Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження та порівняння методів глибокого багатозадачного навчання для класифікації рухів людини
(тема)

Виконав:
здобувач другого року навчання,
групи ДСм-23-1
Жукевич О. А.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Науки про дані (Data Science)
(повна назва спеціалізації)
Керівник доц. Вітько О.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ О.В. Золотухін
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Науки про дані (Data Science)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Жукевичу Олександру Аркадійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та порівняння методів глибокого багатозадачного навчання для класифікації рухів людини

затверджена наказом університету від 22 листопада 20 24 р. № 1238Ст

2. Термін подання студентом роботи до екзаменаційної комісії 16 січня 20 25 р.

3. Вихідні дані до роботи Науково технічні публікації, дані Інтернет-джерел та наукових проектів щодо розробки та дослідження глибоких нейронних мереж, набори даних для визначення активності людини, документація Python

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Теоретичні дослідження
3) Практичні дослідження
4) Результати дослідження

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	25.11.2024	виконано
2	Аналіз предметної галузі та постановка задачі	26.11.2024	виконано
3	Дослідження попередніх робіт	29.11.2024	виконано
4	Аналіз методів вирішення поставленої задачі	03.12.2024	виконано
5	Аналіз та обробка наборів даних	05.12.2024	виконано
6	Розробка базових архітектур	09.12.2024	виконано
7	Реалізація методів багатозадачного навчання	15.12.2024	виконано
8	Написання пояснювальної записки	20.12.2024	виконано
9	Попередній захист	10.01.2025	виконано
10	Захист перед ЕК	16.01.2025	

Дата видачі завдання 25 листопада 20 24 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Вітько О.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 89 с., 21 рис., 2 табл., 3 дод., 30 джерел.

БАГАТОЗАДАЧНЕ НАВЧАННЯ, ГЛИБОКІ НЕЙРОННІ МЕРЕЖІ, МЕРЕЖІ ПЕРЕСІЧНОГО ЗШИВАННЯ, ОБМІН ПАРАМЕТРАМИ, РОЗПІЗНАВАННЯ ДІЯЛЬНОСТІ ЛЮДИНИ, РОЗПІЗНАВАННЯ ЕМОЦІЙ.

Об’єкт дослідження – нейронні мережі.

Предмет дослідження – застосування глибоких нейронних мереж для розпізнавання та класифікації діяльності людини.

Мета роботи – дослідження та порівняння методів глибокого багатозадачного навчання для визначення доцільності використання даних методів у проблемах класифікації, зокрема, при застосуванні до часових рядів.

Методи дослідження – аналіз технічної літератури та новісних досліджень в сфері глибокого навчання, експериментальний підбір архітектури та конфігурацій, аналіз результатів.

У ході цієї роботі було порівняно три різні підходи, а саме жорсткий обмін параметрами, м’який обмін параметрами та гібридний підхід, на двох наборах даних у сфері розпізнавання людської діяльності та розпізнавання емоцій – OPPORTUNITY та DEAP. Було продемонстровано, що не кожен підхід є однаково корисним. Зокрема, спостерігалися переваги використання жорсткого обміну параметрами (HPS) на обох наборах даних і мережі пересічного зшивання (CSN) лише на одному наборі даних, а обраний підхід м’якого обміну параметрами (SPS) не є оптимальним для жодної з задач.

ABSTRACT

Master’s thesis contains: 89 pp., 21 fig., 2 tabl., 3 ann., 30 references.

CROSS-STITCH NETWORKS, DEEP NEURAL NETWORKS, EMOTIONS RECOGNITION, HUMAN ACTIVITY RECOGNITION, MULTI-TASK LEARNING, PARAMETER SHARING.

The object of the research is neural networks.

The subject of the research is the application of deep neural networks for recognition and classification of human activity.

The purpose of the work is to study and compare methods of deep multitask learning to determine the potential effectiveness of using these methods in classification problems, in particular, when applied to time series.

Research methods are an analysis of technical literature and recent research in the field of deep learning, experimental selection of architecture and configurations, and analysis of results.

In this thesis, three different approaches, namely hard parameter sharing, soft parameter sharing and a hybrid approach, were compared on two datasets in the field of human activity recognition and emotion recognition, OPPORTUNITY and DEAP. It was demonstrated that not every approach is equally useful. In particular, the advantages of using HPS on both datasets and CSN on only one dataset were observed, but the chosen SPS approach was not optimal for either task.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ГНМ – глибокі нейронні мережі;

ТСН – тензорна слідова норма;

ШНМ – штучні нейронні мережі;

CNN – Convolutional Neural Network – згорткова нейронна мережа;

CSN – Cross-Stitch Network – мережі пересічного зшивання;

HAR – Human Activity Recognition – розпізнавання людської активності;

HPS – Hard Parameter Sharing – жорсткий обмін параметрами;

LOESS – Locally Estimated Scatterplot Smoothing – локально оцінене згладжування діаграми розсіювання;

LSTM – Long Short-Term Memory – довга короткочасна пам’ять;

MLP – Multi-Layer Perceptron – багатошаровий персептрон;

MTL – Multi Task Learning – багатозадачне навчання;

RNN – Recurrent Neural Network – рекурентна нейронна мережа;

SPS – Soft Parameter Sharing – м’який обмін параметрами;

STL – Single Task Learning – однозадачне навчання.

ВСТУП

Глибокі нейронні мережі (ГНМ) стали революційним проривом у галузі машинного навчання, демонструючи значні переваги порівняно з традиційними підходами. Особливо значущим є внесок ГНМ у розвиток аналізу та класифікації часових рядів, отриманих з носимих сенсорів: акселерометрів, електроенцефалографів, пристроїв відстеження очей та інших датчиків. Ці технології знаходять широке застосування не лише в медичній діагностиці та моніторингу стану здоров’я, але й у несподіваних галузях, таких як індустрії розваг та анімації, де аналіз руху та поведінки людини відіграє ключову роль у створенні реалістичних цифрових персонажів, та сфері побутових технологій, де вони інтегровані в смарт-пристрої, що стали невід’ємною частиною нашого повсякденного життя.

Проте, впровадження ГНМ стикається з певними обмеженнями. Однією з найсуттєвіших перешкод є потреба у великих обсягах якісних навчальних даних, збір яких є складним і, часто, економічно недоцільним проєктом. Це призводить до нерівномірного прогресу в різних сферах застосування ГНМ.

Так, підхід багатозадачного навчання (Multi-Task Learning, MTL) привертає все більшу увагу дослідників як перспективний підхід до подолання цих обмежень. MTL дозволяє моделям одночасно вчитися виконувати кілька пов’язаних завдань, що потенційно покращує їхню ефективність через спільне використання знань між задачами. Цей підхід особливо цінний у контексті аналізу людської поведінки та діяльності, де різні аспекти поведінки часто взаємопов’язані та можуть взаємно збагачувати навчання моделі.

Дана робота присвячена систематичному дослідженню методів багатозадачного навчання та оцінці їх ефективності в контексті аналізу людської діяльності. Особлива увага приділяється визначенню оптимальних

архітектур та підходів для різних типів даних, а також розробці рекомендацій щодо їх практичного застосування.

Метою цього дослідження в першу чергу є оцінка доцільності використання MTL, а також визначення конкретних умов та сценаріїв, де такий підхід може принести найбільшу користь.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис предметної галузі

За останні роки разом із стрімко зростаючою популярністю нейронних мереж почали з’являтися нові методи їх реалізації та роботи із ними, зумовлені еволюцією програмної та технічної складових. Так, хоч підхід багатозадачного навчання і є досить старою ідеєю, що була вперше запропонована у 1997 році, дійсної популярності він почав набувати відносно нещодавно. Сама ідея створення мережі, що має на меті вирішення одразу кількох взаємопов’язаних завдань, є досить простою і інтуітивно зрозумілою. З точки зору людини припущення, що система, яка «застосовує знання, які ми отримали під час вивчення пов’язаних завдань» [1], є кращою порівняно з ізольованими системами, є логічним, адже безпосередньо саме цей принцип і є основним при навчанні людини.

1.2 Актуальність дослідження

Актуальність роботи полягає в дослідженні підходів багатозадачного навчання та їх можливих переваг в області класифікації людської активності. Багатозадачні підходи мають широкий спектр потенційних застосувань, як в розважальній галузі, зокрема анімації, технологічній галузі при створенні засобів безпеки чи відстеження, так і в медицині або спорті, для більш ефективного моніторингу пацієнтів, виявлення падінь, аналізі фізичної активності, тощо. Саме для цього різні підходи MTL мають бути досліджені для виявлення їх практичної цінності, порівняно із стандартними методами.

В першу чергу, ця робота зосереджена на детальному та порівняльному аналізі декількох підходів класифікації рухів та емоцій людини, так як такий аналіз досі не був детально проведений.

Внесок цієї роботи головним чином полягає в порівняльній оцінці трьох основних підходів. Два з них, метод жорсткого поширення параметрів та метод м’якого поширення параметрів, є історично найпопулярнішими та найпродуктивнішими. Проте із розвитком технологій було запропоновано декілька нових методів, зокрема метод перехресного зшивання, який і буде третім суб’єктом дослідження.

1.3 Постановка задачі

Варто зазначити, що основною метою цієї роботи не є знаходження або створення моделі та архітектури, що перевершує решту у галузі. Оцінка проведених експериментів здебільшого виконується якісно, щоб зробити висновки, чи три підходи, і, отже, багатозадачне навчання, можуть допомогти в цих проблемах класифікації і, таким чином, є потенційно ефективними при застосуванні до данних цього типу, а саме – часових рядів.

2 ТЕОРЕТИЧНІ ДОСЛІДЖЕННЯ

2.1 Глибокі нейронні мережі

Хоч штучні нейронні мережі (ШНМ) і не є новою концепцією, сфера та доцільність їх застосування доволі довгий час була обмежена недостатнім розвитком технологій, що не дозволяли ШНМ бути повністю ефективними, порівняно з іншими методами вирішення задач. Проте стрімкий розвиток технологій у сфері апаратного та програмного забезпечення дозволили застосовувати не лише прості нейронні мережі, а й набагато кращі, хоч і складніші, глибокі нейронні мережі (ГНМ). Зараз терміни ШНМ та ГНМ часто використовуються як взаємозамінні, проте в контексті цього дослідження доцільно буде використовувати термін ГНМ, який можна визначити як мережу з принаймні двома прихованими шарами. Загалом, основну ідею роботи таких мереж можна порівняти з роботою нейронів в живих організмах, звідки вона безпосередньо була запозичена. Проте варто зазначити, що сучасні нейронні мережі не призначені для моделювання роботи мозку організмів і, як правило, вважаються низькоякісними моделями функціювання мозку.

ГНМ є мережею пов’язаних нейронів із входами та виходами, що об’єдані в шари. Кожен з нейронів обробляє отриману інформацію, застосовуючи до неї функцію з параметрами, де кількість вхідних нейронів є $n \in N$, а вихідних – $k \geq 1$. Вихідна інформація подається у вигляді векторів $\vec{y}_k \in R^*$, розмірність яких залежатиме від задачі. Для прикладу, при небінарній класифікації, мережа може виводити оцінки впевненості для декількох міток, що можуть містити різну кількість класів. Таки чином, мережа може бути описана як деяка (зазвичай дуже нелінейна) функція F на певній площині $X \in R^n$, що повертає наближену оцінку $\hat{Y}$.

Створення нейронної мережі зазвичай реалізовано в декілька етапів. Перший етап – вибір та реалізація архітектури мережі. Тип мережі зазвичай

обирається відповідно до задачі, що необхідно вирішити. Для задач обробки зображень зазвичай використовуються згорткові нейронні мережі, для текстових задач, таких як обробка природньої мови чи класифікація послідовностей, найкраще підходять рекурентні нейронні мережі. Проте, вибір також може залежати і від типу чи розміру даних, обчислювальних ресурсів та вимог до швидкості обробки, тощо.

Наступним етапом після реалізації архітектури мережі є етап навчання. Основна мета – зменшити похибку при виконанні завдання. Цей процес полягає в налаштуванні параметрів функцій кожного нейрона, за допомогою функції втрат, яка приймає дані, прогнозовані результати та очікувані результати як вхідні дані та повертає величину, що показує, наскільки результати відрізняються від очікуваних значень. Після цього за допомогою алгоритму оптимізації параметри нейронів можуть бути змінені так, щоб функція втрат стала мінімальною.

Основний алгоритм оптимізації, що використовується при навчанні нейронних мереж – алгоритм градієнтного спуску. Він полягає в мінімізації цільової функції шляхом ітеративного руху до мінімуму. Алгоритм працює, роблячи кроки в напрямку, протилежному градієнту (або наближеному градієнту) функції в поточній точці, так як це і є напрямком найкрутішого спуску. Алгоритм стохастичного спуску є найпростішим методом оптимізації, і на його основі побудовані новіші та продуктивніші методи, такі як метод моментума, метод адаптивного градієнтного спуску та метод розповсюдження середнього квадратичного кореня. Комбінацією цих мсетодів є метод оцінки адаптивного моменту чи Adam, що є найпоширенішим та, зазвичай, найпродуктивнішим методом, якщо швидкість навчання є важливим фактором при виборі оптимізатора.

Перед процесом навчання мережі необхідно також обрати безпосередньо тип навчання. В більшості випадків при роботі з ГНМ використовують контрольоване навчання або навчання з учителем. Так, навчальні дані подаються вже позначеними згідно з певними потребами,

відповідно до задачі. Кожний приклад містить принаймні одну правильну мітку. Тренуючись на великій кількості прикладів, модель знаходить відповідності та вивчає взаємозв'язок між вхідними даними та цільовими виходами, що дозволяє їй в подальшому відтворювати розподіл міток і класифікувати раніше невідомі дані. Після, функція втрат кількісно визначає наскільки прогноз моделі відрізняється від фактичної мітки.

Альтернативою контрольованому навчанню є неконтрольоване навчання, де мітки відсутні. При неконтрольованому навчанні мережа має знайти приховані структури або патерни в даних без явних вказівок на правильні відповіді. Така класифікація часто не відповідає вже відомому розподілу міток, тож вона може не відповідати тому, що очікує користувач. Тож, основні сфери застосування неконтрольовано навчання – кластеризація, де користувач не має можливості отримати попередньо мічені дані, або задачі генерації даних. Так як в цій роботі розглядається безпосередньо процес класифікації вже маркованих даних, для реалізації було обрано контрольоване навчання.

Процес навчання з учителем можна розділити на кілька етапів. Спочатку навчальні дані подаються до мережі на етапі прямого проходу. Кожен шар, через який проходять дані, застосовує функцію зваженого перетворення та нелінейну функцію активації. Останній шар генерує прогноз моделі. При першому проході параметри зазвичай генеруються випадковим чином, тому похибка зазвичай є досить великою, проте це дозволяє моделі зробити перший прогноз на основі її поточного стану, який потім буде порівнюватися з правильною міткою на наступному етапі.

Після обчислення функції втрат, мережа корегує свої параметри, використовуючи процес зворотнього поширення помилки [2]. Так, частково диференціюючи функцію втрат для кожного параметра в загальному параметрі Θ, тобто обчислюючи градієнт $\nabla_{\Theta} L$, оптимізатор може обчислити оновлення для кожного параметра, яке наближає втрати до нуля.

На практиці, таке навчання зазвичай відбувається шляхом надання мережі під час навчання невеликих пакетів (батчів), тобто невеликих підмножин набору даних по кілька зразків у кожній. Такий підхід поєднує переваги використання повного градієнтного спуску і стохастичного градієнтного спуску, роблячи процес навчання більш ефективним і стабільним, запобігаючи перенавчанню. Це пов'язано з тим, що подання зразків у пакетах та обчислення середнього з їх градієнтів допомагає уникати застряганню в локальному мінімумі. Альтернативні методи, наведені вище, схильні або бути занадно шумними та нестабільними через оновлення ваг після кожного зразка, або процес навчання може займати надзвичайно довгий час, особливо коли набір даних великий.

