МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ І ПРИКЛАДНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ФЕДОРИШИН БОРИС ВОЛОДИМИРОВИЧ

Допускається до захисту:
в.о. завідувача кафедри
інформаційних технологій,
д-р. техн. наук, професор
_____Наталія ВЕСЕЛОВСЬКА
«____»_____ 2026 р.

АНАЛІЗ НАЯВНИХ МЕТОДІВ ТА РОЗРОБКА ПРОГРАМНОГО РІШЕННЯ ДЛЯ РОЗПІЗНАВАННЯ ТА ПЕРЕКЛАДУ ДИНАМІЧНИХ ЖЕСТІВ УКРАЇНСЬКОЇ ЖЕСТОВОЇ МОВИ

Спеціальність 122 Комп’ютерні науки

Кваліфікаційна (бакалаврська) робота

Керівник:
Олександр ПАВЛЮК,
старший викладач кафедри
інформаційних технологій, к. т. н.,
_____

Оцінка: _____ / _____ / _____
(бали/за шкалою ЄКТС/за національною шкалою)

Голова ЕК: _____
(підпис)

Вінниця – 2026

АНОТАЦІЯ

Федоришин Б.В. Аналіз наявних методів та розробка програмного рішення для розпізнавання та перекладу динамічних жестів Української жестової мови. Спеціальність 122 «Комп'ютерні науки», освітня програма «Комп'ютерні науки». Донецький національний університет імені Василя Стуса, Вінниця, 2026.

У кваліфікаційній (бакалаврській) роботі проведено аналіз лінгвістичних особливостей УЖМ та методів автоматичного розпізнавання жестів. Розроблено програмний комплекс реального часу на основі фреймворку MediaPipe Holistic для екстракції ключових точок тіла та рекурентної нейромережі архітектури LSTM для класифікації динамічних жестів. Датасет включає 940 послідовностей для 10 класів. Модель досягла 100% точності на тестовій вибірці.

Ключові слова: українська жестова мова, розпізнавання жестів, MediaPipe, LSTM, нейронні мережі, комп'ютерний зір.

ABSTRACT

Fedoryshyn B.V. Analysis of Existing Methods and Development of a Software Solution for Recognition and Translation of Dynamic Ukrainian Sign Language Gestures. Specialty 122 «Computer Science», educational program «Computer Science». Vasyl Stus Donetsk National University, Vinnytsia, 2026.

The bachelor's thesis analyzes the linguistic features of Ukrainian Sign Language (USL) and reviews gesture recognition methods. A real-time software system was developed using the MediaPipe Holistic framework for keypoint extraction and an LSTM recurrent neural network for dynamic gesture classification. The dataset comprises 940 sequences across 10 gesture classes. The model achieved 100% accuracy on the test set.

Keywords: Ukrainian Sign Language, gesture recognition, MediaPipe, LSTM, neural networks, computer vision.

Зміст

ВСТУП

Актуальність теми

В Україні, за даними ГО «Відчуй», налічується понад 2,1 мільйона людей із вадами слуху різного ступеня, з яких приблизно 230 тисяч використовують жестову мову як основний або один з основних засобів комунікації [1]. З початком повномасштабного вторгнення Росії у лютому 2022 року ситуація суттєво погіршилась: кількість людей, які повністю або частково втратили слух, зросла приблизно на 20% [1]. Особливо вразливою групою є військовослужбовці, за оцінками фахівців, кожен другий український військовий зазнає ураження слухового апарату внаслідок акустичних баротравм від вибухів і потребує слухопротезування та подальшої реабілітації. Комунікаційні бар'єри систематично обмежують їхній доступ до освітніх, медичних, адміністративних послуг та ринку праці, що гостро актуалізує потребу у впровадженні ефективних інструментів цифрової доступності. [2]

Створення систем автоматичного сурдоперекладу тривалий час стримувалося технологічними обмеженнями. Проте масштабна еволюція методів штучного інтелекту, від алгоритмів класичного комп'ютерного зору, які вимагали використання спеціалізованих сенсорних рукавичок чи маркерів, до сучасних нейромережевих підходів, докорінно змінила ситуацію. Використання глибинного навчання дозволяє здійснювати безконтактний трекінг ключових точок тіла та аналізувати складні просторово-часові патерни рухів із високою точністю.

Незважаючи на значний прогрес у машинному розпізнаванні американської чи британської жестових мов, УЖМ залишається критично малодослідженою. Ключовим викликом є відсутність повноцінних автоматизованих систем, здатних розпізнавати та перекладати динамічні жести УЖМ безпосередньо з відеопотоку. Більшість існуючих рішень для українського сегмента обмежуються функціоналом текстових відеословників або систем

прямого перекладу (з тексту в жести за допомогою аватарів), ігноруючи складніше завдання зворотного відеорозпізнавання вільного мовлення.

Отже, розробка програмного комплексу для розпізнавання та перекладу динамічних жестів УЖМ становить значний науково-практичний інтерес. Вирішення цього завдання стане важливим кроком до подолання соціальної ізоляції нечуючих людей та забезпечення їхніх прав на безбар'єрне комунікаційне й цифрове середовище в Україні.

Мета і завдання дослідження

Метою дипломної роботи є розробка та програмна реалізація комплексу, здатного в режимі реального часу розпізнавати та класифікувати динамічні жести української жестової мови (УЖМ) на основі вхідного відеопотоку. Досягнення цієї мети базується на застосуванні методів комп'ютерного зору (зокрема, фреймворку MediaPipe) у поєднанні з рекурентними нейронними мережами архітектури LSTM/GRU для обробки часових рядів.

Для досягнення поставленої мети передбачено виконання таких завдань:

1. Проаналізувати лінгвістичні й просторово-часові особливості УЖМ, а також дослідити еволюцію підходів до автоматичного перекладу жестових мов.
2. Обґрунтувати вибір технологічного стека для реалізації програмного рішення, зокрема використання інструментарію MediaPipe Holistic, нейромережевих архітектур LSTM/GRU та мови програмування Python.
3. Сформувати, розмітити та підготувати до машинного навчання спеціалізований набір даних (датасет), що складається з відеозаписів динамічних жестів УЖМ.
4. Розробити топологію та здійснити навчання моделі глибокого навчання, орієнтованої на аналіз часових послідовностей виділених ключових точок тіла та рук користувача.
5. Реалізувати наскрізний конвеєр обробки відеоданих та інтегрувати навчену модель у вигляді функціонального прототипу програмного комплексу.

6. Провести комплексне експериментальне дослідження розробленого програмного забезпечення з метою кількісного оцінювання точності класифікації жестів та загальної продуктивності системи в реальному часі.

Об'єкт, предмет, наукова новизна, практичне значення

Об’єктом дослідження є процес автоматичного розпізнавання динамічних жестів на основі даних, отриманих за допомогою технологій комп’ютерного зору. Цей процес охоплює етапи безконтактної фіксації, екстракції та подальшої комп'ютерної інтерпретації рухів людини з неперервного відеопотоку.

Предметом дослідження виступають методи та архітектури глибокого навчання, зокрема рекурентні нейронні мережі типів LSTM та GRU. Вони розглядаються як основний інструмент для ефективного аналізу складних просторово-часових послідовностей ключових точок тіла та рук користувача.

Наукова новизна одержаних результатів полягає у зміщенні фокуса дослідження УЖМ до парадигми безпосереднього аналізу відеоданих. На відміну від більшості попередніх вітчизняних розробок, які спиралися на символьні, статистичні чи онтологічні підходи і обмежувалися текстовим перекладом вже вручну анотованих глос, у цій роботі вирішується завдання первинного візуального розпізнавання жестів. Застосування нейромережевих моделей дозволяє системі самостійно видобувати семантику з динамічних рухів у реальному часі, минаючи етап проміжної ручної транскрипції.

Практичне значення полягає у створенні дієвого програмного прототипу, що підтверджує можливість автоматичного перекладу української жестової мови за допомогою звичайної вебкамери. Запропоноване рішення здатне стати технологічним ядром для розробки сучасних асистивних технологій, спеціалізованих освітніх платформ та систем безбар'єрного обслуговування. Впровадження подібних інструментів інклюзивної комунікації сприятиме подоланню соціальної ізоляції нечуючих осіб та їхній глибшій інтеграції в суспільство.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ ТА АНАЛІЗ МЕТОДІВ

1.1 Лінгвістичні аспекти та особливості УЖМ

Українська жестова мова (УЖМ) є повноцінною та самостійною лінгвістичною системою, яка функціонує незалежно від звукової української мови і не є її жестовою транслітерацією чи спрощеним кодом. Вона володіє власним унікальним лексичним складом, самобутньою етимологією та незалежною граматичною структурою, що формувалися протягом десятиліть всередині спільноти нечуючих [3]. Цей факт є принципово важливим для розуміння задачі автоматичного перекладу: УЖМ – це не закодована версія усної мови, а окрема мовна система з власними правилами породження висловлювань.

Лінгвістична самостійність та географічна специфіка. Оскільки УЖМ історично формувалася в межах вітчизняного співтовариства нечуючих осіб, вона має суттєві лексичні та граматичні відмінності від американської (ASL), британської (BSL), польської (PJM) та інших жестових мов [4]. Цей факт безпосередньо впливає на задачу розробки систем автоматичного перекладу: жодна з існуючих зарубіжних моделей, навчених на ASL або DGS, не може бути безпосередньо перенесена на УЖМ без повного перенавчання на спеціалізованому корпусі. Географічна та культурна зумовленість жестового лексикону є однією з головних причин критичного дефіциту автоматизованих рішень для УЖМ порівняно з ресурсно забезпеченими мовами.

Просторовий синтаксис. Визначальною типологічною характеристикою УЖМ є її просторовий синтаксис. На відміну від лінійних звукових мов, де граматичні відношення передаються переважно через порядок слів та морфологічні маркери, в УЖМ граматичні зв'язки, часові відношення та направленість дії кодуються шляхом розташування референтів у тривимірному комунікативному просторі перед мовцем [5]. Наприклад, встановивши місця двох іменників у просторі, мовець може вказати на відношення між ними, просто спрямувавши рух дієслівного знаку від одного місця до іншого. Цей механізм

принципово відрізняється від усіх відомих усних мов і є важливим аргументом на користь того, що системи автоматичного перекладу не можуть механічно відображати граматику УСМ на УЖМ.

Морфологічна класифікація жестів. З точки зору морфологічної структури, жестові одиниці поділяються на дві базові категорії: статичні та динамічні. Статичні жести – дактильний алфавіт (окремі літери) та базові числівники – характеризуються незмінним положенням руки у просторі протягом усього часу фіксації. З точки зору комп'ютерного зору вони є значно простішими для автоматичного розпізнавання: задача зводиться до класифікації одиночного зображення кисті, що може бути вирішене навіть за допомогою CNN на статичних кадрах.

Натомість динамічні жести, що передають повноцінні слова, концепції або фрази, передбачають зміну стану в часі [6]. Їхня семантика закодована не в окремому положенні рук, а в траєкторії переходу між станами – в самому русі. Для комп'ютерних систем обробка динамічних жестів становить якісно вищий рівень складності: вимагається аналіз не розрізнених статичних кадрів, а неперервних відеопослідовностей із збереженням темпорального контексту. Додатковою складністю є ефект коартикуляції – злиття кінцевої фази одного жесту з початковою фазою наступного, характерне для природного темпу мовлення. Через коартикуляцію межі між сусідніми жестами розмиваються, і алгоритм не може покластися на чіткі «паузи» між знаками як на маркери сегментації.

