Міністерство освіти і науки України
Державний університет «Одеська політехніка»

Інститут штучного інтелекту та робототехніки
Кафедра Комп'ютерних систем

Міщенко Микита Дмитрович

студента групи УК-162

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

Дослідження розпізнавання жестів української мови у режимі реального часу

Спеціальність:
123 Комп'ютерна інженерія

Спеціалізація:
Спеціалізовані комп'ютерні системи

Керівник:

Нестерюк Олександр Геннадійович,

Кандидат техн. наук, доцент

Одеса – 2021

ЗМІСТ

Міністерство освіти і науки України
Державний університет «Одеська політехніка»
Інститут штучного інтелекту та робототехніки
Кафедра Комп'ютерних систем

Рівень вищої освіти: другий (магістерський)
Спеціальність: 123 – Комп'ютерна інженерія
Спеціалізація: Спеціалізовані комп'ютерні системи

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____
(підпис)
к.т.н., проф. Ситніков В.С.

« » _____ 20__ р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Міщенко Микиті Дмитровичу, групи УК-162

1. Тема роботи: Дослідження розпізнавання жестів української мови у режимі реального часу

Керівник роботи: Нестерюк Олександр Геннадійович, кандидат техн. наук

затверджені наказом вищого ректора від 1 жовтня 2021 року № 346-в

2. Зміст роботи: Вступ. Аналітичний огляд. Дослідження та проектування згорткової нейронної мережі для розпізнавання жестів української мови. Реалізація системи для розпізнавання жестів алфавіту української мови. Висновки. Перелік посилань. Додатки.

3. Перелік ілюстративного матеріалу: Комп'ютерна презентація за темою роботи

4. Дата видачі завдання 01.09.2021

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналітичний огляд	21.09.21	виконано
2	Дослідження та проектування згорткової нейронної мережі для розпізнавання жестів української мови	25.10.21	виконано
3	Реалізація системи для розпізнавання жестів алфавіту української мови	21.11.21	виконано
4	Оформлення пояснювальної записки	15.12.21	виконано
5	Нормоконтроль		
6	Попередній захист		
7	Захист		

Здобувач вищої освіти _____ Міщенко М.Д.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Нестерюк О.Г.
(підпис) (прізвище та ініціали)

ВСТУП

Актуальність теми: Актуальність роботи полягає в розробці нейронної мережі, яка буде мати функціонал розпізнавання жестів алфавіту української мови в режимі реального часу, що забезпечує розуміння мови людей з вадами апарату мовлення та слуху. На даний момент в Україні налічується близько 50 тисяч людей, які мають порушення слуху або мовлення та спілкуються між собою за допомогою мови жестів. За оцінками ВООЗ, до 2050 року майже 2,5 млрд людей страждатимуть від проблем зі слухом тією чи іншою мірою, з них 700 млн буде потрібно реабілітація через інвалідизуючу втрату слуху. Незважаючи на стрімке поширення цифрових форм комунікації, люди з порушеннями слуху досі зазнають певних бар'єрів при живому спілкуванні [1].

Отже, можна виявити тенденцію до того, що з кожним роком підвищується необхідність створення програмного забезпечення, яке б допомогло налагоджувати комунікацію людей з вадами апарату мовлення або слуху з людьми, яких цих вад не мають.

Зв'язок магістерської роботи з науковими програмами, планами, темами: Дане дослідження проводилося в рамках дипломної роботи студентом Кафедри комп'ютерних систем Інституту штучного інтелекту та робототехніки Державного університету «Одеська політехніка».

Мета і задачі дослідження: Метою даного дослідження є розробка системи розпізнавання жестів алфавіту української мови навченої за допомогою згорткової нейронної мережі на основі існуючих аналогів згідно виділення кращих сторін аналогів. Для виконання даної мети необхідно реалізувати такі завдання: аналіз аналогів та вимог до нейронної мережі, проектування та вдосконалення вже існуючих аналогів, а саме підвищення ефективності розпізнавання навченого класифікатору за допомогою глибинного навчання нейронною мережею та ручною

вибіркою гіперпараметрів системи, а також тестування нейронної мережі на реальному наборі текстових даних.

Об'єкт дослідження: Об’єктом дослідження є процес розпізнавання жестів алфавіту української мови в режимі реального часу з функцією виводу тексту на екран комп’ютеру.

Предмет дослідження: Предметом дослідження є методи машинного навчання для розпізнавання жестів.

Методи дослідження: Методами, які були використані при проведенні даного дослідження, є аналітичний підхід та синтез даних для виділення слабких та сильних сторін вже існуючих аналогів, а також для вибору набору даних, на основі якого буде проводитися машинне навчання нейронної мережі. Також будуть використані методи статичного аналізу, експеримент та опис для проведення тестування нейронної мережі та забезпечення відповідності результату розпізнавання заявленим технічним характеристикам та описання її роботи для користувачів.

Наукова новизна: В ході проведення дослідження було вдосконалено підхід в розпізнаванні жестів в режимі реального часу, що може бути використано як в програмному забезпеченні для спілкування, розширюючи його функціонал для розуміння людей з вадами апарату мовлення або слуху. Розроблена нейронна мережа може бути розглянута в інших програмних рішеннях з подальшим поліпшенням.

Практичне значення одержаних результатів: Отримані результати є практичною пропозицією поліпшення вже існуючих додатків для розпізнавання жестів, якими передаються символи українського алфавіту в режимі реального часу, які створені за допомогою нейронної мережі. Після проведення дослідження та розробки нейронної мережі, її можна використовувати для розпізнавання жестів, або інтегрувати в програмне забезпечення для подальшого доопрацювання та розширення функціоналу.

1. АНАЛІТИЧНИЙ ОГЛЯД

1.1 Вибір алгоритму для розпізнавання жестів та аналіз його призначення

Вибір алгоритму для розпізнавання жестів, які означають символи алфавіту української мови є невід'ємною частиною створення нейронної мережі. Від вибору алгоритму залежить якість, швидкодія та відмовостійкість реалізованої системи, тому на даному етапі якомога важливо розглянути обраний алгоритм та його переваги.

Для того, щоб забезпечити високу швидкодію та відмовостійкість нейронної мережі, що розробляється, було запропоновано алгоритм з основою у вигляді сітки з нейронною мережею, яка має невелику архітектуру. Основою згорткової нейронної мережі є шари згортки [3]. Кожному шару нейронної мережі належать фільтри для кожного каналу, які мають обробляти попередній шар не повністю, а по частинами, а саме за допомогою підсумовування фрагментів у вигляді матриць. Наприкінці шару згортки завжди стоїть функція активації [4]. Для того, щоб передати інформацію на інший шар, необхідно використовувати функцію активації, яка перетворює обчислену інформацію для передачі у вигляді, відповідному стандартам наступного шару. Значення на виході залежить від функції активації і може бути як дійсним, і цілим [5]. Це є показником того, наскільки активований нейрон поточного шару. Оскільки кожному фільтру згортки відповідає одна карта ознак, це дозволяє нейронної мережі навчитися виділяти ознаки незалежно від їх розташування у вхідному зображенні.

Пулінг можна розтлумачити наступним чином: якщо на попередній операції згортки були виявлені ознаки, що містять деяку важливу інформацію, щодо розпізнавання, то для подальшого процесу розпізнавання системі настільки докладне зображення вже не потрібно, і воно зменшується, тобто ущільняється до менш докладного розміру. До того ж, фільтрація вже непотрібних деталей зменшує

перенавчання. Шар пулінга, як виявлено, краще вставляти після згорткового шару перед наступним шаром згортки. За рахунок шару пулінга мережа стає більш стійкою до змін вхідного зображення, наприклад, його зсувів чи шуму який може міститися в самому зображені. Також зменшується розмірність наступних верств [6]. Повнозв'язковий шар (багатошаровий перцептрон) – це прихований шар, з'єднаний з усіма нейронами попереднього шару.

Останнім шаром багатошарового перцептрону є один або кілька нейронів, кількість яких дорівнює кількості класів. Або простіше кажучи, на вхід усієї згорткової нейронної мережі подається зображення, а на виході мережа видає клас, до якого це зображення відноситься [7]. Згорткові нейронні мережі можуть забезпечити часткову стійкість до різних змін з якими система може зіткнутися в реалії, наприклад: проблема масштабу, зсуву, поворотів чи зміни ракурсу та інших спотворень вхідного сигналу (зображення). Загальну топологію зображено на рисунку 1.1.

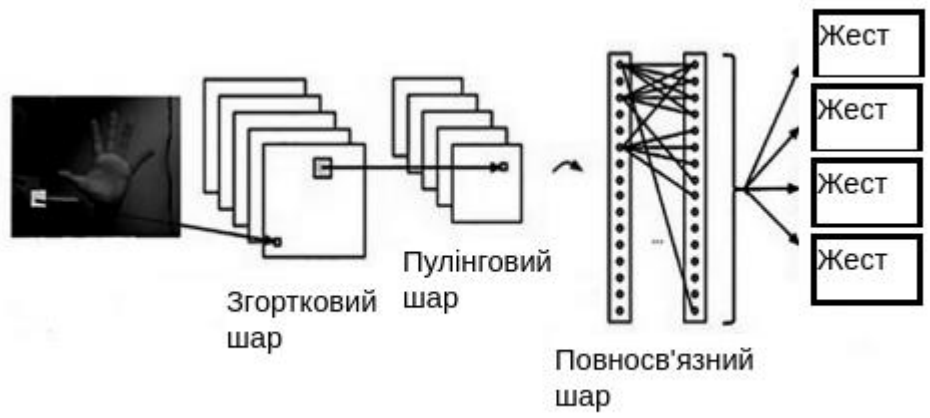

Рисунок 1.1 – Загальна структура нейронної мережі

Для того, щоб визначити нейронну мережу PyTorch, яка містить три згорткових шари, за якими будуть слідувати три повнозв’язкових шари, необхідно реалізувати наступний клас:

```
class Net(nn.Module):
    def __init__(self):
```

```
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 6, 3)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(6, 6, 3)
        self.conv3 = nn.Conv2d(6, 16, 3)
        self.fc1 = nn.Linear(16 * 6 * 6, 120)
        self.fc2 = nn.Linear(120, 48)
        self.fc3 = nn.Linear(48, 34)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = self.pool(F.relu(self.conv2(x)))
        x = self.pool(F.relu(self.conv3(x)))
        x = x.view(-1, 16 * 6 * 6)
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x
```

Після реалізації даного класу необхідно ініціалізувати нейронну мережу, визначити функцію втрат та параметри оптимізації, додавши наступний програмний код:

```
def main():
    net = Net().float()
    criterion = nn.CrossEntropyLoss()
    opt = optim.SGD(net.parameters(), lr=0.01, momentum=0.9)

    trainloader, _ = get_train_test_loaders()
    for epoch in range(12):  # loop over the dataset multiple times
      train(net, criterion, opt, trainloader, epoch)
    torch.save(net.state_dict(), "checkpoint.pth")
```

Отже, завдяки даному програмному коду відбувається навчання дванадцяти періодів, а також визначається період ітерації навчання, в процесі якого кожен зразок використовується один і тільки один раз. Наприкінці основної функції параметри моделі зберігаються в файл.

1.2 Вибір підходу для реалізації нейронної мережі

При виборі підходу потрібно зосередитися на основній меті, задля якої розробляється нейронна мережа, знайти підходи та задачі, які вони вирішують, після чого проаналізувати переваги та недоліки кожного підходу.

Для реалізації нейронної мережі з ціллю розпізнавання жестів, які позначають символи українського алфавіту теоретично можна використовувати такі алгоритми як K-найближчих сусідів, прихованої Марківської моделі, метод опорних векторів, метод ансамблю, а також метод глибинного навчання.

Для розпізнавання жестів активно використовується алгоритм K-найближчих сусідів, який неодмінно треба взяти до уваги при оцінці підходу. Даний алгоритм класифікує вхідні дані по відстані.

Іншим алгоритмом, який використовується при розпізнаванні жестів, є алгоритм прихованої Марківської моделі, який використовує алгоритм максимізації очікувань та розширює його.

Метод опорних векторів є класифікатором, який використовується для невеликих наборів даних, так як кількість векторів підтримки звеличується лінійно відповідно до розміру навчальної вибірки.

Метод ансамблю є методом, який широко використовується та не потребує великої кількості даних для навчання.

Метод глибинного навчання є однією з найбільш розвиненою технологію при розпізнаванні жестів у сучасному світі. Такі нейронні мережі використовуються при розпізнаванні жестів, обличь, зображень та мовлення. Найбільш популярними

серед галузі інформатики, а саме машинного навчання, є згорткові нейронні мережі, які мають тенденцію до роботи з глибинними архітектурами.

Порівняння різних підходів, які можуть бути використані при розпізнаванні жестів, зображено в таблиці 1.1.

Таблиця 1.1 – Переваги та недоліки різних підходів до розпізнавання жестів

Назва підходу або алгоритму	***Переваги***	***Недоліки***
K-найближчих сусідів	Простота	Недостатня точність, висока вірогідність помилки при виборі K
Прихованої Марківської моделі	Прозорість моделі, гнучкість	Необхідність регуляції великої кількості параметрів
Опорних векторів	Можуть застосовуватися різні функції ядра	З кількістю вхідних даних звеличується кількість векторів
Ансамблю	Не потрібна велика кількість даних для навчання	Легко перенавчити, чутливість до шумів та викидів
Глибинне навчання	Гнучка, висока точність навчання, перевершує інші методи машинного навчання	Потрібна велика кількість даних для навчання

Отже, посилаючись на таблицю 1.1, можна зробити висновок, що найкращим підходом для розробки нейронної мережі з ціллю розпізнавання жестів буде використання глибинного навчання, а саме згорткових нейронних мереж.

1.3 Вибір бібліотеки для реалізації нейронної мережі

TensorFlow є бібліотекою, яка створена компанією Google та написана на мовах програмування Python та C++. Таку бібліотеку зазвичай використовують для складних проектів та створення багатошарових нейронних мереж. Серед її переваг можна виділити багату документацію, велику спільноту, зручність для роботи з мобільними пристроями. Якщо розглядати недоліки, потрібно зазначити, що дана бібліотека програє по швидкості та завдяки своїй низькорівневості потребує багато шаблонного коду, що також робить процес написання програмного коду більш складним і тривалим. Єдина мова програмування, яку підтримує дана бібліотека, - Python.

Бібліотека Keras була створена з ціллю прискорення експериментів та зазвичай запускається разом із іншими бібліотеками. Дана бібліотека відома своєю швидкодією та простотою, має конфігураційні модулі та зрозумілий інтерфейс, а також може бути використаний для побудови моделей глибокого навчання, однак дана бібліотека може діяти тільки в рамках Tensorflow, що є недоліком, так як це означає, що дана бібліотека забезпечує менше функціоналу і це може стати обмеженням при побудові високорівневих моделей.

Наступною бібліотекою, яку необхідно розглянути, є бібліотека PyTorch, яка розроблена компанією Facebook та використовується на різних комерційних проектах, наприклад, в соціальній мережі Twitter. Основним завданням PyTorch є навчання моделей швидко та ефективно, тому дана бібліотека є вибором більшості розробників. Серед переваг важливо відзначити простий процес створення моделі завдяки архітектурі фреймворку, підтримка популярних інструментів для дебагу, підтримка паралелізму даних, наявність великої кількості попередньо навчених моделей та готових модульних частин. Серед недоліків можна виділити недостатню кількість інтерфейсів для моніторингу та візуалізації, але цей недолік компенсується наявністю підключення до TensorBoard.

Отже, проаналізувавши всі бібліотеки, можна зробити висновок, що бібліотекою, яка найбільш якісно впорається з поставленою задачею реалізації нейронної мережі для розпізнавання жестів, є бібліотека PyTorch.

1.4 Аналіз аналогів нейронної мережі

Системи розпізнавання жестів тільки починають свій розвиток та набувають популярності, а системи розпізнавання жестів для спілкування з людьми з вадами апарату мовлення або слуху ще недостатньо розвинуті та знаходяться на етапі розробки. Наразі є лише декілька комерційних систем, які успішно існують на ринку, тому було вирішено проаналізувати дві комерційні системи та одну систему з відкритим доступом.

Першою комерційною системою є система, розроблена компанією “Атанор”, яка створена з метою розпізнавання жестів для прискорення ведення презентацій. Система розпізнає жести у тривимірній проекції, завдяки чому можна не лише перегортати слайди презентації, але також наближати та віддаляти зображення, повертати та розтягувати на екрані 3D об'єкти, керувати налаштуваннями аудіо та відео [8]. Дана технологія корисна в таких ситуаціях, коли важлива кожна хвилина та максимальна віддача системи, за допомогою якої проводиться презентація.

Другою комерційною системою є мобільний додаток Сурдофон, який перетворює мову на мову жестів на екрані мобільного телефону. Місія компанії Сурдофон — створити комфортне середовище для спілкування та розвитку людей, для яких рідною є мова жестів [9]. Даний додаток слугує для подолання бар’єру в комунікації між людьми та його місія дуже близька до місії нейронної мережі, що розробляється. Однак переклад мови жестів здійснюється не за допомогою глибинного навчання та нейронних мереж, а за допомогою відеозв’язку з людиною, яка може перекласти мову, тож реалізація даного додатку є зовсім не такою, як мається на увазі в даному дослідженні.

Третім аналогом, який потребує огляду, є система CNNGestureRecognizer, програмний код якої знаходиться у відкритому доступі. Розпізнавання жестів було реалізовано за допомогою нейронної мережі та інструментів для роботи з нею Keras, Theano та бібліотеки OpenCV. Всі ці інструменти підходять для невеликих проектів, одним з яких є CNNGestureRecognizer. Варіанти жестів, які можуть бути розпізнані програмою: “Все добре”, “Мир”, “Стоп”, “Удар” та “Нічого” (коли жоден з перелічених жестів не зображений). Система дозволяє додати нові жести, однак функціонал її обмежений. Приклад роботи системи розпізнавання жестів CNNGestureRecognizer зображено на рисунку 1.2.

