ЗМІСТ

[ВСТУП 3](#_Toc100047879)

[1 АНАЛІТИЧНИЙ ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ 6](#_Toc100047880)

[1.1 Машинне навчання 6](#_Toc100047881)

[1.2 Нейронні мережі 10](#_Toc100047882)

[1.3 Еволюційні нейронні мережі 20](#_Toc100047883)

[1.4 Генетичні алгоритми для навчання нейронних мереж 23](#_Toc100047884)

[1.5 Прикладна задача про визначення можливості руйнування структурних елементів 25](#_Toc100047885)

[1.6 Застосування нейронних мереж для розв’язання задачі про визначення можливості руйнування структурних елементів 27](#_Toc100047886)

[ВИСНОВКИ 28](#_Toc100047887)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 29](#_Toc100047888)

# ВСТУП

Для багатьох елементів конструкції руйнування виникає через втому, поступове накопичення пошкоджень і розпочинається з невеликої тріщини, яка розростається під навантаженням. Втомна тріщина утворюється переважно в концентраторі напружень, тобто в місці пошкодження, яке послаблює область поперечного перерізу матеріалу. Тріщина росте доти, доки матеріал здатний витримувати навантаження. Тому основними факторами, що впливають на міцність елементів конструкцій, є дефекти поверхні деталей, температура та зовнішнє середовище під час експлуатації, характер та умови навантаження.

Найбільші дослідження щодо руйнування структурних елементів конструкцій було проведено в авіаційній галузі, де широко застосовуються сплави алюмінію. Тому найбільш повні експериментальні дані щодо утворення тріщин саме щодо цих матеріалів.

Тому важливо вивчати вплив умов навантаження на ріст втомних тріщин. Найчастіше отримані експериментальні дані містять певний розкид, який слід враховувати при їх аналізі.

Методи оцінки міцності та довговічності опорних елементів конструкції часто потребують складних розрахунків. Отже, варто навчитися розв’язувати задачі механіки руйнування методами машинного навчання, зокрема, нейронними мережами, форсованими деревами, випадковими лісами, машинами опорних векторів та методом *k*-найближчих сусідів, які дозволяють досягти високої точності розв’язків.

Це дозволяє застосовувати зазначені вище методи в різних галузях науки і промисловості, де прогноз довговічності елементів конструкції має велике значення: металургії, авіабудуванні, залізничному та трубопровідному транспорті, ядерних технологіях тощо. Остаточне рішення прийматимуть експерти, але доступність інформаційних даних підвищить професійний рівень таких рішень.

Метою дослідження є оцінка міцності та залишкової довговічності елементів конструкції методами машинного навчання. Для досягнення цієї мети необхідно було виконати наступні завдання:

• провести аналітичний огляд методів машинного навчання, а саме навчання з учителем, навчання без учителя, навчання з підкріпленням, нейронних мереж, посилених дерев, випадкових лісів, машин опорних векторів,   
k-найближчих сусідів;

• визначити оптимальну архітектуру еволюційних нейронних мереж;

• спрогнозувати швидкість наростання втомних тріщин (FCG) опорних елементів конструкції з конкретного матеріалу.

*Об'єктом дослідження* є методи машинного навчання, зокрема, нейронні мережі, форсовані дерева, випадкові ліси, машини опорних векторів, метод *k*-найближчих сусідів як інструменти для прогнозування швидкості наростання втомних тріщин.

*Предметами дослідження* є опорні структурні елементи.

*Актуальність дослідження*: визначення тривалості дії структурних елементів дозволить вчасно визначити необхідність заміни структурного елементу для запобігання його виходу з ладу протягом робочого процесу.

*Результати дослідження можуть бути застосовані*: Методи оцінки міцності та залишкової довговічності відповідальних елементів конструкції часто вимагають складних і дорогих експериментальних методів для отримання діаграм швидкості наростання втомних тріщин. Тому, маючи експериментальні дані, можна розв’язувати задачі механіки руйнування методами машинного навчання, зокрема, нейронними мережами, форсованими деревами, випадковими лісами, машинами опорних векторів та методом *k*-найближчих сусідів, що дозволяє досягти високої точності. Запропонований підхід до прогнозування швидкості наростання втомних тріщин при регулярному навантаженні методами машинного навчання дозволяє визначити залишкову довговічність на етапі проєктування. Це дозволяє застосовувати зазначені вище методи в різних галузях науки та промисловості, де прогнозування довговічності опорних елементів конструкції має велике значення.

# АНАЛІТИЧНИЙ ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ

## Машинне навчання

Машинне навчання – галузь штучного інтелекту, мета якої – розробка методів, що втілюватимуть різноманітні форми навчання, особливо механізми, що здатні утворювати знання, базуючись на прикладах або даних.

Основними методами машинного навчання [1] є кероване навчання (або навчання з учителем). Таким методами надається певна кількість пар *(x, y)*. Задача методів – знайти правильне передбачення *y\**, маючи на вході *x\**. Вхідні значення *x* можуть бути векторами чисел, або ж більшими об’єктами, наприклад документами, зображеннями, ланцюгами ДНК чи графами. Методи керованого навчання використовується для визначення спаму в електронній пошті, розпізнавання облич на зображеннях або в системах визначення медичних діагнозів для пацієнтів.