Ефективним методом керуванням навчанням та процесом оновлення є також швидкість навчання, що емпірично завдається користувачем, та не може бути змінена оптимізатором. Цей гіперпарметр визначає, наскільки величина ваг або параметрів моделі має змінюватися під час кожного кроку оновлення в процесі оптимізації. Правильний вибір швидкості навчання є критичним для успішного навчання моделі, оскільки він безпосередньо впливає на якість і швидкість навчання, а також на здатність моделі до узагальнення. Процес навчання можна описати як процес мінімізації довільної функції втрат L для мережі F з тензором параметрів Θ у вигляді:

$$\min_{\Theta \in R} L\left(\hat{Y}, Y\right) = \min_{\Theta \in R} L(F(X|\Theta), Y). \tag{2.1}$$

При використанні міні-партій, оптимізатор використовує кожну партію для обчислення градієнта та функції помилки, після чого знаходить похідну цієї помилки за кожним параметром, що наближає помилку до мінімуму. Правило оновлення тензора [3] параметрів Θ на кроці t зі швидкістю навчання η має вигляд:

$$\Theta^{(t+1)} = \Theta^{(t)} - \eta \nabla_{\Theta} L\ . \tag{2.2}$$

2.1.1 Багатошарові персептрони

Багатошарові персептрони (в подальшому MLP) є одним із основних типів глибоких нейронних мереж. Загальна формула персептрону з вхідним вектором $\vec{x} \in R^n$ (або в подальшому з вектором активації $\vec{h} \in R^n$), одним виходом y, матрицею ваг $\vec{w} \in R^n$ та коефіцієнтом зміщення вихідного нейрону $b \in R$ може бути представлена у вигляді формули:

$$y(\vec{x}|\vec{w}, b) = f(x \cdot w + b), \tag{2.3}$$

де f – нелінійна функція, яка називається функцією активації, що трансформує лінійне комбінаційне значення, отримане від суми зважених входів нейрона, у нове значення, яке буде передано наступному шару мережі. Популярні варіанти активаційної функції включають *tanh, Softmax* або *ReLU* [4].

Вибір функції активації залежить від конкретної задачі. Для простих бінарних задач класифікації часто використовується функція *Sigmoid*, для складніших багатокласових – *Softmax. ReLU* та *Leaky ReLU* зазвичай використовуються у прихованих шарах мережі для прискорення навчання та запобігання проблеми зникаючого градієнту. Для задач, де важливо обчислювати ймовірності, також часто використовують *Softmax* у вихідному шарі.

Типова архітектура мережі MLP зазвичай складається з трьох основних типів шарів. Кожен нейрон кожного шару пов’язаний з кожним нейроном попереднього шару. Перший – вхідний шар – не виконує обчислень, а лише отримує вхідні значення та передає до наступного шару. Другий тип шару – прихований. Приховані або латентні шари називаються так, бо їх входи та виходи невідомі для зовнішніх по відношенню до нейронної мережі програм та користувачів [5]. Кількість прихованих шарів та нейронів у кожному з них зазвичай визначається як гіперпараметр.

Приховані шари генерують ознаки та залежності за допомогою алгоритму зворотнього поширення. Останній шар – вихідний шар, що повертає результат роботи мережі. Кількість нейронів на виході залежить від типу завдання: для задачі класифікації з двома класами буде один нейрон, для багатокласової класифікації – кілька нейронів, а для задач регресії – один або більше нейронів.

Нейрони вихідного шару MLP виводять скалярні значення, які після цього підлягають обробці функцією *softmax*. У задачах класифікації буде один вихідний нейрон на кожен можливий клас, і зважена сума, обчислена в кожному нейроні, яка найчастіше подається до функції *softmax*, що перетворює ці значення на ймовірності того, чи належить поданий зразок до певного класу, де найвищий показник і вважається виходом мережі.

2.1.2 Згорткові нейронні мережі

Мережі MLP також часто називають повністю зв’язаними мережами, оскільки кожен нейрон пов’язаний з кожним виходом попереднього шару. Такі мережі мають велику кількість параметрів, через що процес навчання дуже глибоких мереж дещо ускладнено [6]. Ця проблема може бути вирішена при використанні згорткових нейронних мереж (в подальшому CNN) [7], що зазвичай використовуються при виконанні задач, пов’язаних з обробкою зображень.

Основна особливість CNN полягає в використанні операції згортки, що полягає в використанні фільтрів до вхідних даних. Ці фільтри часто є одно або двовимірними тензорами з кількома пікселями кожен. Вони рухаються по зображенню, обчислюючи скалярне значення для кожної позиції. Фільтри зазвичай накладаються один на одного. В результаті, виходом нейрона, як правило, буде тензор довільної форми (його також називають активаційною картою), який буде використано як вхідні дані для наступного каналу [8].

Більшість архітектур CNN також реалізують шари об’єднання чи пулінгу. Вони призначені для зменшення розміру активаційних карт зі збереженням найважливіших ознак. Для цього часто використовують оператор понижувальної максимізації дискретизації до невеликих блоків вхідного тензора, часто оператор максимізації або усереднення. Зменшення розмірності позитивно впливає на проблему перенавчання та складність обчислення.

2.1.2 Рекурентні нейронні мережі

Рекурентні нейронні мережі (в подальшому RNN) – нейронні мережі, що призначені для обробки послідовних даних, такі як текст, аудіо, відео чи часові ряди. Їх основні відмінність від інших типів нейронних мереж – здатність враховувати попередній контекст та історію вибірки. Це означає, що кожен вихід нейрона можу бути переданий як і до наступного нейрона, що реалізовано в інших типах ГНМ, так як і до цього ж нейрона.

Така архітектура реалізується за допомогою блоків з довгою короткочасною пам’яттю (LSTM) [3], [9]. Цей підхід забезпечує механізм не лише запам’ятовування даних, а і забування. Архітектура LSTM складається з чотирьох основних елементів: блок пам’яті, ворота входу, ворота забуття та ворота виходу. Ворота входу керують потоком інформації до блоку пам’яті, використовуючи сигмоїдну функцію активації, щоб визначити, яку інформацію з поточного входу слід зберегти. Ворота виходу використовують сигмоїдну функцію у поєднанні з функцією активації *tanh* для визначення, яку частина блоку пам’яті слід зберегти. Ворота забуття використовують сигмоїдну функцію для визначення даних, що мають бути видалені.

2.2 Багатозадачне навчання

Багатозадачне навчання (БЗН, MTL) – тип машинного навчання, що характеризується можливістю навчання однієї моделі для розв’язання декількох завдань одночасно. Основна ідея полягає в тому, що одночасна обробка кількох різних, проте дещо пов’язаних завдань, може покращити загальні результати мережі. Додаткові завдання можуть бути як основними за природою, де їх якість та продуктивність мають таку саму важливість як і якість основної задачі, так і додатковими, котрі мають на меті лише покращення основної задачі, і продуктивність яких не є обов’язковою [11].

2.2.1 БЗН в нейронних мережах

Попередні дослідження визначають два основні типи багатозадачного навчання [10]. Основна відмінність полягає в тому, як мережа організовує та обробляє спільні та специфічні ознаки:

– жорсткий обмін параметрами (Hard Parameter Sharing – HPS), де мережа використовує спільні шари для усіх задач, проте вихідні шари є індивідуальними для конкретних задач;

– м’який обмін параметрами (Soft Parameter Sharing – SPS), де кожна задача використовує окрему мережу, але ваги мереж частково узгоджуються.

Методи MTL в нейронних мережах вперше були сформульовані у 1998 році Річардом Каруаною [11]. У тому ж дослідженні був запропонований підхід HPS. З того часу від набув найбільшого поширення, порівняно з SPS. Проте, окрім цих двух архітектур також існують новіші підходи, що в більшості є гібридними підходами між HPS і SPS. Ці методи були реалізовані пізніше та ще не здобули достатньої популяризації.

Незважаючи на потенційні переваги багатозадачного навчання, більшість проблем машинного навчання все ще вирішується за допомогою

однозадачного підходу (Single Task Learning, STL). Це пояснюється відносною складністю реалізації та налаштування архітектури мережі MTL. Навіть у випадках, коли проблема природно розкладається на кілька взаємопов’язаних завдань, традиційним рішенням залишається використання окремих STL моделей, які навчаються на одному й тому ж наборі даних незалежно одна від одної. Такий підхід, хоча і простіший у реалізації, має суттєвий недолік: окремі моделі не мають механізму обміну інформацією та не можуть використовувати потенційні зв’язки між підзадачами. Методи MTL, в свою чергу, дозволяють уникнути цієї проблеми. Різницю підходів до вирішення двозадачної проблеми можна побачити на рисунку 2.1.

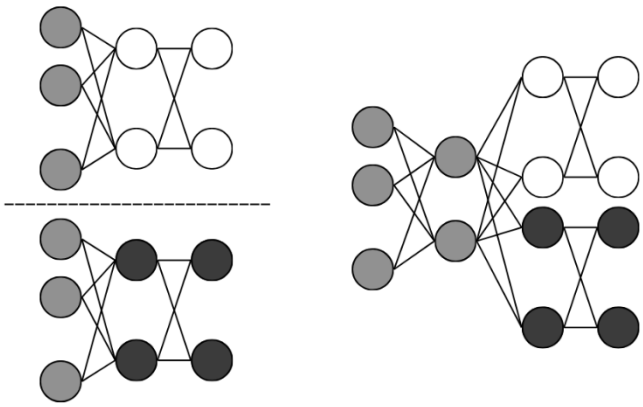

Рисунок 2.1 – Порівняння методу STL з методом HPS

Основним етапом реалізації MTL є правильне визначення набору завдань. Складність цього процесу цілком залежить від набору даних та необхідного результату роботи. Деякі задачі можна розділити на підзадачі без великих проблем, оскільки оригінальне завдання вже вимагає класифікації даних у більш ніж одному міченому вимірі. Проте з деякими задачами розділити одну задачу на декілька настільки ж важливих задач не є можливим. В такому разу визначаються допоміжні задачі, продуктивність яких не враховується, і вони слугують для покращення виконання

основного завдання. Як зазначено в попередніх дослідженях [12], набір задач має бути пов’язаний між собою. Більшість реалізацій потребують один спільний набір даних, на якому ці підзадачі будуть оперувати, проте новіші підходи можуть допускати різні вхідні дані для задач.

При реалізації HPS найпоширенішею архітектурою є така, що складається з декількох спільних шарів, що відповідають за виявлення загальних ознак та окремих шарів виводу, що відповідають за інтерпретацію цих ознак у контексті кожного завдання. Таким чином, при створенні мережі типу HPS, часто використовується попередньо створені мережі STL. Так, така мережа адаптується чи об’єднується з іншими мережами для формування єдиної мережі. Проте, при перетворені набору добре працюючих STL-моделей в єдину MTL-модель, архітектура цих підмереж має бути адаптована під конкретне завдання, в тому числі створення абсолютно нових шарів для конкретних завдань. Крім того, така спільна архітектура зазвичай потребує більшу кількість регульованих гіперпараметрів. Тож створення такої системи мереж не є тривіальним завданням, а результуюча мережа доволі часто може страждати від збільшеного часу навчання та необхідності більшої кількості обчислювальних ресурсів.

Підхід SPS, на відміну від HPS, підтримує реалізацію через використання окремих мереж для кожного завдання. Проте, для приведення усіх мереж та їх результатів до певної схожості, між моделями вводяться певні обмеження за допомогою функції регуляризації. Поширеними прикладами регуляризаторів є норма відстані L2 [12], а також тензорна норма сліду (Tensor Trace Norm) [13]. Функція регуляризації працює за рахунок додавання певного штрафу до кожної функції втрат, що дозволяє контролювати відмінності між вагами різних мереж. Такий підхід зменшує відмінності між вагами підмереж, навчаючи їх так, щоб вони використовували подібні ознаки, проте дозволяє загальній мережі бути

значно гнучкішою за підхід HPS, котрий страждає при збільшенні неоднорідності підзавданнь.

2.2.2 Переваги MTL

У своїх дослідженнях Р. Керуано [12] навів низку гіпотез, які мають на меті пояснити переваги MTL порівняно з попередніми підходами. В подальшому, ці гіпотези були обгрунтовані та доведені в новіших дослідженнях, зокрема [11]. Серед них були визначені такі властивості MTL, як:

– боротьба з упередженістю: так як мережа ставить на меті згенерувати підходящі ваги для усіх задач одночасно, вихідна мережа буде здатна знаходити корисні ознаки, які застосовуються до кількох задач, а не підлаштовується під специфічні вимоги лише однієї;

– боротьба з зашумленими даними: так усім задачам притаманний певний рівень шуму, котрий проте буде різним для кожної з них, мережа здатна легше усереднювати шуми, що знижує ризик перенавчання;

– підслуховування: процес підслуховування полягає в здатності усіх підмереж системи ефективно визначати ознаки, навіть якщо певні з мереж не здатні на це самостійно, за рахунок підслуховування до сусідніх мереж. Так, позичаючи знання, мережа може легше змоделювати необхідні зв’язки, непритаманні її підзадачі;

– запобігання перенавчанню: навчання певній ознаці декількома підмережами запобігає перенавчанню щодо цієї ознаки, так як при перенавчанні однієї підмережі, загальна мережа враховує результати другої підмережі та коригує параметри навчання обох підмереж.

2.2.3 Попередні дослідження в сфері MTL

Вперше концепція MTL була запроваджена та популяризована Р. Каруано в його дисертації [12]. Через відповідний рівень технологій та знань, найпоширенішими були неглибокі архітектури з лише одним чи двома прихованими шарами. Проте останнім часом з'являється все більше досліджень щодо використання сучасних глибоких мереж, що використовують велику кількість шарів різних типів. Одним з прикладів таких іновацій є дослідження [14], що довело доцільність використання MTL при виконанні задачі розпізнавання людської активності, зокрема виявлення орієнтирів обличчя на основі CNN. Було виявлено, що використання додаткових підзадач (наприклад, оцінка пози голови та виведення атрибутів обличчя) не тільки перевершує існуючі методи, але й зменшує складність моделі з точки зору кількості необхідних обчислень.

Більш того, деякі дослідження пропонують нові підходи до MTL. Так дослідження [15] представляє новий метод між HPS і SPS, що полягає в реалізації структури SPS, проте із видаленням регуляризатора для шарів на однаковій глибині. Замість цього використовувалися спеціальні блоки зшивання, які об'єднують вихідні дані попередніх шарів попіксельно, виконуючи восьмирозрядне матричне додавання.

Наприклад, якщо між двома підмережами розміщено блок пересічного зшивання, який об'єднує виходи двох згорткових нейронів, то він параметризується як $\Phi \in R^{2\times 2}$. Він обчислює виходи для кожної позиції однакових за формою вихідних тензорів згорткових нейронів. Для простоти можна припустити, що ці тензори можна індексувати за допомогою одновимірного індексу, тобто вони згорнуті до одновимірного вектора. Отже, тензори, що повертаються операторами згортки, можна розглядати як вектори $\vec{x}^{(A)}, \vec{x}^{(B)} \in R^l$.

Для будь-якої позиції i у цих розгорнутих тензорах функція, обчислена блоком пересічного зшивання, буде:

$$CS_{\Phi}(x_i) = \Phi \cdot x_i = \begin{bmatrix} \alpha_{A,A} & \alpha_{A,B} \\ \alpha_{B,A} & \alpha_{B,B} \end{bmatrix} \cdot \begin{bmatrix} x_i^{(A)} \\ x_i^{(B)} \end{bmatrix} =$$

$$= \begin{bmatrix} \alpha_{A,A} \cdot x_i^{(A)} + \alpha_{A,B} \cdot x_i^{(B)} \\ \alpha_{B,A} \cdot x_i^{(A)} + \alpha_{B,B} \cdot x_i^{(B)} \end{bmatrix}. \quad (2.4)$$

Так, матриця параметрів α визначає, яка частина інформації передається між завданнями, а яка залишається локальною. Наприклад, при $\alpha_{12}=\alpha_{21}=0$ завдання працюють незалежно, а при великих значеннях інформація активно передається між завданнями. Значення $\alpha_{A,A}$ буде залежати віж кількості інформації з блоку CNN для першої задачі, яку слід зберегти для задачі A, тоді як $\alpha_{A,B}$ дорівнюватиме кількості інформації з задачі B, яку слід передати для задачі A. Цей розрахунок виконується для кожної позиції в тензорному виході попередніх шарів, які, таким чином, повинні мати однакову форму.

3 ПРАКТИЧНІ ДОСЛІДЖЕННЯ

В ході цього дослідження було проведено експерименти з двома наборами даних. Так як метою було визначення доцільності використання обраних методів для визначення та класифікації активності людини, було обрано один датасет із області розпізнавання людської активності, зокрема руху, і другий – із області розпізнавання емоцій.

