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)

Гнучкість нейронних мереж як метод змагального захисту
(тема)

Виконав:
студент 2 курсу, групи СШІм-20-3
Малик Д. Г.
(прізвище, ініціали)

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

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

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник професор Терзіян В. Я.
(посада, прізвище, ініціали)

Допускається до захисту

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

2022 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-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

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

# ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Малик Діані Геннадіївні
(прізвище, ім'я, по батькові)

1. Тема роботи Гнучкість нейронних мереж як метод змагального захисту

затверджена наказом університету від 24 березня 20 22 р. № 414Ст
2. Термін подання студентом роботи до екзаменаційної комісії 11 травня 20 22 р.
3. Вихідні дані до роботи Науково-технічні публікації, дані Інтернет-джерел та відомих наукових проектів щодо розробки та дослідження адаптивних функцій, матеріали галузі змагального машинного навчання, документація python, pytorch, tensorflow

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі, виявлення наявних проблем та існуючих способів рішення
2) Пошук та дослідження власного рішення проблеми, постановка задач дослідження
3) Теоретичне дослідження конфронтаційних атак та методів захисту
4) Теоретичне дослідження адаптивних функцій та їх взаємозв'язку з конфронтаційною стійкістю
5) Експериментальне дослідження висунутих гіпотез, аналіз результатів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1 )

| Найменування розділу | Консультант (посада, прізвище, ім'я, по батькові) | Позначка консультанта про виконання розділу | |
|---|---|---|---|
| | | підпис | дата |
| | | | |
| | | | |

**КАЛЕНДАРНИЙ ПЛАН**

| № | Назва етапів роботи | Терміни виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Аналіз предметної галузі, вибір засобів та методів | 04.04.2022 | виконано |
| 2 | Теоретичне дослідження конфронтаційних загроз | 08.04.2022 | виконано |
| 3 | Аналіз та порівняння існуючих адаптивних пулінгів | 12.04.2022 | виконано |
| 4 | Розробка та теоретичне дослідження власного рішення | 17.04.2022 | виконано |
| 5 | Практична реалізація та проведення експериментів | 24.04.2022 | виконано |
| 6 | Написання пояснювальної записки | 30.04.2022 | виконано |
| 7 | Захист перед ЕК | 11.05.2022 | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Дата видачі завдання 28 березня 20 22 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка: 77 с., 42 рис., 4 табл., 14 фор., 3 дод., 45 джерел.

АДАПТИВНІСТЬ НЕЙРОННИХ МЕРЕЖ, КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ, КОНФРОНТАЦІЙНІ АТАКИ, КОНФРОНТАЦІЙНІ ПРИКЛАДИ, КОНФРОНТАЦІЙНА СТІЙКІСТЬ.

Об’єкт дослідження – вразливість глибоких нейронних мереж до конфронтаційних прикладів.

Предмет дослідження – адаптивні елементи у нейронних мережах, взаємозв’язок гнучкості моделей із конфронтаційною стійкістю.

Мета роботи – розробити гнучкі елементи нейронних мереж, оцінити вплив адаптивності на стійкість моделей до змагальних прикладів.

Методи дослідження – аналіз технічної літератури та досліджень у предметній області, моделювання конфронтаційних атак, порівняння різноманітних гнучких елементів моделей, експериментальне дослідження адаптивних елементів та їх робастності. В якості мови програмування використовується python3. Також застосовується фреймворк для машинного навчання pytorch. В роботі досліджено стійкість адаптивних елементів нейронних мереж до конфронтаційних атак. Порівняно різноманітні гнучкі елементи моделей та запропоновано власний варіант. Змодельовано різноманітні види конфронтаційних атак.

В результаті досліджень запропоновано стійкішу до змагальних прикладів архітектуру нейронної мережі з використанням адаптивних елементів. Пропонована архітектура має кращі показники надійності та безпеки. Розробка може бути інтегрована до застосунків або веб-сервісів, що потребують більш високого рівня безпеки. Можливе комбінування запропонованого методу з іншими типами змагального захисту.

# ABSTRACT

Explanatory note: 77 p., 42 fig., 4 tabl., 14 form., 3 ann., 45 sources.

ADAPTIVENESS IN NEURAL NETWORKS, ADVERSARIAL ATTACK, ADVERSARIAL EXAMPLE, ADVERSARIAL ROBUSTNES, IMAGE CLASSIFICATION.

The object of research is the weakness of deep neural networks to adversarial examples.

The subject of research are adaptive elements in neural networks and relantionships between model adaptivness and adversarial robustness.

The purpose of the work is to develop adaptive layers for neural network and evaluate their influence on protection against adversarial attacks.

Research methods – analysis of theoretical sources, technical literature, modelling of adversarial attacks, comparison of methods for generating adversarial examples, experimental research of adversarial training methods.

Research methods – analysis of technical literature and researches, modelling of adversarial attacks, comparison of adaptive elements of neural networks, experimental research of adversarial robustness of adaptive layers.

As a programming language, python3 is used. Also used machine learning framework PyTorch.

Adversatial robustness of various adaptive layers was investigated. The comparison of different adaptive elements was performed. Also developed our own adaptive layers. Various types of adversarial attacks are modelled. As a result of the research, we propose and implement more secure to adversarial examples architecture of neural network with adaptive elements.

The proposed architecture is more safe and reliable. The development can be integrated into applications or services that require higher level of security. Also this defense method can be combined with other adversarial defenses.

**ЗМІСТ**

# ПЕРЕЛІК СКОРОЧЕНЬ ТА ТЕРМІНІВ

Адаптивність / Гнучкість – здатність математичної функції змінювати свій вид для більш точного підлаштування до даних під час навчання;

Класифікатор – нейронна мережа, що застосовується для передбачення класу наданих даних;

Зловмисник / Противник – описує агента, який створює змагальний приклад. Однак в деяких випадках сам приклад також називають противником.

Змагальний шум – ретельно підібрані конфронтаційні збурення, що додаються до початкових даних для створення конфронтаційного прикладу;

Змагальний приклад /зображення – це модифікована версія чистого зображення, яке навмисно збурюється (наприклад, додаванням шуму), щоб заплутати / обдурити техніку машинного навчання, таку як глибокі нейронні мережі;

Модель загрози – описує тип потенційних атак, що поділяються за підходом, наприклад, атаки чорних ящиків;

AML – Adversarial Machine Learning – Конфронтаційне машинне навчання – галузь науки, що знаходиться на перетині кібребезпеки та глибокого навчання;

Adversarial environment – Конфронтаційне середовище – поняття з теорії ігор, що описує середовище, де існує прямий противник, що протидіє гравцю, як, наприклад, у шахах.

Adversarial training – метод конфронтаційного захисту, який полягає у використанні і змагальних, і чистих зображень, для навчання моделей машинного навчання;

CNN – convolutional neural network – згорткова нейронна мережа;

GAN – generative adversarial network – генеративно-змагальна мережа;

GLM – Generalized Lehmer Mean – запропоноване у роботі функція адаптивного середнього значення;

GLP – Generalized Lehmer Pooling – запропонована у роботі адаптивна субдискретизація, заснована на GLM;

PGD – Projective Gradient Descent – назва конфронтаційної атаки, що застосовується для створення конфронтаційних прикладів;

Pooling – операція субдискретизації, що застосовується у нейронних мережах;

Transferability – Переносимість – властивість змагального прикладу залишатися ефективним навіть для моделей, відмінних від тієї, яка використовується для його створення;

V1 – Primary Visual Cortex – Зорова кора головного мозку – це область кори головного мозку, яка обробляє візуальну інформацію.

# ВСТУП

В нинішньому світі активно відбувається перехід до індустрії нового покоління, яке називають поколінням 4.0 [1]. Інтелектуальні моделі все частіше використовуються в критичних інфраструктурах, таких як медицина, авіоніка, виробництво, робототехніка та інші. Нейронні мережі захищають наші ПК від вірусів та спаму, а оплачувати товари можливо за допомогою розпізнавання обличчя. Тож безпека таких моделей стає надзвичайно важливим питанням [2].

Зараз відомо, що більшість глибоких мереж вразливі до ретельно збурених даних, які називають конфронтаційними прикладами. Додані збурення достатньо малі, щоб не впливати на сприйняття людини або зовсім не бути помітними, проте здатні обманути модель.

При цьому варто враховувати, що змагальні приклади не є проблемою лише цифрового світу або деякої конкретної області глибокого навчання. Вони можуть бути застосовані майже у будь яких задачах та видах даних, не зважаючи, текст це, аудіо чи картинки, більш того, конфронтаційні приклади можуть бути створені і в реальному світі у вигляді деяких носимих об’єктів або патчів.

Дослідженням цього феномену займається галузь конфронтаційного машинного навчання, метою якої є адаптація сучасних розумних алгоритмів до безпечного функціонування у конфронтаційному середовищі [3].

В одночас, невпинно продовжується розвиток самих мереж, зокрема, досліджуються можливості застосування більш гнучких та біологічно відповідних елементів. Відомо, що характеристики мережі, такі як її глибина та продуктивність певним чином пов'язані з конфронтаційною стійкістю [4].

Тож, в даній роботі висувається гіпотеза, що адаптивність мережі також має вплив на захищеність мережі від конфронтаційних прикладів.

Метою даної роботи є дослідження впливу адаптивних елементів на

захищеність нейронної мережі від конфронтаційних атак на прикладі прошарків субдискретизації, аналіз існуючих гнучких прошарків пулінгу та розробка власного.

Актуальність роботи забезпечується декількома чинниками. Головним з них є зростання поширеності використання глибоких моделей, які наразі присутні в житті майже кожної людини і можуть прямо впливати на наше життя. Адже навіть голосовий асистент, такий як Siri, може бути атаковано. Крім того, в сучасному світі триває гібридна війна, значення кібербезпеки важко переоцінити, так як постійно виникають нові загрози. Відповідно, зростає потреба у методах захисту.

Не менш важливою причиною є потреба у освітленні викликів галузі конфронтаційного машинного навчання, сприянні її активному розвитку.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

## 1.1 Змагальне машинне навчання

Нейронні мережі здатні опрацьовувати великі об'єми даних, навчатись закономірностям та в майбутньому використовувати отримані знання для виведення на прикладах, з якими не зустрічалися раніше. Завдяки цьому у багатьох областях стало можливим замінити людину-експерта за допомогою інтелектуальної моделі [5]. Наразі частота використання нейронних мереж у різноманітних сферах значно зросла, їх популярність стрімко поширюється.

З іншого боку, безпечність застосування цих моделей все ще залишається відкритим питанням. У 2004 році Dalvi вперше виконав змагальну атаку на лінійний класифікатор [6]. Пізніше, у 2013 Szegedy та ін. виявили та дослідили деякі неоднозначні властивості інтелектуальних моделей, серед яких була вразливість до конфронтаційних прикладів – спеціально підібраних даних з непомітними збуреннями, що здатні суттєво впливати на роботу мережі [7].

Публікація Dalvi започаткували створення нової галузі науки про дані – adversarial machine learning (конфронтаційне машинне навчання), а робота Szegedy et al. призвела до початку її активного розвитку. Задачами галузі AML є дослідження вразливостей інтелектуальних моделей та алгоритмів, розробка як нових методів атак, так і пошуку ефективних методів протидії [8].

Найбільш провідними дослідниками цієї галузі є N. Carlini, D. Wagner, Biggio B., C. Szegedy, Ian J. Goodfellow, N. Papernot, Madry A., F. Tramèr та інші. Крім того, дослідженнями в цій сфері займаються провідні компаніїї, такі як Google, Facebook, DeepMind та інші.

## 1.2 Оцінка загрози конфронтаційних атак

Згадані збурення називають змагальним шумом, а утворені за їх допомогою дані – змагальними або конфронтаційними прикладами.

Змагальний шум відрізняється від випадкових викривлень та іншого природного шуму, що можуть виникати на зображеннях. Його числові значення та розташування ретельно підбираються зловмисником для обману інтелектуальної моделі.

Для підбору таких збурень і створення змагальних прикладів застосовуються змагальні атаки. Наразі відомо, що конфронтаційні атаки можуть застосовуватись і під час тренування, і під час використання моделі (data poisoning або evasion attack відповідно) [9]. При цьому зловмисник може працювати з мережою як з «білим» або «чорним» ящиком, тобто, прямий доступ до моделі підвищує шанс на успіх, проте не є необхідністю.

Наведемо деякі можливості конфронтаційних атак:

– створення «back door» у моделях [10];

– викривлення передбачень аж до неможливості роботи мережі;

– можливість «обходу» систем інфраструктури, що спираються на інтелектуальні моделі;

– у деяких межах, перепрограмування мереж для виконання інших задач;

– атаки на приватність, що дозволяють деанонімізувати (рис. 1.1) використані під час навчання дані [11];

– «викрадення» моделей (тобто підбір аналогічної архітектури та навчених вагів) [12] і т.д.


Рисунок 1.1 – Приклад відновлення даних з тренувального датасета

Кількість досліджень з напряму змагального машинного навчання невпинно зростає (рис. 1.2). Так, в 2022 році було опубліковано близько 5000 робіт, тоді як у 2014 їх налічувалось близько ста [13]. Не дивлячись на це, галузь все ще потребує більшої уваги, нових досліджень та рішень.

Застосування конфронтаційного захисту як обов'язкового етапу при створенні моделей на даний момент не впроваджено через недостатню дослідженість методів, відсутність зрозумілого алгоритму дій, слабку освітленість галузі AML та її нових загроз.

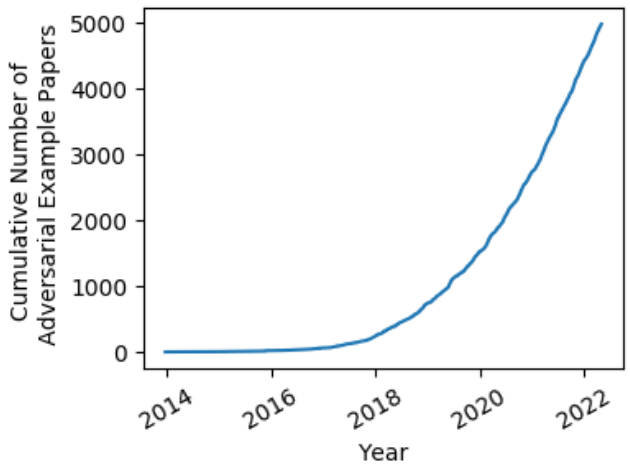


Рисунок 1.2 – Кількість робіт у галузі AML

Отож, не дивлячись на очевидну загрозу, широка практика змагального захисту нейронних мереж наразі відсутня. На даний момент основна частина досліджень та зусиль спрямована на досягнення найвищої точності моделей. При цьому значно менше уваги приділяється безпеці і стабільності цих моделей. На жаль, такі пріорітети призвели до виникнення серйозних проблем безпеки.

## 1.3 Методи конфронтаційного захисту

Методи конфронтаційного захисту бувають реактивними та проактивними [14]. Реактивними називають такі методи, які намагаються трансформувати вхідні дані деяким чином або виявити конфронтаційні приклади. Інші їх назва – захисні оболонки. Проактивні методи передбачають попередню підготовку до можливих атак. Прикладами є захисна дистиляція, змагальне тренування, вибір специфічних архітектур.

Реактивні методи загалом не є універсальними, так як кількість методів атак постійно зростає і природа конфронтаційних збурень може різнитись [15].

Останнім часом було запропоновано багато проактивних методів захисту, таких як нові евристичні алгоритми або методи гарантованого захисту. Проте найбільш відомим та ефективним серед них залишається конфронтаційне тренування [16]. Цей метод дозволяє достатньо ефективно протидіяти як відомим, так і невідомим атакам, проте значно збільшує час навчання моделі та може негативно вплинути на її метрики.

Отож, застосування додаткових методів захисту часто потребує деяких компромісів або є доволі складним. Крім того, велика частина запропонованих методів захисту з часом компроментуються, підбираються методи їх обходу.

Оскільки на сьогоднішній день мало що відомо про характер і причини «успіху» змагальних атак, велике значення мають не тільки

дослідження, спрямовані на розробку методів захисту від атак, але і роботи, спрямовані на їх безпосереднє вивчення. Тому наразі дослідження методів конфронтаційного захисту є відкритою задачею у галузі AML.

## 1.4 Адаптивні прошарки

В одночас, останні дослідження показують, що можливо підвищити початкову конфронтаційну стійкість мереж за рахунок використання деяких специфічних елементів, без додаткових процедур. Наприклад, функції активації низької кривизни [17] або більш біологічно відповідний блок V1 [18] дозволяють надійніше захищати моделі лише за рахунок заміни деяких їх прошарків.

В попередніх дослідженнях також було помічено, що використання різних активаціних функцій може чинити значний ефект на результати навчання, наприклад, зменшувати час тренування GAN мереж та покращувати ефективність змагальних атак [19]. Найкращих результатів було досягнуто при використанні адаптивних функцій (рис. 1.3).

Рисунок 1.3 –Змагальні приклади створені GAN з адаптивними активаціями

Адаптивність є однією з найважливіших характеристик біологічних організмів. Як відомо, глибокі нейронні мережі засновані на імітації систем біологічних організмів, таких як візуальні кортекси (рис. 1.4) та інше [20].