Інший важливий метод машинного навчання – навчання без учителя. Воно полягає в аналізі даних без чітких позначень, припускаючи, що відомі певні структурні особливості досліджуваних даних. Одним із прикладів є кластеризація, що полягає в поділі інформації на певну кількість груп (кластерів) і знаходженні правила, що дозволить віднести дані, що будуть отримані в майбутньому, до одного з уже утворених кластерів.

Ще один метод – навчання з підкріпленням – полягає в тому, що замість надання чітких прикладів результатів, залежно від заданих вхідних даних, передбачається, що вхідні дані надають лише вказівку, чи є певна дія правильною або ні. У випадку неправильності дії питання про те, як знайти правильну, залишається дійсним. Зазвичай за наявної послідовності входів сигнали про результат дій надаються по завершенню всієї послідовності дій, а не кожної окремої дії [2].

До моделей, що втілюють методи машинного навчання відносяться дерева ухвалення рішень, метод опорних векторів, метод *k*-найближчих сусідів, нейронні мережі тощо.

Дерева ухвалення рішень будуються за допомогою рекурсивної розбивки вибірки даних, яка потім представляється у вигляді логічної структури дерева. Цей процес називається фазою росту. Унаслідок її кожен лист дерева асоціюється з одним атрибутом даних. На фазі обрізки дерева воно узагальнюється шляхом створення піддерева. Цей процес унеможливлює перенавчання. Для обрізки дерева частим орієнтиром є точність класифікації [3].

Класифікація за допомогою дерев ухвалення рішень відбувається сортуванням від кореня дерева до певного листка, що надає класифікацію певної ознаки вхідних даних. Кожен листовий вузол дерева перевіряє значення цієї ознаки, а кожна гілка, що виходить із листового вузла є одним із можливих її значень. Вхідні дані класифікують рухом із кореня дерева, перевіркою значення відповідної ознаки, рухом відповідною гілкою до наступної ознаки та повторенням цього процесу для нового піддерева, утвореного після класифікації за попередньою ознакою. Дерева ухвалення рішень мають широку сферу застосування у практичних задачах. Їх було застосовано для діагностики захворювань, визначення кредитного ризику для заявників на отримання кредиту та багатьох інших задач із реального життя [4].

До методів, що базуються на використанні дерев ухвалення рішень, відносяться підсилені дерева, випадкові ліси та обертові ліси.

Підсилені дерева – дерева ухвалення рішень, для яких застосовано алгоритм градієнтного підсилення. Алгоритм градієнтного підсилення (або алгоритм підсилення) в загальному випадку може використовуватися не лише для дерев ухвалення рішень, а до будь-якого класифікатора, у тому числі нейронних мереж. Через складність утворення достатньо гарного дискримінанту для певної кількості дерев ухвалення рішень, доцільним є утворення декількох простіших, що більш схильні до помилки. Такі дискримінанти називаються слабкими класифікаторами. Підсилення полягає в об’єднанні слабких класифікаторів в один, що буде більш стабільним, точнішим та менш схильним до помилки [5].

Підсилення дерев розпочинається з класифікації певного входу за допомогою звичайного дерева ухвалення рішень. У випадку помилки класифікації входу надається певна вага, шо збільшуються після кожної помилкової класифікації. Зазвичай для класифікації цим методом будуються від 1000 до 2000 дерев.

Після проходження входу кожним із цим дерев, йому надається певне значення оцінки. Якщо класифікація відбувається між двома групами, потрапляння входу до однієї надає події значення 1, до іншої – -1. Нормалізована сума цих оцінок стає фінальним значенням оцінки цього входу. Часто оцінки, що підсумовуються, використовуються у зваженому вигляді. Вища фінальна оцінка позначає належність до групи 1, нижча – до групи 2. Зазвичай вибирається певне значення, що визначає поділ між групами. Події з вищими за це значення оцінками належать до групи 1, з нижчими – до групи 2 [6].

Серед методів підсилення дерев як основні виділяють AdaBoost та стиснення (або *є*-підсилення). Для AdaBoost (Adaptive Boosting, адаптивне підсилення) значення помилки для дерева m визначається як відношення між вагою помилково класифікованих входів та загальною вагою цього дерева. Для обчислення ваг задається параметр , де попередньо задана стала – значення похибки для дерева m. Вага кожного помилково класифікованого входу множиться на . Ваги нормалізуються таким чином, що сума всіх ваг входів дорівнює 1. Фінальна оцінка .

Для методу стиснення вага кожного помилково класифікованого входу множиться на , де *є* – попередньо задана стала. Фінальна оцінка [7].

Ідея випадкових лісів полягає в побудові певної кількості дерев ухвалення рішень та об’єднання їх у комітет, після чого кожне дерево класифікує об’єкт за певними ознаками. Після цього для об’єкту вибирається клас, за який проголосувала найбільша кількість дерев із комітету [8].

Обертові ліси базуються на чутливості дерев ухвалення рішень до обертання за осями. Тобто, факт того, що класифікація може відрізнятися, залежно від напряму обертання, може бути перевагою в тому випадку, коли дерева використовуються як члени комітету (чи ансамблю), хоча зазвичай це вважається недоліком дерев ухвалення рішень. Дерева прийняття рішень, отримані з оберненої множини вхідних даних, можуть бути точними, при цьому будучи достатньо відмінними [9].

