МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕКОНОМІКИ І МЕНЕДЖМЕНТУ

Кафедра кібернетики та інформатики

КВАЛІФІКАЦІЙНА РОБОТА

освітній ступінь - «Бакалавр»

на тему: **ВІРТУАЛЬНИЙ ТРЕНАЖЕР «МЕРЕЖА ХОПФІЛДА» ДЛЯ ВИБІРКОВОГО ОСВІТНЬОГО КОМПОНЕНТУ «ІНТЕЛЕКТУАЛЬНІ ІНФОРМАЦІЙНІ СИСТЕМИ»**

Виконав: студент спеціальності
126 «Інформаційні системи та технології»

Юлдашова Кумуш Базарбаївна

Керівник: ***Шелехов Ігор Володимирович***
к.т.н., доцент кафедри кібернетики та інформатики

Рецензент: ***Москаленко Альона Сергіївна***
к.т.н., доцент кафедри комп'ютерних наук СумДУ

Суми – 2025

СУМСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ

Факультет Економіки і менеджменту
Кафедра Кібернетики та інформатики
Освітній ступінь *«Бакалавр»*
Спеціальність *126 «Інформаційні системи та технології»*
ОП «Інформаційні системи та технології»

Затверджую:
Завідувач кафедри Світлана АГАДЖАНОВА

«_____ «_____ 202____ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Юлдашовій Кумуш Базарбаївні

1. Тема роботи: Віртуальний тренажер «Мережа Хопфілда» для вибіркового освітнього компоненту «Інтелектуальні інформаційні системи»
2. База практичного дослідження: Центр дистанційного навчання СНАУ.
керівник роботи Шелехов Ігор Володимирович к.т.н., доцент
затверджена наказом по університету від «09» грудня 2024 р. № 4058/ос
3. Строк подання студентом закінченого проекту (роботи) «27» червня 2025 року
4. Вихідні дані до проекту (роботи):
Результати аналітичних досліджень зарубіжних та вітчизняних вчених, інтернет-джерела, нормативні документи за чинним законодавством.
5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
- *виконати постановку задачі;*
- *вибрати середовище для реалізації;*
- *розробити технічне завдання;*
- *розробити програмний продукт відповідно до поставлених задач;*
- *розробити інструкцію для користувача.*

6. Дата видачі завдання: **« 09 » грудня 2024 р.**

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів проекту	Примітка
1	Визначення об'єкту, предмету дослідження, формулювання мети та задач кваліфікаційної роботи, складання плану	*січень 2025 р.*	
2	Підбір та вивчення літературних джерел, технічної документації	*лютий 2025 р.*	
3	Розробка технічного завдання	*березень 2025 р..*	
4	Розробка вебзастосунку	*лютий-квітень 2025 р.*	
5	Створення інструкції користування вебзастосунком для користувачів	*до 14 травня 2025 р.*	
6	Оформлення пояснювальної записки	*до 30 травня 2025 р..*	
7	Оформлення презентації	*до 1 червня 2025 р.*	
8	Опрацювання зауважень керівника, перевірка на автентичність	*5 червня 2025 р.*	
9	Підготовка доповіді та її попередній захист	*09 червня 2025 р.*	
10	Перевірка з академічної доброчесності кваліфікаційної роботи	*09 червня 2025 р.*	
11	Представлення кваліфікаційної роботи на кафедру, деканат	*13 червня 2025 р.*	
12	Захист кваліфікаційної роботи	*27 червня 2024р.*	

Студент _____ ***Кумуш ЮЛДАШОВА***

(підпис)

Керівник роботи _____ ***Ігор Шелехов***

(підпис)

Перевірку на автентичність проведено _____ ***Надія БАРАННІК***

(підпис)

Кваліфікаційна робота допущена до захисту _____ ***Світлана ЛУКАШ***

(підпис)

АНОТАЦІЯ

Юлдашова К. Б. Віртуальний тренажер «Мережа Хопфілда» для вибіркового освітнього компоненту «Інтелектуальні інформаційні системи».