Параметрична модель динамічного жесту. Для коректної формалізації задачі комп'ютерного зору динамічний жест описується як багатовимірний часовий ряд векторів ознак. В лінгвістиці жестових мов загальноприйнятою є параметрична модель, введена Стокоу (Stokoe, 1960) та розширена наступними дослідниками [7]. Відповідно до неї семантичне значення жесту формується сукупністю п'яти ключових параметрів.

Перший параметр – конфігурація кисті (handshape) – визначає форму, утворену пальцями: стиснутий кулак, відкрита долоня, вказівний жест тощо. В

УЖМ нараховується кілька десятків базових конфігурацій. Другий параметр – орієнтація долоні – описує просторовий напрямок, у який звернена долоня: до мовця, від мовця, вгору, вниз. Третій параметр – місце артикуляції (location) – вказує на ту зону простору перед тілом або на тілі, де виконується жест: нейтральний простір перед грудьми, голова, плечі тощо. Четвертий параметр – рух (movement) – є найбільш інформативним з точки зору динамічного розпізнавання: він описує векторну траєкторію кисті, напрямок, амплітуду та швидкість переміщення. П'ятий параметр – немануальні компоненти – охоплює супроводжуючі мімічні маркери: артикуляцію губ, рух брів, нахили та повороти голови, якими граматикалізується модальність висловлювання [8]. Зміна хоча б одного з цих параметрів здатна повністю трансформувати семантику жесту або перетворити його на омонім.

Зв'язок лінгвістичної специфіки з вибором методів ML. Зазначені лінгвістичні та біомеханічні особливості безпосередньо визначають вимоги до архітектури системи автоматичного розпізнавання. Оскільки семантика динамічного жесту розкривається виключно через просторово-часовий характер розгортання рухів, класичні методи розпізнавання статичних патернів є принципово непридатними. Задача потребує моделі, здатної одночасно відстежувати просторові характеристики (координати ключових точок тіла та рук у кожному кадрі) та часові залежності між ними (послідовність позицій у багатокадровому контексті). Саме ця вимога безпосередньо обумовлює вибір рекурентних нейронних мереж архітектури LSTM або GRU – єдиного класу архітектур, здатного ефективно моделювати довгострокові залежності у часових рядах довільної природи [9].

1.2 Еволюція підходів: від символьних до нейромережевих

1.2.1 Символьні та онтологічні підходи

Історично першими спробами автоматизації перекладу жестових мов стали символьні та онтологічні підходи, що базуються на парадигмі формалізованого кодування знань (knowledge engineering). Їхня фундаментальна ідея полягає в тому, що лінгвістичні знання можна вичерпно формалізувати у вигляді

скінченного набору детермінованих правил, синтаксичних граматик та структурованих семантичних мереж, а процес перекладу звести до послідовного застосування цих правил [10]. Такий підхід є природним першим кроком у будь-якій галузі автоматизованої обробки мови і відповідає загальній траєкторії розвитку машинного перекладу: від правилових RBMT-систем 1950–1980-х років до статистичних та нейронних методів.

У дослідженні УЖМ найбільш ґрунтовним прикладом цієї парадигми є праці О. Лозинської (2016, 2019) [11]. Розроблений нею метод спирається на граматично доповнену онтологію та багаторівневий словник концептів. Онтологія формалізує семантичні відношення між поняттями УЖМ та УСМ у вигляді ієрархічних мереж концептів із властивостями та зв'язками. Словник концептів зіставляє кожній лексичній одиниці УСМ відповідний жестовий еквівалент у формі глоси – умовного текстового позначення жесту. Механізм трансляції реалізовується у два кроки: вхідний текст УСМ розбирається граматичним аналізатором із виявленням частин мови та синтаксичних ролей, після чого кожна лексична одиниця зіставляється зі словником концептів засобами онтологічного виведення для отримання відповідної глоси УЖМ.

Принципово важливо підкреслити: система повністю оперує на символьному рівні, працюючи з текстовими метаданими та анотаціями. Задача розпізнавання жесту безпосередньо з відео залишається поза межами підходу – передбачається, що жест уже ідентифікований фахівцем-сурдоперекладачем і поданий у вигляді глоси. Іншими словами, онтологічна система вирішує задачу «глоса → текст» або «текст → глоса», але не «відео → текст».

Безперечними перевагами rule-based систем є детермінованість поведінки, повна прозорість логіки прийняття рішень та висока інтерпретованість результатів. Для завдань, де коректність кожного кроку перекладу є критичною, ці властивості мають значну цінність. Водночас онтологічні підходи мають системні обмеження. Вкрай низька масштабованість: кожне нове правило та кожна нова лексична одиниця потребують ручного опису кваліфікованого лінгвіста. Абсолютна залежність від попередньої ручної анотації та принципова

неможливість роботи з неструктурованими відеоданими [13]. Нарешті, жорсткі правила не здатні алгоритмізувати природну варіативність артикуляції, індивідуальні стилі виконання жестів та ефекти коартикуляції реального мовлення. Ці обмеження закономірно зумовили еволюційний зсув до підходів, керованих даними.

1.2.2 Статистичні методи (SMT)

Статистичний машинний переклад (Statistical Machine Translation, SMT) виник як спроба подолати головний недолік rule-based систем – необхідність ручного кодування знань. Парадигмальний зсув полягав у наступному: замість того щоб програмувати правила перекладу вручну, системі надаються великі масиви паралельних двомовних текстів, і вона автоматично виявляє статистичні закономірності відповідності між мовами, оптимізуючи ймовірнісні параметри моделі [14].

Математичну основу SMT становить теорема Байєса, застосована до задачі перекладу: система шукає цільове речення T, яке максимізує умовну ймовірність P(T|S) при заданому вихідному реченні S. За теоремою Байєса, $P(T|S) \propto P(S|T) \cdot P(T)$, де P(S|T) – модель перекладу (translation model), що описує лексичні відповідності між мовами, а P(T) – мовна модель (language model), що забезпечує граматичну правильність цільового тексту. Навчання моделі перекладу потребує вирівнювання (alignment) – встановлення відповідностей між словами вихідного та цільового текстів на рівні пар речень паралельного корпусу.

У контексті УЖМ застосування SMT-підходу досліджено у праці О. Лозинської (2024). Для автоматичного вирівнювання між лексичними одиницями УСМ та глосами УЖМ використано класичні моделі вирівнювання IBM Models 1–4. Навчання та оптимізація ймовірнісних параметрів здійснювались за допомогою ітеративного алгоритму очікування-максимізації (EM-алгоритму) на двомовних корпусах [11]. EM-алгоритм працює в два кроки: на E-кроці (expectation) обчислюються очікувані значення латентних вирівнювань при поточних параметрах моделі; на M-кроці (maximization) параметри оновлюються для максимізації повної правдоподібності. Ітераційне

чергування цих кроків гарантує монотонне зростання правдоподібності навчальних даних.

Ключовим викликом при застосуванні SMT до жестових мов є фундаментальна структурна розбіжність між УСМ та УЖМ. Порядок знаків у УЖМ суттєво відрізняється від порядку слів в УСМ через просторовий синтаксис жестової мови: підмет, присудок та додаток можуть розташовуватись у зовсім іншій послідовності порівняно з відповідним реченням УСМ. Це призводить до так званих «схрещених вирівнювань» (crossing alignments), які є проблемою для всіх фразових SMT-моделей і знижують якість перекладу цілих речень.

SMT-системи демонструють вищу гнучкість порівняно з онтологічними підходами та здатні обробляти більший словник без ручного кодування. Однак вони зберігають принципове фундаментальне обмеження: усі операції відбуваються виключно на текстовому рівні – вхідні дані являють собою глоси, а не відеозаписи. Критична залежність від масштабних паралельних корпусів із ручною розміткою та неспроможність видобувати семантику безпосередньо з візуального сигналу зумовили перехід до методів комп'ютерного зору.

1.2.3 Класичний комп'ютерний зір

Принциповим якісним кроком у розвитку систем автоматичного сурдоперекладу стало залучення методів класичного комп'ютерного зору. Вперше стало можливим працювати безпосередньо з візуальним сигналом – відеозаписом виконання жесту, – минаючи проміжний етап ручної лінгвістичної анотації. Традиційний пайплайн обробки в межах цієї парадигми складається з чотирьох послідовних етапів, кожен з яких вирішує окрему підзадачу.

Попередня обробка. На першому етапі вхідний відеокадр підготовляється до аналізу. Застосовуються алгоритми підсилення контрасту, шумозаглушення та колірної нормалізації. Ключовим завданням є сегментація ділянок рук: виділення пікселів, що належать кистям, з фону. Для цього традиційно використовувались методи порогової бінаризації в просторах HSV або YCbCr, де

шкірний тон займає компактну область, та алгоритм підрахунку гістограм кольорів для адаптивного налаштування порогів.

Виділення ознак. Другий етап – формування компактного інформативного векторного опису сегментованого зображення. Дескриптор направлених градієнтів (HOG, Histogram of Oriented Gradients) обчислює розподіл напрямків градієнтів яскравості в локальних ділянках зображення, кодуючи форму та текстуру кисті. Алгоритм Canny застосовувався для виявлення контурів рук – ліній різкого перепаду яскравості, що відповідають краям пальців та долоні. Хвилеподібні перетворення (Wavelet Transform) надавали мультимасштабний аналіз текстур. Результатом цього етапу є числовий вектор ознак фіксованої розмірності, що описує поточний кадр.

Тимчасове моделювання. Оскільки динамічний жест є часовою послідовністю, сукупність векторів ознак послідовних кадрів потребує моделювання з урахуванням часових переходів. Для цього широко застосовувались Приховані Марковські Моделі (HMM). HMM моделює жест як послідовність прихованих станів (наприклад, «початкова фаза», «пікова фаза», «завершення»), де кожен стан генерує спостережувані вектори ознак відповідно до деякого ймовірнісного розподілу, а переходи між станами описуються матрицею ймовірностей переходів. Параметри HMM навчались на прикладах виконання кожного жесту за допомогою алгоритму Баума-Велша (різновид EM). Детальний методологічний аналіз застосування таких конвеєрів для жестових мов наведено у праці Т. Басюка та А. Василюка (2024) [15].

Переваги та обмеження. Класичний комп'ютерний зір вперше дозволив вирішувати задачу розпізнавання безпосередньо з відеосигналу без текстової анотації, що стало концептуальним проривом порівняно з попередніми підходами. Водночас ці методи виявились критично залежними від умов зйомки. Алгоритми сегментації за кольором шкіри давали збої при зміні освітлення, наявності схожого за кольором фону або при різних відтінках шкіри різних виконавців [16]. Жорстка необхідність ручного проектування ознак (hand-crafted features) – процесу, де дослідник вручну обирав математичні дескриптори,

вважаючи їх інформативними – призводила до низької стійкості та слабкої узагальнюваності алгоритмів на нових користувачів та умови. Ці системні недоліки закономірно зумовили перехід до методів глибокого навчання.

1.2.4 Сучасні підходи на основі глибокого навчання

Перехід до методів глибокого навчання (Deep Learning) ознаменував фундаментальну зміну епістемологічної парадигми в задачах розпізнавання жестів. Принциповий зсув полягав у відмові від ручного проектування математичних ознак: замість того щоб дослідник вирішував, які характеристики сигналу є інформативними, нейронна мережа самостійно навчається виділяти ієрархічні абстракції безпосередньо з «сирих» даних в процесі оптимізації функції втрат [17].

