлу.

Протокол потокового відео - нормалізований метод доставки, який дозволяє розбивати відео на частини, надсилати глядачам і знову збирати. Це просте пояснення: протоколи потокової передачі потенційно мають досить складні структури. Існує багато доступних протоколів потокової передачі з адаптивним бітрейтом. Ця технологія забезпечує найкращу якість, яку глядачі можуть підтримувати в будь-який час. Деякі протоколи мають на меті зменшити затримку або затримку між подією, що відбувається в реальному житті, та її відображенням на екрані глядача. Деякі протоколи працюють лише в певних системах. Інші зосереджені на керуванні цифровими правами (DRM).

Протокол дейтаграм користувача (UDP) є одним із протоколів у стеку TCP/IP. Він відрізняється від TCP тим, що працює без налагодження з'єднання. Він є одним із найпростіших протоколів на транспортному рівні моделі OSI, що дозволяє обмінюватися повідомленнями без необхідності підтвердження або гарантії доставки. Тобто він надсилає потік метаданих окремими маленькими пакетами. При використанні цього протоколу відповідальність за опрацювання

помилок та повторення передачі даних лежить на вищому рівні протоколу. Але незважаючи на всі свої недоліки, UDP все ще ефективний для серверів, які відправляють невеликі відповіді багатьом клієнтам.

Протокол потокової передачі у реальному часі (RTSP) - аналогічний протокол, призначений для передачі аудіо і відео. Це протокол програми, який описує команди управління відео-потоком. RTSP не виконує стиснення та не визначає методи упаковки мультимедійних даних або транспортних протоколів. Потокова передача даних як така перестала бути частиною протоколу даного протоколу. Більшість серверів RTSP використовують звичайний протокол передачі в реальному часі для трансляції аудіо та відео.

Протокол обміну повідомленнями у реальному часі (RTMP) - це спеціалізований протокол передачі потокових даних, основною метою якого є передача відео та аудіо з веб-камер через Інтернет. Компанія Adobe, розробник програвача Flash, створила RTMP з метою допомогти веб-серверам ефективно доставляти контент за запитом через мережу. Мінімізація затримки є ключовою для плавного перегляду відео у браузері. Порівняння затримок різних протоколів наведено для порівняння на рис. 2.8.

Сервери RTMP, такі як Flash Media Server, підтримують потокове відео в реальному часі та можуть передавати потокове аудіо й інші види даних. У випадку втрати з'єднання з Інтернетом під час перегляду контенту RTMP система може автоматично відновити зв'язок та продовжити передачу даних. Інтернет-користувачам сподобаються відео, які швидко запускаються й плавно відтворюються під час потокової передачі контенту за допомогою RTMP.

MPEG DASH - технологія що може забезпечити для HTTP динамічну адаптивну потокову передачу. Так само, як HDS і HLS, MPEG DASH є стандартом доставки відео, що використовується для прямого трансляційного відтворення в реальному часі через Інтернет. Метою цього стандарту є забезпечення найкращої якості контенту з мінімальною кількістю перерв та найменшою можливою затримкою буферизації.

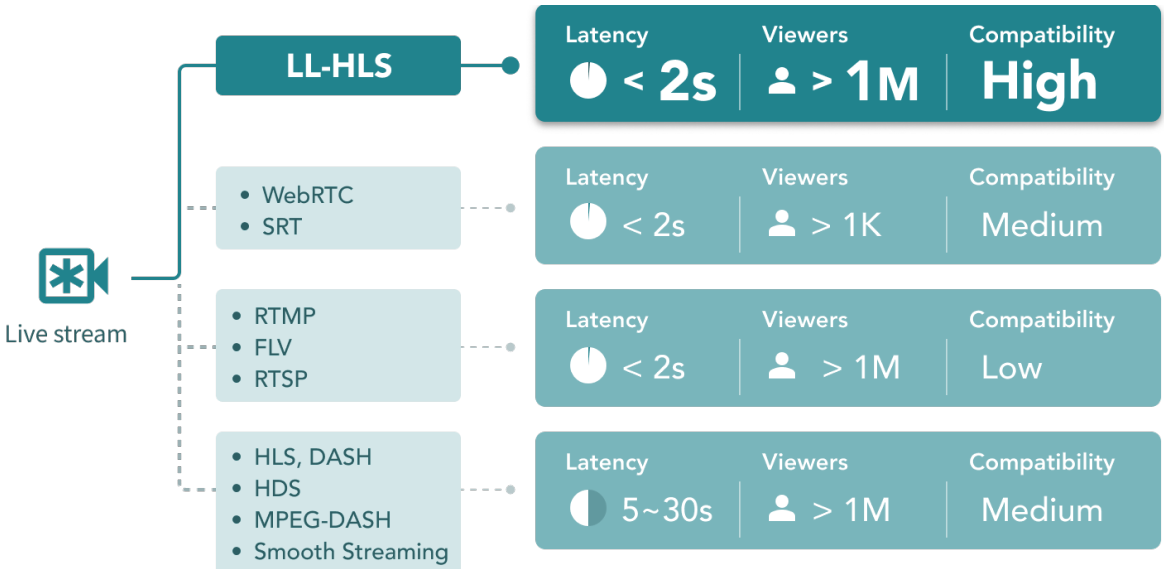


Рис. 2.8 Порівняння потокових затримок у різних протоколах

З технічної точки зору, MPEG DASH використовує подібну до інших протоколів потокового мультимедіа адаптивну систему передачі файлів.

Спочатку вихідний відео-файл розділяється на кілька частин, які буде скопійовано та закодовано з різними швидкостями передачі даних. Далі маніфест допомагає в ідентифікації різноманітних фрагментів файлів у запиті на фрагменти або діапазони байтів того самого файлу. Ці файли передаються через мережу й об'єднуються під час перегляду наживо.

Основні плюси MPEG DASH полягають, в тому, що в його основі - HTTP. Як наслідок - протоколи MPEG DASH можна доставляти через стандартні веб-сервери і це дає високу швидкість з малими витратами та налаштуванням.

Adaptive Bitrate Streaming (ABS) - це технологія управління продуктивністю потокової передачі мультимедіа через комп'ютерні мережі. Історично більшість методів відеопотокової передачі базувалися на RTP або RTSP, проте сьогодні більшість адаптивних потокових технологій призначено для використання HTTP через мережі з великою пропускною здатністю. ABS автоматично налаштовує якість потоку медіа в реальному часі відповідно до доступної ширини смуги у користувача, забезпечуючи плавне відтворення шляхом динамічного перемикання між різними бітрейтами та роздільною здатністю.

ABS працює шляхом динамічного моніторингу об’єму і пам'яті процесора, а потім вносить відповідні коригування якості відео. Цей процес лежить на кодуванні вихідного відео з різними бітрейтами, а потім сегментуванні кожного потоку з різними бітрейтами. Зазвичай тривалість сегмента становить від 2 до 10 секунд. Медіа-плеєр користувача може отримати переваги від ABS, так як він може автоматично переходити між різними сегментами бітрейту, обираючи ті сегменти, які найкраще відповідають пропускній здатності комп’ютера користувача та оптимізовує цим якість відтворення мультимедійного контенту.