Кваліфікаційна робота зі спеціальності 126 «Інформаційні системи та технології» ОП Інформаційні системи та технології Сумського національного аграрного університету. Суми, 2025 р. – Рукопис.

Кваліфікаційна робота присвячена розробці віртуального тренажера для моделювання роботи мережі Хопфілда в рамках вибіркового освітнього компоненту «Інтелектуальні інформаційні системи». У роботі розглянуто теоретичні засади функціонування асоціативних нейронних мереж, зокрема мережі Хопфілда, та їх застосування в задачах розпізнавання образів і відновлення пошкоджених даних. Проаналізовано сучасні засоби візуалізації та програмної реалізації нейронних мереж, а також методи навчання та оптимізації параметрів. Розроблений тренажер призначений для візуального представлення роботи мережі Хопфілда, що дозволяє користувачам експериментально досліджувати її поведінку, змінювати параметри та спостерігати результати динаміки станів нейронів у режимі реального часу. Тренажер має інтерактивний інтерфейс, зручний для використання в освітньому процесі.

Об'єктом дослідження є процес моделювання та візуалізації нейронної мережі, що реалізує асоціативну пам’ять, а предметом дослідження – моделі, методи та інформаційна технологія моделювання та візуалізації роботи мережі Хопфілда. Метою роботи є створення інтерактивного програмного засобу, який сприяє ефективному засвоєнню принципів роботи нейронних мереж. Проведено тестування функціональності тренажера на прикладах типових конфігурацій мережі Хопфілда. Результати тестування підтвердили правильність реалізації алгоритмів та ефективність тренажера як навчального інструменту.

Ключові слова: Мережа Хопфілда, віртуальний тренажер, асоціативна пам’ять, моделювання штучної нейронної мережі, штучний інтелект.

SUMMARY

Uldashova K.B. Virtual Simulator "Hopfield Network" for the Elective Educational Component "Intelligent Information Systems".

Qualification work on specialty 126 «Information systems and technologies» OP Information systems and technologies of the Sumy National Agrarian University. Sumy, 2025 - Manuscript.

The qualification thesis is dedicated to the development of a virtual simulator for modeling the operation of a Hopfield network within the elective educational component "Intelligent Information Systems." The work examines the theoretical foundations of associative neural networks, particularly the Hopfield network, and their applications in pattern recognition and damaged data recovery tasks. Modern tools for visualization and software implementation of neural networks, as well as methods for training and parameter optimization, are analyzed.

The developed simulator is designed to visually represent the operation of the Hopfield network, allowing users to experimentally explore its behavior, modify parameters, and observe the results of neuron state dynamics in real time. The simulator features an interactive interface that is convenient for use in the educational process.

The object of the study is the process of modeling and visualizing a neural network that implements associative memory, while the subject of the study includes the models, methods, and information technology for modeling and visualizing the operation of the Hopfield network. The aim of the work is to create an interactive software tool that facilitates the effective learning of neural network principles.

The simulator's functionality was tested using typical Hopfield network configurations. The test results confirmed the correctness of algorithm implementation and the effectiveness of the simulator as a learning tool.

Keywords: Hopfield network, virtual simulator, associative memory, artificial neural network modeling, artificial intelligence.

ЗМІСТ

ВСТУП

Актуальність теми. Актуальність обраної теми зумовлена зростаючим значенням нейронних мереж у розвитку сучасних інформаційних технологій та необхідністю забезпечення якісної підготовки фахівців у галузі штучного інтелекту. Мережа Хопфілда, як класична модель асоціативної пам'яті, посідає важливе місце в системі формування базових знань з нейрообчислень, оскільки дозволяє продемонструвати фундаментальні принципи функціонування штучних нейронних мереж. Попри відносну простоту математичної моделі, її практична реалізація та візуалізація потребують спеціалізованих засобів, які б забезпечували доступність і наочність викладу матеріалу.