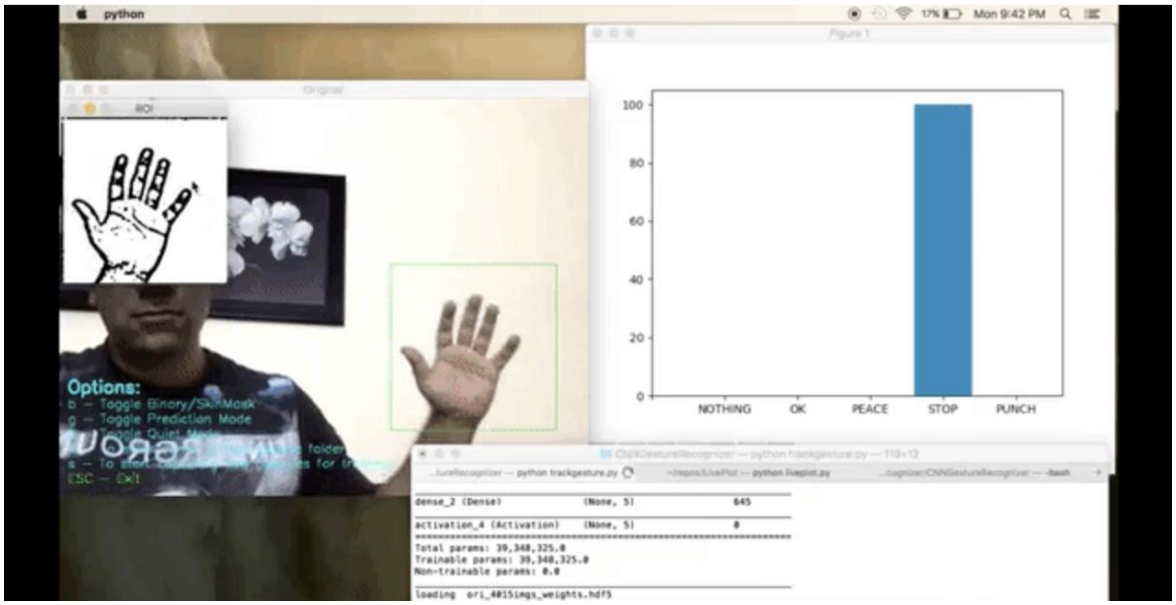

Рисунок 1.2 – Приклад роботи системи CNNGestureRecognizer

Отже, можна зробити висновок, що жодна розглянута система не відповідає принципам та вимогам, яких необхідно дотримуватися при створенні нейронної мережі для розпізнавання жестів, яка б допомогла спілкуватися з людьми, що мають вади апарату мовлення або слуху без сторонніх користувачів за допомогою глибинного навчання.

Для досягнення мети дослідження необхідно створити систему, яка в режимі реального часу буде перетворювати жести, що записуються через веб-камеру, на символи українського алфавіту. Система має бути точною та швидкою для того,

щоб всі жести було класифіковано та була можливість об’єднати їх в слова. Системи-аналоги, які представлені, мають дуже обмежений функціонал та мають бути розширені та покращені для реалізації нейронної мережі, запланованої в дослідженні. Після огляду аналогів можна точно сказати, що запланований функціонал тільки потребує розробки, що підтверджує актуальність дослідження.

1.5 Висновки

В ході написання першого розділу магістерської роботи було розглянуто та обрано алгоритм для розпізнавання жестів, які означають символи українського алфавіту, в реальному часі у відеопотоці, з використанням нейронної мережі.

Після огляду та вибору підходу для реалізації нейронної мережі було зроблено висновок, що найкращим підходом для розробки нейронної мережі з ціллю розпізнавання жестів буде використання глибинного навчання, а саме згорткових нейронних мереж.

Важливим етапом після вибору підходу є вибір бібліотеки для реалізації згорткової нейронної мережі, цей пункт також був освітлений в першому розділі магістерської роботи. Проаналізувавши всі бібліотеки, які використовуються для розпізнавання, був зроблений висновок, що бібліотекою, яка найбільш якісно впорається з поставленою задачею реалізації нейронної мережі для розпізнавання жестів, є бібліотека PyTorch.

Крім цього, було проведено аналіз аналогів нейронної мережі та зроблено висновок, що жоден з аналогів не відповідає поставленій задачі, це означає, що система, що розробляється, буде унікальною та її створення є актуальним.

2. ДОСЛІДЖЕННЯ ТА ПРОЕКТУВАННЯ ЗГОРТКОВОЇ НЕЙРОННОЇ МЕРЕЖІ

2.1 Мета та задачі дослідження нейронної мережі для розпізнавання жестів українського алфавіту

Нейронна мережа, що розробляється є програмним забезпеченням, основною функцією якого є розпізнавання жестів українського алфавіту в режимі реального часу та виводу розпізнаного жесту у вигляді символу (букви) українського алфавіту на робочий дисплей, що також містить відеопоток з камери робочої машини, на якій система розпізнавання буде працювати.

Мета дослідження нейронної мережі: програмне забезпечення повинне містити можливість перевести датасет фотокарток однакового розміру, що відповідає жесту українського алфавіту у вигляді MNIST датасету, або, іншими словами, у файл, що має тип розширення CSV. Також програмне забезпечення повинне мати можливість навчати нейронну мережу, що буде розпізнавати жести алфавіту української мови. Для взаємодії між користувачем та програмним забезпеченням повинний бути мінімальний командний інтерфейс та вікно, що буде виводити відеопоток с камери робочої машини на дісплей з виводом розпізнаного жесту.

Цільова аудиторія: люди з вадами слухового або мовного апаратів, організації з надання допомоги в спілкуванні таким людям; люди, які прагнуть вивчити мову жестів; організації, які бажають впровадити технологію розпізнавання жестів в свої системи, наприклад компанії з надання послуг відеозв’язку; розробники програмного забезпечення, що будуть бажати використовувати нейронну мережу в цілях розширити та використовувати у власному продукті.

Головними функціями програмного забезпечення є:

- функція розпізнавання жестів, що позначають символи алфавіту української мови, за допомогою використання нейронної мережі в відеопотоці в режимі реального часу;

- функція побудови власного MNIST датасету з фотокарток для наступного навчання класифікатору за допомогою машинного навчання;

- функція створення та навчання класифікатор української мови за допомогою глибокого навчання нейронною мережею;

- можливість оцінювання класифікатору жестів, який попередньо був навчений, на тестових даних для перевірки, щоб зрозуміти її точність;

- скрипт експорту моделі (класифікатору) у формат ONNX для підвищення швидкості формування логічного виводу системи;

- функція запуску програмного коду для прив'язки камери комп'ютера до класифікатора мови жестів, який будете отримувати вхідні дані камери (відеопоток), класифікувати відображуваний користувачем жест і потім повідомляти класифікований жест користувачеві.

Для того, щоб представити наочно вхідні та вихідні дані системи, можна зобразити їх у вигляді інформаційних потоків.

Інформаційні потоки системи розпізнавання жестів української мови зображені на рисунку 2.1.

На рисунку 2.1 можна побачити, що на вхід системи подається вхідний відеопоток з камери робочої машини, користувач може впливати на систему, а на виході системи користувач отримує розпізнаний жест, який позначає символ українського алфавіту, а також вихідний відеопоток з камери робочої машини.

Рисунок 2.1 – Інформаційні потоки системи розпізнавання жестів українського алфавіту

2.2 Типи користувачів

В системі розпізнавання жестів алфавіту української мови є єдиний тип користувача – це користувач системи.

Повний функціонал системи доступний користувачу, крім того, він має змогу створювати власний MNIST датасет у форматі CSV файлу, який буде використовуватися для навчання класифікатору розпізнавання жестів. Для досягнення мети навчання класифікатору, користувач має можливість за допомогою командного інтерфейсу у терміналі робочої машини активувати функцію глибинного навчання, процес якого буде відбуватися за допомогою нейронної мережі. Також користувач буде мати змогу перетворювати навчений класифікатор жестів української мови у формат ONNX для можливості переносити класифікатор у іншу систему без повторного навчання. Для користувача доступна функція оцінювання навченного класифікатору за допомогою проведення розпізнавання тестових фотокарток, які не використовувались при створення датасету. Також до основного функціоналу, яким володіє користувач системи,

належить увімкнення та вимкнення розпізнавання на робочій машині користувача.

2.3. Функціональні вимоги до системи розпізнавання жестів

Вимога 1 – налаштування робочої машини

F1.1 Функція розпізнавання жестів українського алфавіту

FR1.1.1 Користувач системи розпізнавання жестів може спостерігати за відеопотоком, що передається з камери робочої машини на її дісплей. При цьому жест особи, що розпізнається, повинен бути обмежений рамками та має містити літеру у лівому верхньому куті розпізнаного жесту користувача.

FR1.1.2 Якщо робоча машина не містить пристрій камери, то розпізнавання жестів проводитися не буде, оскільки іншого вхідного сигналу для роботи системи немає бути.

FR1.1.3 Будь-який жест, який показує користувач, має бути розпізнаний системою, навіть якщо він не є вірним.

FR1.1.4 При відсутності підключення до пристрою камери на робочій машині, користувач має бачити відповідний текст у терміналі з якого система вмикається для початку роботи.

FR1.1.5 Часове обмеження на процес розпізнавання жесту системою має складати не більш ніж 0.08 секунди на кожен фрейм с відеопотоку. Після відведеного часу, системою має бути виведено відповідне повідомлення про розпізнаний жест на робоче вікно на дісплеї робочої машини.

F1.2 Функція управління станом системи розпізнавання

FR1.2.1 Користувач має можливість вмикати і вимикати систему для розпізнавання. При успішному ввімкненні розпізнавання почнеться розпізнавання жестів українського алфавіту, при відключенні розпізнавання повинне бути припинено при будь-яких обставин.

FR1.2.2 При виникненні помилок при ввімкненні і вимкненні розпізнавання

користувач системи розпізнавання бачить повідомлення про конкретну проблему.

FR1.2.3 Часове обмеження на підключення складає 5 секунд після початку роботи системи. Після даного проміжку часу має бути виведено відповідне повідомлення про неможливість підключення до пристрою робочої машини для початку роботи системи.

Вимога 2 – передача інформації з пристрою камери робочої машини на її дисплей

F2.1 Функція отримання розпізнаного жесту користувача

FR2.1.1 Користувачі мають можливість отримати належний символ розпізнаного жесту в режимі реального часу.

FR2.1.2 Для коректного використання системи розпізнавання жестів української мови робота система повинна працювати в реальному часі без затримок з програмного боку.

FR2.1.3 Для можливості розпізнавання жесту на отриманому системою зображенні, зображення має містити жест, який вона буде порівнювати та розпізнавати.

Вимога 3 – Створення власного MNIST датасету

F3.1 Функція створення датасету

FR3.1.1 Для створення власного датасету список фотографій с певним типом класу (жесту) повинний знаходитись в одній папці з іншими за для можливості ідентифікувати певний жест системою розпізнавання.

FR3.1.2 Для створення датасету фотографія с певним типом класу (жесту) повинна мати певний розмір, що буде підтримуватися системою розпізнавання, а самне який становить 32 на 32 пікселя.

FR3.1.3 Для створення датасету фото с певним типом класу (жесту) повинні знаходитися в одному проекції на ряду з іншими.

FR3.1.4 Для створення датасету назва фото с певним типом класу (жесту) має містити лейб класу та власний унікальний номер фотокартки.

Вимога 4 – Навчання класифікатору розпізнавання жестів української мови

F4.1 Функція навчання класифікатору

FR4.1.1 Модуль навчання класифікатору за допомогою глибинного навчання нейронною мережею має містити щонайменше 2 згорткових на 2 повнозв’язних шари для коректного розпізнавання жесту українскої мови.

FR4.1.2 Згорткові шари нейронної мережі мають містити не менше ніж 6 вхідних та не більше ніж 16 вихідних сигналів.

FR4.1.3 Повнозв'язні шари нейронної мережі мають містити не більше ніж 576 вхідних сигналів та 33 вихідних класів.

FR4.1.4 Нейронна мережа системи розпізнавання жестів українського алфавіту має розпізнавати 33 жести українського алфавіту.

FR4.1.5 Класифікатор розпізнавання жестів українського алфавіту має розпізнавати жест у відеопотоці при різному типі освітлювання.

FR4.1.5 Класифікатор розпізнавання жестів українського алфавіту має розпізнавати жест у відеопотоці при повороті жесту вліво чи вправо, але не більше ніж на 20 градусів від коректного формування жесту.

Вимога 5 – Оцінювання системи розпізнавання жестів українського алфавіту

F5.1 Функція оцінювання навченого класифікатору

FR5.1.1 Результати оцінювання класифікатору на даних, які застосовувались для навчання системи, мають перевищувати 95% розпізнаних жестів українського алфавіту

FR5.1.2 Результати оцінювання класифікатору на тестових даних, які не застосовувались для навчання системи, мають перевищувати 90% розпізнаних жестів українського алфавіту

2.4 Нефункціональні вимоги

NF1. Мінімальні вимоги для початку роботи системи розпізнавання має містити інтерпретатор для роботи з мовою програмування Python v3.5.9 та щонайменше 1 ГБ вільної оперативної пам'яті на робочій машині.

NF2. Система розпізнавання жестів українського алфавіту має коректно працювати в актуальних Unix-подібних операційних системах:

- Linux Mint (версії 18.04 та вище);
- Ubuntu (версії 16 та вище);
- Debian (версії 10.0 та вище);
- Mageia (версії 7 та вище);
- Fedora (версії 30 та вище);
- OpenSUSE (версії 10.0 та вище);

NF3. При пошкодженні файлів класифікатору або модулів додатку, система має повідомляти користувачу про конкретну помилку завдяки користувальницькому командному інтерфейсу.

NF4. Максимальна кількість жестів для коректної роботи розпізнавання у відеопотоці становить 1 жест на кожен фрейм.

NF5. Відмови нейронної мережі внаслідок некоректних дій користувача при взаємодії з системою через програмний інтерфейс неприпустимі.

NF6. Мінімальна кількість персоналу, необхідного для роботи системи має становити 1 штатну одиницю – користувача нейронної мережі

2.5 Моделювання прецедентів

Моделювання прецедентів та подальше формування діаграми прецедентів або діаграми варіантів використання є найважливішою частиною проектування системи, так як саме на цьому етапі можна зобразити повний функціонал, доступний користувачу, типи користувачів та їхні можливості.

Завдяки визначенню функціональних вимог та типів користувачів, можна побудувати діаграму прецедентів, за допомогою якої є можливість візуалізувати функції, якими володіють користувачі системи розпізнавання жестів з використанням нейронної мережі. Діаграма прецедентів може надати більше

розуміння про можливі варіанти поведінки користувача в системі.

Діаграма варіантів використання системи розпізнавання зображена на рисунку 2.2.

На рисунку 2.2 можна побачити, що користувач системи може запускати роботу нейронної мережі та завершувати її, оцінювати навчання класифікатору жестів, створювати власний датасет, експортувати модель у форматі ONNX, переглядати та проводити розпізнавання безпосередньо в відеопотоці, а також навчати класифікатор за допомогою глибинного навчання.

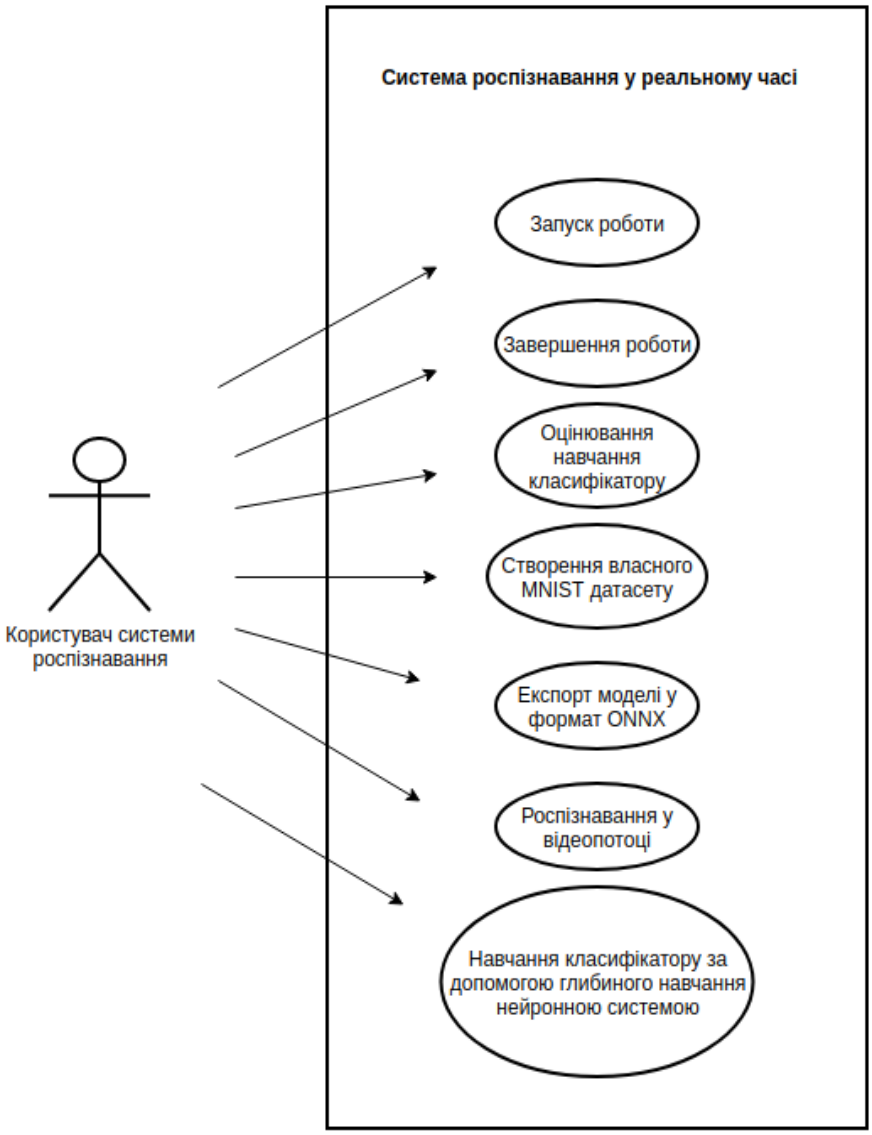

Рисунок 2.2 – Діаграма варіантів використання системи розпізнання жестів алфавіту української мови

2.6 Загальні характеристики для проведення наукового дослідження

2.6.1 Нейронна мережа

Нейронна мережа – це система, яка створена з метою відтворити роботу людського мозку за допомогою математичних моделей для створення програмного забезпечення штучним інтелектом, яке буде відтворювати запрограмовані правила поведінки. Наразі найрозвинутішою сферою, яка використовує нейронні мережі, є класифікація та розпізнавання образів.