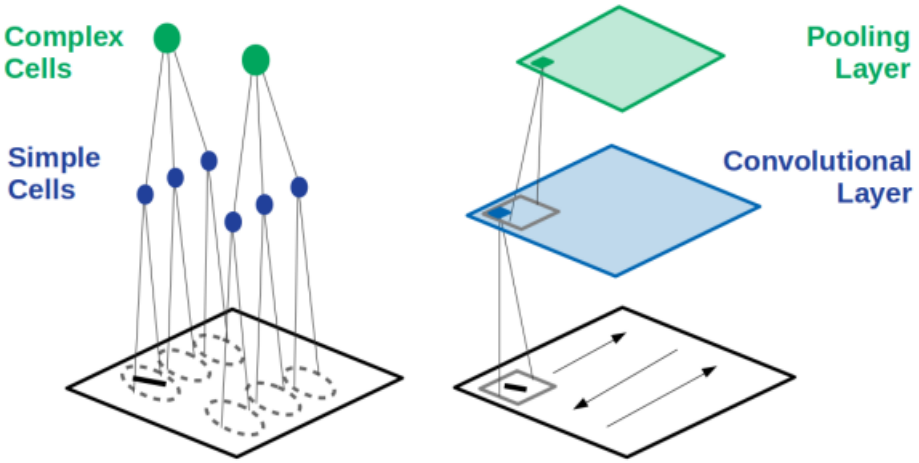


Рисунок 1.4 – Імітація візуального кортексу за допомогою нейронних мереж

Вважається, що глибокі нейронні мережі імітують реакції біологічних організмів доволі точно. Було проведено експеримент, в якому тваринам та нейронній мережі показували однакові зображення. Органи зору у відповідь на деяку отриману інформацію виконують деяку передобробку, після чого сигнал передається до відповідних зон головного мозку, де в свою чергу спостерігається активація певних елементів. Значення відповідних активацій у нейронної мережі та тварин у експерименті зчитувались та порівнювались. Відповідність сигналів була настільки суттєвою, що навіть дозволяла за виходами нейронної мережі передбачати активаційні сигнали біологічних організмів (рис. 1.5).

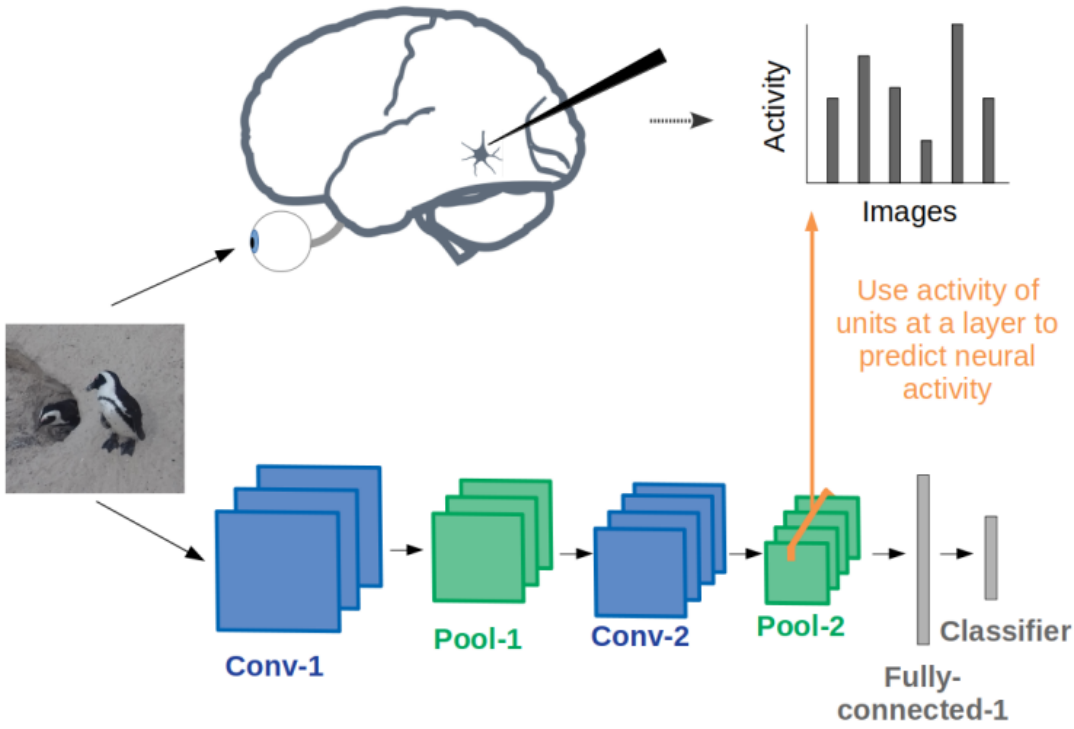


Рисунок 1.5 – Передбачення сигналів мозку тварин за виходами нейронної мережі

Існують і зворотні приклади. Dappelo et al. досліджували сигнали мозку шимпанзе, щоб відповідним чином налаштували фіксований блок V1 та застосувати його як вхідний блок конволюційної мережі [21]. В результаті використання більш біологічно відповідного візуального кортексу, стійкість класифікаторів до змагальних атак та природних спотворень було покращено. Один з дослідників, Marques T., заявив про це, що AI все ще має багато чого навчатись від нейронауки.

Однією з існуючих відмінностей наразі є те, що для мозку людини більш властивими є нелінійні функції. Цей факт також говорить на користь необхідності дослідження застосування гнучких математичних функцій, так як вони зазвичай містять нелінійні обчислення. Виходячи з досліджень [22], можна зробити припущення, що гнучкі елементи за рахунок більшої біологічної подібності будуть початково стійкішими до такого виду загроз.

Тож надання більшої гнучкості елементам інтелектуальних моделей є логічним наступним кроком розвитку нейронних мереж. Особливо важливим це є для галузі AML, так як добре відомо, що конфронтаційні збурення специфічні саме для штучних нейронних мереж і майже не помітні для людини (рис. 1.6) [23], тобто, основною причиною проблеми є різниця у сприйнятті даних людиною та штучною інтелектуальною моделлю.

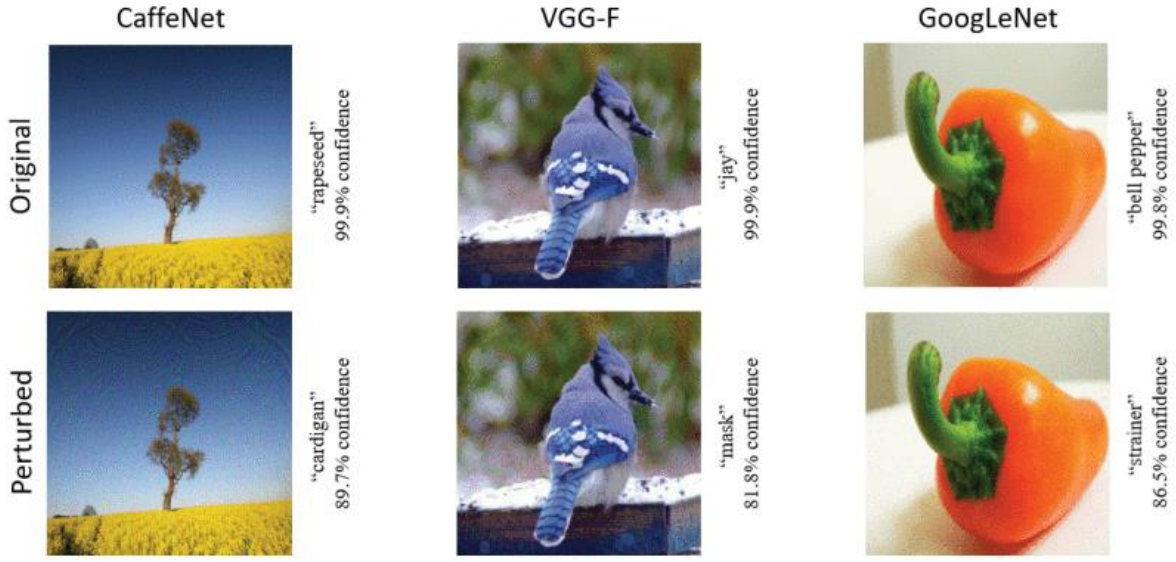


Рисунок 1.6 – Змагальні приклади

Адаптивні методи застосовують додаткові гіперпараметри, що навчаються. Це дозволяє замість фіксованої математичної функції використовувати деяке сімейство функцій. Конкретний вид функції обирається під час навчання за допомогою зміни гіперпараметрів, тож, такі функції адаптуються до конкретної задачі.

Отож, модель з гнучкими прошарками може нібито змінювати архітектуру в процесі навчання. Така мінливість може ускладнити для зловмисника проведення конфронтаційної атаки.

Підсумовуючи, застосування гнучких прошарків має бути досліджено у якості можливого варіанту забезпечення певного ступеню нативного захисту моделей від конфронтаційних прикладів.

## 1.5 Вибір інтелектуальної моделі та задачі

Конфронтаційні атаки можуть застосовуватися до багатьох типів задач, таких як классифікація, регрессія, генерування даних та інші. При цьому область може будь-якою, змагальні збурення успішно застосовують як для зображень, так і до текстів або аудіозаписів (рис. 1.7), або навіть коду зловмисних програм (мальварів) для обходу антивірусів або спам-фільтрів [24].

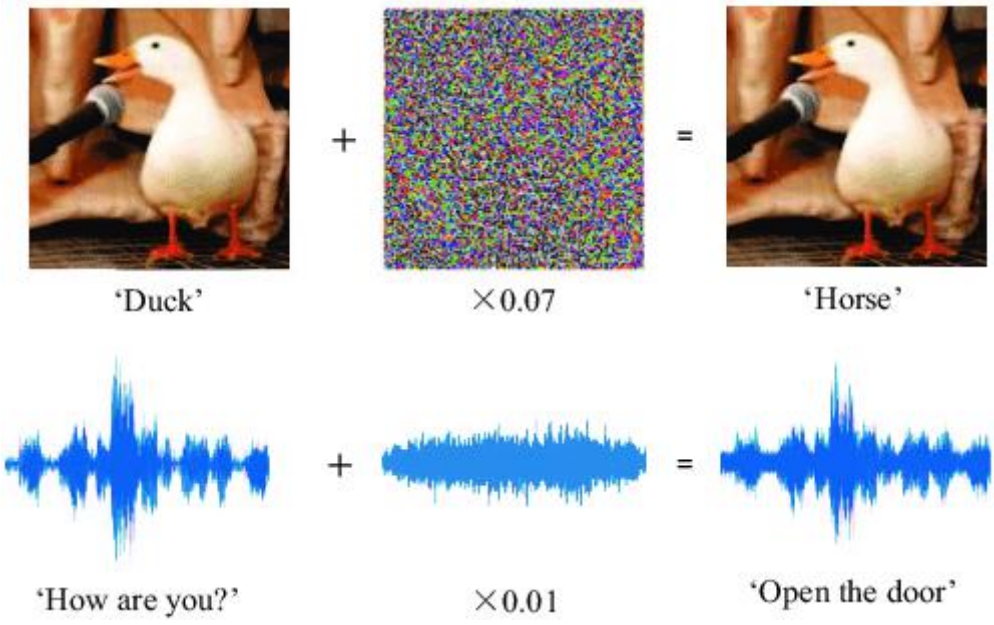


Рисунок 1.7 – Затосування змагального шуму до різних типів даних

Проте для досліджень було вирішено обрати область компьютерного бачення, а саме – класифікацію зображень. Моделювання конфронтаційної атаки на класифікатор буде виконуватись у тестовий час, за принципом «білого ящика», тобто з повним доступом до моделі та її результатів, що являється найгіршим випадком.

Ця задача є однією з найбільш класичних та добре досліджених. Існує велика кількість відкритих датасетів та моделей для її вирішення.

Основними перевагами з боку AML є те, що на прикладі зображень

легко пояснювати основні принципи роботи конфронтаційних атак, збурення можливо продемонструвати наглядним та зрозумілим способом. Викривлення у роботі моделі при класифікації також досить явні.

Тож, класифікація зображень має велику кількість переваг для демонстративних та дослідницьких цілей.

В якості моделі буде застосовано конволюційну нейронну мережу, так як це найбільш ефективний і широко використовуваний тип моделей для вирішення такої задачі.

## 1.6 Визначення критеріїв ефективності

Як наслідок аналізу методів, що будуть застосовуватись, та відповідної технічної літератури, було обрано такі критерії ефективності.

Для оцінки продуктивності класифікатора:

- точність класифікації на тестових даних;
- значення функції втрат.

При моделюванні змагальних атак буде застосовуватись рейтинг успіху.

Важливо, щоб нейронні мережі, що будуть застосовуватись у експериментах, мали достатню точність та здатність узагальнення даних, так як відомо про взаємозв’язок між продуктивністю моделі та її вразливістю до конфронтаційних прикладів.

З цієї ж причини необхідно запобігати перенавчання. Цей процес зазвичай контролюється за допомогою відстежування значення функції втрат на валідаційних даних під час навчання класифікатору. Для уникнення перенавчання також існують спеціальні техніки, такі як регуляризація або рання зупинка.

Для того, аби оцінювати процесс моделювання атаки, застосовують рейтинг успіху. Це величина обернена до точності класифікації, яка обраховується на конфронтаційних прикладах.

Рейтинг успіху приведено у формулі 1.1:

$$SR = \frac{N}{M} \quad , \qquad (1.1)$$

де N – кількість успішних спроб;

M – загальна кількість спроб.

Основною задачею є збереження або покращення попереднього рівня точності класифікації на тестових даних та зменшення рейтингу успіху при конфронтаційних атаках.

## 1.7 Постановка задачі

В даній роботі ставиться за мету розробка та дослідження адаптивних елементів нейронних мереж, в тому числі таких, що запропоновані нами, а також дослідження їх взаємозв'язку з конфронтаційною стійкістю нейронних мереж.

Отож, для досягнення поставленої мети має бути виконано дві основні задачі – розробка адаптивних елементів мереж та дослідження з моделюванням конфронтаційних атак для обраної інтелектуальної моделі – класифікатора.

Задачі містять як теоретичну, так і практичну частину, що стосується реалізації та виконання експериментів.

В рамках теоретичного дослідження необхідно виконати такі завдання:

– провести аналіз технічної літератури та новітніх технологій розробки;

– виявити наявні проблеми та сформулювати задачі;

– визначити та порівняти наявні адаптивні елементи;

– обрати мову, середовище та програмні інструменти для навчання

моделі;

– визначити архітектуру моделі та задачу, на прикладі якої буде виконуватись дослідження;

– обрати набір даних, що містить достатню кількість розмічених прикладів та відповідає задачі дослідження;

– описати математичну модель та властивості запропонованого прошарку GLP;

– сформулювати гіпотези щодо використання адаптивних елементів та їх взаємозв'язку захищеністю моделі.

В рамках практичної частини дослідження необхідно виконати такі завдання:

– реалізувати адаптивні елементи мереж за допомогою обраної мови та програмних інструментів;

– реалізувати архітектуру штучної нейронної мережі;

– виконати навчання нейронної мережі на обраному датасеті з використанням різноманітних адаптивних елементів;

– дослідити змагальну стійкість отриманих моделей за допомогою моделювання конфронтаційних атак;

– виконати аналіз та порівняння отриманих результатів;

– підтвердити або заперечити попередньо сформульовані гіпотези;

– на основі проведеного дослідження визначити оптимальну комбінацію методів та налаштувань.

## 1.8 Висновки

Під час вивчення літератури та останніх досліджень, огляду та аналізу існуючих методів було виявлено низку проблем та напрямів для дослідження.

З боку конфронтаційного машинного навчання, виявлено відсутність широкої практики застосування захисту під час навчання інтелектуальних

моделей. Більшість мереж, що використовуються зараз у задачах бізнесу та інших сферах діяльності, не є захищеними, а отже, призводять до вразливості систем, у яких застосовуються.

В той же час можна зробити висновок, що наявні методи захисту складні у застосуванні та недостатньо досліджені. Наразі мають досліджуватись додаткові способи і методи, що можуть бути застосовані для підвищення стійкості моделей до конфронтаційних атак.

На основі аналізу останніх досліджень, було висунуто гіпотезу, що конфронтаційна стійкість моделей може бути покращена за рахунок змін у архітектурі мережі. А саме, передбачається, що застосування гнучких прошарків у нейронних мережах може покращити рівень захисту моделей від конфронтаційних атак. Для перевірки цієї гіпотези було вирішено обрати прошарки субдискретизації у якості адаптивних елементів мережі та конволюційний класифікатор зображень як тестову архітектуру.

В рамках цієї гіпотези було проведено огляд та аналіз останніх досліджень та технічної літератури. Можемо зробити висновок, що наразі ведуться активні розробки адаптивних субдискретизацій з застосуванням різних підходів. Крім того, існують дослідження, які підтверджують, що заміна деяких елементів мережі може вплинути на рівень її нативної захищеності.

Відповідно до наведених фактів, дослідження впливу використання адаптивних елементів в мережі на конфронтаційну стійкість є актуальним та перспективним напрямом.

Також в роботі запропоновано Generalized Lehmer Pooling як більш досконалий варіант адаптивного пулінгу, його ефективність має бути досліджена.

# 2 ТЕОРЕТИЧНЕ ДОСЛІДЖЕННЯ

## 2.1 Змагальні приклади

Нехай класифікатор буде позначено h, а вхідні дані та відповідну мітку класу (x, y). Тоді процес класифікації описується формулою 2.1:

$$h(x) = y. \tag{2.1}$$

Конфронтаційний приклад позначимо x`. Для його створення необхідно додати до початкового зображення деяке збурення δ, що описується формулою 2.2:

$$x' = x + \delta. \tag{2.2}$$

Розмір змагальних збурень зазвичай обмежується однією з $L_p$-norm (рис. 2.1) зверху.

