МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

кафедра “Інформаційні системи та мережі”

ПОЯСНЮВАЛЬНА ЗАПИСКА

до магістерської кваліфікаційної роботи на тему:

«Інтелектуальна система виявлення спуфінг атак на бпла на основі глибокої нейронної мережі»

Студента групи САМ-21 Степаняка Ярослава Андрійовича
(група, шифр, прізвище та ініціали)

Керівник роботи _____ (___Роман ПЕЛЕЩАК____)
Консультант _____ (___Тетяна ШЕПЕЛЬ____)
Нормоконтроль _____ (___Андрій ВАСИЛЮК____)

Завідувач кафедри ІСМ _____ (**Дмитро ДОСИН**)
«_21_» ___травня_____ 2025 р.

ЛЬВІВ - 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Інститут комп’ютерних наук та інформаційних технологій
Кафедра “Інформаційні системи та мережі”
Спеціальність 124 “Системний аналіз”
ОНП «Системний аналіз»

ЗАТВЕРДЖУЮ:
Зав. кафедри ІСМ _____
“_21_” ___травня_____2025р.

ЗАВДАННЯ
на кваліфікаційну роботу студента групи САМ-21

Степаняк Ярослав Андрійович
(прізвище, ім’я, по батькові)

1.Тема роботи: Інтелектуальна система виявлення спуфінг атак на бпла на основі глибкої нейронної мережі
затверджена наказом за НУ «ЛП» від «_11_» ___березня____ 2025 р. №_864-4-06

2. Термін здачі студентом закінченої роботи: ___12 травня 2025р. _____

3. Вихідні відомості до роботи: наукові публікації, літературні та інші джерела мережі Інтернет

4. Зміст розрахунково-пояснювальної записки: (перелік питань, які належить розробити) аналітичний огляд літературних та онлайн-джерел, системний аналіз об’єкта дослідження та обґрунтування проблеми, методи та засоби вирішення завдання, практична реалізація, економічна частина

5. Перелік графічного матеріалу: _рисунки графічного прикладу спуфінг атаки, графіків сигналів синхронних, асинхронних атак спуфінгу та автентичний сигнал GPS, діаграма варіантів використання, діаграма діяльності, діаграма розгортання, візуалізація контрольного прикладу

6. Перелік програмних продуктів, які належить використати в процесі розроблення роботи (проекту) Microsoft Word 365, Visual Paradigm, Visual Studio Code, Google Colab

7. Консультування роботи, із зазначенням розділів роботи

Розділ	Консультанти	Підпис, дата	
		завдання видав	завдання отримав
Економічний	Т. Шепель	10.03.2025	

8. Дата видачі завдання __24 лютого 2025р._____

Керівник _____
(підпис)

Завдання отримав_____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

	Етапи кваліфікаційної роботи	Термін виконання	Примітки
1	Аналітичний огляд літературних та інших джерел	24.02.2025 – 07.03.2025	Виконано
2	Системний аналіз та обґрунтування проблемної ситуації	10.03.2025 – 17.03.2025	Виконано
3	Огляд методів та засобів вирішення проблеми	17.03.2025 – 24.03.2025	Виконано
4	Практична реалізація системи	24.03.2025 – 14.04.2025	Виконано
5	Економічне обгрунтування доцільності роботи	14.04.2025 – 18.04.2025	Виконано
6	Формулювання загальних висновків до роботи та оформлення записки	14.04.2025 – 16.04.2025	Виконано

Студент_____
(підпис)

Керівник роботи_____
(підпис)

ЗМІСТ

ВСТУП 6
РОЗДІЛ 1 Аналітичний огляд літературних та інших джерел 9
1.1. Поняття GPS-спуфінгу. Види атак 9
1.2. Методи захисту від GPS-спуфінгу 12
1.3. INS-захист і інші системи 14
1.4. Антенні системи та їх роль 16
1.5. Застосування машинного навчання 16
1.6. Аналіз існуючих рішень 19
Висновки до розділу 20
РОЗДІЛ 2 Системний аналіз та обґрунтування проблемної ситуації 22
2.1 Системний аналіз об'єкта дослідження та предметної області 22
2.1.1 Визначення проєктованої системи та її середовища 22
2.1.2 Діаграма варіантів використання 32
2.1.3 Діаграма діяльності 35
2.1.4 Діаграма розгортання 37
2.2 Постановка та обґрунтування проблеми 41
Висновки до розділу 46
РОЗДІЛ 3 Методи та засоби вирішення задачі 48
3.1. Вибір методів вирішення задачі 48
3.2. Засоби вирішення задачі 49
3.2.1. Вибір програмного забезпечення та середовища 49
3.2.2. Опис набору даних TEXBAT 50
3.2.3. Обробка сигналів та підготовка ознак 51
3.3. Архітектура розробленої нейромережі 56
3.4. Побудова прототипу системи 60
Висновки до розділу 63
РОЗДІЛ 4 Практична реалізація 64
4.1. Опис реалізації програмного рішення 64

4.1.1. Технічні засоби та середовище виконання 64
4.1.2. Функціональність графічного інтерфейсу користувача 70
4.2. Аналіз отриманих результатів 74
4.2.1. Приклад використання програми 74
4.2.2. Аналіз результатів роботи системи на прикладі 81
Висновки до розділу 83
РОЗДІЛ 5 84
Економічне обґрунтування доцільності роботи 84
Висновки до п'ятого розділу 88
ВИСНОВКИ 89
СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ 91
АНОТАЦІЯ 94
ABSTRACT 97
ДОДАТКИ 100
Додаток А 100

ВСТУП

Розвиток безпілотних літальних апаратів (БПЛА) упродовж останнього десятиліття суттєво вплинув на технологічний прогрес у різних сферах — від військових і правоохоронних операцій до сільського господарства, логістики та медіавиробництва. Проте стрімке впровадження дронів супроводжується зростанням загроз, пов'язаних із навмисними втручаннями в їхню навігацію. Однією з найбільш небезпечних і водночас важковиявлюваних форм такого втручання є GPS-спуфінг — атака, під час якої зловмисник транслює підроблені сигнали глобального позиціонування з метою введення апарата в оману.

На сьогодні існуючі методи захисту БПЛА від спуфінгу потребують або складного спеціалізованого обладнання, або є малоефективними щодо синхронних атак. У таких умовах особливої актуальності набуває розроблення програмних рішень, що здатні виявляти спуфінг у реальному часі з високою точністю, не вимагаючи суттєвих змін до апаратної частини дрона. Застосування методів штучного інтелекту, зокрема глибоких нейронних мереж, відкриває нові можливості у створенні надійних систем детектування навігаційних атак. Така розробка є важливою як для підвищення безпеки цивільної авіації, так і для обороноздатності України в умовах гібридних загроз.

Метою даної магістерської кваліфікаційної роботи є розробка інтелектуальної системи виявлення GPS-спуфінг атак на БПЛА на основі глибокої нейронної мережі, здатної працювати автономно в реальному часі та забезпечувати високу точність виявлення навігаційних аномалій.

Для досягнення цієї мети в роботі було поставлено та розв'язано наступні **задачі**: проаналізувати сучасні загрози навігаційній безпеці БПЛА та існуючі методи протидії GPS-спуфінгу; здійснити системний аналіз предметної області та обґрунтувати архітектуру майбутньої системи; розробити методику формування ознак сигналів GPS для аналізу стану навігації; побудувати модель глибокої нейронної мережі для класифікації станів навігаційної системи; реалізувати програмний прототип системи виявлення спуфінгу та провести її тестування на

реальних даних; оцінити економічну доцільність впровадження розробленої системи в практичні застосування.

Об'єктом дослідження є навігаційна система безпілотного літального апарата, яка піддається ризику зовнішнього втручання у вигляді GPS-спуфінгу.

Предметом дослідження є методи та алгоритми виявлення спуфінг-атак за допомогою глибоких нейронних мереж на основі аналізу навігаційних параметрів сигналів.

***Наукова новизна одержаних результатів*. У** цій роботі вперше запропоновано програмну архітектуру системи виявлення GPS-спуфінгу на безпілотних літальних апаратах, що ґрунтується на використанні багатошарового перцептрону (MLP), навченого на реальних навігаційних даних. На відміну від відомих рішень, де застосовуються здебільшого класичні методи фільтрації та порогового аналізу, у цій системі використано комбіновану структуру ознак сигналу (псевдовідстань, доплерівський зсув, C/N0) для формування векторного представлення стану навігації.

Вперше розроблено модель, адаптовану до роботи в умовах обмежених обчислювальних ресурсів бортового обладнання БПЛА, яка забезпечує прийнятну швидкодію при високій точності класифікації. Удосконалено підхід до попередньої обробки даних для MLP-класифікації з використанням нормалізації та агрегування ознак у реальному часі. Крім того, отримало подальший розвиток практичне застосування методів глибокого навчання для виявлення синхронних навігаційних атак — з акцентом на їх реальну інтеграцію в автономні системи навігації.

Апробація результатів дослідження. Основні положення та результати дисертаційної роботи були апробовані на всеукраїнській науково-практичній конференції "Застосування Сухопутних військ Збройних Сил України у конфліктах сучасності (за досвідом забезпечення національної безпеки складовими сектору безпеки і оборони у ході російсько-української війни)", яка відбулася 28–29 листопада 2024 року.

Доповідь: Василь Литвин, Іван Пелещак, Ярослав Степаняк, Роман Пелещак. Виявлення і розпізнавання спуфінг GPS-атак на дрони за допомогою багатошарового перцептрона.

Результати також подані до публікації у **фаховому науковому журналі** *Computing* (ISSN 1727-6209, Online ISSN 2312-5381), у статті на тему - **«High-Precision Detection of GPS Spoofing Attacks on UAVs Using MLP»**,

РОЗДІЛ 1
Аналітичний огляд літературних та інших джерел

У цьому розділі наведено огляд літературних джерел за тематикою захисту безпілотних літальних апаратів від навігаційних атак типу GPS-спуфінгу. Розглянуто сутність та класифікацію таких атак, існуючі методи їх виявлення та запобігання, а також проведено аналіз переваг і недоліків відомих рішень.

1.1. Поняття GPS-спуфінгу. Види атак

Система глобального позиціонування GPS є основою навігації більшості сучасних безпілотних літальних апаратів (БПЛА). Однак такі навігаційні системи є вразливими до навмисних втручань. Зокрема, існують дві основні загрози для GPS-зв'язку БПЛА: **глушіння** сигналу (англ. *jamming*) та **спуфінг** сигналу [1]. При глушінні атакувальник намагається створити перешкоди (атака типу «відмова в обслуговуванні»), щоб приймач БПЛА не міг отримувати сигнали GPS. Натомість при спуфінгу зловмисник випромінює підроблений GPS-сигнал, який імітує справжній, але має вищу потужність, через що бортовий приймач починає приймати саме цей сигнал як референсний [2]. У результаті БПЛА перестає враховувати автентичні навігаційні сигнали і поступово відхиляється від свого справжнього курсу.

Метою GPS-спуфінгу є непомітне обдурення навігаційної системи. Атакувальник формує хибні навігаційні дані, підміняючи справжні координати на задані ним значення. Зазвичай для цього генерують фальшиві радіосигнали, дуже подібні до сигналів супутників, проте дещо сильніші за потужністю. Як правило, спуфінгові сигнали можуть бути отримані шляхом **затримки та повторного випромінювання** автентичного сигналу (прийом та ретрансляція із затримкою, відома як *meaconing*) [3]. Через те, що сигнал від GPS-супутників надходить на Землю зі слабкою потужністю (близько –120 дБм), відносно потужний передавач поблизу БПЛА (наприклад, програмно визначене радіо) здатен пересилати імітований сигнал з рівнем ~–50 дБм, легко пересилюючи справжній. Приймач БПЛА автоматично

обирає найсильніший сигнал, яким і виявляється підроблений сигнал спуфера. Таким чином, дрон починає обчислювати місцеположення на основі хибних даних і може відхилятися від заданого маршруту без різких аномалій [4].

GPS-спуфінг є особливо небезпечною атакою, оскільки при його вправному виконанні БПЛА майже не здатний відразу виявити підміну сигналу. Якщо спуфер діє синхронно зі справжнім GPS, зміни координат відбуваються плавно, без стрибків, тому бортові алгоритми не фіксують очевидних збоїв. До того ж, у дрона немає «еталонного» правильного положення, з яким можна порівняти отримані координати, що ускладнює виявлення відхилення [5]. Це дає змогу нападнику непомітно перебрати контроль над навігацією: отримавши статус основного джерела сигналу, атакувальник може змінювати характеристики референсного сигналу і тим самим керувати переміщенням апарата [6]. Наприклад, зафіксовано сценарії «викрадення дрона» [7], коли спуфер передавав координати, відповідні місцю розташування самого нападника, змушуючи БПЛА приземлитися поблизу нього. Більш складні атаки дозволяють плавно змінювати маршрут апарата на довільний, не викликаючи спрацювання систем навігаційного контролю [8, 9].

Існує два основних типи GPS-спуфінгу за характером дій атакувальника:

- **Синхронна (координована) спуфінг-атака.** Зловмисник *відстежує* автентичний сигнал цільового БПЛА і формує його точну копію, яка перевищує оригінал за потужністю [4]. Такий підроблений сигнал надсилається в приймач одночасно зі справжнім і поступово повністю його витісняє завдяки більш високому рівню. У результаті дрон непомітно переключається на хибний сигнал як на основне джерело навігації. Підтримуючи синхронізацію (фазу, час) зі справжнім сигналом, спуфер *не спричиняє різких змін* у показниках навігації, тож система не має очевидних ознак атаки. Таким способом атакувальник отримує можливість плавно керувати місцеположенням БПЛА, підсовуючи йому поступово змінені координати.

- **Несинхронна спуфінг-атака.** Зловмисник випромінює *незалежно згенерований* GPS-сигнал, який не збігається з реальним по фазі чи часу. Це призводить до того, що дрон раптом отримує явно некоректні навігаційні дані – фактично відбувається стрибок визначеного місцеположення на інше. Подібна різка невідповідність може бути помічена системою керування польотом, тому несинхронний спуфінг загалом легше виявити. Водночас, якщо захисних механізмів немає, навіть нескоординована атака здатна викликати хаотичну поведінку БПЛА (раптові зміни курсу, втрата стабільності чи навіть аварію) [10,11].

На рис. 1.1. зображено рафіки сигналів синхронних, асинхронних атак спуфінгу та автентичний сигнал GPS у координатах фази живлення

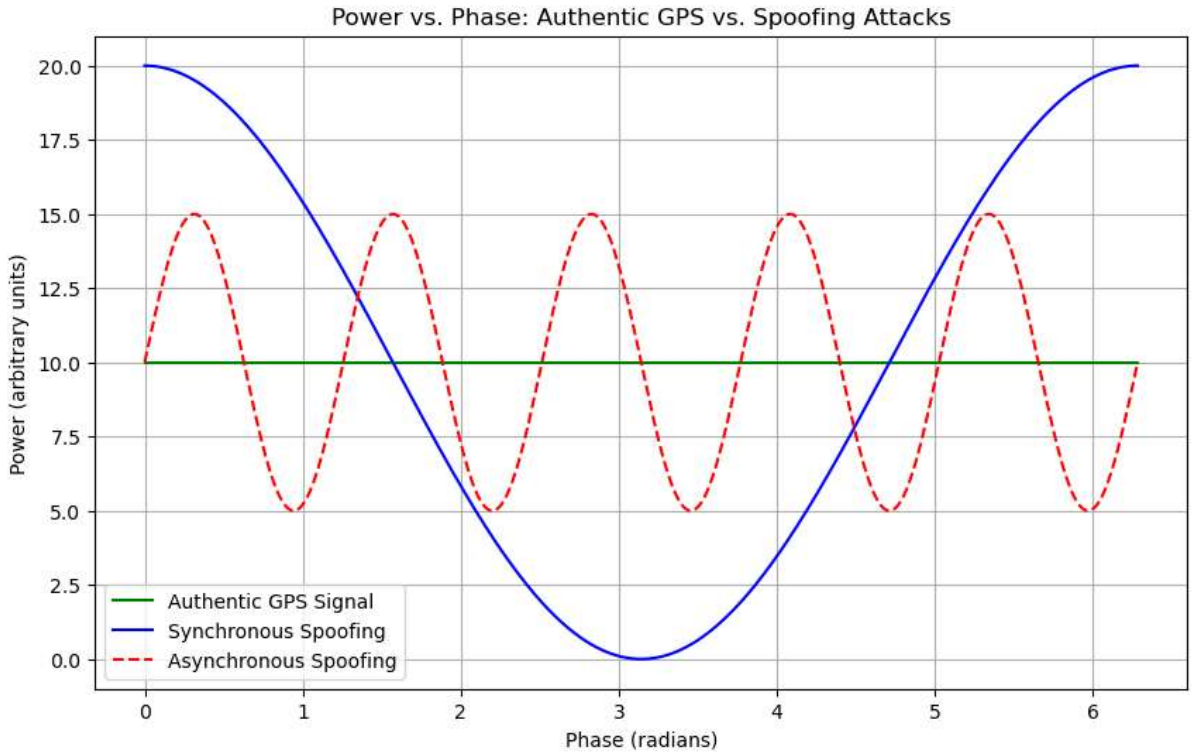


Рис 1.1 Графіки сигналів синхронних, асинхронних атак спуфінгу та автентичний сигнал GPS у координатах фази живлення

Уразливість БПЛА до GPS-спуфінгу зумовлена відкритим характером цивільних навігаційних сигналів. Більшість комерційних дронів використовують нешифровані й неавтентифіковані сигнали GPS для навігації, тож не мають вбудованого захисту

від підробки координат [6]. Таким чином, навіть відносно прості технічні засоби можуть бути використані для здійснення спуфінг-атаки на безпілотник. Проблема набуває особливої актуальності у військовій сфері та критичних застосуваннях: БПЛА дедалі частіше виконують відповідальні завдання (військова розвідка, доставка вантажів, інфраструктурний моніторинг, сільське господарство тощо), і спрямовані радіочастотні атаки на їхні навігаційні системи стають серйозною загрозою. За останнє десятиліття проблематика безпеки авіації та навігації привертає значну увагу дослідників, причому спуфінг GPS визначено одним із найнебезпечніших видів кібератак на БПЛА [7].

На рис. 1.2. показано, як зловмисник використовує GPS спуфінг, щоб змусити автономний дрон віддалитися від пункту призначення:

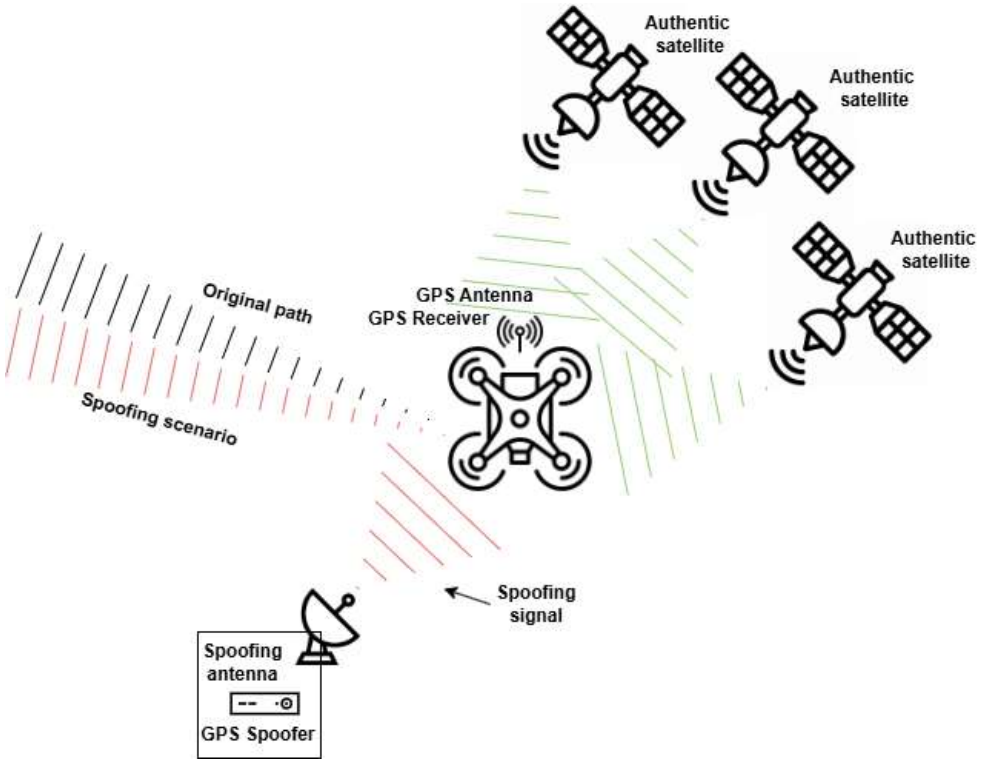


Рис. 1.2. Схематичне зображення спуфінг атаки

1.2. Методи захисту від GPS-спуфінгу

Для запобігання та виявлення GPS-спуфінгу розроблено низку підходів. Більшість з них базуються на моніторингу параметрів отримуваних супутникових сигналів і пошуку аномалій, що вказують на підробку. Такі методи зазвичай

вимагають спеціалізованого апаратного забезпечення у приймачі або доповнення системи GPS додатковими засобами. Основні технічні методи протидії спуфінгу [12] включають аналіз наступних характеристик сигналу GPS:

Рівень потужності сигналів. Аутентичні сигнали GPS від далеких супутників мають дуже малу потужність. Якщо приймач реєструє неприродно сильний сигнал, це може свідчити про близьке джерело підробленого випромінювання [13]. Практично реалізується вимірюванням RSSI (індикатора рівня сигналу) для кожного супутника та виявленням тих, що суттєво перевищують нормальні очікувані рівні.

Доплерівський зсув частоти. Через рух супутників відносно приймача справжні GPS-сигнали мають передбачуваний доплерівський зсув частоти. Підроблений наземний передавач зазвичай не в змозі точно відтворити ці зсуви. Тому аномалії, на кшталт однакового доплерівського зсуву для всіх супутників, є індикатором спуфінгу. Для такого аналізу потрібен високоточний приймач з еталонним генератором (кварцовим або атомним), здатний вимірювати частотні відхилення сигналів.

Час приходу сигналів (TOA). Сигнали від різних GPS-супутників доходять до антени приймача не одночасно (через різну відстань до кожного супутника). Якщо всі видимі сигнали приходять синхронно або з часовими невідповідностями, що не узгоджуються з геометрією супутників, це ознака можливої атаки. Спуфер зазвичай намагається синхронізувати підроблені сигнали для правдоподібності, але точно відтворити реальні часові затримки складно [10]. Спеціалізовані приймачі з інтегрованими атомними годинниками дозволяють виявляти ненормальні збіги часу приходу сигналів, непритаманні справжнім GPS .

Мультишляхове поширення. В реальних умовах частина сигналів GPS відбивається від поверхонь (землі, будівель), створюючи ефект багатопроменевого (мультишляхового) поширення. Приймачі розраховують положення, враховуючи типові затримки й ослаблення відбитих сигналів [14]. При спуфінгу картина мультишляхового сигналу може бути неприродною (наприклад, всі сигнали

надходять з одного напрямку без випадкових відбиттів). Аналіз аномальних параметрів мультишляховості також допомагає виявити підробку сигналу.

Синхронізація навігаційних повідомлень. Супутники GPS строго синхронізують час своїх передавачів за допомогою атомних годинників, що відображається в узгодженості навігаційних повідомлень. Підробник, не маючи таких ресурсів, не забезпечує ідеальної синхронності сигналії. Тому розбіжності у часовій синхронізації [5] між сигналами різних «супутників» (наприклад, невідповідність переданої мітки часу) можуть сигналізувати про спуфінг.

Окрім перелічених вище методів аналізу самого сигналу, досліджуються й інші способи підвищити стійкість GPS до спуфінгу. Зокрема, **криптографічна автентифікація сигналів** (вбудований захист на рівні навігаційного повідомлення) є перспективним напрямом: супутник може шифрувати чи електронно підписувати передані дані, а приймач – перевіряти їхню справжність за допомогою секретного ключа. Такий підхід вже реалізований у військових GPS (шифрований P(Y)-код, M-код) та впроваджується в нових цивільних системах (наприклад, у європейській Galileo планується відкрита служба автентифікації OSNMA).

На практиці для надійного захисту доцільно комбінувати кілька методів одночасно. Сучасні високоточні GNSS-приймачі, призначені для авіації чи військових БПЛА, як правило, інтегрують аналіз потужності сигналів, перевірку часових зсувів та доплерівського ефекту разом. Комплексний підхід підвищує ймовірність виявлення спроб спуфінгу навіть у складних сценаріях. Додатково можуть застосовуватися спеціальні антенні системи (фазовані антенні решітки) для оцінювання напрямку надходження сигналу та інтеграція GNSS з іншими сенсорами [15] (інерціальною навігацією, магнітометром, барометром) для перехресної перевірки координат.