2.6.2 Штучна нейронна мережа

В роботі буде використовуватися штучна нейронна мережа, тож необхідно дати визначення даному поняттю.

Штучна нейрона мережа є типом нейронної мережі, яку тренують разом із викладачем, що означає, що система має навчальний набір даних (наприклад, фотокартки із зображеннями символів), який містить приклади з реальними значеннями: теги, класи, символи, жести. В даній роботі при створенні нейронної мережі, що буде розпізнавати жести, які означають символи українського алфавіту, датасетом буде виступати набір фотокарток с жестами, що означають літери української мови, які будуть поставлятися для навчання класифікатору моделі класів для нейронної мережі у вигляді CSV файлу, також його ще називають MNIST Image Dataset.

2.6.3 Гіперпараметри

При реалізації штучної нейронної мережі та її проектуванні особливу увагу потрібно приділити гіперпараметрам, так як від їх вибору залежить ефективність системи.

Гіперпараметри – це параметри, які потрібно вводити в систему власноруч, оскільки нейронна мережа використовується для автоматизації вибору функцій.

До найбільш важливих гіпермараметрів належить швидкість навчання в нейронній мережі для розпізнавання жестів, що позначають символи українського алфавіту. Якщо темп навчання занадто повільний, то навіть після тривалого навчання нейронної мережі він буде далекий від оптимального.

2.6.4 Штучний нейрон

Штучний нейрон є одним з компонентів нейронної мережі, що отримує отримані вхідні сигнали (початкові дані або вихідні дані від інших нейронів нейронної мережі) через декілька вхідних каналів, тож подати його характеристики надзвичайно важливо при проектуванні нейронної мережі.

Кожен вхідний сигнал має проходити через ваговий зв'язок для отримання результату, куди далі потрібно відправити цей нейрон. З кожним нейроном пов’язане відповідне порогове значення.

Так як для того, щоб отримати результат, система повинна розрахувати зважену суму вхідних даних, після чого з якої віднімається поріг, і результатом є значення активації нейрона. Сигнал активації перетворюється за допомогою функції активації, яка виробляє вихідний сигнал від нейрона.

На рисунку 2.3 представлено приклад штучного нейрона.

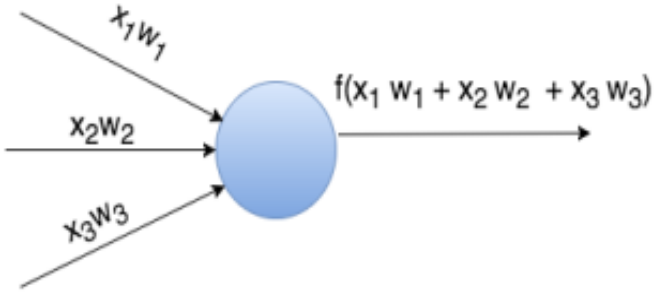

Рисунок 2.3 – Штучний нейрон

Де: x_i− вхідний сигнал, w_i− вага вхідного сигналу, $f(.)$− функція активації.

2.6.5 Функція активації

Функція активації є одним з найважливіших інструментів впливу на потужність, яка приписується нейронним мережам, оскільки функція активації частково вирішує, які нейрони будуть активовані, тобто яка інформація буде передана на наступні шари нейронної мережі. Без активації глибокі мережі втрачають значну частину своїх можливостей навчання, оскільки ці функції допомагають виміряти вагу вхідного сигналу, що обробляються системою для розпізнавання класу (якщо брати до уваги систему, що створюється, то розпізнавання жесту). Нелінійність цих функцій відповідає за певне збільшення ступеня свободи для системи, що дозволяє узагальнювати задачі великої розмірності в менших вимірах. При створенні нейронної мережі для розпізнавання жестів, що означають символи української мови, функцією активації буде виступати функція активації ReLU.

2.6.6 Функція активації ReLU

Для задачі реалізації нейронної мережі для розпізнавання жестів, що означають символи української мови, було вирішено використовувати функцію активації ReLU.

Використовуючи формулу 2.1 стає зрозуміло, що функція активації ReLU повертає x, якщо x є додатним, і 0 в іншому випадку. Ця функція активації є нелінійною, як і її комбінація, тобто є можливість складати докупи шари. Діапазон всіх дійсних значень для функції активації ReLU становить [0, inf), тобто активація може «вибухнути». На рисунку 2.4 зображений графік функції активації ReLU.

$$f(x) = max(0, x)$$

Формула 2.1 – Функція активації ReLU

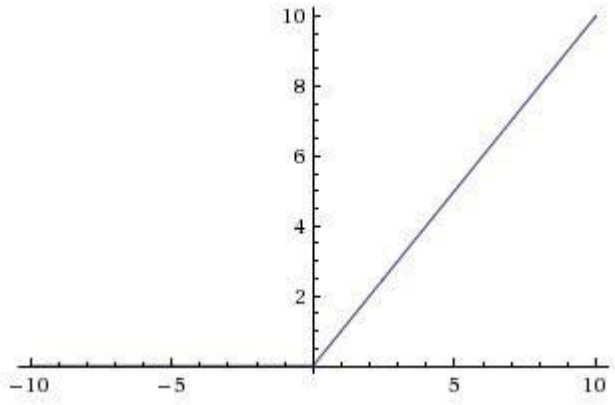

Рисунок 2.4 – Графік функції ReLU або лінійний випрямляч

Основною перевагою функції активації ReLU є її математична форма, функція вимагає менше ресурсів, які необхідні для обчислення, що прискорює час фізичного навчання нейронної мережі та підходить для саме для створення глибоких нейронних мереж, які було вирішено використовувати при створення нейронної мережі для розпізнавання жестів, що означають символи української мови.

Використовуючи функцію активації ReLU, з’являється можливість для того, щоб деактивувати деякі нейрони, що може також прискорити нейронну мережу з точки зору часу навчання епохи.

Недоліки функції активації ReLU пов'язані з тим, що негативні вхідні значення можуть призвести до нульового градієнта на виході, що може негативно вплинути на процес навчання нейронної мережі через відсутність оновлення вагового коефіцієнта. Таку часту проблему частково вирішує модифікація функції активації ReLU, в якій для від’ємних значень вхідного параметра використовується лінійна функція з малим коефіцієнтом пропорційності, що дозволяє нейронній мережі позбутися нульового градієнта та коригувати вагові коефіцієнти в процесі навчання нейронної мережі.

Функція втрат знаходиться в центрі нейронної мережі та створена з метою обчислення похибки між фактичним та отриманим результатами. Основною метою при роботі з даним поняттям є мета мінімізувати цю помилку. Функція втрат є одновимірною, а не векторною, оскільки вона оцінює, наскільки добре працює нейронна мережа в цілому.

Таким чином, функція втрат може наблизити навчання нейронної мережі до цієї мети та мінімізувати помилку. Функція втрат буде вимірювати та оцінювати, наскільки нейронна мережа готова для роботи з даним навчальним набору даних та очікуваних відповідей та результатів. Це також може залежати від ваги та систематичних помилок, дані змінні можуть надавати великий вплив.

2.6.7 Глибоке навчання нейронної мережі

Клас алгоритмів глибокого навчання є класом алгоритмів машинного навчання, які навчаються глибше розуміти вхідні дані на етапі навчання нейронної мережі. Глибоке навчання може бути представлене як каскад, тобто передається послідовно, використовується з різних рівнів обробки для вилучення та перетворення ознак, тобто є нелінійним.

Глибинне навчання може бути побудовано на основі вивчення характеристик та вхідної інформації без контрольованого навчання. Функції більш високого рівня,

які знаходяться в останніх шарах, отримуються з функцій нижчого рівня, тобто тих, які знаходяться в шарах початкового шару.

Глибинне навчання може розглядати багатошарові структури, які відповідають різним рівням абстракції, а рівні в свою чергу представляють собою ієрархію.

2.6.8 Унітарне кодування

Унітарне кодування є технологією, яка буде використовуватися при реалізації нейронної мережі для розпізнавання жестів, що позначають символи українського алфавіту, тож необхідно дати його визначення.

Пряме унітарне кодування, з англ. one-hot encoding (OHE) – це представлення кінцевого набору значень ознак у вигляді двійкового коду фіксованої довжини, що містить лише одну 1, тобто значення ознаки дорівнює 1, а всі інші дорівнюють 0. Для наведення прикладу, у зворотному кодуванні є лише один 0. Довжина цього коду визначається кількістю закодованих значень атрибута або об'єктів.

2.7 Архітектура згорткової нейронної мережі

З появою величезних обсягів різноманітніх даних та дуже великих обчислювальних можливостей у сучасному світі стали активно використовуватися різного роду нейронні мережі. Особливу популярність набули згорткові нейронні мережі, архітектура яких була запропонована Яном Лекуном [10] і націлена на ефективне розпізнавання зображень.

Архітектура згорткової нейронної мережі отримала свою назву, тому що для її реалізації використовується операція згортки та архітектура такої нейронної

мережі будується навколо неї. Сутність операції згортки в тому, що поелементно кожен фрагмент зображення повинен бути помножений відповідно на матрицю згортки чи, як її ще називають, ядро, поелементно, після чого результат сумується та записується у відповідну позицію зображення на виході. Також архітектура нейронної мережі підсумовує знання, які були отримані в ході досліджень області комп’ютерного зору, наприклад, те, що сусідні пікселі пов’язані більш за інші, а також те, що об’єкт може бути помічений у будь-якому місці на зображенні.

Особливу увагу згортковим нейронним мережам було встановлено після конкурсу ImageNet, який відбувся у жовтні місяці 2012 року та був присвячений класифікації різноманітних об'єктів чи купі об’єктів на певних фотографіях. У конкурсі потрібно розпізнавати велику кількість образів у 1000 категорій. Переможцем цього конкурсу став Алекс Крижевський, який використовуючи згорткову нейронну мережу, значно перевершив інших учасників [11].

Отже, можна підсумувати, що згорткові нейронні мережі активно використовуються у галузі розпізнавання та класифікації об’єктів, а також можуть бути використані в процесі розпізнавання жестів. В сучасному світі згорткові нейронні мережі набувають все більшої популярності, успіху та розвитку, тож можна стверджувати, що для реалізації мети, поставленої в даній роботи, підходять краще за інші.

Згорткова нейронна мережа зазвичай є чергуванням згорткових шарів (convolution layers), субдискретизирующих шарів (subsampling layers) та за наявності повнозв'язних шарів (fully-connected layer) на виході. Усі види шарів можуть чергуватись у довільному порядку [12].

Задача класифікації жестів є найбільш підходящою для алгоритмів згорткової нейронної мережі, так як існує кінцевий набір символів в алфавіті та безліч зображень, на яких можна тренувати нейронну мережу для того, щоб довести розпізнавання до потрібних характеристик та дотриматися поставлених вимог, що призведе до використання системи та надання їй розголосу.

Для демонстрації приклад архітектури згорткової нейронної мережі зображено на рисунку 2.5. На рисунку можна бачити, що нейрони з однаковими

вагами формують карти ознак, а також що кожен нейрон зв’язаний з нейронами, які знаходяться на попередньому шарі системи.

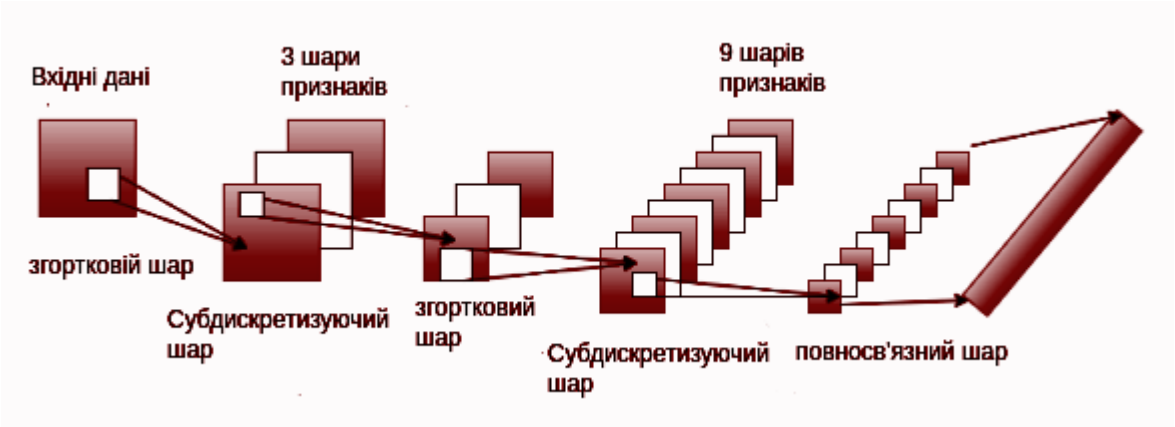

Рисунок 2.5 – Приклад архітектури згорткової нейронної мережі

2.7.1 Повнозв'язний шар нейронної мережі

Повнозв'язний шар – це шар, у якому кожний нейрон пов’язаний з нейронами на попередньому рівні. Особливістю повнозв’язного шару є те, що кожен зв’язок характеризується своїм ваговим коефіцієнтом.

Приклад архітектури повнозв’язного шару подано на рисунку 2.6.

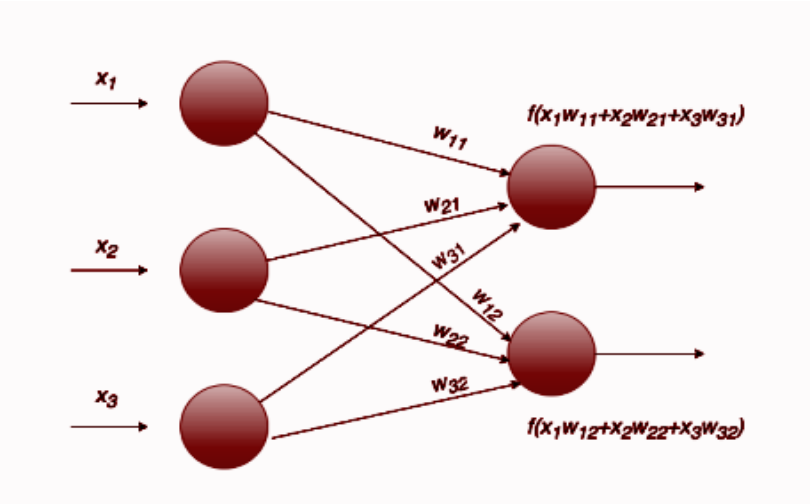

Рисунок 2.6 – Архітектура повнозв’язного шару згорткової нейронної мережі

2.7.2 Згортковий шар нейронної мережі

Архітектура згорткового шару нейронної мережі кардинально відрізнається від повнозв’язного шару тим, що нейрон поєднаний з обмеженою кількістю нейронів на попередньому рівні, а не зі всіма нейронами, як це було для повнозв’язного шару. Однак, згортковий шар можна порівняти з операцією згортки та виявити особливість того, що використання згорткового шару буде впливати на вхідне зображення та стирати межу зображення через крайові ефекти, що використовуються. Даний ефект можна побачити на рисунку 2.7, де зображена робота згорткового шару на ядрі згортки, яке має розмір 3 на 3.

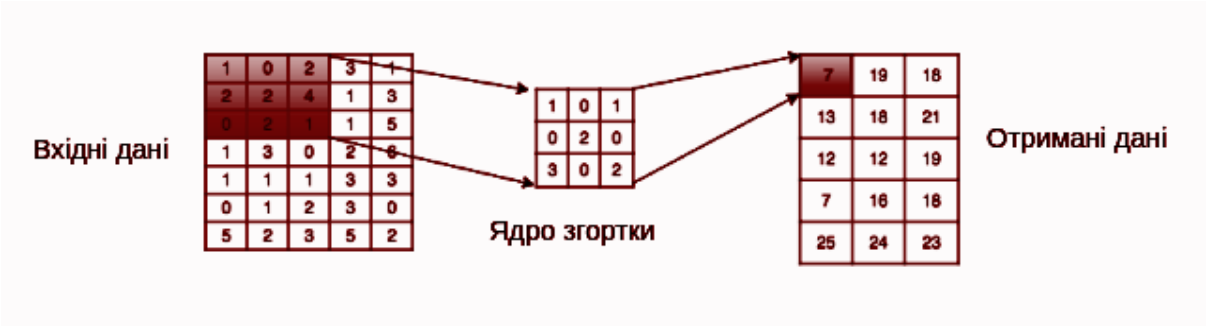

Рисунок 2.7 – Робота згорткового шару нейронної мережі

2.7.3 Субдискретизуючий шар нейронної мережі

Субдискретизуючий шар нейронної зазвичай використовується з метою зменшення розміру зображення в декілька разів. Найрозповсюдженіше використання субдискретизуючого шару зазвичай відбувається в парі з методом вибору максимального елементу з вибірки, тобто елементу, який несе в собі найбільшу кількість інформації. Карта ознак поділяється на частини, після чого серед елементів, що потрапили до частини, обирається та залишається найбільший елемент.