Метод опорних векторів утілює метод машинного навчання з учителем. Основною сферою його застосування є бінарна класифікація. У цьому методі відбувається поділ простору на певну кількість підпросторів, що відповідають класам. Точки, побудовані на поверхні, називаються опорними векторами. Основна робота відбувається саме з цими векторами та з простором, що розташовується в них.

Завдання методу опорних векторів полягає у знаходженні лінійної моделі вигляду , де *x* – вхідний вектор, *w* та *b* – параметри, що добираються для певних моделей. Знаходження цих параметрів відповідає розв’язанню задачі опуклої оптимізації. Метод опорних векторів може замінювати нейронні мережі, однак його процес навчання дуже повільний.

Метод *k*-найближчих сусідів [10] є одним із найпростіших алгоритмів класифікації, що часто застосовується також і для задач регресії. Класифікація за цим алгоритмом базується на обчисленні відстані до кожного об’єкта навчальної вибірки та вибору k об’єктів навчальної вибірки, відстань до яких мінімальна.

## Нейронні мережі

Нейронні мережі початково розроблялися як математичні моделі можливостей мозкової обробки інформації. Базова структура нейронної мережі включає невеликі процесори чи вузли, поєднані зв’язками з певною вагою. Вузли відповідають нейронам мозку, тоді як зв’язки відображають силу синапсів між нейронами. Мережа активується за допомогою надання вхідних даних усім або певним вузлам та подальшому поширенні активації через зважені зв’язки між цими вузлами [11].

Поширення активації відбувається за допомогою спрощеної моделі нейрона, розробленої Маккалохом і Піттсом, званої також штучним нейроном, математичним нейроном Маккалоха–Піттса чи формальним нейроном.

За цією моделлю до кожного вузла за допомогою зважених зв’язків надходять вхідні значення від інших вузлів. Процесор утворює лінійну комбінацію вхідних значень із ваговими коефіцієнтами відповідних зважених зв’язків і застосовує до них функцію активації. Значення, отримане за допомогою функції активації, передається як вхідне для наступних вузлів.

Тобто, кожен вузол приймає сигнали від інших вузлів. Вони накопичуються, передаються функції активації, після чого відбувається розрядка у вигляді передачу сигналу іншим вузлам із певною вагою зв’язку. Після цього сигнали можуть розпочати знову накопичуватися у вузлі.

Топологія – спосіб сполучення нейронів, що формують розглянуту нейронну мережу. Топологію можна розглядати як відношення між нейронами мережі, утілене за допомогою зв’язків між ними. Вона має досить важливу роль для функціональності та продуктивності нейронної мережі. Незважаючи на досить поширене використання терміну топологія як синоніму до термінів структура чи архітектура, їхні значення не є чітко визначеними. Тому в деяких роботах терміни структура та архітектура нейронної мережі стосуються не лише топології нейронної мережі, або означають щось цілковито відмінне.

У загальному випадку топологія складається з двох частин: каркасу нейронної мережі та структури зв’язків. Каркас нейронної мережі, у свою чергу, поділяється на кількість шарів у заданій мережі та кількість нейронів на кожному шарі. Структура зв’язків визначає спосіб, за допомогою якого нейрони сполучено між собою. Виділяються чотири види зв’язків між нейронами:

* Міжшаровий (interlayer) – вид зв’язків між нейронами сусідніх шарів;
* Внутрішньошаровий (intralayer) – вид зв’язків між нейронами того самого шару;
* Зв’язок із самим собою (selfconnection) – особливий вид внутрішньошарового зв’язку, зв’язок нейрона зі самим собою;
* Надшаровий (supralayer) – зв’язок між нейронами, що знаходяться на несусідніх і відмінних шарах, тобто зв’язки між ними «перестрибують» один або декілька проміжних шарів.

Кожна нейронна мережа, що поділена на шари, має принаймні один міжшаровий зв’язок. У випадку відсутності достатньої кількості таких зв’язків, над нейронною мережею можливо провести просторове перетворення, що дозволить перетворити інші види зв’язків у міжшарові для новоутвореної мережі, поділеної на шари [12].

Мережі прямого зв’язку – нейронна мережа, у якій рух інформації (у вигляді активації нейронів) відбувається лише в напрямку від вхідного шару до вихідного. У таких мережах наявний лише один тип зв’язків: міжшарові. У випадкові наявності надшарових зв’язків мережа перетворюється на так звану залишкову нейронну мережу, у випадку наявності зв’язків із самим собою чи міжшарових зв’язків, що не ведуть у напрямку від вхідного шару до вихідного – на рекурентну.

За архітектурою нейронні мережі прямого зв’язку поділяються на одношарові та багатошарові. Одношарові містять лише два шари: вхідний і вихідний. Нейрони вхідного шару отримують вхідні сигнали, а вихідного – повертають вихідні сигнали мережі. Синаптичні зв’язки поєднують кожен нейрон вхідного шару з нейроном вихідного, але не у зворотному порядку. Незважаючи на наявність двох шарів (вхідного та вихідного), лише один із них відповідає за обчислення, тому такі нейронні мережі й мають назву одношарових.