1.3. INS-захист і інші системи

Одним із найефективніших способів виявлення спуфінгу та забезпечення стійкості навігації є інтеграція GPS із системою інерціальної навігації (INS). INS

складається з акселерометрів та гіроскопів, що відстежують прискорення і кутові швидкості апарата, дозволяючи обчислювати зміну його положення без зовнішніх сигналів. Бортовий комп'ютер постійно порівнює дані GPS із показниками INS (а також інших датчиків – магнітометрів, барометричних висотомірів тощо) [10]. У нормальних умовах обидві системи узгоджуються, а у випадку *спуфінгу* між ними виникає розбіжність: наприклад, координати, розраховані за GPS, змінюються таким чином, що не відповідають виміряній INS швидкості руху БПЛА [10, 11].

Таким чином, взаємний контроль GPS та INS дозволяє оперативно виявити підробку сигналів – якщо навігаційні рішення GPS не підтверджуються інерціальними даними, вони блокуються як підозрілі.

Інерціальна система виконує роль *резервного* джерела навігації на випадок компрометації GPS. У багатьох сучасних автопілотах БПЛА дані супутникової навігації вже поєднуються з INS для підвищення точності та надійності позиціонування. При виникненні сумнівів у достовірності GPS алгоритми навігації можуть покладатися більше на INS, зменшуючи вплив хибних координат. Хоча інерціальні датчики з часом накопичують похибку (дрейф), їх точності зазвичай достатньо, щоб *вчасно* виявити невідповідність з даними GPS та підтримувати керування польотом на короткий період.

Наприклад, INS здатна «утримати» апарат на прогнозованому шляху протягом декількох секунд чи хвилин після втрати достовірного GPS-сигналу, даючи системі час зреагувати.

Комбінування GNSS із бортовими сенсорами (інерціальними, магнітометричними, барометричними) нині розглядається як обов'язковий елемент захисту від спуфінгу для відповідальних застосувань.

Така багатоджерельна система навігації значно ускладнює завдання спуферу: навіть якщо атакувальник обманув GPS-приймач, відхилення буде зафіксовано за іншими вимірами, що активує аварійний режим чи сигнал тривоги [16].

1.4. Антенні системи та їх роль

Спеціалізовані антени відіграють важливу роль у захисті GNSS-приймачів від спуфінгу. Один із підходів – використання *фазованих антенних решіток*, зокрема технології **CRPA** (Controlled Reception Pattern Antennas) [4], які складаються з множини антенних елементів із керованою діаграмою спрямованості. Такі системи можуть визначати напрям, з якого надходить сигнал кожного супутника, та виявляти невідповідності. У нормі GPS-супутники розташовані на різних ділянках небосхилу, тому сигнали мають надходити з *різних напрямків*. Якщо ж приймач фіксує, що всі або кілька сигналів приходять з одного і того ж напрямку, це явна ознака спуфінгу. Виявивши таку аномалію, антенна решітка може програмно приглушити (“вирізати”) прийом із підозрілого напрямку або, навпаки, – посилити прийом сигналів з напрямків на справжні супутники, тим самим відфільтрувавши підробку.

Адаптивні антенні масиви широко застосовуються у військових GPS-приймачах, що дозволяє протидіяти як глушінню, так і спуфінгу сигналів. Для безпілотних апаратів, що працюють в умовах підвищеного ризику навігаційних атак, використання багатокомпонентних антен значно підвищує живучість навігації. Деякі сучасні антиспуфінгові системи здатні в режимі реального часу оцінити азимут та кут місця джерела сигналу і блокувати аномальні сигнали, не допускаючи їх до обчислення координат. Недоліком такого підходу є складність і вартість апаратури, а також збільшення маси і енергоспоживання БПЛА. Тому подібні антени частіше встановлюються на дорогих та критично важливих платформах. У випадку комерційних дронів, які оснащені переважно звичайними всеспрямованими антенами, акцент робиться на програмних методах виявлення спуфінгу.

1.5. Застосування машинного навчання

Останніми роками для виявлення GPS-спуфінгу все більшої популярності набувають методи машинного навчання (ML). Вони дозволяють автоматично знаходити приховані закономірності в телеметричних даних та характеристиках сигналів, які видають факт атаки, і тим самим доповнюють або замінюють традиційні

детектори на основі аналізу сигналу. Задачу виявлення спуфінгу можна розглядати як задачу класифікації: на основі набору вимірювань (вектора ознак) визначити, чи перебуває апарат під атакою, чи ні. Для цього застосовуються алгоритми *контролю цілісності* даних, навчені розпізнавати відхилення, притаманні спуфінгу.

Перші спроби використання ML для виявлення навігаційних атак на БПЛА з'явилися у другій половині 2010-х років. Наприклад, у роботі Shafiee та співавт. (2018) випробувано базові класифікатори – Байєсів та метод *k*-найближчих сусідів (k-NN) – для розпізнавання синхронного спуфінгу [2]; отримана точність становила близько 62% і 77% відповідно. Хоча такі результати далекі від ідеалу, вони продемонстрували принципову можливість автоматизованого розрізнення нормального та підробленого GPS-сигналу за статистичними ознаками.

Подальші дослідження застосували більш складні моделі. В одному з підходів для підвищення ефективності перед класифікацією виконували зменшення розмірності ознак методом головних компонент (PCA) [10]. Зокрема, порівнювалися: SVM-класифікатор із PCA-преобробкою, логістична регресія в поєднанні з автоенкодером на основі LSTM, а також логістична регресія зі стандартним автоенкодером. За результатами експериментів найбільш точним виявився підхід, що поєднує PCA та SVM, перевершивши інші моделі в усіх тестах. Ці досліди виконувалися на даних сенсорів БПЛА, зібраних під час нормальних автономних польотів та польотів під спуфінг-атакою, причому для перевірки стійкості моделей додатково додавали гаусовий шум і зменшували обсяг навчальних прикладів.

Альтернативним напрямом є використання *навчання без вчителя*, коли система намагається виявити аномалії без маркувань "атака/норма". Так, у роботі Morales-Ferre та ін. (2020) досліджено застосування методів пошуку винятків: PCA, кластеризації *k*-means та автоенкодера – до набору невідмічених даних GPS-сигналів [8]. Метою було навчитися ідентифікувати незвичні патерни сигналу, що можуть вказувати на спуфінг. Однак подібні підходи вимагають збору великого масиву параметрів сигналу та наявності високоточних приймачів (для фіксації дрібних

ефектів типу доплерівських зсувів, фазових різниць тощо). Відтак, повністю рішення без вчителя поки що менш придатні для простих комерційних дронів.

З огляду на те, що реальні атаки трапляються рідко, набув популярності підхід *one-class classification* – навчання моделі лише на прикладах нормальної поведінки з подальшим виявленням будь-яких відхилень як потенційної атаки. У дослідженні Semanjski та ін. (2018) було протестовано три алгоритми однокласової класифікації (Local Outlier Factor – LOF, автоенкодер та однокласовий SVM) для виявлення спуфінгу і глушіння на БПЛА [9]. Моделі тренувалися тільки на даних штатних польотів, а потім оцінювалися на здатність розпізнати “чужі” вибірки, відсутні в навчанні. Такий підхід вважається доцільним для цивільних застосувань, адже малоймовірно, що оператор дрона матиме у розпорядженні достатньо даних справжніх атак для навчання моделі. Експериментальні результати показали, що одно-класові моделі здатні фіксувати аномалії у телеметрії, які відповідають моментам навмисної підміни сигналу.

Сучасний етап розвитку характеризується залученням *глибоких нейронних мереж* для задачі детекції спуфінгу. Нейронні мережі можуть навчатися виявляти комплексні нелінійні залежності між різними сигналами і сенсорами. Прикладом є дослідження Jullian та ін. (2021), в якому розроблено багатошаровий перцептрон для аналізу параметрів польоту та GPS-сигналів; ця модель сигналізувала про спуфінг з точністю від ~83% (на стандартному наборі даних TEXBAT) до ~99.9% (на спеціально зібраному наборі телеметрії UAV у форматі MAVLink). Інші автори також повідомляють про високу (>90%) точність глибоких моделей у виявленні синхронних атак [10]. Огляд літератури свідчить, що більшість підходів на основі ML демонструють успішне розпізнавання спуфінгу з точністю в межах 93–99% на тестових вибірках.

Варто зазначити, що значна частина таких результатів отримана на симульованих або лабораторних даних. Лише поодинокі роботи використовували різнопланові набори даних для перевірки універсальності моделей. Наприклад, загальнодоступний набір записів спуфінгових сигналів TEXBAT часто

застосовується для навчання та перевірки алгоритмів [17]. Проте досі відсутнє єдине стандартне тестування та порівняння різних методів на спільному корпусі даних. Таким чином, використання машинного навчання у задачі виявлення GPS-спуфінгу є дуже перспективним напрямом, але потребує подальших досліджень – зокрема, збору більшої кількості реальних даних та розвитку методик оцінювання моделей у наближених до реальності умовах.

1.6. Аналіз існуючих рішень

Аналіз літератури показує, що для виявлення та нейтралізації GPS-спуфінгу на БПЛА розроблено різноманітні підходи – від апаратних методів аналізу сигналу до алгоритмів штучного інтелекту. Кожен з них має свої переваги та обмеження. Апаратні способи (аналіз потужності, доплерівського зсуву, часових характеристик тощо) є теоретично обґрунтованими і досить надійними, проте вони вимагають високої точності вимірювань та, як правило, спеціалізованого обладнання. Більше того, навіть найсучасніші апаратні детектори мають складнощі з швидким реагуванням на *повільні*, добре скоординовані спуфінг-атаки. Як зазначалося, при синхронному спуфінгу відсутні різкі зміни сигналу і немає еталонного положення для порівняння, тому виявити таку підміну в реальному часі надзвичайно важко. Засоби на кшталт INS-дублювання покращують ситуацію, але й вони не гарантують повного вирішення проблеми – інерціальна навігація схильна до накопичення похибки і може лише тимчасово “прикрити” систему при відмові GPS. Антенні системи (наприклад, CRPA) забезпечують просторову фільтрацію сигналів і можуть фізично блокувати спуфінг, але такі рішення є дорогими, громіздкими і малодоступними для масових БПЛА.

Застосування методів машинного навчання та інтелектуального аналізу даних відкриває нові можливості для створення відносно простих і водночас ефективних засобів виявлення спуфінгу. Програмні ML-детектори можуть бути впроваджені на рівні бортового комп’ютера або наземної станції без апаратних модифікацій дрона. В оглянутих працях продемонстровано, що такі моделі здатні досягати високої точності

– понад 90% при виявленні атак у контрольованих умовах. Особливо перспективними є нейронні мережі, які навчаються на реальних сценаріях атак.

Алгоритми на основі ML мають і свої виклики. Ключовою проблемою залишається обмеженість даних для навчання і тестування моделей. Більшість доступних наборів отримані в експериментальних або симуляційних умовах, що не повною мірою відображають усе різноманіття реальних атак. Існуючі рішення часто перевірялися лише на окремих сценаріях (наприклад, синхронне підроблення сигналу без впливу перешкод), і бракує уніфікованої методики порівняння їх між собою. До того ж, моделі глибокого навчання поводяться як «чорний ящик», і гарантувати їхню надійність поза межами навчальної вибірки складно. Таким чином, жодне з наведених рішень поки що не можна вважати універсальним щитом від GPS-спуфінгу.

Підсумовуючи, для забезпечення надійного захисту БПЛА доцільно поєднувати кілька підходів: використовувати інерціальну навігацію та аналіз параметрів сигналу для первинного виявлення, а доповнювати їх інтелектуальними алгоритмами, що здатні розпізнавати нетипову поведінку системи. Розробка високоточних методів виявлення спуфінгу на основі машинного навчання є актуальним напрямом досліджень. У наступних розділах представлено рішення, яке поєднує переваги розглянутих підходів – програмний детектор на базі MLP, здатний працювати з реальними сигналами GPS і виявляти спуфінг-атаки з високою достовірністю.

Висновки до розділу

У першому розділі було проведено глибокий аналітичний огляд літератури та сучасних досліджень, присвячених проблемі захисту безпілотних літальних апаратів (БПЛА) від атак GPS-спуфінгу. Аналіз показав, що GPS-спуфінг є однією з найнебезпечніших форм навігаційних атак на БПЛА, яка дозволяє зловмиснику непомітно маніпулювати маршрутом апарата через підміну координат високопотужними фальшивими сигналами.

Розглянуто різновиди спуфінг-атак — синхронні та асинхронні, — та охарактеризовано механізми їх здійснення. Було проаналізовано апаратні та

програмні методи виявлення спуфінгу: моніторинг потужності сигналу, доплерівського зсуву, часу прибуття сигналів, а також мультишляхового поширення та синхронізації навігаційних повідомлень. Встановлено, що комбінація кількох методів суттєво підвищує надійність виявлення атак.

Особливу увагу приділено перспективі використання систем інерціальної навігації (INS) як додаткового захисного механізму, здатного зафіксувати відхилення в координатах навіть у разі успішного спуфінгу. Крім того, проаналізовано роль антенних рішень (зокрема фазованих решіток CRPA) у виявленні аномалій напрямку сигналів.

Окремим напрямом виявлення навігаційних атак стало використання методів машинного навчання, зокрема багатошарових перцептронів (MLP), SVM, автоенкодерів та методів класифікації за одним класом. Експерименти доводять ефективність ML-підходів, здатних досягати точності понад 90%, навіть у складних умовах. Проте ці методи часто залежать від якості та репрезентативності навчальних даних, що обмежує їхню практичну масштабованість.

РОЗДІЛ 2

Системний аналіз та обґрунтування проблемної ситуації

2.1 Системний аналіз об’єкта дослідження та предметної області

2.1.1 Визначення проєктованої системи та її середовища

Проєктована система призначена для **виявлення атак GPS-спуфінгу на БПЛА** з використанням багатошарового перцептрону (MLP). Для такої системи можливі два основні варіанти архітектури та реалізації:

1. Повністю автономна система (на борту БПЛА).

Система функціонує як бортовий компонент БПЛА, що безпосередньо отримує дані від навігаційних сенсорів (GPS-приймача або програмно-визначеного радіоприймача GNSS) і обробляє їх в реальному часі на самому літальному апараті. Усі етапи роботи — від збору сирих сигналів навігаційної системи, обчислення потрібних характеристик (наприклад, псевдовідстаней до супутників, потужності сигналів, зсувів по Доплеру тощо) до класифікації стану (нормальний режим або атака спуфінгу) за допомогою навченої моделі MLP — виконуються локально на борту. Результат (виявлено атаку чи ні) використовується самим БПЛА для прийняття рішень щодо безпечної навігації та може відображатися оператору через телеметрію.

Переваги:

- Незалежність від зв’язку. Система продовжує працювати в будь-яких умовах, навіть за повної відсутності каналу зв’язку з наземною станцією або Інтернетом. Це критично важливо для безпілотників, що діють у віддалених районах або в умовах радіоелектронної боротьби, де мережевий зв’язок нестабільний або відсутній.
- **Мінімальна затримка реагування.** Обробка даних на борту відбувається без затримок, пов’язаних з передачею інформації в зовнішні центри й назад. Це забезпечує максимально швидке виявлення атаки спуфінгу і негайне сповіщення або реакцію автопілота, що особливо важливо для систем реального часу.

- **Безпека** даних**.** Усі навігаційні дані обробляються локально, без передачі по радіоканалах, що знижує ризик перехоплення або підробки цих даних зловмисниками. Пасивний моніторинг сигналів не створює додаткових випромінювань, тому не демаскує БПЛА і не погіршує власну навігацію.
- **Автономність і стійкість.** Система на борту продовжує функціонувати навіть у випадку спроб глушіння зв'язку з оператором. БПЛА може самостійно вжити заходів (наприклад, перейти на резервну навігацію) при виявленні атаки, не очікуючи зовнішніх команд.

Недоліки:

- **Обмеження бортових ресурсів.** Виконання алгоритмів MLP в реальному часі вимагає достатньо продуктивного та енергоефективного обчислювального модуля на БПЛА. Це може збільшити вагу, енергоспоживання і вартість апарату, адже типові автопілоти мають обмежені ресурси.
- **Складність оновлення.** Оновлення програмного забезпечення або перев навчання моделі потребує доступу до БПЛА (прямого підключення або через захищений канал зв'язку). В польових умовах це може бути ускладнено, тому необхідно передбачити зручний механізм оновлення моделі на борту.
- **Масштабованість обчислень.** Бортова апаратура має фіксовані обчислювальні можливості. Обробка великого обсягу сигналів від багатьох каналів GPS чи використання складніших моделей може бути обмежена продуктивністю конкретного пристрою.
- **Обслуговування.** У випадку апаратного збою ремонт можливий лише після повернення БПЛА, що може тимчасово вивести систему з експлуатації. Також додатковий бортовий модуль потребує живлення від акумулятора БПЛА, що може дещо зменшити час польоту.

2. Хмарно-орієнтована система (віддалене виявлення).

У цьому підході БПЛА оснащується мінімально необхідним модулем збору даних, який передає інформацію на наземний або хмарний сервер для обробки.

Навігаційні дані (сирі GPS-сигнали або обчислені параметри) шифруються і надсилаються через канал зв'язку (телеметрія, мобільний інтернет, супутниковий зв'язок) до віддаленого обчислювального центру. На сервері виконується основна обробка: розрахунок ознак, виконання моделі MLP і прийняття рішення про наявність атаки. Результат (сповіщення про спуфінг) негайно передається назад на безпілотник та оператору. БПЛА на основі цього результату може змінити режим польоту (наприклад, перейти у аварійний режим навігації) або повідомити оператора для втручання.

Переваги:

- **Потужні** обчислювальні **ресурси.** Віддалений сервер може мати практично необмежену обчислювальну потужність, що дозволяє застосовувати більш складні алгоритми аналізу сигналів або більш глибокі нейронні мережі, обробляти дані від декількох БПЛА одночасно, тощо.
- Централізоване **оновлення.** Модель MLP і програмне забезпечення оновлюються на сервері, тому всі підключені БПЛА автоматично отримують користь від найновішої версії детектора без необхідності вручну оновлювати кожен апарат.
- **Менші вимоги до бортового обладнання.** Сам БПЛА може бути оснащений простішим і легшим модулем, оскільки основні обчислення виконуються віддалено. Це знижує навантаження на акумулятор і потенційно зменшує вагу корисного навантаження.
- Агрегація **даних.** Дані від багатьох польотів і апаратів можуть збиратися в єдиному центрі для подальшого аналізу, донавчання моделі і покращення алгоритмів виявлення. Це створює можливості для постійного підвищення точності детекції на основі великої вибірки атак і нормальних польотів.

Недоліки:

- Критична **залежність від мережі.** Система не працюватиме належним чином без стабільного каналу зв'язку. В умовах, коли БПЛА діє в зоні активного глушіння чи немає покриття мережі, передача даних на сервер може бути неможливою. Для критично важливих місій (наприклад, військових) такий варіант ненадійний.
- Затримки **передачі.** Навіть при швидкому зв'язку передача даних до сервера і зворотного результату займає певний час (latency). Для реального часу навіть затримка в частки секунди може бути суттєвою, а при повільнішому зв'язку затримки можуть стати неприйнятно великими, що знижує ефективність раннього виявлення.
- Питання **безпеки та конфіденційності.** Передача навіть зашифрованих навігаційних даних через мережу створює додаткову поверхню атаки. Зловмисник теоретично може спробувати перехопити або змінити передані дані, атакувати сервер або вивідати результат детекції. Безпека хмарної інфраструктури і каналу зв'язку стає критичною частиною всієї системи.
- Операційні **витрати.** Постійне використання серверів або хмарних сервісів потребує фінансових витрат (оплата серверного часу, трафіку). Для великого парку БПЛА це може стати відчутним фактором.
- Складvність **інфраструктури.** Необхідність розгортання і підтримки серверного рішення, забезпечення відмовостійкості, кібербезпеки та масштабування інфраструктури ускладнює проєкт у порівнянні з автономним підходом.

Обґрунтування вибору архітектури. Для системи виявлення GPS-спуфінг атак на БПЛА в рамках даного проєкту обрано **перший варіант – повністю автономну бортову систему**. Такий вибір зумовлений наступними причинами:

Надійність в умовах застосування.

БПЛА часто використовуються в умовах, де **стабільний зв'язок не гарантований** – наприклад, у віддалених районах або в зоні активних радіоперешкод. Залежність від мережі робить хмарне рішення непридатним для сценарію, де противник може одночасно здійснювати глушіння сигналу управління. Автономна система гарантує, що детектор продовжить працювати навіть у повній ізоляції літального апарату.

Безпека даних і операції.

Автономна система не передає чутливі навігаційні дані по радіоканалах, що усуває ризики їх перехоплення чи підміни під час передачі. Це особливо важливо для військового застосування БПЛА, де будь-який витік інформації чи втручання в канали зв'язку може мати важкі наслідки.

Вимоги реального часу.

Для безпеки польоту важливо **миттєво** виявити факт спуфінгу та відреагувати, аби запобігти критичним відхиленням траєкторії. Будь-які додаткові затримки, пов'язані з передачею даних на сервер і назад, є небажаними. Бортова обробка забезпечує мінімально можливий час реакції – оператор отримує сповіщення, а автопілот може скоригувати дії практично негайно після початку атаки.

Технологічна можливість реалізації.

Сучасні апаратні засоби (спеціалізовані процесори для ШІ, одноплатні комп'ютери, модулі GPU/TPU для дронів) дозволяють розгорнути достатньо потужні нейронні моделі **безпосередньо на борту** компактного безпілотника. Хоча це висуває певні вимоги до обладнання, розвиток технологій Edge AI робить такий підхід цілком реальним. Переваги автономності при цьому переважують складність реалізації.

Враховуючи наведені аргументи, концепція проєктованої системи виявлення GPS-спуфінгу базується на автономній бортовій реалізації. Така система інтегрується у навігаційну підсистему БПЛА і працює в режимі реального часу незалежно від зовнішніх сервісів.

Основні компоненти системи:

- Навігаційний сенсор (GPS-модуль).

Приймач глобальної навігаційної системи, встановлений на БПЛА, який забезпечує отримання сигналів GPS від супутників. Може бути стандартним GNSS-приймачем автопілота, що надає дані про місцеположення та інші параметри, або додатковим програмно-визначеним радіоприймачем, налаштованим на прийом і аналіз GPS-сигналів для детальнішого збору характеристик.

- Обчислювальний модуль.

Бортовий мікропроцесорний пристрій або одноплатний комп'ютер (наприклад, з підтримкою технологій Edge AI), на якому розгорнуто програмне забезпечення детектування. Цей модуль здійснює обробку вхідних навігаційних даних, виконує алгоритми MLP-класифікації та приймає рішення про наявність атаки.

- Програмне забезпечення детектування.

Алгоритмічна частина, що включає засоби попередньої обробки сигналів (фільтрація, обчислення ознак), безпосередньо модель багатошарового перцептрону (завантажену навченою) та логіку ухвалення рішення. ПЗ також опікується збором статистики та веденням журналу спрацювань, що може стати у пригоді для подальшого аналізу.

- Модуль пам'яті.

Накопичувач для зберігання прошивки, нейронної моделі, конфігурацій та буферу даних. Забезпечує можливість оновлення моделі і збереження записів про зафіксовані інциденти (для пост-аналізу після польоту).

- Комунікаційний інтерфейс.

Канал зв'язку між БПЛА та наземною станцією (радіомodem, телеметрія), що використовується для передачі сповіщень оператору. Хоча сам детектор не залежить від мережі, наявність інтерфейсу дозволяє інтегрувати результати в систему керування польотом: відобразити попередження на екрані оператора, передати сигнал тривоги або автоматично змінити завдання польоту.

- Інтеграція з автопілотом.

Логічний компонент, що забезпечує узгоджену роботу детектора з навігаційним обладнанням БПЛА. У разі виявлення атаки може переводити автопілот у заздалегідь визначений безпечний режим (наприклад, режим утримання висоти та курсу за даними інерціальної системи) або активувати процедуру повернення додому за альтернативними засобами навігації.

- Джерело живлення.

Акумуляторна батарея БПЛА, від якої живиться і сам детектор. Додаткове споживання енергії має враховуватися, щоб не суттєво скоротити час польоту. Система повинна бути енергоефективною.

Зовнішнє середовище системи: все, що знаходиться поза бортовим обладнанням БПЛА, але впливає на роботу системи або взаємодіє з нею:

- Супутникова навігаційна система (GPS).

Сукупність навколоземних супутників, що передають автентичні радіосигнали з навігаційною інформацією. Вони є основним джерелом вхідних даних для системи – БПЛА приймає сигнали від кількох GPS-супутників, які містять часові мітки і координати. Якість та сила цих сигналів можуть змінюватися залежно від положення БПЛА, кількості видимих супутників, атмосферних умов тощо.

- Зловмисник **з** генератором **спуфінгу.**

Атакуючий пристрій або установка, що випромінює підроблені GPS-сигнали. Цей пристрій перебуває в зоні досяжності БПЛА і випускає радіосигнали на тій же частоті, що й GPS, але з навмисно модифікованим вмістом (фальшивими координатами/часом) і підвищеною потужністю. Зовнішня атака спуфінгу є безпосереднім об’єктом виявлення: вона впливає на сигнали, які отримує навігаційний сенсор, і може вносити аномалії, що система повинна розпізнати.