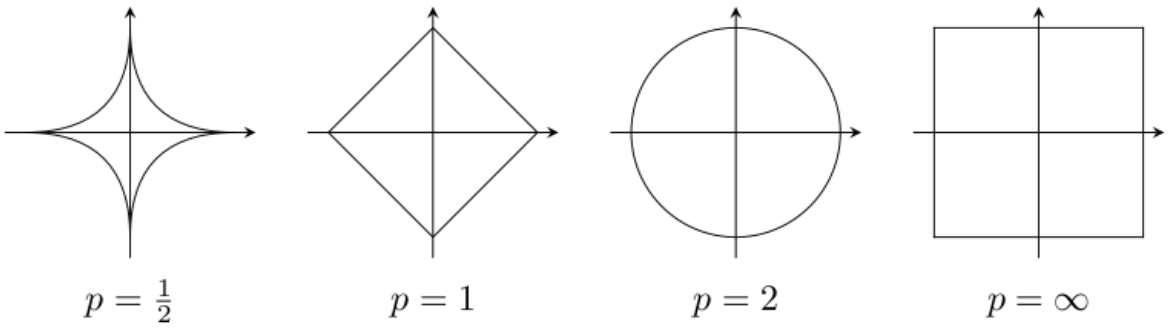


Рисунок 2.1 – Види $L_p$-норм

До того ж, доки δ буде меншим деякого малого граничного значення ε, при класифікації x та x` буде присвоєно однаковий клас. [25] В іншому випадку, x` вдасться обманути модель і такий приклад буде змагальним.

Можлива причина цього феномену полягає у застосуванні матричного множення в процесі навчання мережі. Вектори ваг прошарків w перемножуються зі зміненим вхідним сигналом x`, формула 2.3:

$$w^T x' = w^T x + w^T. \tag{2.3}$$

Внесення конфронтаційних збурень призводить до того, що значення активаційної функції зростає. Через це невеликі зміни вхідних даних можуть призвести до значної зміни результату. При цьому різниця у виходах згорткової частини нейронної мережі (вектори ознак) досить сильна і може бути більшою, ніж різниця між двома природними різними зображеннями, що належать до одного і того ж, а іноді і різних класів.

Наприклад, на рисунку 2.2 зображено візуалізовані вектори ознак 500 зображень та їх відповідник змагальних прикладів для набору зображень двох класів. Зелені точки позначають клас 1, а сині клас 2. Змагальні приклади позначені помаранчевими та червоними точками.

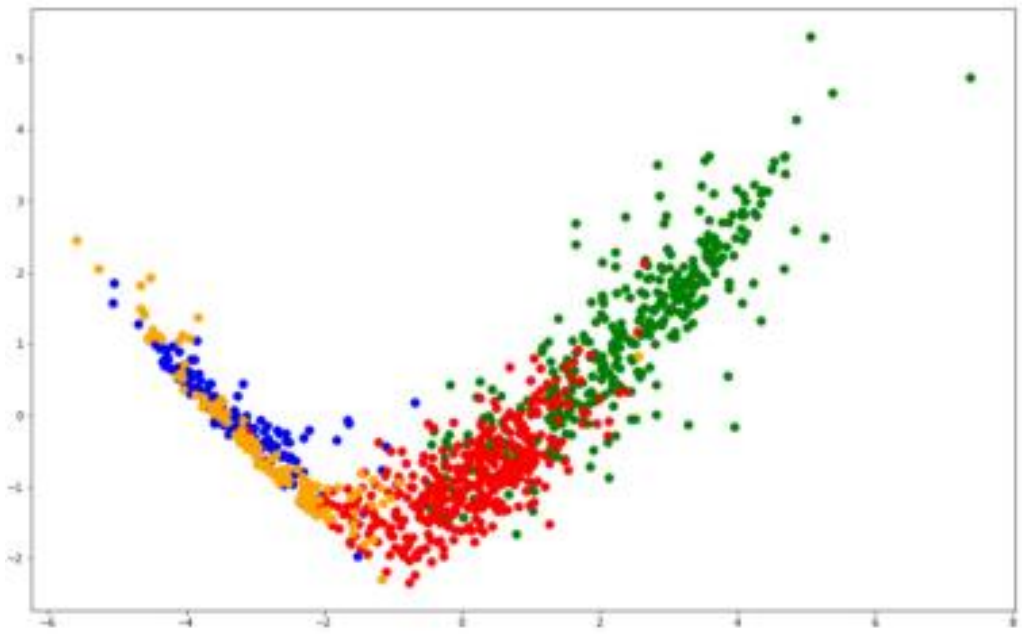

Рисунок 2.2 – Приклад розподілу векторів ознак природних та змагальних даних

Як бачимо, відстань між екземплярами змагальних прикладів може бути більшою, ніж між природними даними різних класів.

Крім цього можна помітити з формули 2.3, що існує зв'язок між конфронтаційним шумом та ваговими коефіцієнтами мережі. Так як різні інтелектуальні моделі можуть навчатись подібним вагам для апроксимації функції, це призводить до виникнення ще однієї властивості конфронтаційних прикладів – переносимості (transferability) [26].

Переносимість це явище, завдяки якому конфронтаційні приклади, створені для однієї мережі, також неправильно розпізнаються іншою моделлю. До того ж такі мережі можуть мати різні архітектури та навчатись на непов’язаних підмножинах набору даних [27].

## 2.3 Змагальні атаки

Процес створення змагальних прикладів називають змагальною атакою. Проаналізувавши формулу (2.2), можемо зробити висновок, що метою змагальної атаки є підбір такого δ, при додаванні якого рішення класифікатору буде змінено. Цей процес продемонстровано на рисунку 2.3 [28].

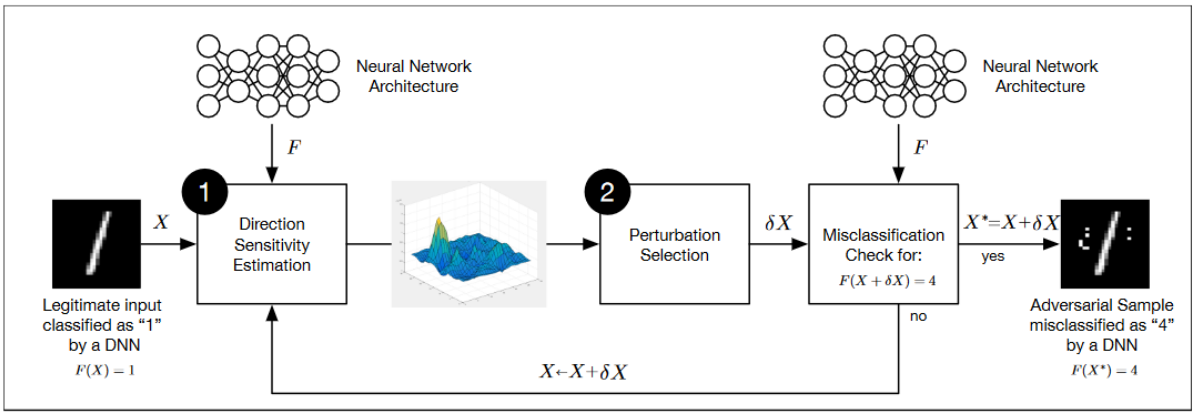


Рисунок 2.3 – Створення конфронтаційних прикладів

Незважаючи на те, що, як згадувалося раніше, існує досить багато алгоритмів генерації і їх евристичних поліпшень, багато з них ми можемо узагальнити і виділити деякий алгоритм, щоб побачити всю картину і полегшити їх розуміння.

В результаті вивчення літератури було встановлено, що основна послідовність існуючих алгоритмів генерації змагальних прикладів представлена наступними кроками:

– візьмемо деяке зображення з предметної області атакованої нейронної мережі;

– виконаємо пошук та згенеруємо змагальне збурення;

– додамо згенероване збурення до вихідного зображення.

Результатом буде змагальний приклад.

Наведена вище схема є основою для ключових алгоритмів генерації змагальних прикладів.

Як бачите, ці алгоритми можуть відрізнятися на першому та другому кроці і кроці. Різниця в першому кроці може лежати в джерелі оригінального зображення. Це може бути навчальний зразок, тестовий зразок, а також довільне зображення, передбачене мережею.

Також розглянемо процес конфронтаційної атаки з іншої точки зору.

Нехай існує деяка площина функції, що оптимізується, а класи на ній будуть представлені деякими зонами. В такому випадку мета змагальної атаки не лише змінити початкові дані таким чином, щоб він потрапив до іншої зони, а й визначити найбільш короткий напрям. Тобто, віднайти adversarial direction на площині функції, що апроксимується моделлю.

Наприклад, на рисунку 2.4 наведено проекцію площини функції, що оптимізується, у вигляді 2D простору. На ній представлено 3 класи об'єктів – Dog, Truck та Airplane. Міткою X позначено положення початкових даних – зображення собаки. Для цього зображення змагальним напрямом буде червона лінія, а рух у випадковому напрямі призведе до значних, помітних оку збурень.

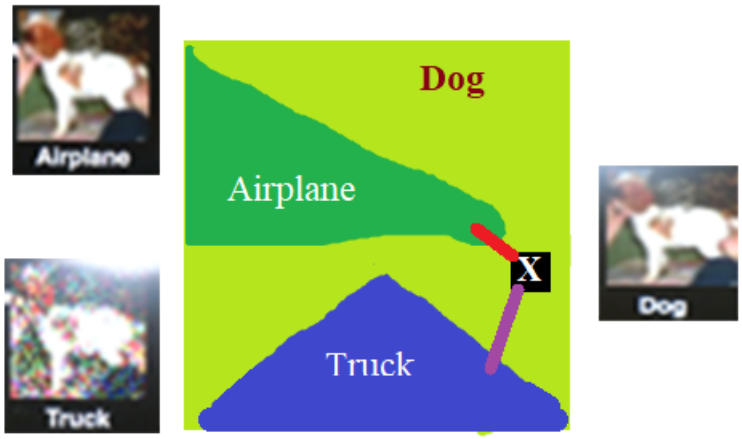


Рисунок 2.4 – Площина функції класифікації (червона лінія – adversarial direction, фіолетова лінія – random direction)

Розглянемо класифікацію існуючих загроз (рис. 2.5)

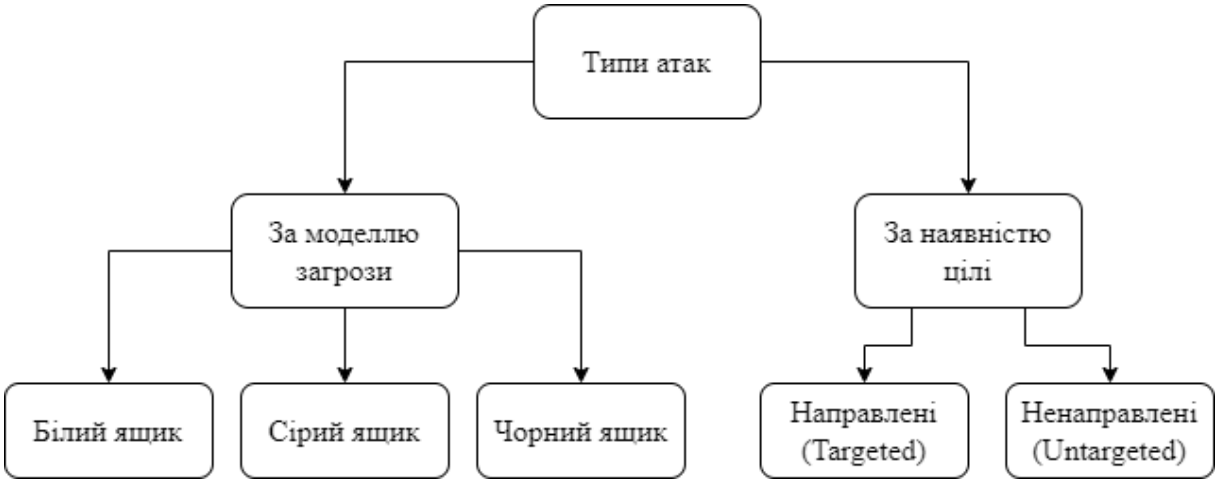


Рисунок 2.5 – Класифікація типів атак

За моделлю загрози виділяють атаки на білий та чорний ящик. Деякі дослідники також розглядають частково-білий або сірий ящик.

При моделі загрози «білий ящик» у зловмисника є можливість використання моделі, доступ до передбачень, знання її архітектури та параметрів [29]. Тому модель та її передбачення безпосередньо використовуються для пошуку в просторі змагальних прикладів.

Найчастіше використовується деякий процес оптимізації, але існують різноманітні способи атакування.

При атаці на чорний ящик атакуючий може лише здійснювати запити до мережі, тобто для входу x відома мітка класу y, присвоєна класифікатором [30]. У такому випадкову можливі варіанти коли відповіддю на запит є одна або декілька міток та коли відомі також ймовірності, що x належить тому чи іншому класу.

Для проведення атаки типу «чорний ящик» необхідно створити аналог цільової моделі, орієнтуючись на значення передбачень моделі. Таким чином задача зводиться до описаної атаки на білий ящик.

За наявністю цільового класу виділяють ненаправлені та направлені атаки. Ненаправлені атаки намагаються віднайти таки δ, щоб результат класифікації було змінено на y` ≠ y, де y` – клас присвоєний змагальному прикладу x`. Направлені атаки додатково враховують заданий клас c, змінюючи результат так, щоб y` = c.

При дослідженнях в цій роботі застосовувались атаки на білий ящик, за принципом найгіршого випадку. Також задавати певний клас немає необхідності, тому атаки ненаправлені. З усіх атак цього типу для проведення моделювання було обрана атаку PGD.

Розглянемо детальніше Projective Gradient Descent (PGD) [31]. Представимо підбір конфронтаційного шуму як оптимізаційну проблему. В такому випадку найбільш підходящий змагальний приклад буде розміщено у сідловій точці.

PGD мінімізує втрати на конфронтаційних даних за допомогою градієнтного спуску, локальний мінімум відповідає згенерованому змагальному прикладу.

Функція втрат при цьому позначається як L(θ, x, y), де D розподіл даних, а θ параметри моделі. Формула 2.4 описує функцію, що має бути мінімізована.

$$\rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y)\right], \quad (2.4)$$

де E[L] – мінімізація емпіричного ризику.

Отож, PGD відноситься до класу ітеративних атак, що базуються на градієнтній оптимізації. Отримані в її результаті дані називають універсальними конфронтаційними прикладами першого порядку.

## 2.3 Огляд адаптивних елементів

Базовими елементами конволюційних мереж для класифікації зображень є прошарки конволюції, субдискретизації, повнозв'язні прошарки у хвості мережі та активаційні функції.

В даній роботі було вирішено застосовувати в якості адаптивних елементів прошарки субдискретизації. Традиційно операція субдискретизації є фіксованою та не навчається [32], що дозволить виконувати більш справедливе порівняння через зменьшення кількості факторів, що можуть вплинути на результати експериментів.

Принцип роботи традиційного пулінгу зображено на рисунку 2.6. Початковий набір даних прдставлено у вигляді деякого тензору X. Для виконання субдискретизації застосовується лпаваюче вікно, розмір якого може бути задано.

Дві найбільш відомі функції, що застосовуються для субдискретизації, це average pooling та max pooling, які підсумовують середню та найактивнішу присутність функції в кожному наборі даних відповідно.

У випадку MaxPooling, для кожного такого плаваючого вікна обирається максимальний елемент. У випадку AvgPooling – середнє значення елементів, що потрапили у вікно. Відповідно, виконання цих операцій призводить до зменшення розмірності даних.

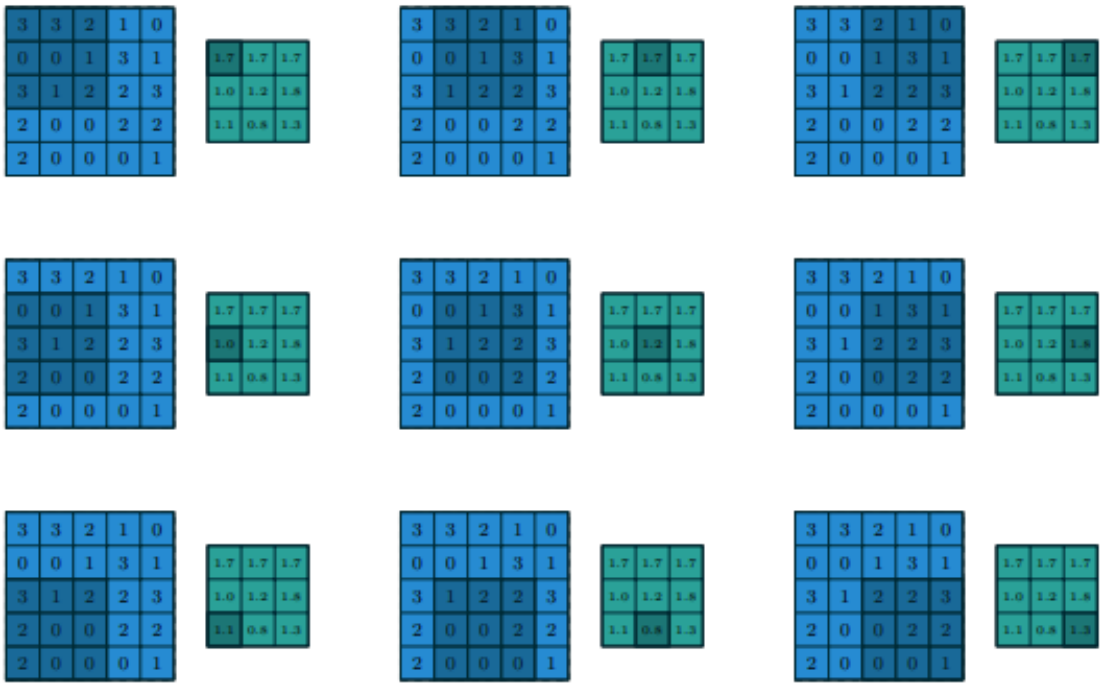

Рисунок 2.6 – Операція субдискретизації

Такі негнучкі операції не можуть гарантувати мінімізацію помилки навчання та містять низку інших недоліків.