Категорія сегмента та бітрейтів записані у маніфест-файл. Після отримання користувачем доступу до медіа-файлу, комп'ютер зчитує потокові сегменти з найменшою швидкістю передачі даних, зазначеним у маніфесті. Якщо відео-програвач користувача виявляє, що швидкість завантаження перевищує бітрейт початкового сегмента, він запитує наступний сегмент із вищим бітрейтом.

Цей процес триває до того моменту, доки точна відповідність між доступною пропускною здатністю та поточним сегментом бітрейту не буде знайдено. При зміні пропускної здатності користувача система автоматично запитує інший сегмент із відповідним бітрейтом. Результатом є мінімізація буферизації, пришвидшення ініціалізації відео та висока якість роботи як на високошвидкісних, так і на низькопропускних з’єднаннях.

Динамічна потокова передача HTTP (HTTP Dynamic Streaming, HDS) - це технологія потокової передачі відео та аудіо в мережі Інтернет за допомогою протоколу HTTP. HDS була розроблена компанією Adobe Systems та використовується для відтворення мультимедійного контенту у високій якості на веб-платформах.

Основна ідея HDS полягає в тому, що відеофайли розділяються на маленькі фрагменти, а потім ці фрагменти передаються через HTTP клієнту. Однак, відмінністю HDS від інших методів є його можливість адаптувати якість відео на льоту, швидко переключаючи різні роздільні здатності в залежності від швидкості Інтернет-з'єднання користувача. Це дозволяє підтримувати плавний перегляд відео навіть при зміні умов мережі.

HDS була популярною технологією відтворення мультимедіа веб-контенту, проте у зв'язку зі змінами у сфері технологій і розвитком інших методів потокової передачі, Adobe припинила підтримку HDS на користь інших рішень, таких як MPEG-DASH або HLS.

Microsoft Smooth Streaming (MSS) - це технологія потокової передачі мультимедійного контенту через HTTP. Вона адаптує якість відеопотоку в реальному часі в залежності від швидкості передачі даних у глядача та можливостей пристрою. Плавна потокова передача підтримує адаптивні бітрейти та включає деякі потужні інструменти DRM. Однак варто зазначити що дана технологія не застосовується широко, окрім екосистеми Microsoft.

HTTP Live Streaming (HLS) - це протокол потокової передачі мультимедійного контенту через мережу Інтернет. Розроблений компанією Apple, HLS використовує протокол HTTP для передачі відео, аудіо та інших медіа-даних до веб-плеєрів на різних пристроях, таких як смартфони, планшети та комп'ютери.

Основна ідея HLS полягає в тому, що вихідне відео розбивається на невеликі фрагменти, які потім передаються через HTTP сервер. Один і той же контент може бути наданий у різних роздільних здатностях, що дозволяє пристосовувати якість відео до швидкості і якості з'єднання користувача. Це робить HLS досить універсальним та гнучким під різні умови мережі та різні пристрої.

Однією з важливих переваг HLS є його сумісність з різними пристроями та платформами, оскільки більшість сучасних веб-плеєрів та пристроїв підтримують цей протокол. HLS також має підтримку кешування, що полегшує швидке завантаження контенту та його перегляд.

HLS став популярним рішенням для відтворення відео онлайн на різних платформах, і він залишається одним із стандартів для потокової передачі мультимедійного контенту через Інтернет. Хоча він і має декілька недоліків, серед яких затримки у відтворенні, підвищена витрата трафіку, оптимізація першочергово для Apple пристроїв та залежність від кешування.

WebRTC - це комбінація протоколів, стандартів та API, яка забезпечує спілкування в режимі реального часу. Користувачі, які підключаються через Chrome, Firefox або Safari, можуть спілкуватися безпосередньо через свій веб-браузер, що забезпечує затримку прийому менше 500 мілісекунд. Це проєкт із відкритим вихідним кодом, призначений для підтримки прямої потокової передачі між програмами що підтримують WebRTC та/ або браузерами.

Термін «кодек» описує технології зжаття відео. Різні види кодеків використовуються для різних завдань: наприклад, Apple ProRes застосовується для редагування відео, а H.264 є найпоширенішим відеокодеком, який широко використовується для онлайн-відео. Формат відноситься до формату контейнера відео-файлу.

Різноманітні формати контейнерів, такі як .mp4, .m4v та .avi, функціонують як структури, що упаковують різні потоки відео, аудіо та інші дані в одному файлі. Контейнер визначає, як ці дані організовані та зберігаються. Важливо враховувати, що різні протоколи передачі можуть підтримувати різні кодеки - це спеціальні алгоритми стиснення, які використовуються для зменшення розміру відео та аудіо файлів. Обираючи контейнер, слід враховувати, які кодеки можуть бути підтримані в конкретній системі передачі даних.

Кодек - є способом стиснення відео-файлів. Необроблений відео файл містить багато фотографій, які відтворюються у швидкій послідовності (зазвичай 30 кадрів на секунду), 32-мегапіксельні фотографії займають багато місця. Для вирішення цієї проблеми використовується стиснення, яке використовує математичні алгоритми для інтелектуального видалення неважливих даних. Наприклад, якщо один кут відео чорний і залишається таким протягом кількох секунд, ви можете виключити дані окремих пікселів і просто ввімкнути посилання. Стандарти стиснення існують окремо від потокового передавання. Непотокове медіа також використовує стиснення, однак деякі протоколи потокової передачі підтримують лише певні кодеки.

Відео можна уявити як тривимірний масив пікселів, де два виміри відображають вертикальне та горизонтальне розширення кадру, а третій вимір -

це час. Кожен кадр представляє собою колекцію пікселів, видимих камерою протягом конкретного часового інтервалу.

Неможливо уявити стистеснення, якби кожне зображення було унікальним, а розташування пікселів на зображенні було було абсолютно випадковими. Тому спочатку можна стиснути саме зображення, наприклад, фотографія тихого моря без деталей фактично зводиться до зображення граничних точок і градієнтів заливки. Можна стискати подібні суміжні зображення. Стиснення відео схоже на стиснення зображень. Алгоритми обробки відео спираються на ту ж концепцію стиснення зображень, розглядаючи відео як тривимірне зображення.

Окрім стиснення з втратами, відео також можна стиснути без втрат. Це означає, що після відкриття результат точно збігатиметься з оригіналом (біт за бітом). Однак за допомогою стиснення без втрат неможливо досягти високого рівня стиснення реального відео. З цієї причини практично все широко розповсюджене відео стискається з втратами (зокрема на споживчих цифрових відеодисках, відео-архівах). Веб-сайти іноді використовують прості формати GIF і APNG для невеликих відео без звуку.

Однією з найпотужніших підходів, що забезпечує збільшення рівня стиснення, є компенсація артефактів - зокрема руху. Зараз використовується схожість фрагментів з попередніми кадрами для підвищення ступеня стиснення. Незважаючи на це використання схожостей сусідніх кадрів є неповним через рух об’єктів або самої камери. Підхід рухової компенсації полегшує знаходження схожих областей незважаючи на наявність дефектів руху.

Ще одним можливим джерелом непорозумінь є транспортний формат. Це визначає форму, у якій відео передається, у т.зв. "контейнерах" або "пакетах". Такий формат зазвичай містить вміст, який включає стисле відео, аудіо та метадані. Ці дані передаються через протокол потоку, який відповідає за організацію даних під час передачі. Прикладами таких форматів є MP4 (фрагменти) і MPEG-TS [19].