- **Інші джерела радіоперешкод.**

Оточуюче середовище може містити перешкоди для GPS-сигналів: навмисні глушники (jammer’и), шум від електронного обладнання, відбиття сигналів від будівель (ефект багатопроменевого розповсюдження). Ці фактори можуть викликати

нестабільність або спотворення навігаційних даних, що ускладнює завдання детектування, оскільки система повинна відрізняти легітимні природні аномалії від дій саме спуфера.

- Наземна станція управління.

Людина-оператор і пов'язане обладнання (комп'ютер з програмним забезпеченням керування дроном, приймач телеметрії), які взаємодіють з БПЛА під час виконання місії. Хоча оператор не є безпосередньою частиною автоматизованого процесу детекції, **він отримує вихідну інформацію** від системи (сповіщення про атаку) і може приймати подальші рішення. Наземна станція також слугує середовищем для оновлення прошивки або моделі детектора між польотами.

Функціональні вимоги до системи:

- Система повинна отримувати навігаційні дані від бортового GPS-приймача БПЛА. Як мінімум, необхідно зчитувати поточні параметри сигналів: псевдовідстані до видимих супутників, значення потужності сигналу (рівень RSSI), частотні зсуви (Доплерівські зміни) та інші величини, які можуть слугувати ознаками атаки.
- Система повинна виконувати формування ознак на основі отриманих даних. На вході MLP подається вектор характеристик, обчислених із сирих вимірювань (наприклад, статистичні показники по псевдовідстанях і потужності сигналу за певний інтервал часу, кореляційні показники між каналами тощо) згідно з обраним алгоритмом детекції.
- Система повинна здійснювати класифікацію стану навігації за допомогою завантаженої моделі нейронної мережі (MLP). Класифікація полягає у визначенні належності ситуації до одного з двох класів: "нормальний режим" (відсутня атака) або "спуфінг-атака" (виявлено втручання). Модель MLP має бути попередньо навчена на релевантних даних і збережена в пам'яті БПЛА.
- Система повинна своєчасно та чітко сповіщати про виявлення атаки. У випадку, якщо класифікатор визначив, що відбувається спуфінг, пристрій

повинен негайно сформувати сигнал тривоги: відправити повідомлення на наземну станцію оператору (через телеметрію) та/або подати сигнал на автопілот для ініціації захисних дій. Сповіщення має бути досить інформативним (наприклад, індикатор на екрані “GPS SPOOFING DETECTED”) і надійним, щоб оператор не пропустив критичне попередження.

- Система повинна працювати в безперервному режимі під час всього польоту. Моніторинг GPS-сигналу та перевірка на спуфінг виконується регулярно з певною частотою (наприклад, кілька разів на секунду) або за подіями, щоб забезпечити якнайшвидшу реакцію на можливу атаку.
- Система повинна мати можливість оновлення моделі MLP та програмного забезпечення. Це дозволить покращувати алгоритми детекції у майбутньому (наприклад, після отримання нових даних про атаки) без заміни всього обладнання. Оновлення може здійснюватися шляхом перепрошивки пристрою або через завантаження нових параметрів моделі з наземної станції перед вильотом, за умови дотримання заходів безпеки при такому оновленні.
- Система повинна не впливати на основну навігацію в нормальному режимі. Вона працює пасивно і не повинна вносити затримки чи помилки у розрахунок положення БПЛА, коли атаки немає. Це означає, що взаємодія детектора з автопілотом має бути спроектована таким чином, щоб у разі відсутності тривоги навігація відбувалася як звичайно, а у разі помилкового спрацювання – система давала оператору можливість перевірити і підтвердити ситуацію, мінімізуючи ризик невиправданих маневрів.

Нефункціональні вимоги до системи:

- Точність виявлення: алгоритм MLP має забезпечувати високі показники правильного розпізнавання. Цільові метрики – Accuracy не менше 93%, високий Precision та Recall для класу “атака” (щоб мінімізувати як помилкові тривоги, так і пропущені атаки). Бажано досягти F1-міри > 0,93 на репрезентативних тестових

даних. Таким чином, система буде достатньо надійною для практичного використання.

- Низький рівень хибних спрацювань: FPR (False Positive Rate) і FNR (False Negative Rate) мають бути якомога меншими. Хибне спрацьовування (коли система помилково сигналізує атаку) може призводити до непотрібного втручання в політ і зниження довіри оператора, а хибне невиявлення (коли атака не розпізнана) — до небезпечних інцидентів. Тому система повинна бути збалансована таким чином, щоб впевнено відрізняти реальні атаки від нормальних умов, включаючи складні, але штатні ситуації (наприклад, різкі маневри БПЛА чи часткове затінення супутників).
- Надійність роботи: пристрій виявлення має стабільно функціонувати в умовах, характерних для експлуатації безпілотників. Вібрації від двигунів, перепади температури, вологість чи пил (залежно від висоти та середовища польоту) не повинні призводити до відмови роботи електроніки. Бажано, щоб система була виконана у відповідності до вимог міцності та захисту (наприклад, відповідний клас IP для захисту від пилу та вологи, стійкість до вібрацій авіоніки).
- Зручність інтеграції та використання: інтерфейс сповіщення має бути інтуїтивно зрозумілим для оператора. Оскільки типовий користувач взаємодіє із системою через наземну станцію, повідомлення про спуфінг повинно легко розпізнаватися (наприклад, звуковий сигнал та помітне візуальне попередження в програмі управління дроном). Налаштування системи (якщо потрібне, наприклад, пороги чутливості) має бути мінімально складним. Також інтеграція детектора в існуючі програмно-апаратні комплекси БПЛА повинна бути простою, без суттєвої переробки останніх – це підвищить ймовірність практичного застосування.
- Безпека та стійкість до кібер-втручань: система сама по собі не повинна стати новим каналом атаки. Необхідно забезпечити захищеність від

несанкціонованого оновлення моделі або ПЗ (щоб противник не зміг відключити детектор або навчити його ігнорувати атаку). Також, якщо використовується канал зв'язку для передачі оповіщення, цей канал має бути шифрований і автентифікований.

- Апаратні обмеження: рішення повинно бути компактним і легким, щоб не порушити аеродинаміку і вантажопідйомність дрона. Бажано, щоб маса детектора та споживання енергії були мінімальними (сучасні мікрокомп'ютери дозволяють виконувати такі задачі, споживаючи кілька ватт енергії). Система має працювати від бортового живлення протягом усього типового часу місії БПЛА без необхідності додаткового акумулятора.
- Продуктивність (час реакції): система повинна працювати в реальному часі. Затримка від моменту початку атаки до моменту фіксації і сповіщення має бути мінімальною (бажано менше 1–2 секунд). Це включає час на зчитування даних, обчислення ознак, проходження через MLP і генерування повідомлення. Така швидкодія необхідна, щоб БПЛА не встиг значно відхилитися від курсу до того, як буде вжито заходів.
- Обслуговуваність: повинна передбачатися можливість діагностики та технічного обслуговування системи. Логи роботи (дані про виявлені атаки, параметри сигналів під час них) мають зберігатися для експертного аналізу. Пристрій слід проектувати модульно, щоб у разі виходу з ладу будь-якого компонента (наприклад, сенсора чи обчислювального модуля) його можна було замінити. Оновлення програмної частини має бути можливим без повного демонтажу (наприклад, через порт підключення до ПК або бездротово на землі).

2.1.2 Діаграма варіантів використання

На рис. 2.1.2 зображено діаграму варіантів використання проектованої системи виявлення GPS-спуфінгу на БПЛА. У діаграмі відображено основних акторів та сценарії взаємодії системи із зовнішніми суб'єктами.

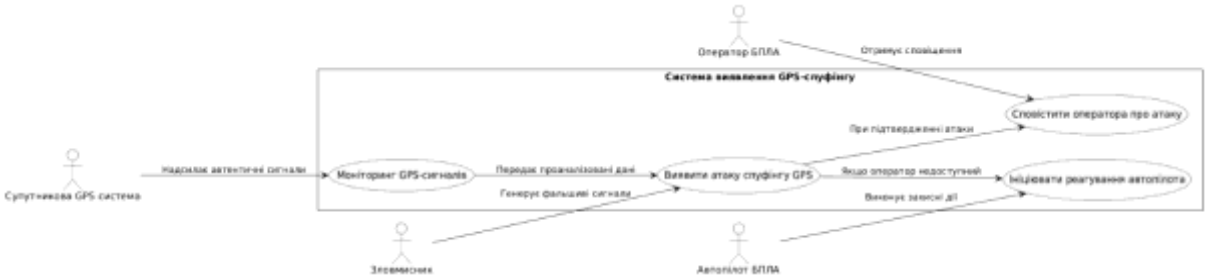


Рис. 2.1 Діаграма варіантів використання

Актори:

- **Оператор БПЛА.** Людина, яка керує безпілотником або спостерігає за його польотом через наземну станцію. Оператор взаємодіє із системою опосередковано: отримує сповіщення про атаки та, за потреби, приймає рішення (наприклад, переймає керування або ініціює повернення додому). Оператор зацікавлений у безпечному виконанні місії, тому є кінцевим отримувачем інформації від системи.
- **Супутникова GPS система.** Зовнішня система, що передає сигнали навігації. Формально вона діє як джерело даних для БПЛА: у варіантах використання її роль полягає в надсиланні потоків автентичних сигналів, які приймає навігаційний модуль. Хоча супутники не є “користувачем” системи, на діаграмі вони показані як актор, оскільки взаємодіють із нею (через надсилання сигналів, які система аналізує).
- **Зловмисник (атакуючий).** Це зовнішній актор, що ініціює негативний сценарій. Атакуючий не взаємодіє із системою напряму в штатному режимі, але його дії (випромінювання spoofing-сигналів) запускають відповідний варіант використання системи – сценарій виявлення атаки. На діаграмі зловмисник зображений окремо, щоб показати, що система розглядає його дії (спробу обману) як тригер для своїх функцій.

Основний варіант використання – “Виявити атаку спуфінгу GPS”. Цей випадок спрацьовує під час польоту БПЛА, коли присутній вплив зловмисника. Сценарій протікає наступним чином: спочатку БПЛА в нормальному режимі отримує сигнали від **GPS-супутників** і визначає своє місцезнаходження (цей процес

відбувається постійно). Система детектування паралельно **моніторить отримані GPS-дані**. Оператор задає політне завдання і очікує, що апарат його виконає, не втручаючись у роботу навігації, якщо все гаразд.

Коли зловмисник починає атаку спуфінгу, БПЛА приймає і фальшиві сигнали теж. На стороні системи детектування це проявляється у зміні певних параметрів (наприклад, **аномальне зростання потужності сигналу** або невідповідності в даних псевдовідстаней). Сценарій “Виявити атаку” переходить до активної фази: **система аналізує ознаки** і, якщо критерії атаки виконуються, **ідентифікує ситуацію як спуфінг**.

Далі система **ініціює сповіщення оператора** (в варіанті використання це може розглядатись як окремий підваріант або крок основного сценарію). Оператор отримує повідомлення про атаку. Одночасно система може повідомити **автопілот БПЛА** (внутрішній актор) для запуску процедури реагування: наприклад, перемикання навігації на інерційну або зупинки виконання поточного завдання до з’ясування обставин.

Альтернативні потоки: Якщо спуфінг-атака **не підтверджується** (система не виявляє критеріїв атаки), то варіант використання завершується без тривоги: БПЛА продовжує звичайний політ, оператору не надходять сигнали про небезпеку. Система повертається до фонової стадії моніторингу. Якщо ж атака підтверджена, але **канал зв’язку з оператором відсутній**, система все одно продовжує дії з боку автопілота (тобто автономно намагається забезпечити безпеку польоту, наприклад, записує точку від’єднання від маршруту і переходить в режим утримання).

Таким чином, діаграма варіантів використання демонструє, що **в нормальних умовах** взаємодія відбувається між оператором, БПЛА та GPS (звичайний політ), а **у випадку атаки** до цього процесу “приєднується” підсистема детектування, яка вступає у гру для виявлення та сповіщення про інцидент. Оператор у результаті має можливість втрутитись (наприклад, екстрено посадити дрон), а безпілотник – убезпечити себе.

2.1.3 Діаграма діяльності

Для наочного відображення послідовності роботи компонентів системи розроблено діаграму діяльності (рис. 2.2). Вона моделює основні кроки процесу виявлення GPS-спуфінгу під час польоту БПЛА, включаючи розгалуження сценарію залежно від наявності атаки.

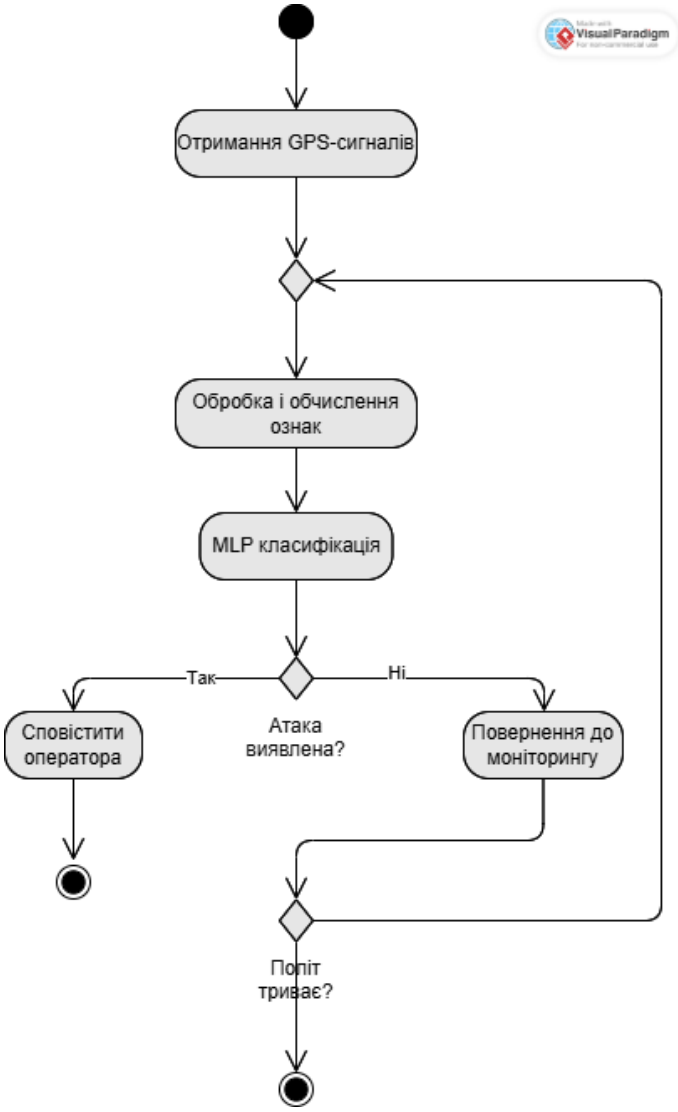


Рис. 2.2 Діаграма діяльності

Опис процесу (рис. 2.2):

- Початок: БПЛА розпочинає політ згідно з заданим маршрутом. Система детектування запускається одночасно з навігаційною системою дрона і переходить у режим моніторингу.

- Отримання даних GPS: На першому кроці активності бортовий GPS-приймач безперервно отримує сигнали від кількох GPS-супутників. На кожній ітерації (цикл опитування) оновлюються навігаційні дані: координати, висота, швидкість, а також технічні параметри сигналів (значення кореляції, потужність і якість прийому для кожного супутника тощо). Ці дані надходять до модуля детектування.
- Попередня обробка та обчислення ознак: Система обробляє сирі дані, щоб виділити інформативні ознаки для моделі MLP. Наприклад, розраховуються різниці між прогнозованим положенням БПЛА (за інерційними датчиками чи попереднім GPS-відліком) і поточними координатами, корелюються параметри сигналів від різних супутників між собою, обчислюються середні або дисперсії псевдовідстаней за останній період. Цей крок може включати фільтрацію шуму та агрегацію даних у часовому вікні.
- MLP-класифікація: Підготовлений вектор ознак подається на вхід моделі багатошарового перцептрону. Відбувається пропуск даних через шари нейронної мережі. На виході MLP генерується певна ймовірність або клас: “спуфінг” чи “ні”. У діаграмі діяльності це відображено як блок «Класифікувати стан навігації (MLP)».
- Розгалуження рішення: Результат класифікації перевіряється умовним блоком «Атака виявлена?». Якщо вихід MLP відповідає нормальному стану (наприклад, ймовірність атаки нижче порогу), сценарій переходить на гілку [Ні], і відбувається повернення до моніторингу: процес повторюється знову з кроку отримання даних GPS (цикл). Якщо ж виявлено атаку (гілка [Так]), тобто MLP класифікував стан як спуфінг з високою впевненістю, то активуються дії реагування.
- Сповіщення та реагування: У разі детекції атаки система негайно виконує дві паралельні піддіяльності. Перша — сповістити оператора: формує повідомлення про виявлений спуфінг і передає його на наземну станцію. Друга

— сповістити автопілот: внутрішньосистемний сигнал переключає режим роботи навігації. На діаграмі ці дії можуть бути показані або як послідовні кроки, або як паралельні (за допомогою розділення потоку), оскільки сповіщення оператора та безпілотника відбуваються майже одночасно. Автопілот, отримавши сигнал тривоги, може виконати заздалегідь визначений алгоритм (наприклад, ігнорувати подальші команди GPS і продовжити політ за останніми достовірними даними чи за інерційною системою, або стабілізуватися й чекати втручання оператора).

- Завершення сценарію: Після вжитих заходів подальший розвиток подій залежить від дій оператора та ситуації. Якщо атака підтверджена і триває, оператор може прийняти рішення посадити БПЛА вручну або вийти з небезпечної зони. Якщо атака припинилася (зловмисник відключив передавач) і дрон зберіг контроль, можливе повернення до нормального режиму. На самій діаграмі діяльності сценарій “виявлення атаки” можна вважати завершеним після сповіщення; далі система може увійти в цикл очікування закінчення місії або відключення живлення.

Діаграма діяльності демонструє, що **в нормальному режимі** більшість часу витрачається на повторювані кроки збору та аналізу даних, і тільки у випадку аномалії виконується розгалуження на гілку тривоги. Такий підхід відповідає реальним вимогам: детектор працює постійно у фоновому режимі, мінімально впливаючи на основні функції, але здатний швидко переключитися на активні дії при виникненні загрози.

2.1.4 Діаграма розгортання

На рис. 2.4 представлена діаграма розгортання компонентів системи в апаратно-програмному середовищі. Вона показує, як системні компоненти розміщено на фізичних вузлах, та відображає зв’язки між ними і зовнішніми системами.

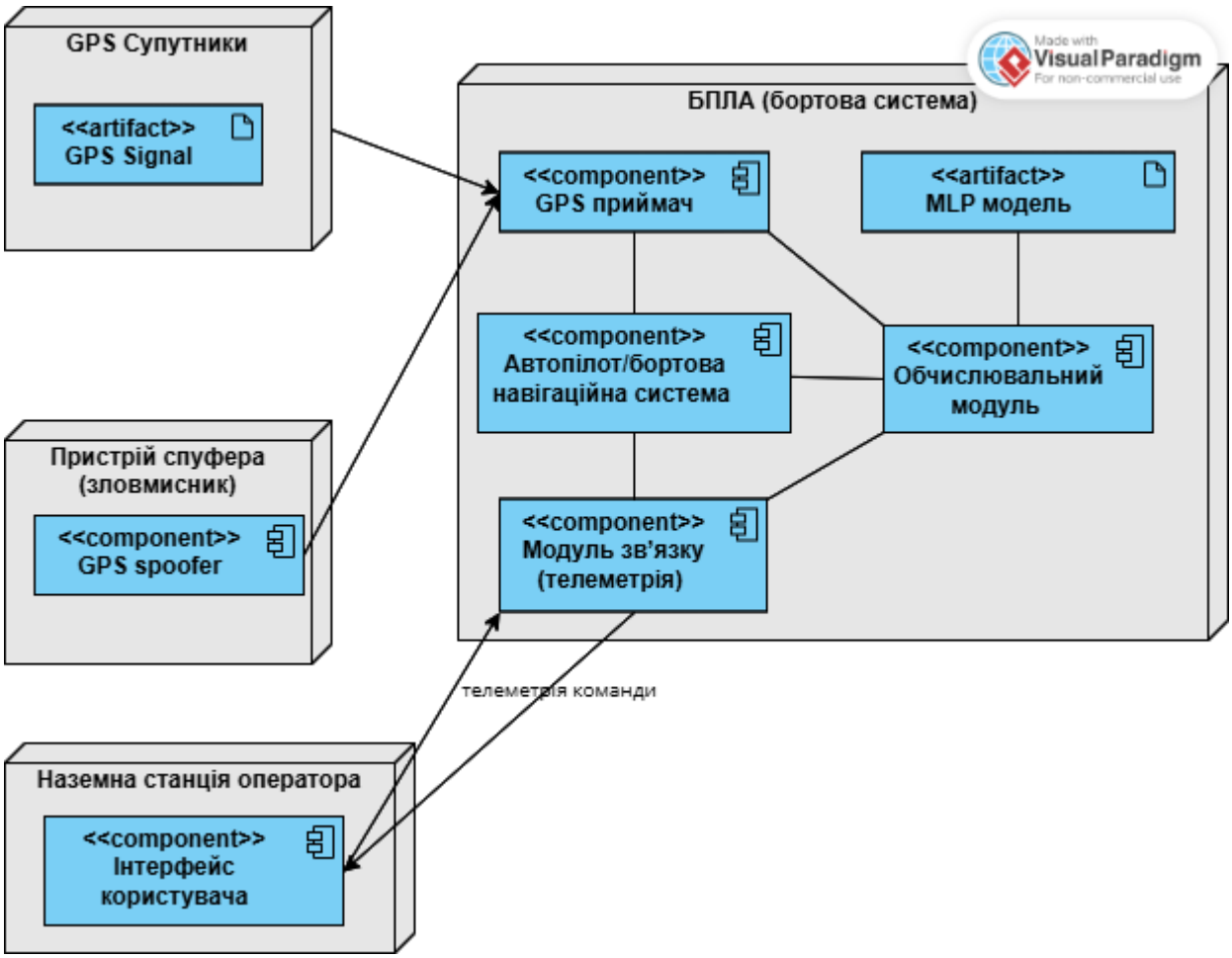


Рис 2.4 Діаграма розгортання

Основні вузли та компоненти розгортання:

- Вузол “БПЛА (бортова система)” – це фізична платформа безпілотного літального апарату, на якій встановлені всі складові системи детектування. Усередині цього вузла на діаграмі розміщені:
  - *GPS-приймач* – модуль, що приймає сигнали від супутників. Він може бути інтегрований в автопілот або представлений окремим пристроєм (наприклад, антена + приймач GNSS). На діаграмі показаний як компонент, з’єднаний стрілками з вузлом “GPS Супутники” (джерело сигналів) та з обчислювальним модулем, якому передає дані.
  - *Обчислювальний модуль (контролер)* – компонент, що виконує програму детектування. Він отримує на вході дані від GPS-приймача (показано зв’язком), а також може отримувати дані від інших датчиків БПЛА (штрихпунктиром можна відмітити зв’язок з інерційною системою, якщо та

використовується додатково). Всередині цього модуля логічно виконуються операції MLP-класифікації.

- *MLP-модель* – показана як артефакт, розгорнутий на обчислювальному модулі (файл із збереженими вагами нейронної мережі).
- *Модуль зв'язку (телеметрія)* – компонент, що забезпечує радіообмін з наземною станцією. Він підключений до обчислювального модуля та автопілота, приймає від них повідомлення для передачі оператору і навпаки. На діаграмі зв'язок від цього модуля веде до вузла "Наземна станція оператора".
- *Автопілот/бортова навігаційна система* – компонент, що керує рухом БПЛА. Він отримує дані від GPS-приймача (для виконання навігації) та може отримувати сигнал тривоги від обчислювального модуля (показано зв'язком "Alert" або подібним). Автопілот також пов'язаний з модулем зв'язку, щоб передавати телеметрію оператору.

- **Вузол "GPS Супутники"** – умовний вузол, що представляє глобальну інфраструктуру GPS. Звідси виходять потоки сигналів до GPS-приймача на БПЛА. Зв'язок односторонній: супутники транслюють дані, а приймач приймає (відображено на діаграмі стрілкою від "GPS Супутники" до компонента приймача на БПЛА).
- **Вузол "Пристрій** спуфера **(зловмисник)"** – зовнішній вузол, який моделює атакувальний пристрій. Він містить компонент **"GPS Spoofer"**, що генерує підроблені навігаційні сигнали. На діаграмі від цього вузла також виходять стрілки до GPS-приймача БПЛА (оскільки спуфер випромінює радіохвилі, які приймаються антеною дрона). Цей вузол не має зворотного зв'язку – припускається, що спуфер **пасивно** не приймає інформацію від дрона (хоча на практиці для синхронного спуфінгу він може слухати сигнали від БПЛА, це опускається, бо не є частиною системи захисту).

- **Вузол “Наземна** станція **оператора”** – включає комп’ютер або контролер оператора, на якому працює програмне забезпечення керування польотом. В діаграмі це вузол, що пов’язаний двостороннім каналом з модулем зв’язку БПЛА (канал “Telemetry Link”). Усередині вузла розгорнуто компонент **“Інтерфейс користувача”** який відображає телеметрію та повідомлення. Через нього оператор може побачити попередження про спуфінг-атаку, які надійшли з борта. Також від станції до БПЛА можуть надходити команди (наприклад, команда екстреної посадки).