CNN для просторового аналізу. Для обробки візуальної інформації в глибокому навчанні традиційно застосовуються згорткові нейронні мережі (CNN). Операція згортки локально аналізує просторові патерни пікселів за допомогою навчених фільтрів, автоматично виявляючи краї, текстури та форми на різних масштабах. Послідовне застосування кількох шарів згортки формує ієрархію ознак від низькорівневих (краї пікселів) до високорівневих (конфігурації кистей, положення рук). Проте CNN аналізують окремі кадри ізольовано і не зберігають контексту між ними, що є критичним обмеженням для задачі динамічних жестів [18].

RNN/LSTM/GRU для часового моделювання. Для моделювання динаміки рухів у часі використовуються рекурентні архітектури, насамперед LSTM та GRU. На відміну від CNN, рекурентні мережі мають прихований стан h_t, що передається від кроку до кроку і акумулює інформацію про всю попередню послідовність. Механізм вентилів LSTM (forget, input, output gates) забезпечує вибіркове оновлення та збереження стану клітини c_t, що вирішує проблему зникаючого градієнта класичних RNN і дозволяє утримувати контекст на довгих послідовностях [27].

MediaPipe як революційний проміжний шар. Значним технологічним проривом стала поява фреймворку MediaPipe Holistic від Google. Традиційний

підхід «CNN на пікселях» потребував потужного GPU навіть для базового інференсу. MediaPipe запропонував принципово інший підхід: замість безпосередньої обробки «сирих» пікселів нейронними мережами, попередньо навчені легковагові детектори перетворюють кадр у компактне структуроване представлення – набір нормалізованих координат ключових точок тіла та рук [19]. Отримані координати є значно більш компактними (258 чисел замість мільйонів пікселів) та інваріантними до освітлення, кольору шкіри та фону. Це дозволяє навчати класифікуючу LSTM-модель лише на послідовностях координат, що не вимагає GPU ні для навчання (можливе на CPU або хмарному GPU), ні тим більше для інференсу в реальному часі.

Transformer-архітектури як перспективний напрям. Поряд з LSTM/GRU активно досліджуються Transformer-архітектури з механізмом самовзаємозв'язку (self-attention). Механізм уваги дозволяє моделі зважувати внесок кожного часового кроку послідовності при формуванні передбачення, що теоретично краще вловлює довготривалі залежності порівняно з рекурентними мережами. Проте Transformer-моделі потребують значно більшого обсягу даних для навчання та вищих обчислювальних ресурсів для інференсу [20]. За умов обмеженого датасету, характерних для нішевих жестових мов типу УЖМ, рекурентні архітектури LSTM/GRU демонструють вищу ефективність навчання.

Практичні реалізації для УЖМ. Практичним прикладом сучасної реалізації для українського сегмента є дослідження О. Зайцевої (2025), що демонструє успішну інтеграцію методів глибокого навчання для розпізнавання динамічної лексики УЖМ із відеопотоку [21]. Це підтверджує принципову застосовність нейромережевого підходу до специфіки УЖМ і слугує важливим орієнтиром при проектуванні розроблюваного комплексу.

Таким чином, архітектурний симбіоз MediaPipe та LSTM/GRU є оптимальним інженерним вибором для задачі реального часу: MediaPipe ефективно вирішує задачу просторової локалізації суглобів, абстрагуючи систему від варіацій зовнішнього середовища, а LSTM/GRU класифікують отримані часові ряди координат.

1.3 Огляд існуючих систем розпізнавання жестових мов

Проектування ефективного програмного комплексу вимагає ретельного аналізу існуючих аналогів – як для обґрунтування наукової новизни, так і для свідомого запозичення ефективних архітектурних рішень. Сучасний ландшафт систем автоматичного розпізнавання жестових мов характеризується значною фрагментованістю: від потужних комерційних продуктів із закритим кодом до академічних прототипів і open-source рішень. Переважна більшість існуючих систем орієнтована на добре ресурсно забезпечені жестові мови – насамперед ASL та DGS – і не може бути безпосередньо застосована до УЖМ.

Комерційні системи на основі глибокого навчання. Найвідомішим продуктом у сегменті комерційних рішень є екосистема SignAll, що розробляє системи перекладу ASL у текст. Система використовує багатокамерне захоплення відеопотоку та складні ансамблі CNN-моделей для просторового аналізу рухів рук. Паралельно Google розробив алгоритми Sign Language Detection, інтегровані у відеосервіси: вони виявляють факт жестикуляції у відеопотоці та виконують базову класифікацію жестів ASL. Обидва рішення демонструють високу точність у своєму домені, однак мають критичні практичні обмеження: закритий вихідний код унеможливлює адаптацію до інших мов, висока вартість апаратного забезпечення обмежує доступність, а орієнтація виключно на ASL робить їх нерелевантними для завдань УЖМ [12].

Системи на основі спеціалізованого обладнання. Паралельним напрямом стало використання сенсорів глибини (RGB-D), зокрема Microsoft Kinect. Інфрачервоний структурований світловий датчик Kinect генерує карту глибини сцени, що дозволяє ефективно сегментувати тіло людини від фону незалежно від кольору та освітлення. Це суттєво спрощувало завдання виділення ознак і давало доступ до тривимірних координат суглобів скелету. Академічних досліджень з використанням Kinect для жестових мов опубліковано значну кількість [18]. Однак прив'язка до конкретного обладнання (Microsoft Kinect знято з виробництва у 2017 році), його висока вартість та необхідність спеціального монтажу унеможливлюють масове практичне застосування. Аналогічна

проблема стосується рукавичок із сенсорами (data gloves) – попри їхню точність, вони вимагають фізичного надягання і незручні в повсякденному використанні.

Академічні системи на стандартизованих датасетах. В академічному середовищі роль де-факто стандарту займають системи, навчені та оцінені на датасеті RWTH-PHOENIX-Weather 2014. Цей корпус містить близько 9 годин відеозаписів телевізійних прогнозів погоди з DGS-сурдоперекладом і є найбільшим публічно доступним датасетом для неперервного розпізнавання жестів. На його основі запропоновано десятки архітектур: від CNN+HMM гібридів до повністю нейронних Transformer-моделей [22]. Проте ці системи мають суттєве обмеження застосовності: вони навчені на вузькому предметному домені (лексика прогнозів погоди) і демонструють різке падіння точності поза ним. Крім того, специфіка DGS і зовсім не відповідає УЖМ, що виключає можливість трансферу без повного перенавчання.

Open-source рішення на MediaPipe. Останніми роками значна кількість open-source проєктів, що комбінують MediaPipe із рекурентними мережами, з'явилась на GitHub. Більшість із них реалізують концептуально схожий підхід: MediaPipe Holistic виділяє координати ключових точок, LSTM класифікує послідовності [23]. Такі проєкти є цінними навчальними ресурсами та демонструють принципову реалізовність підходу. Водночас переважна більшість обмежена 10–30 ізольованими жестами ASL або ізраїльської жестової мови (ISL), не має стандартизованої методології оцінки якості, не оптимізована для реального часу на CPU та не адаптована до специфіки УЖМ.

Слов'янські та вітчизняні розробки. Щодо УЖМ, в академічному та комерційному просторі спостерігається критичний дефіцит готових рішень. Ресурсна асиметрія між добре описаними мовами (ASL, DGS, BSL) та малодослідженими (УЖМ, РЖМ, польська PJM) є значною. Існуючі вітчизняні розробки здебільшого представлені відеословниками УЖМ (статичні бази даних) або системами синтезу жестів із тексту за допомогою 3D-аватарів – тобто системами прямого перекладу. Задача зворотного відеорозпізнавання вільного жестового мовлення УЖМ залишається практично не вирішеною [15]. Наявні

дослідження (Лозинська, Басюк) вирішують суміжні задачі на текстовому рівні, що підтверджено у підрозділі 1.2.

Результати систематизованого порівняльного аналізу наведено в таблиці 1.1.

Таблиця 1.1 Порівняльний аналіз існуючих систем розпізнавання жестових мов

Система / Проєкт	Цільова мова	Використаний підхід	Ключові обмеження
SignAll / Google	ASL	Багатокамерний комп'ютерний зір, Deep Learning	Закритий код, висока вартість, відсутність підтримки УЖМ
Системи на базі MS Kinect	Різні	RGB-D сенсори глибини, Azure CV	Прив'язка до специфічного дороговартісного обладнання
PHOENIX-2014 (Академічні)	DGS	Transformer, CNN + RNN	Вузька предметна область (погода), складність адаптації
Open-source (MediaPipe+LSTM)	ASL / ISL	MediaPipe Holistic + рекурентні мережі	Вкрай малий розмір словника (10-30 жестів), відсутність оптимізації
Існуючі рішення для УЖМ	УЖМ	Текстовий/онтологічний переклад, 3D-аватари	Відсутність функції розпізнавання відео в реальному часі
Розроблюваний комплекс	УЖМ	MediaPipe + LSTM/GRU	Обмежений словник (прототип)

Проведений аналіз беззаперечно засвідчує наявність суттєвої технологічної прогалини: жодна з існуючих систем не вирішує задачу автоматичного відеорозпізнавання динамічних жестів УЖМ у режимі реального часу на стандартному апаратному забезпеченні. Розроблюваний комплекс займає

саме цю нішу, поєднуючи доступність звичайної вебкамери, ефективність MediaPipe Holistic для екстракції координат та LSTM-архітектуру, адаптовану до специфіки УЖМ.

Висновки до розділу 1

У першому розділі проведено комплексний аналіз предметної області, лінгвістичних особливостей української жестової мови (УЖМ) та еволюції методів її автоматизованого перекладу . За результатами теоретичного дослідження зроблено такі висновки:

1. Лінгвістична специфіка задачі. Доведено, що УЖМ є багатовимірною системою із власним просторовим синтаксисом. Динамічні жести мають складну просторово-часову природу, а їхнє значення формується сукупністю параметрів (конфігурація, локалізація, траєкторія, міміка). Наявність ефекту коартикуляції та неперервності рухів робить неможливим використання алгоритмів аналізу статичних зображень і вимагає обробки часових рядів.
2. Еволюція методів розпізнавання. Проведений аналіз дозволяє простежити чіткий хронологічний перехід від формалізованих підходів до нейромережевих методів. Символьні та онтологічні підходи, а також статистичні методи (досліджені у вітчизняних працях О. Лозинської), заклали теоретичну базу формального опису УЖМ, однак їхнім критичним недоліком залишається нездатність обробляти сирий відеосигнал. Водночас класичні методи комп'ютерного зору (систематизовані Т. Басюком та А. Василюком), хоч і вперше забезпечили роботу з відеоданими, виявилися нестійкими до змін умов зйомки через ручне проектування ознак.
3. Обґрунтування технологічного підходу. Методи глибокого навчання усувають зазначені обмеження завдяки автоматичному виділенню ієрархічних ознак та стійкості до варіацій середовища. Визначено, що найефективнішим архітектурним рішенням для розпізнавання динамічних жестів у реальному часі є гібридний підхід. Фреймворк MediaPipe забезпечує надійну екстракцію просторових координат ключових точок тіла, абстрагуючись від візуального шуму. Для подальшого аналізу та класифікації

отриманих векторних послідовностей оптимальним математичним апаратом є застосування рекурентних нейронних мереж архітектури LSTM або GRU.