3.1 Основні концепти та архітектури

3.1.1 Однозадачне навчання

Незважаючи на те, що основна мета цієї роботи – дослідження багатозадачного навчання та визначення, чи є доцільним його використання, створення та тестування мережі STL все ще є необхідним елементом роботи. Причиною цього є те, що для якісної оцінки та порівняння методів MTL і STL необхідно мати еталонний примірник. Нажаль, порівняння лише з літературними прикладами із попередніх досліджень не є оптимальним, через різницю між підходами та методами, що були використані. Саме тому, має бути створена та навчена мережа типу STL, що буде оцінена за такою ж самою метологією, як і мережі MTL.

Більш того, велике значення при оцінці методів багатозадачного навчання має безпосередньо складність та вартість перетворення мережі із підходу STL до MTL. Оскільки методи MTL за своєю природою вимагають більших обчислювальних ресурсів, вибір оптимальної архітектури потребує ретельного балансу між продуктивністю та ефективністю. Так, обиралися архітектури, що надавали прийнятні результати, не будучи при цьому занадто складними, навіть якщо складніші варіанти дають кращі результати.

Важливими критеріями оцінки стали не лише результуючі показники ефективності, але й часові характеристики процесу навчання та динаміка

покращення результатів протягом тренування. Варто зазначити, що вибір мережі напряму залежить від конкретного набору даних, тому детальний опис обраних архітектурних рішень буде представлено разом із датасетами.

3.1.2 Жорсткий обмін параметрами

Як вже зазначалося раніше, одним із методів створення мережі багатозадачного навчання є об’єднання декількох базових мереж однозадачного навчання. В випадку, якщо завдання є подібними одне до одного, це завдання можна вважати досить тривіальним – архітектура HPS передбачає спільне використання нижнії шарів мережі, тож основна проблема полягає у виборі, які саме шари будуть поширюватися. І лише після цього створюються окремі верхні шари для кожної із задач. На рисунку 3.1 можна побачити приклади такої архітектури, де підмережі можуть мати різну структуру відповідно до типу підзадачі. Білим позначено спільні шари, а сірим – індивідуальні.

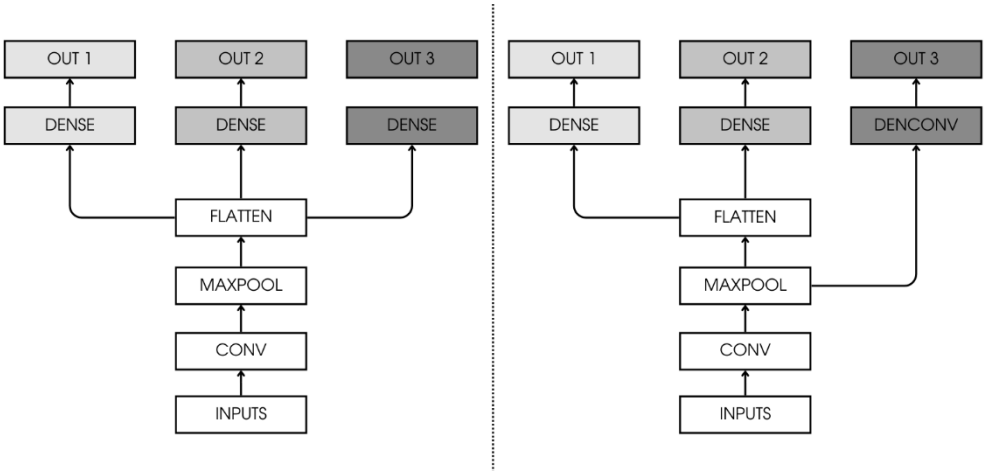

Рисунок 3.1 – Простий приклад мережі жорсткого обміну параметрами

Проте, незалежно від задачі, ці окремі вихідні шари все ще мають на меті покращення загальної, а не індивідуальної функції втрат. Зазвичай

загальна функція втрат визначається як зважена сума функцій втрат для кожної задачі. Так, якщо N – кількість задач, L_i – функція втрат, а α_i – ваговий коефіцієнт, то загальну функцію втрат можна представити у вигляді формули:

$$L_{total} = \sum_{i=1}^{N} \alpha_i L_i. \quad (3.1)$$

Таким чином, при визначенні загальної функції втрат як лінійної комбінації індивідуальних функцій втрат та вагових коефіцієнтів, з'являється можливість враховувати та гнучно коригувати внесок кожної задачі. Такий підхід забезпечує ефективну оптимізацію параметрів, де ваги можуть налаштовуватися як вручну, так і автоматично, за допомогою адаптивного процесу навчання.

Функції втрат між задачами можуть відрізнятися. Так, для умовної задачі класифікації з трьома підзадачами, перші дві підзадачі можуть бути задачами класифікації, функціями втрат яких будуть функції крос-ентропії. Третя задача в свою чергу може виконувати роль автоенкодера, основною метою якого буде зменшення розмірності вхідних даних для виявлення як основних і найважливіших ознак, так і для виявлення і видалення аномалій та шуму. Це можна побачити на рисунку 2.1 на прикладі правої архітектури. Проте, це дослідження було зосереджене на задачі класифікації, в результаті чого усі індивідуальні функції втрат були визначені як функції категоріальної крос-ентропії. Із попередніх робіт, зокрема [3], можна побачити, що ця функція є домінуючою у задачах багатокласової класифікації, де дані представлені у вигляді закодованих за допомогою гарячого кодування векторів. Таким чином, результатом роботи мережі є вектор оцінок ймовірностей приналежності зразка до кожного із класів, а клас із найбільшою ймовірністю вважатиметься безпосередньо передбаченням мережі. З цього ж дослідження можна взяти і формулу для обчислення функції втрат крос-ентропії [3]:

$$CE(Y, \hat{Y}) = -\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i. \quad (3.2)$$

3.1.3 М’який обмін параметрами

Основна методологія реалізації методу м’якого обміну параметрами була описана у дослідженні [15], і я ця робота використовує хоч і адаптовану, проте схожу архітектуру.

Принципова особливість даного методу полягає у впровадженні окремих обчислювальних шарів для кожного завдання, що відрізняє його від попереднього підходу. Після формування цих індивідуальних шарів параметри кожного набору шарів піддаються нормі, яку мінімізують, з метою досягнення певної подібності між ними. Таким чином, хоч параметри підмереж не стають ідентичними, процес регуляризації та штрафування змушує підмережі до певної стандартизації.

В якості регуляризатора використовується тензорна слідова норма (ТСН). Фактично, ця норма – розширення поняття слідової норми для матриць на випадок багатовимірних тензорних структур. Подібно до того, як слідова норма матриці обчислюється через суму її власних чисел, ТСН визначається як сукупність слідових норм усіх можливих режимів тензора. Практична реалізація ТСН вимагає попередньої обробки даних, а саме – розгортання тензора у двовимірну матричну форму. Для виконання такої трансформації існують різні методологічні підходи, серед яких можна виділити: метод Такера [16], метод на основі тензорно-потокове розкладання, метод LFA [12]. В цій роботі буде використано підхід Такера, що математично можна представити у вигляді формули:

$$\|W\| = \sum_{i=1}^{m} \alpha_i \left\|W_{(i)}\right\|, \quad (3.3)$$

де m – розмірності тензору, а $\|W\|$ – безпосередньо слідова норма матриці після розгортання. Також, цей підхід використовує перемінну α_i , що

створена для контролю над балансом між функцією втрат та тензорною слідовою нормою. Значення α_i можуть змінюватися для кожного із значень моди тензору *i*, проте в цій роботі її значення було сталим у 0.01.

В подальшому, функцію втрат мережі із кількістю задач T, можна вирахувати за сумою норми $\|W\|$ та існуючої функції втрат, за формулою:

$$L_{total} = a_{TTN}\|W\| + \sum_{t\in T} L_{CE}. \quad (3.4)$$

3.1.4 Мережі пересічного зшивання

Метод пересічного зшивання (Cross-Stitch), вперше представлений у дослідженні [15], можна розглядати як синтез попередніх підходів до багатозадачного навчання. Ключова ідея цього методу полягає у створенні балансу між незалежністю окремих підмереж та їх здатністю обмінюватися корисною інформацією.

Так, кожне завдання використовує свої власні обчислювальні шари, проте підхід пересічного зшивання дозволяє декільком підмережам ефективно ділитися своїми ознаками. Спеціальні блоки зшивання виконують роль «розумних регуляторів» інформаційного обміну між підмережами та використовують механізм зваженого множення для визначення оптимальної кількості інформації, якою мають обмінюватися підмережі перед передачею даних до наступного шару. Математичне обґрунтування цього процесу детально описано у попередньому розділі, зокрема у формулі 2.4. Графічно, цей процес можна представити у вигляді рисунка 3.3, запозиченого з оригінальної роботи [15], який наглядно демонструє роботу одного блоку.

Такий підхід дозволяє мережі автоматично визначати оптимальний рівень взаємодії між різними завданнями, забезпечуючи більш гнучкий та ефективний механізм спільного навчання порівняно з попередніми методами.

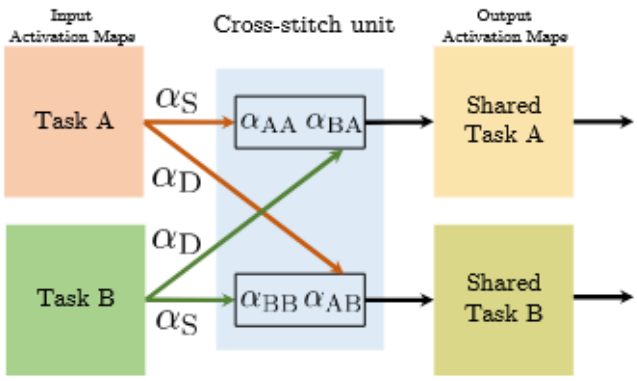

Рисунок 3.3 – Вигляд окремого елементу пересічного зшивання

3.2 Використані датасети

3.2.2 Датасет OPPORTUNITY

OPPORTUNITY [17] – датасет, вперше представлений у 2010 році. Він є набором даних, призначеним для задач розпізнавання людської активності (HAR), що містить дані зібрані за допомогою носимих сенсорів та датчиків у навколишньому середовищі.

Незважаючи на те, що це не єдиний загальнодоступний набір даних, більшість існуючих наборів даних недостатньо багаті для дослідження розпізнавання опортуністичної активності. Найбільш відомі з них: набір даних PlaceLab, що зосереджується на зондуванні навколишнього середовища та об’єктів; набір даних Van Kasteren з особливо довгими записами (помісячними), але з меншою кількістю датчиків, а також Дармштадтський рутинний набір даних, що використовується для виявлення патернів неконтрольованої активності, тобто довгий запис активності тіла, зібраний системою Porcupine [17].

Тож, унікальність цього набору даних відображена в самій його назві, яка підкреслює його головне призначення – розвиток опортуністичного розпізнавання активності людини. Опортуністичний підхід представляє

собою гнучку систему, яка може ефективно функціонувати навіть за відсутності ідеальних умов чи жорстко заданої конфігурації, адаптуючись до наявних джерел інформації та використовуючи їх оптимальним чином.

Структурно набір даних організовано як серію маркованих експериментальних прогонів, під час яких учасники виконують різноманітні дії у спеціально обладнаному середовищі з великою кількістю датчиків. Детальний опис структури та змісту набору даних можна знайти в оригінальному дослідженні [17], що буде узагальнено нижче.

Експериментальне середовище імітує однокімнатну квартиру, обладнану ліжком, міні-кухнею та столом. Для реалізації принципів опортуністичної класифікації було створено розгалужену мережу датчиків, що дозволяє гнучко використовувати різні їх комбінації залежно від конкретного сценарію. Така архітектура сенсорної мережі забезпечує можливість моделювання різноманітних опортуністичних сценаріїв під час подальшої класифікації. Налаштування лабораторії та розміщення датчиків показано на рисунку 3.4.

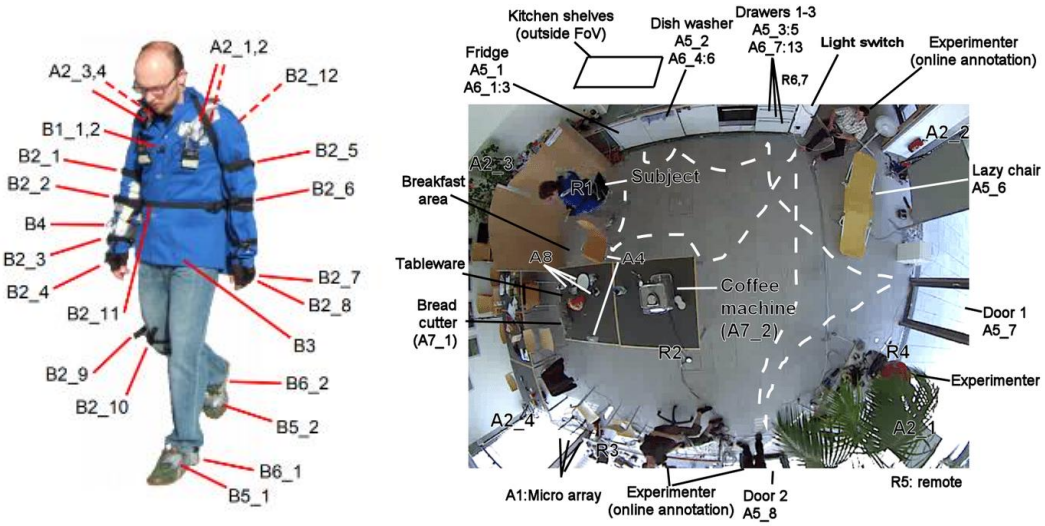

Рисунок 3.4 – Розміщення сенсорів на тілі та у лабораторії

Можна побачити, що датчики поділяються на два великі класи:

– тілесні: датчики прискорення, інерційні вимірювальні прилади (які також показують положення), системи відносного позиціонування (що показують відстань від руки до тіла), а також мікрофони, що встановлені на тілі суб’єкта. Загалом на тілі було зафіксовано 133 сенсорні канали;

– навколишні: саме середовище було оснащене акселерометрами, герконами (що реагують на відкриття та закриття дверей та шухляд), а також камерами та системами визначення надширокосмугового позиціонування. Загалом доступно 109 таких вимірювань.

Дані з різних каналів датчиків були синхронізовані під час постобробки та субдискретизовані приблизно до 30 Гц. Після цього дані були розмічені вручну до п’яти (точніше сьоми) різних доріжок:

– Locomotion: проста активність, що залежить від того, чи об’єкт стоїть, ходить, сидить чи лежить;

– HL_Activity: активність високого рівня, така як: «Відпочинок», «Час на каву», «Ранкова рутина», «Прибирання», «Час на сендвічі»;

– LL_Left_Arm & LL_Right_Arm: що робить відповідна рука, напр. відкривати, пити або рухатися;

– LL_Left_Arm_Object & LL_Right_Arm_Object: з яким об’єктом відповідна рука взаємодіє, напр. двері 1, чашка або холодильник;

– ML_Both_Arms: комбінація двох попередніх каналів, напр. відкриття двері 1 або пиття з чашки.

Автори OPPORTUNITY рекомендують роботу та класифікацію щодо каналу ML_Both_Arms, і попередні дослідження справджують цю тезу, повідомляючи про найкращі результати.

Так як кількість даних під час кожного проходу суб’єктів через сценарій дуже велика, кожен канал також може приймати значення NULL, якщо жоден із можливих варіантів не підходить. Проте, пусте значення не завжди має означати деяку втрату пакетів. За оцінками авторів, було втрачено близько 2,5% пакетів, через проблеми зі з’єднанням через

оклюзію, розряджені батареї або переповнений спектр через велику кількість датчиків, більшість з яких працює в діапазонах 2,4 ГГц і 5 ГГц. Решта пустих значень може означати певні характеристики активності суб'єкта. Наприклад, канал ML_Both_Hands містить не усі можливі комбінації активності обох рук, так як суб'єкти є переважно правшами, і так як їх заохочували взаємодіяти з навколишнім середовищем природним шляхом, інформації щодо роботи правої руки значно більше за ліву.