Наприклад, AvgPooling призводить до розмивання об'єктів на зображеннях, що майже унеможливлює його використання, MaxPooling же створює певні патерни, які називають «шахова дошка», що не містились в початкових даних. Такі викривлення даних в процесі зменшення розмірності здатні призвести до погіршення роботи інтелектуальних моделей.

Деякі відомі приклади також наведено у публікації Yu D. et al., рисунок 2.7. В них говориться про особливості роботи традиційних пулінгів у граничних випадках. Наприклад, через те, що чорна лінія на білому фоні достатньо мала, MaxPooling повністю ігнорує її присутність. AvgPooling при цьому зберігає наявність лінії, проте вона стає ледве помітною через розмиття.

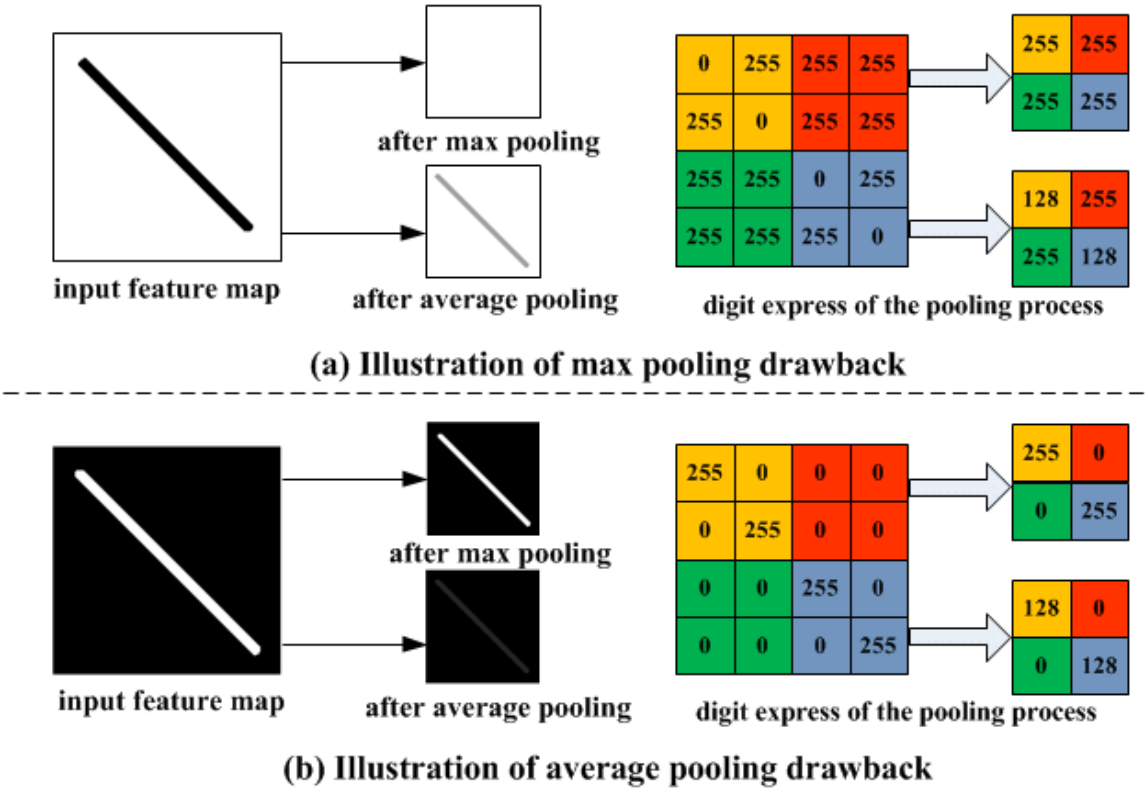


Рисунок 2.7 – Недоліки Max та AvgPooling

Щоб подолати ці недоліки, Yu D. et al. пропонували застосування змішаного пулінгу між максимальним та середнім, що базується на принципах регуляризації [33]. Тобто, для отримання значень у межах інтервалу між середнім і максимальним пулінгом, AvgPooling домножується на вагу λ, що може вивчатись, а відповідну вага 1- λ застосовується до MaxPooling (рис. 2.8).

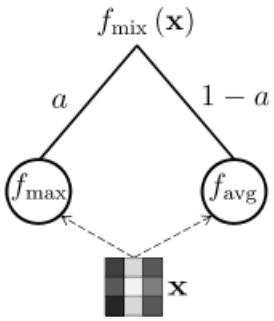


Рисунок 2.8 – Схема змішаного пулінгу

Результат субдискретизації обчислюється як середньозважене для обох (формула 2.5).

$$y_{kij} = \lambda \cdot max_{(p,q)\in R_{i,j}} + (1-\lambda) \cdot \frac{1}{|R_{i,j}|} \sum_{(p,q)\in R_{i,j}} x_{kpq} \ , \qquad (2.5)$$

де x, y – пара вхідних даних та результату класифікації;

R – множина реальних чисел;

λ – вільний коєфіцієнт.

Пізніше Sun et al. пропонують операцію субдискретизації, що навчається [34], яка називається LEAP (LEArning Pooling). Відповідно до запропонованої ними архітектури, одна спільна лінійна комбінація нейронів у регіоні вивчається для кожної карти ознак, що робить AvgPooling лише одним окремим випадком LEAP (коли ваги для нейронів рівні).

Така операція буде працювати як своєрідне зважене середнє значення, чутливе до місцевості, з ваговими коефіцієнтами навчання, що робить її схожою на фільтри об'єднання навчання або об'єднання дерев (рис. 2.9), як запропоновано Lee et al. (2017) [35].

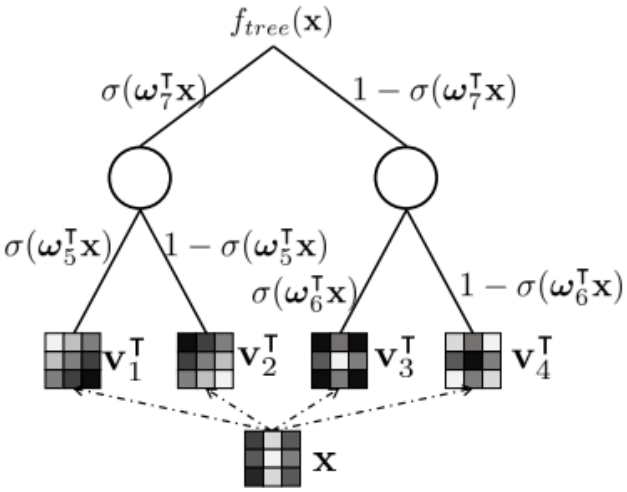


Рисунок 2.9 – Схема Tree Pooling

Більш простою та популярною модифікацією є power mean або LPPool [36], що описується формулою 2.6.

$$f(X) = \sqrt[p]{\sum_{x \in X} x^p} \quad , \tag{2.6}$$

де x – вхідні дані;

p – гіперпараметр.

Ця функція включає MaxPooling при p = ∞ та при p = 1 є аналогом SumPooling.

З проведеного порівняння можна зробити висновок, що наразі ведуться розробки адаптивних методів субдискретизації. Розглянуті методи дозволяють за допомогою однієї функції та використання гіперпараметрів, що навчаються, охопити більш широке сімейство функцій, що водночас включають в себе найбільш популярні – average та max pooling.

Крім того, в даній роботі досліджується використання запропонованого нами Generalized Lehmer Pooling, який побудовано на базі параметризованого General Lehmer Mean. Ця розробка потенційно більш ефективна за рахунок особливих властивостей функції лемеровського середнього.

## 2.4 Generalized Lehmer Mean

Lehmer Mean (формула 2.7) це нелінійна монотонна функція середнього з унікальними властивостями, що була досліджена Beliakov & Špirková (2016) [37] та Terziyan (2017)[38].

$$LM_k(x_1, x_2, \dots, x_n) = \frac{x_1^{k+1} + x_2^{k+1} + \cdots + x_n^{k+1}}{x_1^k + x_2^k + \cdots + x_n^k} = \frac{\sum_{i=1}^n {x_i}^{k+1}}{\sum_{i=1}^n {x_i}^k} \ . \tag{2.7}$$

Вигляд функції Lehmer mean в залежності від p та в порівнянні з Holder mean наведено на рисунку 2.10. Зазначимо з наведеного графіку, що значення *a* мають бути строго позитивними.

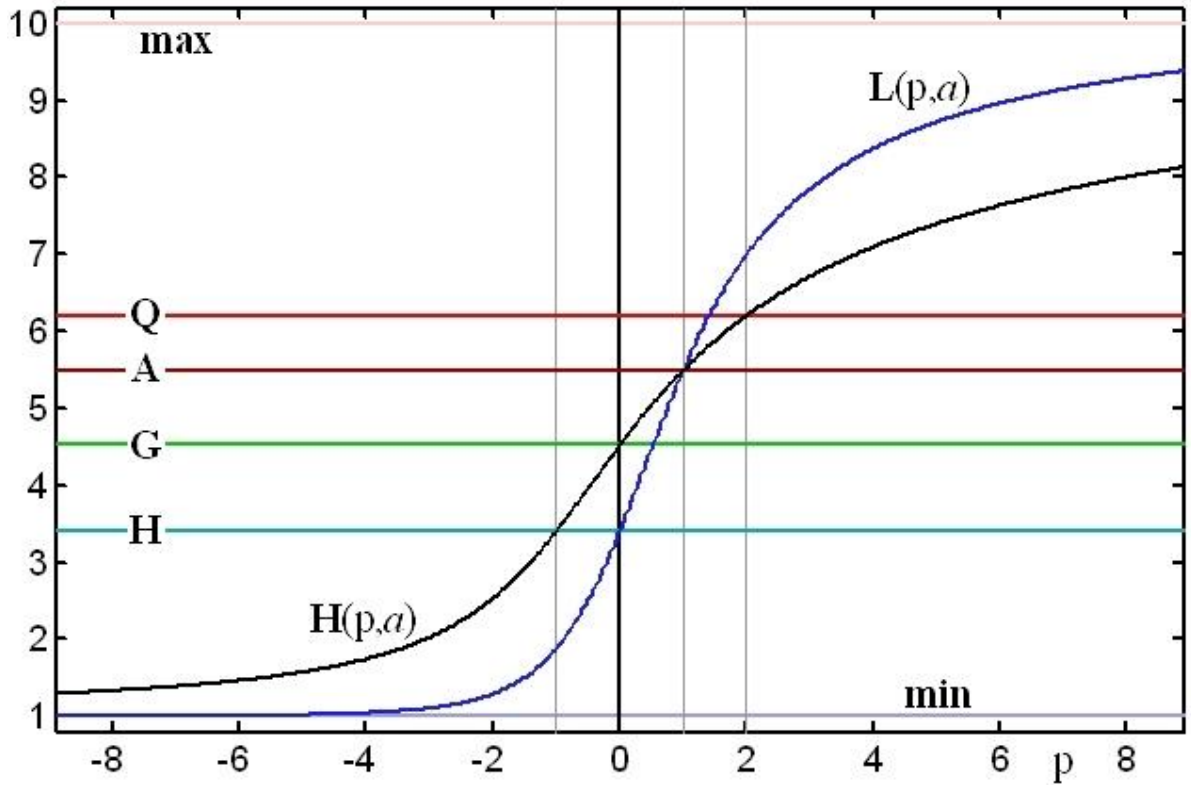


Рисунок 2.10 – Графік Lehmer mean (L) та Holder mean (H)

Як відомо, арифметичне середнє значення набору реальних чисел X розраховується за формулою 2.8.

$$AM(x_1, x_2, \dots, x_n) = \frac{x_1+x_2+\cdots+x_n}{n} = \frac{\sum_{i=1}^{n} x_i}{n}. \qquad (2.8)$$

Також наведемо формулу середнього Колмогорова (формула 2.9).

$$KM(x_1, x_2, \dots, x_n) = \varphi^{-1}\{AM[\varphi(x_1), \varphi(x_2), \dots, \varphi(x_n)]\} = \varphi^{-1}\left[\frac{\sum_{i=1}^{n} \varphi(x_i)}{n}\right]. \qquad (2.9)$$

Середнє Колмогорова є узагальненням арифметичного середнього, де φ може позначати будь-яку ін'єктивну, неперервну та строго монотонну функцію. Виходячи з цього, можна сказати, що середнє Колмогорова це деякий набір функцій середнього, вид яких залежить від вибору φ.

Застосуємо в якості φ експоненту ($\varphi = \alpha^x$), відповідно функція обернена до неї буде логаритмом ($\varphi^{-1} = \log_\alpha x$).

Далі у випадку заміни у формулі 2.7 AM на Lehmer Mean, буде утворено нову функцію середнього, Generalized Lehmer Mean, що описується формулою 2.10.

Як було сказано, стандартний Lehmer Mean вимагає строго позитивні значення x. Тоді як Generalized Lehmer Mean завдяки застосуванню експонент може працювати з будь якими числовими значеннями.

$$\forall\alpha(\alpha \in \mathbb{R};\ \alpha > 1), \forall\beta(\beta \in \mathbb{R}), \forall n(n \in \mathbb{N}^+), \forall x_i(x_i \in \mathbb{R}),$$

$$GLM_\beta^\alpha(x_1, x_2, \dots, x_n) = log_\alpha\left\{LM_\beta[\alpha^{x_1}, \alpha^{x_2}, \dots, \alpha^{x_n}]\right\} =$$

$$log_\alpha\left[\frac{\alpha^{(\beta+1)\cdot x_1}+\alpha^{(\beta+1)\cdot x_2}+\dots+\alpha^{(\beta+1)\cdot x_n}}{\alpha^{\beta\cdot x_1}+\alpha^{\beta\cdot x_2}+\dots+\alpha^{\beta\cdot x_n}}\right] = \frac{ln\left[\frac{\alpha^{(\beta+1)\cdot x_1}+\alpha^{(\beta+1)\cdot x_2}+\dots+\alpha^{(\beta+1)\cdot x_n}}{\alpha^{\beta\cdot x_1}+\alpha^{\beta\cdot x_2}+\dots+\alpha^{\beta\cdot x_n}}\right]}{ln\,\alpha} = \quad (2.10)$$

$$\frac{ln\left[\frac{\sum_{i=1}^n \alpha^{(\beta+1)\cdot x_i}}{\sum_{i=1}^n \alpha^{\beta\cdot x_i}}\right]}{ln\,\alpha}$$

Вид функції Generalized Lehmer Mean контролюється двома параметрами: альфа та бета. Згідно до формули, параметр альфа має бути більшим одиниці. Крім цього строгого обмеження, було виявлено, що оптимальними проміжками для параметру альфа є значення від 1 до 2.7, а для бета від -2.5 до 1.5. Ці значення є достатніми для прояву всіх властивостей формули та не занадто великі, що спрощує роботу з ними.

Приклад поведінки функції при різних значеннях альфа та бета для заданого X наведено на рисунку 2.11.

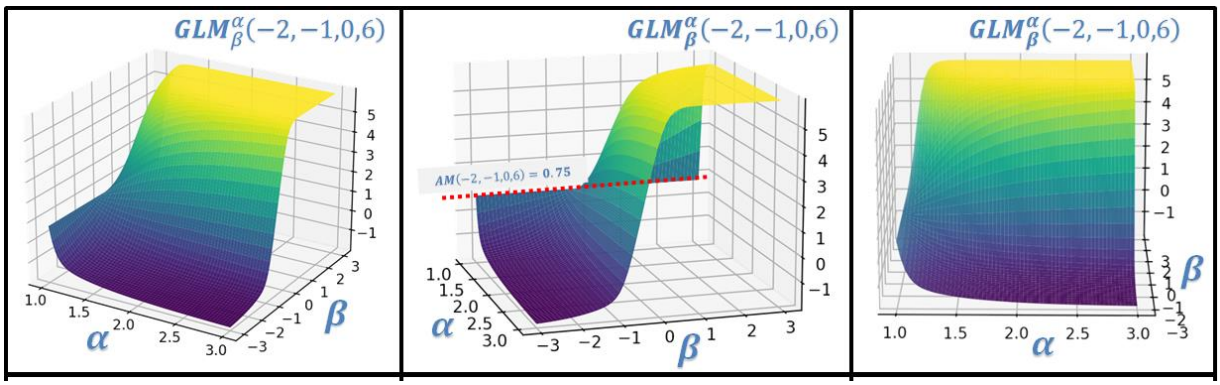


Рисунок 2.11 – Залежність виду функції від параметрів

Як бачимо, при альфа близькій до одиниці, GLM поводить себе як арифметичне середнє.Крім того, ця функція має ще декілька цікавих властивостей.