Отже, у розділі сформовано ґрунтовну теоретико-методологічну базу, яка слугує концептуальним фундаментом для проектування архітектури програмного комплексу, підготовки датасету та навчання нейромережевих моделей, що буде детально розкрито в наступних розділах роботи.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО КОМПЛЕКСУ

2.1 Обґрунтування вибору технологічного стеку

Проектування ефективної системи автоматичного сурдоперекладу вимагає формування збалансованого технологічного стеку, здатного забезпечити високошвидкісне виконання операцій захоплення відео, екстракції просторових ознак та нейромережевої класифікації в режимі реального часу. При виборі компонентів враховувались три ключові критерії: обчислювальна ефективність (здатність системи підтримувати прийнятну частоту кадрів на стандартному апаратному забезпеченні без GPU), зрілість екосистеми (наявність документації, активної спільноти та стабільних версій), а також сумісність компонентів між собою на рівні версій залежностей.

Python як основна мова реалізації. Базовою мовою програмування для реалізації обчислювального ML-ядра обрано Python (версія 3.10). Такий вибір зумовлений його статусом де-факто індустріального стандарту у сфері машинного навчання та наукових обчислень. Ключовими перевагами є наявність масштабної екосистеми спеціалізованих математичних бібліотек (NumPy для матричних операцій, SciPy для наукових обчислень, scikit-learn для оцінювання моделей), висока швидкість алгоритмічного прототипування та широка підтримка з боку хмарних платформ навчання (зокрема Google Colaboratory) [24]. Водночас слід зазначити, що Python є інтерпретованою мовою з відносно низькою «сирою» швидкістю виконання порівняно з компільованими мовами. Проте у задачах ML цей недолік нівелюється тим, що ресурсомісткі операції (матричні множення, нейромережевий інференс) виконуються оптимізованими C++-бінарниками через відповідні Python-прив'язки (bindings).

OpenCV для низькорівневої обробки відео. Для розв'язання задач низькорівневого комп'ютерного зору інтегровано відкриту бібліотеку OpenCV (Open Source Computer Vision Library). Її функціонал застосовується на першому рівні конвеєра: апаратне захоплення відеопотоку з вебкамери через інтерфейс VideoCapture, декодування стисненого відеосигналу в матриці пікселів,

конвертація колірних просторів BGR → RGB (необхідна для коректної роботи MediaPipe, що очікує RGB-вхід) та нормалізація яскравості кадрів. OpenCV реалізована на C++ і надає Python-прив'язки, що забезпечує продуктивність C++ при зручності Python. Бібліотека підтримує широкий спектр форматів відеопристроїв, включаючи USB-камери, IP-камери та мобільні камери, підключені через програмні мости (наприклад, Iriun Webcam).

MediaPipe Holistic як екстрактор просторових ознак. Ключовим компонентом конвеєра екстракції ознак виступає фреймворк MediaPipe Holistic від компанії Google. Вибір MediaPipe над альтернативами (OpenPose, AlphaPose) зумовлений кількома технічними перевагами. По-перше, MediaPipe використовує легковагові нейромережеві детектори, оптимізовані для роботи виключно на CPU в режимі реального часу, тоді як OpenPose вимагає потужного GPU [25]. По-друге, модуль Holistic забезпечує одночасний синхронний трекінг трьох типів ключових точок за один прохід: 33 точки пози тіла (Pose), 468 точок сітки обличчя (Face Mesh) та по 21 точці для кожної кисті (Hands) – загалом 543 landmarks. По-третє, координати всіх точок нормалізуються відносно розмірів кадру в діапазон [0, 1], що забезпечує інваріантність до роздільної здатності вхідного відео. Нарешті, параметр model_complexity=0 дозволяє перейти в режим максимальної швидкості за рахунок незначного зниження точності детекції, що є оптимальним компромісом для задачі реального часу.

Важливим інженерним рішенням стало виключення 468 facial landmarks із фінального вектора ознак. Незважаючи на те, що MediaPipe Holistic здатний їх виділяти, аналіз показав, що детекція точок обличчя є нестабільною при нетипових кутах зйомки та різному освітленні. Крім того, включення 468×3=1404 додаткових числових значень збільшило б розмірність вхідного простору з 258 до 1662 елементів, підвищивши обчислювальну складність моделі та час навчання без пропорційного приросту точності для обраного набору жестів, семантика яких визначається переважно рухами рук та пози тіла.

TensorFlow/Keras для проєктування та навчання моделей. Проєктування та навчання рекурентних архітектур (LSTM/GRU) реалізовано на базі платформи

TensorFlow (версія 2.16.2) із використанням високорівневого API Keras. Вибір TensorFlow над PyTorch зумовлений кількома факторами. Keras надає декларативний підхід до конструювання моделей, що значно прискорює прототипування складних рекурентних архітектур: топологія LSTM-мережі описується кількома рядками коду з автоматичним підбором розмірностей шарів. TensorFlow містить оптимізовані реалізації алгоритму зворотного поширення помилки в часі (BPTT) та забезпечує нативну підтримку GPU-прискорення через технологію CUDA [26]. Окремою перевагою є потужна екосистема супутніх інструментів: TensorBoard для моніторингу процесу навчання в реальному часі та TensorFlow Lite для оптимізації та розгортання моделей на мобільних пристроях у перспективі.

Слід також зазначити практичний аспект вибору версії: комбінація TensorFlow 2.16.2 та MediaPipe 0.10.14 є перевіреною конфігурацією, що не має конфліктів по версії бібліотеки protobuf у середовищі Google Colaboratory з Python 3.10. Це критично важливо для відтворюваності результатів, оскільки конфлікти залежностей між різними версіями цих бібліотек є поширеною проблемою у подібних проєктах.

Tkinter та Pillow для графічного інтерфейсу. Для розробки графічного інтерфейсу користувача (GUI) використано стандартну бібліотеку Tkinter, що входить до базового дистрибутива Python. Tkinter є кросплатформною обгорткою над бібліотекою Tk/Tcl та не потребує встановлення додаткових залежностей. Для відображення відеокадрів у вікні Tkinter залучено бібліотеку Pillow (PIL): вона виконує конвертацію масивів NumPy у формат PhotoImage, який є нативним форматом зображень Tkinter. Архітектура GUI реалізує схему «виробник-споживач» із використанням черги (queue.Queue): фоновий потік Engine виконує захоплення відео та нейромережевий інференс, передаючи готові результати в чергу, тоді як основний потік GUI опитує чергу та оновлює інтерфейс через метод after(). Така архітектура запобігає «зависанню» інтерфейсу під час ресурсомістких обчислень.

Обґрунтування вибору базових технологій порівняно з популярними альтернативами наведено в таблиці 2.1.

Таблиця 2.1 Порівняльний аналіз компонентів технологічного стеку

Категорія	Обране рішення	Альтернатива	Обґрунтування вибору
Виділення ключових точок	MediaPipe Holistic	OpenPose	MediaPipe є значно легшим, працює на CPU в реальному часі (>25 FPS). OpenPose вимагає потужного GPU і є складнішим у розгортанні.
Фреймворк Deep Learning	TensorFlow / Keras	PyTorch	Keras надає швидший старт для прототипування RNN-мереж. TensorFlow має кращі інструменти для конвертації моделей у продакшен (TFLite).
Графічний інтерфейс	Tkinter + Pillow	PyQt	Tkinter є вбудованим у Python і не потребує додаткових залежностей. Достатній функціонал для прототипу з багатопотоковим відеопотоком.

Обраний технологічний стек поєднує продуктивність C++-ядер (OpenCV та TensorFlow) із гнучкістю та швидкістю розробки на Python, формуючи надійну та відтворювану архітектуру для вирішення задачі класифікації динамічних жестів у режимі реального часу.

2.2 Архітектура програмного комплексу

Розробка ефективної системи розпізнавання динамічних жестів у режимі реального часу вимагає побудови чіткої конвеєрної архітектури (pipeline). Запропонований програмний комплекс побудовано за принципом багаторівневості та слабкої зв'язності компонентів (loose coupling): кожен модуль виконує єдину чітко визначену функцію та взаємодіє з сусідніми модулями виключно через стандартизовані інтерфейси передачі даних. Така архітектура

спрощує тестування та налагодження окремих компонентів, гарантує низьку затримку обчислень та дозволяє замінювати або вдосконалювати будь-який рівень незалежно від решти системи. Архітектура складається з п'яти логічних рівнів обробки.

Рівень 1. Модуль введення даних. Відповідає за первинне захоплення візуальної інформації та реалізований засобами OpenCV. Модуль абстрагує джерело відеосигналу, підтримуючи два режими: захоплення живого відеопотоку з вебкамери (основний режим для реального використання) та зчитування попередньо збережених відеофайлів (для офлайн-тестування та налагодження). На апаратному рівні модуль встановлює параметри камери: роздільну здатність (1280×720 або адаптовану під пристрій), частоту кадрів (30 FPS як цільове значення) та розмір буфера захоплення (1 кадр для мінімізації затримки). Кожен захоплений кадр горизонтально дзеркалюється (flip) для природного відображення з точки зору користувача – щоб рухи рук відповідали інтуїтивним очікуванням. На виході модуль генерує послідовність матричних зображень у просторі BGR із розмірністю (H, W, 3), що передаються на наступний рівень.

Рівень 2. Модуль екстракції просторових ознак. Здійснює ключове перетворення: від «сирих» піксельних даних до структурованих математичних векторів. Вхідний BGR-кадр конвертується у RGB та передається у MediaPipe Holistic з параметром model_complexity=0 для максимальної швидкості. Модуль генерує координати 75 ключових точок із загальною розмірністю 258 числових ознак: 33 точки пози тіла (кожна з чотирма значеннями: x, y, z, visibility) та по 21 точці для лівої та правої руки (кожна з трьома значеннями: x, y, z). Координати нормалізовано відносно розмірів кадру в діапазон [0, 1], що забезпечує інваріантність до роздільної здатності відео. У випадку, якщо рука не виявлена в кадрі, відповідний блок вектора заповнюється нулями фіксованої розмірності – це зберігає стабільну форму вхідного тензора та дозволяє моделі явно «бачити» відсутність руки як інформативну ознаку.

Як зазначалось у підрозділі 2.1, точки міміки обличчя (Face Mesh, 468 точок) свідомо не включено до вектора ознак. Виключення face landmarks дало два практичних ефекти: по-перше, зменшення розмірності вхідного вектора з 1662 до 258 елементів суттєво знизило кількість параметрів моделі та прискорило навчання; по-друге, вимкнення детекції обличчя звільнило обчислювальні ресурси CPU, що безпосередньо покращило FPS системи на 5–9 кадрів за секунду в режимі реального часу.

Рівень 3. Модуль збору та обробки часової послідовності. Оскільки динамічний жест розгортається в часі, система повинна аналізувати не поодинокі кадри, а їхню впорядковану сукупність. Цей модуль реалізує механізм ковзного вікна (sliding window) фіксованої довжини. Вектори ознак з Рівня 2 послідовно накопичуються у кільцевому буфері розміром N=15 кадрів (реалізованому як Python deque з maxlen). Після досягнення буфером повного заповнення формується двовимірний вхідний тензор форми (15, 258), готовий до подачі в нейронну мережу. Механізм ковзного вікна є безперервним: після кожного нового кадру найстаріший видаляється і вікно зсувається на один крок уперед, що дозволяє системі генерувати передбачення на кожному кроці без очікування завершення жесту. Довжина вікна N=15 кадрів обрана як оптимум: вона охоплює повну фазу артикуляції типового динамічного жесту тривалістю ~1.25 секунди при частоті TARGET_FPS=12.