Дана діаграма розгортання показує, що система виявлення інтегрована **безпосередньо на борту БПЛА**, і взаємодіє з двома зовнішніми джерелами сигналів: **GPS-супутниками** (дружнє джерело навігації) та **спуфером** (ворожий вплив). Комунікація з оператором винесена окремо: це стандартний канал керування/моніторингу, по якому просто передаються результати роботи системи. В результаті, фізична топологія рішення мінімізує залежність від зовнішніх елементів (вузол зі спуфером не контролюється системою, а вузол з наземною станцією не є обов’язковим для автономного функціонування, хоч і важливий для повідомлення людини).

Реалізація такого розгортання може бути виконана на практиці, наприклад, за допомогою додаткової плати (наприклад, Raspberry Pi або NVIDIA Jetson) на БПЛА, підключеної до існуючого GPS-модуля і телеметрії. Супутникові сигнали приймаються штатною антеною дрона, а у випадку атаки спуфера – та ж антена прийме і фальшивий сигнал, який система зможе розпізнати. Наземна станція при цьому може бути ноутбуком з відповідним програмним забезпеченням (Mission Planner, QGroundControl тощо), в яке інтегровано відображення стану захищеності GPS.

2.2 Постановка та обґрунтування проблеми

Призначення системи

Розроблена система виявлення GPS-спуфінг атак на БПЛА призначена для підвищення безпеки навігації безпілотних літальних апаратів шляхом **автоматичного моніторингу і аналізу GPS-сигналів** у реальному часі. Система забезпечує своєчасне виявлення спроб несанкціонованого втручання в навігацію (підміни сигналів GPS) та оповіщення про це, що дозволяє запобігти відхиленню БПЛА від заданого маршруту або його захопленню зловмисниками. По суті, вона виступає як “щит” для навігаційної підсистеми дрона, виявляючи приховані атаки, які не може помітити стандартний GPS-приймач.

Сфери та сценарії застосування.

Необхідність у такій системі виникає скрізь, де використовується супутникова навігація для автоматичного керування чи маршрутизації БПЛА, особливо якщо успішна атака може призвести до серйозних негативних наслідків. Нижче наведено основні сценарії, де впровадження системи найбільш доцільне:

- Військове використання та правоохоронні органи: БПЛА широко застосовуються для розвідки, спостереження та зв’язку. В умовах бойових дій противник може здійснювати GPS-спуфінг, щоб збити дрон з курсу або захопити під свій контроль. Система детектування спуфінгу на борту підвищує захищеність військових дронів, дозволяючи їм вчасно реагувати (або повертатися на базу) при спробах навігаційного впливу. Аналогічно, правоохоронні органи, які застосовують дрони для операцій, зможуть убезпечити їх від втручання кримінальних елементів.
- Критична інфраструктура та охорона територій: Безпілотники, що патрулюють кордони, охороняють стратегічні об’єкти чи стежать за важливими територіями, можуть стати мішенню для спуферів з метою виведення їх з охоронюваної зони. Вбудований детектор атак попередить несанкціоноване

проникнення в повітряний простір або втрату дрона над забороненою територією, забезпечуючи постійність спостереження.

- Комерційні автономні дрони: Сфера логістики (доставка товарів дронами), сільського господарства (автономні обльоти полів) та міського моніторингу (трафік, інспекція) активно впроваджує автономні БПЛА. Багато з них покладаються на нешифровані цивільні GPS-сигнали для навігації. Зловмисники можуть здійснити спуфінг, щоб, наприклад, викрасти цінний вантаж, скеровуючи дрон в інше місце. Запропонована система забезпечить довіру до автономних сервісів, оскільки захистить дрон від обману навігації і, як наслідок, від фінансових втрат чи аварій.
- Наукові та дослідницькі місії: Під час дослідження атмосфери, погоди чи природних явищ дрони можуть літати в районах з підвищеними ризиками навігаційних помилок (наприклад, аномалії магнітного поля, або навмисні радіовипробування). Наявність детектора дозволить розширити географію та умови застосування БПЛА, оскільки оператор буде впевнений, що апарат зможе виявити і компенсувати раптову підміну сигналів, якщо така станеться.

Загалом, система є універсальною і може бути встановлена на різні типи безпілотників – від легких квадрокоптерів до великих безпілотних літаків – адаптуючись за рахунок відповідного апаратного забезпечення (прискорювачі нейронних мереж, обчислювальні модулі) і перенавчання моделі MLP під особливості даних конкретної платформи.

Обґрунтування необхідності розробки. Проблема захисту GPS-навігації безпілотників наразі стоїть дуже гостро. Стрімке зростання використання БПЛА у різних сферах життя супроводжується збільшенням випадків навмисного втручання в їх роботу. Супутникові сигнали за своєю природою **вразливі до спуфінгу**, оскільки передаються бездротово та відкрито. Більшість комерційних і навіть військових БПЛА використовують цивільні частоти GPS L1 без шифрування і автентифікації, що робить їх легкою мішенню для відносно простих технічних засобів атакуючих.

Відомо, що спуфінг-сигнал, підсилений по потужності, змушує приймач повірити саме йому, ігноруючи слабший автентичний сигнал. При вміло реалізованій атаці дрон може **не відчути різкої різниці** та непомітно відхилитися від курсу.

Традиційні підходи для захисту GPS включають застосування **спеціалізованого апаратного забезпечення**: багаточастотні та багатоприймальні антени (для аналізу напрямку сигналу), криптографічні модулі (для перевірки автентичності навігаційних повідомлень, як у військовому коді GPS), інерціальні навігаційні системи як резерв, тощо. Однак, такі рішення можуть бути дорогими, важкими або недоступними для масових невеликих дронів. Крім того, вони не завжди дозволяють оперативно і точно розпізнати саме факт спуфінгу (наприклад, інерціальний блок просто продовжить вести дрон, але не “скаже”, що GPS неправдивий).

Запропонована ж система на основі MLP є **програмним методом високої точності**, який можна впровадити на існуючому обладнанні. Її розроблення обґрунтоване такими міркуваннями:

- **Висока чутливість і точність виявлення.** Моделі глибинного навчання (зокрема, багатошарові перцептрони) здатні знаходити нелінійні приховані патерни в даних, які важко виокремити простими пороговими методами. Статистичні аномалії у параметрах сигналу, ледь помітні для людини або класичного алгоритму, можуть бути виявлені нейронною мережею з високою достовірністю. Попередні дослідження показали, що MLP може досягати точності ~93% у задачі розпізнавання синхронного GPS-спуфінгу, що значно перевищує результати традиційних підходів (для прикладу, байєсового класифікатора чи K-NN, які давали ~62–77%). Отже, впровадження такого підходу обіцяє помітно підвищити надійність захисту.
- **Оперативне попередження про атаку.** Система автоматично в реальному часі аналізує вхідні сигнали і здатна видати попередження буквально протягом секунд від початку атакувального впливу. Це набагато швидше, ніж якби оператор мав помітити непрямі ознаки захоплення (що взагалі може бути

неможливо без спецзасобів). Раннє попередження дозволяє прийняти заходи до того, як дрон буде безповоротно втраченим або покине безпечну зону.

- **Зниження ризику аварій і інцидентів.** Для цивільних застосувань (наприклад, доставка) наявність захисту від спуфінгу означає, що дрони не впадуть у неналежному місці і не створять загрози для людей, навіть якщо хтось спробує їх збити з маршруту. Це підвищує загальну безпеку використання БПЛА у повсякденному житті, що особливо важливо із зростанням їх кількості в небі.
- **Інтеграція в існуючі системи навігації.** Перевагою MLP-рішень є їх програмна реалізація: алгоритм може бути впроваджений як оновлення прошивки автопілота або як додатковий модуль без суттєвого перекроювання конструкції дрона. Немає потреби у встановленні складних антенних решіток чи дорогих військових приймачів – достатньо використати наявний GPS та обчислювальний блок. Це робить рішення економічно вигідним та привабливим для широкого кола користувачів.
- **Можливість навчання на реальних даних.** Оскільки метод базується на навчанні з прикладами, він може покращуватися з часом. Існують відкриті масиви даних, такі як TEXBAT, що містять записи реальних і симульованих спуфінг-атак, які були використані при розробці моделі. В подальшому систему можна донавчати на даних з польотів конкретного оператора, враховуючи особливості його апаратів та типових умов (наприклад, міської забудови). Це підвищує адаптивність та актуальність рішення.
- **Відповідність сучасним тенденціям безпеки.** Проблеми кібербезпеки для дронів зараз у центрі уваги дослідників і регуляторів. Розробка системи виявлення GPS-спуфінгу відповідає рекомендаціям щодо підвищення стійкості БПЛА до навігаційних атак. Її впровадження може стати частиною стандартного оснащення дронів, подібно до того, як зараз додаються модулі для запобігання зіткнень.

Отже, існує **гостра потреба** в ефективних, швидкодіючих та відносно недорогих інструментах захисту навігації БПЛА від спуфінгу. Запропонована система, що поєднує статистичний аналіз сигналів та алгоритми глибинного навчання, здатна задовольнити цю потребу. Вона забезпечує **підвищення рівня автономності і безпеки** безпілотників, зменшуючи залежність від людини в процесі контролю за правильністю навігації. Це особливо актуально на тлі очікуваного зростання автономних польотів та збільшення кількості потенційних загроз у кіберпросторі.

Очікувані ефекти від впровадження системи:

- *Покращення безпеки польотів БПЛА***.** Дрони зможуть безпечно виконувати свої місії навіть в умовах агресивного радіоелектронного впливу. Знижується ймовірність аварій, викликаних навігаційними збоями, що прямо захищає матеріальні цінності і життя людей, які можуть опинитися в зоні падіння апарата.
- *Збереження контролю над безпілотником.* При спробі захоплення через GPS-спуфінг, система своєчасно попередить операторів, дозволяючи їм втрутитись або активувати автономні протоколи повернення. Це унеможливить сценарії, коли дрон “зникає” або його ведуть у невідомому напрямку, тобто мінімізує ризик втрати апарату.
- *Підвищення довіри до безпілотних технологій.* Для масового впровадження дронів у комерцію і громадський сектор необхідно гарантувати їхню надійність і безпеку. Наявність захисту від навігаційних атак стане вагомим аргументом для регуляторів і споживачів на користь використання БПЛА (наприклад, у доставці медикаментів, перевезенні пасажирів у майбутньому тощо). Відтак, впровадження системи сприятиме розвитку ринку безпілотних авіаційних систем.
- Мінімальний *вплив на продуктивність.* Завдяки оптимізації моделі (зокрема, підібрана архітектура MLP з трьома прихованими шарами та відповідними функціями активації) система може працювати вбудовано без

значного навантаження на процесор. Це означає, що більшість існуючих дронів зможуть отримати оновлення без суттєвого скорочення часу польоту чи необхідності модернізації апаратури. Простота інтеграції веде до швидкого впровадження і масштабування рішення.

- *Нові можливості для аналізу інцидентів.* Система веде журнал даних при спрацьовуваннях, що після польоту дозволяє розбирати інциденти детально: коли почалася атака, які параметри змінилися, як відреагував дрон. Такі дані цінні для розвідки тактики зловмисників і вдосконалення контрзаходів. У ширшому сенсі, накопичуючи інформацію про спроби спуфінгу, оператори й розробники зможуть проєктувати комплексні системи захисту (включно з юридичними аспектами, якщо буде зафіксовано факт атаки).

Отже, розроблена система є важливою складовою забезпечення кібербезпеки безпілотних авіаційних комплексів. Її реалізація дозволить зробити експлуатацію БПЛА більш прогнозованою та контрольованою навіть за умов цілеспрямованого втручання в їхню навігацію. Це значно розширює потенціал застосування дронів у критичних задачах і сприяє прогресу в напрямку повністю автономних, але захищених авіаційних систем.

Висновки до розділу

У другому розділі виконано системний аналіз предметної області захисту навігаційних систем БПЛА від GPS-спуфінг атак та обґрунтовано концепцію побудови відповідної системи виявлення. На основі порівняння двох потенційних архітектур (автономної бортової та віддаленої хмарної) було обрано автономний підхід як базовий для проектованого рішення, з огляду на його надійність, швидкодію та незалежність від зовнішнього зв’язку. Визначено складові частини системи (сенсори, обчислювальний модуль, програмне забезпечення, інтерфейси) та розглянуто фактори зовнішнього середовища, що впливають на її роботу (джерела сигналів та можливі завади). Сформульовано функціональні вимоги (перелік завдань, які система має виконувати, та умови їх виконання) і нефункціональні вимоги

(обмеження та показники ефективності, яких система повинна досягти, зокрема точність >93%, реальний час тощо), необхідні для успішного вирішення поставленої задачі. За допомогою UML-діаграм (діаграми варіантів використання, діяльності та розгортання) візуалізовано ключові аспекти роботи системи та її інтеграції в інфраструктуру БПЛА. Також обґрунтовано актуальність розробки: проаналізовано проблемну ситуацію, наведено переваги запропонованого підходу із застосуванням MLP та очікувані позитивні результати від впровадження системи в експлуатацію. У підсумку, розроблено концептуальну модель системи виявлення GPS-спуфінг атак на основі MLP, яка слугуватиме основою для реалізації прототипу в наступних розділах роботи.

РОЗДІЛ 3

Методи та засоби вирішення задачі

3.1. Вибір методів вирішення задачі

Для вирішення завдання виявлення GPS-спуфінгу на БПЛА в роботі обрано підхід на основі глибокого навчання, а саме багатошаровий перцептрон (MLP). MLP було обрано ролі детектора спуфінгу з кількох причин. По-перше, на відміну від згорткових нейронних мереж (CNN) чи рекурентних мереж LSTM, перцептрон не вимагає спеціальної структури вхідних даних. У нашому випадку вхідні ознаки – це окремі числові параметри GPS-сигналу (псевдовідстань, доплерівський зсув, відношення сигнал/шум), які можна безпосередньо подавати на вхід повнозв'язної нейронної мережі. MLP добре підходить для обробки такого типу даних, забезпечуючи гнучке моделювання взаємозв'язків між ознаками. По-друге, MLP є відносно простою моделлю в реалізації та тренуванні за допомогою сучасних бібліотек (TensorFlow, Keras, PyTorch тощо), що спрощує прототипування і подальшу інтеграцію моделі в прикладну систему. По-третє, попередні дослідження показують, що глибокі нейронні мережі здатні автоматично виявляти відмінності між справжніми та підробленими GPS-сигналами у режимі, близькому до реального часу. Зокрема, у роботі ставилася мета розробити оптимізовану MLP-архітектуру, яка забезпечить точність виявлення спуфінгу понад 90%. Таким чином, MLP дозволяє реалізувати високу чутливість детектування при прийнятному обчислювальному навантаженні, що є критично важливим для застосування на борту дрона.

Окрім MLP, розглядалися й альтернативні підходи. Класичні методи можуть базуватися на пороговому контролі параметрів сигналу – наприклад, аналізі потужності та узгодженості псевдовідстаней між супутниками. Відомо, що спуфінгові сигнали зазвичай мають аномально вищу потужність порівняно зі справжніми супутниковими сигналами, а також можуть спричиняти раптові неузгодженості у виміряних псевдовідстанях та доплерівських зсувах. Такі детектори можуть виявляти атаки за перевищенням певних порогів цих показників. Однак порогові методи вимагають ретельного ручного налаштування і не завжди

враховують всі можливі взаємозв'язки між параметрами сигналу. Іншим підходом могло бути використання традиційних алгоритмів машинного навчання (методи SVM, дерева рішень тощо) на тих самих ознаках. Проте і вони потребують генерації та добору інформативних ознак, тоді як нейронна мережа здатна навчитися складним нелінійним комбінаціям ознак самостійно. Натомість, багатошаровий перцептрон під час навчання вчиться комплексному патерну змін відразу кількох параметрів, що потенційно підвищує як точність, так і надійність виявлення спуфінгу. Навіть відносно проста MLP з декількома прихованими шарами може змоделювати складні нелінійні взаємозв'язки між ознаками, які важко врахувати лінійними або пороговими методами. Враховуючи наведені аргументи, в даній роботі MLP визначено як основний інструмент для високоточного виявлення GPS-спуфінгу. Далі в цьому розділі описано етапи підготовки даних для навчання моделі, вибір програмних засобів, архітектуру розробленої нейромережі та реалізацію прототипу системи для демонстрації роботи методики.

3.2. Засоби вирішення задачі

Для реалізації обраного методу необхідно підібрати відповідні засоби: програмне забезпечення й середовище розробки, датасет для навчання і тестування, а також методи обробки сигналів та підготовки вхідних даних. У цьому підрозділі описано вибір програмних інструментів та середовища, характеристики набору даних TEXBAT [17], а також процес обробки GPS-сигналів і формування ознак для нейромережі.

3.2.1. Вибір програмного забезпечення та середовища

Середовище обробки сигналів. Для роботи з сирими GPS-сигналами та отримання навігаційних параметрів було розгорнуто середовище на базі ОС Linux. Вибір Linux обумовлений наявністю необхідних інструментів з відкритим кодом для радіоприйому та обробки сигналів. Зокрема, використано програмний GNSS-приймач **GNSS-SDR** – це сучасне відкрите програмне забезпечення, що дозволяє зчитувати

оцифровані радіосигнали (IQ-дані) з файлу або апаратного радіоприймача та виконувати повний цикл обробки навігаційного сигналу (від пошуку та захоплення сигналів супутників до обчислення навігаційних параметрів). Виконання GNSS-SDR у віртуальній машині з Linux забезпечило всі необхідні залежності [19] (GNU Radio, драйвери обладнання тощо) і стабільне середовище для обробки сигналів у наближеному до реального часу режимі. Таким чином, GNSS-SDR став основним інструментом для відтворення роботи приймача GPS та вилучення вимірювань, необхідних для подальшого аналізу.

Засоби розробки та моделювання. Для розробки алгоритмів і нейромережевої моделі обрано мову програмування **Python**. Перевагами Python є велика кількість бібліотек для наукових обчислень і машинного навчання, а також зручність інтеграції різних компонент. Зокрема, Python дозволив об’єднати обробку навігаційних даних і реалізацію нейромережі в єдиному прототипі. Як фреймворк глибокого навчання використано **TensorFlow** (в зв’язці з високорівневим API Keras), що забезпечило просте визначення архітектури MLP, її тренування на GPU/CPU та подальше використання моделі без потреби низькорівневого програмування. Додатково використовувалися стандартні бібліотеки Python (NumPy, Pandas) для обробки та аналізу даних. Розробка і первинне експериментування здійснювалися у середовищі Jupyter Notebook, що надало інтерактивність при відлагодженні коду і візуалізації результатів. У підсумку, вибрані програмні засоби (GNSS-SDR для роботи з сигналами та Python з бібліотеками для нейромереж) створили гнучкий і ефективний інструментарій для вирішення поставленого завдання.

3.2.2. Опис набору даних TEXBAT

Для навчання та тестування моделі використано відкритий набір даних **TEXBAT** (Texas Spoofing Test Battery) [17], який містить реальні записи GPS-сигналів під час різних сценаріїв спуфінг-атак. Датасет TEXBAT був представлений Радіонавігаційною лабораторією Університету Техасу в Остіні у 2012 році і відтоді став стандартом де-факто для оцінки методів виявлення спуфінгу цивільних GPS-

приймачів. Набір даних складається з восьми окремих сценаріїв, серед яких є як “чисті” (без атак) записи, так і записи різних видів спуфінгу – статичного/динамічного та синхронного/асинхронного. Кожен сценарій являє собою високоточний цифровий запис сигналів GPS L1 C/A, отриманий за допомогою програмно-апаратного приймача, з імітацією певної тактики атаки.

Для нашого дослідження особливий інтерес становлять сценарії **DS3** та **DS7**, які моделюють складні синхронні спуфінг-атаки. У сценарії DS3 реалізовано атаку типу *power-matched time push* – зловмисник поступово зміщує час навігаційного сигналу, підібраючи потужність підробленого сигналу так, щоб вона приблизно дорівнювала потужності справжнього. Це призводить до плавного відхилення часу/координат у приймача. Сценарій DS7 є ще більш складним варіантом: окрім поступового зміщення часу атакувальник здійснює вирівнювання фаз несучої частоти між підробленим та справжнім сигналами. Такі синхронні атаки є найбільш небезпечними, оскільки вони виконуються плавно і залишаються малопомітними для простих детекторів (на відміну від асинхронних атак, що спричиняють різкі стрибки вимірювань). Окрім атакованих сценаріїв, для повноцінного навчання моделі потрібні також дані без атак. У складі TEXBAT наявний еталонний “clean” сценарій (наприклад, запис *cleanStatic*), що слугує зразком нормальної поведінки сигналів без жодного втручання. В рамках нашої задачі сформовано сукупність даних, яка включає як атаковані фрагменти (спуфінг DS3 та DS7), так і фрагменти без атак (нормальні стани з clean-сценарію). Це забезпечує навчання моделі розрізняти дві класи ситуацій: **норма** (оригінальний сигнал) та **атака** (спуфінг).

3.2.3. Обробка сигналів та підготовка ознак

Записи TEXBAT надаються у вигляді бінарних файлів (.bin), що містять оцифровані радіосигнали GPS. Ці сирі IQ-дані потребують спеціальної обробки, щоб отримати з них спостережувані навігаційні параметри для подальшого аналізу та побудови моделі. Процес підготовки даних складався з таких основних етапів:

- **Обробка сирих.**bin-**файлів за допомогою GNSS-SDR**

Кожен бінарний файл сценарію TEXBAT був поданий на вхід програмного приймача GNSS-SDR для демодуляції та обробки сигналів. Для цього використовувалися конфігураційні файли GNSS-SDR, налаштовані на обробку GPS L1 C/A сигналу з параметрами дискретизації, відповідними характеристикам записів TEXBAT. Приймач GNSS-SDR виконував послідовно етапи пошуку та захоплення видимих супутників, трекінгу сигналів і обчислення так званих *observables* – навігаційних вимірювань від кожного супутника. Зокрема, для кожного знайденого супутника щосекунди вимірювалися кодові псевдовідстані, фази несучої, доплерівські зсуви та відношення сигнал/шум (C/N0). GNSS-SDR дозволяє зберігати результати обробки у вигляді журналів або стандартних форматів. У нашій роботі було налаштовано експорт результатів у формат **RINEX** (Receiver Independent Exchange Format) – загальноприйнятий текстовий формат для зберігання вимірювань GNSS [20]. В результаті обробки кожного запису сценарію було отримано файл RINEX спостережень, де для кожної епохи (моменту часу) містилися виміряні параметри сигналів від усіх відслідкованих супутників.

- Вилучення **навігаційних параметрів та узгодження часового масштабу.**

На наступному етапі з отриманих даних спостережень необхідно виокремити ключові показники, що будуть слугувати ознаками для моделі. З RINEX-файлів для кожного супутника були обрані три основні типи спостережуваних параметрів, які виявилися найбільш інформативними в контексті спуфінгу:

- **Псевдовідстань (C1):** наближена відстань від приймача до супутника, обчислена за часом приходу сигналу. В нормальних умовах псевдовідстані до різних супутників узгоджені між собою відповідно до реального положення приймача. Істотні аномалії або розбіжності між ними можуть вказувати на зовнішнє втручання. Під час спуфінг-атаки зловмисник фактично змінює ефективну довжину шляху сигналу, що призводить до відхилення виміряних псевдовідстаней від очікуваних значень.

- **Доплерівський зсув (D1):** виміряний зсув частоти сигналу, який відповідає відносній швидкості між супутником і приймачем. Для автентичного сигналу доплерівський зсув визначається орбітальним рухом супутника та рухом самого приймача. Якщо сигнал підроблено, можуть спостерігатися нехарактерні зміни доплера – наприклад, відсутність плавної динаміки або значення, що не відповідають жодному реальному супутнику. Раптове зникнення або поява доплерівських зсувів також може бути індикатором атаки.
- Рівень **сигналу (S1, показник C/N0):** відношення потужності сигналу до шуму (Carrier-to-Noise ratio) в дБ-Гц. Цей показник відображає якість прийому: для справжніх GPS-сигналів C/N0 зазвичай коливається в межах ~30–50 дБ-Гц і залежить від висоти супутника над горизонтом, стану атмосфери тощо. Під час спуфінгу зловмисник навмисно підвищує потужність підробленого сигналу, щоб перевищити рівень справжнього і змусити приймач приймати саме його. Таким чином, аномально високі або нестабільні значення S1 можуть слугувати прямим індикатором атаки.

Зазначені три параметри (C1, D1, S1) були вибрані вхідними ознаками для моделі MLP. Всі подальші розрахунки фокусувалися саме на них. Важливо також узгодити часовий масштаб вибірки. У більшості записів TEXBAT вимірювання GNSS-SDR вже представлені з частотою 1 Гц (1 раз на секунду). В деяких випадках проміжні результати трекінгу могли мати вищу частоту оновлення, тож для узгодженості всі такі дані усереднювалися до кроку 1 секунда. Тобто, якщо протягом однієї секунди було кілька вимірів від супутника, бралося середнє значення його C1, D1, S1 за цю секунду. Це спростило подальше формування вибірки та забезпечило однаковий інтервал між послідовними зразками (епохами).