Значення GLM знаходиться у проміжку між мінімум та максимум (формула 2.11).

$$\text{MIN}(x_1, x_2, \dots, x_n) \leq GLM^{\alpha}_{\beta}(x_1, x_2, \dots, x_n) \leq \text{MAX}(x_1, x_2, \dots, x_n)\,. \qquad (2.11)$$

При цьому при значенні β, що прагне до нескінченності, значення прагне максимума, а при значенні β, що прагне до від'ємної нескінченності, значення функції прагне мінімума (формула 2.12).

$$\begin{aligned} &\text{(a) } \forall\alpha(\alpha \in \mathbb{R};\ \alpha > 1), GLM^{\alpha}_{\beta\to\infty}(x_1, x_2, \dots, x_n) \to \text{MAX}(x_1, x_2, \dots, x_n);\\ &\text{(b) } \forall\alpha(\alpha \in \mathbb{R};\ \alpha > 1), GLM^{\alpha}_{\beta\to-\infty}(x_1, x_2, \dots, x_n) \to \text{MIN}(x_1, x_2, \dots, x_n) \end{aligned} \qquad (2.12)$$

Схожі властивості спостерігаються для α (формула 2.13). Коли альфа прагне до нескінченності, при значенні $\beta^*$ функція поводить себе як арифметичне середнє, при $\beta > \beta^*$ подібна до MaxPooling, а при $\beta < \beta^*$ аналогічна до MinPooling.

$$
\begin{aligned}
&\text{(a) } \exists!\,\beta^*(\beta^* \in \mathbb{R}; -1 \le \beta^* \le 0), GLM_{\beta^*}^{\alpha\to\infty}(x_1, x_2, \dots, x_n) \to AM(x_1, x_2, \dots, x_n);\\
&\text{(b) } \forall\beta(\beta \in \mathbb{R}; \beta > \beta^*), GLM_{\beta>\beta^*}^{\alpha\to\infty}(x_1, x_2, \dots, x_n) \to \mathrm{MAX}(x_1, x_2, \dots, x_n);\\
&\text{(c) } \forall\beta(\beta \in \mathbb{R}; \beta < \beta^*), GLM_{\beta<\beta^*}^{\alpha\to\infty}(x_1, x_2, \dots, x_n) \to \mathrm{MIN}(x_1, x_2, \dots, x_n);\\
&\text{(d) } \forall\alpha[(n = 2) \to (\beta^* = -0.5)].
\end{aligned}
\tag{2.13}
$$

## 2.4 Generalized Lehmer Pooling

У даній роботі пропонується використання Generalized Lehmer Mean у прошарку субдискретизації [39] замість традиційних фіксованих пулінгів. Такий гнучкий пулінг завдяки вільним параметрам зможе краще підлаштовуватись під конкретний датасет і потенційно буде більш стійким до конфронтаційних змагальних прикладів.

Розглянемо, як може змінюватись математична функція субдискретизації (рис. 2.12 та 2.13) на конкретному прикладі ($x_1 = 2, x_2 = 5, x_3 = 5, x_4 = 8$).

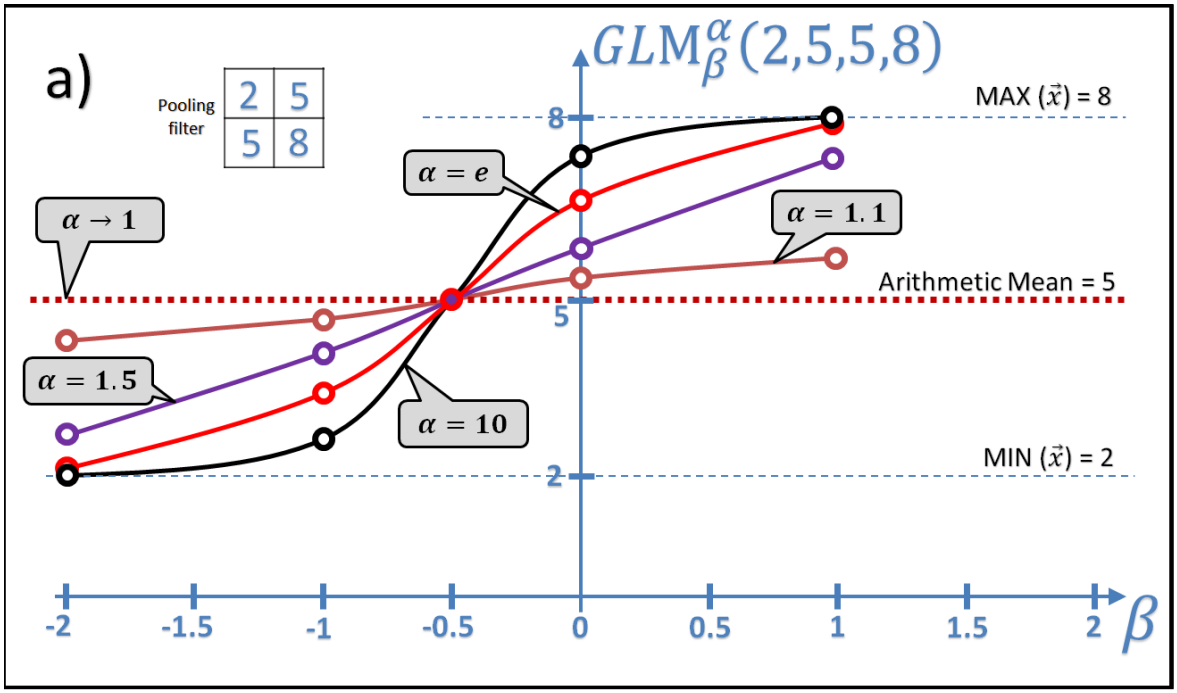


Рисунок 2.12 – Вид функції GLM в залежності від параметру бета

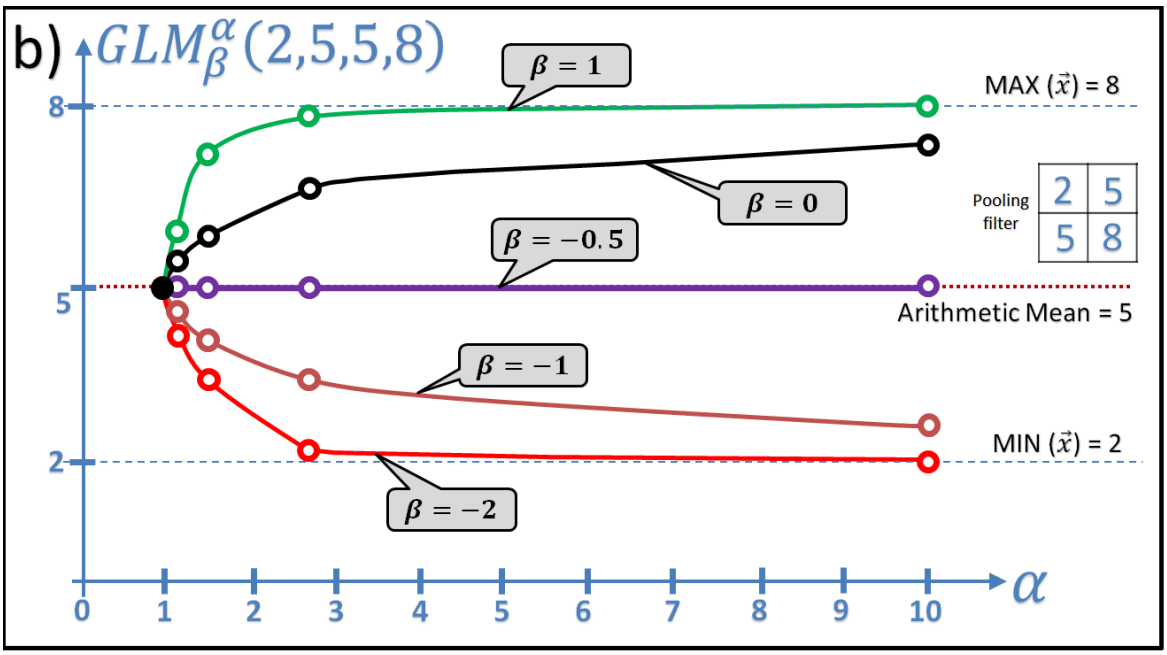


Рисунок 2.13 – Вид функції GLM в залежності від параметру альфа

На рисунку 2.12 можна побачити, що зміна $\beta$ від від'ємних до позитивних значень призводить до зміни значення GLP від Min pooling до Max pooling через Average Pooling. Швидкість цієї зміни контролюється параметром $\alpha$ (тобто, при $\alpha = 1$ немає ніяких змін і якщо $\alpha$ зростає відповідна швидкість зростає).

На рисунку 2.13 можна побачити, що зростання $\alpha$ може призвести до:

– функція залишаться незмінною при $\beta = -0.5$;

– функція зросте від AvgPooling до MaxPooling, при $\beta > -0.5$;

– функція зменшиться від AvgPooling до MinPooling, коли $\beta < -0.5$ (чим менше $\beta$ , тим швидше зменшиться).

Крім того, GLP потенційно може подолати деякі відомі недоліки AvgPooling та MaxPooling (рис. 2.14). Як бачимо, у обох граничних випадках GLM може діяти більш  ефективно, крім того, він може підлаштуватись під дані.

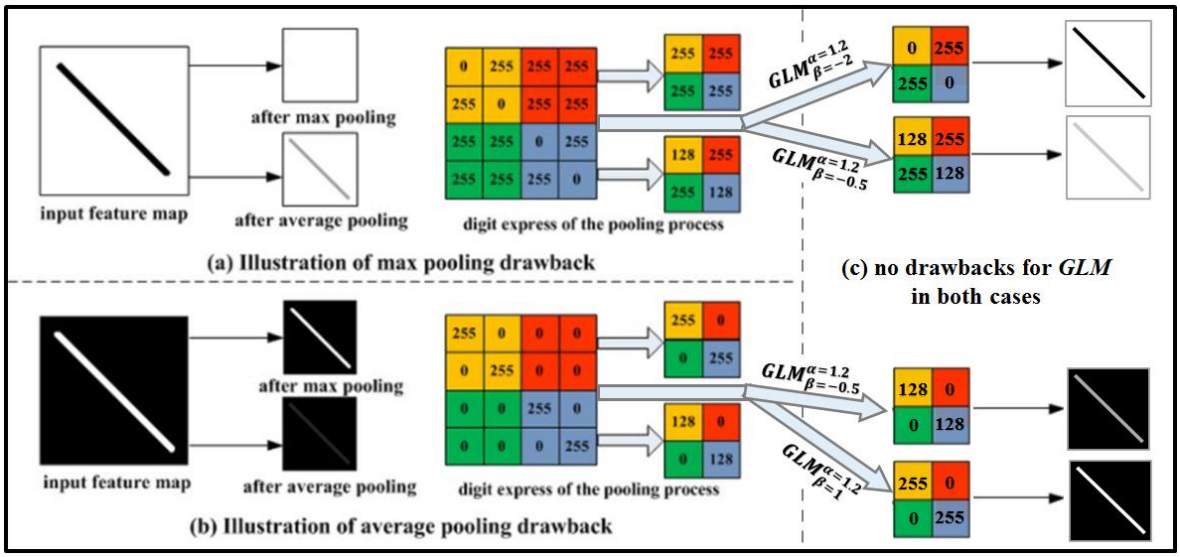


Рисунок 2.14 – Приклад подолання недоліків субдискретизацій

Також GLP матиме значну перевагу для нових датасетів, коли оптимальна архітектура моделі ще невідома. В той же час, на більш стандартних датасетах перевага може бути незначною.

## 2.5 Висновки

У даному розділі виконано опис математичної моделі вразливості моделей до конфронтаційних прикладів та конфронтаційних атак. Окрім того, описано математичну модель запропонованого прошарку субдискретизації Generalized Lehmer Pooling.

Наведена математична модель та аналіз формул дозволяють зрозуміти, що пропонований прошарок Generalized Lehmer Pooling має однозначні переваги.

Завдяки параметрам альфа та бета, математична функція може змінювати свій вигляд, отож, вона включає як окремі випадки велику кількість інших відомих функцій, таких як Max Pooling та Average Pooling. Це дозволяє припустити, що за її використання ми отримаємо результати не гірше або краще, ніж з застосуванням інших субдискретизацій.

Також, проаналізувавши попередні дані, можемо зробити такі припущення:

– метрики навчання моделі з застосуванням запропонованих функцій будуть не гіршими або кращими;

– застосування адаптивних елементів у нейронній мережі призведе до покращення конфронтаційної стійкості;

– існує взаємозв'язок між ступенем адаптивності мережі та її конфронтаційною стійкістю;

– запропонований GLP є більш ефективним захистом, ніж інші адаптивні елементи.

Наведені гіпотези мають бути перевірені експериментально.

# 3 ПРАКТИЧНА РЕАЛІЗАЦІЯ

## 3.1 План експериментів

Для перевірки гіпотез, сформульованих в результаті теоретичного дослідження, необхідно програмно реалізувати описані елементи та провести серію експериментів.

Отож, нас цікавлять відповіді на такі питання:

– чи призводить застосування гнучких елементів у мережі до збільшення конфронтаційної стійкості?

– якщо так, то який зв'язок між ступінню адаптивності мережі та її захищеністю?

– чи виконує запропонований Generalized Lehmer Pooling цю функцію краще, аніж інші гнучкі пулінги?

Звичайно, абсолютно однозначні відповіді на ці питання знайти, напевно, неможливо: все буде сильно залежати від набору даних, від параметрів мережі і т.д. Більш того, так як ефективність нейронної мережі теоретично не до кінця виправдана (звичайно, доведено, що алгоритми мережевого навчання якимось чином додадуть такі значення у масштабі, щоб мережа працювала добре, але теоретично не доведено, що мережа може працювати відмінно), теоретично виправдати ефект змагальних атак поки не вдається. Тому логічно діяти індуктивно: розглядати окремі випадки, знаходити закономірності і, можливо, підтверджувати їх теоретично.

Отож, тепер визначимось, які дослідження будуть проводитись.

Для підтвердження припущення, що Generalized Lehmer Pooling не погіршує метрики мережі, було вирішено розробити та навчити декілька версій класифікатору зображень.

Версії матимуть однакову архітектуру, проте одна з них буде використовувати Generalized Lehmer Pooling, а інша звичайний Max Pooling. Отримані моделі будуть застосовуватись у подальших експериментах, тож

доцільно в одній з версій моделі також застосувати інший адаптивний пулінг для порівняння.

Для підтвердження гіпотези, що адаптивні елементи покращують рівень конфронтаційної стійкості, слід змоделювати змагальну атаку на попередньо навчені мережі, порівняти рівень успіху атак на архітектурах з використанням адаптивних субдискретизацій та Max Pooling.

Далі необхідно дослідити взаємозв'язок між широтою використання адаптивних елементів в мережі та її стійкістю до змагальних атак. Дослідження виконуватиметься з декількох напрямів:

– вплив кількості елементів у мережі на захищеність;

– вплив ступеню свободи субдискретизацій на захищеність.

Для цього буде навчено декілька варіантів архітектур класифікатору зображень з різною кількістю адаптивних елементів та змодельовано конфронтаційні атаки на ці моделі, що дасть змогу оцінити чи є зв'язок між адаптивністю та захищеністю мережі.

Також застосування інших субдискретизацій з вільними параметрами, окрім запропонованого Generalized Lehmer Pooling, дасть змогу оцінити, чи впливає ступінь свободи адаптивного елемента на конфронтаційну стійкість.

Проаналізувавши результати експериментів, можна буде також зробити висновок, чи є запропонована GLP більш ефективною для вирішення цієї задачі.

## 3.2 Вибір методів та засобів реалізації

Отож, для проведення описаних експериментів, необхідно вибрати засоби для реалізації, навчання, датасет та архітектуру моделі.

В якості мови програмування в роботі застосовується python3. Python3 є однією з найбільш популярних мов для задач глибокого навчання [40], тому містить велику кількість фреймворків та бібліотек, що полегшують

роботу у цьому напрямі. Значною перевагою також є швидкість розробки та прототипування з застосуванням мови python. Також мова існує доволі велику кількість часу, проте не припиняє свого розвитку, та активно підтримується та оновлюється. Це, крім того, говорить про те, що існує велика спільнота розробників та дослідників, що користуються мовою python.

Для реалізації необхідних елементів використовується фреймворк PyTorch [41]. Це один з найбільш великих та популярних фреймворків для глибоко навчання, що має детальну та якісну документацію. За допомогою цього фреймворку створено багато проектів, моделей глибокого навчання та бібліотек, що можуть застосовуватись для досліджень.

Для логування метрик та результатів навчання використовується бібліотека tensorboard [42]. Це частина фреймворку Tensorflow, проте вона сумісна і з іншими найбільш популярними фреймворками глибокого навчання, зручна у роботі та має докладну документацію. За допомогою tensorboard можна логувати різні типи даних – скаляри, гістограми, текст, аудіо та зображення та інше. Після цього усі залоговані значення можуть бути переглянуті за допомогою зручного інтерактивного дашборду.

Також набір експериментів передбачає навчання класифікаторів зображень. В якості датасету для проведення досліджень було обрано CIFAR-10 [43]. Приклад даних наведено на рисунку 3.1. Датасет розрахований на мультикласову класифікацію, містить велику кількість даних, добре збалансований та очищений. Всі зображення кольорові та мають однаковий розмір.