Відкрита версія датасету складається з чотирьох сеансів, кожен з яких має шість записаних прогонів. З цих шести запусків перші п'ять були запусками ADL, де суб'єкт виконував послідовність повсякденних дій за сценарієм. Шостим запуском був запуск Drill, де суб'єкт виконував 20 повторень коротшої послідовності. Один прогон ADL займав від 15 до 20 хвилин, а прогон Drill зайняв приблизно від 20 до 35 хв. Тож загальна довжина записів сягає приблизно 155 хвилин на частоті 30 Гц.

Одним з основних етапів роботи з нейронними мережами є безпосередньо вибір даних для навчання мережі та подальшої класифікації. Проте на виборі, робота з датасетом не закінчується – для ефективного використання машинного навчання усі дані попередньо мають бути певним чином оброблені. Така обробка може бути проведена в декілька способів та за різною методологією. Зокрема, найпопулярнішими методами обробки даних є очищення даних від пропущених даних, шумів чи аномалій, перетворення вхідних даних, як то маштабування, кодування чи перетворення (логарифмічне чи степеневе), аугументація чи навпаки – зниження розмірностей та видалення нерелевантних ознак, тощо.

Зокрема, при роботі з часовими рядами важливим етапом обробки є сегментація даних. Основною метою сегментації часових рядів є спрощення подальшого аналізу, як за допомогою спрощення виявлення певних патернів, так і за допомогою зменшення часу навчання. Так як часовий ряд можна визначити як потік одновимірних даних, сегментація цього потоку результує у наборі коротких фрагментів, відносний розмір яких і є основною

перевагою. Основною причиною є зменшення параметрів, які мережа має навчити, так як чим довшим є кожен зразок, тим більше параметрів потрібно для шарів, які обробляють ці великі тензори.

Крім того, така обробка також може слугувати як ефективний спосіб збільшення даних. Для двох обраних датасетів основна причина сегментації різна – датасет DEAP має недостатню кількість зразків для навчання, в той час як зразки датасету OPPORTUNITY значно довші, по декілька хвилин кожен, що призводить до проблем з обмеженнями у пам’яті та часі.

В результаті детальні кроки обробки даних значно відрізняються між двома датасетами. Основна складність задачі сегментації полягає в правильному виборі часових вікон, різних між датасетами, доцільності накладання фреймів між сегментами, та виборі стратегії маркування та об’єднання міток.

Структура набору даних OPPORTUNITY представляє собою комплексну систему записів, розподілених між 24 текстовими файлами. Кожен файл містить матрицю даних, де рядки відповідають окремим часовим точкам вимірювань, а 250 стовпців містять різноманітні числові показники. Крім того, надані ще два файли із інформацією про ключі каналів та систему міток.

Перший стовпець завжди містить часову мітку, що дозволяє точно визначити момент збору даних. Наступні 133 стовпці представляють фізіологічні показники, отримані з датчиків, розміщених безпосередньо на тілі учасників експерименту. Ще 109 стовпців містять дані про навколишнє середовище, зібрані зовнішніми сенсорами. Завершують кожен запис сім стовпців із закодованими мітками активності. У останніх семи стовпцях закодовані мітки активності.

Процес обробки був розділений на два етапи, за методологією попередньо застосованою у [18].

На першому етапі проводилася ретельна очистка даних, основною метою якої є видалення відсутніх (позначені як NaN) даних. Таким чином

було вилучено 38 каналів, які найбільше постраждали від проблем із записами, в тому числі записи із активністю правої чи лівої рук. Для решти 107 каналів, відсутні дані були замінені попередніми значеннями. Розподіл значень NULL показано на рисунку 3.5.

Другий етап включав сегментацію даних із використанням методу ковзаючого часового вікна. Цей метод полягав у розбитті масиву даних на двосекундні відрізки, кожен з яких містив по 64 зразки. Особлива увага приділялася визначенню міток для кожного сегмента – вони призначалися за принципом більшості, тобто обиралася та мітка, яка найчастіше зустрічалася у межах конкретного часового вікна.

labels	count
NAN	35390
Close Dishwasher	307
Close Drawer 3	110
Close Drawer 2	172
Close Door 1	109
Close Door 2	318
Close Drawer 1	154
Close Fridge	624
Toggle Switch	27
Open Dishwasher	348
Open Drawer 3	96
Open Drawer 2	158
Open Door 1	133
Open Door 2	363
Open Drawer 1	204
Open Fridge	413
Drink from Cup	1649
Clean Table	1013

Рисунок 3.5 – Кількість міток датасету OPPORTUNITY у каналі ML_Both_Arms

Основна відмінність обробки данних в ході цієї роботи, порівняно із [18] – додатковий пошук та обробка аномалій, присутніх у деяких каналах датасета у невеликій кількості, що зображені на рисунку 3.6. З графіків видно, що ці сплески не відповідають діям суб’єкту чи певним подіям, через відсутність подібних сплесків на інших каналах відповідних сенсорів. Тож,

через збільшення динамічного діапазону даних та відсутність кореляції між різними каналами, було вирішено їх видалити для запобігання проблем з навчанням. Настільки великі аномалії були присутні тільки на записах із декількох сессій, зокрема S3-ADL1 та S4-ADL2, тож їх вплив на ефективність мережі був доволі низьким і відчутної різниці при навчанні на обробленому датасеті та необробленому знайдено не було.

Ці сплески було усунено за рахунок заміни їх значень на значення середнього μ із певним стандартним відхіленням σ у вигляді $\mu \pm \tau\sigma$, де $\tau \in R$. Було вибрано значення $\tau = 10$. Такий підхід не усував сплески повністю, але обмежував їх до більш прийнятного діапазону.

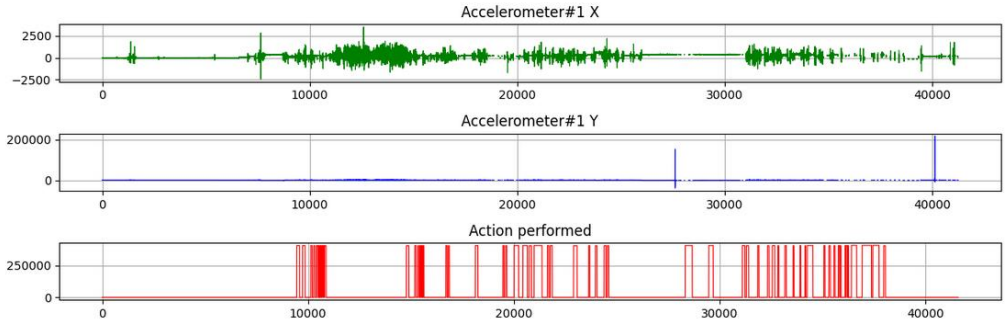

Рисунок 3.6 – Аномалії датасету OPPORTUNITY

3.2.2 Датасет DEAP

Основними датасетами для аналізу емоцій за допомогою фізіологічних сигналів є датасет DEAP (Database for Emotion Analysis using Physiological Signals) та SEED (SJTU Emotion EEG Dataset). Обидва датасети фокусуються на створенні даних на основі сенсорів, зокрема електроенцефалографії (ЕЕГ). Проте основна відмінність між ними – фокус SEED на ЕЕГ, як єдиному джерелі інформації. В цей час, датасет DEAP додатково використовує сигнали з електроокулографії (ЕОГ) та електроміографії (ЕМГ), що слугують для фіксації руху лицьових м’язів і очних яблук, значення з датчиків щодо електропровідності шкіри суб’єкта,

його дихання та температури шкіри. Всього доступно 40 сенсорних каналів, з яких 32 походять від ЕЕГ. Таким чином, датасет DEAP набагато краще підходить для цього дослідження, а саме багатозадачного навчання [19].

Учасники дослідження, а саме 16 чоловіків та 16 жінок у віці 19–37 років, оцінювали надані їм кліпи та емоції, що вони викликали. Основною проблемою при створені датасету було визначення системи категоризації емоцій, за якими буде проводитися оцінка.

Дослідження щодо класифікації емоцій рідко сходяться на одній системі класифікації, і дискусії на цю тему є досить активними навіть в цей час. Існує три основні підходи до визначення та класифікації емоцій.

Перший – підхід, що визначає шість основних емоцій, де усі інші походять від комбінацій основних [20]. Основними емоціями при такій класифікації є радість, сум, гнів, огида, страх та здивування. Ті дослідження, що погоджуються з системою основних і комбінованих емоцій, також погоджуються із цим визначенням шости основних емоцій, проте існують такі, що визначають інші набори [21]. Схожим є ще класифікація емоцій у вигляді ієрархічної структури, проте такий підхід не набув популярності.

Другий підхід полягає у визначенні емоцій за допомогою вимірних шкал, як то коло емоцій Плачтика [20] чи шкала валентності-збудження за Расселом [22]. Датасет DEAP зупинив свій вибір саме на класифікації Рассела. Так, згідно з його теорією, кожну емоцію можна розташувати на двовимірній площині з горизонтальною віссю збудження та вертикальною – валентності. Його дослідження також зазначає, що хоч ці дві шкали можуть визначити майже будь-яку емоцію, інколи можна включити третю шкалу – шкалу домінантності. Збудження може варіюватися від неактивного (байдужість, нудьга) до активного (пильність, збудження), в той час як валентність варіюється від неприємного (сум, стрес) до приємного (щастя, ейфорія). Домінування в свою чергу, варіюється від безпорадного та слабкого відчуття (без контролю) до відчуття повної сили (контроль над усім).

Крім того, існує третій підхід, відомий як модель компонентних процесів [20]. Він є менш популярним за перші два, проте його все ще варто брати до уваги при роботі емоцій. Загалом, усі три підходи використовуються дослідниками і жоден із них не є однозначно домінуючим над іншими.

Тож, в результаті, було вирішено використовувати модель основного афекту для вимірювання емоцій. Суб'єктів просили призначити оцінку валентності, збудження та домінантності в діапазоні від 1 до 9. Оцінка була зроблена з використанням манекенів SAM, запропонованих у [23] та проілюстрованих на рисунку 4.2. Проте суб'єкти не були обмежені цілими значеннями та могли обрати будь-яку точку на діапазоні. Крім того, також були запитані відомості, чи знайомі суб'єкти з відео і наскільки (ціле число від 1 до 5) і наскільки їм сподобався кліп (від 1 до 9). Систему манекенів можна побачити на рисунку 3.7.

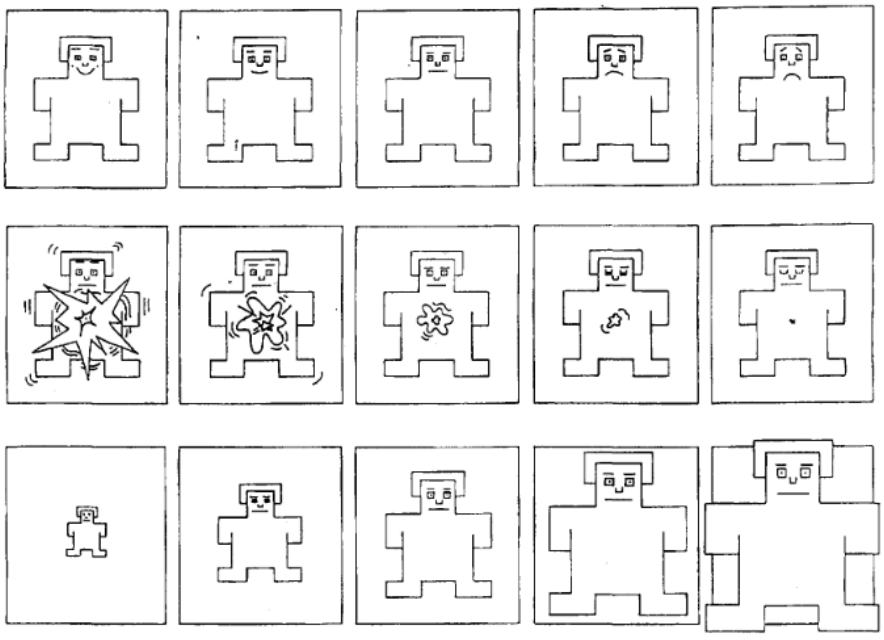

Рисунок 3.7 – Манекени для самооцінки, зверху вниз: валентність, збудження, домінування

Основна причина для використання складної багатозадачної мережі класифікації при роботі із цим датасетом, замість простішої моделі регресії – неоднорідність рейтингів, наданих суб'єктами. Вони можуть бути незнайомі з моделлю оцінки, мати попередні упередження щодо певних відео, тощо. Тому оцінки схожих емоцій неминуче відрізнятимуться між суб'єктами, а також можуть бути різними для 40 різних відео.

Тож, якщо всі емоції, класифіковані Расселом [22], приблизно розподіляються на чотири окремі квадранти високої та низької валентності та збудження, проблеми, як правило, повторюються як дві проблеми класифікації *n*-класу.

Попередні дослідження для класифікації емоцій за допомогою двовимірної моделі основного афекту зазвичай використовують однозадачне навчання, із двома незалежнимим мережами для кожного із завдань. Проте, оскільки збудження та валентний вимір емоцій мають корелювати, можна припустити, що реалізація методів MTL може принести свої переваги.

На відміну від OPPORTUNITY, датасет DEAP в свою чергу надається вже в обробленому форматі, проте це дослідження потребує ще декілька не менш важливих кроків. Основною відмінністю між двома датасетами є те, що у DEAP до кожного експерименту прикріплена лише одна мітка, на відміну від OPPORTUNITY, де мітки змінюються з часом. Саме тому додатково до сегментації обробка має включати в себе процес присвоєння мітки до кожного із зразків. Це було зроблено шляхом присвоєння кожному кадру тих міток, які присвоєні повному запису. Проте так як самооцінка емоцій у тривимірнорному просторі є надзвичайно суб'єктивною, мітки були перетворені на задачу класифікації за методологією, наведеною в попередніх дослідженнях [19]. А саме, кожен канал міток було перетворено на задачу бінарної класифікації, де розділення на класи відбулося за медіаною діапазону оцінювання.

Процес сегментації в свою чергу потребував більшої роботи через таку ж саму проблему – суб’єктивність оцінювання. Незважаючи на те, наскільки об’єктивними учасники намагались бути, їх симпатії чи, навпаки, антипатії до виконавців, музичних жанрів чи кліпів будуть пливати на фінальну оцінку.

Більш того, як було приведено в літературі по темі [17], неінвазивна ЕЕГ є неідеальним методом для вимірювання активності, пов’язаної з індукованими емоціями, оскільки «більшість нейронної активності під час емоцій відбувається значно глибше в мозку [ніж на шкірі голови]». Такі вимірювання можуть фіксувати лише невеликі активації, які можуть бути доволі важкими для інтерпретації. В результаті цього, безпосередньо вибір даних, що буде використовуватися, і, що важливіше, як він буде розподілений на навчальний та тестувальні набори, є надзвичайно важливим і впливовим на результати навчання мережі.

Загалом визначено два основні шляхи розподілення:

– підхід, залежний від суб’єкту: навчальні дані забрані із усіх суб’єктів, без виключень, а тестувальні дані складаються з випадково вилучених елементів. Такий підхід дозволяє вивчати патерни для кожного із суб’єктів, а продуктивність мережі не залежить від зміни розподілу даних, проте її ефективність падає при класифікації невідомих до цього учасників;

– підхід, незалежний від суб’єкту: при такому підході тестувальні дані складається із повністю вилучених даних певних суб’єктів, які не з’являлися в навчальному наборі. Таким чином, мережа вчиться узагальненню та пошуку більш універсальних ознак, що допоможе при класифікації нових елементів, проте її ефективність сильно залежить як саме був розподілений датасет.

Можна також визначити третій підхід – підхід для одного учасника, але він скоріше є окремим прикладом суб’єкто-залежного підходу, де використовуються лише дані одного суб’єкту. Такий підхід може бути

корисним в деяких випадках, проте він потребує надзвичайно великої кількості даних, для запобігання проблемі перенавчання.

В ході цього дослідження було обрано суб’єкто-залежний підхід. Такий підхід є більш точнішим за альтернативний [24], а також більш підходящим безпосередньо для теми цієї роботи.

При сегментації, в якості часового інтервалу було обрано 1 секунду із накладанням у 0.5 секунд. Такі параметри були обрані в результаті проведених експериментів, що полягали у навчанні експериментальних мереж із разними параметрами для визначення ідеальних. В ході експериментів було визначено, що розмір вікна у 3 секунди, що використовувався як оптимальний у попередніх дослідженнях [25], не є таким для цієї задачі, через проблему перенавчання, зумовлену завеликим розміром вікна. В ході експерименту були оцінені результати із розміром вікна в 2 та 1.5 секунд, що все ще не дозволяло досягти бажаної продуктивності. Тож, в кінці інтервал був визначен як 1 секунда.