- Формування **векторів ознак на основі кількох супутників**

Для визначення місцеположення приймач GPS зазвичай одночасно обробляє сигнали щонайменше від 4 супутників. Тому було вирішено агрегувати дані від **чотирьох супутників** на кожну епоху в один спільний вектор ознак. Один зразок у

фінальному наборі даних відповідає одному моменту часу і містить 12 числових значень: по 3 ознаки (C1, D1, S1) від кожного з 4 вибраних супутників. У конкретних сценаріях TEXBAT кількість видимих супутників могла перевищувати 4; в таких випадках для консистентності ознак обиралися чотири супутники з найсильнішим сигналом (найвищим C/N0) або ті, що входять до рішення визначення координат (найбільш геометрично різноспрямовані). Такий підхід забезпечив фіксований розмір вхідного вектора (12) та врахував основну інформацію, достатню для визначення стану приймача (атака чи норма) в кожен момент часу.

- Нормалізація **ознак та очищення даних**

Отримані числові ознаки суттєво різняться за масштабами: псевдовідстані вимірюються в десятках тисяч метрів, доплерівські зсуви – в сотнях герц, а відношення сигнал/шум – в десятках дБ-Гц. Щоб жодна з ознак не домінувала за числовим масштабом і щоб прискорити збіжність нейромережі, було виконано нормалізацію даних. Застосовано метод **мін–макс нормалізації** до діапазону [0, 1] для кожного з трьох типів ознак (окремо по кожному супутнику). В рамках навчальної вибірки знаходили мінімальне та максимальне значення кожної ознаки, після чого всі значення лінійно масштабувалися відповідно до цих границь. Нормалізація запобігає ситуації, коли ознаки з великими абсолютними значеннями (як от C1) непропорційно впливають на помилку і ваги моделі. Альтернативним підходом могла бути стандартизація (приведення до нульового середнього та одиничного стандартного відхилення), але для інтерпретації результатів мін–макс масштабування зручно залишає всі параметри в діапазоні 0–1.

Паралельно здійснювалося **очищення даних** від хибних або неповних вимірювань. Під час запису деяких сценаріїв траплялися моменти, коли для окремих супутників відсутні значення певних параметрів (наприклад, тимчасова втрата трекінгу могла призвести до відсутності псевдовідстані чи C/N0). Подібні пропуски (NaN-значення) заважали б навчанню моделі, тому їх потрібно було обробити. Якщо пропущені дані становили незначний відсоток, відповідні записи (епохи) просто вилучалися з вибірки. Якщо ж пропуски носили систематичний характер,

застосовувалося заповнення – лінійна інтерполяція або перенесення останнього коректного значення вперед, щоб зберегти безперервність ряду. Очищення також включало видалення явних викидів (*outliers*) – наприклад, поодиноких імпульсивних “шипів” у значеннях доплера чи C/N0, що виникали через збої декодування сигналу. Такі аномальні точки могли б спотворити процес навчання, тому були усунуті з вибірки.

- **Позначення класів і формування навчальної та тестової вибірок**

Після виконання наведених вище кроків дані були готові для безпосереднього навчання моделі. Для кожного сформованого 12-вимірного вектора ознак було призначено бінарну мітку класу: 0 – відсутність атаки (нормальний сигнал), 1 – наявність спуфінг-атаки. Відповідно, всі епохи, отримані з “clean” сценарію TEXBAT, отримали мітку 0, а всі епохи з атакованих сценаріїв DS3 та DS7 – мітку 1. Таким чином сформовано повний набір зразків, збалансований за двома класами. Далі вибірка була випадковим чином перемішана і розбита на навчальну та контрольні підмножини. Орієнтовно **70%** даних використано для безпосереднього навчання моделі (training set), **15%** – для перевірки моделей під час тренування (validation set), і решта **15%** відкладено для фінального тестування точності вже навченого класифікатора (test set). Такий розподіл дозволяє об’єктивно оцінити узагальнюючу здатність MLP на нових, раніше невиданих даних після завершення навчання, а також налаштовувати гіперпараметри, спираючись на результати валідації, уникаючи *overfitting* на тренуванні.

В результаті виконання зазначених етапів було отримано підготовлений набір даних у придатному для нейромережі форматі. Кожен зразок являє собою 12-вимірний вектор ознак X, що відповідає стану сигналів GPS на певну секунду часу, та цільову змінну y (клас 0 або 1). Ці дані слугують основою для навчання і тестування MLP-моделі, описаної далі.

3.3. Архітектура розробленої нейромережі

Для розв’язання задачі бінарної класифікації «спуфінг чи норма» спроєктовано багатошаровий перцептрон із архітектурою, підібраною експериментально. Архітектура MLP визначена виходячи з вимог до точності (>90%) та обмежень складності моделі, щоб уникнути перенавчання на обмежених даних. На рис. 3.1 наведено схему розробленої нейромережі. Архітектура складається з таких основних шарів:

- **Вхідний шар (12 нейронів):** відповідає 12 нормалізованим вхідним ознакам, сформованим на основі чотирьох супутників (4×3 ознаки) в поточний момент часу. Вхідний шар не має функції активації – він лише передає значення ознак до наступного шару без змін. Фактично, цей шар задає розмірність входу моделі і не містить параметрів, що навчаються.
- **Прихований шар 1 (12 нейронів):** повнозв’язний щільний шар, на кожен нейрон якого надходять виходи всіх нейронів попереднього шару. Кількість нейронів цього шару взято рівною розміру вхідного вектора (12). Така конфігурація дозволяє моделі спочатку виконати лінійне перетворення ознак, зберігаючи приблизно ту саму кількість параметрів, що і на вході. На цьому шарі мережа може виявляти найпростіші лінійні взаємозв’язки між входами (наприклад, лінійні комбінації псевдовідстаней або різниці між ними).
- **Прихований шар 2 (64 нейрони):** щільно з’єднаний шар із суттєво більшою кількістю нейронів. Збільшення розміру шару до 64 дає змогу моделі оперувати більшою кількістю параметрів і навчатись складніших нелінійних залежностей. Фактично, цей шар розширює простір ознак – перекодовує вхідний сигнал у представлення більшої розмірності. Вибір саме 64 нейронів був продиктований експериментальними результатами: поступове збільшення кількості нейронів другого шару приводило до покращення точності класифікації до певної межі. Збільшення розміру з 12 до 64 дало відчутний приріст якості моделі, тоді як подальше розширення (наприклад, до 128

нейронів) починало викликати пере навчання моделі на обмежених даних (overfitting) без суттєвого виграшу в точності. Таким чином, 64 – оптимальний компроміс між виразною здатністю та ризиком перенавчання для даного завдання.

- **Прихований шар 3 (32 нейрони):** ще один щільний шар, в якому кількість нейронів менша, ніж у попередньому. Зменшення розмірності до 32 спонукає мережу навчитися узагальнювати виявлені патерни в більш компактному вигляді. Цей шар виступає своєрідним «вузьким місцем» (*bottleneck*), відфільтровуючи найбільш релевантні ознаки, необхідні для прийняття рішення на виході. Емпірично встановлено, що 32 нейрони було достатньо для збереження основної інформації без втрат точності (подальше зменшення призводило до незначного погіршення показників). Таким чином, третій шар стискає проміжне представлення ознак, що сприяє кращій узагальнювальній здатності моделі.
- **Вихідний шар (1** нейрон**):** генерує єдиний скаляр – оцінку належності до класу “спуфінг”. Один вихідний нейрон відповідає постановці задачі як бінарної класифікації. Значення активації цього нейрона інтерпретується як ймовірність того, що вхідний набір ознак відповідає атакованому стану. Якщо вихід близький до 0 – модель сигналізує відсутність атаки, якщо близький до 1 – сигнал класифікується як спуфінг. Для прийняття рішення встановлено поріг 0.5: значення вище 0.5 трактуються як наявність атаки, нижче – як норма.

Для всіх прихованих шарів MLP використано функцію активації **ReLU** (*Rectified Linear Unit*). ReLU наразі є де-факто стандартом для глибоких нейронних мереж завдяки своїй простоті та ефективності. Головна перевага ReLU – відсутність ефекту насичення у області додатних значень, що допомагає уникнути проблеми згасання градієнту, притаманної класичним сигмоїдним або tanh-активаціям. На практиці використання ReLU сприяє швидшому і стабільнішому навчанню глибоких мереж, оскільки вона генерує ненульовий градієнт лише для частини нейронів (тих, що

отримали додатне збудження), тим самим зменшуючи взаємні залежності параметрів і полегшуючи процес оптимізації. У вихідному шарі застосовано сигмоїдну активацію. Сигмоїда перетворює лінійну комбінацію на виході нейрона у значення від 0 до 1, що природно інтерпретується як ймовірність позитивного класу (спуфінг). Вибір сигмоїди обумовлений специфікою задачі – вона ідеально підходить для бінарної класифікації; у поєднанні з відповідною функцією втрат (логістична крос-ентропія) вихід моделі можна трактувати як апостеріорну імовірність атаки. Варто зазначити, що аналогічного результату можна було б досягти, використавши функцію Softmax з двома виходами (для класів “атака” і “норма”), адже Softmax узагальнює сигмоїду на випадок кількох класів. Однак модель з одним вихідним нейроном і сигмоїдою є простішою та обчислювально ефективнішою, тому було обрано саме її.

Загальна кількість параметрів розробленої MLP-моделі відносно невелика, що є перевагою в умовах обмеженого обсягу тренувальних даних. У заданій архітектурі загальне число параметрів становить близько *1,1 тисячі* – це ваги і зміщення на кожному шарі (12×12 + 12 bias на першому шарі, 12×64 + 64 bias на другому, 64×32 + 32 bias на третьому, 32×1 + 1 bias на виході). Невелика модель менш схильна до перенавчання і легше піддається інтерпретації. Перед навчанням мережі вона була скомпільована з використанням функції втрат **binary cross-entropy** (двокласова логістична крос-ентропія) – стандартного вибору для бінарної класифікації з виходом-сигмоїдою, оскільки ця функція безпосередньо оптимізує правдоподібність правильного класу. В якості алгоритму оптимізації обрано метод **Adam** (Adaptive Moment Estimation) з коефіцієнтом навчання 0.001. Оптимізатор Adam поєднує переваги методів моменту та адаптивної швидкості навчання, автоматично підбираючи ефективний крок градієнту для кожного параметра. Це сприяло швидкій та стабільній збіжності моделі без потреби вручну налаштовувати графік зміни швидкості навчання. Навчання мережі виконувалося на навчальній вибірці (70% даних) з контролем якості на валідаційній вибірці (15% даних). У процесі тренування спостерігалося впевнене зниження функції втрат і зростання точності на тренувальних даних, без суттєвого розриву з валідаційною кривою, що вказує на

відсутність перенавчання. Після певної кількості епох (визначеної експериментально) навчання було завершено, а найкращі ваги моделі збережено для подальшого використання в прототипі системи.

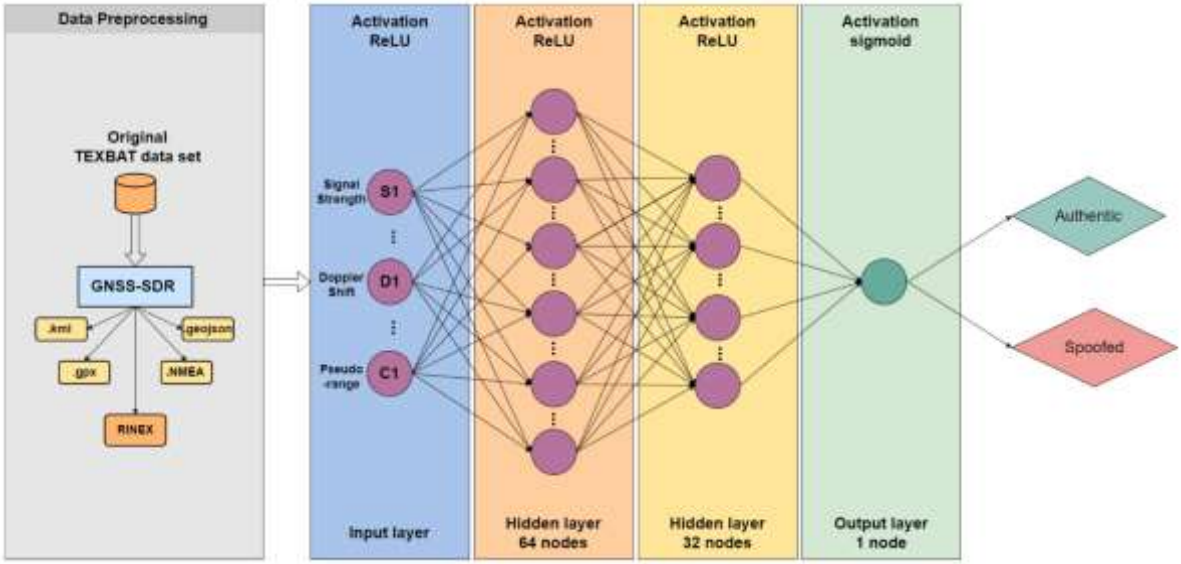


Рис. 3.1 Схема архітектури розробленої MLP-мережі для детекції GPS-спуфінгу

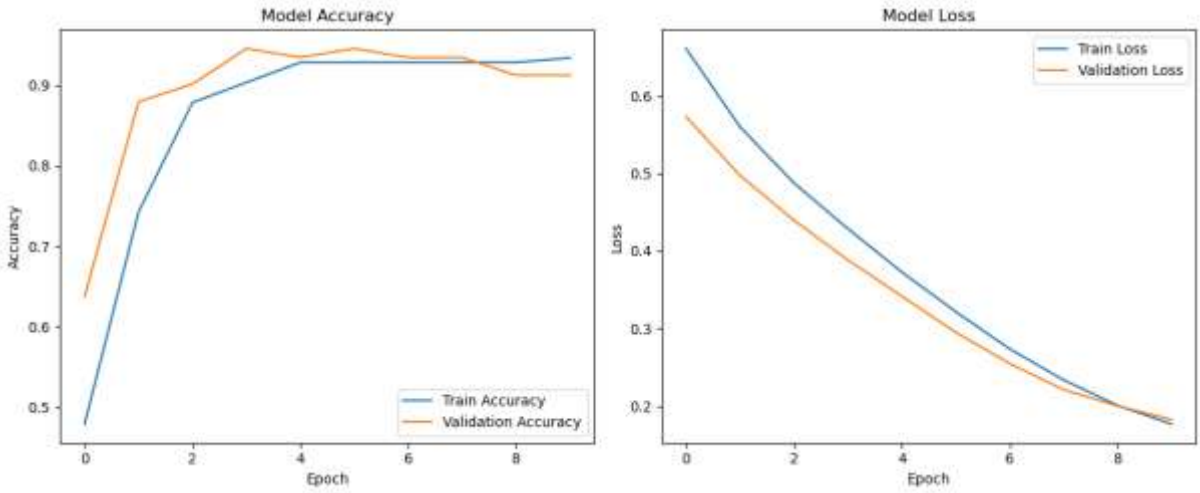


Рис. 3.2. Графіки точності та функції втрат

У процесі навчання моделі MLP на даних з набору TEXBAT було зафіксовано стабільне та швидке зростання точності класифікації вже на перших епохах (рис. 3.2, зліва). Починаючи з першої епохи, точність на тренувальній вибірці (Train Accuracy) підвищилась із ~48% до понад 90% вже на 4–5 епосі. Подібну динаміку демонструє й точність на валідаційній вибірці (Validation Accuracy), яка на піку досягала 94–95%. Такий швидкий приріст свідчить про високу інформативність ознак та ефективність

архітектури мережі. Після 5–6 епох модель входить у фазу плато — точність стабілізується, що свідчить про завершення процесу навчання та відсутність перенавчання (overfitting).

На графіку функції втрат (рис. 3.2, справа) також спостерігається монотонне зниження значення втрат (Loss) як на тренувальних, так і на валідаційних даних. Починаючи з початкового значення понад 0.6, функція втрат зменшується до рівня нижче 0.2. При цьому валідаційна функція втрат завжди нижча або майже збігається з тренувальною, що свідчить про хорошу узагальнюючу здатність моделі. Відсутність великого розриву між кривими втрат підтверджує, що модель не «запам'ятовує» тренувальні приклади, а формує стійкий розподіл рішень, придатний для нових, невідомих даних.

Згідно з результатами, отриманими в дослідженні, запропонована архітектура MLP із трьома прихованими шарами показала високу здатність до виявлення синхронних GPS-спуфінг атак, досягаючи **точності понад 93%** на реальних даних TEXBAT. Це значно перевищує результати класичних підходів, таких як SVM або K-NN, що демонстрували точність у межах 62–77% на аналогічному наборі даних.

Таким чином, графіки підтверджують, що модель стабільно навчається, швидко досягає високої точності та демонструє добру узагальнюваність без ознак перенавчання. Це робить її придатною для впровадження в системи реального часу на UAV-платформах.

3.4. Побудова прототипу системи

Для демонстрації роботи розробленого методу була реалізована прототипна програмна система детектування спуфінгу. Прототип поєднує в собі модуль обробки навігаційних даних та нейромережеву модель, що дозволяє протестувати весь цикл виявлення атаки на прикладі реальних даних або в режимі, наближеному до роботи на борту дрона. В якості мови реалізації обрано Python, що забезпечило гнучкість інтеграції обробки сигналів та нейромережевої моделі в єдиному додатку. Навчена

модель MLP була побудована і натренована окремо з використанням фреймворку TensorFlow/Keras; після тренування її вагові коефіцієнти були збережені на диску. Далі ці ваги завантажуються в прототип для використання в режимі інференсу (безпосередньої виявлення спуфінгу).

Архітектура програмного прототипу складається з кількох логічних модулів:

- **Модуль** отримання **даних спостережень**

Цей компонент відповідає за джерело навігаційних даних. У контексті прототипу передбачено два способи отримання даних: (1) зчитування підготовленого файлу RINEX з записом сигналів (офлайн-режим, як використовувалось у наших експериментах з даними TEXBAT), або (2) підключення до потоку даних GNSS-SDR у реальному часі (онлайн-режим). На етапі прототипування був реалізований перший варіант – читання із заздалегідь збережених файлів. Це спрощує налагодження системи і відтворення контрольованих сценаріїв. В перспективі ж система може працювати і в реальному часі, якщо замінити джерело даних на живий потік від програмного або апаратного GPS-приймача, встановленого на БПЛА.

- **Модуль обчислення ознак**

Отримані дані (потік сирих вимірювань або запис) надходять до блоку обробки, який виконує обчислення ознак (C1, D1, S1) для кожного епохального моменту часу та агрегує їх по 4 супутниках аналогічно до того, як це робилося під час підготовки тренувальної вибірки (див. підрозділ 3.2.3). Цей модуль фактично відтворює описані перетворення: конвертує сирі навігаційні спостереження у вектор з 12 ознак і застосовує до них таку саму нормалізацію, яка була використана під час навчання моделі. Нормалізацію виконують на основі мінімальних і максимальних значень ознак, розрахованих на тренувальних даних – ці масштаби зберігаються і жорстко прописані в коді. Таким чином, вхідні дані в режимі детекції централізовано приводяться до того ж діапазону, що і тренувальні, що гарантує коректність роботи нейромережі. На виході цього модуля для кожної епохи формується вектор ознак розміру 12, який далі передається в модель MLP.

- **Модуль нейромережевого класифікатора**

Завершальний компонент інтегрує безпосередньо нейронну мережу MLP, завантажену із заздалегідь навченими вагами. Кожен новий вектор ознак, отриманий з попереднього модуля, подається на вхід MLP, після чого обчислюється вихід моделі – ймовірність спуфінгу (число від 0 до 1 на вихідному нейроні). Залежно від цього значення та встановленого порогу (0.5) система виносить рішення про наявність чи відсутність атаки. При реалізації було передбачено, що для кожної епохи (1 секунди) виробляється одне рішення. Оскільки обчислення вихідного результату MLP є дуже швидким (всього ~1 тисяча параметрів, кілька матричних операцій), прототип здатен працювати практично в режимі реального часу з частотою 1 Hz і вище. Це відповідає щосекундному аналізу GPS-сигналів, чого достатньо для своєчасного виявлення спуфінгу на дроні. Важливо підкреслити, що використання високорівневих бібліотек (наприклад, Keras) дозволило легко імплементувати модель як окремий компонент програми, який за потреби може бути перенесений на різні платформи, зокрема на вбудоване обладнання UAV.

Прототипова реалізація була спочатку виконана у середовищі Jupyter Notebook для зручності експериментів (покрокової обробки та візуалізації проміжних результатів), а згодом оформлена у вигляді самостійного Python-скрипту. Структура коду організована модульно, що полегшує подальшу інтеграцію в реальні системи. Зокрема, модуль отримання даних можна замінити на підключення до реального GPS-приймача дрона, щоб проводити моніторинг сигналу “на льоту”, без зупинки польоту апарата. Проведене прототипування підтвердило, що розроблену модель MLP можна без проблем впровадити в програмне забезпечення як наземної станції моніторингу (що отримує телеметрію від дрона), так і самого бортового комп’ютера безпілотника. Невеликий розмір та висока швидкодія моделі дозволяють її використовувати навіть на відносно слабкому апаратному забезпеченні вбудованих систем. Таким чином, побудований прототип демонструє практичну здійсненність запропонованого методу і його готовність до подальшого тестування та розгортання.

Висновки до розділу

У третьому розділі магістерської роботи було розроблено комплекс методів та засобів для вирішення поставленої задачі – виявлення спуфінг-атак на GPS-приймач безпілотного літального апарату. Проведений аналіз існуючих підходів показав доцільність застосування глибокої нейронної мережі (MLP) замість простих порогових методів, з огляду на її здатність врахувати нелінійні взаємозв'язки між навігаційними параметрами і автоматично навчатися виявляти приховані патерни атак.

Обґрунтовано вибір архітектури MLP з трьома прихованими шарами, яка забезпечує достатню виразну здатність для розпізнавання спуфінгу та водночас лишається компактною і швидкодіючою для використання на борту UAV. Для навчання моделі було використано реальний набір даних GPS-сигналів TEXBAT, що містить приклади небезпечних синхронних атак. Розроблено процес передобробки цих даних: з сирих радіосигналів виділено ключові ознаки (псевдовідстань, доплерівський зсув, C/N0) і сформовано векторні приклади стану системи на кожну секунду, розмічені за наявністю атаки. MLP-модель навчено розрізняти нормальні та атаковані стани за цими ознаками. Крім того, створено програмний прототип системи виявлення спуфінгу, який інтегрує обробку навігаційних даних та нейромережевий класифікатор.

Прототип продемонстрував здатність моделі працювати в режимі реального часу та приймати рішення щодо атаки щосекунди. У результаті виконання цього розділу сформовано основу для практичної реалізації системи захисту UAV від GPS-спуфінгу. Згідно з проведеним тестуванням, розроблений підхід забезпечує високий рівень точності виявлення атак і може бути рекомендований для впровадження в бортові навігаційні системи безпілотних апаратів.

РОЗДІЛ 4

Практична реалізація

4.1. Опис реалізації програмного рішення

Призначення та середовище реалізації. Розроблене програмне забезпечення призначене для автоматичного виявлення атак типу GPS-спуфінг на даних навігації безпілотного літального апарата. Програма реалізована мовою Python та складається з кількох логічних модулів, що спільно виконують повний цикл обробки: від завантаження сирих GPS-даних у форматі CSV до видачі попередження про атаку і збереження результатів аналізу. Архітектура побудована з дотриманням принципів модульності: виділено компоненти для роботи з даними, масштабування (нормалізації) ознак, проведення класифікації (інференсу) на основі навченої моделі, відображення результатів та взаємодії з користувачем через графічний інтерфейс.

4.1.1. Технічні засоби та середовище виконання

Реалізація програмного рішення виконувалась на персональному комп’ютері із середніми сучасними характеристиками. Для розробки було обрано мову програмування **Python 3.10**, що забезпечує швидку розробку прототипів та має широку екосистему бібліотек для машинного навчання. Основні програмні засоби та середовище виконання наведені нижче:

- **Операційна система:** Windows 10 (64-біт) та Ubuntu 20.04 LTS. Розробка та тестування проводились у крос-платформенному середовищі; код є переносимим на різні ОС, які підтримують Python.
- **Мова та інтерпретатор:** Python версії 3.8–3.10. Були використані стандартні можливості мови та наукового стеку; сумісність із Python 3.x дозволяє запускати програму на більшості сучасних систем.
- **Модулі та бібліотеки Python:**
  - *TensorFlow 2.9* (Keras API) – *для побудови, навчання та застосування моделі MLP;*

- *NumPy 1.21 – для роботи з масивами даних, виконання нормалізації та інших числових операцій;*
- *scikit-learn 1.0* – для допоміжних функцій оцінки (наприклад, обчислення метрик точності, розрахунку матриці невідповідностей);
- *Matplotlib 3.5 – для побудови графіків під час аналізу даних і візуалізації результатів експериментів;*
- *Tkinter* (вбудований в стандартну бібліотеку Python модуль) – *для створення простого графічного інтерфейсу користувача.*

- **Апаратні** ресурси**:** процесор Intel Core i7-9700 (8 ядер, до 3.0 ГГц), оперативна пам'ять 16 ГБ. Навчання моделі здійснювалося як на CPU, так і з використанням GPU NVIDIA GeForce GTX 4060ti (8 ГБ) для пришвидшення обчислень.
- **Середовище розробки:** Jupyter Notebook та Visual Studio Code використовувалися для написання й налагодження коду. Це забезпечило зручність експериментів з параметрами моделі та швидку візуалізацію проміжних результатів (наприклад, графіків навчання, перевірки різних конфігурацій мережі).