3 ВПРОВАДЖЕННЯ ТА ДОСЛІДЖЕННЯ МОДЕЛІ КЛІЄНТ-СЕРВЕРНОЇ ВЗАЄМОДІЇ З ВЕБ-ДОДАТКАМИ НА ОСНОВІ ГОЛОСОВОГО І ЖЕСТОВОГО УПРАВЛІННЯ

3.1 Вибір інструментів програмування

Оскільки алгоритмом роботи запропонованої системи передбачено використання нейромережі, було вирішено використати програмний пакет машинного навчання з відкритим вихідним кодом Tensor Flow, що розроблений на мовах C++ та Python і використовується для експериментів та розрахунків в галузі ML, для проєктування та навчання нейронних мереж для автоматичного пошуку та класифікації зображень [20]. Tensor Flow містить пакет модулів під машинне навчання під назвою Tensor Flow Hub (рис. 3.1). Модулі у цьому контексті є частинами графа нейронної мережі, які разом з активами та вагами можуть бути повторно використані для завдань трансферного навчання.

У цій роботі для отримання просторових характеристик зображення з відеокадрів використовується Tensor Flow Hub. Модуль побудований за допомогою попередньо нетренованої моделі на базі архітектури Inception V3. Вона навчалася на наборі даних Image Net, що складається із мільйонів зображень. При цьому бібліотека TFlearn використовувалася для реалізації PHM, потрібної для визначення тимчасових ознак. Ця бібліотека побудована на основі Tensor Flow для глибокого навчання. Він був розроблений для представлення API високого рівня для спрощення та прискорення досліджень, залишаючись при цьому повністю прозорим та сумісним із ним.

Для розробки основних частин моделі обрано Python, який має наступні переваги для вирішення завдань роботи:

- підтримка - Python пропонує широкий спектр стандартних бібліотек: Інтернет, строкові операції, інструменти веб-сервісів, протоколи операційних систем, тощо. Найпоширеніші завдання програмування вже

записані в бібліотеках, що скорочує кількість коду, який ви пишете на Python;

- підвищена продуктивність роботи – Python має широку підтримку бібліотек та чистий об'єктно-орієнтований дизайн, що підвищує продуктивність роботи програміста на 100 - 400 відсотків порівняно з C, C#, C++ , Java;

- продуктивність - потужні інтеграції, системи тестування пристроїв і розширені функції управління допомагають підвищити швидкість і продуктивність роботи більшості додатків. Це відмінний варіант для створення багатопротокольних мережних додатків, що масштабуються.

Тому Python можна вважати легкою у використанні та надійною мовою програмування. Він використовується для big data, штучного інтелекту в цілому та ML зокрема по всьому світу - що і потрібно для цього проєкту.

Розробка на Python буде вестись на середовищі розробки VSCode, а залежності встановлюватимуться за допомогою менеджера пакетів pip.

Графічний інтерфейс користувача було вирішено реалізувати на мобільній платформі Android через те, що ця ОС в даний час широко використовується в гаджетах по всьому світу які доступні великій кількості людей. Станом на третій квартал 2023 року налічувалося понад 2,8 млрд пристроїв на Android по всьому світі [21]. Також варто відзначити, що система відео-зв'язку, що розробляється, вимагає наявності камери достатньої роздільної здатності, що цілком відповідає поточному стану ринку мобільних девайсів у світі та нівелює витрати на додаткове обладнання.

Для розробки мобільного додатка було обрано нативний набір інструментів Android SDK і об'єктно-орієнтована мова програмування Java. Такий вибір має бути обґрунтований такими факторами:

- Java - кроссплатформенна мова програмування і, як наслідок, вже скомпільовані програми виконуються на всіх операційних системах;

- високорівневість - завдяки абстракціям розробникам Java не потрібно вручну керувати пам'яттю, тощо;

- безпека - Java менеджить безпеку для кожного додатку через створення політики безпеки з певним набором правил доступу;
- багатопоточність - Java дозволяє запускати потоки одночасно, що дозволяє ефективно використовувати процесорний час;
- широка екосистема перевірених бібліотек та фрейм-ворків, що дозволяє виконання безлічі завдань;
- спільнота та стабільність - багаторічна розробка Java забезпечується спільнотою, підтримкою Oracle та різноманітністю програм на JVM;
- ця мова є високорівневою та сучасною, фахівці, які володіють мовою Java, затребувані на ринку праці на момент написання дисертації.

Як середовище розробки та системи збирання мобільних додатків використовувалися Android Studio та Gradle, рекомендовані документацією [22].

Отже, обрані засоби розробки інтерфейсу користувача повністю задовольняють поставленим цілям, деталі реалізації кожного компонента ми розглянемо надалі.

3.2 Деталі реалізації компонентів програми

Наш світ з кожним днем стає мобільнішим, у світі налічується понад 3,3 мільярди смартфонів. Таким чином, мобільна розробка має потенціал охопити всі куточки та аспекти сучасного світу. Це однаково справедливо і щодо машинного навчання. Створення моделей машинного навчання, які ми можемо використовувати на мобільних пристроях, відкриє безмежні можливості для творчості, автоматизації та ефективності. Можливість запуску заздалегідь підготовлених моделей на мобільних пристроях є важливим зрушенням у розвитку обчислювальної техніки. Завдяки можливості обробляти дані безпосередньо з телефону користувача, особисті дані залишаються в його руках, програми працюють плавніше, не чекаючи стільникових мереж та необхідності дорогих послуг хмарного зберігання.

Але між мобільною розробкою та машинним навчанням існує значний розрив. Справа в тому, що методи машинного навчання вимагають для своїх обчислень значних апаратних ресурсів та пам'яті, яких зазвичай не вистачає навіть на найпотужніших смартфонах. Саме тому інтеграція нейромереж на мобільних пристроях потребує додаткових маніпуляцій.

Для перетворення моделі, підготовленої та навченої у розділі 2, ми будемо використовувати пакет інструментів TensorFlow Lite [23]. TensorFlow Lite призначений для ефективного запуску моделей на мобільних та інших вбудованих пристроях з обмеженими обчислювальними ресурсами та ресурсами пам'яті. Частково ця ефективність обумовлена використанням спеціального формату зберігання моделей. Моделі TensorFlow необхідно перетворити на цей формат, перш ніж їх можна буде використовувати в TensorFlow Lite.

Перетворення моделей може не впливаючи на точність зменшує розмір файлів чим оптимізує використання пам’яті. Перетворювач TensorFlow Lite надає можливості для подальшого зменшення розміру файлу та збільшення швидкості виконання. Щоб конвертувати нашу модель ми використовуватимемо код, показаний на рис. 3.2.

convert.py

```
1  import tensorflow as tensorflow
2
3  modelConverter = tensorflow.lite.TFLiteConverter.from_saved_model(initial_model)
4  converted_model = modelConverter.convert()
5  open("converted_model.tflite", "wb").write(converted_model)
6
```

Рис. 3.2 Перетворення моделі на мобільному пристрої

На певних пристроях можна використати апаратне прискорення для машинного навчання. Наприклад, більшість мобільних телефонів оснащені графічними процесорами або графічними процесорами, які мають змогу робити матричні операції з дробовими типами даних аніж центральні процесори. Тут TensorFlow Lite як інтерпретатор може бути налаштованим спеціальними делегатами, що дозволяє використовувати апаратне прискорення самого

пристрою. Делегат GPU дозволяє інтерпретатору виконувати відповідні операції на графічному процесорі пристрою. Ми користуємося цією можливістю, підключивши делегата GPU.