3.3 Вибір архітектури

3.3.1 Датасет OPPORTUNITY

Наступним етапом після обробки даних був безпосередньо пошук методів MTL для роботи із наданими датасетами. Це полягало в створенні окремих базових мереж для кожного із двох датасетів, що в подільшому можуть бути модифіковані для використання із будь-якими техніками MTL.

Ці базові моделі були обрані на основі попередніх досліджень, проте перевага надавалась простішим методам через обмеження у часі та доступному програмному забезпеченні. Проте, з огляду на певні припущення [14], можна вважати, що техніки MTL застосовані до простих архітектур мають хоча б зрівнятися із результатами STL на основі складніших архітектур.

Як вже було зазначено раніше, основною роботою в сфері виявлення людської активності можна вважати [17], що використовувала датасет OPPORTUNITY для порівняння різних методів класифікації. Ця робота була взята за основу як при виконанні попередніі обробки даних, так і при пошуку необхідної моделі. Основні методи глибокого навчання, що були досліджені в роботі [17], вже були описані в розділі 2.1. Решта методів, такі як автоенкодери, підходи Code-Book та підходи ручного визначення ознак, не були оглянуті в цій роботі, так як вони не підходили для роботи із багатозадачним навчанням.

Основна продуктивність моделі оцінювалася за допомогою трьох метрик: точності, зваженої F_1-оцінки, та середньої F_1-оцінки. Валідаційний набір даних складав приблизну четверту частину усього датасету.

Точність можна визначити формулами як:

$$c(a,b) = \begin{cases} 1 & a = b \\ 0 & a \neq b \end{cases}, \tag{3.6}$$

$$acc(Y,\hat{Y}) = \sum_{i=1}^{N} \frac{c(argmax\ \hat{y}_i, argmax\ y_i)}{N}, \tag{3.7}$$

де Y і $\hat{Y}$ – множини істинних та передбачених міток, що містять N кількість елементів (y і $\hat{y}$), котрі належать до R^l, де l – загальна кількість можливих міток для окремого каналу. Дані були закодовані за допомгою гарячого кодування, де істинне значення елементу $\hat{y}$ дорівнює 1, а решта – 0.

Метрика F_1 в свою чергу є оцінкою, що визначається як середнє між точністю та пригадуванням. Показник F_1 ввикористовується для забезпечення гармонійності мережі, де показники точності та пригадування враховуватимуться однаково. На відміну від показника точності, показник F_1 поєднує в собі показники достовірністі та здатності до відтворення, пропонуючи більш цілісну оцінку продуктивності моделі.

Для бінарної класифікації з одним негативним і одним позитивним класом, метрика повертає матрицю невідповідності, що складається з чотирьох основних компонентів: істинно-позитивний прогноз, хибнопозитивний, істинно-негативний та хибнонегативний. Оцінки точності та пригадування математично визначаються наступним чином:

$$recall = \frac{TP}{TP+FN}, \tag{3.8}$$

$$precision = \frac{TP}{TP+FP}. \tag{3.9}$$

Тоді показник F_1 також можна записати таким чином:

$$F_1 = \frac{TP}{TP+\frac{1}{2}(FN+FN)}. \tag{3.10}$$

Загалом, показник F_1 – окремий випадок показника F_β, де β відображає відношення між пригадуванням та точністю. Так, $\beta > 1$ надає більшої ваги пригадуванню, тоді як $\beta < 1$ надає перевагу точності. Формулу F_β можна записати як:

$$F_\beta = \frac{(1+\beta^2)TP}{(1+\beta^2)TP+FP+FN\cdot\beta^2}. \tag{3.11}$$

Для обчислення показника F_1 для багатокласового набору даних використовується метод «один проти всіх», який дозволяє обчислити індивідуальні показники для кожного класу в наборі даних. Береться середнє гармонійне значення для значень точності та пригадування для кожного класу. Потім обчислюється чистий показник F_1 за допомогою трьох основних методів усереднення:

– макроусереднений показник: F_β обчислюється окремо для кожної мітки, після чого його усереднено за кількістю можливих міток;

– мікроусереднений показник: F_β обчислюється як єдине значення після об’єднання усіх істинно-позитивних та передбачених значень;

– зважений показник: F_β обчислюється окремо для кожної мітки, але кожен результат має бути зважено відповідно до кількості зразків у кожному класі під час усереднення, що враховує дисбаланс класів.

Було розглянуто три основні архітектури мереж: CNN, LSTM та Hybrid-CNN-LSTM. Дослідження [17] показало, що гібридна архітектура надає накращі результати. Проте в ході експериментів було зазначено, що архітектура CNN наближається до гібридної за точністю, а при додавванні додаткових шарів відсіювання (dropout) – навіть перевершує, як можна побачити на рисунку 3.8.

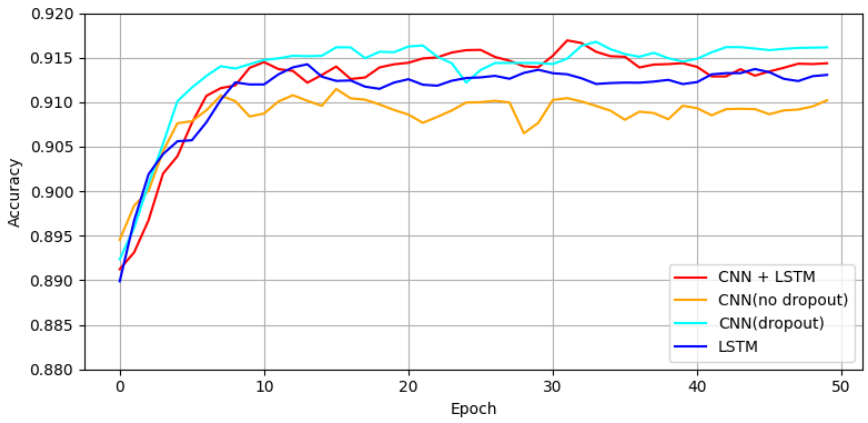

Рисунок 3.8 – Значення точності при тренуванні базових моделей

3.3.2 Датасет DEAP

Архітектура мережі для датасету DEAP схожа на ту, що була вибрана для OPPORTUNITY. Більшість попередніх реалізацій використовують методи екстракції ознак перед навчанням, часто враховуючи лише дані з ЕЕГ, що поділяються на різні частотні діапазони за допомогою методу

перетворення Фур’є, методу на основі вейвлетів (wavelet-based approach), або просто використовують статичні ознаки [26].

В якості архітектури була обрана мережа CNN без рекурентних елементів. Також в ході роботи було помічено, що на ефективність мережі значною мірою впливає яка саме частина даних була надана для навчання, що відбувалося при завантажуванні даних через процес їх перемішування. Так як було вирішено уникати завантаження елементів до пам’яті тільки на етапі навчання, був використан буфер певного розміру, що завантажував елементи до того, як вони були запрошені мережею, а після – надавав на вхід випадковий елемент з буферу, замінюючи його наступним із датасету. Такий підхід дозволив зменшити негативний ефект перемішування даних під час навчання, залишаючи усі позитивні.

На відміну від датасету OPPORTUNITY, що використовував функцію категоріальної крос-ентропії, для навчання була використана функція втрат типу розрідженої категоріальної крос-ентропіх та розрідженої категоріальної точності.

3.4 Технічні деталі реалізації

Для реалізації обраних методів багатозадачного навчання програмними методами була обрана мова програмування Python. Основними перевагами Python над іншими мовами програмування є простота синтаксису та найбагатша колекція бібліотек, як у сфері машинного навчання, так і поза нею. Python також має підтримку паралельних обчислень через бібліотеки на кшталт CUDA, CuDNN та NumPy. Ці технології дозволяють використовувати графічні процесори (GPU) для пришвидшення обчислень, що є однією із основних переваг при тренуванні великих нейронних мереж.

Основними бібліотеками мови Python для створення та тренування нейронних мереж є Keras та Tensorflow. Tensorflow – бібліотека низького

рівня, що надає широкий набір інструментів для роботи з тензорами, автоматичного диференціювання та оптимізації. Ця бібліотека дозволяє отримати точнішу відладку нейронних мереж за рахунок більших вимог до користувача, потребуючи більшу кількість безпосередньо програмування. Keras, в свою чергу, є високоабстрактним API, що дозволяє створювати моделі з мінімальною кількістю коду, використовуючи більш простий і зрозумілий синтаксис. Хоч Keras і може використовувати різні пакети та інструменти, як то CNTK, MXNet і Theano, зазвичай він використовується як інструмент бібліотеки Tensorflow. Так, пакет Tensorflow реалізує API Keras у модулі tensorflow.keras.

Більшість функціоналу бібліотеки Keras за замовчуванням працюють із пакетом numpy. Так, усі дані зберігаються у вигляді багатовимірних масивів, що є зручним рішенням, так як часто обробка даних виконується за допомогою цього ж пакету numpy. Проте, при передачі даних до мережі у функцію model.fit, вони передаються у вигляді повного масиву, розбиваючись на батчі вже потім. Так як при навчанні кожне значення масиву має бути завантажене в оперативну пам’ять, процес навчання за такою технологією є дуже вимогливим до характеристик машини, що зазвичай негативно впливає на продуктивність.

При тренуванні на основі великих датасетів, таких як OPPORTUNITY та DEAP в цій роботі, подібний підхід результує у надзвичайно довгому процесі навчання, та навіть можливих збоях у роботі системи загалом, через недостачу оперативної пам’яті. Саме тому було вирішено використовувати нову версію TensorFlow 2.0, що підтримує формат tf.dataset. Такий формат зберігання даних значно зменшує технічні вимоги до машини, на якій виконується процес тренування мережі.

Основні інструменти для роботи з даними у новій версії TensorFlow – модуль tf.data та формат серіалізації даних Protocol Buffer (protobuf). Зазвичай ці технології використовуються разом, проте вони виконують різні функції.

Модуль tf.data забезпечує високоефективну обробку та завантаження даних для навчання моделей, шляхом надання простих інструментів для створення складних конвеєрів обробки даних. Цей модуль є надзвичайно гнучким інструментом, що полегшує процес обробки даних, оптимізації мережі та дозволяє ефективно працювати із різними форматами даних та у різних чередовищаї одночасно.

Protobuf, в свою чергу, є форматом серіалізації даних, розробленим Google. Він використовується для зберігання структурованих даних у компактному і ефективному форматі, що підходить для передачі між системами чи зберігання. Він дозволяє зберігати дані у найзвичайно компактному вигляді, що має чітку структуру, котра може динамічно змінюватися, без втрати зворотної сумісності.

При використанні разом, tf.data і Protobuf об’єднується до формату TFRecord. Він дозволяє ефективно зберігати дані, серіалізовані у форматі Protobuf, і зчитувати їх за допомогою tf.data. Це особливо корисно для великих наборів даних, де потрібна висока швидкість обробки.

Серіалізація формату Protobuf базується на примітиввних структурах, таких як списки цілих чисел, чисел із плаваючою комою та байтів. І хоч серіалізація використаних датасетів у вигляді чисел із плаваючою комою є можливою, документація TensorFlow рекомендує серіалізувати ці масиви у вигляді байтового рядка. Таким чином, для обох використаних датасетів, вхідні дані, які представлені у вигляді двовимірних тензорів для кожного зразка, були серіалізовані у екземпляри BytesList.

Важливо, що для використання такого датасету для навчання мережі, його потрібно десеріалізувати. Під час навчання, має бути задана певна функція десеріалізації, котра повертає кортеж із вхідних та вихідних даних, що також можуть бути у формі кортежей. При серіалізації даних форма тензору не зберігається, тож функція десеріалізації має мати параметр, що встановлює бажану форму. Така реалізація дозволяє працювати лише з тими

даними, які кожна підмережа потребує безпосередньо в той момент, не витрачаючи пам’ять на зберігання масиву усіх елементів.

Процес перемішування та розбиття на батчі зазвичай відбувається саме після етапу десеріалізації. Процес перемішування вже був описан раніше в цій роботі: при використанні формату TFRecord TensorFlow використовує лінивий підхід – тобто дата зчитується з диску тільки при потребі. Зазвичай такий метод реалізовано за допомогою асинхронного буферизування, тобто поточний батч, разом із кількома наступними батчами, завжди будуть у оперативній пам’яті. Сам процес буферизації полягає у створенні буферу певного розміру, котрий повністю заповнюється прикладами. При запиті нового зразка, TensorFlow випадковим чином замінює існуючий зразок у буфері на наступний елемент датасету. Таким чином, розмір цього буферу є важливим параметром мережі, що впливає на її продуктивність, і зазвичай задається користувачем.

Так, при реалізації мережі для роботи із датасетом DEAP було з’ясовано, що розмір цього буферу є одним із найважливіших параметрів. При початковому розмірі у 1000 елементів, точність класифікатора була дуже низькою. При використанні фреймів по 1 секунді із накладанням у півсекунди, кожен експеримент розбивався на 125 секцій по 5000 кадрів для кожного учасника. Така кількість даних очевидно не була задовільною і низький розмір буферу тільки погіршував ситуацію. Хоч збільшення буферу покращувало результати, вони все ще були незадовільними. Для подальшого покращення мережі датасет був попередньо перемішений під час попередньої обробки, деталі якої були описані в попередньому розділі. На відміну від обробки данних під час навчання, цей процес був реалізований за допомогою методів numpy.

Реалізація самих мереж була реалізована в два етапи. Для методів HPS і SPS окремо були реалізовані базові мережі CNN, та окремо – індивідуальні мережі для кожного із завдань, що об’єднувалися при навчанні. При створені CSN, в свою чергу, мережа створювалася повністю, через

неможливість розподілення її на декілька етапів. Основаною проблемою при створенні такої мережі, були обмеження TensorFlow при автоматичному обчисленні графу мережі. Для вирішення проблеми *graph disconnect*, вихідні дані паралельних підмереж об'єднувалися перед передачею із блоку CrossStitch, а потім негайно розділялися на початковий набір початкових тензорів.

3.5 Візуалізація створених архітектур

Після створення, нейронні мережі було візуалізовано за допомогою вебзастосунку Netron. Цей застосунок дозволяє детально зобразити нейронні мережі у вигляді графів із повним описом шарів, їх назв, розмірностей, тощо. Так, на рисунку 3.9 можна побачити нейронну мережу однозадачного навчання для обох датасетів.

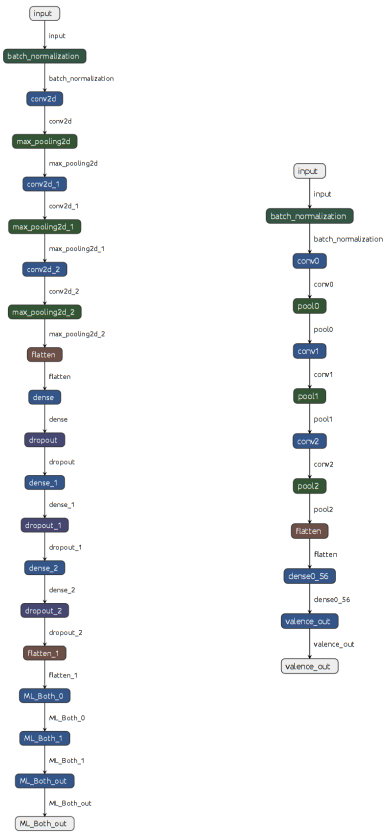

Рисунок 3.9 – Архітектура мережі STL для OPPORTUNITY та DEAP

Як вже було зазначено, базовою мережею було обрано згорткову нейронну мережу із розширенною структурою. На рисунку можна побачити дві різні архітектури, що несуттєво відрізняються між собою. Основними елементами обох є блок нормалізації перед першим згортковим шаром, що значно покращує процес навчання мережі та допомагає боротися з проблемою зникаючих градієнтів. Після кожного згорткового шару використовується операція максимального об’єднання, яка зменшує просторові розміри даних, зберігаючи при цьому найбільш важливу інформацію. Така послідовність операцій повторюється кілька разів, що дозволяє мережі формувати багаторівневе представлення вхідних даних.