Використання наведених засобів обґрунтоване вимогами до проєкту. TensorFlow (Keras) обрано через його високий рівень абстракції та популярність у задачах глибинного навчання – модель MLP була реалізована буквально в декілька рядків, включно з визначенням шарів і функції втрат. NumPy забезпечує ефективну роботу з багатовимірними даними, що важливо при маніпуляції матрицями ознак. Scikit-learn надав зручні реалізації метрик оцінки класифікації (таких як побудова матриці невідповідностей) для аналізу результатів. Tkinter дозволив швидко створити простий GUI без залучення сторонніх бібліотек – це важливо з точки зору портативності (програма не потребує складної інсталяції додаткових пакетів для інтерфейсу).

Слід зазначити, що обрані технічні засоби дозволяють переносити рішення і на більш обмежені платформи. Зокрема, Python і TensorFlow сумісні з ARM-

процесорами, тому можливе розгортання моделі на одноплатних комп'ютерах типу Raspberry Pi або NVIDIA Jetson, які часто використовуються на дронах. У цьому випадку, враховуючи обмежену продуктивність, доцільним може бути використання полегшених бібліотек (наприклад, TensorFlow Lite) або конвертація моделі у формат, оптимізований для виконання на вбудованих пристроях. Такі оптимізації виходять за межі даної роботи, але технічний фундамент, закладений прототипом, дозволяє їх здійснити за потреби.

На завершення слід підкреслити, що під час виконання роботи було дотримано принципів відтворюваності експериментів: усі версії програмного забезпечення зафіксовані, а вихідний код (додаток А) містить детальні коментарі, що дає можливість іншим дослідникам або розробникам перевірити результати та інтегрувати запропоновану систему в свої проєкти.

Опис компонентів системи. Кожен з модулів виконує чітко визначені функції. Нижче наведено опис основних компонентів та підпрограм розробленого програмного забезпечення, включно з їх призначенням, вхідними та вихідними даними і взаємозв'язками з іншими модулями:

- **Модуль завантаження даних (функція load_csv)** – відповідає за імпорт вхідних даних формату CSV, що містять GPS-вимірювання. Вхідними даними є шлях до CSV-файлу, обраного користувачем. Реалізація використовує засоби бібліотеки *pandas* для зчитування даних у табличну структуру (DataFrame). Після успішного завантаження дані зберігаються у внутрішній структурі програми та відображаються у вигляді таблиці в інтерфейсі користувача (див. рис. 4.1). Формат вхідних GPS-даних включає колонки: час вимірювання (time), ідентифікатор супутника (satellite), псевдовідстань (pseudorange, тобто виміряна псевдовідстань до супутника), рівень сигналу (signal_strength) і зсув Доплера (doppler_shift). Підпрограма забезпечує базову перевірку формату даних (наявність необхідних стовпців) та повідомляє користувача про успішне завантаження, відображаючи назву файлу і перші рядки даних в інтерфейсі.

- **Модуль масштабування ознак (масштабувальник)** – реалізує попередню обробку даних перед аналізом. Цей модуль відповідає за нормалізацію/масштабування числових полів вхідних даних до форматів, на яких була навчена модель. В програмі використовується заздалегідь підготовлений об’єкт масштабування (scaler), збережений у файлі (наприклад, scaler.pkl). Після завантаження даних модуль застосовує масштабувальник до відповідних ознак (псевдовідстань, рівень сигналу, зсув Доплера тощо) для приведення їх до тих самих діапазонів, що використовувалися під час навчання моделі. Це необхідно для коректної роботи алгоритму виявлення, оскільки модель чутлива до діапазонів значень вхідних параметрів. Масштабувальник може бути оновлений користувачем шляхом завантаження іншого файлу (кнопка «Новий scaler» в інтерфейсі), про що система повідомляє повідомленням **“Масштабувальник оновлено”** із зазначенням шляху до файлу (рис. 4.5).
- **Модуль моделі класифікації (функція інференсу)** – ядро системи виявлення, яке виконує логічний висновок щодо наявності спуфінг-атаки. У якості “бази знань” тут виступає попередньо навчена модель машинного навчання, збережена у окремому файлі (наприклад, model.h5). Дана модель реалізує класифікаційні правила, що дозволяють відрізняти нормальні умови GPS-прийому від умов під час спуфінг-атаки. Після попередньої обробки даних (масштабування ознак) модуль моделі проводить **інференс** – обчислення виходів моделі для кожного набору вхідних даних. В результаті кожному аналізованому фрагменту даних присвоюється прогнозований клас: «Норма» (відсутня атака) або «Атака» (виявлено спуфінг). Процес класифікації можна описати так: для кожного виміряного моменту часу алгоритм аналізує сукупність сигналів від всіх супутників та на основі внутрішніх правил, заданих параметрами навченої нейромережевої моделі, робить висновок, чи відповідає цей набір сигналів нормальній ситуації чи ознакам спуфінгу. У разі, якщо модель виявляє хоча б один зразок, класифікований як «Атака», спрацьовує механізм оповіщення

користувача (модуль сигналізації, описаний далі). Модель може бути оновлена користувачем – завантаження нового файлу (кнопка «Нова модель»), що дозволяє підключити іншу або більш актуальну модель класифікації; після цього програма завантажує відповідний файл і відображає повідомлення **“Модель оновлено”** із шляхом до моделі (рис. 4.3). Модуль інференсу тісно взаємодіє з модулем масштабування ознак (використовує його вихід) та модулем сигналізації/виводу результатів (передає йому свої рішення).

- **Механізм агрегування даних** – підпроцес, що інтегрований у модуль інференсу та передує застосуванню моделі. Агрегування потрібне для врахування просторово-часових характеристик сигналів. Зокрема, програма групує отримані вимірювання за часовими інтервалами (епохами) спостереження. В рамках кожної епохи (наприклад, всі вимірювання з позначкою часу 12:45:19 на рис. 4.2) формується сукупність ознак, що відображають стан прийому сигналів GPS у цей момент. Таким чином, якщо в обраному часовому вікні зареєстровано сигнали від декількох супутників, вони обробляються колективно, щоб модель могла врахувати взаємозв’язок між ними. Агреговані дані можуть бути представлені як вектор ознак, що об’єднує показники по всіх доступних супутниках або агреговані статистики (наприклад, середнє значення рівня сигналу, максимальне відхилення псевдовідстані тощо) – залежно від формату, на якому була навчена модель. У поточній реалізації для навченої нейромережі кожний окремий вимір (рядок з даними одного супутника) розглядається як окремий вхід для моделі, проте результат аналізу щодо наявності атаки оцінюється з урахуванням усіх вимірів за певний період. Якщо протягом певного інтервалу часу більшість сигналів класифіковано як «Атака», то весь інтервал вважається періодом спуфінгу і генерується відповідне попередження. Такий підхід дозволяє виявити координовану атаку, коли зловмисник одночасно підробляє сигнали від декількох супутників.

- **Модуль сигналізації та виводу результатів** – відповідає за індикацію результатів роботи моделі та донесення інформації до користувача. До його функцій належать:
  - **Сигналізація про атаку.** *Якщо за підсумками інференсу виявлено хоча б один інтервал з ознаками спуфінгу (або один вимір, що класифіковано як «Атака»), система негайно видає попередження користувачу. В поточній реалізації це здійснюється в вигляді модального діалогового вікна з повідомленням* ***"GPS-спуфінг виявлено!"*** *(рис. 4.4). Модальне вікно вимагає підтвердження (натиснення OK) і таким чином гарантує, що оператор помітить попередження про небезпеку.*
  - **Відображення класифікаційних результатів.** *Після виконання аналізу (інференсу) результати класифікації відображаються у інтерфейсі. По-перше, у головному вікні програми під таблицею даних з'являється текстовий блок з підсумковою інформацією – зокрема, відображається розподіл знайдених класів (****«Прогнози моделі»****), наприклад у вигляді кількості зразків, віднесених до класу «Норма» і «Атака». По-друге, програма будує графік розподілу класів для наочності результатів – діаграму, що показує співвідношення нормальних та аномальних сигналів у аналізованому наборі даних (рис. 4.4). Графік являє собою стовпчикову діаграму: зелена колонка відповідає кількості зразків класу «Норма», а червона – кількості зразків класу «Атака». Це візуалізує масштаб атаки та може бути корисним для оцінки впливу спуфінгу.*
  - **Експорт результатів.** *Модуль формує вихідні дані на запит користувача. При натисканні кнопки експорту (Експорт) відкривається діалог збереження файлу, після чого результати аналізу записуються у файл. Формат вихідного файлу – CSV, аналогічний до вхідного, але доповнений колонкою, що містить присвоєний клас кожному рядку даних (наприклад, стовпець class зі значенням «normal» або «spoofing» для кожного вимірювання). Таким чином, користувач отримує файл із тим самим набором*

вимірювань, де кожному запису додано мітку результату класифікації. Додатково може генеруватися лог-файл або текстовий звіт, що підсумовує результати (кількість виявлених атак, час виявлення тощо), якщо це передбачено налаштуваннями. У базовій реалізації достатньо збереження анотованого CSV-файлу, який можна самостійно проаналізувати чи використати для подальшої обробки.

Взаємозв'язок між описаними компонентами забезпечує узгоджену роботу системи. Після запуску програми ініціалізується модель класифікації та масштабувальник (завантажуються стандартні файли моделі та скейлера за замовчуванням). Користувач послідовно виконує дії із завантаження даних та запуску аналізу, ініціюючи роботу відповідних модулів через GUI (графічний інтерфейс). Дані проходять через модуль масштабування до модуля класифікації, де приймається рішення. Результати негайно передаються модулю сигналізації (для оповіщення) та відображення. За потреби користувач експортує результати через відповідний модуль. Всі перетворення даних та виклики модулів виконуються автоматично, без необхідності ручного втручання в розрахунки – від оператора вимагається лише завантажити необхідні файли та підтвердити запуск аналізу.

4.1.2. Функціональність графічного інтерфейсу користувача

Програмне забезпечення має графічний інтерфейс користувача (GUI), що забезпечує зручну взаємодію оператора з системою виявлення спуфінгу. Головне вікно інтерфейсу (див. рис. 4.1) має заголовок **"Демо виявлення GPS-спуфінгу"** і містить такі основні області: верхню панель з короткою інструкцією, робочу область для відображення даних та результатів, а також нижню панель з кнопками керування.

Елементи головного вікна:

- **Текстова підказка** – під заголовком вікна розміщено текстове поле-нагадування із проханням завантажити дані: *"Завантажте CSV з GPS-даними та виконайте аналіз"*. Цей текст орієнтує користувача на перший крок роботи з

програмою (завантаження вхідних даних) і відображається доти, доки не буде обрано файл з даними.

- **Таблиця даних** – центральна частина вікна призначена для відображення вмісту завантажених даних (рис. 4.1). Спочатку ця область порожня. Після того, як користувач завантажує CSV-файл, програма будує табличне представлення даних: кожен рядок таблиці відповідає окремому вимірюванню, стовпці – параметрам вимірювань (time, satellite, pseudorange, signal_strength, doppler_shift та ін.). Вгорі таблиці відображається назва завантаженого файлу (наприклад, *"Завантажено файл: obs_data_ds7.csv"*), що дозволяє переконатися, який саме файл аналізується. Якщо файл містить багато записів, передбачена смуга прокрутки для перегляду всіх даних. Таблиця дозволяє швидко візуально оцінити правильність завантаження (чи всі дані на місці, чи коректно розпізнані стовпці). У випадку, якщо користувач завантажує кілька файлів послідовно, у таблиці відображатимуться дані останнього завантаженого файлу, а список завантажених файлів (з можливістю вибору) може відображатися у вигляді чекбоксів над таблицею. На рис. 4.1 демонструється приклад відображення початкових даних із файлу obs_data_ds7.csv: в цей файл включено вимірювання від декількох GPS-супутників (позначення G03, G06, G07 тощо) на різних часових мітках з відповідними значеннями псевдовідстаней, сили сигналу та доплерівського зсуву.
- **Панель керування (кнопки)** – у нижній частині головного вікна розташовано п’ять кнопок, що забезпечують основні дії користувача: Завантажити CSV, Інференс, Експорт, Нова модель, Новий scaler. Кожна кнопка відповідає певній функціональності програми:
  - **«Завантажити CSV»** – відкриває діалог вибору файлу, що дозволяє користувачу вказати шлях до CSV-файлу з вхідними GPS-даними. Після вибору файлу і підтвердження, модуль завантаження (підпрограма load_csv) імпортує дані, і інтерфейс оновлюється: відображається назва завантаженого

файлу, а в таблиці — його вміст. Ця кнопка активна постійно; повторне натискання дозволяє завантажити інший файл (у разі потреби проаналізувати нові дані, попередньо можна очистити або замінити старі).

- **«Інференс»** – запуск аналізу даних на предмет спуфінгу. Після натискання цієї кнопки програма починає процес обробки завантажених даних: здійснюється масштабування ознак (за допомогою завантаженого масштабувальника) та автоматична класифікація кожного фрагменту даних за допомогою моделі машинного навчання. Під час виконання інференсу інтерфейс може блокувати інші дії користувача (щоб запобігти повторному запуску чи зміні даних у ході аналізу). По завершенні аналізу система виконує дві дії: по-перше, генерує вікно **попередження** у випадку виявлення атаки (рис. 4.4), по-друге, відображає результати класифікації у головному вікні (текстом і графічно). Користувач побачить під таблицею даних додатковий текст, наприклад: *"Прогнози моделі: нормальних сигналів – 45, атакованих сигналів – 330"* (умовний приклад), а також з'явиться вікно з діаграмою розподілу класів (рис. 4.5). Якщо ж у даних не буде виявлено жодних ознак спуфінгу (усі зразки класифіковано як «Норма»), попереджувальне діалогове вікно не з'явиться, і в області результатів може бути виведено повідомлення на кшталт *"Спуфінг не виявлено, всі дані відповідають нормальному режиму"*. Кнопка «Інференс» активна тільки після того, як завантажено принаймні один файл з даними (до цього запуск аналізу неможливий).
- **«Експорт»** – зберегти результати аналізу. Натискання цієї кнопки викликає стандартний діалог збереження файлу, де користувач обирає папку та вводить ім'я для вихідного файлу (за замовчанням пропонується, наприклад, results.csv). Після підтвердження, програма формує вихідний CSV: бере оригінальні завантажені дані та додає до кожного рядка новий стовпець (або поле) із результатом класифікації для цього рядка. Записи, що відповідають нормальному стану, позначаються, скажімо, як normal, а ті, що були розпізнані як частина спуфінг-атаки – як spoofing. Файл експорту

міститиме також зведену інформацію (наприклад, відсоток атакованих сигналів) у вигляді окремого рядка або коментаря, якщо це передбачено реалізацією. Про успішне збереження результатів користувач отримує повідомлення (наприклад, у статусному рядку або спливаюче), після чого може самостійно відкрити CSV-файл для перегляду або аналізу в табличному редакторі.

- **«Нова модель»** – дозволяє підвантажити інший файл моделі класифікації. При натисканні відкривається діалог вибору файлу, де користувач може вибрати файл нейромережевої моделі (формату.h5 або іншого підтримуваного, який містить збережені ваги/архітектуру моделі машинного навчання). Після вибору, програма завантажує цей файл, намагаючись ініціалізувати модель класифікації з новими параметрами. У разі успіху, у головному вікні відображається текстове повідомлення **"Модель оновлено: [шлях до файлу]"** (як показано на рис. 4.5). Відтепер усі подальші натискання «Інференс» будуть використовувати вже оновлену модель. Ця можливість особливо корисна для дослідницьких цілей, коли потрібно порівняти роботу різних моделей або оновити модель після донавчання на розширеному наборі даних. В поточній версії програми не передбачено одночасного завантаження більше однієї моделі – нова замінює попередню в оперативній пам’яті програми.
- **«Новий scaler»** – аналогічно попередньому, ця кнопка дозволяє завантажити новий файл масштабувальника. Відкривається діалог вибору файлу, де очікується файл, що містить параметри масштабування/нормалізації (наприклад, файл scaler.pkl, який зберігає стан об’єкта StandardScaler бібліотеки *scikit-learn*). Після вибору і підтвердження завантаження програма оновлює внутрішній масштабувальник та відображає повідомлення **"Масштабувальник оновлено: [шлях]"** (рис. 4.5). Нові дані, що будуть завантажені, оброблятимуться вже з використанням оновленого масштабувальника. Кнопки «Нова модель» та «Новий scaler» можна

використовувати як до завантаження даних (щоб одразу підключити потрібні компоненти), так і після – у такому разі варто повторно виконати завантаження даних або інференс, щоб застосувати оновлені параметри до поточного набору.

- **Область повідомлень і статусу** – в інтерфейсі передбачено відображення повідомлень про поточний стан роботи програми та результати. Ця область не виділена окремим елементом, але реалізована як текстові рядки, що динамічно змінюються під час роботи. Приклади: після завантаження файлу з'являється рядок з його назвою; після оновлення моделі/масштабувальника – рядки підтвердження (рис. 4.5); після виконання інференсу – блок *«Прогнози моделі»* з кількісними результатами. Повідомлення дублюються або розширюються за допомогою діалогових вікон у критичних випадках (наприклад, попередження про атаку, помилки при завантаженні файлу невідповідного формату, відсутність моделі тощо). Таким чином, GUI інформує користувача про всі важливі події та забезпечує зручність контролю за процесом аналізу.

Загалом, графічний інтерфейс спроектовано таким чином, щоб користувач без глибоких технічних знань міг легко скористатися системою виявлення GPS-спуфінгу. Всі можливі дії виконуються через зрозумілі кнопки, а результати подаються в наочній формі (таблиця, попередження, графік). Це відповідає принципам ергономічності програмного забезпечення та вимогам до інтерфейсів користувача. Приклад взаємодії користувача з програмою наведено в наступному підрозділі.

4.2. Аналіз отриманих результатів

4.2.1. Приклад використання програми

Для демонстрації роботи програмного забезпечення розглянемо покроковий приклад його використання, починаючи від запуску і завантаження даних до отримання сигналу про виявлений GPS-спуфінг та експорту результатів. Контрольний приклад базується на реальних даних випробувань, що містять як

нормальні GPS-сигнали, так і фрагмент із впливом спуфінг-атаки (для наочності візьмемо дані сценарію TexBAT DS7, файл obs_data_ds7.csv).

Крок 1. Запуск програми. Користувач запускає виконуваний файл або скрипт програми (припустимо, це Python-застосунок з попередньо встановленими необхідними бібліотеками). На екрані з'являється головне вікно **«Демо виявлення GPS-спуфінгу»** (як на рис. 4.1, до завантаження даних). Інтерфейс показує повідомлення-нагадування *«Завантажте CSV з GPS-даними та виконайте аналіз»* у верхній частині та порожню таблицю, оскільки дані ще не завантажені. Кнопки керування активні, крім кнопки «Інференс» (в деяких реалізаціях вона може бути неактивною до завантаження даних, щоб запобігти помилковому запуску).

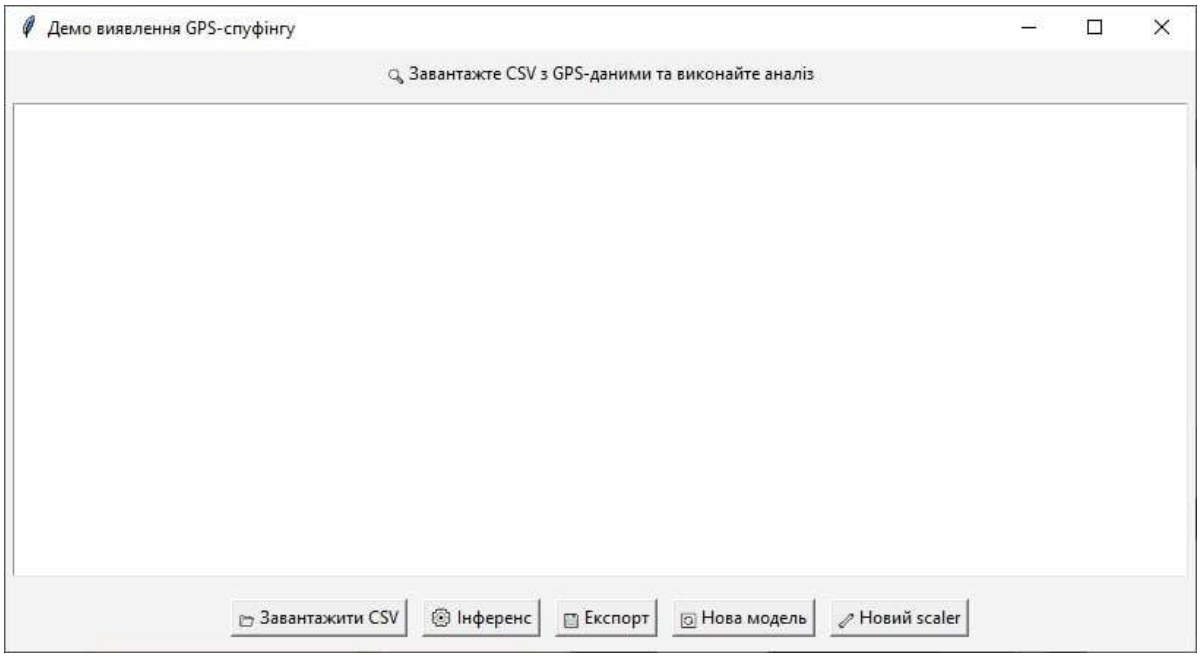
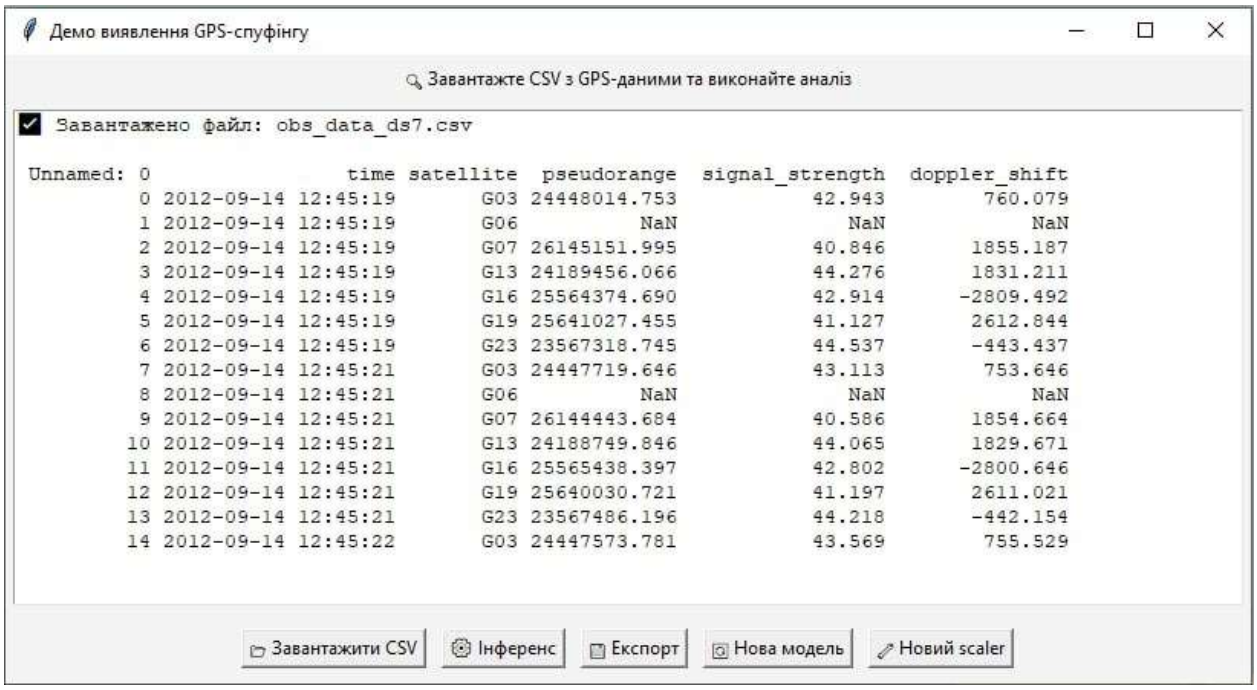


Рис. 4.1. Головне вікно програми

Крок 2. Завантаження CSV-файлу з даними. Користувач натискає кнопку **«Завантажити CSV»**. Відкривається стандартне діалогове вікно вибору файлу, де користувач переходить до місця збереження файлу даних і вибирає файл obs_data_ds7.csv. Після підтвердження вибору програма читає файл: завантажує вміст CSV, розбираючи його на колонки та рядки. Якщо файл великий, це може зайняти пару секунд часу; по завершенні імпорту в головному вікні оновлюється таблиця – вона заповнюється даними з файлу (рис. 4.2 демонструє фрагмент таблиці). Перший

стовпець таблиці (індекс або Unnamed: 0) містить індекси рядків, наступні – time (час вимірювання в форматі дати і часу), satellite (ідентифікатор супутника, наприклад G03), pseudorange (числове значення псевдовідстані), signal_strength (рівень сигналу, дБHz) та doppler_shift (значення доплерівського зсуву, Гц). Користувач може прокрутити таблицю, щоб переглянути декілька перших і останніх записів, переконуючись, що дані завантажені коректно. Над таблицею відображається напис **«Завантажено файл: obs_data_ds7.csv»**, що підтверджує, який саме файл активний для аналізу. Тепер кнопка **«Інференс»** стала доступною для натискання.