Наступним кроком необхідно підключаємо потрібні залежності. Програма використовує попередньо створений Android-архів TFLite.

Конфігуруємо додаток так, щоб він не стикав .tflite (рис. 3.3). Завдяки цьому файл буде доступний в оперативній пам’яті (що не працює при стисненні файлу)_

Для реалізації відеозв'язку ми використовуватимемо Chat API Mesibo - відкриту бібліотеку для Android, яка інкапсулює функціонал відео-дзвінків, а також надає готовий сервер для надсилання та отримання запитів. Результат відео-чату показано рисунку 3.4.

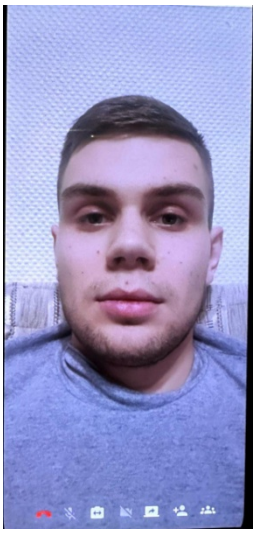


Рис. 3.4 Екран відео-зв'язку

3.3 Перевірка параметрів нейронної мережі

Завданнями цього розділу стануть:

- вибір ознак нейромережі
- валідація параметрів

Нам потрібно верифікувати отримані на навчальній вибірці дані. Ми будемо використовувати метод перехресної перевірки, який полягає у використанні під час ітеративного навчання додаткового набору даних – набору даних перевірки. Розділяємо вибірку у відношенні 75 на 35 – навчання/перевірка (рис. 3.5). Необхідно запустити навчання на наборі навчальних даних та верифікувати правильність підібраних параметрів на наборі перевірочних даних. Якщо похибка знаходиться в допустимих межах, вважатимемо обрані параметри оптимальними.

split_dataset.py

```
1  if iTrain:
2      x_train, x_test, y_train, y_test = dataset_split(X, Y, size=0.25, state=60)
3      return x_train, x_test, y_train, y_test
4  else
5      return X, Y
6
```

Рис. 3.5 Поділ набору даних навчання та перевірки

Кожного циклу вираховується похибка, пов'язана з новим значенням ваги та зсувів. Чим довше застосовується алгоритм навчання, то менше буде помилка в навчальному наборі. Фактично, з часом ми згенеруємо значення ваг і зміщень так, щоб помилка на навчальному наборі буде практично нульова, що з великою ймовірністю спричиняє перенавчання моделей.

Але коли поточні значення ваг і зсувів застосовуються до набору даних перевірки, на певний момент помилка, ймовірно, починає збільшуватися. Як бачимо з графіку помилки навчання графік помилки під час навчання та валідації на рис. 3.6, помилка валідації починає збільшуватися після 9 епохи. Це означає, що навчання повинне бути зупинене, а значення ваги використане для поточної епохи.

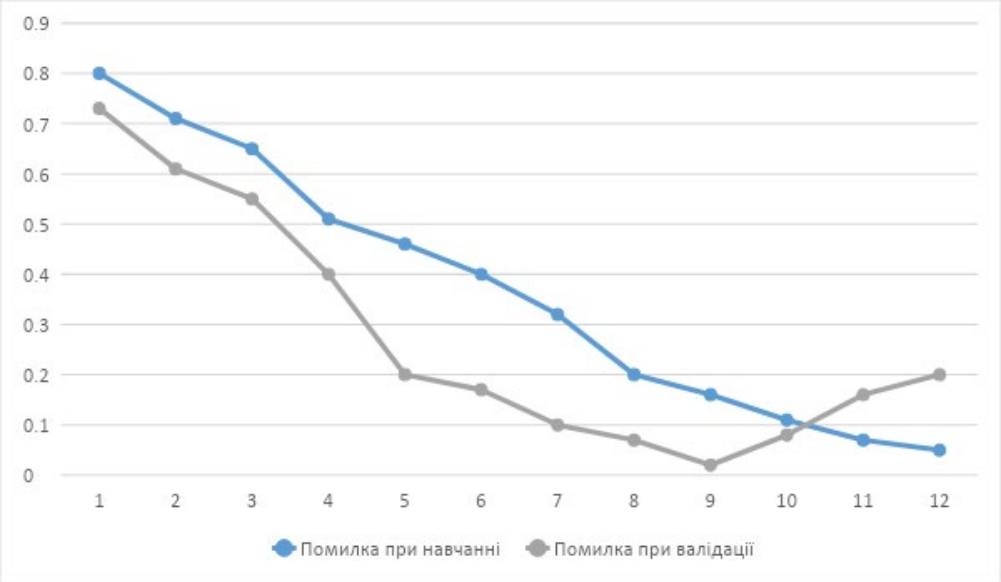


Рис. 3.6 Помилка навчання та валідації

3.4 Тестування програми: оцінка функції витрат та оцінка точності

Результат роботи нейронної мережі зазвичай називають гіпотезою. Його позначають h(X) - гіпотеза від вхідних параметрів. Зрештою ми прагнемо щоб гіпотези максимально відповідали реальності (реальні класи об'єктів, у цій задачі класи жестів). Тепер нам потрібна міра, яка описує якість нейронної мережі. Цю міру називають «функцією втрат» [5] та позначають J(W), показуючи її залежність від коефіцієнтів вагової матриці. Чим менше функціонал, тим рідше наша нейромережа припускається помилок і тим вона краща. Навчання зводиться до мінімізації цього функціоналу. Залежно від коефіцієнтів вагової матриці, нейронна мережа може мати різну точність. Процес навчання є рухом по гіперповерхні функціоналу втрат, мета якого - мінімізувати цей функціонал.

Отже, поставимо гіперпараметри і запустимо нейронну мережу на навчання на 12 епох. Максимальний розмір пакета для двошарової моделі з 256 нейронами в кожній становив 32. Це досить мало в порівнянні з 10 000 екземплярами даних, що робило процес навчання такої великої моделі досить повільним.

Під час навчання деталі виводилися на консоль (таблиця 3.1), а модель зберігалася як точка відліку щоразу, коли функція втрат зменшувала своє значення. На кожному етапі модель навчалася із використанням однієї партії.

Процес навчання припинявся, коли функція втрат оціночному наборі переставала зменшуватися. На рис. 3.8 дає приклад того, як функція вартості зменшується з часом. По осі X відкладено кількість епох, а по осі Y значення функції втрат. Модель навчалася на 12 епох по 30 хвилин. Виявилося, що для навчання такої великої моделі потрібно дуже мало часу. Навчання займає так багато часу через великі вхідні послідовності, велику кількість діалогів і залучення принципу уваги, що ще більше ускладнює та ускладнює навчання цієї моделі. Ця модель мала 2 шари, кожен із яких містив 256 нейронів.