У сучасних умовах, коли освітній процес дедалі частіше орієнтується на інтерактивні, практикоорієнтовані методи навчання, розробка віртуального тренажера для моделювання роботи мережі Хопфілда є доцільною та своєчасною. Такий інструмент не лише сприяє глибшому розумінню складних теоретичних понять, але й дозволяє формувати навички самостійного експериментування, що є необхідним компонентом підготовки висококваліфікованих фахівців. Крім того, у контексті зростання ролі дистанційної та змішаної форми навчання, створення програмних засобів, які забезпечують інтуїтивно зрозумілий інтерфейс і підтримку моделювання в реальному часі, набуває особливої цінності.

Таким чином, розробка віртуального тренажера для вивчення особливостей функціонування мережі Хопфілда відповідає актуальним тенденціям розвитку освітніх технологій та забезпечує практичну реалізацію компетентнісного підходу у підготовці фахівців з інтелектуальних інформаційних систем.

Мета і завдання дослідження. **Мета** роботи полягає у розробці віртуального тренажера для моделювання роботи мережі Хопфілда, який може бути інтегрований у наявну платформу дистанційного навчання закладу вищої освіти та слугуватиме ефективним засобом підтримки освітнього процесу з дисципліни «Інтелектуальні інформаційні системи».

Відповідно до поставленої мети сформульовані такі задачі:

- дослідити теоретичні основи побудови та функціонування рекурентних нейронних мереж, зокрема структуру та принципи роботи мережі Хопфілда як моделі асоціативної пам’яті;
- проаналізувати особливості навчання рекурентних нейронних мереж та механізми стабілізації їх динаміки, включаючи алгоритми енергетичної мінімізації;
- обґрунтувати вибір програмного середовища та технологій для реалізації тренажера з урахуванням вимог сумісності з платформою дистанційного навчання, що використовується у ЗВО;

  розробити технічне завдання на створення програмного продукту, що включає функціональні та нефункціональні вимоги;
- реалізувати віртуальний тренажер відповідно до поставлених задач, забезпечивши інтерактивність, наочність та можливість експериментального дослідження поведінки мережі Хопфілда;
- забезпечити можливість інтеграції тренажера в інфраструктуру електронного навчання з метою його практичного використання в освітньому процесі;
- перевірити працездатність розробленого віртуального тренажеру.

Об’єктом дослідження є процес моделювання та візуалізації нейронної мережі, що реалізує асоціативну пам’ять.

Предметом дослідження є моделі, методи та інформаційна технологія моделювання та візуалізації роботи мережі Хопфілда

Апробація результатів кваліфікаційної роботи. Результати кваліфікаційної роботи апробовані на міжнародній науковій конференції молодих учених «Інформатика, математика, автоматика : 2025» (21–25 квітня 2025 року), (Суми – Астана). (Доповіді англійською і українськими мовами відповідно на теми:

1) «Methodology and Tools for Interactive Testing Tasks in Base Recurrent Neural Networks»
2) «Концепція та функціональні можливості інтерактивного тренажеру для вивчення рекурентних нейронних мереж Хопфілда»

Публікації. За матеріалами кваліфікаційної роботи опубліковано 2 тези наукових конференцій.

Структура і обсяг роботи. Кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел з 21 найменувань, додатків. Основний текст викладений на 89 сторінках комп'ютерного тексту, робота містить 6 таблиць, 4 додатки.

РОЗДІЛ 1

АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Інформаційно-комунікаційні технології в освіті

Сучасна система освіти зазнає значних трансформацій під впливом розвитку інформаційно-комунікаційних технологій (ІКТ), що забезпечують нову якість навчального процесу. Перехід від традиційних форм навчання до цифрових середовищ відкриває широкі можливості для підвищення доступності, ефективності та персоналізації навчання. ІКТ-технології дозволяють реалізувати індивідуальний підхід, що відповідає як принципам компетентнісно зорієнтованої освіти, так і сучасним вимогам до професійної мобільності випускників.