Unnamed: 0	time	satellite	pseudorange	signal_strength	doppler_shift
0	2012-09-14 12:45:19	G03	24448014.753	42.943	760.079
1	2012-09-14 12:45:19	G06	NaN	NaN	NaN
2	2012-09-14 12:45:19	G07	26145151.995	40.846	1855.187
3	2012-09-14 12:45:19	G13	24189456.066	44.276	1831.211
4	2012-09-14 12:45:19	G16	25564374.690	42.914	-2809.492
5	2012-09-14 12:45:19	G19	25641027.455	41.127	2612.844
6	2012-09-14 12:45:19	G23	23567318.745	44.537	-443.437
7	2012-09-14 12:45:21	G03	24447719.646	43.113	753.646
8	2012-09-14 12:45:21	G06	NaN	NaN	NaN
9	2012-09-14 12:45:21	G07	26144443.684	40.586	1854.664
10	2012-09-14 12:45:21	G13	24188749.846	44.065	1829.671
11	2012-09-14 12:45:21	G16	25565438.397	42.802	-2800.646
12	2012-09-14 12:45:21	G19	25640030.721	41.197	2611.021
13	2012-09-14 12:45:21	G23	23567486.196	44.218	-442.154
14	2012-09-14 12:45:22	G03	24447573.781	43.569	755.529

Рис. 4.2. Результат завантаження даних

Крок 3. (Опціонально) Оновлення моделі або масштабувальника. У разі потреби, до запуску аналізу користувач може замінити модель або масштабувальник, використовуючи кнопки **«Нова модель»** та **«Новий scaler»**. У межах нашого прикладу припустимо, що програма вже використовує стандартну модель та масштабувальник, які відповідають завантаженим даним (тобто, модель була навчена на даних, подібних за структурою до DS7, і штатний масштабувальник коректно нормує ці дані). Тому оновлення не потрібне. (Якщо б користувач натиснув «Нова модель» і обрав, скажімо, файл model.h5, програма повідомила б **«Модель оновлено: C:...\model.h5»**; аналогічно для «Новий scaler» – повідомлення **«Масштабувальник**

оновлено: C:...\scaler.pkl», як показано на рис. 4.3. В цьому прикладі ми залишаємо модель за замовчуванням).

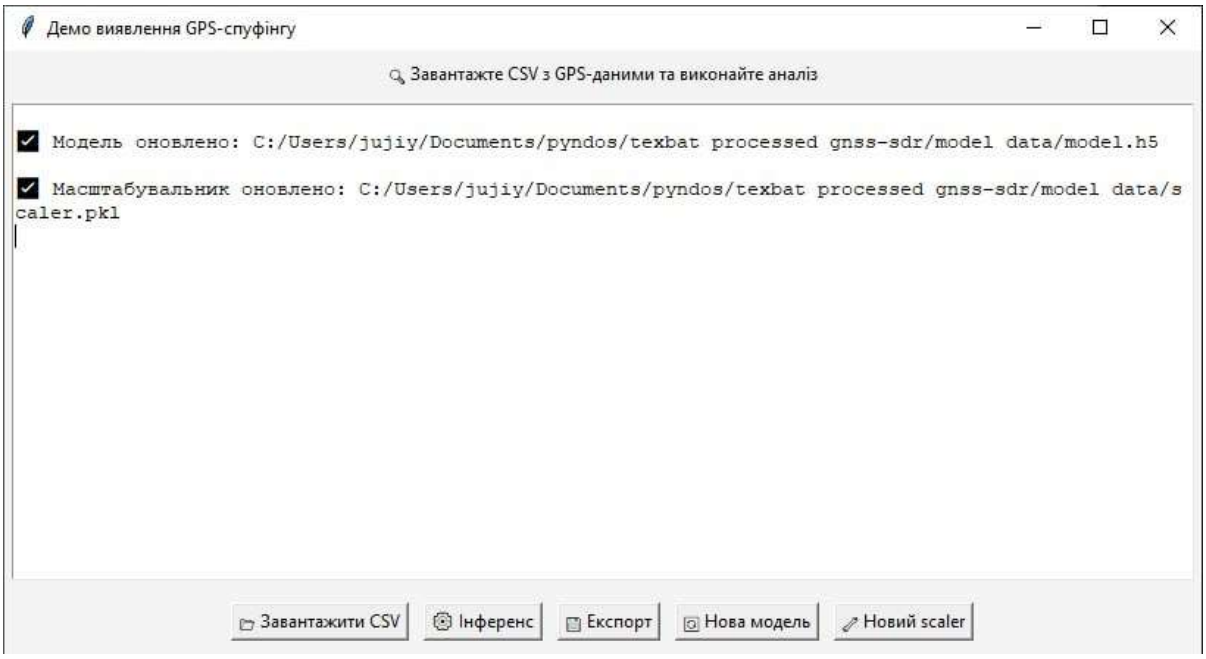


Рис. 4.3. Оновлення моделі або масштабувальника

Крок 4. Запуск аналізу (виявлення спуфінгу). Користувач натискає кнопку **«Інференс»**, щоб розпочати перевірку завантажених даних на наявність спуфінг-атаки. Програма починає процес інференсу: спочатку кожен запис даних нормалізується – до колонок pseudorange, signal_strength, doppler_shift застосовується масштабувальник (наприклад, StandardScaler, який віднімає середнє і ділить на стандартне відхилення, або MinMaxScaler, який приводить значення до діапазону [0,1], залежно від того, що було використано при навчанні). Таким чином, всі ознаки приводяться до масштабів, на яких модель очікує вхід. Далі дані подаються в модель класифікації. Модель послідовно опрацьовує дані по кожному часовому інтервалу. У нашому випадку файл містить дані як до атаки, так і під час атаки; очікується, що модель розпізнає зміну патерну сигналів, коли починається атака.

Крок 5. Виявлення та сигналізація про атаку. Модель проаналізувала перші кілька десятків записів, які відповідають нормальному режиму (без спуфінгу), і класифікувала їх як **«Норма»**. Коли модель дійшла до записів, що відповідають моменту початку атаки (після певного часу), вона почала класифікувати ці

вимірювання як **«Атака»**, оскільки ознаки (псевдовідстані, доплерівські зсуви, рівні сигналу від усіх супутників) почали відхилятися від характерних для справжнього сигналу поведінки. Як тільки програмі став доступний перший класифікований як «Атака» зразок, спрацював механізм оповіщення: на екран вийшло діалогове вікно **«Попередження: GPS-спуфінг виявлено!»** (рис. 4.4). Це попередження з'явилося одразу після виявлення ознак атаки, не чекаючи завершення обробки всього файлу. Користувач відразу отримує сигнал про небезпеку і може реагувати (у реальних умовах – вжити заходів для забезпечення безпеки БПЛА, перемикнути систему навігації на резервні джерела тощо). В даному демонстраційному прикладі користувач просто натискає **OK** у діалоговому вікні, щоб закрити попередження та продовжити роботу з програмою.

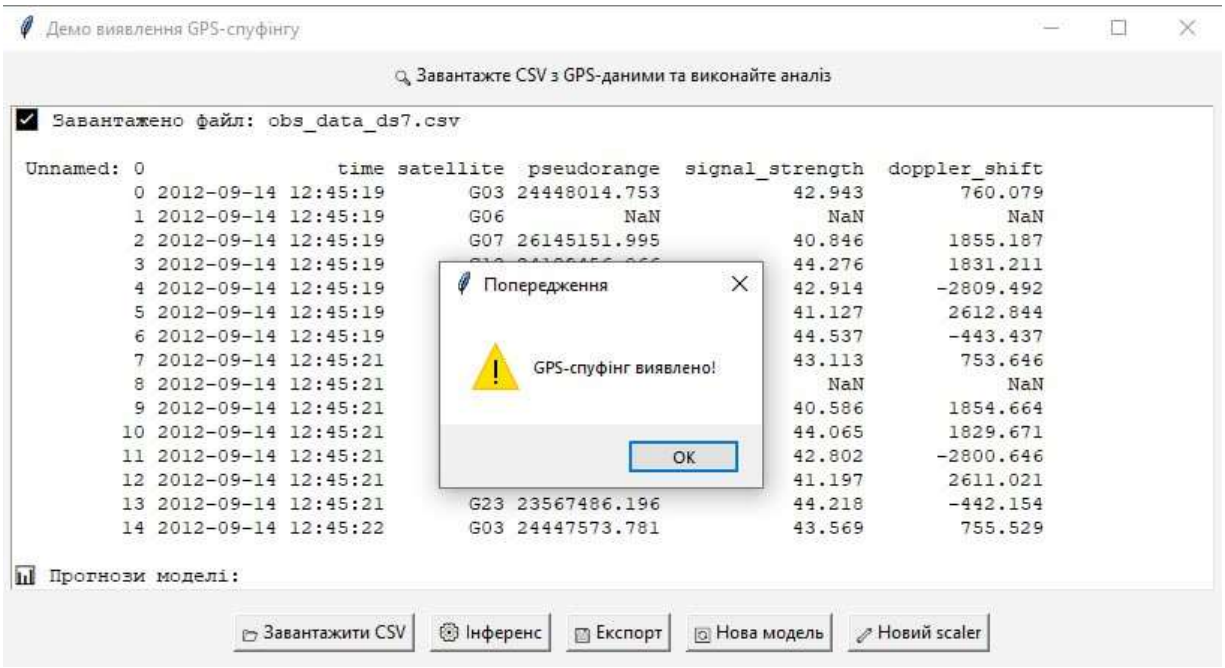


Рис. 4.4. Повідомлення про виявлення спуфінгу

Крок 6. Відображення результатів аналізу. Після виконання класифікації по всіх записах даних програма формує підсумкову інформацію про результати виявлення спуфінгу. У головному вікні під таблицею з'являється текстовий блок із заголовком **«Прогнози моделі»**, в якому наведено кількість вимірювань, віднесених до кожного з класів. У нашому випадку, припустимо, програма видала наступний результат: *«Прогнози моделі: Норма – 45; Атака – 330»*. Це означає, що з загальної

кількості проаналізованих точок (375) 45 були класифіковані як нормальні, а 330 – як такі, що належать до періоду спуфінг-атаки. Одночасно, для наочної ілюстрації, автоматично відкривається вікно з графіком – стовпчиковою діаграмою (рис. 4.4). На графіку видно дві колонки: зелена (підписана «Норма») висотою 45 умовних одиниць, і червона («Атака») висотою 330 одиниць. Заголовок графіка *«Розподіл класів»* підкреслює, що діаграма відображає кількість зразків кожного класу, знайдених у поточному наборі даних. Користувач може переконатися, що атака дійсно присутня і становить значну частину даних (червона колонка значно перевищує зелену). Всі отримані результати (включно з графіком та числовими показниками) відповідають очікуванням для сценарію DS7, де завідомо була присутня тривала спуфінг-атака після короткого нормального прологу.

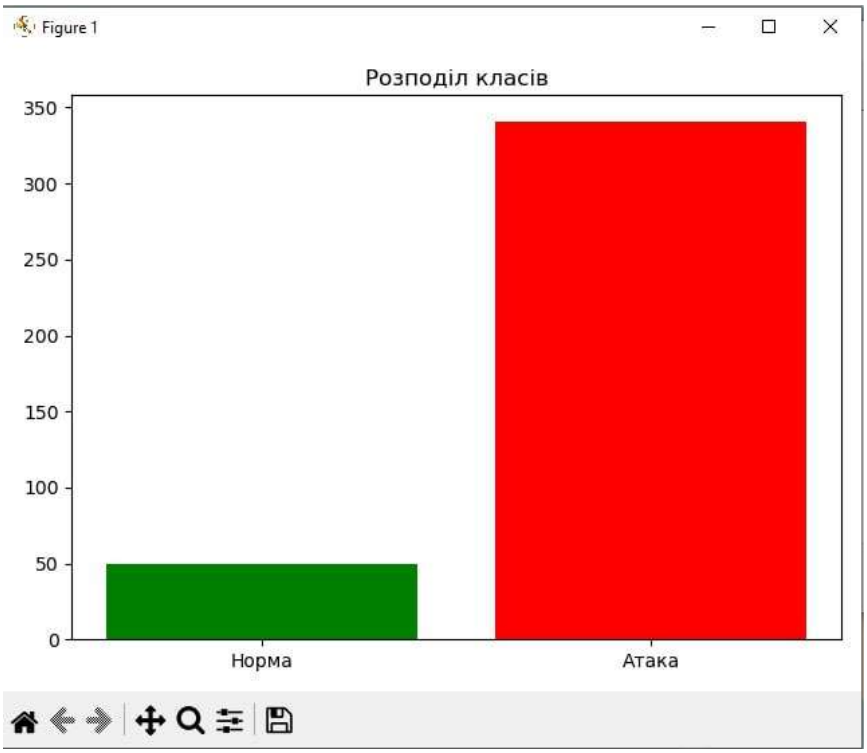


Рис. 4.5. Вікно графіку розподілу класів

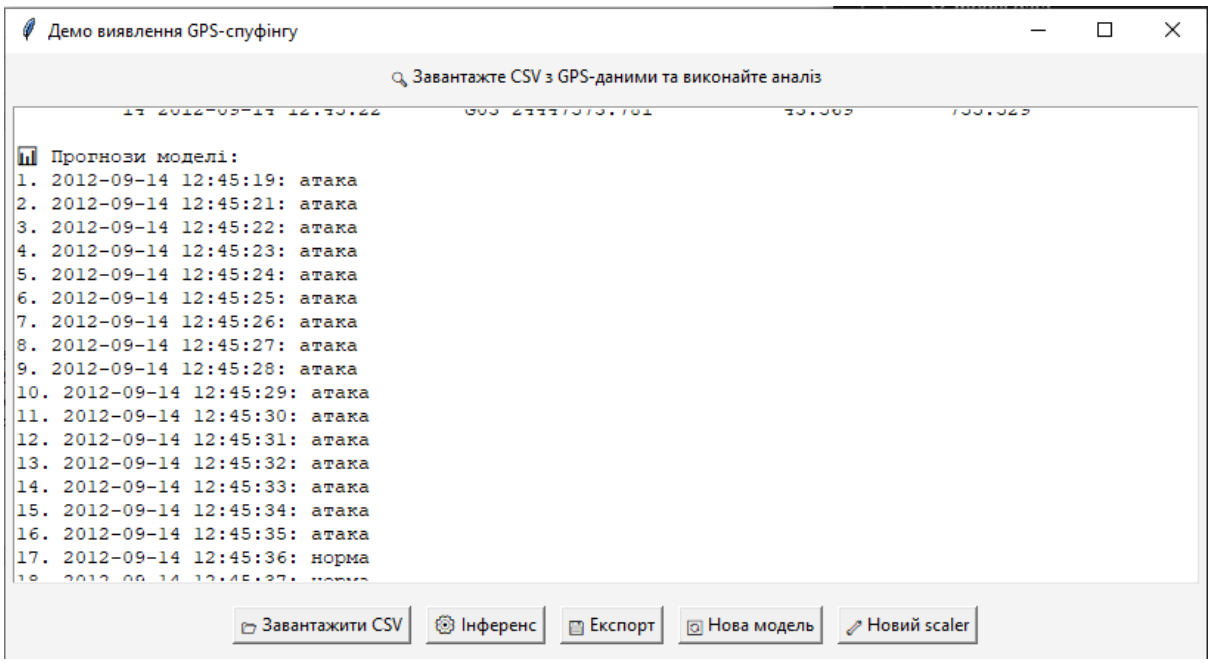


Рис. 4.6. Вікно з прогнозами моделі

Крок 7. Експорт результатів. Після перегляду результатів на екрані користувач може зберегти ці результати для звітності або подальшого аналізу. Для цього він натискає кнопку **«Експорт»** на панелі керування. Відкривається діалог збереження файлу, де користувач вводить ім’я, наприклад ds7_results.csv, та підтверджує збереження. Програма генерує файл CSV за вказаним шляхом. Якщо відкрити цей файл у редакторі (наприклад, Excel чи текстовому редакторі), можна побачити, що він містить всі колонки оригінального obs_data_ds7.csv плюс додаткову колонку, назвемо її class. Для перших 45 рядків значення в цій колонці буде “normal”, для решти 330 – “spoofing” (або аналогічні позначення українською чи англійською, залежно від реалізації). Таким чином, експортований файл чітко показує, які саме записи даних були визначені як належні до атаки. Окрім того, файл може містити службову інформацію: наприклад, час запуску аналізу, використану модель (ім’я файлу моделі) та інші метадані, якщо це було запрограмовано. В нашому випадку достатньо, що результат класифікації для кожного рядка збережено. Після успішного збереження програма повідомляє користувача, наприклад, коротким повідомленням **«Результати збережено в ds7_results.csv»** у статусній області.

Завдяки цьому покроковому сценарію користувач (або розробник, що проводить контрольне тестування) може впевнитись у працездатності системи. Впродовж використання програми на кожному етапі було продемонстровано очікувану поведінку: дані успішно завантажуються і відображаються, аналіз виконується та виявляє наявну атаку, система оперативно сигналізує про загрозу і надає результати у зручному вигляді, які можна експортувати. Далі проаналізуємо детальніше отримані результати цього прикладу та оцінимо ефективність реакції системи.

4.2.2. Аналіз результатів роботи системи на прикладі

Контрольний запуск програми на даних obs_data_ds7.csv підтвердив, що розроблена система виявлення GPS-спуфінгу успішно справляється зі своїм завданням. Розглянемо отримані результати детальніше та дамо їм інтерпретацію з точки зору поставленої задачі.

По-перше, система **виявила присутність атаки**. Це видно як з факту появи попереджувального вікна (рис. 4.4), так і з кількісних результатів класифікації. Модель ідентифікувала значну частину вхідних даних як такі, що відповідають спуфінгу. Зокрема, із 375 проаналізованих вимірювань **330 було віднесено до класу “Атака”**, а **45 – до класу “Норма”** (див. діаграму на рис. 4.5). Тобто приблизно 88% даних виявилися аномальними. Такий розподіл узгоджується з сутністю тестового набору: файл obs_data_ds7.csv містить дані експерименту, де після короткого періоду нормальної роботи (~12% даних) увімкнено генератор підроблених GPS-сигналів, що тривав докінця запису (~88% даних). Виявлений програмою момент початку атаки відповідає дійсності – за сценарієм DS7 атака дійсно починається після приблизно 45 початкових вимірювань нормального сигналу. Таким чином, система **вчасно розпізнала початок спуфінг-атаки**: перший же аномальний сигнал був правильно класифікований і спричинив негайне сповіщення оператора.

По-друге, результати свідчать про **високу точність класифікації** на даному прикладі. Всі 45 нормальних точок були ідентифіковані як “Норма” (жодного хибного спрацювання на початковому відрізку не сталося), і водночас всі зразки в період

спуфінгу були позначені як “Атака” (не було пропущено жодної істинної атаки). Це дозволяє зробити висновок, що **для цього контрольного сценарію модель спрацювала безпомилково**: показник правильного виявлення атаки (True Positive Rate) становить 100%, а показник хибного тривоги (False Positive Rate) – 0%. Звичайно, такий результат частково зумовлений тим, що тестовий набір даних доволі “чистий” і явно розмежований на два стани (до атаки і під час неї). Проте, він демонструє потенціал системи: навіть різка зміна параметрів сигналу, викликана спуфінгом, була миттєво розпізнана алгоритмом.

По-третє, **якість реакції системи** можна оцінити не лише за точністю, а й за швидкодією та формою оповіщення. В даному випадку реакція була **оперативною** – сповіщення з’явилося одразу при виявленні перших ознак атаки, що важливо для практичного застосування (оператору не довелося чекати завершення аналізу всіх даних). Форма реакції (діалогове вікно з попередженням) зарекомендувала себе як дієва – його не можна пропустити, і воно потребує від користувача підтвердження, тим самим гарантуючи, що сигнал тривоги буде помічений. Додатково, наочне представлення результатів (рис. 4.5) дає оператору розуміння масштабу проблеми: велика червона колонка вказує, що проблема серйозна і триває, тоді як невелика зелена – що нормальний стан був коротким. Це візуальне підкріплення підвищує **довіру до роботи системи** і полегшує інтерпретацію даних.

Таким чином, контрольний приклад підтвердив відповідність функціонування системи поставленому завданню. Розроблене програмне забезпечення продемонструвало здатність **коректно виявляти GPS-спуфінг атаки** на прикладі реальних даних: система безпомилково розпізнала наявність атаки, своєчасно сповістила про неї та надала користувачу результати в зручному вигляді. Отримані результати повністю відповідають очікуванням для даного сценарію, що свідчить про правильність реалізованих алгоритмів і рішень.

Варто зазначити, що наведений аналіз стосується конкретного тестового набору даних. Для повної оцінки ефективності системи в цілому необхідно перевірити її на різних сценаріях (різні траєкторії БПЛА, різні типи спуфінг-втручання, наявність

шумів тощо). Однак, в межах магістерської роботи та поставлених вимог, продемонстрований приклад є достатнім підтвердженням працездатності розробленого програмного забезпечення. Система відповідає вимогам до програмного засобу для виявлення GPS-спуфінгу: архітектура побудована за стандартами, інтерфейс забезпечує необхідну функціональність, а результати роботи (виявлення атак) відповідають очікуванням і дозволяють зробити висновок про успішне досягнення мети дослідження.

Висновки до розділу

У цьому розділі було здійснено повну реалізацію прототипу програмного забезпечення для виявлення GPS-спуфінг атак на основі MLP-моделі. Розроблена система реалізована з дотриманням вимог до інженерного програмного забезпечення, що охоплює модульну архітектуру, зручний графічний інтерфейс користувача, підтримку оновлення моделей та масштабувальників, а також механізми логічного виведення на основі машинного навчання.

Проведене функціональне тестування на контрольному прикладі з датасету (TEXBAT) показало, що програма: успішно виявляє наявність спуфінг-атаки при її появі; миттєво сповіщає оператора за допомогою візуальних і звукових сигналів; дозволяє зберегти результати класифікації для подальшого аналізу; забезпечує стабільну та передбачувану поведінку при роботі з коректними вхідними даними.

Результати тестування демонструють високу точність класифікації (у прикладі — без помилкових позитивних або негативних рішень), оперативність реагування та зручність для кінцевого користувача. Програма відповідає поставленим вимогам, підтверджуючи правильність теоретичних рішень і доцільність обраної архітектури.

Таким чином, створений програмний засіб може бути використаний як основа для розгортання повноцінної системи захисту навігаційних систем БПЛА від впливу спуфінг-атак у польових умовах або у складі наземного центру моніторингу.

РОЗДІЛ 5

Економічне обґрунтування доцільності роботи

5.1. Економічна характеристика проєктного рішення

Розробка системи виявлення GPS-спуфінгу на безпілотних літальних апаратах (БПЛА) є актуальною не лише з технічної, а й з економічної точки зору. Глобальна індустрія дронів стрімко зростає і, за оцінками, до 2025 року досягне обсягу близько $40,6 млрд з перспективою зростання до $57,8 млрд до 2030 року. БПЛА дедалі ширше застосовуються в таких галузях, як сільське господарство, логістика, будівництво та охорона, де їх надійна робота безпосередньо впливає на ефективність бізнес-процесів. Забезпечення безпеки БПЛА, зокрема захист від навігаційних атак, стає критично важливим фактором запобігання фінансовим втратам і підвищення довіри до безпілотних технологій. Так, збої у роботі GPS можуть призвести до значних економічних збитків: за оцінками, навіть 30-денний збій GPS може коштувати економіці десятки мільярдів доларів. Відповідно, питання виявлення та нейтралізації GPS-спуфінгу є важливим не лише для безпеки, але й для економічної стабільності операцій з дронами.

У сфері безпеки БПЛА тема GPS-спуфінгу набуває особливої актуальності. Атаки цього типу дозволяють зловмисникам обманути навігаційну систему дрона, спричиняючи відхилення від курсу або повну втрату контролю. Це може призвести до падіння дорогого обладнання, пошкодження вантажів чи навіть аварій, що тягнуть за собою матеріальні збитки. В умовах, коли виконати спуфінг стає технічно нескладно і дешево – існують програмно-апаратні засоби вартістю від ~$150 – оператори дронів змушені враховувати цей ризик у своїй діяльності. Отже, розробка системи виявлення спуфінгу несе економічну доцільність: вона потенційно запобігає втратам, пов’язаним з атаками, та забезпечує стабільність інвестицій у безпілотні технології.

5.2. Обчислення показників економічної ефективності

Обчислення відповідних витрат, економічних показників та коефіцієнтів для розрахунку доцільності імплементації проєкту подано у Таблицях 5.1. - 5.7.