Таблиця 3.1

Логи проведеного тренування

Training Step	Time (s)	Epoch	Loss	Acc	Val Loss	Val Acc	Iter
1	21.304	1	0.00000	0.0000	-	-	032/120
2	23.235	1	0.00608	0.1800	-	-	064/120
3	25.249	1	0.03848	0.3192	-	-	096/120
4	31.637	1	0.03659	0.4370	0.01450	1.2000	120/120
5	2.644	2	0.03492	0.5326	-	-	032/120
6	5.027	2	0.03422	0.6105	-	-	064/120
7	7.249	2	0.03332	0.6743	-	-	096/120
8	10.638	2	0.03069	0.7266	0.01074	1.2000	120/120
...	...	...	...	...	...	...	...
33	2.312	9	0.02610	0.9900	-	-	032/120
34	4.357	9	0.02383	0.9911	-	-	064/120
35	6.455	9	0.02229	0.9921	-	-	096/120
36	9.671	9	0.02068	0.9929	0.00343	1.2000	120/120
37	2.539	10	0.01936	0.9937	-	-	032/120
38	4.855	10	0.03419	0.9911	-	-	064/120
39	6.770	10	0.03135	0.9920	-	-	096/120
40	9.778	10	0.02924	0.9928	0.00326	1.2000	120/120

Рис. 3.7 Графік функції втрат

Точність - співвідношення вірних припущень до сумарної кількості, (4.1)

$$Accuracy = \frac{correct}{total}, \qquad (4.1)$$

де correct - кількість правильних припущень;

total - сумарна кількість припущень.

ЇЇ оцінка може бути виконана за допомогою перехресної перевірки.

Перехресна перевірка - це процедура перевірки точності оцінки даних із тестового набору, який також називається набором перехресної перевірки. Точність оцінки тестового набору порівнюється з точністю оцінки навчального набору. Якщо оцінка тестового набору дає приблизно ті самі результати точності, що і класифікація навчального набору, ми приймаємо цей результат як проходження перехресної перевірки.

Поділ на навчальну та тестову вибірки здійснюється шляхом поділу набору даних у певній пропорції, наприклад, дві третини навчальних даних та одна третина тестових даних. Цей метод можна використовувати для розділення наборів даних із великою кількістю прикладів. Якщо розмір вибірки невеликий,

рекомендується використовувати спеціальні методи, під час яких навчальна і тестова вибірки можуть частково перекриватися.

У нашому випадку досить було розділити набір даних у співвідношенні 75 до 25, в результаті за 10 епох було отримано точність, показану на малюнку 3.9.

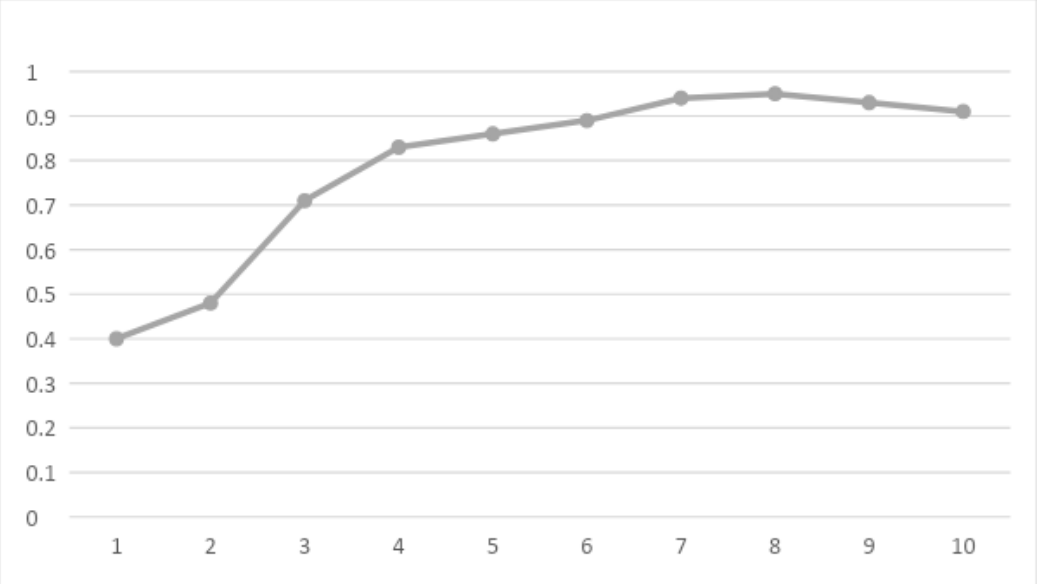


Рис. 3.8 Графік точності моделі

У цьому останньому розділі нейронну мережу було навчено на наборі відеоданих жестів, її параметри були перевірені, а тестування було виконано на наборі перевірочних даних.

В результаті було визнано необхідним переглянути спочатку обрані параметри для навчання на наборі даних через недостатню точність отриманих результатів. У результаті вихідний набір даних, у якому навчалася нейромережа, збільшився вдвічі. Це призвело до зміни інших параметрів, таких як кількість епох і кількість пакетів даних, і, у свою чергу, збільшення часу навчання в кілька разів.

Також виявлено ітераційне зменшення функції втрат як основної характеристики відповідності прогнозованих даних реальним зі збільшенням числа епох. Це вказує на правильно вибрані гіперпараметри нейронної мережі.

3.5 Репрезентація розробленої моделі клієнт-серверної взаємодії

В моделі за основу було взято архітектуру товстого клієнта. Клієнт відповідальний за отримання відеоданих з пристрою користувача, стрімінг на сервер, опрацювання вхідного зображення та інтерпретацію жестів та використання даних для потреб клієнтського додатку з відправленням опрацьованих даних на сервер (рис. 3.9).

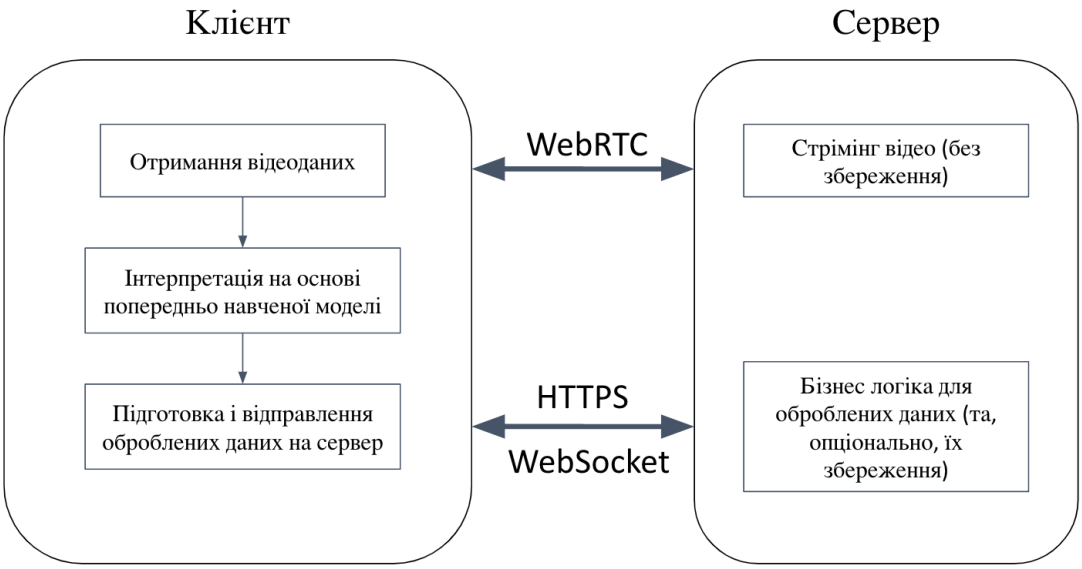


Рис. 3.9 Схема моделі клієнт серверної взаємодії

Сервер при такій моделі не бере жодної участі в інтерпретації відео чи аудіоданих. Головна його мета полягає в ретрансляції стрімінгу відео.