Це один з класичних датасетів глибокого навчання, тож, його якість і відповідність задачі було багаторазово перевірено дослідниками. Крім того, велика кількість публікацій використовують його для перевірки своїх результатів, що надає можливість порівняння.

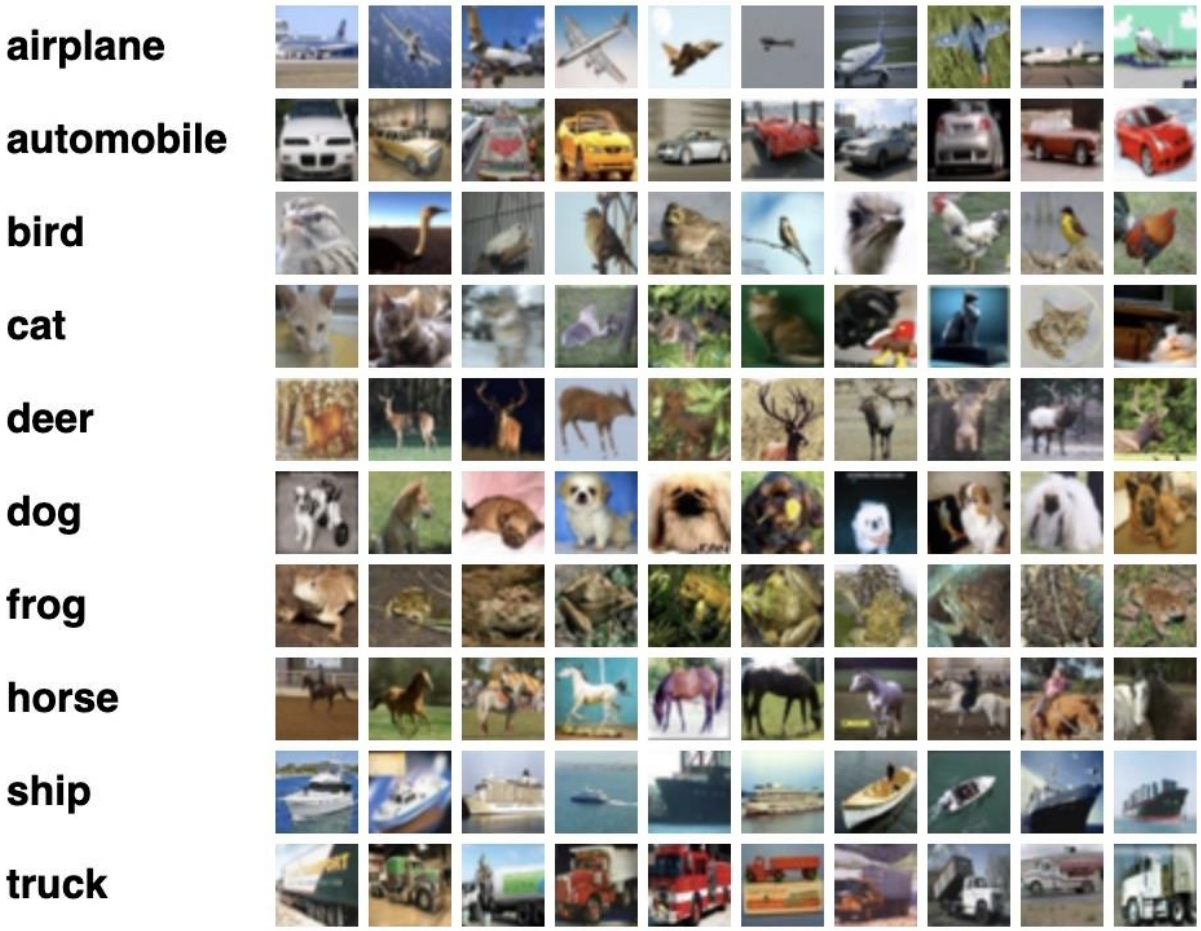


Рисунок 3.1 – Приклади зображень за класами у наборі даних CIFAR-10

Також експерименти передбачають порівняння декількох адаптивних елементів. Для цього були обрані MaxPooling, LPPooling та запропонований Generalized Lehmer Pooling.

## 3.3 Програмна реалізація

Для подальших експериментів необхідно здійснити програмну реалізацію обраних елементів.

Як відомо, набір даних CIFAR-10 містить кольорові (трьохканальні) зображення однакового розміру – 32x32. Усі зображення розбито на десять класів, кожен з яких включає 6000 тренувальних та 1000 тестових зображень. Перед поданням до класифікатора, дані було приведено до меж

[0, 1] та нормалізовано. В якості аугментації застосовувались випадкові горизонтальні відображення.

Для роботи з даними та моделлю було розроблено тренувальний цикл. Він складається зі стандартних частин пайплайну глибоко навчання – завантаження та обробка даних, ініціалізація оптимізатору, планувальнику та функції втрат, цикл за епохами, в якому виконується тренування та валідація моделі, та логування навчальних метрик моделі.

Для роботи з датасетом було розроблено архітектуру мережі, що наведена на рисунку 3.2.

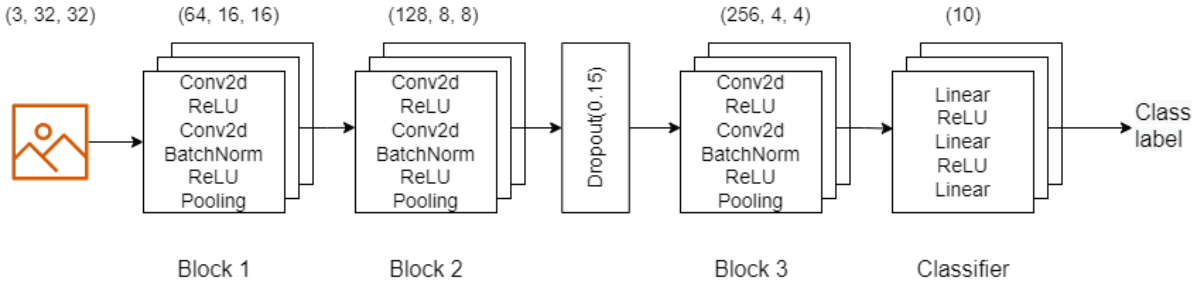


Рисунок 3.2 – Архітектура класифікатора зображень

Модель складається з трьох блоків, що містять прошарки згортки, батч нормалізації та субдискретизації, та четвертого блоку, який виконує роль класифікатора, та складається з лінійних прошарків. В якості активаційних функцій застосовуються ReLU. Також у нейронній мережі використовуються Dropout із ймовірністю 0.15, для запобігання перенавчання моделі.

Більш детальний опис отриманої моделі наведено у лістингу 3.1.

Класифікатор налічує майже 6 мільйонів параметрів та має вагу близько 27 МБ. Такий розмір моделі для глибокого навчання є досить малим, тож, ця мережа абсолютно невибаглива до ресурсів. При цьому має змогу достатньо добре узагальнювати дані.