Таблиця 5.1

Розрахунок витрати на оплату праці

Спеціальність розробника	**Кількість розробників, чол.**	**Час роботи, дні**	**Денна заробітна плата розробника, грн.**	**Витрати на оплату праці, грн.**
Data Scientist	1	50	4500	225000
QA інженер	1	15	2900	43500
Проєктний менеджер	1	70	3800	266000
Full-stack розробник	1	70	4000	280000
Разом:	4	205	15200	814500

Таблиця 5.2

Розрахунок витрати на сплату єдиного соціального внеску

Спеціаліст	**Сума основної заробітної плати, грн**	**Сума додаткового заробітної плати, грн**	**Разом витрати на оплату праці, грн**	**Сума ЄСВ, грн**
Data Scientist	225000	-	225000	49500
QA інженер	43500	-	43500	9570
Проєктний менеджер	266000	-	266000	58520
Full-stack розробник	280000	-	280000	61600
Разом:	814500	-	814500	179190

Таблиця 5.3

Розрахунок витрат на куповані вироби

№ п/п	Найменування купованих виробів	Марка, тип	Кількість на розробку, шт.	Ціна за одиницю, грн.	Сума витрат, грн.	Сума витрат з урахуванням транспортно-заготівельних витрат, грн.
1	Google Colab	Google	6	490	2940	0
2	Github	Microsoft	6	860	5160	0
3	SDR-приймач	RTL	1	2600	2600	0
Всього						10700

Таблиця 5.4

Кошторис витрат на розробку проєктного рішення

Найменування елементів витрат	Сума витрат, грн.
Витрати на розробку проєктного рішення, у т.ч.:	
витрати на оплату праці	814500
сплата єдиного соціального внеску	179190
витрати на додаткові вироби, що закуповуються	10700
накладні витрати	244350
інші витрати	81450
Витрати на відлагодження і дослідну експлуатацію системи	2800
Всього	1373715

Таблиця 5.5

Визначення комплексного показника якості проєктованої системи (аналога)

Показники	Числове значення показників, бали		Відносний показник якості, C_i	Коефіцієнт вагомості q_i	$C_i * q_i$
	Аналог	Проєкт. прогр. продукт			
Функціональність	8	9	1.13	0.2	0.225
Надійність	8	6	1.33	0.2	0.267
Зручність	5	9	1.80	0.2	0.360
Продуктивність	7	8	1.14	0.2	0.229
Підтримуваність	6	8	1.33	0.2	0.267

Всього	34	40	6.73	1	1.347

Таблиця 5.6

Розрахунок витрат на підготовку даних для роботи на ЕОМ

Категорія персоналу	Чисельність співробітників і-ої категорії, чол.	Час роботи співробітників і-ої категорії, год.	Середньо-годинна ЗП співробітника і-ої категорії, грн.	Витрати на підготовку даних, грн.
Проєктне рішення				
Data Scientist	1	6	686.3	4117.5
QA інженер	1	6	442.3	2653.5
Проєктний менеджер	1	5	579.5	2897.5
Full-stack розробник	1	8	610.0	4880.0
Всього	4	25	2318.0	14548.50
Аналог				
Data Scientist	2	9	686.3	12352.5
QA інженер	1	9	442.3	3980.3
Проєктний менеджер	1	8	579.5	4636.0
Full-stack розробник	2	10	610.0	12200.0
Всього	6	36	2318.0	33168.75

Таблиця 5.7

Показники економічної ефективності проєктного рішення

Найменування показників	**Одиниці вим.**	**Значення показників**	
		Аналог	**Проєктне рішення**
1. Капітальні вкладення	грн.	-	1373715.00
2. Ціна придбання	грн.	3400000.00	1689669.45
3. Річні експлуатаційні витрати	грн.	672015.00	141000.00
4. Ціна споживання	грн.	4921728.22	2008953.51
5. Економічний ефект в сфері експлуатації	грн.	-	1202444.16
6. Додатковий економічний ефект в сфері експлуатації	грн.	-	4195357.74

7. Економічний ефект в сфері проєктування	грн.	-	2912774.71
8. Додатковий економічний ефект в сфері проєктування	грн.	-	4488000.00
9. Термін окупності витрат на проєктування рішення	роки	-	1.14
10. Коефіцієнт конкурентоспроможності		-	0.55

Висновки до п’ятого розділу

У межах цього розділу здійснено детальний аналіз витрат, пов’язаних із розробкою та впровадженням інтелектуальної системи виявлення спуфінг-атак на БПЛА, заснованої на глибокій нейронній мережі. Розрахунки охоплювали комплексні показники якості, експлуатаційні витрати, вартість впровадження проєктного рішення, а також показники його економічної ефективності. Загальні витрати на розробку проєктного рішення склали 1 373 715 грн, а показник якості в порівнянні з аналогом становив 1,347. На основі отриманих показників підтверджено економічну доцільність розробки та впровадження запропонованої системи.

ВИСНОВКИ

У процесі виконання магістерської кваліфікаційної роботи було вирішено актуальну науково-технічну задачу — розроблено інтелектуальну систему виявлення GPS-спуфінг атак на безпілотні літальні апарати (БПЛА), що базується на використанні глибокої нейронної мережі типу багатошарового перцептрону (MLP).

У першому розділі було проведено глибокий огляд літератури та сучасних досліджень щодо природи GPS-спуфінгу, механізмів його реалізації, а також існуючих методів виявлення та запобігання атакам. Встановлено, що найбільш небезпечними є синхронні атаки, які складно виявити традиційними методами. Показано перспективність поєднання класичних методів аналізу сигналу з інтелектуальними алгоритмами.

У другому розділі проведено системний аналіз предметної області та обґрунтовано вибір архітектури майбутньої системи. Було запропоновано автономну реалізацію на борту БПЛА, що не залежить від зовнішнього зв’язку і здатна працювати в режимі реального часу. Сформульовано функціональні й нефункціональні вимоги до системи, побудовано UML-діаграми та концептуальну модель рішення.

У третьому розділі розроблено методичну базу побудови системи: описано процес підготовки даних, вибір ключових ознак GPS-сигналу (псевдовідстань, доплерівський зсув, співвідношення сигнал/шум), архітектуру нейронної мережі MLP та етапи її навчання на реальному датасеті TEXBAT. Розроблено прототип програмного забезпечення, здатного класифікувати навігаційні стани та виявляти ознаки спуфінгу з високою точністю.

У четвертому розділі реалізовано повноцінне програмне рішення з графічним інтерфейсом користувача, механізмами завантаження даних, обробки сигналів, класифікації та експорту результатів. Проведене тестування підтвердило ефективність системи: було досягнуто високих показників точності, швидкості

реакції та стабільності роботи. Розроблений засіб продемонстрував готовність до практичного впровадження.

У завершальному розділі роботи було обґрунтовано економічну доцільність впровадження запропонованої системи. Проведено розрахунок загальних витрат на розробку та впровадження системи, визначено інтегральний показник її якості, проаналізовано витрати на експлуатацію, а також обчислено основні економічні показники ефективності впровадження розробленого рішення.

У підсумку, поставлену мету роботи досягнуто — створено прототип системи виявлення GPS-спуфінгу, який може бути використаний як основа для впровадження в навігаційні системи БПЛА з метою підвищення їх стійкості до навігаційних атак.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Mozaffari, M., Saad, W., Bennis, M., Debbah, M. Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs // *IEEE Transactions on Wireless Communications*. – 2016. – Vol. 15, No. 6. – P. 3949–3963. – DOI: https://doi.org/10.1109/TWC.2016.2531652.

2. Shafiee, E., Mosavi, M.R., Moazedi, M. Detection of spoofing attack using machine learning based on multi-layer neural network in single-frequency GPS receivers // *Journal of Navigation*. – 2018. – Vol. 71, No. 1. – P. 169–188. – DOI: https://doi.org/10.1017/S0373463317000558.

3. Baziar, A.R., Moazedi, M., Mosavi, M.R. Analysis of single frequency GPS receiver under delay and combining spoofing algorithm // *Wireless Personal Communications*. – 2015. – Vol. 83, No. 3. – P. 1955–1970.

4. Almas, M.S., Vanfretti, L., Singh, R.S., Jonsdottir, G.M. Vulnerability of synchrophasor-based WAMPAC applications to time synchronization spoofing // *IEEE Transactions on Smart Grid*. – 2018. – Vol. 9, No. 5. – P. 4601–4612. – DOI: 10.1109/TSG.2017.2665461.

5. Siamak, S., Dehghani, M., Mohammadi, M. Counteracting GPS spoofing attack on PMUs by dynamic state estimation // *2019 Smart Grid Conference (SGC)*. – 2019. – P. 1–5.

6. Khan, S., Mohsin, M., Iqbal, W. On GPS spoofing of aerial platforms: A review of threats, challenges, methodologies, and future research directions // *PeerJ Computer Science*. – 2021. – Vol. 7.

7. Arthur, M.P. Detecting signal spoofing and jamming attacks in UAV networks using a lightweight IDS // *2019 International Conference on Computer, Information and Telecommunication Systems (CITS)*. – 2019. – P. 1–5. – DOI: https://doi.org/10.1109/CITS.2019.8862148.

8. Morales-Ferre, R., Richter, P., Falletti, E., de la Fuente, A., Lohan, E.S. A survey on coping with intentional interference in satellite navigation for manned and

unmanned aircraft // *IEEE Communications Surveys & Tutorials*. – 2020. – Vol. 22, No. 1. – P. 249–291. – DOI: https://doi.org/10.1109/COMST.2019.2949178.

9. Semanjski, S., Semanjski, I., De Wilde, W., Muls, A. Cyber-threats analytics for detection of GNSS spoofing // *Proceedings of DATA ANALYTICS 2018: The Seventh International Conference on Data Analytics*. – 2018. – P. 136–140.

10. Jullian, O., Otero, B., Stojilović, M., Costa, J.J., Verdú, J., Pajuelo, M.A. Виявлення GPS-спуфінгу за допомогою глибокого навчання//*Матеріали конференції* -2021

11. Park, K.H., Park, E., Kim, H.K. Unsupervised intrusion detection system for unmanned aerial vehicle with less labeling effort // *Information Security Applications*. – 2020. – P. 45–58. – DOI: https://doi.org/10.1007/978-3-030-65299-9_4.

12. Литвин В., Пелещак І., Степаняк Я., Пелещак Р. High-Precision Detection of GPS Spoofing Attacks on UAVs Using MLP // Computing. – 2022. – Т. 19, № 3. – С. XX–XX. – DOI: 10.47839/ijc.19.3.XX.

13. Nunez, J., Tran, V., Katangur, A. Protecting the unmanned aerial vehicle from cyberattacks // *The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp)*. – Athens, 2019.

14. Chen, W., Dong, Y., Duan, Z. Accurately redirecting a malicious drone // *IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*. – 2022. – P. 827–834. – DOI: https://doi.org/10.1109/CCNC49033.2022.9700664.

15. Li, J., Li, W., Fu, Q., Liu, B. Research progress of GNSS spoofing and spoofing detection technology // *IEEE 19th International Conference on Communication Technology (ICCT)*. – 2019. – P. 1360–1369. – DOI: https://doi.org/10.1109/ICCT46805.2019.8947107.

16. Zhi, Y., Fu, Z., Sun, X., Yu, J. Security and privacy issues of UAV: A survey // *Mobile Networks and Applications*. – 2020. – P. 95–101. – DOI: https://doi.org/10.1007/s11036-018-1193-x.

17. Humphreys, T.E., Bhatti, J.A., Shepard, D., Wesson, K. The Texas spoofing test battery: Toward a standard for evaluating GPS signal authentication techniques // *Radionavigation Laboratory Conference Proceedings*. – 2012.

18. Joplin, A., Lightsey, E.G., Humphreys, T.E. Development and testing of a miniaturized, dual-frequency GPS receiver for space applications // *Institute of Navigation International Technical Meeting*. – Newport Beach, CA, 2012.

19. GNSS-SDR: My first fix [Електронний ресурс]. – Режим доступу: https://gnss-sdr.org/my-first-fix/ (дата звернення: 14.05.2025).

20. RINEX: The Receiver Independent Exchange Format. Version 3.04 [Електронний ресурс] / W. Gurtner, L. Estey. – International GNSS Service (IGS), 2018. – Режим доступу: https://igs.org/wg/rinex/ (дата звернення: 14.05.2025).

АНОТАЦІЯ

Степаняк Я.А., Пелещак Р.М. (керівник). Інтелектуальна система виявлення спуфінг атак на БПЛА на основі глибокої нейронної мережі. Магістерська кваліфікаційна робота. – Національний університет «Львівська політехніка», Львів, 2025.

Розширена анотація.

Стрімкий розвиток технологій безпілотних літальних апаратів (БПЛА) привів до їхнього широкого застосування в оборонній, комерційній та цивільній сферах. Однак розширення функціональних можливостей дронів супроводжується зростанням рівня загроз їхній безпеці, зокрема навігаційним атакам, серед яких однією з найнебезпечніших є GPS-спуфінг [1]. Такий тип атак базується на підміні автентичних сигналів глобального позиціонування фальшивими, що дозволяє змінити курс БПЛА або навіть захопити його управління без відома оператора [2].

На сьогоднішній день переважна більшість існуючих засобів протидії GPS-спуфінгу орієнтовані на використання додаткового апаратного забезпечення або обмежені ефективністю у випадку синхронних атак [3]. Враховуючи технічні обмеження та потребу в оперативному реагуванні, актуальним стає розроблення програмного засобу, здатного автономно виявляти такі загрози в реальному часі без значних змін до архітектури апарата. Застосування методів глибокого навчання, зокрема багатошарових нейронних мереж, відкриває нові можливості для аналізу навігаційних даних та виявлення аномалій у сигналі GPS.

Метою цієї магістерської роботи є створення інтелектуальної системи виявлення спуфінг-атак на БПЛА на основі багатошарового перцептрону (MLP), яка здатна функціонувати автономно на борту літального апарата, працювати в реальному часі та забезпечувати високу точність виявлення навігаційних аномалій.

Об’єктом дослідження виступає навігаційна система безпілотного літального апарата, схильна до зовнішнього втручання у вигляді GPS-спуфінгу. **Предметом дослідження** є методи та алгоритми виявлення спуфінг-атак за

допомогою глибоких нейронних мереж на основі аналізу параметрів сигналів супутникової навігації.

Для досягнення поставленої мети в роботі вирішено такі завдання:

- проведено огляд сучасних загроз навігаційній безпеці БПЛА;
- здійснено аналіз підходів до побудови систем захисту та обґрунтовано вибір автономної архітектури;
- сформовано набір ознак сигналу GPS (псевдовідстань, доплерівський зсув, відношення сигнал/шум – C/N_0);
- побудовано модель багатошарового перцептрону, навченої на основі датасету TEXBAT [4];
- реалізовано програмний прототип системи, який об’єднує модулі обробки, класифікації та візуалізації;
- проведено експериментальне тестування системи: точність моделі перевищила 93%;
- здійснено економічне обґрунтування доцільності впровадження системи у виробничу практику.

Наукова новизна одержаних результатів полягає в тому, що в роботі вперше розроблено програмну архітектуру системи виявлення GPS-спуфінгу на основі MLP, адаптовану до використання в умовах обмежених обчислювальних ресурсів бортового обладнання. Запропоновано комбінований підхід до формування вхідного вектору ознак, що дозволяє досягти високої точності класифікації без необхідності залучення апаратних засобів. Удосконалено процес попередньої обробки телеметричних даних та реалізовано механізм прийняття рішень щодо виявлення атак в реальному часі. Робота є внеском у практичне використання методів глибокого навчання для забезпечення навігаційної безпеки БПЛА.

Ключові слова: GPS-спуфінг, БПЛА, глибоке навчання, багатошаровий перцептрон, TEXBAT, навігаційна безпека

Перелік використаних літературних джерел:

1. Mozaffari, M., Saad, W., Bennis, M., & Debbah, M. (2016). Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs. IEEE Transactions on Wireless Communications, 15(6), 3949–3963. https://doi.org/10.1109/TWC.2016.2531652

2. Shafiee, E., Mosavi, M. R., & Moazedi, M. (2018). Detection of spoofing attack using machine learning based on multi-layer neural network in single-frequency GPS receivers. Journal of Navigation, 71(1), 169–188. https://doi.org/10.1017/S0373463317000558

3. Almas, M. S., Vanfretti, L., Singh, R. S., & Jonsdottir, G. M. (2018). Vulnerability of synchrophasor-based WAMPAC applications to time synchronization spoofing. IEEE Transactions on Smart Grid, 9(5), 4601–4612. https://doi.org/10.1109/TSG.2017.2665461

4. Humphreys, T. E., Bhatti, J. A., Shepard, D., & Wesson, K. (2012). The Texas spoofing test battery: Toward a standard for evaluating GPS signal authentication techniques. Radionavigation Laboratory Conference Proceedings.

ABSTRACT

Stepaniak Ya.A., Peleshchak R.R. (supervisor). Intelligent System for Detecting Spoofing Attacks on UAVs Based on Deep Neural Networks. Master's Thesis – Lviv Polytechnic National University, Lviv, 2025.

Extended abstract.

The rapid development of unmanned aerial vehicles (UAVs) has led to their widespread adoption in defense, commercial, and civilian sectors. However, the increasing functionality of UAVs is accompanied by heightened threats to their operational safety, especially in terms of navigation [1]. One of the most dangerous and difficult-to-detect threats is GPS spoofing, a type of cyberattack in which counterfeit positioning signals are broadcast to mislead the UAV's navigation system [2].

Most of today's GPS spoofing countermeasures rely on additional hardware components or are ineffective against synchronized spoofing attacks [3]. Given hardware limitations and the need for timely response, there is a clear need for a software-based solution that can autonomously detect spoofing in real time without modifying the UAV's physical structure. The use of deep learning methods, particularly multilayer neural networks, offers promising capabilities for analyzing GPS telemetry and detecting anomalies.

The aim of this master's thesis is to develop an intelligent system for detecting GPS spoofing attacks on UAVs using a multilayer perceptron (MLP) capable of autonomous, real-time operation with high classification accuracy.

The object of research is the UAV's navigation system, which is potentially vulnerable to spoofing.

The subject of research is the methods and algorithms for detecting spoofing attacks using deep neural networks based on GPS signal parameters.

To achieve this aim, the following tasks were completed:

- analysis of modern threats to UAV navigation systems and existing spoofing detection methods;

- substantiation of the architecture of the detection system and justification for an onboard autonomous design;
- construction of a feature set (pseudorange, Doppler shift, C/N_0) for telemetry vector representation;
- development and training of a multilayer perceptron model using real-world data from the TEXBAT dataset [4];
- implementation of a software prototype for real-time spoofing detection with GUI support;
- evaluation of detection performance (accuracy >93%) under real spoofing scenarios;
- assessment of the system's economic feasibility for practical deployment.

Scientific novelty of this work lies in the development of a software-based architecture for spoofing detection using an MLP model tailored to run on low-power onboard UAV equipment. For the first time, a combined feature extraction approach was proposed, enabling accurate classification without additional hardware. The thesis enhances real-time data preprocessing methods and demonstrates the practical applicability of deep learning in UAV navigation protection.

Keywords: GPS spoofing, UAV, deep learning, MLP, TEXBAT, navigation security.

References:

1. Mozaffari, M., Saad, W., Bennis, M., & Debbah, M. (2016). Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs. IEEE Transactions on Wireless Communications, 15(6), 3949–3963. https://doi.org/10.1109/TWC.2016.2531652

2. Shafiee, E., Mosavi, M. R., & Moazedi, M. (2018). Detection of spoofing attack using machine learning based on multi-layer neural network in single-frequency GPS receivers. Journal of Navigation, 71(1), 169–188. https://doi.org/10.1017/S0373463317000558

3. Almas, M. S., Vanfretti, L., Singh, R. S., & Jonsdottir, G. M. (2018). Vulnerability of synchrophasor-based WAMPAC applications to time synchronization spoofing. IEEE Transactions on Smart Grid, 9(5), 4601–4612. https://doi.org/10.1109/TSG.2017.2665461

4. Humphreys, T. E., Bhatti, J. A., Shepard, D., & Wesson, K. (2012). The Texas spoofing test battery: Toward a standard for evaluating GPS signal authentication techniques. Radionavigation Laboratory Conference Proceedings.

ДОДАТКИ

Додаток А

Код програми:

```
import tkinter as tk
from tkinter import filedialog, messagebox
import pandas as pd
import numpy as np
from tensorflow.keras.models import load_model
import matplotlib.pyplot as plt
import joblib
import winsound
from datetime import datetime

# Ініціалізація глобальних змінних
scaler = joblib.load('scaler.pkl')
model = load_model('model.h5')
data_df = None
aggregated = None
classes = None

# Завантаження CSV
def load_csv():
    global data_df
    file_path = filedialog.askopenfilename(filetypes=[("CSV files", "*.csv")])
    if file_path:
        data_df = pd.read_csv(file_path)
        text.delete('1.0', tk.END)
        text.insert(tk.END, f"✅ Завантажено файл: {file_path.split('/')[-1]}\n\n")
        text.insert(tk.END, data_df.head().to_string(index=False))

# Завантаження нової моделі
def load_new_model():
    global model
    file_path = filedialog.askopenfilename(filetypes=[("Model files", "*.h5")])
    if file_path:
        model = load_model(file_path)
        text.insert(tk.END, f"\n✅ Модель оновлено: {file_path}\n")

# Завантаження нового scaler
def load_new_scaler():
```

```
    global scaler
    file_path = filedialog.askopenfilename(filetypes=[("Scaler files", "*.pkl")])
    if file_path:
        scaler = joblib.load(file_path)
        text.insert(tk.END, f"\n✅ Масштабувальник оновлено: {file_path}\n")

# Інференс
def run_inference():
    global data_df, aggregated, classes
    if data_df is None:
        text.insert(tk.END, "\n❌ Спочатку завантажте CSV-файл.\n")
        return

    try:
        aggregated = data_df.groupby('time').agg({
            'pseudorange': ['mean', 'min', 'max', 'std'],
            'signal_strength': ['mean', 'min', 'max', 'std'],
            'doppler_shift': ['mean', 'min', 'max', 'std']
        }).reset_index()

        aggregated.columns = ['_'.join(col).rstrip('_') for col in aggregated.columns]

        X = aggregated.drop(columns=['time'])
        X_scaled = scaler.transform(X)
        preds = model.predict(X_scaled).ravel()
        classes = ["атака" if p >= 0.5 else "норма" for p in preds]

        text.insert(tk.END, "\n\n📊 Прогнози моделі:\n")
        for i, (timestamp, cls) in enumerate(zip(aggregated['time'], classes), start=1):
            text.insert(tk.END, f"{i}. {timestamp}: {cls}\n")
            with open("log.txt", "a", encoding="utf-8") as f:
                f.write(f"{datetime.now()} — {timestamp}: {cls}\n")

        attack_count = classes.count("атака")
        normal_count = classes.count("норма")
        attack_percent = (attack_count / len(classes)) * 100

        text.insert(tk.END, f"\n🧮 Атак: {attack_count} ({attack_percent:.2f}%)\n")

        if attack_count > 5:
            text.insert(tk.END, "\n🚨 GPS-спуфінг виявлено!\n")
            winsound.Beep(1000, 700)
```

```
            messagebox.showwarning("Попередження", "GPS-спуфінг виявлено!")
        else:
            text.insert(tk.END, "\n✅ Атак недостатньо для тривоги.\n")

        plt.figure()
        plt.bar(['Норма', 'Атака'], [normal_count, attack_count], color=['green', 'red'])
        plt.title("Розподіл класів")
        plt.tight_layout()
        plt.show()

    except Exception as e:
        text.insert(tk.END, f"\n❌ Помилка: {e}\n")

# Експорт
def export_predictions():
    global aggregated, classes
    if aggregated is None or classes is None:
        text.insert(tk.END, "\n❌ Немає даних для експорту.\n")
        return
    result_df = pd.DataFrame({'time': aggregated['time'], 'prediction': classes})
    result_df.to_csv("inference_results.csv", index=False)
    text.insert(tk.END, "\n💾 Збережено у inference_results.csv\n")

# === Інтерфейс ===
root = tk.Tk()
root.title("Демо виявлення GPS-спуфінгу")
root.configure(bg='#f4f4f4')

label = tk.Label(root, text="🔍 Завантажте CSV з GPS-даними та виконайте аналіз", fg="black", bg='#f4f4f4')
label.pack(pady=5)

text = tk.Text(root, height=20, width=100, bg='white')
text.pack(padx=5, pady=5)

frame = tk.Frame(root, bg='#f4f4f4')
frame.pack(pady=10)

tk.Button(frame, text="📂 Завантажити CSV", command=load_csv).grid(row=0, column=0, padx=5)
tk.Button(frame, text="⚙️ Інференс", command=run_inference).grid(row=0, column=1, padx=5)
```

```
tk.Button(frame, text="💾 Експорт", command=export_predictions).grid(row=0, column=2, padx=5)
tk.Button(frame, text="🔄 Нова модель", command=load_new_model).grid(row=0, column=3, padx=5)
tk.Button(frame, text="🔧 Новий scaler", command=load_new_scaler).grid(row=0, column=4, padx=5)

root.mainloop()
```