Опціонально він також може імплементувати бізнес логіку яка працює з “розшифрованими жестами”. Для прикладу збереження історії “розмови” в текстовому форматі або ж виконання команд.

Було обрано декілька протоколів передачі даних:

- WebRTC - для стрімінгу відео з найменшою затримкою
- HTTPS - для безпечної комунікації з сервером
- WebSocket - для оновлення даних на клієнті в режимі реального часу, якщо вони помінялися на сервері.

Так як логіка інтерпретації відбувається на клієнті - завдяки цьому ми отримуємо головні переваги і недоліки моделі.

Обробка даних на клієнті. Це являється як перевагою і недоліком. Завдяки цьому ми не залежимо від з’єднання з мережею, зменшуємо навантаження на сервер, отримуємо кращу швидкодію. Водночас такий підхід може вимагати підвищеної кількості ресурсів на клієнті.

Приватність та безпека. Важлива частина обробки відбувається локально, що дозволяє зберігати конфіденційні дані на пристрої користувача.

ВИСНОВКИ

За підсумками магістерської роботи було розглянуто варіанти клієнт-серверної архітектури, протоколи передачі даних, використання штучних нейронних мереж у завданнях розпізнавання та класифікації жестів із відеопослідовностей. Розроблено наукову літературу зі штучних нейронних мереж та методів їх навчання та оптимізації, методів передачі та обробки відеоданих, методів інтеграції нейронних мереж на мобільних пристроях, у тому числі іноземною мовою.

Було досліджено наявні методи вирішення задачі розпізнавання жестів та після їх аналізу обрано ті, які вирішують поточні задачі. Було підібрано набір даних розміром 60 знакових слів для навчання нейромережі.

У цій роботі вибрано дві моделі нейронних мереж - CNN і RNN, які разом дозволяють розпізнавати відео як просторові, так і тимчасові особливості. Експериментальні дослідження показали, що модель може класифікувати прості жести з точністю 92%.

На основі навченої нейромережі було розроблено модель, що має навчальний модуль та адаптацію нейромережі для мобільних пристроїв у презентації відео-месенджера з можливістю обробки жестової мови ASL. За логіку розпізнавання відповідає бібліотека Tensor Flow. Графічний інтерфейс та інтерфейс передачі даних був виконаний за допомогою Android SDK та фрейм-ворку Mesibo API відповідно.

Розроблена модель клієнт-серверної взаємодії для додатка з розпізнаванням жестової мови. За основу було взято архітектуру товстого клієнта, тобто логіка розпізнавання були винесена на клієнт. Для клієнт-серверної комунікації було використано протоколи: WebRTC, HTTPS, WebSocket.

Дана модель дозволяє використати навчену нейромережу з підвищеною точністю та є більш доступною за рахунок відсутності вимог до додаткового спеціального обладнання

ПЕРЕЛІК ПОСИЛАНЬ

1. SignalAll - Інформація про продукт. [Електронний ресурс] - Режим доступу: https://www.signall.us/product_info/

2. Звіт Smart Audio. [Електронний ресурс] – Режим доступу: https://www.nationalpublicmedia.com/uploads/2019/10/ The_Smart_Audio_Report_Spring_2019.pdf

3. Масуд С., Шрівастава А., Тувал Х.К., Ахмад М. (2018) Розпізнавання жестів мови жестів (слів) у реальному часі з відеопослідовностей з використанням CNN та RNN. В: Бхатеджа Ст, Коелло Коельо К., Сатапаті С., Паттнаїк П. (ред.) Інтелектуальна інженерна інформатика. Досягнення в галузі інтелектуальних систем та обчислень, тому 695. Спрінгер, Сінгапур.

4. Ольга Русаковська*, Цзя Ден*, Хао Су, Джонатан Краузе, Санджів Сатіш, Шон Ма, Чжихен Хуанг, Андрій Карпаті, Адітья Хосла, Майкл Бернштейн, Олександр К. Берг та Лі Фей-Фей. (* = рівний вклад) Масштабний конкурс візуального розпізнавання ImageNet. ЦЖКВ, 2015.

5. Салман Хан, Хосейн Рахмані, Сайєд Афак Алі Шах та Мохаммед Беннамун. Посібник із згорткових нейронних мереж для комп'ютерного зору. / Узагальнюючі лекції з комп'ютерного зору, 2018 №15

6. К. Сегеді, Ст Ванхук, С. Іоффе, Я. Шленс. Переосмислення початкової архітектури комп'ютерного зору // Препринт arXiv: 1512.00567. – 2015.

7. Круз, Холк (2006). Нейронні мережі як кібернетичні системи (2-ге та виправлене видання). 615. Мозок, розум і медіа, Білефельд, Німеччина

8. Кінгма, Дідерік та Джиммі Ба. Адам: Метод стохастичної оптимізації. // Препринт arXiv: 1412.6980. – 2014.

9. CS231n: нейронні згорткові мережі для візуального розпізнавання [Електронний ресурс] - Режим доступу: https://cs231n.github.io/convolutional-networks/

10. Як працює відстуеження рук [Електронний ресурс] – Режим доступу до ресурсу: https://www.ultraleap.com/company/news/blog/how-hand-tracking-works/.

11. Кордос М., Бялка С., Блахник М. Вибір екземпляра при добуванні логічних правил для завдань регресії // Міжнародна конференція з штучного інтелекту та м'яких обчислень. – Шпрінгер, Берлін, Гейдельберг, 2013. – С. 167-175.

12. Ронкетті, Франко та Кірога, Факундо та Естребу, Сезар та Ланзаріні, Лаура та Розете, Алехандро. LSA64: Набір даних аргентинської мови жестів. / XX II Аргентинський конгрес наук з обчислювальної техніки (CACIC) – 2016 р.

13. Хінтон Г. Нейронні мережі для розпізнавання образів/Х. Джеффрі. – ОХ: Видавництво Оксфордського університету, 1995. – 144 с.

14. Гудфеллоу І., Бенджіо Ю., Курвіль А. Глибоке навчання (серія «Адаптивні обчислення та машинне навчання») / С. Найт. - Великобританія: Springer, 2009. - 287 с.

15. Дж. Лі. Масштабовані системи безперервної потокової передачі мультимедіа: архітектура, проєктування, аналіз та реалізація. Джон Вайлі та сини, 2005. с. 25.