Лістинг 3.1 – Детальний опис моделі класифікатора

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 32, 32, 32]             896
              ReLU-2          [-1, 32, 32, 32]               0
            Conv2d-3          [-1, 64, 32, 32]          18,496
       BatchNorm2d-4          [-1, 64, 32, 32]             128
              ReLU-5          [-1, 64, 32, 32]               0
         MaxPool2d-6          [-1, 64, 16, 16]               0
            Conv2d-7         [-1, 128, 16, 16]          73,856
              ReLU-8         [-1, 128, 16, 16]               0
            Conv2d-9         [-1, 128, 16, 16]         147,584
      BatchNorm2d-10         [-1, 128, 16, 16]             256
             ReLU-11         [-1, 128, 16, 16]               0
        MaxPool2d-12           [-1, 128, 8, 8]               0
       Dropout2d-13           [-1, 128, 8, 8]               0
           Conv2d-14           [-1, 256, 8, 8]         295,168
             ReLU-15           [-1, 256, 8, 8]               0
           Conv2d-16           [-1, 256, 8, 8]         590,080
      BatchNorm2d-17           [-1, 256, 8, 8]             512
             ReLU-18           [-1, 256, 8, 8]               0
        MaxPool2d-19           [-1, 256, 4, 4]               0
           Linear-20                 [-1, 1024]       4,195,328
             ReLU-21                 [-1, 1024]               0
           Linear-22                  [-1, 512]         524,800
             ReLU-23                  [-1, 512]               0
           Linear-24                   [-1, 10]           5,130
================================================================
Total params: 5,852,234
Trainable params: 5,852,234
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.01
Forward/backward pass size (MB): 4.18
Params size (MB): 22.32
Estimated Total Size (MB): 26.52
----------------------------------------------------------------
```

Модель було реалізовано за допомогою pytorch, код наведено у лістингу Б.1 додатку Б. Прошарок субдискретизації, що буде використовуватись, є параметром, що задається при створенні моделі. Це дозволяє більш зручно проводити серії експериментів, які потребують його заміни.

Далі було реалізовано прошарок Generalized Lehmer Pooling. Для коректної роботи та можливості застосування в будь якій моделі pytorch, маємо успадкувати клас від torch.nn.Module [43] та реалізувати метод forward. Код наведено у лістингу Б.2 додатку Б.

GLP має вільні параметри alpha та beta. Для того, щоб вони навчались

разом з іншими параметрами мережі за допомогою оптимізатора, їх було оголошено як torch.nn.Parameter. Метод forward викликає функцію generalized_lehmer_pooling, яка власне реалізує формулу запропонованої субдискретизації та обробляє роботу з ковзаючим вікном.

Через наслідування від класу Parameter, оптимізатори torch здатні самостійно включити альфа та бета до побудованого графу обчислень та провести операцію backward propagation за допомогою застосування autograd функцій.

Проте, формула Generalized Lehmer має ще одну особливість – параметр alpha має знаходитись в межах (1, 2.7), а параметр beta в межах (-2.5, 1.5) – для коректної роботи функції. Стандартні оптимізатори не здатні обмежувати числові значення параметрів, тож, для обробки цього випадку було також розроблено функцію clip_poolings (лістинг Б.4 додатку Б).

Інший прошарок субдискретизації, що застосовується в дослідженні, – LP Pooling. Структура створеного класу повністю відповідає описаній вище, код наведено у лістингу Б.3 додатку Б. LPPooling має вільний параметр norm_type. Його межі було визначено як (1, 4), контроль за значеннями параметру також виконується за допомогою функції clip_poolings.

Після цього було створено код для роботи з даними та навчання моделі. Усі необхідні для експериментів змінні налаштовуються за допомогою параметрів командного рядка, що дозволяє створювати .sh файли для запуску серій експериментів. Приклади та код програми наведено у Додатку Б.

## 3.4 Висновки

В даному розділі було сформовано детальний план експериментів для перевірки попередньо запропонованих гіпотез. Здіснено вибір мови програмування, фреймворків та засобів, що будуть використовуватись.

Описано основні вимоги до тренувального пайплайну та іншого коду для роботи з експериментами.

Також було обрано та описано набір даних для проведення досліджень. Після чого розроблено архітектуру CNN для роботи з ним.

Для проведення досліджень було програмно реалізовано модель класифікатору зображень, прошарки субдискретизації з вільними параметрами LPPooling та запропонований нами Generalized Lehmer Pooling, методи обробки обмежень параметрів, функції для роботи з даними, тренувальний цикл для моделей та інші частини програми, необхідні для організації досліджень. Деякі частини коду наведено у додатку Б.

# 4 ПРОВЕДЕННЯ ЕКСПЕРИМЕНТІВ

## 4.1 Опис програмного та технічного обладнання

Код для проведення досліджень було розроблено за допомогою python 3.8. Необхідні елементи було реалізовано з використанням фреймворку pytorch 1.8. Для логування метрик застосовувалась бібліотека tensorboard 2.0 фреймворку tensorflow.

Для обчислення використовувалась відеокарта Nvidia GeForce GTX 1650 з об'ємом пам'яті 4 Гб. Об'єм ОЗП – 12 Гб. Процессор Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz.

Також застосовувалось веб-базоване середовище colab research.

## 4.2 Опис гіперапараметрів та налаштувань

Для навчання моделей використовувався оптимізатор Adam [45] з кроком навчання 1e-3, що знижувався після кожної епохи за допомогою домноження на 0.9. Для контролю за кроком навчання також застосовувався планувальник StepLR зі значенням gamma 0.9, тобто кожну епоху крок навчання домножувався на 0.9.

В якості функції втрат застосовувалась CrossEntropy. Розмір батчу дорівнював 256. Також було зафіксовано random seed, що дорівнював 12.

При навчанні застосовувалась техніка early stopping, що зупиняла навчання за відсутності покращення валідаційної функції втрат протягом 5 епох.

Для LPPooling параметр norm_type було ініціалізовано значенням 2, для Generalized Lehmer Pooling застосовувалась ініціалізація 1.5 для alpha та 0.1 для beta. Усі прошарки в мережі було ініціалізовано однаково.

## 4.3 Порівняння точності класифікації з GLP, LPPool та MaxPool

В першому експерименті було навчено три версії класифікатора зображень. Графік залежності функції втрат від епохи наведено на рисунку 4.1, ліва частина – тренування, права – валідація.

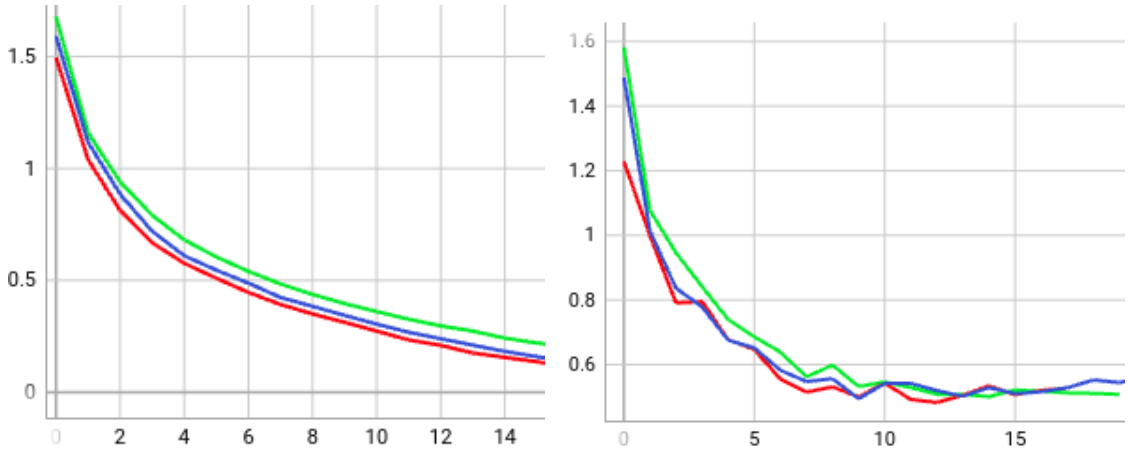


Рисунок 4.1 – Графік функції втрат моделей

Синя лінія позначає класифікатор з використанням MaxPooling, зелена лінія – модель із застосуванням LPPooling, червона лінія – модель з Generalized Lehmer Pooling.

Графік точності класифікації наведено у додатку А. Числові значення точності моделей на валідаційному датасеті наведено у таблиці 4.1.

Таблиця 4.1 – Точність отриманих моделей

| Назва моделі | MaxPool model | LPPool model | GLPPool model |
|---|---|---|---|
| Точність, % | 84.324 | 84.02 | **85.26** |

Результат мережі з застосуванням Generalized Lehmer Pooling є найкращим. Це підтверджує припущення щодо того, що адаптивний пулінг не має негативного впливу на метрики моделі. Більш того, за рахунок гнучкості точність класифікації було покращено.

## 4.4 Порівняння захищеності моделей з GLP, LPPool та MaxPool

Після цього було змодельовано конфронтаційну атаку на отримані мережі. Для цього застосовувалась змагальна атака PGD. Це ітеративний метод, що виконує пошук в просторі рішень мережі для створення змагальних збурень, які зможуть обманути модель. PGD широко застосовується у галузі змагального машинного навчання, так як є доволі надійним та простим.

PGD має декілька параметрів, що дозволяють контролювати силу атаки. Основний з них – ξ, що контролює величину збурень. Тому під час моделювання було застосовано десять спроб, де вона поступово підсилювалась.

Під час тестування класифікаторів сила атаки поступово збільшувалась за допомогою параметра ξ. Графік залежності точностей моделі від сили атаки наведено на рисунках 4.2 та 4.3.

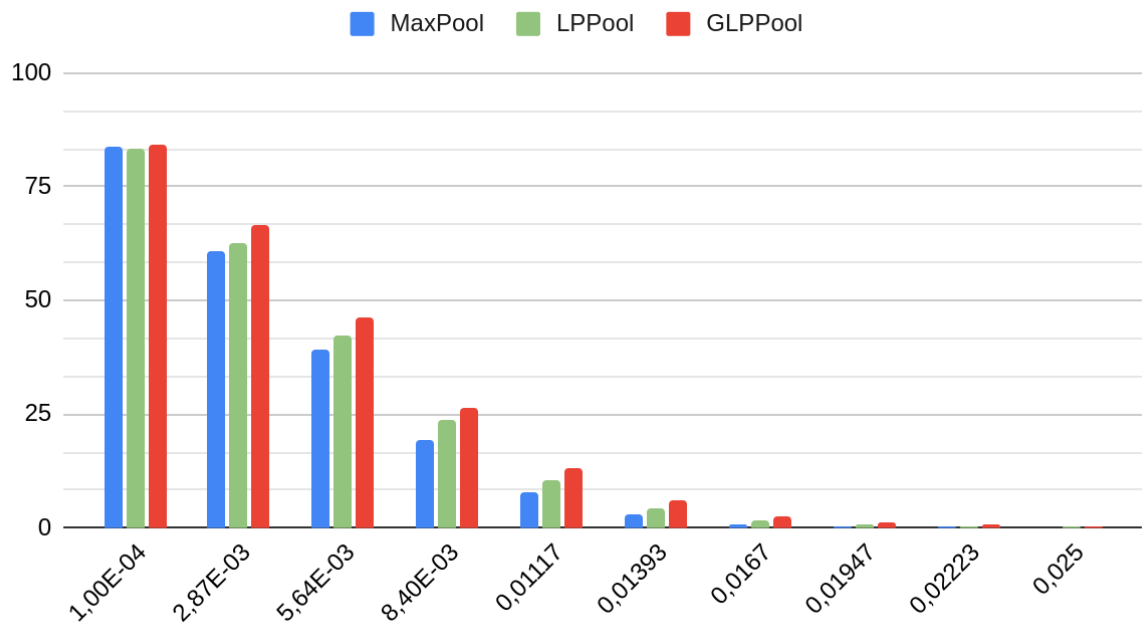


Рисунок 4.2 – Графік залежності точності класифікації від сили атаки

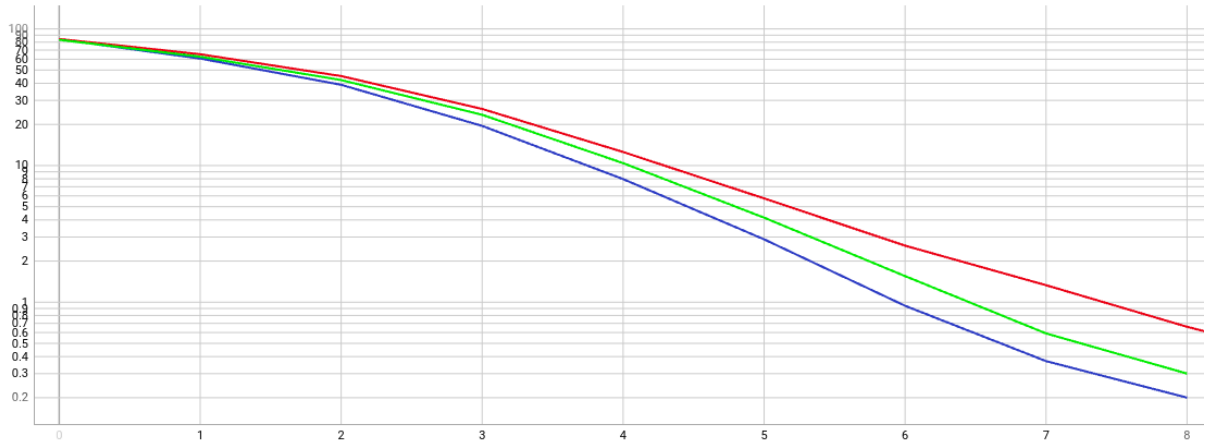

Рисунок 4.3 – Логнормований графік точності класифікації від сили атаки

Точність моделей із застосуванням адаптивних елементів залишалась вищою протягом усього експерименту. Також успішність атаки була найменшою для класифікатора з застосуванням запропонованого Generalized Lehmer Pooling. Більш детальна таблиця з показниками моделей наведена у додатку А.

## 4.5 Дослідження взаємозв'язку кількості адаптивних елементів та захищеності моделі

### 4.5.1 Тренування моделей з одним та двома прошарками пулінгу

Для оцінки впливу кількості адаптивних елементів на конфронтаційну стійкість моделі було навчено додаткові класифікатори у таких конфігураціях:

– з прошарком субдискретизації лише в одному з блоків;

– з прошарком субдискретизація в двох блоках з трьох;

– з прошарком субдискретизації у всіх трьох блоках.

Варіант, у якомі всі три блока моделі містять пулінг, було описано у попередньому наборі експериментів, тож, зосередимось на інших випадках.

Вони, в свою чергу, мають такі конфігурації:

– один пулінг в першому блоці (head);
– один пулінг в другому блоці (middle);
– один пулінг в третьому блоці (tail);
– два пулінги в першому та другому блоках (b1&b2);
– два пулінги в другому та третьому блоках (b2&b3);
– два пулінги в першому та третьому блоках (b1&b3).

Отож, було отримано 6 версій архітектури моделі для кожної з субдискретизацій, тобто, загалом 12 моделей. Графіки функцій втрат та точностей цих моделей наведено у додатку А.

Числові значення точності моделей з однією адаптивною субдискретизацією на валідаційному датасеті наведено у таблиці 4.2. Метрики для моделей з двома адаптивними пулінгами наведено у таблиці 4.3.

Таблиця 4.2 – Точність набору моделей з одним адаптивним пулінгом

| Назва моделі | Точність, % |
|---|---|
| GLP - head | 84.04 |
| GLP - middle | 84.45 |
| GLP - tail | **84.61** |
| LP - head | 83.8 |
| LP - middle | 84.28 |
| LP - tail | 84.4 |

Проаналізувавши табличні дані, можна сказати, що існують певні закономірності. Як для GLP, так і для LPPooling точність моделі найвища при розташуванні адаптивного пулінгу у останньому блоці. Трохи меншою вона стає при розташуванні в середині мережі та найнижча при застосуванні у першому. Точність GLP при цьому вища, ніж при використанні LPPool.

Таблиця 4.3 – Точність набору моделей з двома пулінгами

| Назва моделі | Точність, % |
|---|---|
| GLP – b1 & b2 | 84.42 |
| GLP – b2 & b3 | **85.24** |
| GLP – b1 & b3 | 84.73 |
| LP – b1 & b2 | 84.23 |
| LP – b2 & b3 | 85.176 |
| LP – b1 & b3 | 84.25 |

Результати, отримані в таблиці 4.3, підтверджують попередні результати. При використанні двох гнучких прошарків, точність також вища при їх розташуванні у «хвості», зменшується при схемі b1&b3 та найнижча при розташуванні субдискретизацій на початку мережі.

Підсумувавши отримані дані, можемо зробити висновок, що найбільший вплив на точність моделі має адаптивний прошарок, який розташовано ближче до кінця моделі.

### 4.5.2 Оцінка захищеності отриманих моделей

Оцінимо вплив кількості та місця розташування гнучких пулінгів на конфронтаційну стійкість моделі.

Спочатку було проведено експеримент з нейронними мережами, які використовують один пулінг. Рисунки 4.4 та 4.5 зображують графік падіння точності моделей з одним GLP в залежності від сили атаки, тобто, значення збурень $\xi$. На рисунку 4.5 графік логнормовано по осі Y для більш наглядного представлення результатів.

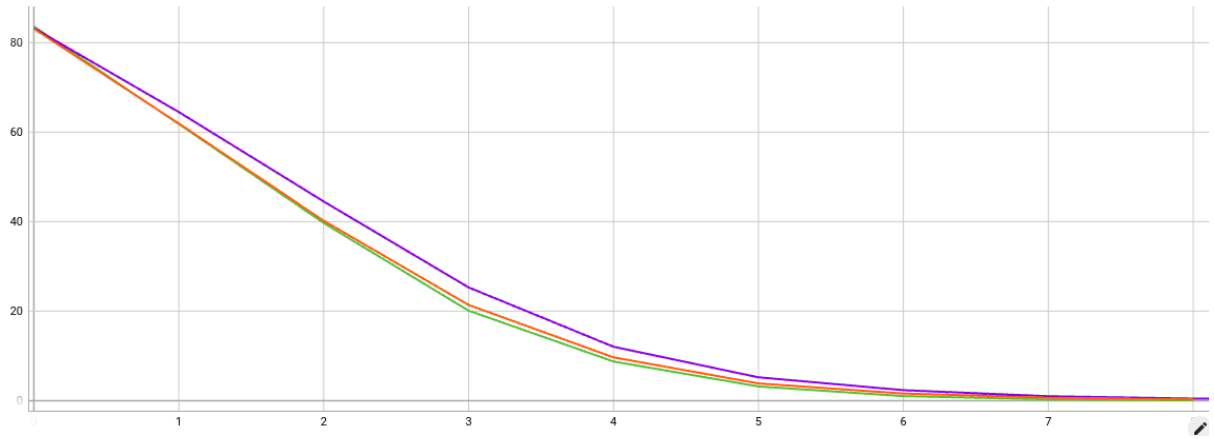


Рисунок 4.4 – Графік точності мережі з одним GLP під час атаки (фіолетова лінія – GLP head, зелена лінія – GLP middle, помаранчева – GLP tail)

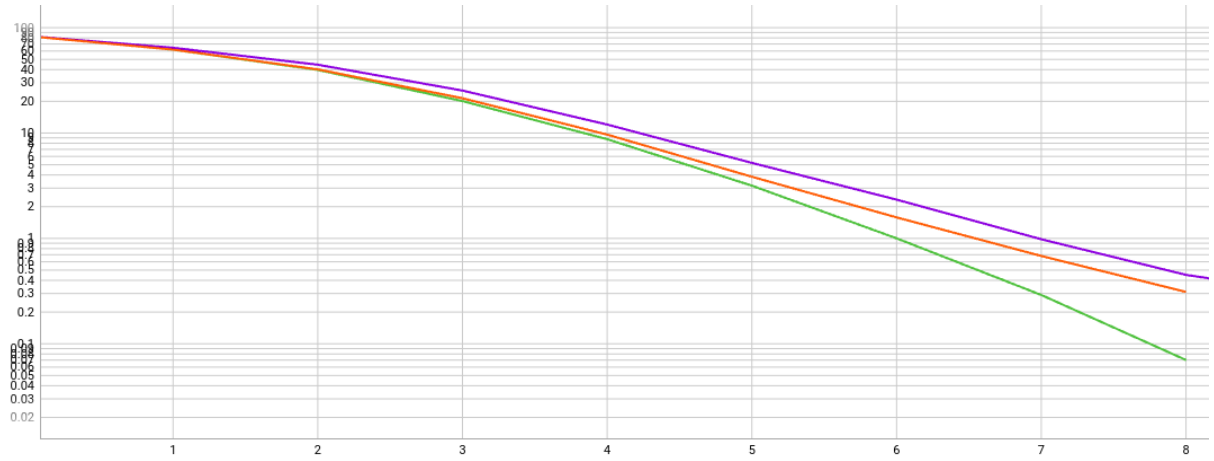


Рисунок 4.5 – Логнормований графік точності мережі з одним GLP під час атаки

Фіолетова лінія позначає мережу з GLP в блоці 1, зелена – в блоці 2, помаранчева – в блоці 3.

Наступними тестувалась стійкість моделей, що використовують GLP у двох блоках. Рисунки 4.6 та 4.7 зображують аналогічні графіки падіння точності в залежності від сили атаки для цього випадку.

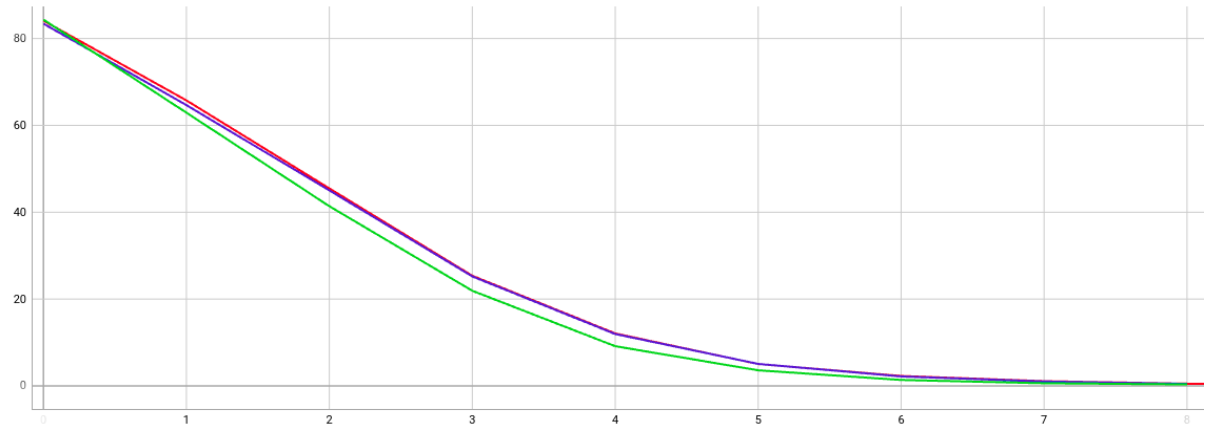


Рисунок 4.6 – Графік точності мереж з двома GLP під час атаки

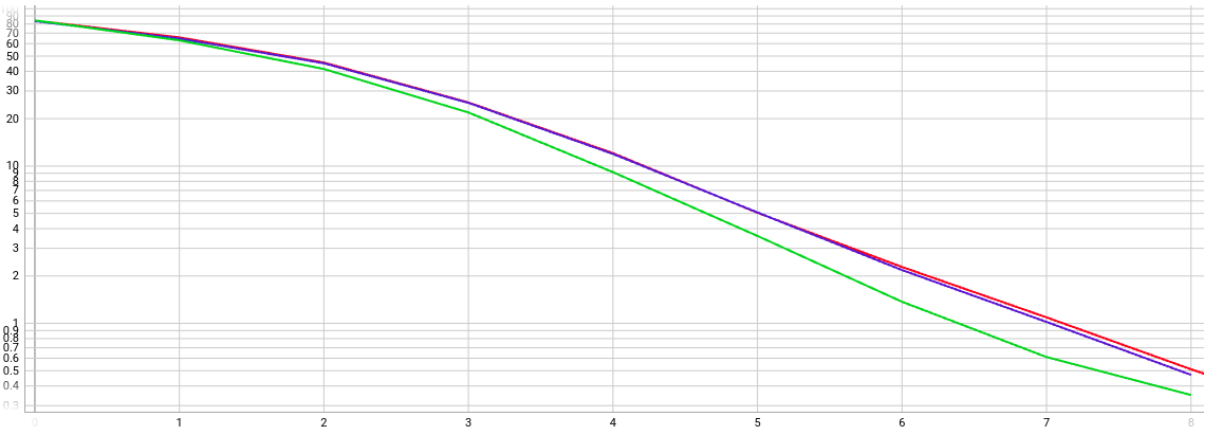


Рисунок 4.7 – Логнормований графік точності мереж з двома GLP під час атаки

Фіолетова лінія позначає мережу з GLP в блоці 1 та 2, зелена – в блоці 2 та 3, помаранчева – в блоці 1 та 3.

Після цього було протестовано мережі, в яких застосовувався LPPooling. Моделювання атаки на конфігурації з одним пулінгом зображено на рисунку 4.8. Рисунок 4.9 зображує аналогічний процес для мереж з двома LPPool відповідно.

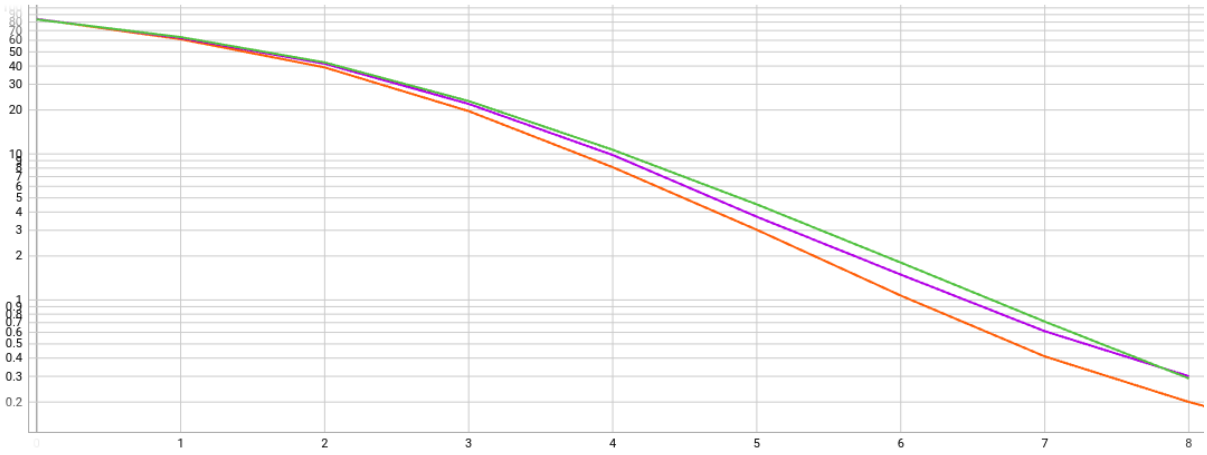

Рисунок 4.8 – Логнормований графік точності мереж з одним LPPool під час атаки (зелений – b1, фіолетовий – b2, помаранчевий – b3)

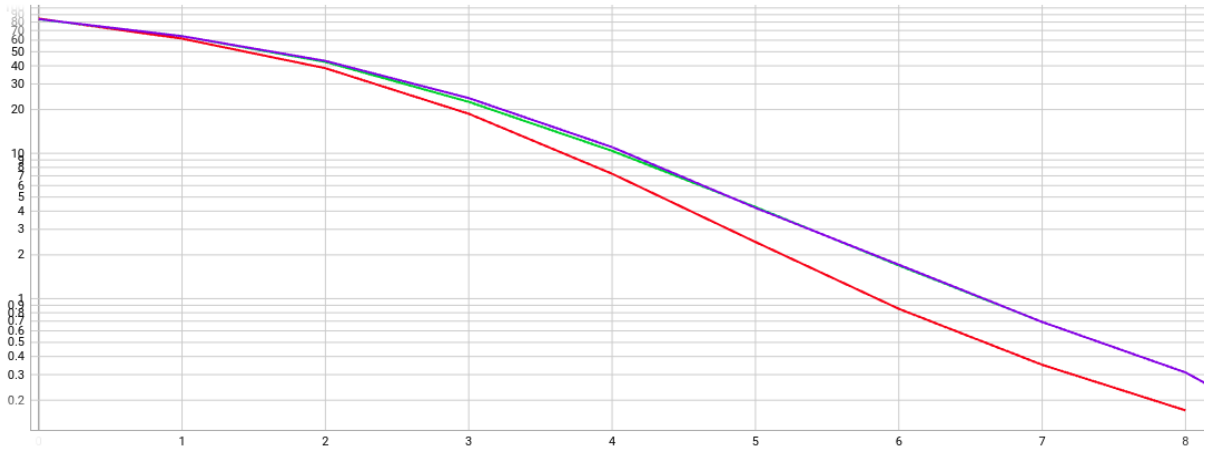

Рисунок 4.9 – Логнормований графік точності мереж з двома LPPool під час атаки (фіолетовий – b1&b3, зелений – b1&b2, помаранчевий – b2&b3)

Проаналізуємо результати моделювання атаки алгоритмом PGD, зображені на графіках вище. Моделі було розбито на дві групи, в кожній з яких порівнювались класифікатори з одним видом адаптивного пулінгу (або GLP, або LPPool), та підгрупи за кількістю прошарків субдискретизації. Проаналізувавши результати моделювання конфронтаційних атак, можемо зробити декілька висновків:

– найбільш вплив на захищеність моделі має використання

адаптивного пулінгу на початку мережі;

– ключову роль у цьому процесі відіграє не кількість пулінгів, а їх розташування;