Основна різниця між двома архітектурами полягає в шарах після цієї початкової структури. Мережа OPPORTUNITY потребує додаткові повнозв’язні шари (Dense), порівняно із одним для аналогічної мережі DEAP, а також додаткові шари Flatten, які перетворюють багатовимірні дані у одновимірний вектор. Також мережа OPPORTUNITY містить шари Dropout, які повністю відсутні в мережі DEAP. Вони зазвичай використовуються для боротьби з проблемою перенавчання, яка була відсутня при навчанні на датасеті DEAP. Тож, така різниця в архітектурах в першу чергу зумовлена різною кількістю даних та їх характером.

Архітектура мереж HPS для обох датасетів схожа на відповідні архітектури для STL. Підхід HPS реалізовано на основі базових згорткових нейроних мереж, де замість останніх спільних шарів Dense, що йдуть після останнього шару Flatten, створено підмережі із індивідуальними повнозв’язними шарами, та окремими виходами для підзадач.

Така архітектура з подвійним виходом дозволяє мережам одночасно прогнозувати два різних параметри, використовуючи спільні низькорівневі ознаки, виділені згортковими шарами, але з різною інтерпретацією на вищих рівнях обробки.

Ці архітектури можна побачити на рисунку 3.10.

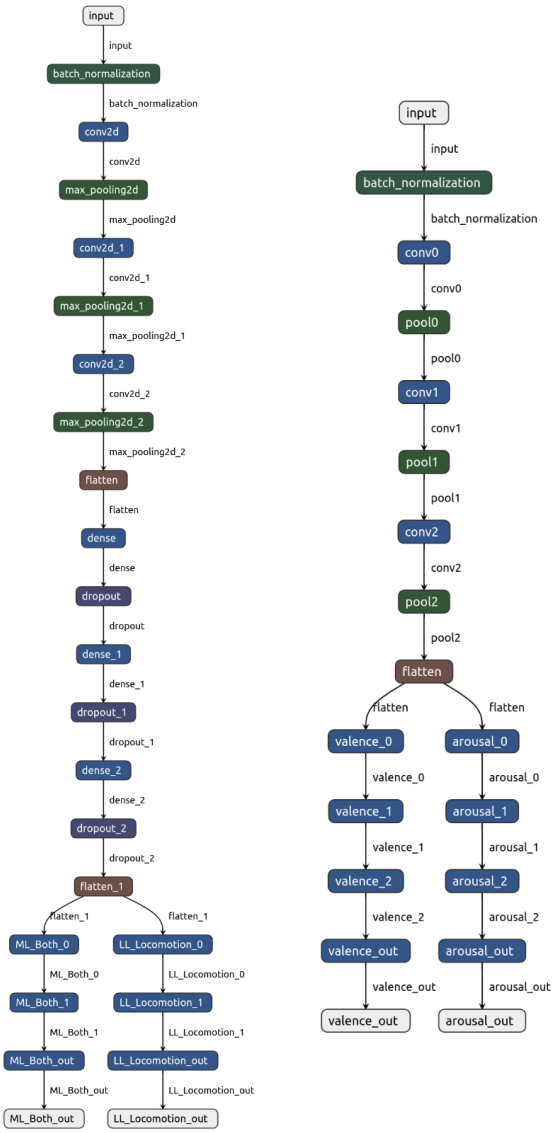

Рисунок 3.10 – Архітектура мережі HPS для OPPORTUNITY та DEAP

Із рисунку 3.11 можна побачити, як саме архітектура мережі SPS реалізує процес паралельного навчання між двома підмережами. Вони так само створені на основі базової згорткової нейронної мережі, проте різниця між HPS і SPS все ж таки присутня. Основна відмінність – відсутність шарів Dropout та спільних шарів Dense для мережі OPPORTUNITY, що йдуть після першого шару Flatten. Підмережі DEAP, в свою чергу, ідентичні до базової мережі. Так, мережа SPS все ще починається із спільного шару нормалізації, і одразу розгалуджується на дві підмережі, кожна із яких містить свої шари Dense та вирішує свою окрему підзадачу.

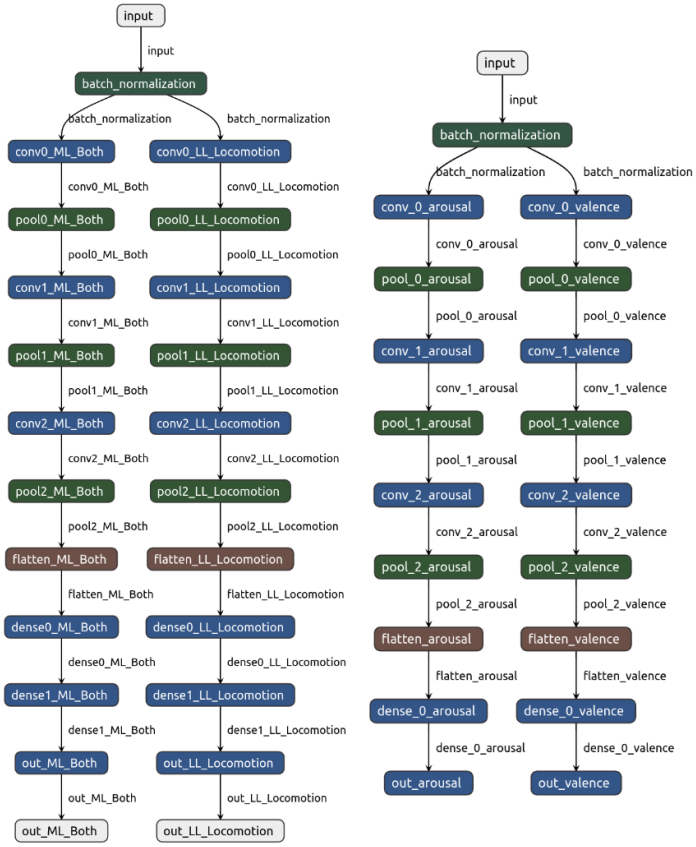

Рисунок 3.11 – Архітектура мережі SPS для OPPORTUNITY та DEAP

Мережа CSN реалізована подібно до мережі SPS, проте із використанням блоків зшивання для обміну інформацією між підмережами. Так, мережа починається із шару нормалізації, після чого, подібно до архітектури SPS, розгалужується на дві підмережі. Проте, після кожної пари згортки і об’єднання шарів реалізовано шар Cross-stitch та шар Unstack. Така архітектура створює цікавий цикл обміну інформацією: спочатку ознаки обробляються окремо, потім комбінуються через блок зшивання, а після цього знову розділяються для подальшої спеціалізованої обробки. Така структура повторюється тричі, що дозволяє мережі на різних рівнях абстракції визначати, які ознаки краще обробляти разом, а які окремо.

Як вже було зазначено раніше, окремий шар Unstack використовується для подолання проблеми «graph disconnect» при використанні методів TensorFlow для створення мережі, що виконує процес розділення тензорів.

В кінці архітектури, навіть після всіх етапів обміну інформацією, кожна гілка зберігає свій окремий вихід, що дозволяє мережі генерувати спеціалізовані прогнози для кожної задачі. Ці архітектури можна побачити на рисунку 3.12.

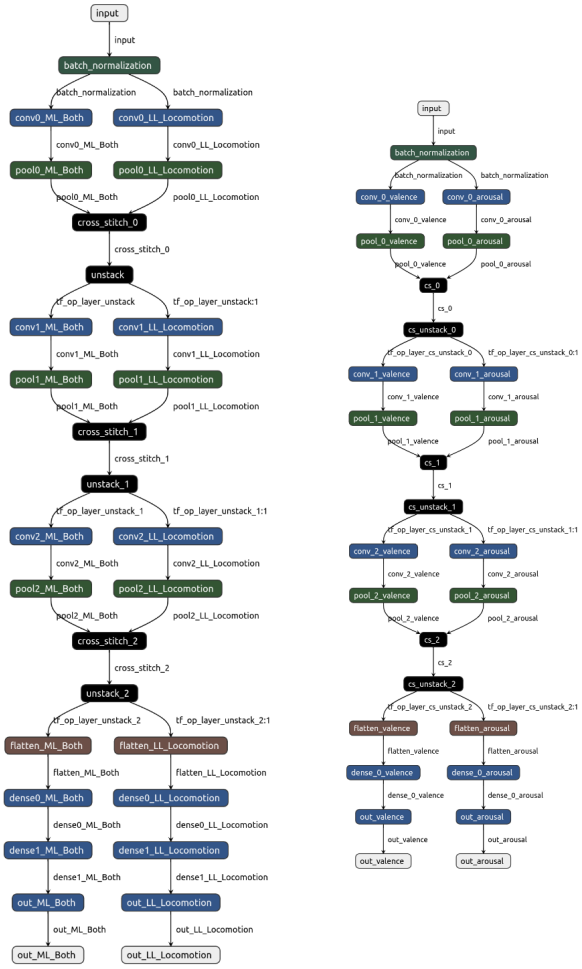

Рисунок 3.12 – Архітектура мережі CSN для OPPORTUNITY та DEAP

Більш детальні зображення будуть представлені у Додатку А, а реалізація цих архітектур – у Додатку Б.

4 РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

В попередньому розділі були детально описані архітектури мереж, які були використані для дослідження багатозадачного навчання. У цьому розділі результати будуть представлені графічним шляхом та детально описані методи оцінки проведених експериментів.

4.1 Процес оцінки

Як вже було наведено, стандартний процес оцінки та порівняння, що полягає у співставленні власних результатів з результатами із літератури та попередніх досліджень, є неідеальним і, навіть, часто помилковим, через неможливість отримання повного уявлення про роботу мереж. Так, хоч мережі STL і MTL можуть надавати схожі результати, велике значення також грають кількість часу та епох, що ці моделі потребували для досягнення цих результатів, а також сам процес досягнення і формування ваг. Ефективно ці дані можна зобразити лише графічно, порівнюючи різні методи, і саме на цьому і буде зосереджено цей розділ.

Так, після кожної епохи у процесі навчання, необхідні метрики, а саме категоріальна точність і показник F_1, були обчислені за допомогою вбудованих методів бібліотек Keras та sklearn. Функції зворотніх викликів бібліотеки Keras дозволили створення журналів та запису до них відповідних метрик.

Через стохастичний процес навчання, хмарки точок для кожного методу здебільшого слідували чітко видимим лініям тренду, але демонстрували велику кількість коливань від цих кривих. Так, для зображення тренду навчання у більш зрозумілому вигляді, до кривих був застосован алгоритм локально оціненого зглажування діаграми розсіювання (LOESS). Цей алгоритм використовує локальну регресію, щоб

підігнати плавну криву до діаграми розсіювання даних. Цей алгоритм є особливо ефективним при наявності викидів.

Для ілюстрації продуктивності кожної мережі були створені графіки, із результатами останніх 10 або 25 епох датасетів OPPORTUNITY та DEAP відповідно. Після цього були створені порівняльні таблиці.

4.2 Жорсткий обмін параметрами

4.2.1 Датасет OPPORTUNITY

Датасет OPPORTUNITY містить 7 класифікаційних завдань. Центральним об’єктом дослідження виступав канал ML_Both_Arms (позначений як завдання 6), вибір якого обґрунтовується попередніми дослідженнями, які було приведено раніше. Саме робота з цим каналом надає найкращі результати і тому основна робота мережі заключалася в покращені класифікації саме цього завдання.

В якості допоміжного завдання для першого експерименту було обрано канал LL_Locomotion (позначений як завдання 0), який характеризується базовими показниками людської активності. Цей канал включає класифікацію стану руху об’єкта з п’ятьма можливими мітками та додатковою міткою NULL.

Так, перший експеримент порівнював базову архітектуру однозадачного навчання із архітектурою багатозадачного навчання у вигляді мережі CNN, що складається з двох додаткових підмереж, створених після шарів dense. Метою цих підмереж було навчання специфічних для кожного завдання ознак.

Основними параметрами, на які було звернуто увагу, були ваги втрат та кількість спільних шарів, які потребувала мережа. Ці параметри були додані в процесі трансформації базової архітектури STL у формат MTL за допомогою підходу HPS.

В якості другого експерименту, спільні шари dense були видалені, а новий шар dropout із коефіцієнтом 0.4 був доданий до кожної підмережі після першого шару. Така модифікація дозволила дослідити вплив різних архітектурних рішень на ефективність багатозадачного навчання та визначити оптимальну конфігурацію мережі для вирішення поставлених завдань.

На рисунку 4.1 можна побачити результати першого експерименту. Результати оцінювалися за двома показниками – макро-F_1 і мікро-F_1. Основна різниця полягає в тому, що мікро-F_1 використовує індивідуальні істинно позитивні, хибно позитивні та хибно негативні значення для кожного класу, тобто не враховує пропорції класів у даних. Таким чином, макро-F_1 краще відображає точність на незбалансованих даних, і саме тому використовується як основна метрика у більшості досліджень. На рисунку можна побачити, що хоч MTL значно перевершує STL за мікро-F_1 метрикою, за макро-F_1 – результати схожі. Більш того, при видаленні спільних шарів, якість класифікації мережею значно знижувалася, і що використання шарів обчислення у індивідуальних підмережах є неоптимальним. Більш того, при визначенні двох задач як однаково важливих, мережа показувала кращі результати на останній епосі, тоді як показники при визначенні першої задачі як головної знижувалися впродовж навчання.

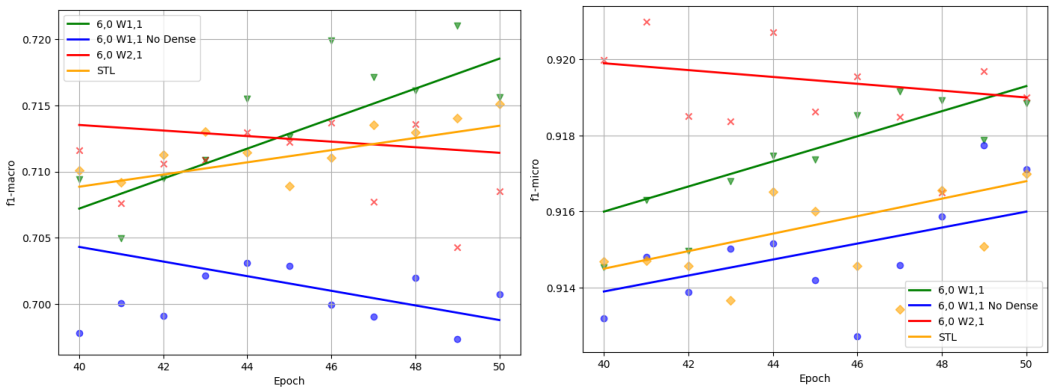

Рисунок 4.1 – Порівняння архітектур STL і HPS (OPPORTUNITY)

Другий експеримент полягав у створенні додаткових підзадач. Було порівняно підходи із двома задачами, чотирма задачами і усіма задачами, а також однозадачний підхід. З рисунку 4.2 можна побачити, що додавання трьох підзадач покращило результати роботи мережі, проте включення усіх каналів результувало у найгірших показниках, що лише доводить тезу про те, що вибір додаткових задач є одним із найголовніших завдань при створенні мережі MTL.

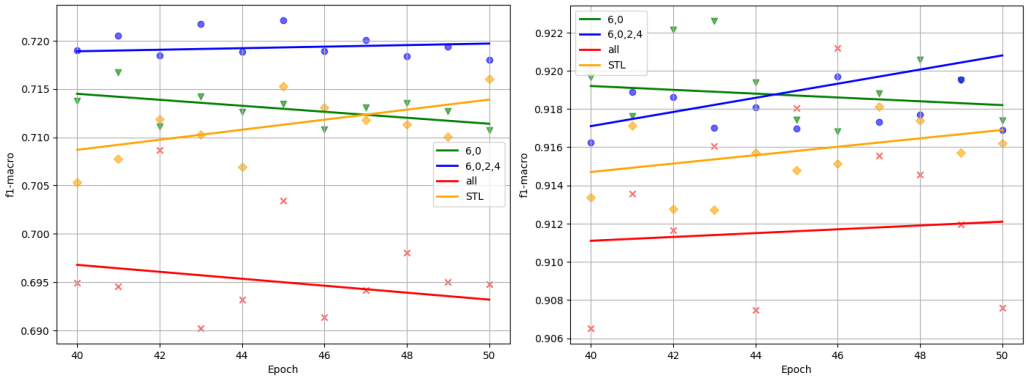

Рисунок 4.2 – Порівняння кількості завдань HPS (OPPORTUNITY)

4.2.2 Датасет DEAP

Експеременти проведені на датасеті DEAP були ідентичними попереднім, проте при класифікації емоції обидва завдання, що були обрані – завдання класифікації збудження та валентності – є однаково важливими, і для їх оцінки використовувалася спільна метрика точності.