16. Який протокол потокового відео слід використовувати [Електронний ресурс] - Режим доступу: https://www.dacast.com/blog/video-streaming-protocol/

17. Протоколи потокової передачі: все, що вам потрібно знати. [Електронний ресурс] - Режим доступу: https://www.wowza.com/blog/streaming-protocols

18. Документація TensorFlow API [Електронний ресурс] – Режим доступу: https://www.tensorflow.org/api_docs/

19. Скільки є зараз Android користувачів. [Електронний ресурс] - Режим доступу: https://www.bankmycell.com/blog/how-many-android-users-are-there

20. Створіть проєкт Android. [Електронний ресурс] - Режим доступу: https://developer.android.com/training/basics/firstapp/creating-project

21. Почніть роботу із TensorFlow Lite. [Електронний ресурс] - Режим доступу: https://www.tensorflow.org/lite/guide/get_started

22. Б. Канг, С. Тріпаті, Т.К. Нгуєн. Розпізнавання відбитків пальців мовою жестів у реальному часі з використанням згорткових нейронних мереж на

карті глибини. [Електронний ресурс] - Режим доступу: https://arxiv.org/abs/1509.03001/

23. Пігу Л., Ділеман С., Кіндерманс П.-Дж., Шраувен Б.: Розпізнавання мови жестів з використанням згорткових нейронних мереж [Електронний ресурс]. – 2015. – Режим доступу:https://ieeexplore.ieee.org/document/5204291/

24. С. Лівицький, М. Еверінгем. Автоматичне розпізнавання слів, написаних відбитками пальців британською мовою жестів [Електронний ресурс]. 2009. - Режим доступу: https://ieeexplore.ieee.org/document/5204291/ .

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

МАГІСТЕРСЬКА РОБОТА

«РОЗРОБКА МОДЕЛІ КЛІЄНТ-СЕРВЕРНОЇ ВЗАЄМОДІЇ З ВЕБ-ДОДАТКАМИ НА БАЗІ ГОЛОСОВОГО ТА ЖЕСТОВОГО КОНТРОЛЮ З ВИКОРИСТАННЯМ FULL-STACK ТЕХНОЛОГІЙ»

Виконав: студент групи ПДМ-62, Максим'юк Богдан Богданович

Керівник: д.т.н., професор, професор кафедри ІПЗ, Бондарчук А.П.

Київ - 2023

МЕТА, ОБ'ЄКТ, ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: покращення точності та доступності взаємодії користувачів з веб-додатками на базі жестового контролю

Об'єкт дослідження: процес взаємодії клієнта з сервером та розпізнавання жестів

Предмет дослідження: протоколи передачі даних між клієнтом та сервером, моделі та алгоритми інтерпретації відеоданих

АНАЛОГИ

Особливість	SignAll	Leap Motion
Мови що підтримуються	ASL	ASL
Технології	Комп'ютерне зорове сприйняття, Машинне навчання	Комп'ютерне зорове сприйняття
Відстеження рук	Так	Так
Розпізнавання жестів	Так, повний спектр жестів ASL	Так, повний спектр жестів ASL
Розпізнавання емоцій	Ні	Так
Точність	Висока (90%)	Висока (90%)
Технічні вимоги	Спеціальна камера та рукавички	Спеціальна камера
Вартість	$10,000+	$200+
Додатки	Інтерпретація мови жестів, Доступність, Освіта	Дизайн, Медичні дослідження, Розваги

СХЕМА МОДЕЛІ КЛІЄНТ-СЕРВЕРНОЇ ВЗАЄМОДІЇ

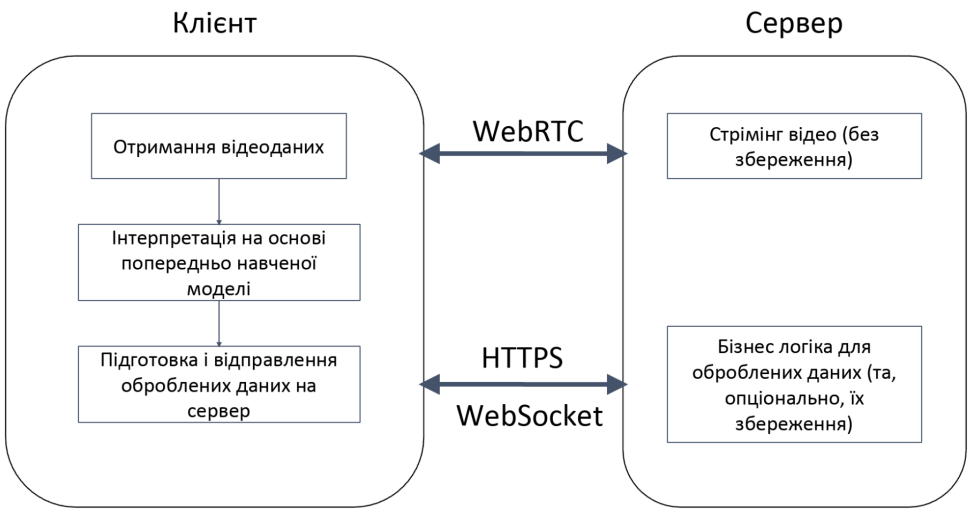
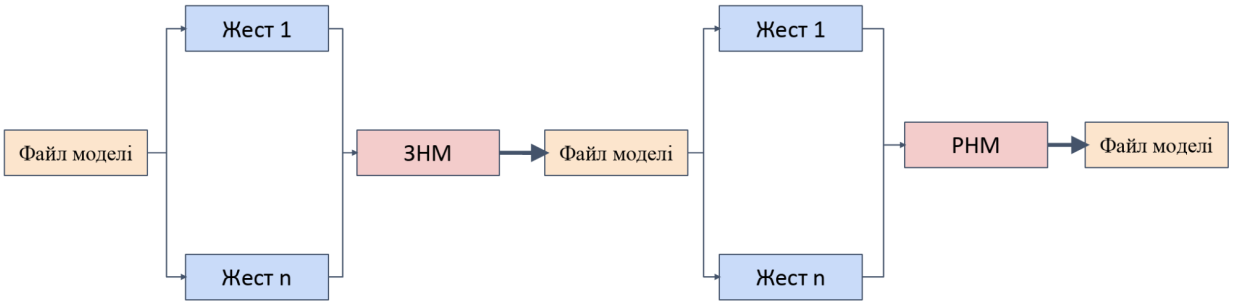