Архітектура класу багатошарових мереж містить не лише вхідний і вихідний шари, а також так звані приховані шари, що знаходяться між вхідним і вихідним шаром і здійснюють необхідні проміжні обчислення перед тим як надати вхід на нейрони вихідного шару. Кількість прихованих шарів у багатошарової нейронної мережі не обмежено. Нейрони, що знаходяться на цих шарах, мають назву прихованих вузлів або прихованих нейронів. Ваги зв’язків між вхідним і прихованим шаром нейронів мають назву вхідних-прихованих зв’язків, а між прихованим і вихідним – приховано-вихідних зв’язків. Багатошарові нейронні мережі, що містять велику кількість прихованих шарів часто називають також глибокими нейронними мережами [13].



Рисунок 1.1 – Одношарова нейронна мережа



Рисунок 1.2 – Багатошарова нейронна мережа

­

Рисунок 1.3 – Залишкова нейронна мережа



Рисунок 1.4 – Рекурентна нейронна мережа

Для навчання нейронних мереж зазвичай застосовується алгоритм зворотного поширення помилки. Навчання здійснюється за допомогою ітеративного оновлення ваг зв’язків між вузлами нейронної мережі. Математичний зміст алгоритму полягає в застосуванні негативного градієнту середньоквадратичної функції похибки.

Алгоритм розпочинається з обчислення значення помилки як різниці між очікуваним виходом нейронної мережі та отриманим. Отримане значення поширюється на попередні рівні нейронної мережі, викликаючи коригування ваг зв’язків між нейронами. Алгоритм зворотного поширення помилки застосовується для багатьох задач, однак його швидкість збіжності порівняно невисока, тому розробляються різноманітні модифікації чи інші алгоритми навчання нейронних мереж [14].

Функції активації нейронних мереж поділяються на три основні категорії, що містять певні переваги й недоліки:

1. Бінарна крокова функція – активує нейрон лише в тому випадку, якщо вхідне значення більше чи менше за певний наперед заданий рівень. У такому випадку вузол надсилає те саме значення, що й прийняв. Недолік крокової функції в тому, що вона не підтримує вихід із декількох значень, тобто не підтримує, наприклад, класифікацію в декілька різних категорій.
2. Лінійна функція – вихідний сигнал цієї функції пропорційний до вхідного. На відміну від крокової, вона може надавати значення, відмінні від 0 та 1 (чи -1 та 1, залежно від типу функції). Її недоліки полягають у неможливості застосування зворотного поширення помилки, оскільки похідна функції – стала, у тому, що вихідний шар завжди є лінійною функцією та в неможливості застосування функції для обробки складних даних, оскільки вона є лише лінійною моделлю регресії.
3. Нелінійні функції – дозволяють утворювати складні відображення між входом і виходом нейронної мережі. Їх можна застосовувати також і для складних даних, як-от відео, аудіо тощо. Перевагами є можливість застосування зворотного поширення помилки, оскільки похідні цих функцій – функції, що залежать від вхідних параметрів та використання для навчання зі складним наборів даних із високим рівнем точності [15].

Одним із слабких місць у нейронних є мережах є так звана проблема зникомого градієнту. Вона найчастіше виникає в мережах, у яких наявна велика кількість прихованих шарів. Із додаванням більшої кількості прихованих шарів навчання мережі стає все важчим і може призвести до того, що нейронна мережа не зможе завершити своє навчання. Цей випадок і називається проблемою зникомого градієнту та є звичним феноменом для глибоких нейронних мереж. Особливо схильними до цієї проблеми є рекурентні нейронні мережі.

Математичний зміст проблеми зникомого градієнту полягає в тому, що значення градієнтів нейронної мережі поступово зменшується, що призводить до менших змін у значеннях зв’язків між нейронами, що у свою чергу ускладнює тренування нейронної мережі, зменшуючи швидкість та збільшуючи час її тренування, а часто й зменшуючи точність класифікації. Існує декілька різних способів вирішення цієї проблеми, що дозволяють стабілізувати градієнти та запобігти появі проблеми зникомого градієнту. До них відноситься так звана пакетна нормалізація чи комбінація двох алгоритмів навчання та функцій активації [16].

До основних функцій активації відносяться:

1. Лінійна функція (linear function). Функція задається формулою *a\*x*, де *x* – вхідне значення нейрона. Графік лінійної функції активації має такий вигляд:

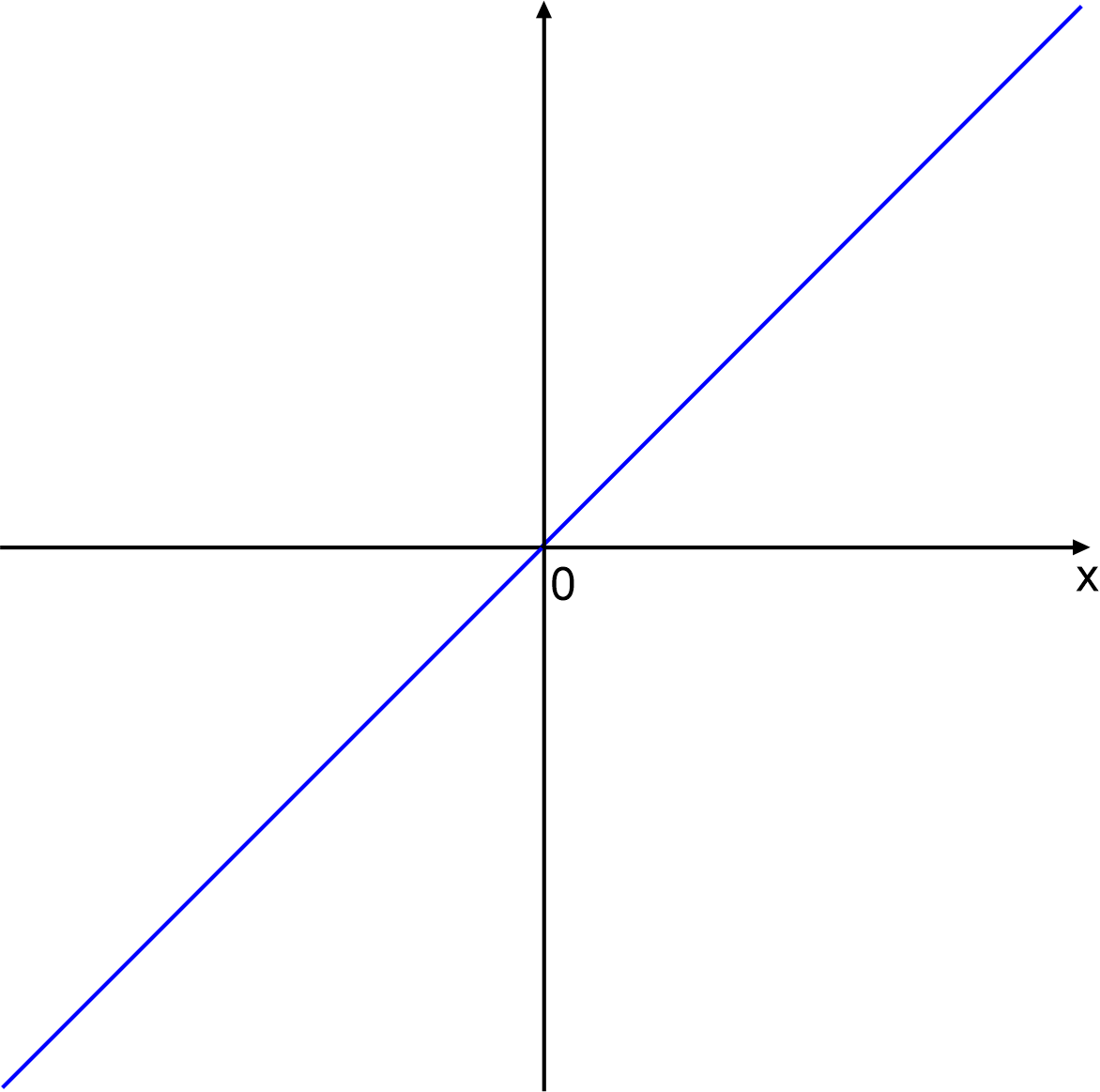


Рисунок 1.5 – Графік лінійної функції активації

1. Зрізаний, або випрямлений лінійний вузол (rectified linear unit, ReLU). Функція задається формулою: *max(0,x)*, де *x* – вхідне значення нейрона. Тобто, ReLU приймає значення 0, якщо на вхід нейрона подається значення  
    x 0 та значення *x*, якщо на вхід нейрона подається значення *x > 0*. Переваги цієї функції полягають у вирішенні проблеми зникомого градієнту та в достатньо меншій складності обчислення, порівняно з логістичною функцією активації чи гіперболічним тангенсом. До недоліків відноситься те, що ReLU доцільно застосовувати лише на прихованих шарах мережі, те, що нейрони зі вхідними значеннями ,меншими чи рівними 0 не будуть змінювати своєї ваги через те, що градієнт ReLU для них буде дорівнювати 0 й те, що значення функції не обмежено вгорі, що може викликати вибух активації за достатньо великих значень x. Графік зрізаного лінійного вузла має такий вигляд:



Рисунок 1.6 – Графік функції активації ReLU

1. Експоненційний лінійний вузол (exponential linear unit, ELU). Функція задається формулою: *x*, якщо та , якщо *x < 0*, де *x* – вхідне значення нейрона. До переваг цієї функції відноситься гладкість функції, що викликає поступове збільшення значень функції, порівняно з різким збільшенням зрізаного лінійного вузла та можливість повертати від’ємні значення. Недоліком є необмеженість угорі, притаманна ELU так само, як і ReLU, що може викликати вибух активації. Графік експоненційного лінійного вузла має такий вигляд:



Рисунок 1.7 – Графік функції активації ELU

1. Логістична функція, або сигмоїда (sigmoid). Функція задається формулою , де *x* – вхідне значення нейрона. До переваг цієї функції належать нелінійність, гладкість градієнту, обмеженість значень як угорі, так і внизу, та здатність до класифікації. Серед недоліків виділяються слабка реакція зміни значень функції до зміни значення *x* на обох кінцях, те, що центр значень функції не знаходиться в точці 0, що ускладнює оптимізацію через розкид значень градієнту та схильність до проблеми зникомого градієнту. Графік логістичної функції має вигляд:



Рисунок 1.8 – Графік логістичної функції активації

1. Гіперболічний тангенс (Tanh). Функція задається формулою , де *x* – вхідне значення нейрона. Перевагою є гладкість похідної, порівняно з логістичною функцією та обмеженість значень унизу та вгорі. Недолік – схильність до проблеми зникомого градієнту. Графік гіперболічного тангенсу має вигляд:



Рисунок 1.9 – Графік функції активації гіперболічний тангенс

## Еволюційні нейронні мережі

Еволюційні алгоритми – стохастичні методи оптимізації, що діють за принципами еволюції у природі: випадково задана популяція особин, що відображає певні точки на просторі пошуку, еволюціонує, слідуючи алгоритму, подібному до дарвінівського виживання найбільш пристосованих. Нові особини утворюються за допомогою генетичних операцій мутації та схрещування. Імовірність виживання певної особини залежить від значення її пристосованості. З високою ймовірністю виживають найбільш пристосовані, найменш пристосовані же швидко зникають. Основні алгоритми, що спираються на цю схему: генетичні алгоритми, еволюційні стратегії та еволюційне   
програмування [17].