– серед варіантів з двома пулінгами, оптимальним за точністю та захищеністю є розташування на початку та в кінці.

Також цікавим буде порівняти результати обох груп моделей між собою. Для цього оберемо ті з них, які мають оптимальні результати як за точністю, так і за рейтингом успіху атаки. Крім того не варто забувати про мережі з трьома прошарками субдискретизації, які було досліджено у попередньому наборі експериментів. Найбільш успішні з них також буде додано до порівняння.

Графіки з порівняння точності під атаками для GLP та LPPool приведено на рисунку 4.10.

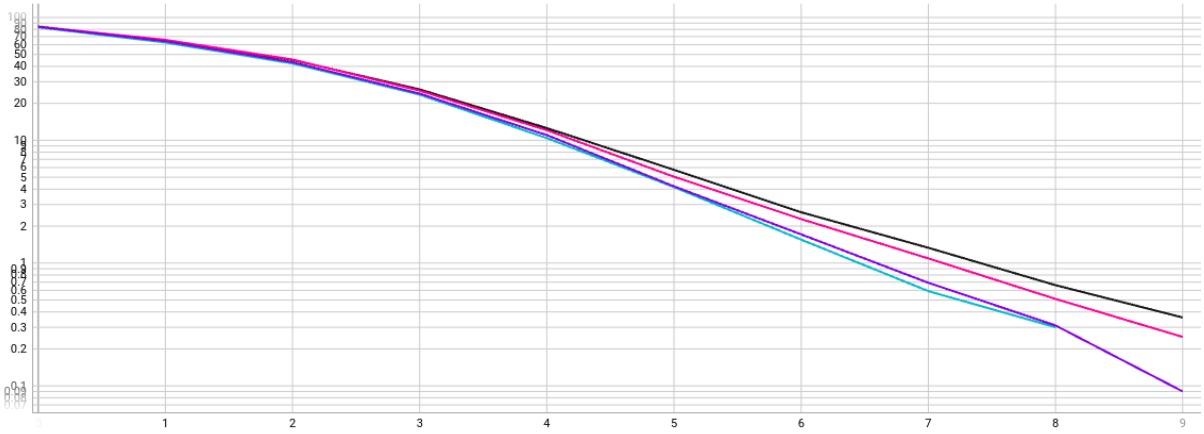


Рисунок 4.10 – Логнормований графік точності мереж під час атаки (чорний – GLP всюди, помаранчевий – GLP b1&b3, фіолетовий – LPPool b1&b3, блакитний – LPPool всюди)

За результатами досліджень найкращий результат за обома параметрами було досягнуто при використанні запропонованого GLP у всіх трьох блоках. Точність класифікації порівняно з MaxPool було покращено на 0,94%, зменшення падіння точності під атакою біля 5%.

## 4.6 Висновки

В даному розділі було проведено декілька наборів експериментів, представлено та проаналізовано їх результати. Деякі додаткові матеріали містяться у додатку А.

На підставі проведених досліджень було підтверджено, що застосування GLP не має негативного впливу на якість роботи моделі. Точність класифікації за його застосування підвищилася на 0.92%. При застосуванні LPPool спостерігалась незначна втрата точності.

Також з експериментів з використання LPPool та GLP можна зробити висновок, що застосування гнучких елементів у нейронній мережі призводить до покращення конфронтаційної стікості, так як використання будь-якого з цих елементів зменшувало падіння точності класифікації під атакою.

Взаємозв'язок між кількістю адаптивних прошарків та захищеністю моделі від збурень виявлено не було.

Проте було з'ясовано, що гнучкі субдискретизації розташовані на початку моделі більш сильно впливають на її конфронтаційну стійкість, тоді як пулінги, розташовані ближче до кінця, мають вплив на точність моделі. Тож, розташування адаптивних елементів має більш важливе значення, ніж їх кількість.

Серед усіх класифікаторів, створених в процесі експериментів, найбільш оптимальною була модель всі прошарки пулінгу у якій було замінено на GLP. Точність моделі склала 85.26 (+0.92%), а рейтинг успіху атак знизився близько на 5%, максимальним зменшенням було 7,31%.

В усіх проведених експериментах спостерігалась перевага мереж з GLP як по точності класифікації, так і по конфронтаційній стійкості, що свідчить про ефективність запропонованого рішення.

# ВИСНОВКИ

В роботі розглядалась проблема вразливості моделей глибокого навчання до змагальних атак. Ретельно підібрані змагальні приклади здатні обманути моделі, завадити її роботі або навіть призвести до викрадення приватних даних. До того ж, кількість методів атак постійно збільшується. З проведеного аналізу предметної галузі було з'ясовано, що проблема становить суттєву загрозу, так як нейронні мережі присутні і в критичній інфраструктурі, і в житті мільйонів людини.

Крім того, було проаналізовано наявні способи захисту та підсумовано, що кількість ефективних засобів мала і цей напрям потребує подальшого дослідження. Також було з’ясовано, що існують приклади, коли застосування гнучких або біологічно подібних елементів зчинило позитивний вплив на конфронтаційну стійкість мережі.

Виходячи з цього, було запропоновано дослідити взаємозв’язок між застосуванням гнучких прошарків у нейронних мережах та їх захищеністю від змагальних атак. Дослідження проводилось на прикладі класифікаторів зображень та адаптивних прошарків субдискретизації. Було розглянуто наявні прошарки адаптивного пулінгу та принципи їх роботи. Крім того, в якості можливого рішення було запропоновано власний прошарок субдискретизації Generalized Lehmer Pooling.

На основі проведеного дослідження було сформульовано та поставлено задачі для дослідження.

При вирішенні задач було виконано теоретичне дослідження феномену конфронтаційних прикладів, розглянуто процес проведення змагальної атаки та описано математичну модель і властивості Generalized Lehmer Pooling. Після цього було висунуто гіпотези щодо взаємозв’язку адаптивності та стійкості мереж, а також можливостей GLP.

Для організації проведення експериментів було програмно реалізовано класифікатор зображень, прошарки адаптивної

субдискретизації GLP та LPPool та методи для роботи з ними, тренувальний пайплайн та інше.

В ході проведення досліджень було навчено набір моделей в різних конфігураціях, з використанням різних прошарків пулінгу. Порівнювались результати моделей з застосуванням MaxPooling та адаптивних прошарків LPPool і GLP. Проаналізовано їх точність на чистих даних та рейтинг успішності змодельованих атак.

Завдяки результатам експериментів було підтверджено гіпотезу про те, що моделі з адаптивними елементами мають більш високий початковий рівень змагального захисту, аніж класифікатори з традиційними фіксованими функціями.

Гіпотеза про вплив кількості елементів на рівень захищеності була спростована. Проте виявлено деякі інші залежності. Так, адаптивні пулінги розташовані глибше у мережі чинили більший вплив на її точність, а пулінги, розташовані на початку, більше впливали на конфронтаційну стійкість.

Також було підтверджено, що за сукупністю метрик моделі та ступеню її змагальної захищеності, найкращі результати показує запропонований GLP. Точність класифікатора при його використанні зросла на 0,92%, а максимальне зниження рейтингу успіху атак дорівнювало 7,31%.

Отож, в даній роботі було досліджено взаємозв'язок між застосуванням гнучких елементів та конфронтаційною стійкістю моделей, проаналізовано наявні адаптивні прошарки субдискретизації та запропоновано власний, доведено його ефективність як з боку покращення змагального захисту мережі, так і для покращення метрик нейронної мережі. Поставлені в роботі задачі виконано в повному обсязі. В роботі реалізовано та досліджено адаптивні елементи, виявлено взаємозв'язок між їх застосуванням та захищеністю мережі, тобто поставлена мета досягнута.

Отримані результати можуть бути використані при плануванні та розробці архітектур нейронних мереж для підвищення їх початкового рівня

стійкості. Крім того, запропоноване рішення може комбінуватись з іншими методами захисту, такими як змагальне тренування.

Також існує декілька можливих напрямів для подальших досліджень. Для більш глибокого аналізу взаємозв’язку між адаптивністю та захищеністю мережі варто провести експерименти з більш специфічними датасетами, які можуть більш широко розкрити здатності функцій до гнучкого підлаштування. Також має бути досліджено застосування інших видів адаптивних елементів.

Цікавим напрямом є комбінування адаптивності та інших видів захисту, як біологічно-подібного, так і евристичного. Також, при досягненні стійких успішних результатів для таких комбінацій, слід дослідити найбільш успішні способи їх атак, що дозволить більш глибоко розкрити властивості такої взаємодії.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Terziyan V., Golovianko M., Gryshko S. Industry 4.0 Intelligence under Attack: From Cognitive Hack to Data Poisoning / Cyber Defence in Industry 4.0 Systems and Related Logistics and IT Infrastructures, 2018. T. 51. C. 110-125.

2. Yuan x. Et al. Adversarial examples: attacks and defenses for deep learning / IEEE transactions on neural networks and learning systems. 2019. T. 30. №. 9. C. 2805-2824.

3. Li G. et al. Security matters: A survey on adversarial machine learning / arXiv preprint arXiv:1810.07339. 2018.

4. Rozsa A., Günther M., Boult T. E. Are accuracy and robustness correlated / 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, 2016. C. 227-232.

5. Alom M. Z. et al. A state-of-the-art survey on deep learning theory and architectures / Electronics, 2019. T. 8. №. 3. C. 292.

6. Dalvi N. et al. Adversarial classification / Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. 2004. C. 99-108.

7. Szegedy C. et al. Intriguing properties of neural networks / arXiv preprint arXiv:1312.6199. 2013.

8. Vorobeychik Y., Kantarcioglu M. Adversarial machine learning / Synthesis Lectures on Artificial Intelligence and Machine Learning. 2018. T. 12. №. 3. C. 1-169.

9. Chakraborty A. et al. A survey on adversarial attacks and defences / CAAI Transactions on Intelligence Technology. 2021. T. 6. №. 1. C. 25-45.

10. Chen X. et al. Targeted backdoor attacks on deep learning systems using data poisoning / arXiv preprint arXiv:1712.05526. 2017.

11. Balle B., Cherubin G., Hayes J. Reconstructing Training Data with Informed Adversaries / arXiv preprint arXiv:2201.04845. 2022.

12. Tramèr F. et al. Stealing Machine Learning Models via Prediction {APIs} / 25th USENIX security symposium (USENIX Security 16). 2016. С. 601-618.

13. A Complete List of All (arXiv) Adversarial Example Papers by Nicholas Carlini. URL: https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html (Last accessed: 25.03.2022)

14. Biggio B., Roli F. Wild patterns: Ten years after the rise of adversarial machine learning / Pattern Recognition. 2018. Т. 84. С. 317-331.

15. Carlini N., Wagner D. Adversarial examples are not easily detected: Bypassing ten detection methods / Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. 2017. С. 3-14.

16. Ren K. et al. Adversarial attacks and defenses in deep learning / Engineering. 2020. Т. 6. №. 3. С. 346-360.

17. Singla V. et al. Low curvature activations reduce overfitting in adversarial training / Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021. С. 16423-16433.

18. Branytskyi V. et al. Generative adversarial networks with bio-inspired primary visual cortex for Industry 4.0 / Procedia Computer Science. 2022. Т. 200. С. 418-427.

19. Малик Д. Дослідження впливу активаційних функцій на навчання мережі AI-GAN. / XXV Форум «Радіоелектроніка та молодь». 2021. Т.5.

20. Lindsay G. W. Convolutional neural networks as a model of the visual system: Past, present, and future / Journal of cognitive neuroscience. 2021. Т. 33. №. 10. С. 2017-2031.

21. Dapello J. et al. Simulating a primary visual cortex at the front of CNNs improves robustness to image perturbations / Advances in Neural Information Processing Systems. 2020. Т. 33. С. 13073-13087.

22. Nayebi A., Ganguli S. Biologically inspired protection of deep networks from adversarial attacks / arXiv preprint arXiv:1703.09202. 2017.

23. Akhtar N., Mian A. Threat of adversarial attacks on deep learning in

computer vision: A survey / IEEE access. 2018. T. 6. C. 14410-14430.

24. Chen L., Ye Y., Bourlai T. Adversarial machine learning in malware detection: Arms race between evasion attack and defense / 2017 European intelligence and security informatics conference (EISIC). IEEE, 2017. C. 99-106.

25. Goodfellow I. J., Shlens J., Szegedy C. Explaining and harnessing adversarial examples / arXiv preprint arXiv:1412.6572. 2014.

26. Kurakin A., Goodfellow I., Bengio S. Adversarial machine learning at scale / arXiv preprint arXiv:1611.01236. 2016.

27. Tramèr F. et al. The space of transferable adversarial examples / arXiv preprint arXiv:1704.03453. 2017.

28. Papernot N. et al. Distillation as a defense to adversarial perturbations against deep neural networks / 2016 IEEE symposium on security and privacy (SP). IEEE, 2016. C. 582-597.

29. Papernot N. et al. The limitations of deep learning in adversarial settings / 2016 IEEE European symposium on security and privacy (EuroS&P). IEEE, 2016. C. 372-387.

30. Papernot N. et al. Practical black-box attacks against machine learning / Proceedings of the 2017 ACM on Asia conference on computer and communications security. 2017. C. 506-519.

31. Madry A. et al. Towards deep learning models resistant to adversarial attacks / arXiv preprint arXiv:1706.06083. 2017.

32. Dumoulin V., Visin F. A guide to convolution arithmetic for deep learning / arXiv preprint arXiv:1603.07285. 2016.

33. Yu D. et al. Mixed pooling for convolutional neural networks / International conference on rough sets and knowledge technology. Springer, Cham, 2014. C. 364-375.

34. Zhou Y. et al. Adaptive pooling in multi-instance learning for web video annotation / Proceedings of the IEEE International Conference on Computer Vision Workshops. 2017. C. 318-327.

35. Lee C. Y., Gallagher P. W., Tu Z. Generalizing pooling functions in

convolutional neural networks: Mixed, gated, and tree / Artificial intelligence and statistics. PMLR, 2016. С. 464-472.

36. Documentation for Torch.NN. URL: https://pytorch.org/docs/stable/generated/torch.nn.LPPool2d.html (Last accessed: 27.03.2022)

37. Beliakov G., Špirková J. Weak monotonicity of Lehmer and Gini means / Fuzzy sets and systems. 2016. Т. 299. С. 26-40.

38. Terziyan V. Social Distance metric: from coordinates to neighborhoods / International Journal of Geographical Information Science. 2017. Т. 31. №. 12. С. 2401-2426.

39. Терзиян В. Я., Малик Д. Г. Вплив адаптивних елементів мереж на конфронтаційну робастність. / XXVI Форум «Радіоелектроніка та молодь». 2022. Т.6.

40. Ozgur C. et al. MatLab vs. Python vs. R / Journal of Data Science. 2017. Т. 15. №. 3. С. 355-371.

41. Paszke A. et al. Pytorch: An imperative style, high-performance deep learning library / Advances in neural information processing systems. 2019. Т. 32.

42. Martinez M. T. An Overview of Google's Machine Intelligence Software TensorFlow. 2016.

43. Krizhevsky A. et al. Learning multiple layers of features from tiny images. 2009.

44. Kingma D. P., Ba J. Adam: A method for stochastic optimization / arXiv preprint arXiv:1412.6980. 2014.

45. Documentation for Torch.NN. URL: https://pytorch.org/docs/stable/generated/torch.nn.Module.html (Last accessed: 11.04.2022)