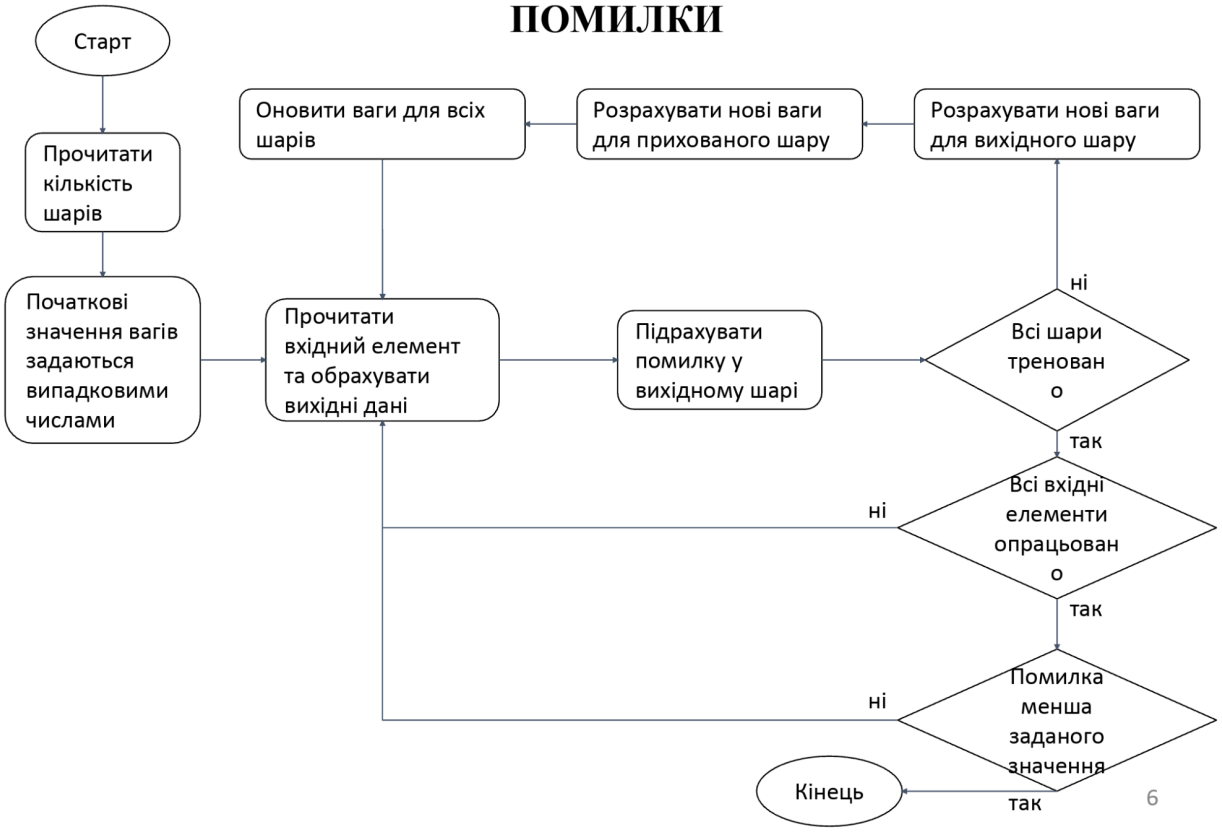
СХЕМА ВЗАЄМОДІЇ ПІД ЧАС НАВЧАННЯ МОДЕЛІ



ЗНМ - згорткова нейронна мережа
КНМ - рекурентна нейронна мережа

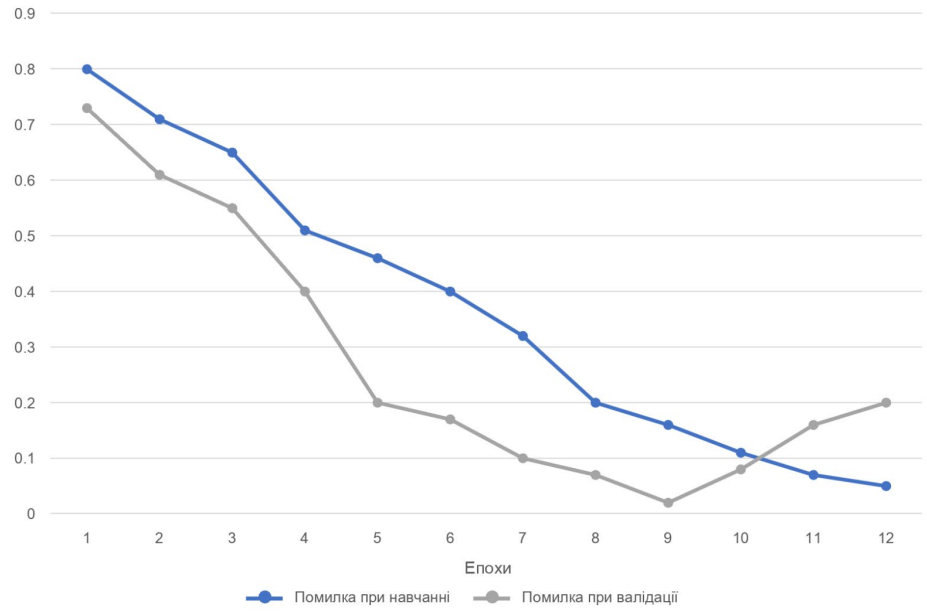
5

АЛГОРИТМ ЗВОРОТНОГО РОЗПОВСЮДЖЕННЯ ПОМИЛКИ



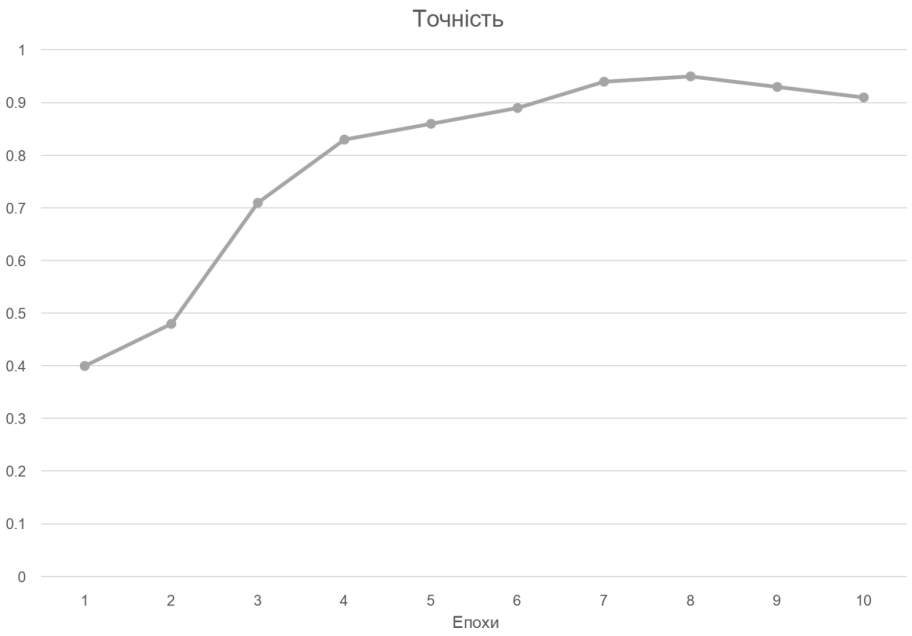
6

ПОМИЛКИ НАВЧАННЯ ТА ПЕРЕВІРКИ



7

ГРАФІК ТОЧНОСТІ МОДЕЛІ



8

ВИСНОВКИ

- Було розглянуто використання штучних нейронних мереж у завданнях розпізнавання та класифікації жестів із відеопослідовностей
- Було обрано базові моделі нейронних мереж (CNN та RNN) для основи навчання моделі інтерпретації
- Було підготовлено набір тренувальних даних
- Було розроблено навчальний комплекс для моделі розпізнавання та, власне, навчено модель (трансферне навчання).
- За результатами тестування навченої моделі отримано точність на рівні 92%.
- Було розроблено модель клієнт-серверної взаємодії за принципом товстого клієнта, що використовує попередньо навчену модель для інтерпретації жестів. В цій моделі вдалося отримати покращення точності розпізнавання жестів, використання мобільної платформи потенційно робить цю модель більш доступною.

9

АПРОБАЦІЯ РОБОТИ

Тези доповідей:

- Bohdan Maksymiuk PERSONALIZED E-LEARNING: LEVERAGING AI FOR INDIVIDUALIZED INCLUSIVE EDUCATION // Актуальні питання розвитку інформаційних технологій: тези доповідей V Всеукраїнської конференції молодих учених (Дніпро, 22 листопада 2023 р.)/ ДВНЗ «ПДТУ». – Дніпро: ПДТУ, 2023. с. 97-98

12

ДЯКУЮ ЗА УВАГУ!