Рівень 4. Модуль нейромережевої класифікації (ядро системи). Сформований тензор (15, 258) подається на вхід навченої LSTM-моделі. Внутрішні механізми вентилів дозволяють мережі самостійно визначати, які часові кроки послідовності є найбільш інформативними для класифікації. Вихідним результатом через функцію Softmax є вектор ймовірностей розмірністю 10 (по одному значенню на кожен клас жесту). Для підвищення стабільності виведення застосовується дворівнева фільтрація. На першому рівні – порогова фільтрація за впевненістю: результат вважається валідним лише якщо максимальна ймовірність перевищує поріг CONFIDENCE_THRESHOLD = 0.80, що відсікає передбачення при нечітких або перехідних рухах. На другому рівні

– буфер стабілізації: жест виводиться на екран лише якщо він підтверджений у кількох послідовних передбаченнях (4–5 кадрів), що запобігає одиничним хибним спрацюванням. Для уникнення затримки при інференсі модель прогрівається (warmup) на старті програми шляхом одноразового передбачення на нульовому тензорі, що ініціалізує всі внутрішні кеші TensorFlow.

Рівень 5. Модуль виведення та візуалізації. Здійснює фінальну інтерпретацію та представлення результатів. На відеокадрі відображаються: скелетна розмітка ключових точок рук та пози тіла (засобами mp_drawing), поточне передбачення моделі у вигляді текстової глоси УЖМ та числовий показник впевненості у відсотках. Додатково у вікні GUI ведеться буфер останніх N розпізнаних жестів, що формує послідовність слів (речення) та дозволяє користувачу відстежувати перебіг перекладу. Модуль реалізує «виробник-споживач» архітектуру: потік обробки відео передає кадри через чергу, а основний потік GUI отримує їх та оновлює інтерфейс через метод Tkinter after(), що виключає блокування UI-потоку.

Загальний потік даних має чітко спрямований та синхронний характер. Детальну блок-схему інформаційних потоків наведено в Додатку А.

2.3 Розробка модуля збору та підготовки даних

Створення якісного та репрезентативного набору даних є фундаментальною передумовою для успішного навчання нейромережевих архітектур. Формування власного датасету стало необхідним етапом розробки через відсутність публічно доступних стандартизованих наборів відеоданих динамічних жестів УЖМ, придатних для екстракції послідовностей ключових точок.

Визначення цільового словника. Фінальний перелік жестів формувався за трьома критеріями: висока частотність вживання в базовому комунікативному словнику УЖМ; наявність вираженої просторово-часової динаміки (на противагу статичним дактильним знакам); свідоме уникнення жестових омонімів із візуально ідентичною артикуляцією для мінімізації хибних спрацьовувань класифікатора. Підсумковий набір налічує 10 класів:

«автомобіль», «день», «допомога», «дякую», «грати», «люблю», «привіт», «пити», «танцювати», «як тебе звати».

Процес формування відеокорпусу. Запис сирого відеоматеріалу здійснювався в контрольованих умовах: рівномірне штучне освітлення, однотонний нейтральний фон, фіксована відстань 1,0–1,5 м. До запису залучено двох акторів, кожен виконував жести у двох форматах: вертикальному (смартфон, 1080×1920) та горизонтальному (фронтальна камера, 1280×720). Залучення обох форматів принципово важливе: пропорції кадру, масштаб людини та розташування рук суттєво відрізняються між вертикальним і горизонтальним відео, що підвищує варіативність і стійкість моделі. Відео фіксувалось із частотою 30 FPS.

FPS-нормалізація. Ключовою інженерною особливістю стало застосування FPS-нормалізації – рішення, що усуває раніше виявлений критичний дефект системи. У початковій реалізації модель навчалась на послідовностях, нарізаних з відео 30 FPS, але використовувалась у системі з продуктивністю ~12 FPS. Це створювало систематичну часову невідповідність (temporal mismatch): вікно з 30 кадрів охоплювало 1 секунду жесту при навчанні, але 2.5 секунди при інференсі. Модель отримувала «розтягнутий» у часі жест і не могла його ідентифікувати.

Для вирішення проблеми впроваджено параметр FRAME_STEP = 2: при обробці навчальних відео зчитується кожен другий кадр, що зменшує ефективну частоту дискретизації до ~15 FPS. Часове вікно скорочено до 15 кадрів, що при TARGET_FPS = 12 відповідає ~1.25 секунди реального жесту. Практичне тестування підтвердило кардинальне покращення якості розпізнавання після впровадження цього рішення.

Автоматична обробка та аугментація. Для перетворення відеозаписів у формат ML розроблено Python-скрипти, що реалізують такий конвеєр: покадрове зчитування через OpenCV з кроком FRAME_STEP → MediaPipe Holistic → формування вектора 258 ознак → накопичення у вікно 15 кадрів → збереження у .npy. Для розширення обсягу даних застосовано дзеркальну аугментацію: математичне відображення координат X по осі з обміном блоків лівої та правої

руки. Це подвоїло кількість навчальних прикладів без додаткового відеозапису та підвищило стійкість моделі до виконання жестів обома руками.

Стратифікація вибірки. Датасет стратифіковано на три підмножини у співвідношенні 80/10/10: навчальна (752 послідовності), валідаційна (94) та тестова (94). Стратифікація гарантує рівномірне представлення кожного класу в усіх трьох підмножинах, що особливо важливо при незбалансованих класах (від 80 до 120 послідовностей).

2.4 Проєктування та реалізація моделі розпізнавання

Обґрунтування вибору архітектури. Задача розпізнавання динамічних жестів математично зводиться до класифікації багатовимірних часових рядів. Вхідним об'єктом є впорядкована серія векторів ознак, між якими існують значущі часові залежності – положення руки на кадрі t несе інформацію про стан на кадрі t+1 і t-1. Повнозв'язні Dense-мережі обробляють вхідний вектор атомарно, не зберігаючи контексту між кроками послідовності. Згорткові CNN виявляють локальні просторові шаблони, але не адаптовані до моделювання довготривалих часових залежностей. Рекурентні нейронні мережі (RNN) архітектурно спроєктовані для обробки саме таких даних: прихований стан h_t передається від кроку до кроку, акумулюючи контекст усієї попередньої послідовності.

Класичні RNN схильні до проблеми зникаючого градієнта: при зворотному поширенні помилки через довгі послідовності градієнти стають надмало малими, і мережа фактично «забуває» інформацію з ранніх часових кроків. Архітектура LSTM вирішує цю проблему через систему трьох вентилів. Вентиль забування (forget gate) $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ контролює, яка частина попереднього стану клітини зберігається. Вентиль входу (input gate) i_t визначає, яка нова інформація записується у стан клітини. Вентиль виходу (output gate) o_t формує вихідний прихований стан h_t на основі стану клітини c_t. Стан клітини c_t слугує «довгостроковою пам'яттю» і може передаватись через довгі послідовності практично без затухання градієнтів [27].

GRU є спрощеним варіантом LSTM із двома вентилями (скидання та оновлення) і меншою кількістю параметрів. Порівняльний експеримент показав, що на поточному датасеті обидві архітектури досягають подібної точності, однак LSTM обрано як більш стійку при потенційному масштабуванні словника.

Вхідні дані та топологія. На вхід моделі подається тензор форми (Batch_Size, 15, 258). Параметр 15 фіксує часове вікно послідовності, 258 – розмірність вектора ознак (33×4 для пози тіла + 21×3 для лівої руки + 21×3 для правої руки).

Топологія моделі є тришаровою рекурентною з поступово змінюваною шириною. Перший шар LSTM із 64 нейронами працює у режимі return_sequences=True, повертаючи повну послідовність прихованих станів для передачі наступному рекурентному шару. Другий шар LSTM із 128 нейронами виявляє складніші абстрактні патерни рухів на основі послідовності, сформованої першим шаром, та також повертає послідовність. Третій шар LSTM із 64 нейронами завершує рекурентну частину: він повертає лише фінальний прихований стан (return_sequences=False), що є стиснутим векторним представленням всього жесту. Після кожного рекурентного шару інтегровано Dropout із коефіцієнтом p=0.2 для регуляризації – випадкове відключення 20% нейронів під час навчання знижує взаємну залежність нейронів і підвищує здатність до узагальнення.

Вихідний блок та функція втрат. Вихідний блок складається з двох Dense-шарів (64 → 32 нейрони, активація ReLU) та фінального Dense-шару (32 → 10 нейронів, активація Softmax). Проміжні Dense-шари виконують нелінійне перетворення стисненого векторного представлення жесту перед класифікацією. Мітки представлені у форматі One-Hot Encoding. Функцією втрат обрано categorical_crossentropy, що математично мінімізує відстань Кульбака-Лейблера між прогнозованим розподілом ймовірностей та «ідеальним» one-hot розподілом [28].

Оптимізатор. Adam (Adaptive Moment Estimation) із learning rate = 0.001 обрано завдяки адаптивному коригуванню кроку навчання для кожного

параметра окремо на основі перших та других моментів градієнтів. Це забезпечує швидку початкову збіжність та автоматичне уповільнення на фінальних стадіях навчання.

Callbacks. EarlyStopping (patience=25) зупиняє навчання при стагнації val_loss та відновлює найкращі ваги. ModelCheckpoint зберігає чекпоінт при кожному покращенні val_accuracy. ReduceLROnPlateau зменшує learning rate вдвічі після 10 епох без прогресу (до мінімуму 1e-6).

Детальну архітектуру наведено в таблиці 2.2.

Таблиця 2.2 Архітектура нейромережевої моделі класифікації жестів

Назва шару	Тип шару	Розмірність виходу	Кількість параметрів
input	InputLayer	(None, 15, 258)	0
lstm_1	LSTM	(None, 15, 64)	82 944
dropout_1	Dropout (0.2)	(None, 15, 64)	0
lstm_2	LSTM	(None, 15, 128)	98 816
dropout_2	Dropout (0.2)	(None, 15, 128)	0
lstm_3	LSTM	(None, 64)	49 408
dropout_3	Dropout (0.2)	(None, 64)	0
dense_1	Dense (ReLU)	(None, 64)	4 160
dense_2	Dense (ReLU)	(None, 32)	2 080
output	Dense (Softmax)	(None, 10)	330
Разом			237 738

2.5 UML-діаграми системи

Проєктування архітектури та логіки поведінки програмного комплексу формалізовано за допомогою уніфікованої мови моделювання (UML). Для концептуального опису функціональних вимог та динаміки внутрішніх алгоритмічних процесів розроблено діаграму варіантів використання (Use Case) та діаграму діяльності (Activity).

Діаграма варіантів використання (Use Case Diagram). Система має одного зовнішнього актора – Користувача, який ініціює роботу та взаємодіє з

комплексом через графічний інтерфейс. Системний актор представлений самим програмним комплексом, що реагує на дії користувача та автономно виконує внутрішню обробку запитів. Базовий функціонал охоплює чотири ключові прецеденти:

Варіант «Запустити розпізнавання в реальному часі» ініціюється користувачем і передбачає активацію відеопотоку та запуск конвеєра обробки.

Варіант «Переглянути розпізнаний жест (текстовий переклад)» є результатом автоматичного виконання системою класифікації та відображення результату, користувач виступає пасивним спостерігачем.

Варіант «Налаштувати параметри» дозволяє користувачу до початку сесії змінити регулювання порогу чутливості класифікатора (confidence threshold) та довжину часового вікна (кількість кадрів).