На рисунку 2.8 зображений результат роботи субдискретизуючого шару та процес вибору максимального елементу з вибірки.

Рисунок 2.8 – Робота субдискретизуючого шару нейронної мережі

2.7.4 Dropout шар нейронної мережі

Dropout шар нейронної мережі зазвичай використовується для того, щоб запобігти перенавчання нейронної мережі. Dropout шар нейронної мережі часто називається Dropout регуляризацією, такий процес полягає у тому, що кожен нейрон видаляється з первною ймовірністю p. Після такого процесу така нейронна мережа без нейронів проходить процес навчання, на ваги робиться градієнтний крок, а потім нейрони, які були видалені, повертаються.

При тестуванні нейронної мережі нейрони вже не видаляються, але з метою уникнення перенавчання, але вихід кожного нейрона множиться на (1 − p), тобто від 1 віднімається ймовірність кожного нейрона.

Даний метод є найрозповсюдженішим та найефективнішим в питанні уникнення перенавантаження та буде використаний в процесі реалізації нейронної мережі для розпізнавання жестів, що позначають символи алфавіту української мови, з використанням глибокого навчання.

2.8. Опис стеку бібліотек та технологій

З ряду технологій, бібліотек, моделей та інструментів, які можуть бути використані при реалізації нейронної мережі для розпізнавання жестів українського алфавіту, були проаналізовані та обрані тільки ті, які підходять найбільше та забезпечують найкращі показники ефективності:

- Python v3.5.9 є високорівневою мовою програмування загального призначення, орієнтована на підвищення продуктивності розробника і читання коду. Синтаксис ядра Python мінімалістичний. У той же час стандартна бібліотека включає великий обсяг корисних функцій. Python підтримує структурне, об'єктно-орієнтоване, функціональне, імперативне і аспектно-орієнтоване програмування [13]. Це мова, яка підтримує найбільш повний комплект сучасних технологій, а саме: динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки виключень, підтримка багатопоточних обчислень, високорівневі структури даних. Програми можуть розбити на модулі та пакети з використанням даної мови програмування;

- numpy v1.14.5 є основним пакетом для наукових обчислень на Python. Це бібліотека Python, яка надає багатовимірний об’єкт масиву, різноманітні похідні об’єкти, а також набір підпрограм для швидких операцій над масивами, включаючи математичні, логічні, маніпуляції з формою, сортування, вибір, введення/виводу, базова лінійна алгебра, основні статистичні операції, випадкове моделювання та багато іншого;

- onnx v1.6.0 є відкритим форматом, який був створений для представлення моделей машинного навчання. ONNX визначає загальний набір операторів, будівельних блоків машинного навчання та моделей глибокого навчання, і загальний формат файлів, щоб дозволити розробникам нейронних мереж та штучного інтелекту використовувати моделі з різноманітними фреймворками, інструментами, середовищами виконання та компіляторами;

- onnxruntime v1.0.0 є продуктивним механізмом висновку для моделей машинного навчання у форматі ONNX на Linux, Windows і Mac;

- opencv-python v3.4.3.18 є останньою версією бібліотеки прив’язок Python, яка призначена для вирішення проблем комп’ютерного зору. OpenCV-Python

використовує одну з найвідоміших бібліотек Numpy, яка є високо оптимізованою бібліотекою для різнопланових числових операцій із синтаксисом мови у стилі MATLAB. Усі структури масивів бібліотеки OpenCV повністю перетворювані в масиви Numpy та з них. Це також полегшує інтеграцію з іншими python бібліотеками, які використовують Numpy, такими як SciPy і Matplotlib;

- Pillow v7.2.0 є бібліотекою зображень Python додає можливості обробки зображень до вашого інтерпретатора Python. Ця бібліотека забезпечує широку підтримку форматів файлів, ефективне внутрішнє представлення та досить потужні можливості обробки зображень. Основна бібліотека зображень розроблена для швидкого доступу до даних, що зберігаються в кількох основних форматах пікселів. Він повинен забезпечити міцну основу для загального інструменту обробки зображень;

- pip v9.0.1 є стандартним менеджером пакетів для Python. Він дозволяє встановлювати та керувати додатковими пакетами, які не є частиною стандартної бібліотеки Python;

- protobuf v3.19.1 є розширюваним механізмом Google, який не залежить від мови та платформи, для серіалізації структурованих даних – уявіть XML, але менший, швидший і простіший;

- setuptools v28.8.0 є бібліотекою, яка призначена для розробки пакетів та полегшення пакування проектів Python шляхом покращення стандартної бібліотеки Python distutils (служб розповсюдження);

- six v1.16.0 є бібліотекою сумісності Python 2 і 3, що надає допоміжні функції для згладжування відмінностей між версіями Python з метою написання коду Python, сумісного з обома версіями Python;

- torch v1.2.0+cpu є бібліотекою машинного навчання з відкритим вихідним кодом, а також науково-обчислювальним фреймворком і мовою сценаріїв на основі мови програмування Lua. Вона надає велику кількість різноманітних алгоритмів для глибокого машинного навчання та й використовує спеціальну мову сценаріїв LuaJIT та базову реалізацію мови програмування C;

- torchvision v0.4.0+cpu є бібліотекою для комп’ютерного зору, яка використовується разом із PyTorch. Вона має утиліти для ефективного перетворення зображень і відео, деякі часто використовувані попередньо навчені моделі та деякі набори даних;

- typing-extensions v3.10.0.2 є допоміжною бібліотекою для синтаксичних підказок за типами даних, які не реалізовані в основному пакеті Python;

- PyCharm CE v2021.2 є інтегрованим середовищем розробки для програмування на мові Python. Надає засоби для аналізу коду, графічний зневаджувач, інструмент для запуску юніт-тестів і підтримує веб-розробку на Django. PyCharm розроблено російською компанією JetBrains на основі IntelliJ IDEA. PyCharm працює під операційними системами Windows, Mac OS X і Linux. Користувачі можуть писати свої плагіни, тим самим розширюючи можливості PyCharm. Деякі плагіни з інших IDE JetBrains можуть працювати з PyCharm. Існує більше тисячі плагінів, сумісних з PyCharm [14].

2.9 Користувальницький інтерфейс робочого вікна

У ході проектування системи розпізнавання важливим етапом є створення макету користувальницького робочого вікна інтерфейсу системи. Це дозволить краще візуалізувати потрібний результат та допоможе виводити символ розпізнаного жесту в кутку робочого вікна системи наряду з відеопотоком з приладу камери робочої машини користувача у режимі реального часу.

На рисунку 2.9 можна бачити інтерфейс робочого вікна, що з’явиться на дисплеї робочої машини користувача. У верхній частині сторінки знаходяться назва системи розпізнавання, а розпізнаний символ жесту, що міститься всередині відеопотоку з приладу камери, знаходиться у верхній частині робочого вікна с напівжирним темним кольором, для кращого виводу та читання символу розпізнаного жесту на дисплеї робочої машини.

На рисунку 2.9 можна побачити, що інтерфейс користувача складається з екрану, в якому зображений відеопоток в реальному часі, відображення розпізнаної букви (символу) українського алфавіту, а також координати для розпізнавання.

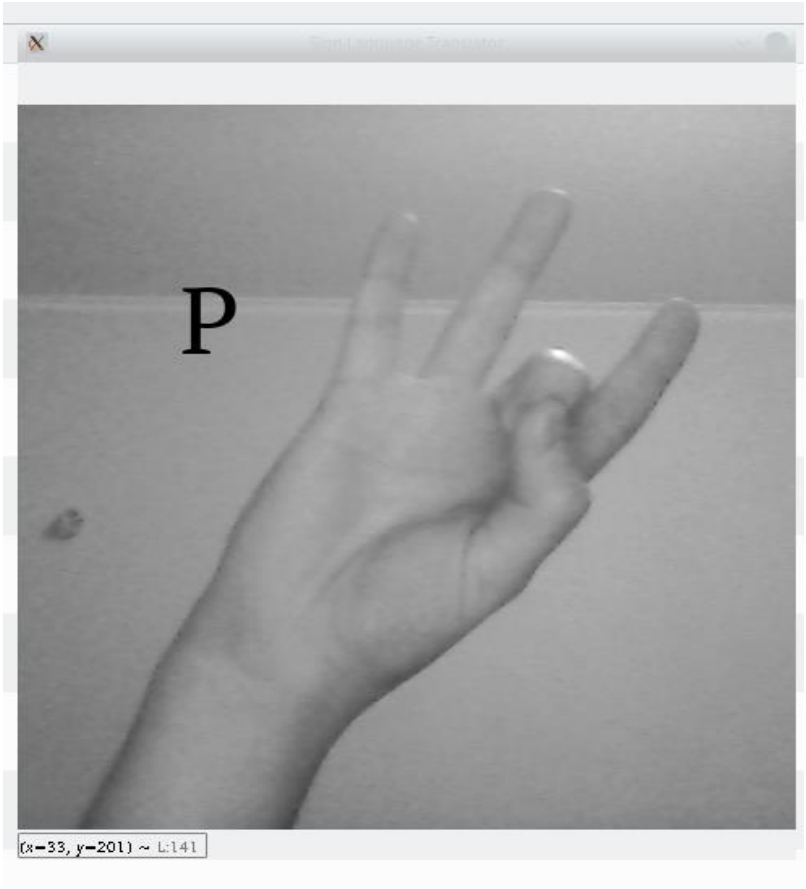

Рисунок 2.9 - Приклад робочого вікна системи

2.10 Логічне уявлення про систему розпізнавання жестів

Для уявлення майбутньої системи побудуємо діаграму, за допомогою якої покажемо особливості зв’язку компонентів системи (рис. 2.10). Логічне уявлення про систему розпізнавання жестів алфавіту української мови з використанням технології глибокого навчання класифікатору жестів за допомогою нейронної мережі, що взаємодіє з користувачем за допомогою камери робочої машини користувача та написаної на мові Python та фреймворку PyTorch, є дуже важливим

для розуміння роботи системи. Досліджувана система розпізнавання жестів розпізнає отриманий жест з відеопотоку приладу камери користувача та відображає оброблений результат в режимі реального часу. Отже, на схемі логічного уявлення системи можна побачити взаємодію користувача з системою розпізнавання жестів.

Логічне уявлення про систему розпізнавання жестів зображене на рисунку 2.10.

На рисунку 2.10 можна побачити, що користувач взаємодіє з системою через інтерфейс користувача, де бачить результат розпізнавання. Також користувач системи взаємодіє безпосередньо з камерою, яка використовується при роботі системи та передає відеопоток до нейронної мережі розпізнавання жестів. Сама система розпізнавання жестів проводить розпізнавання отриманих фреймів з відеопотоку приладу камери та повертає результат роботи та розпізнаний жест до користувальницького інтерфейсу.

Отже, через логічне уявлення системи можна зрозуміти, як користувач взаємодіє з нейронною мережею та як компоненти системи взаємодіють між собою.

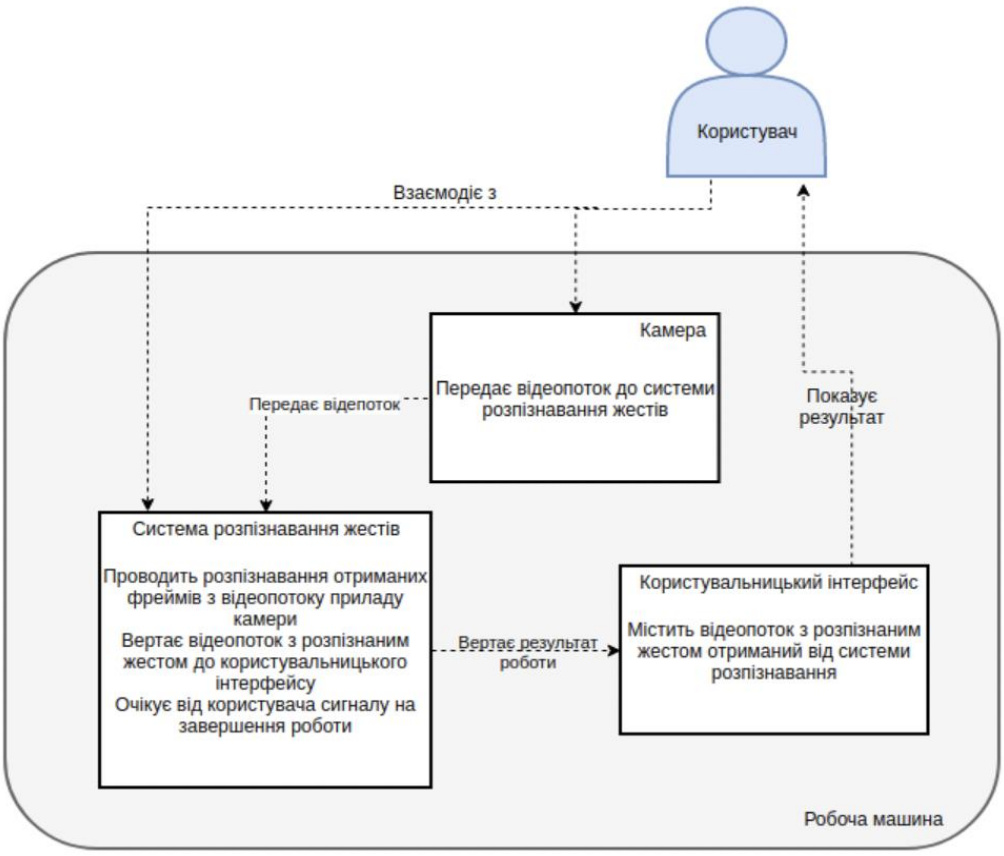

Рисунок 2.10 – Логічне уявлення про систему розпізнавання жестів

2.11 Висновки

В ході написання другого розділу магістерської роботи була розглянута та спроектована нейронна мережа для розпізнавання жестів, які означають символи українського алфавіту.

Були поставлені мета та задачі дослідження нейронної мережі для розпізнавання жестів українського алфавіту, сформована цільова аудиторія, інформаційні потоки системи. Описані типи користувачів системи, функціональні та нефункціональні вимоги до нейронної мережі та системи розпізнавання загалом.

Змодельована діаграма прецедентів, яка наочно ілюструє функціонал, доступний користувачу системи.

Виявлені загальні характеристики для проведення наукового дослідження: подані визначення нейронної мережі та штучної нейронної мережі, визначена архітектура та характеристики нейронної мережі.

Описаний стек бібліотек та технологій, які найбільш вдало підходять для системи та будуть використані при її реалізації.

Спроектований інтерфейс користувача, через який він зможе передавати зображення в систему та отримувати результат.

Було спроектовано логічне уявлення про систему розпізнавання жестів, яке допомагає зрозуміти, як саме взаємодіє з системою користувач та компоненти системи між собою.

3. РЕАЛІЗАЦІЯ СИСТЕМИ ДЛЯ РОЗПІЗНАВАННЯ ЖЕСТІВ АЛФАВІТУ УКРАЇНСЬКОЇ МОВИ

3.1. Уявлення про структуру проекту

Перед тим, як перейти до реалізації нейронної мережі, необхідно представити, як саме файли з програмним кодом будуть зберігатися. Зазвичай для такої задачі зображують уявлення пакетів програми, тобто їхню ієрархічну структуру.

Така процедура важлива для того, щоб при створенні пакетів в середовищі програмування у програмного коду був доступ до всіх необхідних частин.

На рисунку 3.1 зображена структура головного пакету системи sign-recognition-system, в якому зберігаються дані для навчання та тренування нейронної мережі, шрифти для відображення розпізнаної літери українського алфавіту в інтерфейсі користувача в реальному часі, а також датасети та модуль, який відповідає за підключення камери, розбиття відеозображення на фрейми та передачу зображення в систему розпізнавання.

Класифікатор нейронної мережі має бути збережений у файлі checkpoint.pth, він зберігає навчену нейронну мережу для розпізнавання.