Як ще один алгоритм виділяють генетичне програмування. Згадані вище алгоритми вважаються класичними серед еволюційних алгоритмів. До більш нових алгоритмів відносять досить відомі, як-от табу-пошук, алгоритм світлячків та алгоритм мурашиної колонії, так і менш відомі: стрибків жаби, рибної зграї, бджолиної колонії тощо [18].

Ті нейронні мережі, для побудови та навчання яких застосовуються еволюційні алгоритми, називаються еволюційними нейронними мережами. Застосування еволюційних алгоритмів дозволяє досягати кращих результатів у навчанні та узагальненні.

Здійснення еволюції може здійснюватися на трьох різних рівнях, або на їх комбінаціях. Перший рівень – коригування ваг нейронних зв’язків, тобто заміна алгоритму зворотного поширення помилки на один із еволюційних алгоритмів. Другий рівень – еволюція архітектури нейронної мережі. Третій рівень – еволюція правила, за яким навчається нейронна мережа [19].

У цій роботі розглянуто перший рівень еволюції, де нейронна мережа тренується за допомогою кодування ваг зв’язків у хромосому та застосування еволюційного алгоритму для визначення оптимальних значень ваг зв’язків між нейронами. Цей рівень еволюції також можливо покращити, застосовуючи так званий гібридний алгоритм, де еволюція застосовується для знаходження для звуження простору пошуку оптимального розв’язку, після чого оптимальні значення ваг нейронів знаходяться за допомогою алгоритму зворотного поширення помилки.

Другий рівень еволюції полягає в тому, що еволюціонує сама архітектура нейронної мережі. У цьому випадку може змінюватися кількість шарів нейронної мережі, або ж кількість вузлів на кожному шарі. Найбільш ефективно така еволюція працює для невеликих нейронних мереж через порівняну простоту кодування у хромосоми. Для більших нейронних мереж довжина хромосом накладає обмеження, тому часто застосовується інший підхід: пошук найкращого набору правил побудови архітектури нейронної мережі. Саме через те, що розмір правил не збільшується під час збільшення нейронної мережі, цей підхід широко використовується в еволюції архітектури мереж великих розмірів.

На третьому рівні еволюції нейронна мережа еволюціонує за допомогою зміни правила навчання, тобто зміни відбуваються в самому процесі, за яким змінюються ваги зв’язків у нейронній мережі. Зміни можуть полягати в оновленні параметрів алгоритму зворотного поширення помилки, або ж в оптимізації самого алгоритму навчання нейронної мережі.

Наведемо приклад кодування ваг зв’язків нейронної мережі, наведеної на рисунку 1.5 у хромосому, для якої можливо застосувати еволюційний алгоритм. Оскільки значення ваг містять у собі десяткові числа, кодування нейронної мережі за допомогою двійкових чисел є невиправданим. Тому буде використано кодування з використанням дійсних чисел. Відповідна цій мережі хромосома матиме вигляд: *{0.21, 0.4, 0.3, 0.18, 0.75, 0.15, 0.27, 0.55, 0.33, 0.64}.*



Рисунок 1.10 – Приклад нейронної мережі з вагами зв’язків

## Генетичні алгоритми для навчання нейронних мереж

Для навчання еволюційної нейронної мережі було розглянуто та застосовано один із видів еволюційних алгоритмів, що називається генетичний алгоритм.

Генетичний алгоритм – симуляція дарвінівської еволюції. Для нього зберігається популяція хромосом, де кожна хромосома відповідає потенційному розв’язку розглянутої задачі. Для генетичних алгоритмів хромосоми часто називаються ланцюжками. Кожен ланцюжок у свою чергу складається з певної кількості генів, що можуть приймати певну кількість значень, званих алелями. У термінології генетичних алгоритмах гени мають назву ознаки, алелі – значення. Для кожного ланцюжка додається значення придатності, що визначає наскільки пристосованою є особина, кодована цією хромосомою. Придатність визначає відповідна функція придатності. Її можна розглядати як певну міру, яку ми намагаємося максимізувати.