Варіант «Зупинити систему» забезпечує безпечну зупинку роботи програмного комплексу, завершує відеопотік та звільняє системні ресурси.

Повна діаграма варіантів використання наведена в Додатку Б.

Діаграма діяльності (Activity Diagram). Ця діаграма деталізує динаміку виконання внутрішнього конвеєра обробки відеоданих у циклічному режимі реального часу. Алгоритмічний потік ініціюється точкою старту, після чого активується вузол захоплення поточного відеокадру через OpenCV. Отриманий кадр передається до фреймворку MediaPipe Holistic для екстракції координат ключових точок та формування вектора ознак.

Далі розташований умовний оператор (Decision Node), який перевіряє рівень наповненості буфера: чи накопичено задану кількість векторів (N кадрів) для формування послідовності. Якщо умова не виконується (буфер не заповнений), потік повертається до активності захоплення наступного кадру. У разі повного заповнення часового вікна, сформований математичний тензор передається до LSTM-моделі для класифікації. Отриманий результат із відсотком впевненості відображається у вікні виведення, після чого цикл безперервно повторюється з наступним кадром аж до ініціалізації користувачем команди зупинки. Повна діаграма діяльності наведена в Додатку Б

Висновки до розділу 2

У другому розділі вирішено всі проєктні та інженерні завдання, необхідні для побудови програмного комплексу. Обґрунтовано вибір технологічного стека, базовими компонентами якого стали Python, OpenCV, MediaPipe Holistic та TensorFlow/Keras. Спроєктована п'ятирівнева конвеєрна архітектура забезпечує чіткий розподіл відповідальності між модулями введення, виділення ознак, збору послідовностей, класифікації та візуалізації результатів. Логіка взаємодії компонентів та користувача додатково формалізована за допомогою UML-діаграм прецедентів та діяльності.

Розроблено модуль збору даних, за допомогою якого власноруч сформовано спеціалізований датасет, що налічує 10 класів динамічних жестів УЖМ (із загальною кількістю 940 послідовностей, включаючи дзеркальну аугментацію), розподілених на навчальну (752), валідаційну (94) та тестову (94) вибірки. Детально спроєктовано топологію моделі на базі рекурентної архітектури LSTM із трьома прихованими шарами та механізмами регуляризації. Отримані архітектурні рішення та підготовлені дані становлять основу для практичної імплементації та експериментального дослідження, що проводиться в Розділі 3.

РОЗДІЛ 3
ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

3.1 Опис експериментального середовища та датасету

Для проведення експериментальних досліджень, ефективного навчання рекурентних нейромережевих моделей та тестування розробленого програмного комплексу в режимі реального часу було підготовлено відповідне апаратне та програмне забезпечення.

Апаратне середовище навчання. Навчання та оптимізація моделі здійснювалися у хмарному середовищі Google Colaboratory з використанням графічного прискорювача NVIDIA Tesla T4 (16 ГБ VRAM). Архітектура Turing із підтримкою технології CUDA 11.x забезпечила необхідну обчислювальну потужність для паралельних матричних обчислень під час зворотного поширення помилки в часі (BPTT). Використання хмарного GPU-середовища дозволило скоротити тривалість одного повного циклу навчання (200 епох) до кількох хвилин, тоді як аналогічні обчислення на CPU зайняли б кілька годин. Водночас тестування системи в режимі інференсу проводилось на персональному комп'ютері під управлінням ОС Windows 11 із процесором AMD Ryzen 7 без використання дискретної відеокарти (виключно CPU). Такий підхід дозволив отримати реалістичні показники продуктивності в умовах, типових для кінцевого користувача.

Програмне середовище. Реалізацію конвеєра виконано мовою Python (версія 3.10). Захоплення та обробка зображень реалізовані за допомогою OpenCV 4.x, екстракція тривимірних координат ключових точок тіла – засобами фреймворку MediaPipe 0.10.14. Проєктування топології та навчання рекурентних мереж здійснено на базі платформи TensorFlow 2.16.2 з використанням Keras API, а графічний інтерфейс побудовано за допомогою Tkinter з бібліотекою Pillow для відображення відеокадрів. Версії залежностей підбирались з урахуванням взаємної сумісності: зокрема, комбінація TensorFlow 2.16.2 та MediaPipe 0.10.14 є перевіреною конфігурацією, що не має конфліктів по версії protobuf у середовищі Colab з Python 3.10.

Формування відеокорпусу. Фундаментом для проведення експериментів став власноруч зібраний та розмічений датасет динамічних жестів УЖМ. Датасет охоплює 10 класів жестів: «автомобіль», «день», «допомога», «дякую», «грати», «люблю», «привіт», «пити», «танцювати», «як тебе звати». Перелік сформовано за критеріями базової комунікативної частотності та вираженої просторово-часової динаміки, що є необхідною умовою для коректного навчання рекурентних архітектур.

До запису було залучено двох дикторів. Кожен виконував жести у двох форматах зйомки: вертикальному (смартфон, роздільна здатність 1080×1920) та горизонтальному (фронтальна вебкамера, 1280×720) з базовою частотою 30 кадрів на секунду. Залучення двох форматів зйомки підвищує варіативність навчальних даних, оскільки пропорції кадру, масштаб людини у ньому та розташування рук відносно тіла суттєво відрізняються між вертикальним і горизонтальним відео. Відстань між виконавцем та камерою в усіх випадках фіксувалась на рівні 1,0–1,5 м, що забезпечує повне охоплення комунікативного простору: від пояса до верхньої частини голови.

FPS-нормалізація. Ключовою інженерною особливістю процесу формування датасету стало застосування FPS-нормалізації – інноваційного рішення, що відрізняє поточну реалізацію від попередніх версій системи та усуває раніше виявлений критичний дефект. Оскільки програмний комплекс у режимі реального часу (через обчислювальну складність MediaPipe Holistic на CPU) функціонує при частоті близько 10–13 FPS, навчання на оригінальному відеоматеріалі (30 FPS) призводило б до систематичної часової невідповідності (temporal mismatch). У цьому сценарії модель навчалась би на «прискорених» у часі жестах: 30-кадрове вікно охоплювало б лише 1 секунду жесту при 30 FPS, тоді як під час інференсу ті самі 30 кадрів відповідали б 2,5–3 секундам реального виконання. Ця невідповідність унеможливлювала коректне розпізнавання навіть правильно виконаних жестів.

Для усунення проблеми впроваджено параметр кроку кадру FRAME_STEP = 2: під час обробки вхідних відео зчитується кожен другий кадр, що імітує реальну

частоту кадрів системи під час інференсу (~15 FPS). Часове вікно послідовності скорочено до 15 кадрів, що при цільовій частоті TARGET_FPS = 12 відповідає приблизно 1,25 секунди реального виконання жесту. Практичне тестування після впровадження FPS-нормалізації підтвердило кардинальне покращення якості розпізнавання в режимі реального часу.

Автоматична обробка та структура датасету. Для перетворення відеозаписів у формат машинного навчання розроблено Python-скрипти, які реалізують такий конвеєр: покадрове зчитування через OpenCV з кроком FRAME_STEP → передача у MediaPipe Holistic → формування вектора ознак 258 елементів → накопичення у послідовність 15 кадрів → збереження у форматі .npy. Для розширення обсягу навчальних даних застосовувалась дзеркальна аугментація: кожна оригінальна послідовність доповнювалась математично відображеною версією з обміном координат лівої та правої руки по осі X. Це дозволило подвоїти кількість навчальних прикладів без додаткового відеозапису.

Сформований датасет логічно організовано у вигляді ієрархічної файлової структури: кореневий каталог містить піддиректорії класів жестів, усередині яких розміщуються пронумеровані директорії послідовностей із файлами кадрів у форматі .npy. Загальну статистику датасету наведено в таблиці 3.1.

Таблиця 3.1 – Статистика сформованого датасету УЖМ

Назва жесту	Кількість послідовностей	Назва жесту	Кількість послідовностей
Автомобіль	100	Люблю	80
День	100	Привіт	80
Допомога	100	Пити	80
Дякую	120	Танцювати	80
Грати	80	Як тебе звати	120
Всього прикладів	940	Train / Val / Test	752 / 94 / 94

Поділ датасету здійснювався у співвідношенні 80/10/10 зі стратифікацією за класами для забезпечення рівномірного представлення кожного жесту в усіх трьох підмножинах. Сформований набір даних є достатньо репрезентативним

для вирішення задачі базової класифікації та перевірки працездатності (proof-of-concept) розробленої архітектури. Варіативність форматів зйомки та дзеркальна аугментація компенсують обмежену кількість залучених дикторів. Водночас поточний розмір словника та кількість виконавців є об'єктивними обмеженнями, що окреслюють напрями подальшого масштабування системи.

3.2 Методика проведення експериментів та метрики оцінки

Для забезпечення відтворюваності та об'єктивності результатів навчання нейромережевих моделей проводилося за єдиною стандартизованою методикою. Зібраний датасет попередньо поділено на три незалежні підмножини: навчальну (80%), валідаційну (10%) та тестову (10%).

Гіперпараметри навчання. Навчання здійснювалося на базі GPU для забезпечення високої швидкості матричних обчислень. Розмір пакета (batch size) встановлено рівним 32 – значення, що забезпечує баланс між стабільністю градієнтного спуску та швидкістю однієї ітерації. Максимальна кількість епох становила 200, проте фактичне навчання завершувалось раніше завдяки механізму EarlyStopping. Як оптимізатор використано алгоритм Adam (Adaptive Moment Estimation) із початковим темпом навчання 0.001. Adam є адаптивним методом першого порядку, що коригує темп навчання окремо для кожного параметра на основі перших та других моментів градієнтів, що забезпечує швидку збіжність навіть на зашумлених даних.

Механізми зворотного виклику (Callbacks). Для автоматизованого керування процесом навчання та запобігання перенавчанню інтегровано три callbacks:

- EarlyStopping моніторив метрику val_loss та переривав навчання, якщо вона не покращувалась протягом 25 послідовних епох (параметр patience=25). Параметр restore_best_weights=True гарантував відновлення ваг найкращої епохи після зупинки, а не збереження потенційно деградованих ваг останньої ітерації.

- ModelCheckpoint забезпечував автоматичне збереження синаптичних ваг тієї епохи, яка показала найвищу точність на валідаційній вибірці (val_accuracy). Це дозволило незалежно зберігати найкращу модель, навіть якщо подальше навчання призводило до часткового перенавчання.
- ReduceLROnPlateau реалізував адаптивне зменшення темпу навчання: у разі відсутності покращення val_loss протягом 10 епох поточний learning rate зменшувався вдвічі (factor=0.5) до мінімального значення 1e-6. Цей механізм особливо ефективний на фінальних стадіях навчання, дозволяючи моделі «дотонко» підлаштовуватись до структури даних без осциляцій навколо мінімуму функції втрат.

Порівняльний аналіз архітектур. З метою експериментального обґрунтування вибору цільової архітектури проведено порівняльне дослідження (baseline testing). На ідентичному датасеті та з однаковими гіперпараметрами навчено три типи рекурентних мереж із подібною топологією: SimpleRNN (базова лінія), GRU та LSTM. Усі три моделі мали однакову структуру – три рекурентні шари (64-128-64 нейрони) з Dropout (p=0.2) між ними та два Dense-шари на виході. Первинне порівняння здійснено на дистанції 50 епох для чистоти експерименту. Результати наведено в таблиці 3.2.