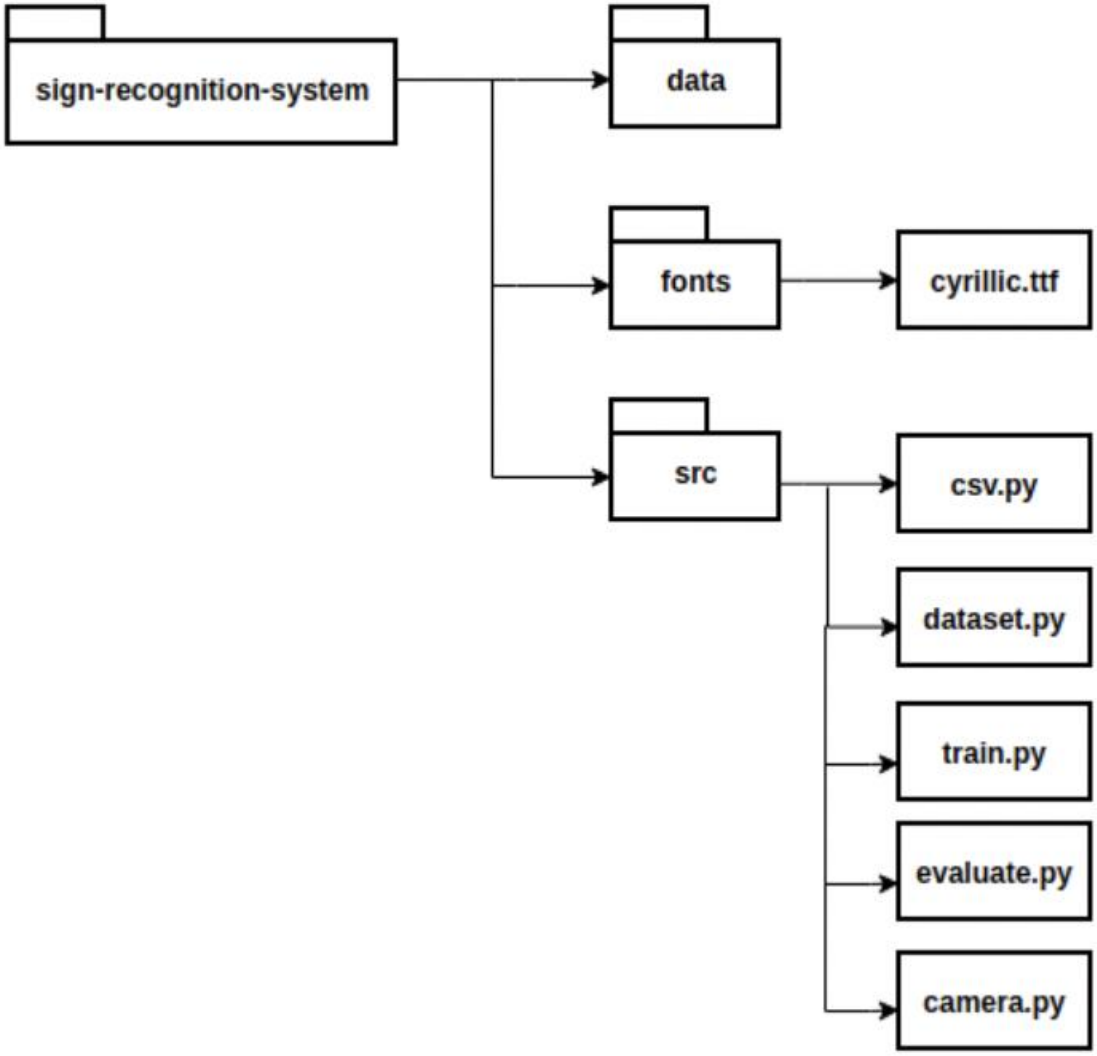

Рисунок 3.1 – Структура головного пакету системи

На рисунку 3.2 зображена розгорнута структура директорії «data». який зберігає зображення у форматі «png» для літер українського алфавіту для навчання та валідації роботи нейронної мережі на тестових даних. Пакет для тренування містить пакети з літерами від А до Я та відповідно до кожної літері належать 300 зображень для навчання нейронної мережі.

В пакеті з тестовими зображеннями збережено по одному зображенню літер українського алфавіту від А до Я.

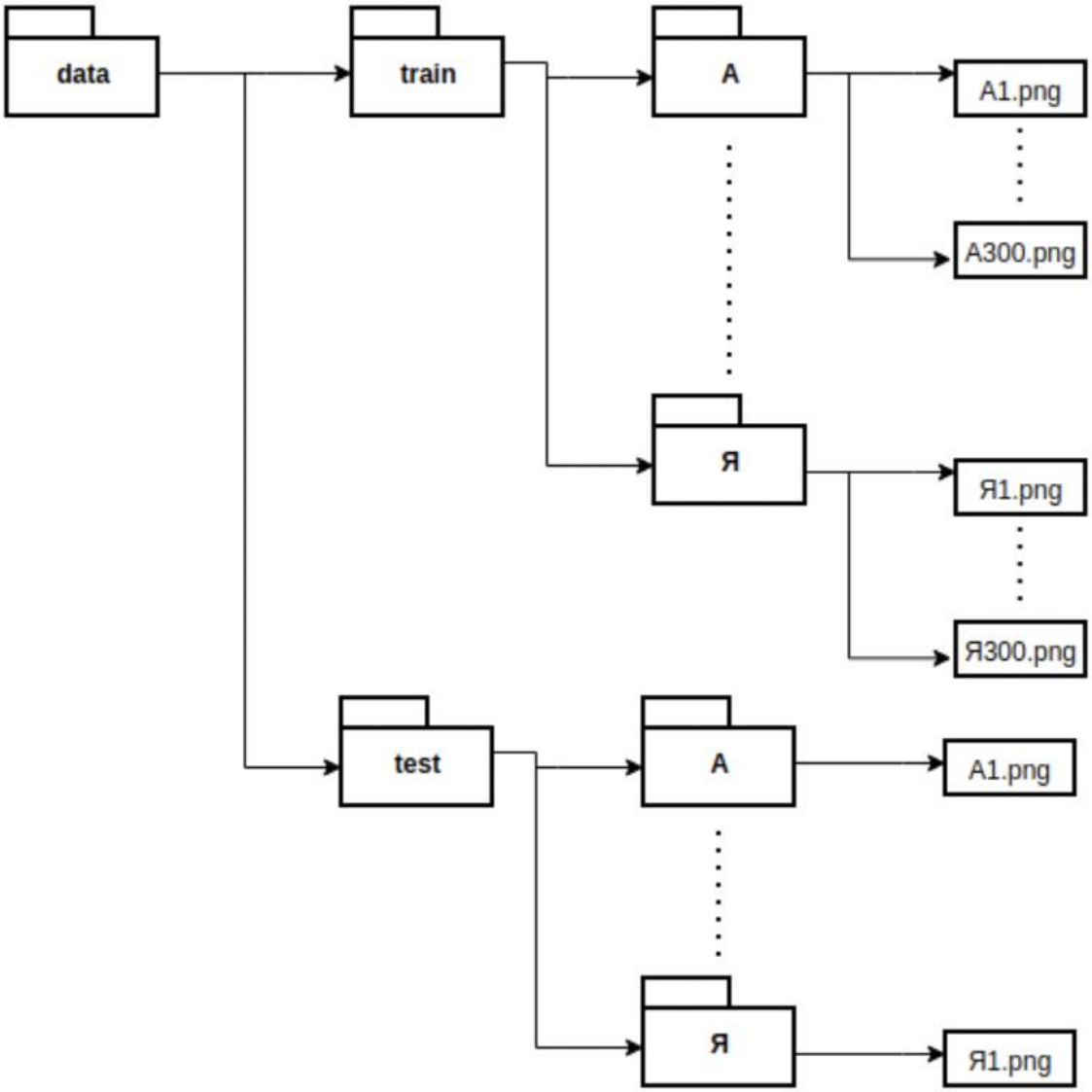

Рисунок 3.2 – Розгорнута структура пакету «data»

3.2 Уявлення процесів роботи нейронної мережі

Для того, щоб коректно представити всі процеси роботи системи розпізнавання символів українського алфавіту з використанням нейронної мережі, спочатку необхідно представити уявлення процесів. В даному випадку, представлення процесів, які проходять в системі, зручніше за все представити у вигляді діаграм діяльності.

На рисунку 3.3 зображена діаграма діяльності процесу розпізнавання жестів від камери до робочого вікна, яке слугує інтерфейсом для користувача.

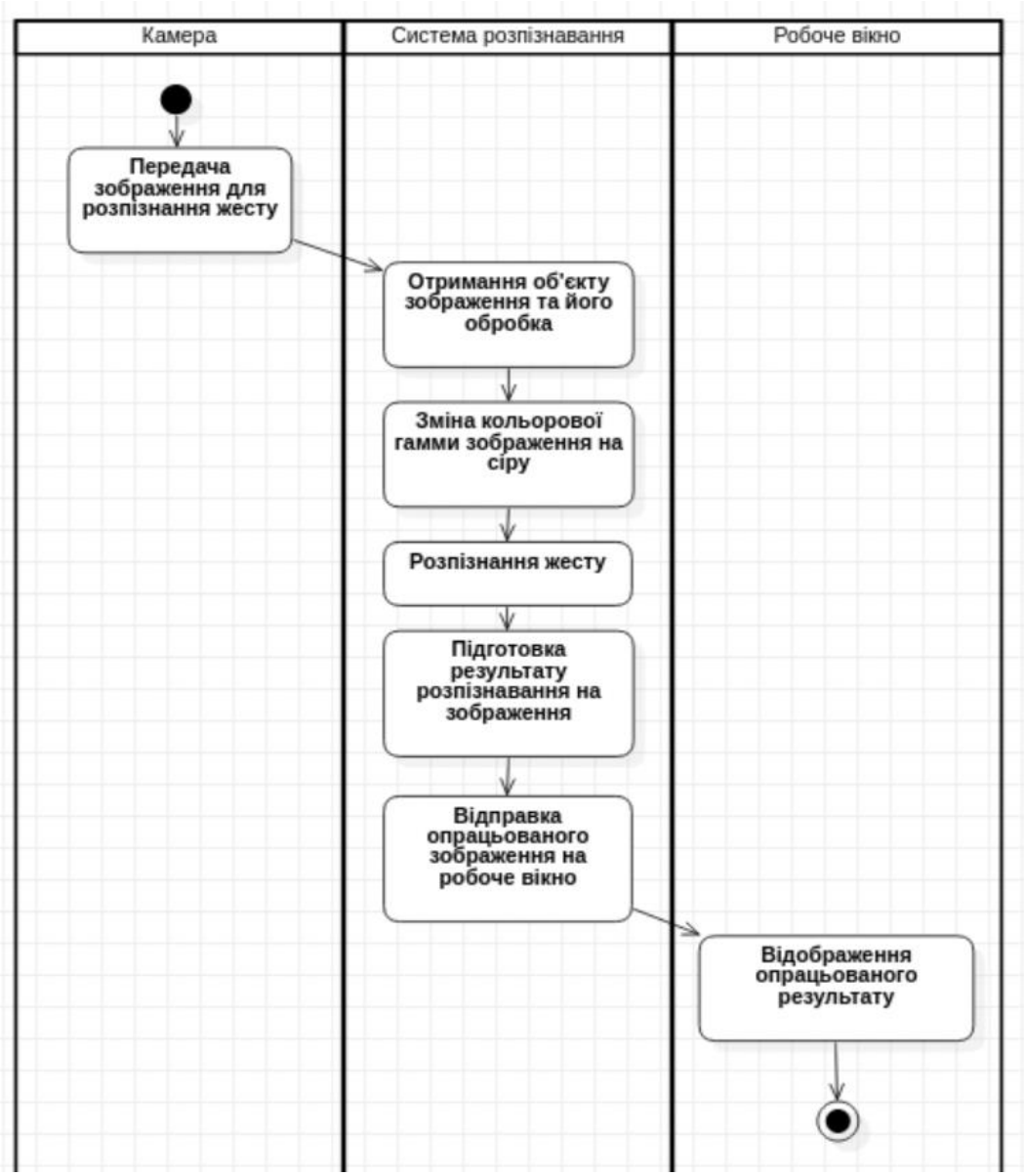

Рисунок 3.3 – Діаграма діяльності процесу розпізнавання жестів

На рисунку 3.3 зображене представлення процесів від камери, яка передає зображення в режимі реального часу в систему розпізнавання, яка обробляє зображення, міняє кольорову гаму зображення на сіру, після чого відбувається безпосередньо розпізнавання жесту, підготовка результатів розпізнавання та відправка результату роботи нейронної мережі для відображення користувачу.

На рисунку 3.4 зображений процес оцінювання навченого класифікатора розпізнавання жестів українського алфавіту.

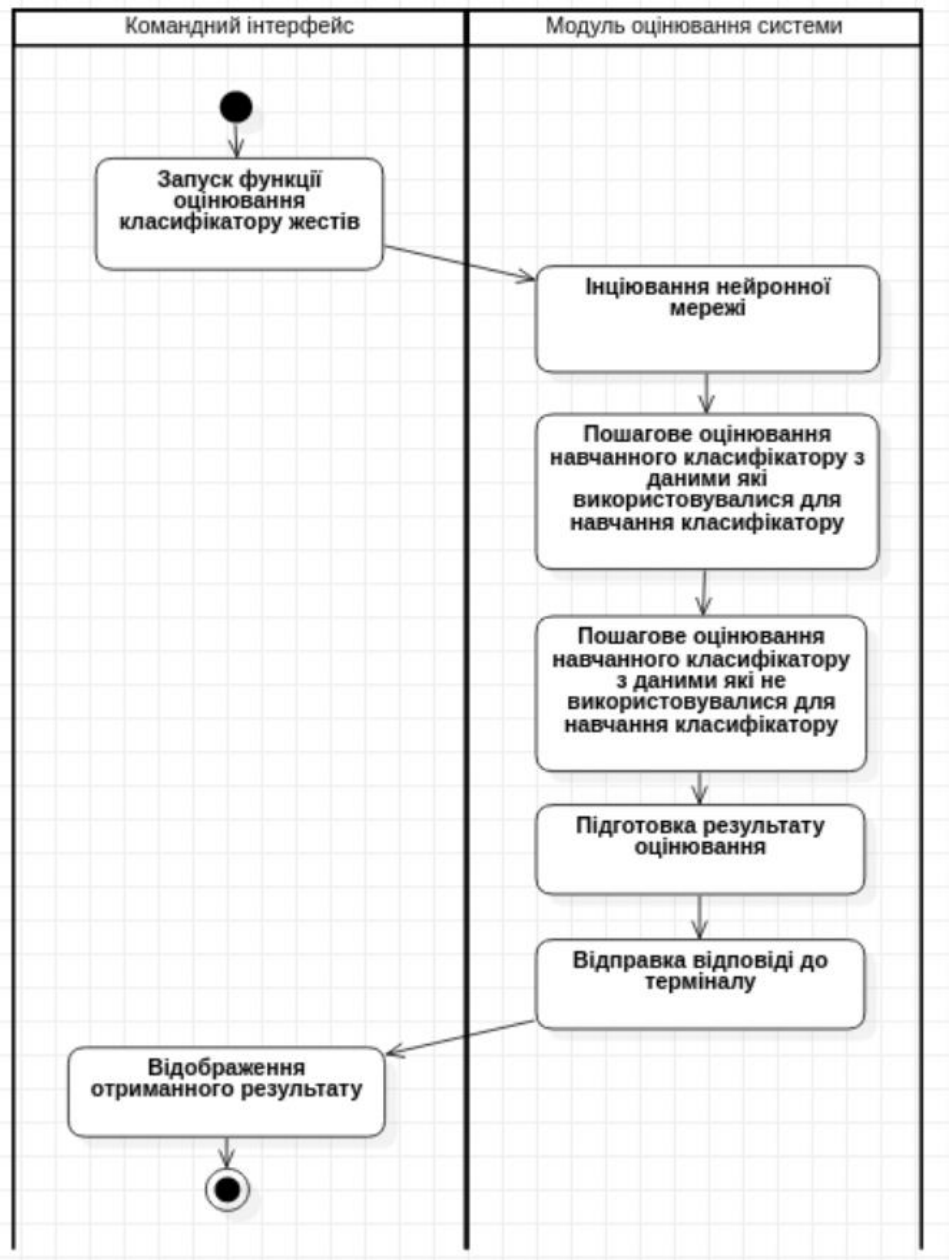

Рисунок 3.4 – Процес оцінювання навченого класифікатора

На рисунку 3.4 зображено командний інтерфейс, через який відбувається запуск функції оцінювання класифікатора розпізнавання жестів українського алфавіту, після чого починається ініціювання нейронної мережі, по шагове оцінювання навчального класифікатору спочатку з даними, які використовувалися для навчання, а потім з новими даними, які не використовувалися для навчання,

після чого відбувається підготовка та відправка результату оцінювання до терміналу та відображення результату користувачу.

3.3 Уявлення про класи системи розпізнавання жестів

Після того, як були представлені пакети системи та сформоване уявлення про те, як повинні проходити процеси в системі, після формування структури пакетів та зображення діаграм діяльності системи розпізнавання, можна перейти до написання класів нейронної мережі.

Представлення діаграм класів є невід’ємною частиною створення будь-якого програмного забезпечення для того, щоб правильно спроектувати структуру програмного коду, наперед визначити, які саме будуть потрібні методи та змінні та як класи будуть взаємодіяти між собою, щоб працювати найбільш ефективно, швидко та якісно.

На рисунку 3.5 зображено діаграму класів для процесу навчання нейронної мережі. На рисунку зображено клас Net, який слугує для навчання класифікатора та є класом нейронної мережі. Клас Module є стандартним класом для перевизначення. Клас Torch є основним модулем для написання нейронної мережі та включає в себе змінні для створення параметрів класу Net. Клас Dataset є модулем для завантаження MNIST датасету в модуль завантаження для ітерування об’єктів для навчання нейронної мережі. Клас Train є модулем, який при старті ініціалізує нейронну мережу та запускає навчання класифікатора.