Існує три оператори, що приводять генетичний алгоритм до гарних результатів. Відтворення – процес, за допомогою якого ланцюжки переходять у наступне покоління без змін. Ті особини, що мають найбільше значення придатності мають найбільшу можливість перейти в наступне покоління. Для цього найчастіше застосовується так званий вибір за допомогою рулетки. У ньому колесо рулетки ділиться на певну кількість частин, кожна з яких відповідає … Розмір частин визначається відповідним значенням придатності. Тому під час обертання колеса найчастіше вибираються найкращі особини. Схрещування або кросовер – процес об’єднання частини одного ланцюжка з частиною іншого. Таким чином ми прагнемо об’єднати гарні частини одного ланцюжка з гарними частинами іншої. Операція схрещування приймає дві особини та утворює дві нові, звані нащадками. Існує багато різних видів схрещування. У деяких із них особини обмінюються між собою певною послідовністю генів, в інших кожен алель добирається окремо. Мутація – процес, за якого випадково обраний ген (або певна їх кількість) у ланцюжку отримує інше значення, відмінне від початкового. Мета цієї операції – ввести новий генетичний матеріал у популяцію чи запобігти його втраті. За допомогою мутації ген може отримати значення, яке не існувало до того в популяції чи те, що було втрачено в результаті відтворення. [20]

Основний принцип поєднання нейронної мережі та генетичних алгоритмів полягає в кодуванні інформації про нейронну мережу в геном. Для цього спочатку утворюється певна кількість особин. Продуктивність кожної з утворених таким чином варіантів нейронної мережі визначається після навчання зі зворотним поширенням помилки. Однак, існують певні підходи, що не застосовують алгоритм зворотного поширення помилки, а спираються лише на генетичний алгоритм. Тоді кожна особина оцінюється та впорядковується за рівнем придатності. Визначення придатності може базуватися не лише на продуктивності нейронної мережі, а ще й на розмірі нейронної мережі з метою зробити її якомога меншою. На останньому кроці відбуваються процеси схрещування та мутації, що утворюють особин, які замінюють найгірших, або ж усіх особин у популяції. Проблема об’єднання генетичних алгоритмів та нейронної мережі полягає в кодуванні самої мережі.

Усі підходи до об’єднання нейронних мереж можна розділити на підтримувальні комбінації, де нейронні мережі та генетичні алгоритми діють один за одним та спільні, де вони діють одночасно. За підтримувального підходу генетичні алгоритми та нейронні мережі працюють із різними частинами задачі. Найчастіше генетичний алгоритм застосовують для попередньої обробки набору даних, на якому тренується нейронна мережа. Генетичний алгоритм може бути застосовано для зменшення цього набору даних за допомогою виключення зайвих або непотрібних ознак. За спільного підходу генетичні алгоритми та нейронні мережі використовуються одночасно для утворення популяції нейронних мереж, що найкраще пасує для розглянутої задачі.   
 Цей підхід може застосовуватися для навчання нейронної мережі за допомогою кодування ваг зв’язків як двійкові або дійсні числа або ж для розвитку та вибору архітектури нейронної мережі, спільно з еволюцією ваг зв’язків, або ж окремо від неї. За комбінованого процесу генетичний алгоритм використовується для тренування нейронної мережі. Серед можливих способів його використання є тренування нейронної мережі за допомогою зміни ваг зв’язків, вибір топології нейронної мережі, тобто встановлення того, яким чином нейрони сполучено між собою та вибір оптимальної конфігурації нейронної мережі й параметрів, що визначають її. Генетичні алгоритми можуть застосовуватися й для оптимізації нейронних мереж, що використовують алгоритм зворотного поширення помилки, замінюючи його як правило навчання, визначаючи найкращу архітектуру нейронної мережі чи адаптуючи саме правило навчання, змінюючи його параметри чи самі формули, що використовуються для оновлення ваг зв’язків [21].

## Прикладна задача про визначення можливості руйнування структурних елементів

Протягом робочого процесу механічні навантаження можуть викликати ушкодження опорних структурних елементів. Причинами таких ушкоджень є невеликі дефекти чи тріщини, наявні з самого початку чи такі, що з’явилися протягом процесу роботи. Зазвичай спочатку тріщини наростають стабільно, тобто з кожним циклом навантаження розмір тріщини незначно збільшується. Це називають наростанням втомної тріщини. Втомна тріщина може зростати протягом тисяч циклів навантаження перед тим, як зростання перетвориться на нестабільне. При досягненні певного критичного ліміту тріщина розпочинає нестабільне зростання, що змушує певні структурні елементи чи всю конструкцію втратити свою придатність до виконання задачі. [22]

Тому важливим є вивчення процесу наростання втомних тріщин і розв’язання задачі, що дозволить передбачати його та визначати можливість руйнування опорних структурних елементів, зважаючи на їхній склад і поточний рівень руйнування за умови регулярних циклічних навантажень на них.

Задача про визначення можливості руйнування структурних елементів складного об’єкту полягає у визначенні тривалості роботи складного об’єкту, залежно від швидкості зростання тріщини, заданої за допомогою функції *f()*, де – проміжок значень коефіцієнту інтенсивності напружень, *R* – величина циклічного навантаження.

Швидкість зростання тріщини задається за допомогою рівняння:

де *N* – кількість циклів навантаження; *a* – довжина тріщини; , де – найбільше та найменше значення коефіцієнту інтенсивності напружень, відповідно; , де – найменше та найбільше навантаження протягом одного циклу.

Зазвичай K подається за допомогою рівняння:

де *Y* – функція виправлення, що залежить від геометрії структури та самої тріщини. [23]

mode 1, mode 2, mode 3

## Застосування нейронних мереж для розв’язання задачі про визначення можливості руйнування структурних елементів

# ВИСНОВКИ

У ході даної роботи було сформульовано математичну постановку задачі про визначення можливості руйнування структурних елементів, розглянуто сфери її застосування для промисловості та задач із реального життя.