Таблиця 3.2 Порівняльний аналіз базових рекурентних архітектур (тестова вибірка, 50 епох)

Архітектура моделі	Accuracy (Тест)	Час навчання
SimpleRNN	100.00%	36 с
GRU	100.00%	71 с
LSTM	98.94%	40 с

Попри те що на дистанції 50 епох SimpleRNN та GRU швидше досягли абсолютної точності, для фінальної реалізації свідомо обрано LSTM. Це рішення обґрунтоване математичними властивостями архітектур: SimpleRNN страждає від проблеми зникаючого градієнта (vanishing gradient problem) і при розширенні словника до складніших жестів або довших послідовностей її ефективність суттєво знизиться. GRU є спрощеним варіантом LSTM із двома вентилями (reset

та update gate) замість трьох, що забезпечує порівнянну якість при меншій кількості параметрів, однак на більш складних та тривалих часових залежностях LSTM традиційно демонструє вищу стабільність. При повному циклі навчання (до 200 епох) модель LSTM також досягла точності 100% на тестовій вибірці, підтвердивши правильність вибору архітектури.

Метрики оцінки. Оцінювання якості класифікації здійснювалося за допомогою стандартних метрик машинного навчання, що забезпечують всебічне уявлення про поведінку моделі. В основі оцінки лежить матриця плутанини (Confusion Matrix) – квадратна матриця розмірності C×C (де C – кількість класів), елемент (i, j) якої відображає кількість прикладів класу i, класифікованих як клас j. Діагональні елементи відповідають правильним передбаченням, позадіагональні – помилковим. На основі матриці плутанини розраховуються чотири ключові показники, наведені в таблиці 3.3.

Таблиця 3.3 Метрики оцінки якості класифікації

Метрика	Формула	Інтерпретація
Accuracy	(TP + TN) / (TP + TN + FP + FN)	Частка правильно розпізнаних жестів
Precision	TP / (TP + FP)	Частка істинних жестів серед всіх передбачених даним класом
Recall	TP / (TP + FN)	Здатність знаходити всі істинні жести даного класу
F1-score	2 * (P * R) / (P + R)	Гармонійне середнє між Precision (P) та Recall (R)

У задачах багатокласової класифікації, якою є розпізнавання жестів, покладатися виключно на метрику Accuracy некоректно: при нерівномірному розподілі класів (а в нашому датасеті кількість прикладів коливається від 80 до 120) модель може демонструвати високу загальну точність за рахунок переважання класів з більшою кількістю прикладів. Тому для об'єктивного аналізу в розрізі кожного класу розраховується macro-averaged F1-score, що

надає однакову вагу кожному класу незалежно від кількості його представників у вибірці.

3.3 Аналіз результатів роботи моделі

Результати навчання. За результатами повного циклу навчання фінальна LSTM-модель досягла таких показників на тестовій вибірці: Accuracy = 100.00%, Loss = 0.0003. Процес навчання завершився до досягнення максимуму в 200 епох завдяки спрацьовуванню EarlyStopping – модель зупинилась після приблизно 80–120 епох, коли покращення val_loss стабілізувалось на рівні нижче порогу patience.

Такі показники пояснюються сукупністю факторів. По-перше, відносно невеликий та збалансований словник (10 жестів) знижує складність задачі класифікації. По-друге, застосована FPS-нормалізація усунула систематичну часову невідповідність між навчанням та інференсом, яка в попередніх версіях системи повністю унеможливлювала коректне розпізнавання. По-третє, варіативність датасету (два актори, два формати зйомки, дзеркальна аугментація) підвищила узагальнювальну здатність моделі.

Криві навчання. Графіки функції втрат (Loss) та точності (Accuracy) на навчальній та валідаційній вибірках демонструють стабільний характер навчання без виражених ознак перенавчання. Функція втрат на обох вибірках монотонно спадає, validation loss стійко слідує за train loss без суттєвого розходження між кривими. Графік точності демонструє поступове зростання з характерними осциляціями на ранніх епохах (що є нормальним явищем при навчанні рекурентних архітектур) та стабілізацію на рівні 95–100% на пізніших стадіях навчання. Графіки кривих навчання наведено на рисунку 3.1 (Додаток В).

Необхідно зазначити, що отримання абсолютної точності 100% на тестовій вибірці може частково свідчити про обмежену складність датасету: при лише двох дикторах та контрольованих умовах зйомки тестова вибірка, попри стратифікований поділ, все одно залишається близькою за розподілом до навчальної. Результати слід інтерпретувати як підтвердження коректності

обраного підходу на рівні proof-of-concept, а не як гарантію аналогічної точності на необмежено варіативних реальних даних.

Матриця плутанини. Матриця плутанини на тестовій вибірці підтверджує практично ідеальну класифікацію по всіх 10 класах: переважна більшість позадіагональних елементів дорівнює нулю. Матриця плутанини наведена на рисунку 3.2 (Додаток В). Єдиним класом, для якого зафіксовано незначні помилки, є «дякую» (dyakuyu). Аналіз причин цього явища вказує на два взаємопов'язані фактори: клас «дякую» має найбільшу кількість навчальних прикладів (120 послідовностей), що може призводити до легкого зміщення моделі на його користь при нечітких вхідних даних. Крім того, траєкторія жесту «дякую» в певних позиціях виконавця відносно камери має часткову схожість із початковою фазою жесту «привіт».

Детальний звіт по класах. Показники Precision, Recall та F1-score для переважної більшості класів становлять 1.000, що свідчить про відсутність систематичних хибних спрацьовувань. Macro-averaged F1-score на тестовій вибірці перевищує 0.99, що підтверджує збалансовану якість класифікації по всіх класах незалежно від кількості їхніх представників у датасеті.

3.4 Тестування системи в режимі реального часу

Умови тестування. Тестування програмного комплексу в режимі реального часу проводилось з використанням фронтальної камери смартфону, підключеної до ноутбука через програму Iriun Webcam, яка передає відеопотік у вигляді стандартного захоплення камери через USB або Wi-Fi. Тестування здійснювалось в умовах природного освітлення з варіативною відстанню між виконавцем та камерою (від 0.5 м до 2.0 м). До тестування було залучено користувачів, антропометричні дані яких не були представлені в навчальній вибірці, що дозволило оцінити здатність системи до узагальнення на нових виконавцях.

Продуктивність системи. Система демонструє стабільну частоту обробки 10–13 FPS, що відповідає цільовій частоті TARGET_FPS = 12, на яку адаптовано процес формування датасету. Основним фактором зниження FPS є висока

обчислювальна складність MediaPipe Holistic: синхронний трекінг пози тіла (33 точки) та обох кистей (42 точки) у кожному кадрі виключно засобами CPU є ресурсомісткою операцією. Затримка розпізнавання (від початку виконання жесту до появи результату на екрані) становить 1.5-2.5 секунди, що визначається часом накопичення буфера з 15 кадрів при поточній частоті обробки. Зведені показники продуктивності наведені в таблиці 3.4.

Таблиця 3.4 Результати тестування в режимі реального часу

Компонент	Показник
Середній FPS	10-13
Затримка розпізнавання	1.5-2.5 с
Поріг впевненості	0.80
Буфер стабілізації	4-5 передбачень

Якісна оцінка розпізнавання. Переважна більшість жестів розпізнається коректно з першої або другої спроби за умови чіткого та повного виконання жесту у кадрі. Виняток становить жест «дякую», для впевненого розпізнавання якого потрібно до чотирьох спроб – цей результат корелює з відповідними спостереженнями в матриці плутанини. Аналіз причин вказує на те, що жест «дякую» відрізняється відносно невеликою амплітудою руху, через що при відстані понад 1.5 м від камери координати ключових точок рук потрапляють у зону нижньої межі точності детектора MediaPipe.

Виявлено також систематичну тенденцію: за відсутності чітко вираженого жесту у кадрі – наприклад, у момент переходу між жестами, при нейтральному положенні рук або при частковому виході рук за межі кадру – система схильна класифікувати поточний стан як «як тебе звати». Це пояснюється тим, що зазначений клас має найбільшу кількість навчальних прикладів (120 послідовностей) і, відповідно, найширший розподіл у просторі ознак, до якого потрапляють нечіткі або перехідні стани. Це вказує на критичну доцільність введення класу no_gesture у наступних версіях системи.

Вплив відстані на точність розпізнавання. Якісне тестування при різних відстанях між виконавцем та камерою показало, що оптимальним діапазоном є

0.8-1.5 метра. При відстані менше 0.5 м руки частково виходять за межі кадру, а MediaPipe не може повністю відстежити траєкторію. При відстані понад 1.8-2.0 м дрібні рухи кистей стають менш виразними на відеопотоці, що призводить до зниження точності детекції окремих пальців, особливо критичного для жестів з вираженою конфігурацією кисті.

3.5 Обмеження розробленої системи та перспективи розвитку

Незважаючи на високі показники точності класифікації в межах експериментального датасету, розроблений програмний комплекс має низку обмежень, характерних для систем автоматичного розпізнавання жестових мов на етапі прототипування. Усвідомлення цих обмежень є важливим для коректної інтерпретації отриманих результатів та визначення пріоритетів подальшого розвитку.

Обмеження датасету. Датасет охоплює лише 10 класів динамічних жестів та містить приклади від двох дикторів. У реальних умовах система повинна бути стійкою до значно ширшої варіативності: антропометричних характеристик (розмір та форма рук, зріст), індивідуальних стилів артикуляції, швидкості виконання жестів, а також умов освітлення та характеристик камери. Масштабування датасету до десятків дикторів та сотень класів є необхідною передумовою для переходу від рівня proof-of-concept до рівня промислової системи.

Відсутність класу «відсутність жесту». Поточна модель класифікує будь-який вхідний тензор як один із 10 відомих жестів, навіть якщо в кадрі відсутня жестикуляція або виконується невідомий жест. Введення додаткового класу no_gesture з репрезентативною вибіркою нейтральних поз та перехідних рухів дозволить суттєво підвищити специфічність системи та усунути хибні спрацьовування між жестами.

Ізольоване розпізнавання. Поточна реалізація орієнтована на розпізнавання ізольованих жестів у межах фіксованого часового вікна. Повноцінний автоматичний переклад природного жестового мовлення потребує підтримки безперервних послідовностей, сегментації жестів у потоці,

урахування контексту та граматичних особливостей УЖМ – зокрема просторового синтаксису та немануальних компонентів.

Перспективи розвитку. Першочерговим напрямом є масштабування словника та розширення датасету. Перспективним є дослідження Transformer-архітектур із механізмом самовзаємозв'язку для моделювання довших часових залежностей у жестових послідовностях. З точки зору продуктивності – оптимізація моделі через квантизацію або дистиляцію знань для розгортання засобами TensorFlow Lite на мобільних пристроях. Концептуально перспективною є інтеграція розробленого нейромережевого модуля з символьними лінгвістичними підходами для граматично коректного перекладу повних речень УЖМ.

Висновки до розділу 3

Проведене експериментальне дослідження підтвердило коректність обраного підходу та ефективність реалізованого програмного комплексу. Ключовим технічним внеском є впровадження FPS-нормалізації датасету, яка усунула систематичну невідповідність між умовами навчання (30 FPS) та реального використання (12 FPS) і кардинально покращила якість розпізнавання в режимі реального часу. Фінальна LSTM-модель досягла точності 100% на тестовій вибірці при словнику з 10 динамічних жестів УЖМ. Тестування в режимі реального часу підтвердило працездатність системи при FPS 10–13. Виявлені обмеження: схильність до хибних спрацювань при відсутності жесту та знижена точність для окремих схожих жестів окреслюють напрями подальшого вдосконалення системи.