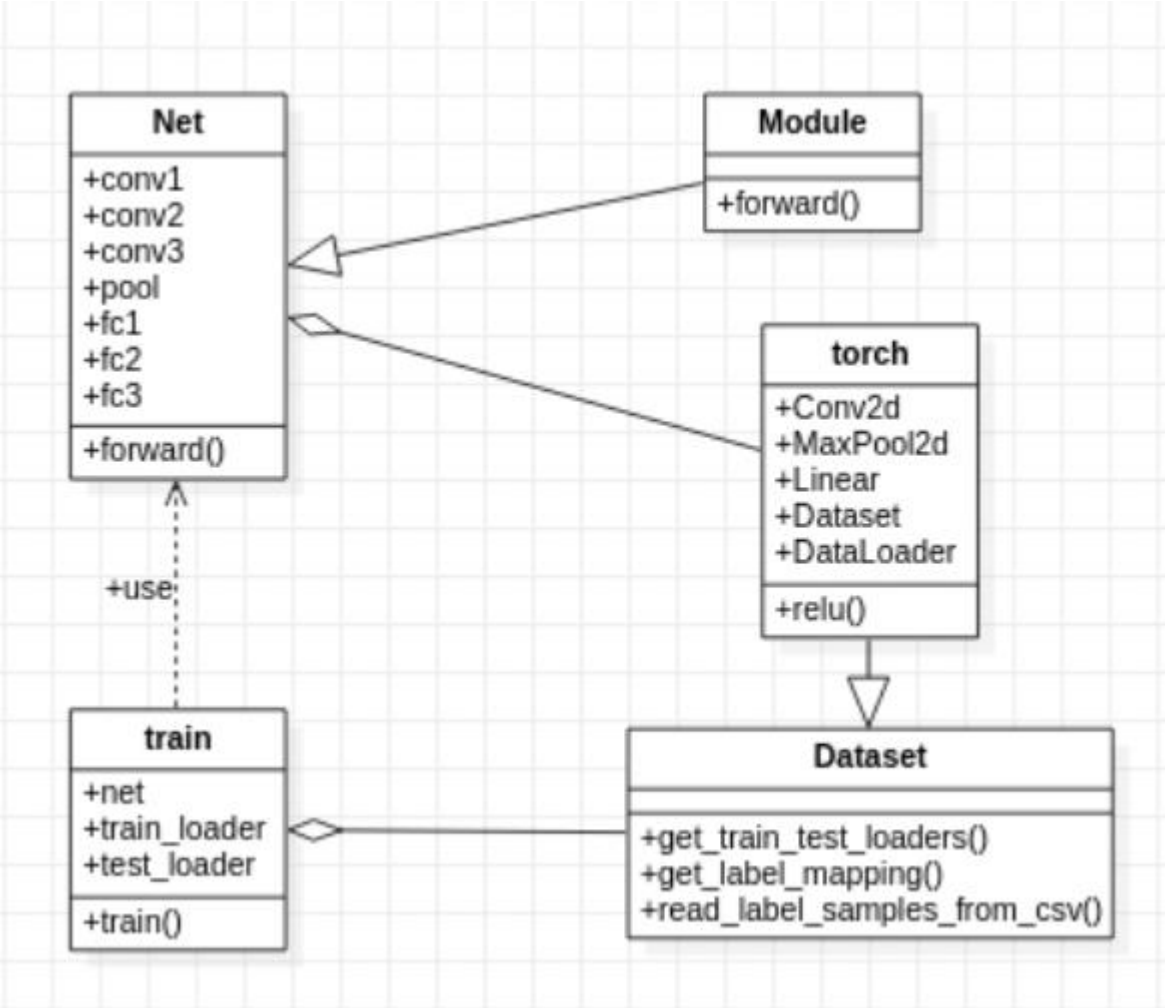

Рисунок 3.5 – Діаграма класів для процесу навчання нейронної мережі розпізнавання символів українського алфавіту

На рисунку 3.6 зображено діаграму класів для процесу старту системи розпізнавання. Клас Camera є модулем, який запускає камеру та систему розпізнавання, клас cv2 є модулем, який відповідає за захоплення камери, клас InferenceSession є модулем для запуску роботи класифікатора нейронної мережі, клас Image є класом для переводу отриманого класифікатором жесту в зображення з масива даних, клас ImageDraw є класом для редагування отриманого зображення, клас ImageFont є класом для показу розпізнаного символу (букви) українського алфавіту нейронною мережею в інтерфейсі користувача в режимі реального часу.

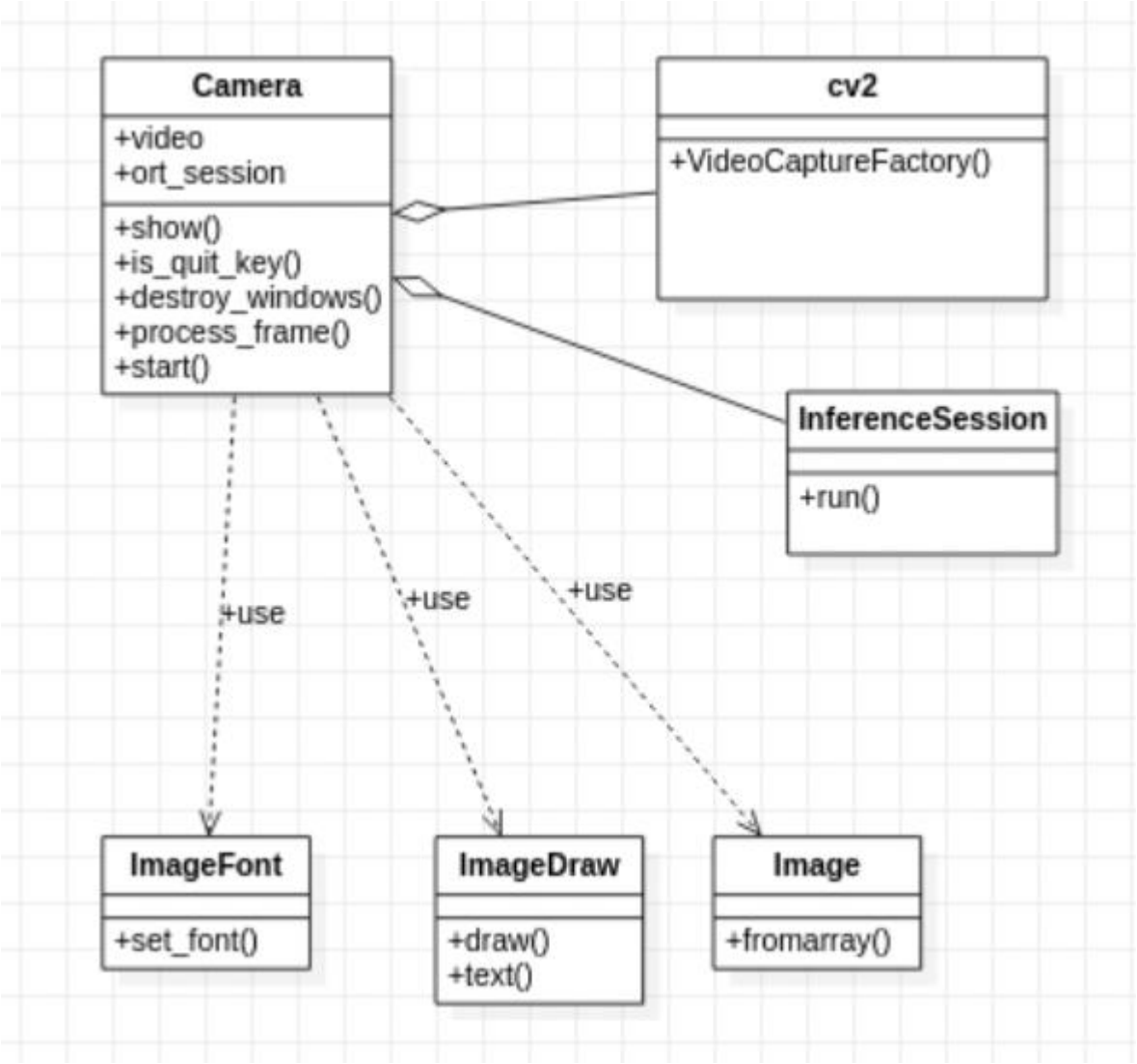

Рисунок 3.6 – Діаграма класів для процесу старту системи розпізнавання символів українського алфавіту нейронною мережею

Для більш розгорнутого представлення методів, які будуть проходити в системі розпізнавання символів українського алфавіту нейронною мережею деякі методи були представлені у вигляді блок-схем. Це найбільш поширений та ефективний спосіб представлення методів та процесів, який використовується в процесі створення програмного забезпечення.

Представлення методів у вигляді блок-схем необхідне для того, щоб найбільш ефективно зобразити їх етапі та швидко орієнтуватися в процесі написання програмного коду.

На рисунку 2.7 зображена блок-схема методу навчання нейронної мережі.

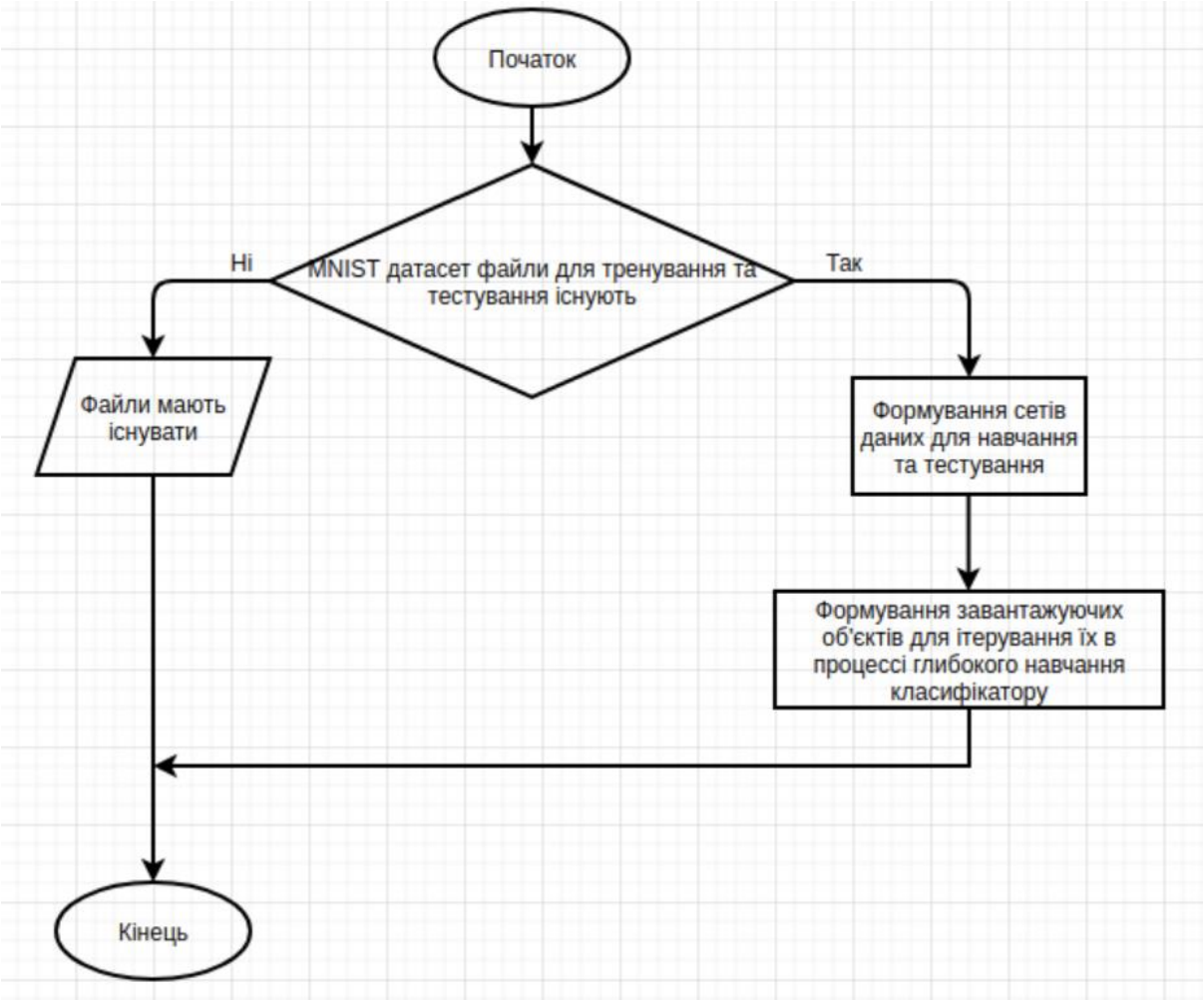

Рисунок 3.7 – Блок-схема методу навчання нейронної мережі

На рисунку 3.8 зображена блок-схема методу створення власного MNIST датасету з певним розширенням файлу, а саме CSV. Такий метод необхідний в програмі, так як для порівняння зображення мають переводитися в даний формат даних для майбутнього інтегрування в процес навчання класифікатора. Файл з розширенням CSV заповнюється даними отриманих з фотокарток у саб-директоріях для подальшого використання.

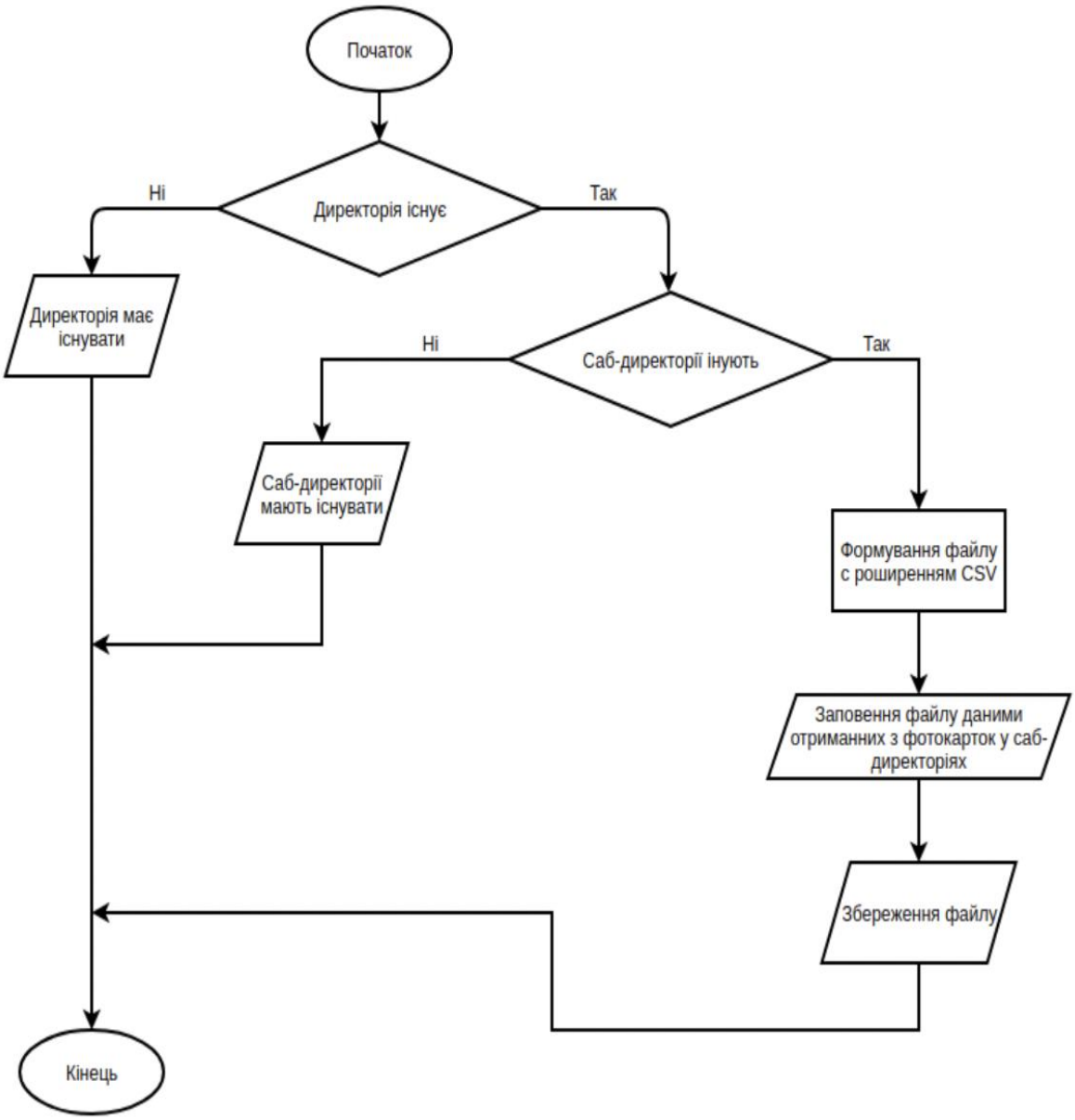

Рисунок 3.8 – Блок-схема методу створення MNIST датасету

На рисунку 3.9 зображено блок-схему методу оцінювання та валідації класифікатора жестів, тобто на скільки відсотків класифікатор розпізнає тестові дані після процесу навчання. Для цього класифікатор спочатку перевіряється на даних, які були застосовані під час тренування нейронної мережі, після цього класифікатор оцінюється на тестових даних, та загальна оцінка класифікатора стає доступна користувачу системи.