На рисунку 4.3 можна побачити порівняння архітектури із спільними шарами і архітектури із окремими шарами. Результати схожі з експериментом над датасетом OPPORTUNITY – мережа зі спільними шарами є краще за однозадачну мережу, яка, в свою чергу, краща за мережі із індивідуальними.

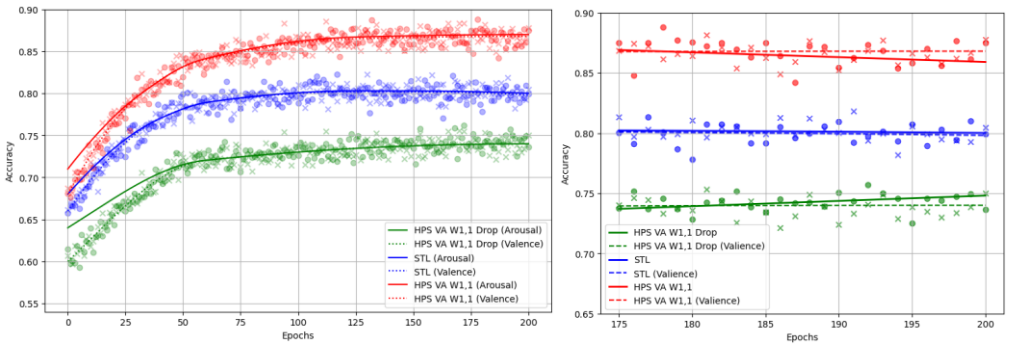

Рисунок 4.3 – Порівняння STL і HPS (DEAP)

4.2.3 Висновки

При дослідженні набору даних OPPORTUNITY спостерігалося невелике підвищення продуктивності, що підтверджується статистичним аналізом з використанням довірчого інтервалу у 95%, що можна побачити на рисунку 4.3 та таблиці 4.1. Проте, відсутність перетину показників ефективності між методами STL та HPS свідчить про все-таки відчутне покращення при використанні MTL, яке може збільшуватися і далі за рахунок краще оптимізованих гіперпараметрів. В той час як на наборі DEAP метод показав себе майже на 6% краще, що вже можна вважати хорошим результатом, який ще може бути суттєво покращеним.

Загалом, те, що метод HPS виявився найкращім із трьох, є досить незвичним результатом, що не був раніше зафіксованим у літературі. Цей метод є найпростішим, якщо порівнювати загальну архітектуру мереж, що має як свої недоліки, так і переваги. Оскільки спільна архітектура для підзавдань є статичною, і мережа не має змоги обирати, якими даними вона має ділитися, вибір завдань є надзвичайно важливим. Це, і складність у пошуку необхідних гіперпараметрів, є основними проблемами при створенні мережі HPS шляхом перетворення декількох базових мереж STL.

Так, потенційні переваги HPS все ще залишаються невизначеними. Чутливість методу до початкових параметрів вказує на значний потенціал

для оптимізації результатів через більш точне налаштування гіперпараметрів. Робота із вдосконалення та зменшення залежності методу від гіперпараметрів досі ведеться, і загалом є активною сферою дослідження. Сучасні дослідження пропонують інноваційні рішення для подолання існуючих обмежень, включаючи впровадження оцінки невизначеності для оптимізації вагових коефіцієнтів загальної функції [27] та розробку моделей з адаптивними швидкостями навчання для різних підзадач [14]. Ці розробки відкривають перспективи для істотного вдосконалення як методології реалізації, так і загальної ефективності мереж типу HPS.

4.3 М’який обмін параметрами

4.3.1 Датасет OPPORTUNITY

На відміну від методу жорсткого обміну параметрів, експерименти за допомогою м’якого обміну були проведені тільки над двозадачною мережею. Це обумовлено значно більшою кількістю параметрів, що мають бути навчені, через що процес навчання чотирьом задачам потребував набагато більше часу і ресурсів, що було визнано непродуктивним для цього дослідження.

Проте, основна різниця – додання нового гіперпараметру, а саме норми тензорного сліду, і його ваги. Основне дослідження, що описує цей параметр, визначило значення у 000.1 оптимальним. Це значення було прийняте як початкове та модифіковане у процесі експериментів. Оптимальним значенням норми було визначено 0.005. Проте, як можна зазначити із рисунку 4.4, усі варіанти методу SPH показали себе значно гіршими за стандартний метод STL.

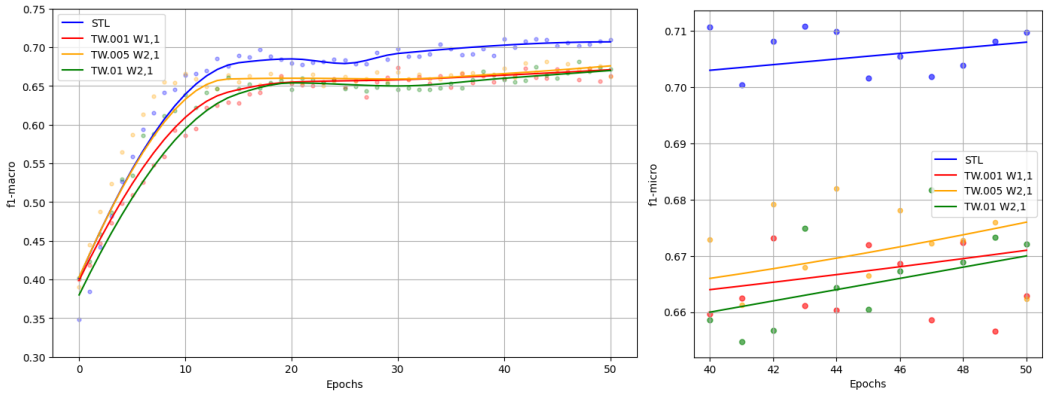

Рисунок 4.4 – Порівняння STL і SPS (OPPORTUNITY)

4.3.2 Датасет DEAP

Експерименти над датасетом DEAP повторювали такі ж над датасетом OPPORTUNITY. Проте, як вже було показано, цей метод демонструє значно гірші результати за однозадачне навчання. Хоч із графіків видно, що при значенні норми у 0.001 мережа SPH показує добрі результати, було вирішено зупинити процес навчання через набагато високі вимоги до часу і ресурсів – тренування 100 епох мережі зайняло майже 25 годин, і подальше навчання було визнано непродуктивним.

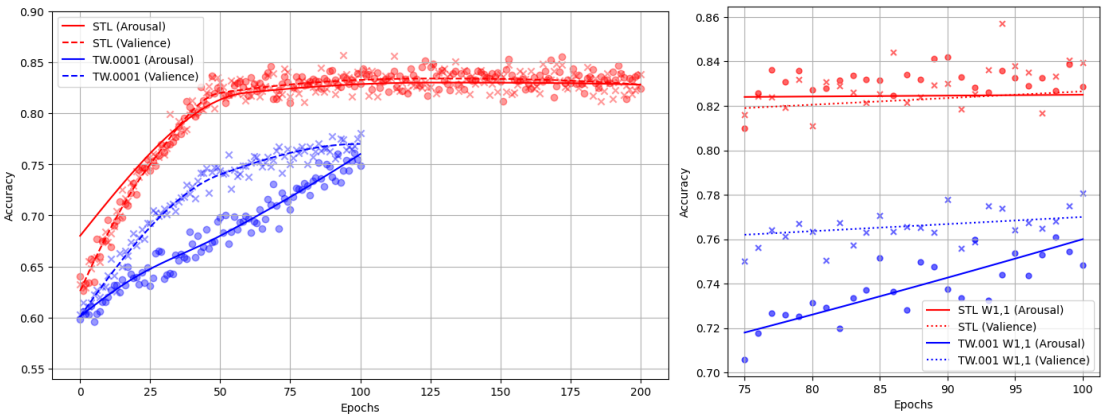

Рисунок 4.5 – Порівняння STL і SPS (DEAP)

4.3.3 Висновки

Архітектурний підхід SPS базується на відносно простому, але ефективному принципі паралельного навчання кількох мереж зі схожою архітектурою для різних завдань. Такі підмережі не мають прямих зв’язків, проте їх параметри змушені бути подібними за рахунок регуляризаторів, які спонукають параметри мереж до конвергенції.

В цьому дослідженні, в якості регуляризатора було обрано тензорну слідову норму, вперше представлену в роботі [13]. Нажаль, цей підхід не зміг покращити результати на обох наборах даних, що було доволі несподіваним результатом, так як у домені обробки природної мови цей підхід досягав надзвичайно високих результатів.

Однією із можливих причин є безпосередньо використання обраної слідової норми до даних у вигляді часових рядів, коли попередні дослідження в більшості були зосереджені на або синтетичних наборах, або низьковимірних реальних наборах [28]. Очевидно, що необхідне подальше дослідження у цьому напряму, адже методи HPS і SPS є ідейно доволі близькими один до одного, і настільки відмінні результати є дивним показником.

Так само було визначено, що при зменшенні або видаленні тензорної слідової норми як регулятора результати класифікації покращуються на датасеті DEAP і досягають середньої точності 0.816±0.005 для валентності та 0.808±0.01 для збудження, проте погіршуються майже у 30% для OPPORTUNITY.

Таким чином, ймовірніше за все проблема полягає у використанні чи реалізації TCH. І хоч інші метрики не були досліджені, вони включають в себе доволі класичні показники, такі як L1 і L2 [29] норми, або їх комбінації. Тож, подальші дослідження слід спрямувати на експериментальну оцінку ефективності цих альтернативних метрик, що потенційно може призвести до суттєвого покращення результатів.

4.4 Мережа пересічного зшивання

Так само, як при дослідженні SPH, експерименти були обмежені двома задачами. Розроблена архітектура включала три послідовні блоки зшивання, кожен з яких мав матричну структуру розміром 2×2. Ця конфігурація дозволяла забезпечити оптимальний баланс між складністю моделі та її здатністю до ефективного обміну інформацією між підзадачами.

Особливо важливим аспектом дослідження CPS стала можливість візуального аналізу процесів обміну інформацією через використання теплових карт. Такий метод візуалізації дозволяє відстежувати інтенсивність інформаційного обміну між різними шарами мережі і виявляти патерни та залежності в процесі навчання. Можливість візуалізувати еволюцію взаємодії між задачами на різних рівнях абстракції відкриває нові можливості для аналізу та оптимізації архітектури багатозадачних нейронних мереж.

4.4.1 Датасет OPPORTUNITY

Основним параметром, щодо якого порівнювалась якість мережі, була розмірність шарів dense. Так у першому шарі кількість нейронів завжди дорівнювала 1000, коли у другому вона варіювалася між 50 та 500. Результати цього експерименту можна побачити на рисунку 4.6.

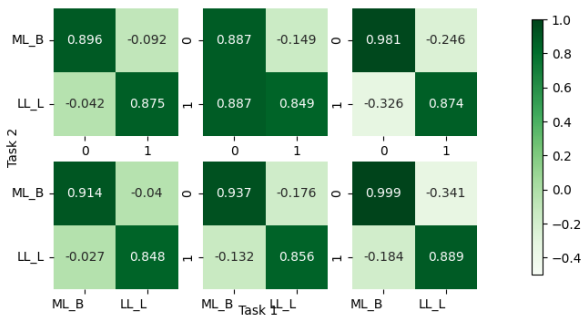

Рисунок 4.6 – Процес обміну параметрами трьох блоків CS

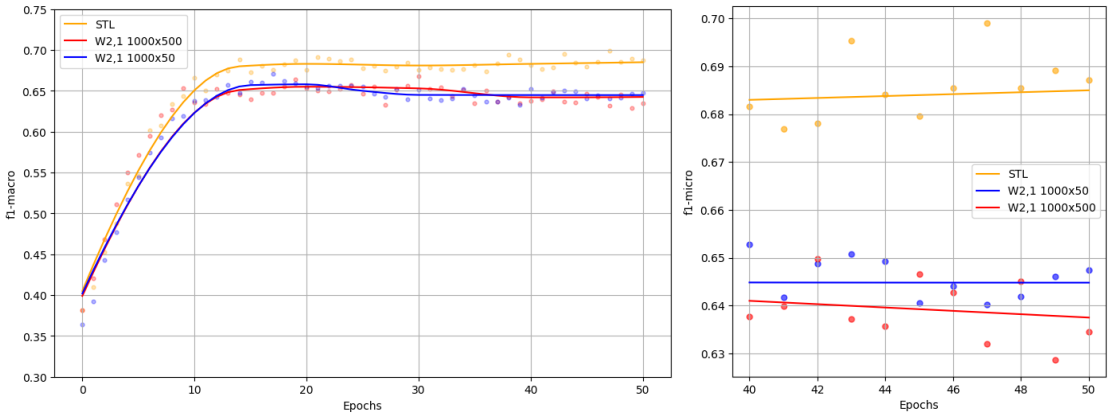

Рисунок 4.7 – Порівняння STL і CSN (OPPORTUNITY)

Можна побачити, що багатозадачний підхід все ще не здатен перевершити однозадачний підхід. Це можна пояснити тим, що мережа CPS є варіацією мережі SPH, яка є неідеальною архітектурою для вирішення подібних задач.

Більш того, при оцінці теплових карт можна побачити, що більшість інформації не була передана між підмережами. І хоч під час останнього блоку кількість обмінянної інформації була значно більшою за попередні, її все одно було недостатньо для оптимального використання цього підходу.

4.4.2 Датасет DEAP

Проте, при проведенні експериментів на датасеті DEAP було виявлено, що підхід CPS перевершує однозадачне навчання, що можна побачити на рисунку 4.7. При цьому, порівнюючи теплові карти, можна побачити ще одну відмінність від датасету OPPORTUNITY – обмін даних відбувався протягом усього процесу навчання, і хоч кількість переданної інформації все ще є відносно малою, це свідчить про доцільність покращення архітектури для подальшого використання.

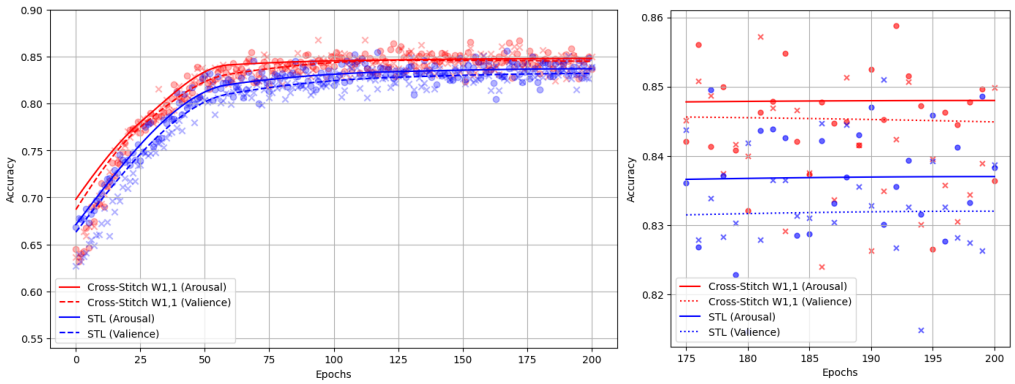

Рисунок 4.8 – Порівняння STL і CSN (DEAP)

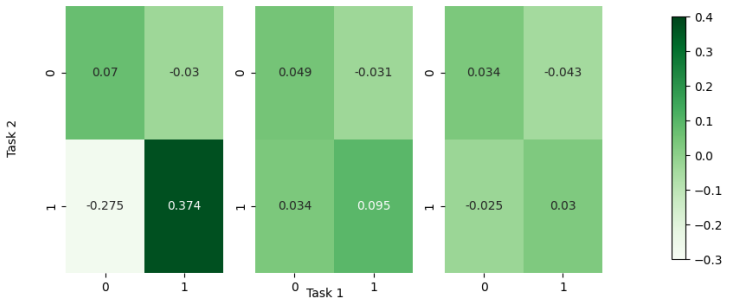

Рисунок 4.9 – Процес обміну параметрами трьох блоків CS

4.4.3 Висновки

Як вже було зазначено, метод пересічного зшивання є гібридним підходом, що поєднує переваги архітектур HPS та SPS. Його освновний принцип заключається в наданні мережі можливостей для самостійного вибіру та визначення ваг, які слід призначити для кожного входу блоку обміну, щоб вирішити, скільки інформації від інших завдань необхідно для кожного виходу.