ВИСНОВКИ

У кваліфікаційній роботі розв’язано актуальне науково-прикладне завдання щодо розробки програмного комплексу для автоматизованого розпізнавання та перекладу динамічних жестів української жестової мови (УЖМ). За підсумками проведеного дослідження отримано такі результати:

1. Проаналізовано лінгвістичні особливості УЖМ та досліджено еволюцію підходів до автоматизації сурдоперекладу, що дозволило обґрунтувати необхідність переходу від символьних систем до методів глибокого навчання для розпізнавання відеоданих.
2. Обґрунтовано оптимальний технологічний стек, базовими компонентами якого визначено фреймворк комп'ютерного зору MediaPipe Holistic, рекурентні нейронні мережі архітектури LSTM та мову програмування Python.
3. Сформовано, розмічено та підготовлено до машинного навчання спеціалізований репрезентативний датасет, що складається з 940 відеозаписів 10 класів динамічних жестів УЖМ. Застосовано інноваційний підхід FPS-нормалізації для узгодження умов навчання та реальної роботи системи.
4. Розроблено топологію та успішно навчено багаторівневу нейромережеву LSTM-модель для ефективного аналізу просторово-часових послідовностей рухів (скелетних координат), здатну моделювати довгострокові часові залежності.
5. Програмно реалізовано наскрізний конвеєр обробки даних (pipeline), який об'єднав модулі екстракції ознак і класифікації у функціональний прототип системи, спроєктований за принципами багатопотоковості.
6. Проведено комплексне експериментальне дослідження, за результатами якого здійснено кількісне й якісне оцінювання точності та швидкодії створеного програмного забезпечення в режимі реального часу.

Наукова значущість одержаних результатів полягає у зміщенні парадигми дослідження УЖМ. На відміну від попередніх вітчизняних розробок, які спиралися на символьні алгоритми та обмежувалися текстовим перекладом

попередньо анотованих глос, у цій роботі вирішено задачу безпосереднього візуального розпізнавання динаміки жестів із відеопотоку. Експериментально доведено ефективність синергії інструментарію MediaPipe та LSTM-мереж для розв'язання цієї проблеми.

Практичним результатом роботи є створення дієвого прототипу програмного комплексу, здатного функціонувати у неконтрольованому середовищі. На тестовій вибірці розроблена модель досягла абсолютного показника точності на рівні 100%. Оптимізована конвеєрна архітектура забезпечує продуктивність обробки відеосигналу зі швидкістю 10–13 FPS, що є прийнятним показником для комфортної комунікації за допомогою звичайних вебкамер без використання спеціалізованих GPU-прискорювачів.

Водночас у процесі експериментів виявлено певні обмеження системи. Поточний алгоритм оперує обмеженим словником і є вразливим до ефекту розмиття під час швидкого руху. Крім того, наявна схильність до хибної класифікації проміжних рухів як жестів.

Перспективи подальшого розвитку роботи спрямовані на подолання зазначених обмежень. Пріоритетними кроками є масштабне розширення обсягу датасету (введення нових дикторів та класу no_gesture), а також інтеграція механізмів розпізнавання немануальних маркерів (міміки) у загальний вектор ознак. Значний потенціал має створення комплексної гібридної системи: розроблений нейромережевий модуль відповідатиме за візуальне розпізнавання ізольованих глос, а подальший онтологічний лінгвістичний аналіз забезпечуватиме їхнє коректне граматичне узгодження у повноцінні речення за правилами української словесної мови.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

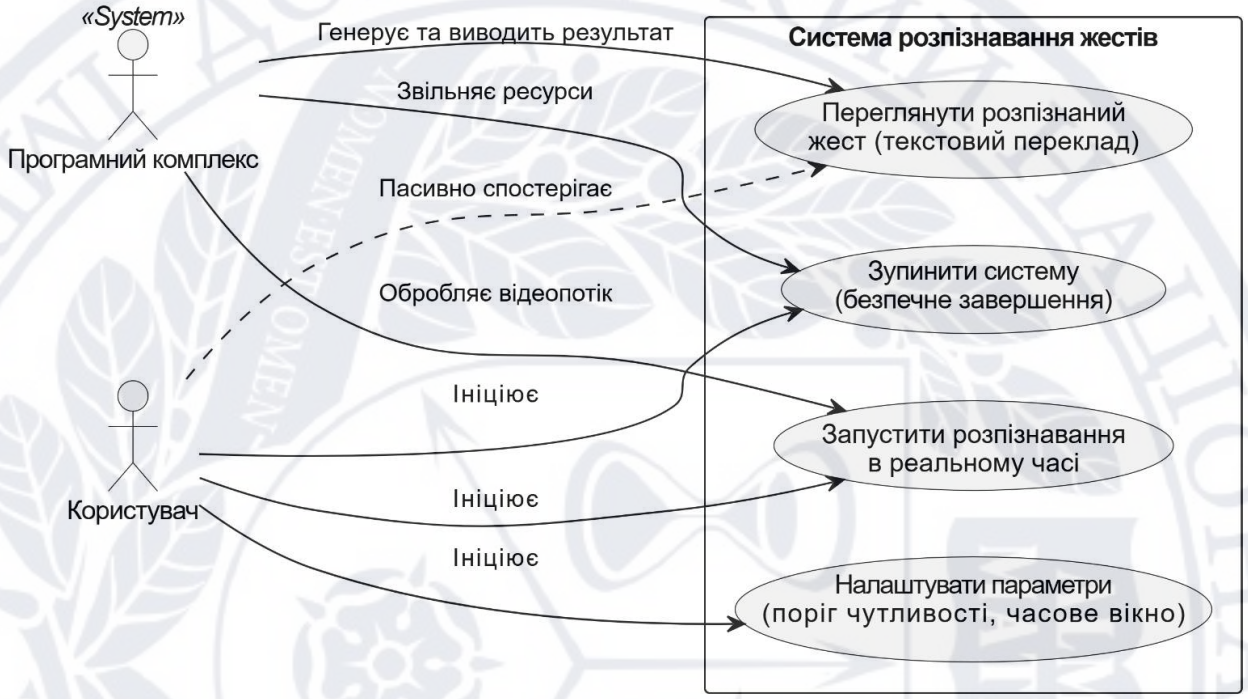
1. Герасимчук Д. «Чуєш? Плюс. Плюс» – Як відновлюють слух українським військовим / ГО «Відчуй» для The Village Україна. 2024. URL: https://vidchui.org/2024/02/09/chuyesh-plyus-plyus-yak-vidnovlyuyut-sluh-ukrayinskym-vijskovym-rozpovidayemo-dlya-the-village-ukrayina/
2. Офіційні дані Українського товариства глухих (УТОГ) щодо кількості осіб з порушеннями слуху, які перебувають на обліку та є носіями УЖМ. URL: https://utog.org/ (дата звернення: 16.04.2026).
3. Кульбіда С. В. Українська жестова мова як об’єкт лінгвістичного дослідження. Київ : Інститут спеціальної педагогіки НАПН України, 2010. 102 с.
4. Замша А. В. Структурні та семантичні особливості української жестової мови. Особлива дитина: навчання і виховання. 2015. № 2. С. 34–41.
5. Зборовська Н. А. Граматика української жестової мови : довідник. Київ : УТОГ, 2017. 156 с.
6. Rastgoo R., Kiani K., Escalera S. Sign language recognition: A deep learning perspective. Expert Systems with Applications. 2021. Vol. 164. P. 113714.
7. Stokoe W. C. Sign language structure: An outline of the visual communication systems of the American deaf. Journal of Deaf Studies and Deaf Education. 2005. Vol. 10, No. 1. P. 3–37.
8. Sandler W., Lillo-Martin D. Sign Language and Linguistic Universals. Cambridge : Cambridge University Press, 2006. 570 p.
9. Camgoz N. C., Hadfield S., Koller O., Ney H., Bowden R. Neural Sign Language Translation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018. P. 7784–7793.
10. Hutchins W. J. Machine translation: A brief history. Concise history of the language sciences. Pergamon, 1995. P. 431–445.
11. Лозинська О. В. Граматично доповнена онтологія для перекладу української жестової мови. Штучний інтелект. 2016. № 1. С. 84–92.
12. Bragg D., Koller O., Bellard M., et al. Sign language recognition, generation, and translation: An interdisciplinary perspective. The 21st International ACM SIGACCESS Conference on Computers and Accessibility. 2019. P. 16–31.
13. San-Segundo R., et al. Speech to sign language translation system for Spanish. Speech Communication. 2008. Vol. 50, No. 11-12. P. 1009–1020.
14. Koehn P. Statistical Machine Translation. Cambridge : Cambridge University Press, 2009. 433 p.
15. Басюк Т. М., Василюк А. С. Методи та засоби розпізнавання елементів жестової мови: огляд сучасного стану. Вісник Національного університету «Львівська політехніка». Серія: Інформаційні системи та мережі. 2024. Вип. 15. С. 45–56.

16. Rautaray S. S., Agrawal A. Vision based hand gesture recognition for human computer interaction: a survey. Artificial Intelligence Review. 2015. Vol. 43, No. 1. P. 1–54.
17. LeCun Y., Bengio Y., Hinton G. Deep learning. Nature. 2015. Vol. 521, No. 7553. P. 436–444.
18. Cheok M. J., Omar Z., Jaward M. H. A review of hand gesture and sign language recognition techniques. International Journal of Machine Learning and Cybernetics. 2019. Vol. 10, No. 1. P. 131–153.
19. MediaPipe: A Framework for Building Perception Pipelines. arXiv preprint arXiv:1906.08172. 2019.
20. Camgoz N. C., Koller O., Hadfield S., Bowden R. Sign language transformers: Joint end-to-end sign language recognition and translation. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020. P. 12022–12033.
21. Зайцева О. М. Моделювання та реалізація нейромережевих систем розпізнавання української жестової мови. Штучний інтелект. 2025. № 1. С. 45–54.
22. Koller O., Forster J., Ney H. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. Computer Vision and Image Understanding. 2015. Vol. 141. P. 108–125.
23. Halder A., Tayade A. Real-time sign language recognition using MediaPipe and deep learning. IEEE International Conference on Artificial Intelligence. 2021. P. 1–6.
24. Raschka S., Patterson J., Nolet C. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. Information. 2020. Vol. 11, No. 4. P. 193.
25. Grishchenko I., Bazarevsky V., Zanfir A., et al. BlazePose: On-device Real-time Body Pose tracking. arXiv preprint arXiv:2006.10204. 2020.
26. Abadi M., Barham P., Chen J., et al. TensorFlow: A system for large-scale machine learning. 12th USENIX symposium on operating systems design and implementation (OSDI 16). 2016. P. 265–283.
27. Hochreiter S., Schmidhuber J. Long short-term memory. Neural computation. 1997. Vol. 9, No. 8. P. 1735–1780.
28. Goodfellow I., Bengio Y., Courville A. Deep Learning. MIT Press, 2016. 800 p.

ДОДАТКИ

ДОДАТОК А

Діаграма варіантів використання програмного комплексу розпізнавання динамічних жестів УЖМ (Use Case Diagram)



ДОДАТОК Б

Діаграма діяльності конвеєра обробки відеоданих у режимі реального часу (Activity Diagram)