Використання ІКТ у загальноосвітніх і вищих навчальних закладах спричинило появу різноманітних моделей навчання, зокрема дистанційної, змішаної (blended learning) та інвертованого класу (flipped classroom). Змішане навчання зарекомендувало себе як ефективна модель, що поєднує переваги онлайн-інструментів та офлайн-взаємодії: дослідження показують, що така модель забезпечує вищі показники успішності та мотивації студентів у порівнянні з традиційною формою без використання цифрових засобів. Аналіз показників впливу ІКТ у змішаних форматах виявив значне покращення якості засвоєння матеріалу, що зумовлено більш гнучкими і персоналізованими навчальними траєкторіями.

Особливою перевагою ІКТ є можливість організації дистанційного навчання, що стало особливо актуальним у XX столітті, а особливо після 2020 року, коли пандемія COVID-19 спричинила світову хвилю переходу до онлайн-освіти. Розвиток платформи електронного навчання (LMS) дозволив реалізувати повноцінний навчальний процес — лекції, семінари, контроль знань — поза межами фізичного аудиторного простору. За даними досліджень [1-3], понад 25 % студентів у США у 2019 році брали щонайменше один онлайн-курс, при цьому

12,5 % навчалися виключно дистанційно . У 2017 році кількість онлайн-курсів на платформі Coursera досягала 3000, а на edX — 1820, що свідчить про значне зростання попиту та пропозицій у сегменті цифрової освіти. Значний внесок у розвиток практичного освітнього середовища роблять віртуальні лабораторії. Проєкт Virtual Labs в Індії дозволяє студентам проводити експерименти в онлайн-середовищі без необхідності доступу до реальних лабораторій.

Поряд із цим, PhET Interactive Simulations [2] створила понад 125 інтерактивних симуляцій з фізики, хімії, біології та математики, що значно підвищують якість навчання та охоплюють різні мови світу. Дослідження демонструють, що такі симуляції не лише покращують розуміння концепцій, але й підвищують рівень залученості учнів.

Використання мультимедіа – ключовий аспект інтеграції ІКТ, який включає відео-уроки, інфографіку, анімації. Дослідження показали, що відео-асистоване навчання сприяє глибшому розумінню матеріалу, зменшенню навантаження на викладача та покращенню оцінок студентів . Популярність відеоплатформ, таких як YouTube, препароване навчання за допомогою інструментів на кшталт Microsoft Forms та Google Forms для оцінювання попередніх знань – визнається викладачами як ефективна практика.

Однією з важливих складових діяльності закладів є інтеграція ІКТ у внутрішню професійну культуру університетів. Значна кількість викладачів потребує постійного підвищення кваліфікації та доступу до підтримки – як адміністративної, так і технічної . Саме тому успішні ініціативи включають як створення MOOC, так і організацію змішаного навчання – на зразок PLAY&LEARN DIGIMEDIA [1], де здійснюється безперервна підтримка викладачів у частині цифрових компетенцій.

Окремо слід виділити роль ІКТ у формуванні інклюзивного навчального середовища. VR/AR-технології дозволяють створювати середовища з високою візуалізацією та залученням, зменшуючи просторові та тимчасові бар’єри. Студенти із обмеженою мобільністю можуть віртуально відвідати музеї,

лабораторії, історичні об’єкти . Дослідження [2, 5] показують підвищену мотивацію та кращу комунікацію в групах, де застосовуються такі технології.

ІСТ також сприяє розвитку колаборативного навчання: системи підтримки спільної роботи (CSCL) [3] показують, що студенти підвищують свою впевненість, мотивованість та залученість у групові проєкти . Використання інструментів на кшталт Clicker-систем, Socrative, Kahoot, Zoom, MS Teams – все це стало звичним способом взаємодії студентів і викладачів у реальному часі й асинхронно.

Важливо підкреслити, що ІКТ дозволяють здійснювати збір аналітичних даних – відвідуваність, результати тестувань, поведінку в LMS, що дає можливість проводити адаптивне навчання на базі отриманих метрик. Учителі завдяки цим інструментам можуть вчасно отримувати зворотний зв’язок, будувати персоналізовані траєкторії для кожного учня або студента.