Експериментальні результати підтверджують проміжну позицію CSN між методами SPS та HPS за показниками ефективності. При роботі з набором даних DEAP мережа продемонструвала вищу точність класифікації порівняно з однозадачним навчанням. Проте на наборі даних OPPORTUNITY спостерігалося зниження показників F_1.

Більш того, час на навчання на датасеті OPPORTUNITY значно перевищував аналогічний для базової моделі. Така проблема не спостерігалася на наборі DEAP. Це свідчить про значно вищу чутливість методу до характеристик вхідних даних порівняно з попередніми підходами, хоча теоретичне обґрунтування цього феномену наразі відсутнє. На наборі даних OPPORTUNITY архітектура CSN продемонструвала значне скорочення параметрів – з 6 мільйонів до 4.5 мільйонів порівняно з базовою STL-моделлю. Натомість, при роботі з набором DEAP спостерігалася протилежна тенденція: кількість параметрів зросла майже вдвічі, досягнувши 174 тисяч. Зазвичай, такі показники мали б свідчити про зворотнє, так як традиційно зменшення кількості параметрів асоціюється з кращою генералізацією та ефективнішим навчанням, тоді як збільшення параметрів часто свідчить про потенційну перенавченість моделі. Однак у нашому випадку результати експериментів демонструють протилежну картину – кращу продуктивність на наборі DEAP, де кількість параметрів зросла, і гірші показники на OPPORTUNITY, де відбулося скорочення параметрів. Такі результати вказують на необхідність глибшого дослідження механізмів взаємодії між архітектурою мережі та специфікою даних, які поки є відсутними через новізну архітектури.

Проте, мережа Cross-Stitch Network демонструє значний потенціал у сфері багатозадачного машинного навчання. Механізм автоматичного визначення оптимальних вагових коефіцієнтів, який дозволяє мережі самостійно налаштовувати рівень взаємодії між різними компонентами задачі, є надзвичайно перспективним підходом до розв’язання завдань багатозадачної класифікації.Технологія CSN все ще знаходиться на етапі активного розвитку. Існуючі дослідження [30] вже заклали міцний фундамент, але залишається ряд важливих напрямків для вдосконалення. Зокрема, необхідно глибше дослідити взаємозв’язок між архітектурою мережі та характеристиками вхідних даних, оптимізувати процес навчання

для різних типів завдань та розробити більш ефективні механізми балансування ресурсів між підзадачами.

4.5 Порівняння методів

Загалом, можна побачити, що експерименти над датасетом DEAP надавали кращі результати, ніж експерименти над датасетом OPPORTUNITY. Це можна побачити у таблиці 4.1.

Більш того, як було зазначено раніше, час на тренування також грав велику роль при оцінці мереж, особливо, якщо тренування відбувається на вілносно слабкій машині, як в цьому дослідженні.

Результати на датасеті OPPORTUNITY виявилися розчаровуючими. Загалом, різниця між STL і MTL або настільки незначна, що реалізація MTL не має сенсу, або досить велика у користь STL.

Таблиця 4.1 – Порівняння методів на датасеті OPPORTUNITY

Підхід	Варіант	F_1-macro ± CI	Час навчання(г)
STL		0.713 ± 0.004	2.06
HPS	6,0 W1,1	0.717 ± 0.004	2.2
	6,0 W2,1	0.712 ± 0.003	2.13
	6,0 W2,1 No Dense	0.698 ± 0.002	2.31
	6,0,2,4 W2,1,1,1	**0.720 ± 0.003**	2.29
	Усі завдання	0.693 ± 0.003	2.53
SPS	TW0.001 W1,1	0.671 ± 0.004	4.01
	TW0.005 W2,1	0.067 ± 0.005	3.95
	TW0.01 W2,1	0.67 ± 0.007	4.12
CSN	W2,1 1000x50	0.645 ± 0.004	4.67
	W2,1 1000x500	0.638 ± 0.002	4.46

Проте, як видно із таблиці 4.2, результати на датасеті DEAP були значно оптимістичнішими.

Таблиця 4.2 – Порівняння методів на датасеті DEAP

Підхід	Варіант	Точність V± CI	Точність A± CI	Час навчання
STL		0.801 ± 0.007	0.806 ± 0.004	5.2
HPS	**VA W1,1**	**0.85 ± 0.003**	**0.865 ± 0.003**	5.7
	VA W1,1 Drop	0.75 ± 0.006	0.741 ± 0.005	5.6
SPS	TW0.0001	0.772 ± 0.006	0.76 ± 0.008	25.3 (100 ep.)
	TW0.001	0.404 ± 0	0.562 ± 0	14.3 (52 ep.)
CSN	W1,1	0.832 ± 0.002	0.848 ± 0.002	6.11

Експерименти показали, що при аналізі датасету DEAP та схожих на нього, існують ситуації, де використання MTL є оптимальним, зокрема при використанні методу HPS. В той час метод SPS показував набагато гірші результати, як і дивлячись на показник точності класифікації, так і дивлячись на час, необхідний для навчання.

Загалом, так як метою цього дослідження була оцінка методів багатозадачного навчання та виявлення доцільності їх використання, її можна вважати успішною. Ця робота показала, що доцільність використання MTL цілком залежить від датасету, над яким мають проводитися операції, та можливості визначити підзавдання, що будуть органічно доповнювати одне одного.

ВИСНОВКИ

В результаті цієї роботи були досліджені методи багатозадачного навчання та проведена практична робота із якісного порівняння методів MTL із звичними методами STL.

В результаті теоретичних досліджень було визначено три основні архітектури MTL: мережа із жорстким обміном параметрами, м’яким обміном параметрами та мережа пересічного зшивання. Основні відмінності полягали у відношеннях підмереж між собою та їх можливостях обмінюватися інформацією. Так, архітектура HPS фактично використовує єдину мережу для підзавдань із спільними параметрами. Архітектура SPS навчає власні параметри для кожного із завдань та використовує систему регуляризаторів до приведення їх до загального вигляду. Архітектура CSN, в свою чергу, використовує блоки зшиття, чи обміну даними, що дозволяють підмережам вирішувати та передавати навчені параметри.

Реалізація обраних архітектур та методів була створена за допомогою мови програмування Python та її бібліотек TensorFlow і Keras.

Проведене дослідження підтвердило початкову гіпотезу про можливі переваги багатозадачного навчання порівняно із однозадачним навчанням. Проте, результати демонструють неоднорідну ефективність різних підходів порівняно із традиційними методами, які зазвичай простіші в реалізації та швидші в навчанні.

Метод жорсткого жорсткого обміну параметрів продемонстрував найвищу ефективність на обох наборах даних. Водночас метод м’якого обміну параметрів не зміг перевершити базові показники STL, потребуючи при цьому значно більше часу та ресурсів на навчання без помітного покращення продуктивності. Мережа пересічного зшивання показала неоднорідні результати: високу ефективність на наборі DEAP при незначному збільшенні часу навчання, але суттєво гірші показники на наборі OPPORTUNITY з істотним зростанням обчислювальних витрат.

Основною причиною погіршення показників було визначено неправильний вибір гіперпараметрів. Так, недостатня кількість проведених експериментів та протестованих гіперапараметрів суттєво вплинути на кінцеві результати, що вказує на необхідність подільших досліджень щодо оптимізації параметрів.

Попередня обробка даних також відіграє критичну роль у фінальних результатах. Набори даних OPPORTUNITY і DEAP вимагали суттєво різних підходів до попередньої обробки, що могло вплинути на ефективність об’єднанної архітектури, обраної на основі експериментів STL. Варто зазначити, що розробка методів попередньої обробки досі залишається активною сферою досліджень.

Так, специфіка набору даних DEAP полягала в мінімальній потребі попередньої обробки завдяки початковій чистоті даних. Проте відсутність покадрових міток змусила використовувати єдину мітку для всього експерименту, що могло призвести до втрати інформації про динаміку емоційних змін під час перегляду відео. Втім, це обмеження однаково впливало як на методи MTL, так і на STL.

Датасет OPPORTUNITY, в свою чергу, не був попередньо оброблений. Цей набір даних містить велику кількість нульових значень та ділянок, де суб’єкт не виконував жодної дії. Вирішити цю проблему можна було за рахунок видалення таких нефрагментованих ділянок, що, нажаль, не є можливим через недостатню кількість результуючих зразків. Це підкреслило необхідність розробки спеціалізованих методів обробки неповних даних у майбутніх дослідженнях.

Загалом, це дослідження довело, що використання багатозадачного навчання може бути доцільним при класифікації та обробці даних у вигляді часових рядів, зокрема активності людини. Проте, для точного визначення наскільки цей підхід є доцільним, подальші дослідження, що зможуть провести експерименти на більшій кількості архітектур, параметрів, метрик і регуляризаторів, необхідні.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Latent Multi-Task Architecture Learning / S. Ruder et al. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2019. Vol. 33. P. 4822–4829. URL: https://doi.org/10.1609/aaai.v33i01.33014822 (date of access: 08.01.2025).

2. Rumelhart D. E., Hinton G. E., Williams R. J. Learning representations by back-propagating errors. *Nature*. 1986. Vol. 323, no. 6088. P. 533–536. URL: https://doi.org/10.1038/323533a0 (date of access: 08.01.2025).

3. Bonaccorso G. Mastering Machine Learning Algorithms: Expert techniques to implement popular machine learning algorithms and fine-tune your models. Packt Publishing - ebooks Account, 2018. 576 p.

4. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit / R. H. R. Hahnloser et al. *Nature*. 2000. Vol. 405, no. 6789. P. 947–951. URL: https://doi.org/10.1038/35016072 (date of access: 08.01.2025).

5. Ghorbani A., Abid A., Zou J. Interpretation of Neural Networks Is Fragile. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2019. Vol. 33. P. 3681–3688. URL: https://doi.org/10.1609/aaai.v33i01.33013681 (date of access: 08.01.2025).

6. Bengio Y. Learning Deep Architectures for AI. *Foundations and Trends® in Machine Learning*. 2009. Vol. 2, no. 1. P. 1–127. URL: https://doi.org/10.1561/2200000006 (date of access: 08.01.2025).

7. Gradient-based learning applied to document recognition / Y. Lecun et al. *Proceedings of the IEEE*. 1998. Vol. 86, no. 11. P. 2278–2324. URL: https://doi.org/10.1109/5.726791 (date of access: 08.01.2025).

8. Johnson J., Karpathy A., Fei-Fei L. DenseCap: Fully Convolutional Localization Networks for Dense Captioning. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27–30

June 2016. 2016. URL: https://doi.org/10.1109/cvpr.2016.494 (date of access: 08.01.2025).

9. Hochreiter S., Schmidhuber J. Long Short-Term Memory. *Neural Computation*. 1997. Vol. 9, no. 8. P. 1735–1780. URL: https://doi.org/10.1162/neco.1997.9.8.1735 (date of access: 08.01.2025).

10. Ruder S. An Overview of Multi-Task Learning in Deep Neural Networks. *CoRR*. 2017. URL: https://doi.org/10.48550/arXiv.1706.05098 (date of access: 08.01.2025).

11. Caruana R. Multitask Learning. *Learning to Learn*. Boston, MA, 1998. P. 95–133. URL: https://doi.org/10.1007/978-1-4615-5529-2_5 (date of access: 08.01.2025).

12. Low Resource Dependency Parsing: Cross-lingual Parameter Sharing in a Neural Network Parser / L. Duong et al. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Beijing, China. Stroudsburg, PA, USA, 2015. URL: https://doi.org/10.3115/v1/p15-2139 (date of access: 08.01.2025).

13. Y. Yang and T. M. Hospedales. Trace Norm Regularised Deep Multi TaskLearning. CoRR. 2016. URL: https://doi.org/10.48550/arXiv.1606.04038 (date of access: 08.01.2025).

14. Facial Landmark Detection by Deep Multi-task Learning / Z. Zhang et al. *Computer Vision – ECCV 2014*. Cham, 2014. P. 94–108. URL: https://doi.org/10.1007/978-3-319-10599-4_7 (date of access: 08.01.2025).

15. Cross-Stitch Networks for Multi-task Learning / I. Misra et al. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27–30 June 2016. 2016. URL: https://doi.org/10.1109/cvpr.2016.433 (date of access: 08.01.2025).

16. Tucker L. R. Some mathematical notes on three-mode factor analysis. *Psychometrika*. 1966. Vol. 31, no. 3. P. 279–311. URL: https://doi.org/10.1007/bf02289464 (date of access: 08.01.2025).

17. Collecting complex activity datasets in highly rich networked sensor environments / D. Roggen et al. *2010 Seventh International Conference on Networked Sensing Systems (INSS)*, Kassel, Germany, 15–18 June 2010. 2010. URL: https://doi.org/10.1109/inss.2010.5573462 (date of access: 08.01.2025).

18. Comparison of Feature Learning Methods for Human Activity Recognition Using Wearable Sensors / F. Li et al. *Sensors*. 2018. Vol. 18, no. 3. P. 679. URL: https://doi.org/10.3390/s18020679 (date of access: 08.01.2025).

19. DEAP: A Database for Emotion Analysis ;Using Physiological Signals / S. Koelstra et al. *IEEE Transactions on Affective Computing*. 2012. Vol. 3, no. 1. P. 18–31. URL: https://doi.org/10.1109/t-affc.2011.15 (date of access: 08.01.2025).

20. Universals and cultural differences in the judgments of facial expressions of emotion. / P. Ekman et al. *Journal of Personality and Social Psychology*. 1987. Vol. 53, no. 4. P. 712–717. URL: https://doi.org/10.1037/0022-3514.53.4.712 (date of access: 08.01.2025).

21. Bochnak J., Kucharz W., Shiota M. On algebraicity of global real analytic sets and functions. *Inventiones Mathematicae*. 1982. Vol. 70, no. 1. P. 115–156. URL: https://doi.org/10.1007/bf01393201 (date of access: 08.01.2025).

22. Russell J. A. Emotion, core affect, and psychological construction. *Cognition & Emotion*. 2009. Vol. 23, no. 7. P. 1259–1283. URL: https://doi.org/10.1080/02699930902809375 (date of access: 08.01.2025).

23. Bradley M. M., Lang P. J. Measuring emotion: The self-assessment manikin and the semantic differential. *Journal of Behavior Therapy and Experimental Psychiatry*. 1994. Vol. 25, no. 1. P. 49–59. URL: https://doi.org/10.1016/0005-7916(94)90063-9 (date of access: 08.01.2025).

24. Liu Y., Sourina O. Real-Time Subject-Dependent EEG-Based Emotion Recognition Algorithm. *Transactions on Computational Science XXIII*. Berlin, Heidelberg, 2014. P. 199–223. URL: https://doi.org/10.1007/978-3-662-43790-2_11 (date of access: 08.01.2025).

25. Investigation of window size in classification of EEG-emotion signal with wavelet entropy and support vector machine / H. Candra et al. *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Milan, 25–29 August 2015. 2015. URL: https://doi.org/10.1109/embc.2015.7320065 (date of access: 08.01.2025).

26. Cross-Subject EEG Feature Selection for Emotion Recognition Using Transfer Recursive Feature Elimination / Z. Yin et al. *Frontiers in Neurorobotics*. 2017. Vol. 11. URL: https://doi.org/10.3389/fnbot.2017.00019 (date of access: 08.01.2025).

27. Cipolla R., Gal Y., Kendall A. Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 18–23 June 2018. 2018. URL: https://doi.org/10.1109/cvpr.2018.00781 (date of access: 08.01.2025).

28. Argyriou A., Evgeniou T., Pontil M. Convex Multi-Task Feature Learning. *SSRN Electronic Journal*. 2007. URL: https://doi.org/10.2139/ssrn.1031158 (date of access: 08.01.2025).

29. Low Resource Dependency Parsing: Cross-lingual Parameter Sharing in a Neural Network Parser / L. Duong et al. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Beijing, China. Stroudsburg, PA, USA, 2015. URL: https://doi.org/10.3115/v1/p15-2139 (date of access: 08.01.2025).

30. Kokkinos I. UberNet: Training a Universal Convolutional Neural Network for Low-, Mid-, and High-Level Vision Using Diverse Datasets and Limited Memory. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 21–26 July 2017. 2017. URL: https://doi.org/10.1109/cvpr.2017.579 (date of access: 08.01.2025).