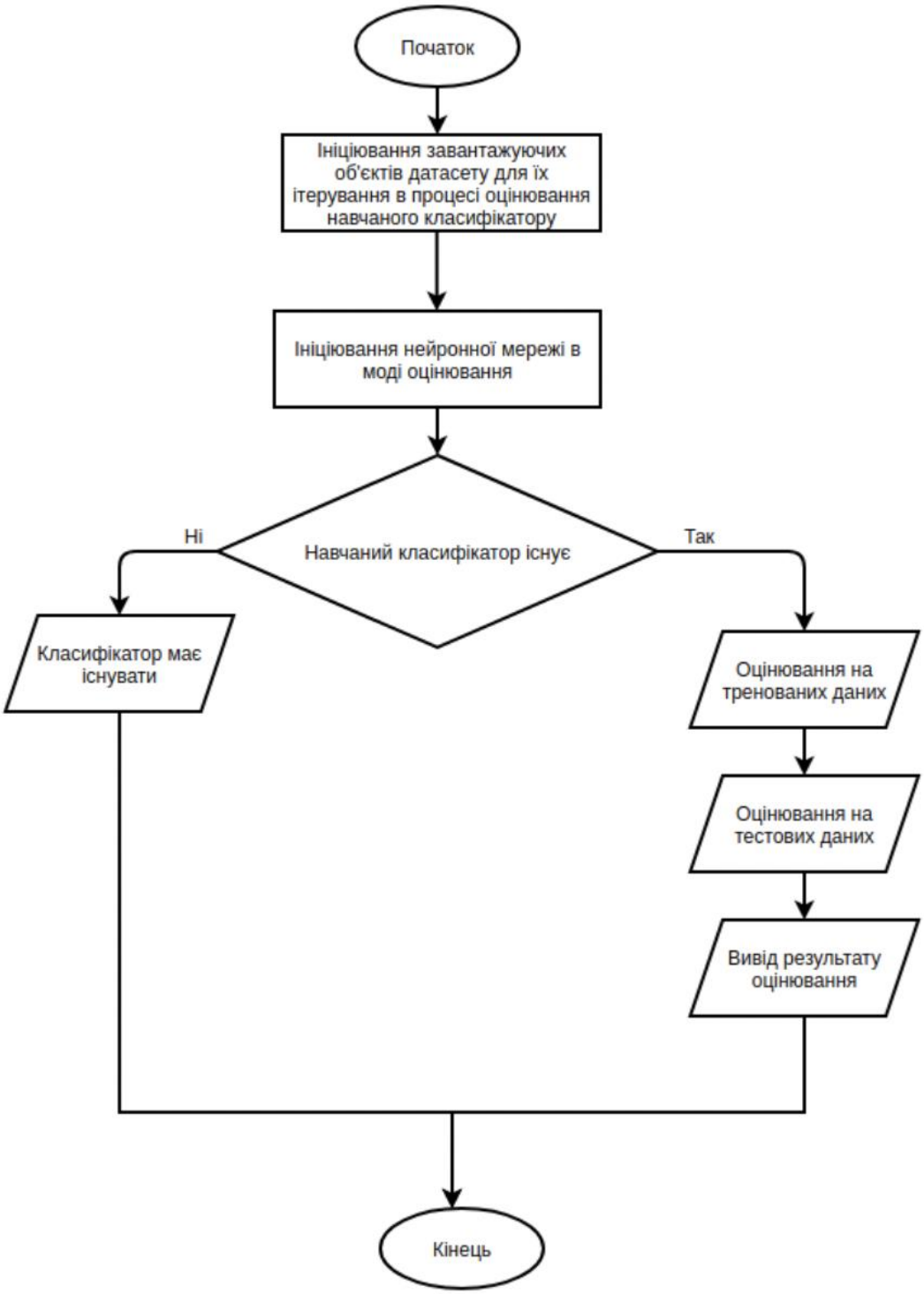

Рисунок 3.9 – Блок-схема методу оцінювання класифікатора жестів

3.4 Результати застосування технології розпізнавання жестів алфавіту української мови з використанням нейронної мережі

В цьому підрозділі будуть наведені приклади роботи системи розпізнавання жестів алфавіту української мови.

На рисунку 3.10 зображений результат розпізнавання нейронної мережі розпізнавання жестів української мови, а саме літери “А”.

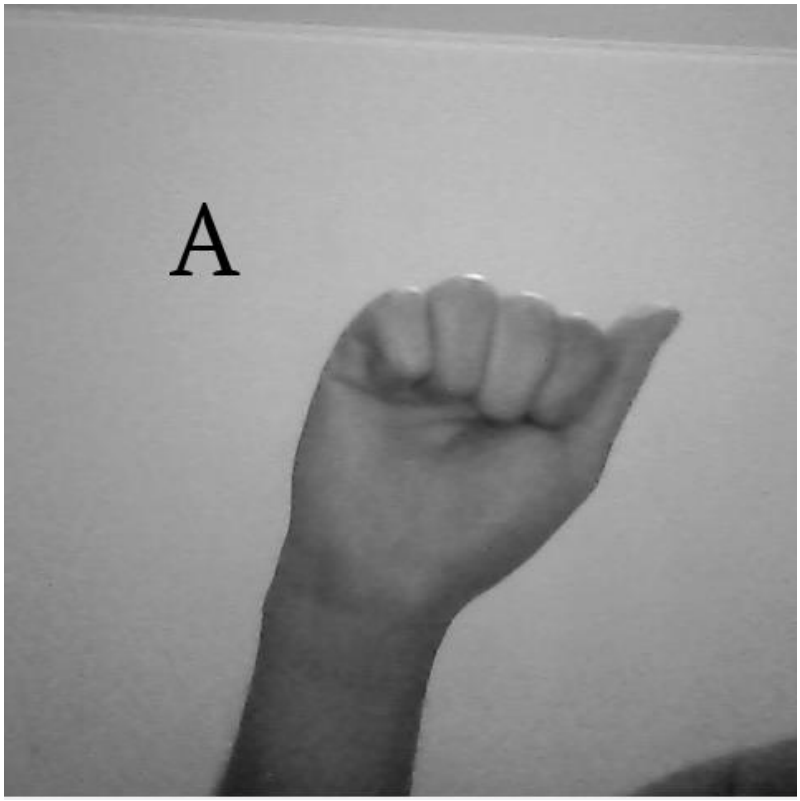

Рисунок 3.10 – Розпізнавання нейронною мережею літери “А”

На рисунку 3.11 зображений результат розпізнавання нейронної мережі розпізнавання жестів української мови, а саме літери “Е”.

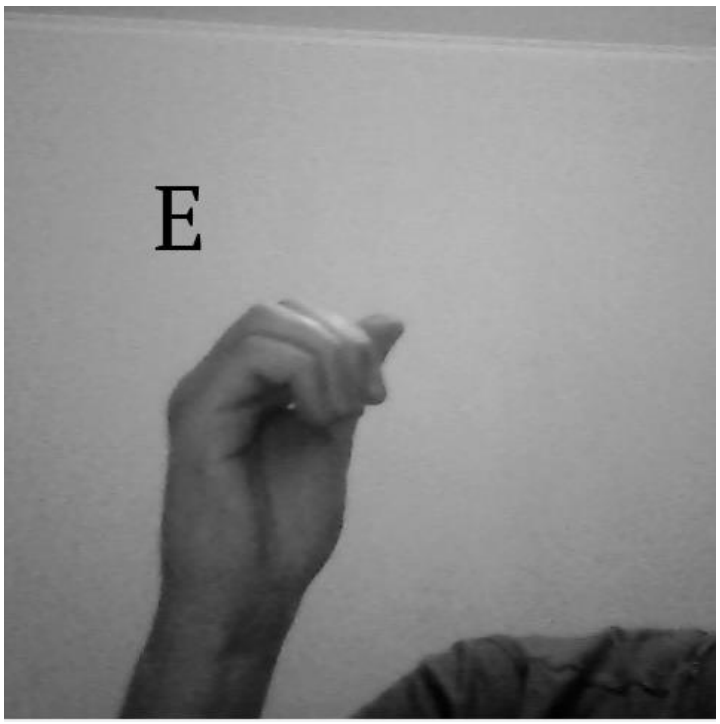

Рисунок 3.11 – Розпізнавання нейронною мережею літери “Е”

На рисунку 3.12 зображений результат розпізнавання нейронної мережі розпізнавання жестів української мови, а саме літери “З”.

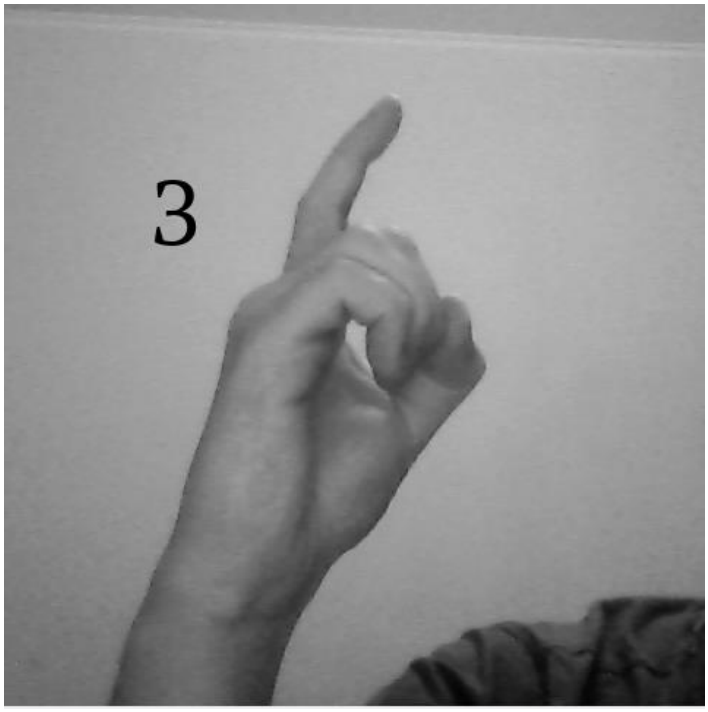

Рисунок 3.12 – Розпізнавання нейронною мережею літери “З”

На рисунку 3.13 зображений результат розпізнавання нейронної мережі розпізнавання жестів української мови, а саме літери “Ї”.

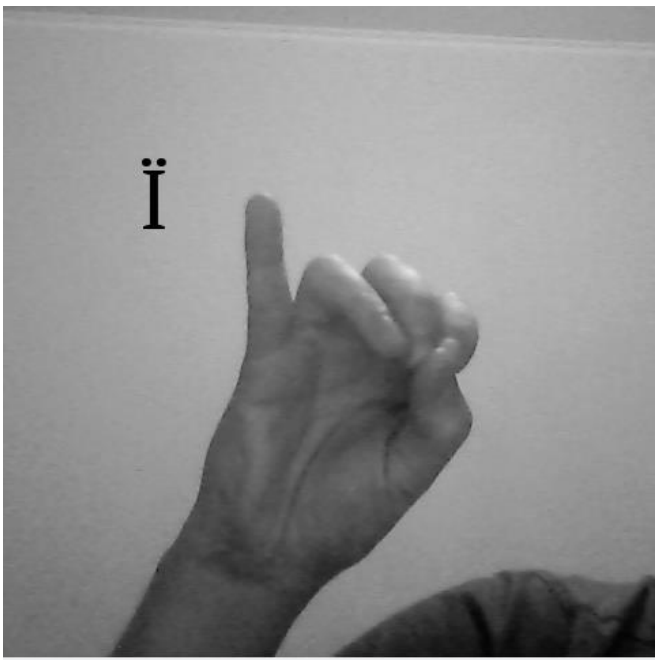

Рисунок 3.13 – Розпізнавання нейронною мережею літери “Ї”

На рисунку 3.14 зображений результат розпізнавання нейронної мережі розпізнавання жестів української мови, а саме літери “Щ”.

Рисунок 3.14 – Розпізнавання нейронною мережею літери “Щ”

На рисунку 3.15 зображений результат розпізнавання нейронної мережі розпізнавання жестів української мови, а саме літери “Ю”.

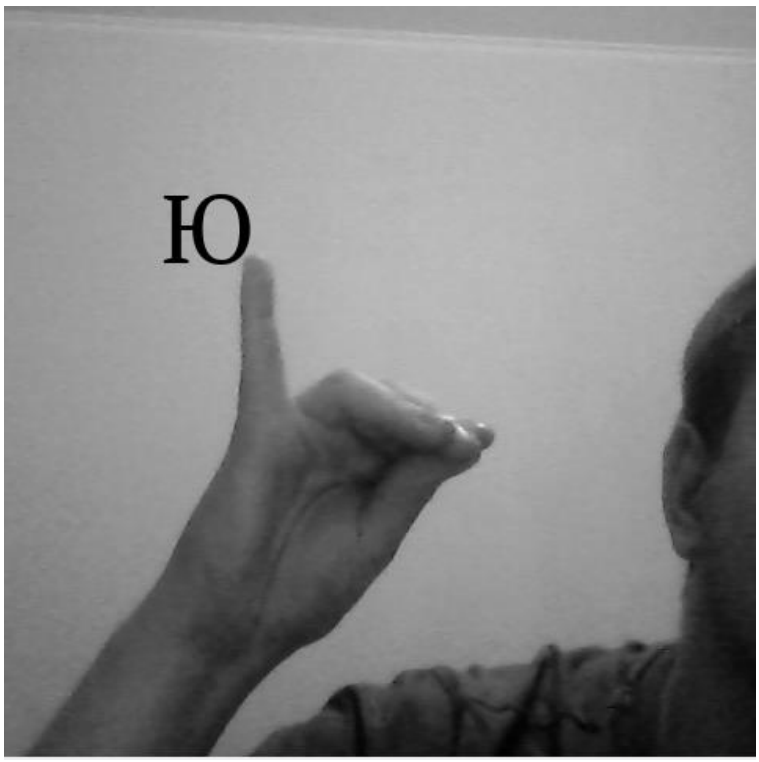

Рисунок 3.15 – Розпізнавання нейронною мережею літери “Ю”

3.5 Тестування системи розпізнавання жестів українського алфавіту з використанням нейронної мережі

Тестування нейронної мережі є дуже важливим етапом для перевірки якості представленої системи та для того, щоб зробити висновок про її актуальність та внесення покращення до систем-аналогів, які існують на ринку. Для того, щоб протестувати класифікатор, в системі є окремий модуль, завдяки якому класифікатор жестів спочатку перевіряється на наборі даних, на якому проводилися тренування та навчання, а потім на наборі тестових даних, які є новими для класифікатора. Але для представлення повної картини було проведено додаткове дослідження, в ході якого були протестовані тільки нові дані для того, щоб

зрозуміти, як саме буде працювати система в реальних ситуаціях. Середній час обробки одного фрейму, а саме зображення жесту, яке формується системою в режимі реального часу, становить 0.08 секунди. Точність розпізнавання становить 94.3%, що співпадає з вимогами до системи. При підрахунку даних оцінок необхідно враховувати різні типи спотворень, а саме віддаленність, освітлення та дозвіл камери.

Оцінка помилки першого роду, яка означає, що жест не розпізнаний та правильна нульова гіпотеза не прийнята, становить 3.8%, а оцінка помилки другого роду, яка означає, що жест розпізнаний неправильно та прийнята неправильна нульова гіпотеза, становить 1.9%.

Результати перевірки нейронної мережі для розпізнавання жестів, що позначають символи українського алфавіту на помилки першого та другого роду, зображені в таблиці 3.1.

Таблиця 3.1 – Результати перевірки нейронної мережі для розпізнавання жестів символів українського алфавіту

Жест розпізнаний, відсотки	Жест не розпізнаний (помилка першого роду), відсотки	Жест розпізнаний неправильно (помилка другого роду), відсотки	Середній час обробки одного фрейму, секунди
94.3	3.8	1.9	0.08

На рисунку 3.16 зображено результати оцінювання навченого класифікатору розпізнавання жестів модулем оцінювання досліджуваної системи. Досліджувана нейронна мережа отримала результат точності навчання 95,7% та точності тестування (перевірки) 94,3%. Ця різниця між показниками точності навчання та тестування повідомляє нам про те, що навчений класифікатор розпізнавання жестів надмірно навчений. Це означає, що досліджувана система замість того, щоб запам’ятовувати загальні патерни жестів навчилася їх розрізняти.

```
========== PyTorch ==========
Training accuracy: 95.7
Validation accuracy: 94.3
(venv) chesterfield@chesterfield:~/dev/master-degree-diplomas/mykyta/sign-language-translator/src$
```

Рисунок 3.16 – Результати оцінювання класифікатору розпізнавання жестів модулем оцінювання досліджуваної системи

Отже, після проведення тестування нейронної мережі можна стверджувати, що нейронна мережа повністю відповідає поставленим вимогам, є ефективною та унікальною.

3.6 Висновки

В ході написання третього розділу магістерської роботи була описана та реалізована система розпізнавання жестів, які означають букви українського алфавіту, у відеопотоці камери, з використанням згорткової нейронної мережі.

Було спроектовано структури головного пакету системи та розгорнуту структуру пакету, який відповідає за збереження тестового набору даних та набору даних для тренування нейронної мережі.

Представлено уявлення процесів роботи нейронної мережі, а саме діаграми діяльності для процесів розпізнавання жестів та оцінювання навченого класифікатора.

Представлені діаграми класів для більш якісного написання програмного коду, їхня структура, змінні та методи, а також взаємозв’язок. Спроектовані блок-схеми методів для більш зручної їхньої реалізації.

Представлені результати застосування технології розпізнавання жестів символів українського алфавіту з використанням нейронної мережі, а також результати тестування класифікатору та перевірки відповідності поставленим функціональним вимогам.

ВИСНОВКИ

В ході написання магістерської роботи була проаналізована, спроектована та реалізована система розпізнавання жестів, які означають символи українського алфавіту, з використанням нейронної мережі згорткового типу.

В першому розділі магістерської роботи було проаналізовано аналоги та виділені основні функції, які повинна реалізовувати система, після чого доданий функціонал, якого не вистачає аналогам, для того, щоб розробити унікальну, максимально зручну та ефективну систему.

В другому розділі магістерської роботи була розглянута та спроектована нейронна мережа для розпізнавання жестів, які означають символи українського алфавіту. Поставлені мета та задачі дослідження нейронної мережі для розпізнавання жестів українського алфавіту, сформована цільова аудиторія, інформаційні потоки системи. Описані типи користувачів системи, функціональні та нефункціональні вимоги до нейронної мережі та системи розпізнавання загалом. Змодельована діаграма прецедентів. Виявлені загальні характеристики для проведення наукового дослідження: подані визначення нейронної мережі та штучної нейронної мережі, визначена архітектура та характеристики нейронної мережі. Описаний стек бібліотек та технологій, спроектований інтерфейс користувача та логічне уявлення про систему розпізнавання жестів.

В третьому розділі магістерської роботи була реалізована система розпізнавання жестів, які означають символи українського алфавіту, у відеопотоці камери, з використанням нейронної мережі згорткового типу.

Було спроектовано структури головного пакету системи. Представлено уявлення процесів роботи нейронної мережі, а саме діаграми діяльності для процесів розпізнавання жестів та оцінювання навченого класифікатора. Представлені діаграми класів, їхня структура, змінні та методи, а також взаємозв’язок. Спроектовані блок-схеми методів для більш зручної їхньої реалізації.

Представлені результати застосування технології розпізнавання жестів символів українського алфавіту з використанням нейронної мережі, а також результати тестування нейронної мережі та перевірки відповідності поставленим вимогам.

Отже, можна вважати, що нейронну мережу реалізовано, вимоги до неї дотримані, а мету магістерської роботи досягнуто.