Розглянуто теоретичні принципи й методи машинного навчання та різні моделі, що реалізують їх: дерева ухвалення рішень та їхні види (підсилені дерева, випадкові ліси, обертові ліси), метод опорних векторів та метод *k*-найближчих сусідів.

Розглянуто теоретичні принципи нейронних мереж, описано різні види топології та архітектури нейронних мереж та різні функції активації, що застосовуються в їх роботі, їхні переваги та недоліки. Описано алгоритм навчання нейронних мереж під назвою алгоритм зворотного поширення помилки, розглянуто проблему зникомого градієнту в навчанні нейронних мереж.

Описано еволюційні алгоритми, їх теоретичні принципи, види та можливі варіанти застосування для навчання нейронних мереж. Зосереджено увагу на генетичних алгоритмах та їх застосуванні в навчанні нейронних мереж як альтернативи для алгоритму зворотного поширення помилки.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Miroslav Kubat, Ivan Bratko, Ryszard Michalski. A Review of Machine Learning Methods.
2. M. I. Jordan, T. M. Mitchell. Machine learning: Trends, perspectives, and prospects.
3. А. В. Литвин. Розробка дерев рішень для прогнозування фінансової кризи в страхових компаніях України // Наукові записки, №172, 2015, 59-64 с. Режим доступу: <http://ekmair.ukma.edu.ua/handle/123456789/6795>
4. Tom M. Mitchell. Machine Learning, 1997, 53-54 c.
5. Yann Coadou. Boosted Decision Trees and Applications // EPJ Web of Conferences, №55, 2013, 25 с. Режим доступу: <https://doi.org/10.1051/epjconf/20135502004>
6. Byron P. Roe et al. Boosted Decision Trees as an Alternative to Artificial Neural Networks for Particle Identification // Nuclear Instruments and Methods in Physics Research, №543, 2005, 6 с. Режим доступу: [10.1016/j.nima.2004.12.018](https://arxiv.org/ct?url=https%3A%2F%2Fdx.doi.org%2F10.1016%2Fj.nima.2004.12.018&v=533ce3e2)
7. Byron P. Roe, Hai-Jun Yang, Ji Zhu. Boosted Trees, A Powerful Event Classifier. // [Statistical Problems in Particle Physics, Astrophysics and Cosmology](https://www.worldscientific.com/worldscibooks/10.1142/p446), 2006, 3 с. Режим доступу: [10.1142/9781860948985\_0029](http://dx.doi.org/10.1142/9781860948985_0029)
8. Відбір ознак за допомогою випадкового лісу у системі виявлення вторгнень / Д. М. Вдовичинський, А. М. Родiонов // XV Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених «Теоретичні i прикладні проблеми фізики, математики та інформатики», 25-27 травня 2017 року, м. Київ. – Київ : ВПI ВПК «ПОЛIТЕХНIКА», 2017. – С. 20-22. – Бібліогр.: 8 назв. Режим доступу: <https://ela.kpi.ua/handle/123456789/20712>
9. Gregor Stiglic, Peter Kokolm Juan J. Rodriguez. Rotation of Random Forests for Genomic and Proteomic Classification Problems // Advances in Experimental Medicine and Biology, 2011.
10. Степанюк, Є. Ю. Математичне та програмне забезпечення для аналізу потоків текстових даних : магістерська дис. : 121 Інженерія програмного забезпечення / Степанюк Євгеній Юрійович . - Київ, 2019. - 109 с. Режим доступу: https://ela.kpi.ua/handle/123456789/31665
11. Alex Graves. Supervised Sequence Labelling with Recurrent Neural Networks // Studies in Computational Intelligence, №385.
12. Emile Fiesler and Russell Beale. Handbook of Neural Computation, 1997, 77-78 с.
13. S. Rajasekaran and G.A. Vijayalkshmi Pai. Neural Networks, Fuzzy Logic and Genetic Algorithms: Synthesis and Applications, 17-18 с.
14. Y.H. Zweiri, J.F. Whidborne, L.D. Seneviratne. A three-term backpropagation algorithm // Neurocomputing, №50, 2003, 305-318 с.
15. Ashay Singh and Ankur Singh Bist. Analysis of Activation Functions.
16. Ken Jon M. Tarnate, Dr. Madhavi Devaraj, Joel C. De Goma. Overcoming the Vanishing Gradient Problem of Recurrent Neural Networks in the Iso 9001 Quality Management Audit Reports Classification // International Journal of Scientific & Technology Research, №9, 2020, 6683-6686 с.
17. Zbigniew Michalewicz, Dipankar Dasgupta, Rodolphe G. Le Riche, and Marc Schoenauer. Evolutionary Algorithms for Constrained Engineering Problems, 28 c.
18. Dan Simon. Evolutionary Optimization Algorithms, 2013, 8 с.
19. Joshua D. Knowles, David W. Corne. Evolving Neural Networks for Cancer Radiotherapy.
20. Barbro Back, Teija Laitinen, Kaisa Sere. Neural Networks and Genetic Algorithms for Bankruptcy Predictions // Expert Systems with Applications, №11, 1996, 407-413 с.
21. Mijanur Rahman, Tania Akter Setu. An Implementation for Combining Neural Networks and Genetic Algorithms // International Journal of Computer Science and Technology, №6, 2015, 218-222 с.
22. Iryna Didych, Oleh Pastukh, Yuri Pyndus, Oleh Yasniy. Evaluation of Structural Elements Lifetime by Neural Network.