Отже, інформаційно-комунікаційні технології в освіті є ключовим чинником модернізації навчального процесу. Вони забезпечують інтерактивність, індивідуалізацію, гнучкість і масштабованість. Використання віртуальних симуляцій, колаборативних середовищ, мультимедійних засобів та аналітики забезпечує освіту, що відповідає викликам 21-го століття – будь то масове використання MOOCs, підтримка інклюзивності, або обробка великих даних для оптимізації навчання.

Для закладів вищої освіти це означає необхідність забезпечити інфраструктуру для дистанційного та змішаного навчання, технічну й методичну підтримку викладачів, а також угоди з платформами LMS. У цьому контексті інтеграція вузькоспеціального тренажера – наприклад, симулятора мережі Хопфілда – в дистанційну платформу представляється не лише технологічно обґрунтованою, але й педагогічно доцільною. Створення такого модуля дозволить візуалізувати складні концепції нейронних мереж, надати студентам можливість експериментувати, а викладачам – отримати аналітичні дані щодо ефективності навчання.

Таким чином, розвиток та впровадження ІКТ у сфері освіти є не просто тенденцією, а стратегічною необхідністю, що дозволяє входити в цифрову еру підготовки фахівців з найвищим рівнем професійних і цифрових компетенцій.

1.2. Інноваційні підходи до навчання з використанням віртуальних тренажерів

Сучасна освіта перебуває у стані трансформації, що зумовлена впливом цифрових технологій, глобалізаційних процесів та зростаючих вимог до якості підготовки фахівців. У цих умовах все більшої актуальності набувають інноваційні підходи до організації освітнього процесу, серед яких особливе місце посідає впровадження віртуальних тренажерів як засобу моделювання професійної діяльності в цифровому середовищі. Цей інструмент поєднує у собі потенціал інформаційно-комунікаційних технологій, гейміфікації, моделювання, адаптивного навчання та індивідуалізації освітніх траєкторій.

Віртуальні тренажери (або симулятори) — це програмні засоби, які дозволяють імітувати функціонування об’єктів або процесів у віртуальному середовищі, наближеному до реального [4]. Вони забезпечують користувача можливістю взаємодії з цифровим середовищем за допомогою інтерфейсів, які імітують реальні умови професійної діяльності або навчання. У контексті освітньої діяльності такі засоби використовуються для відпрацювання практичних навичок, прийняття рішень у змодельованих ситуаціях, опанування алгоритмів виконання певних операцій або дій, а також для перевірки і самоконтролю знань. При цьому віртуальні тренажери створюють безпечне та контрольоване середовище, що дозволяє здійснювати багаторазове відтворення навчального процесу без ризику втрат ресурсів чи помилок, небезпечних у реальних умовах.

Застосування віртуальних тренажерів у навчанні ґрунтується на концепціях конструктивістської та когнітивістської педагогіки [5-6], які підкреслюють важливість активної діяльності учня, побудови знань через практичну взаємодію з предметним середовищем, а також на принципах проблемно-орієнтованого та

ситуаційного навчання. У цьому контексті тренажер виступає не лише інструментом, що відтворює дії, а й джерелом когнітивних викликів, які активізують аналітичне мислення, сприяють формуванню цілісного розуміння системи знань, з якою працює здобувач освіти.

Суттєвою перевагою віртуальних тренажерів є можливість персоналізації навчального процесу. Завдяки вбудованим системам моніторингу та адаптації до індивідуального рівня підготовки користувача, тренажери можуть автоматично налаштовувати складність завдань, надавати підказки, фіксувати типові помилки та будувати траєкторії подальшого навчання. Такий підхід особливо ефективний в умовах змішаного або дистанційного навчання, де фізична присутність викладача обмежена, а зворотний зв’язок є критичним для підтримання мотивації та якості засвоєння матеріалу.