ПЕРЕЛІК ПОСИЛАНЬ

1. Індустрія 4.0 [Електронний ресурс] – Режим доступу до ресурсу: https://trends.rbc.ru/trends/industry/6135d8a69a7947149594c83a

2. Міщенко М.Д., Нестерюк О.Г. Дослідження розпізнавання жестів української мови у режимі реального часу // Project, Program, Portfolio Management. РЗМ-2021: Тези доповідей VI Міжнародної науково- практичної конференції : [у 2т.]. // Відповідальний за випуск П.О. Тесленко — Том 1. — Одеса. : ІШИР, 2021 – С. 100–102.

3. Real-time gesture recognition based on feature recalibration network with multi-scale information / Z. Cao, X. Xu, B. Hu, M. Zhou, Q. Li // Neurocomputing. – 2019. – Vol. 347. – P. 119–130

4. Библиотека Keras. Слой пуллинга. – URL: https://keras.io/layers/pooling/ (дата обращения: 12.12.2019).

5. Библиотека Keras. Полносвязный слой. – URL: https://keras.io/layers/core/ (дата обращения: 12.12.2019).

6. Библиотека Keras. Model class API. – URL: https://keras.io/models/model/ (дата обращения: 12.12.2019).

7. Основы планирования эксперимента: методическое пособие / сост. К.М. Хамханов. – Улан-Удэ, 2001. – URL: http://window.edu.ru/resource/438/18438/files/Mtdukm8.pdf (дата обращения: 12.12.2019).

8. Система распознавания жестов [Електронний ресурс] – Режим доступу до ресурсу: http://www.atanor.ru/engineering/systema-raspoznavania-zesto

9. Сурдофон [Електронний ресурс] – Режим доступу до ресурсу: https://te-st.ru/entries/surdofon/

10. CNNGestureRecognizer [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/asingh33/CNNGestureRecognizer

11. Krizhevsky, A. Imagenet classification with deep convolutional neural networks / Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton // NIPS. — 2012. — 1106 -1114 p.

12. Yann LeCun Leon Bottou, Y. B. Gradient-based learning applied to document recognition / Yoshua Bengio Yann LeCun, Leon Bottou, Patrick Haffner // IEEE. — 1998.

13. Python [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Python_(programming_language)

14. PyCharm [Електронний ресурс] / JetBrains – Режим доступу: https://www.jetbrains.com/ru-ru/pycharm/.

ДОДАТОК А

КОПІЇ ДОКУМЕНТІВ ПРО АПРОБАЦІЮ РЕЗУЛЬТАТІВ РОБОТИ

УДК 005.8

ISSN 2522-9435

ОРГАНІЗАТОРИ

Національний університет «Одеська політехніка»
Кафедра проектного навчання в ІТ НУ«ОП»
Українська асоціація управління проектами
Кафедра ЮНЕСКО НУ«ОП»
Politechnicka Opolska
Українське науково-освітнє ІТ товариство

Матеріали публікуються за оригіналами, що подані авторами.
Претензії щодо змісту та якості матеріалів не приймаються.

Відповідальний за випуск:
Тесленко Павло Олександрович

Project, Program, Portfolio Management. P3M-2021: Тези доповідей VI Міжнародної науково-практичної конференції : [у 2т.]. // Відповідальний за випуск П.О. Тесленко — Том 1. — Одеса. : ІШІР, 2021. – 212 с.

Project, Program, Portfolio Management. P3M-2021: The Proceedings of the International Research Conference, 03 – 04 Desember, 2021, Odesa, Ukraine, 212 p.

У збірнику наведені матеріали VI Міжнародної науково-практичної конференції "Project, Program, Portfolio Management. P3M-2021". Збірник становить інтерес для студентів, викладачів, наукових працівників та фахівців з управління проектами.

Рекомендовано до видання рішенням Вченої ради Інституту штучного інтелекту та робототехніки
№ 2 від 11.11.21

УДК 005.8

ISSN 2522-9435

5. Сова І. М., Сіденко Є. В. Використання загорткової нейромережі для сегментації новоутворень на знімках МРТ головного мозку. // Матеріали XXI Всеукр. наук. – практ. конф.: тези доповідей: Комп'ютерні науки. Технічні науки. (11-16 листопада 2019 р., м. Миколаїв). – Миколаїв: Видавництво ЧНУ ім.. Петра Могили, 2019. 55-61.

6. Krizhevsky A., Sutskever I., Hinton G. ImageNet classification with deep convolutional neural networks // Advances in Neural Information Processing Systems. – 2012. – Pp. 1097–1105.

7. Кривенко Д. М. Аугментація навчальної вибірки для навчання U-NET мережі для задачі сегментації металографічних зображень // Матеріали X Міжнародної наукової конференції студентів та молодих вчених «Сучасні інформаційні технології - 2020» (14-15 травня 2020 р., м. Одеса). – Одеса: Наука і техніка, 2020. С. 68-70.

8. Krylov V. N., Volkova N. P. Vector-difference texture segmentation method in technical and medical express diagnostic systems // Herald of Advanced Information Technology. – 2020. Vol. 3, No. 4. – P. 174 – 186.

ДОСЛІДЖЕННЯ РОЗПІЗНАВАННЯ ЖЕСТІВ УКРАЇНСЬКОЇ МОВИ У РЕЖИМІ РЕАЛЬНОГО ЧАСУ

магістр Міщенко М.Д., кандидат технічних наук Нестерюк О.Г.
Національний університет «Одеська політехніка», Україна

В роботі представлено результати дослідження розпізнавання жестів, що позначають символи алфавіту української мови в режимі реального часу. Проаналізовано аналоги системи, знайдено особливості її реалізації. Обраний алгоритм розпізнавання жестів української мови в режимі реального часу та способи реалізації подібної системи. Обґрунтований вибір створення системи за допомогою згорткової нейронної мережі та глибинного навчання. Приведено приклад та пояснення в якій сфері життя можливе використовування нейронної мережі дослідженої в цій роботі.
Ключові слова: *розпізнавання жестів, згорткова нейронна мережа, розпізнавання в режимі реального часу, мова жестів*

Для того, щоб забезпечити велику швидкодію та відмовостійкість нейронної мережі, що розробляється, було запропоновано алгоритм з основою у вигляді сітки з нейронною мережею, яка має невелику архітектуру. Основою згорткової нейронної мережі є шари згортки [1]. Кожному шару нейронної мережі належать фільтри для кожного каналу, які мають обробляти попередній шар не повністю, а по частинами, а саме за допомогою підсумовування фрагментів у вигляді матриць. Наприкінці шару згортки завжди стоїть функція активації [2]. Для того, щоб передати інформацію на інший шар, необхідно використовувати функцію активації, яка перетворює обчислену інформацію для передачі у вигляді, віповідному стандартам наступного шару. Значення на виході залежить від функції активації і може бути як дійсним, і цілим [3]. Це є показником, наскільки активований нейрон поточного шару. Так як кожному фільтру згортки відповідає карта ознак, це дозволяє нейронній мережі навчитися виділяти ознаки незалежно від їх розташування у вхідному зображенні.

Пулінг можна витлумачити так: якщо на попередній операції згортки були виявлені деякі ознаки, то для подальшої обробки настільки докладне зображення вже не потрібно, і воно зменшується в розмірності, тобто ущільнюється менш докладно. Фільтрація вже

непотрібних деталей зменшує перенавчання. Шар пуллінга, як правило, вставляється після шару згортки перед шаром наступного згортки. За рахунок шару пулінга мережа стає найбільш стійкою до змін вхідного зображення, наприклад, його зсувів. Також зменшується розмірність наступних верств [4]. Повнозв'язковий шар (багатошаровий перцептрон) – прихований шар, з'єднаний з усіма нейронами попереднього шару.

Останнім шаром багатошарового перцептрону є один або кілька нейронів, кількість яких дорівнює кількості класів. Простіше кажучи, на вхід усієї згорткової нейронної мережі подається зображення, а на виході мережа видає клас, до якого це зображення відноситься [5]. Згорткові нейронні мережі забезпечують часткову стійкість до змін масштабу, зсувів, поворотів, зміни ракурсу та інших спотворень. Загальну топологію зображено на рисунку 1.

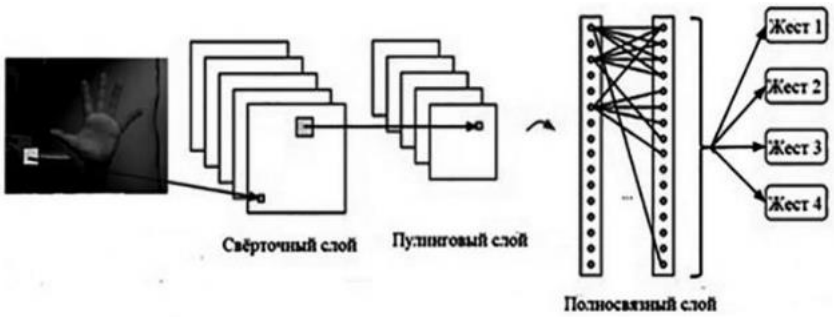

Рис. 1. Загальна структура нейронної мережі

При виборі підходу потрібно зосередитися на основній меті, задля якої розроблюється нейронна мережа, знайти підходи та задачі, які вони вирішують, після чого проаналізувати переваги та недоліки кожного підходу.

Для реалізації нейронної мережі з ціллю розпізнавання жестів, які позначають символи українського алфавіту теоретично можна використовувати такі алгоритми як K-найближчих сусідів, прихованої Марківської моделі, метод опорних векторів, метод ансамблю, а також метод глибинного навчання.

Для розпізнавання жестів активно використовується алгоритм K-найближчих сусідів, який неодмінно треба взяти до уваги при оцінці підходу. Даний алгоритм класифікує вхідні дані по відстані.

Іншим алгоритмом, який використовується при розпізнаванні жестів, є алгоритм прихованої Марківської моделі, який використовує алгоритм максимізації очікувань та розширює його.

Метод опорних векторів є класифікатором, який використовується для невеликих наборів даних, так як кількість векторів підтримки звеличується лінійно відповідно до розміру навчальної виборки.

Метод ансамблю є методом, який широко використовується та не потребує великої кількості даних для навчання.

Метод глибинного навчання є найбільш високо-рівневою та розвиненою технологію при розпізнаванні жестів. Такі нейронні мережі використовуються при

розпізнаванні жестів, обличь, зображень та мовлення. Найбільш популярними є згорткові нейронні мережі, які мають тенденцію до роботи з глибинними архітектурами.

Отже, можна зробити висновок, що найкращим підходом для розробки нейронної мережі з ціллю розпізнавання жестів буде використання глибинного навчання, а саме згорткових нейронних мереж.

Для досягнення мети дослідження необхідно створити систему, яка в режимі реального часу буде перетворювати жести, що записуються через веб-камеру, на символи українського алфавіту. Система має бути точною та швидкою для того, щоб всі жести було класифіковано. Системи-аналоги мають дуже обмежений функціонал та мають бути розширені та покращені для реалізації нейронної мережі, запланованої в дослідженні.

Роботу можливо використовувати в розробці власної нейронної мережі, яка буде мати функціонал розпізнавання жестів алфавіту української мови в режимі реального часу, що забезпечить розуміння мови людей з вадами апарату мовлення та слуху. На даний момент в Україні налічується близько 50 тисяч людей, які мають порушення слуху або мовлення та спілкуються між собою за допомогою мови жестів. За оцінками ВООЗ, до 2050 року майже 2,5 млрд людей страждатимуть від проблем зі слухом тією чи іншою мірою, з них 700 млн буде потрібно реабілітація через інвалідизуючу втрату слуху. Незважаючи на стрімке поширення цифрових форм комунікації, люди з порушеннями слуху досі зазнають певних бар'єрів при живому спілкуванні [6].

Отже, можна виявити тенденцію до того, що з кожним роком підвищується необхідність створення програмного забезпечення, яке б допомогало налагуджувати комунікацію людей з вадами апарату мовлення або слуху з людьми, яких цих вад не мають.

ДЖЕРЕЛА

1. Real-time gesture recognition based on feature recalibration network with multi-scale information / Z. Cao, X. Xu, B. Hu, M. Zhou, Q. Li // Neurocomputing. – 2019. – Vol. 347. – P. 119–130

2. Библиотека Keras. Слой пуллинга. – URL: https://keras.io/layers/pooling/ (дата звернення: 12.11.2021).

3. Библиотека Keras. Полносвязный слой. – URL: https://keras.io/layers/core/ (дата звернення: 12.11.2021).

4. Библиотека Keras. Model class API. – URL: https://keras.io/models/model/ (дата звернення: 12.11.2021).

5. Основы планирования эксперимента: методическое пособие / сост. К.М. Хамханов. – Улан-Удэ, 2001. – URL: http://window.edu.ru/resource/438/18438/files/Mtdukm8.pdf (дата звернення: 12.11.2021).

6. Deafness and hearing loss. – URL: https://www.who.int/ru/news-room/fact-sheets/detail/deafness-and-hearing-loss (дата звернення: 06.06.2021).

ДОДАТОК Б
ЛІСТИНГ ПРОГРАМНОГО КОДУ

Код файлу csv.py:

```
import re

import numpy as np
import cv2
import os

TRAIN_DIR = '/home/chesterfield/Downloads/data/train'
TEST_DIR = '/home/chesterfield/Downloads/data/validation'
SRC = '/home/chesterfield/dev/master-degree-diplomas/mykyta/sign-language-translator/src'
OUTPUT_FILE_PATH = os.path.join(SRC, 'output.csv')
POSSIBLE_NUMBERS = list(range(350))

if __name__ == "__main__":
  sub_folders = []
  for dir_, sub_dirs, files in os.walk(TRAIN_DIR):
    sub_folders.extend(sub_dirs)
    for folder in sub_folders:
      for img in os.listdir(os.path.join(TRAIN_DIR, folder)):
        number_name = re.findall(r'\d+', img)[0]
        if int(number_name) not in POSSIBLE_NUMBERS:
          continue

        img_array = cv2.imread(os.path.join(os.path.join(TRAIN_DIR, folder), img), cv2.IMREAD_GRAYSCALE)
        img_array = img_array.flatten()
```

```
            img_array = img_array.reshape(-1, 1).T
            with open('output.csv', 'ab') as f:
                label = folder + ","
                f.write(label.encode())
                np.savetxt(f, img_array, fmt="%d", delimiter=",")
```

Код файлу train.py:

```
from torch.autograd import Variable
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import torch

from dataset import get_train_test_loaders

class Net(nn.Module):
  def __init__(self):
    super(Net, self).__init__()
    self.conv1 = nn.Conv2d(1, 6, 3)
    self.pool = nn.MaxPool2d(2, 2)
    self.conv2 = nn.Conv2d(6, 6, 3)
    self.conv3 = nn.Conv2d(6, 16, 3)
    self.fc1 = nn.Linear(16 * 6 * 6, 120)
    self.fc2 = nn.Linear(120, 48)
    self.fc3 = nn.Linear(48, 34)

  def forward(self, x):
    x = F.relu(self.conv1(x))
    x = self.pool(F.relu(self.conv2(x)))
    x = self.pool(F.relu(self.conv3(x)))
```

```
        x = x.view(-1, 16 * 6 * 6)
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x


def main():
    net = Net().float()
    _criterion = nn.CrossEntropyLoss()
    _optimizer = optim.SGD(net.parameters(), lr=0.01, momentum=0.9)
    _scheduler = optim.lr_scheduler.StepLR(optimizer, step_size=10, gamma=0.1)

    trainloader, _ = get_train_test_loaders()
    for loop_number in range(12):
        train(net, _criterion, _optimizer, _trainloader, loop_number)
        scheduler.step()
    torch.save(net.state_dict(), "checkpoint.pth")

def train(net, criterion, opt, loader, epoch):
    ls = 0.0
    for i, data in enumerate(loader, 0):
        inputs = Variable(data['image'].float())
        labels = Variable(data['label'].long())
        opt.zero_grad()
        outputs = net(inputs)
        loss = criterion(outputs, labels[:, 0])
        loss.backward()
        opt.step()
```

```
        ls += loss.item()
        if i % 100 == 0:
            print('[%d, %5d] loss: %.6f' % (epoch, i, ls / (i + 1)))
```

Код файлу camera.py:

```
import cv2
import numpy as np
import onnxruntime as ort
from PIL import Image, ImageDraw, ImageFont

def center_crop(frame_img):
    h, w, _ = frame_img.shape
    s = abs(h - w) // 2
    if h > w:
        return frame_img[s: s + w]
    return frame_img[:, s: s + h]

def main():
    index_to_letter = list("АБВГҐДЕЄЖЗИІЇЙКЛМНОПРСТУФХЦЧШЩЬЮЯ")

    # create runnable session with exported model
    ort_session = ort.InferenceSession("signlanguage.onnx")

    cap = cv2.VideoCapture(0)
    while True:
        ret, frame_img = cap.read()
        frame_img = center_crop(frame_img)
        frame_img = cv2.cvtColor(frame_img, cv2.COLOR_RGB2GRAY)
        x = cv2.resize(frame_img, (32, 32))
        x = (x - mean) / std
```

```
    x = x.reshape(1, 1, 32, 32).astype(np.float32)
    y = ort_session.run(None, {'input': x})[0]

    index = np.argmax(y, axis=1)
    letter = index_to_letter[int(index)-1]

    # set cyryllic character to image with specified font
    img_pil = Image.fromarray(frame_img)
    draw = ImageDraw.Draw(img_pil)
    font = '../fonts/cyryllic.ttf'
    font_text = ImageFont.truetype(font=font, size=62, encoding='utf-8')
    draw.text((100, 100), letter, font=font_text)
    img = np.array(img_pil)
    cv2.imshow("Sign Language Translator", img)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

ДОДАТОК В

ЖЕСТИ АЛФАВІТУ УКРАЇНСЬКОЇ МОВИ