Віртуальні тренажери застосовуються в різних галузях освіти, включаючи медицину, авіацію, інженерію, програмування, економіку, управління тощо. Наприклад [6], у медичній освіті активно використовуються симуляційні середовища, такі як Body Interact, Touch Surgery або SimX, які дозволяють студентам відпрацьовувати клінічні навички на віртуальних пацієнтах. У галузі авіації широко поширені пілотні тренажери, що моделюють поведінку літака в умовах польоту, турбулентності або аварійних ситуацій. У сфері інформатики, зокрема при вивченні нейромереж, використовуються такі симуляційні середовища, як TensorFlow Playground, Teachable Machine, NetSim, які дозволяють експериментувати з архітектурами мереж, параметрами навчання, функціями активації та оцінкою результатів. Такі інструменти сприяють глибшому розумінню складних технічних процесів і дозволяють інтерактивно досліджувати поведінку моделей.

Віртуальні тренажери, інтегровані в середовище систем управління навчанням (LMS), значно підвищують ефективність освітнього процесу. Наприклад, у системі Moodle можлива реалізація тренажерів у вигляді SCORM-модулів або HTML5-додатків з повною підтримкою збирання аналітичних даних про активність студентів, їхні дії, час реагування, кількість спроб тощо. Це дозволяє

викладачеві оцінювати не лише підсумковий результат, а й динаміку навчальної діяльності, що дає підстави для точного коригування навчального маршруту або надання індивідуальних рекомендацій. Подібний підхід уже реалізовано в таких міжнародних проектах, як OpenSim (віртуальні лабораторії для фізики)[4], Labster (віртуальні біологічні лабораторії)[5], Virtual Labs Project (Індія)[6] тощо.

Крім того, віртуальні тренажери мають високу мотиваційну цінність. За даними досліджень, навчання з елементами симуляції та візуалізації значно підвищує зацікавленість студентів, сприяє глибшому розумінню матеріалу та закріпленню знань на практиці. Це особливо актуально для сучасного покоління студентів, які зростали в умовах цифрового середовища і мають високі очікування щодо інтерактивності та візуальної привабливості навчального контенту. Саме віртуальні тренажери здатні забезпечити такий рівень взаємодії, який дозволяє зберігати інтерес до навчання протягом тривалого часу.

Не менш важливим аспектом використання віртуальних тренажерів є їх здатність підтримувати інклюзивність освіти [5-6]. Завдяки можливостям адаптації інтерфейсів, масштабування контенту, голосового супроводу або жестового перекладу, такі тренажери можуть бути використані студентами з особливими освітніми потребами. Це створює додаткові можливості для реалізації принципів рівного доступу до якісної освіти, що відповідає вимогам сучасного інклюзивного підходу.

В умовах швидкого розвитку технологій віртуальні тренажери все частіше поєднуються з такими інноваційними напрямами, як доповнена реальність (AR), віртуальна реальність (VR), гейміфікація та штучний інтелект. Зокрема, у VR-середовищах користувач може повністю зануритися у змодельовану ситуацію, що забезпечує високу ступінь присутності та емпатії. Наприклад [5], система zSpace використовує доповнену реальність для демонстрації тривимірних моделей анатомічних структур або інженерних об’єктів, які можна обертати, збільшувати та взаємодіяти з ними за допомогою стилуса. Водночас, елементи гейміфікації — рейтинги, досягнення, бали — стимулюють навчальну активність та формують позитивне ставлення до навчального процесу.

Проблемним аспектом [4], однак, залишається необхідність методично грамотного впровадження віртуальних тренажерів у навчальний процес. Вони не повинні бути самоціллю або заміною викладання, а мають доповнювати традиційні методи, формуючи збалансовану систему навчання. Для цього необхідно розробляти методики інтеграції симуляторів у курси, навчати викладачів методам їх використання, оцінювати ефективність за критеріями результативності та когнітивного впливу. Варто також враховувати технічні ресурси закладу освіти, доступність інтернету, наявність відповідної платформи підтримки.

Отже, інноваційні підходи до навчання з використанням віртуальних тренажерів ґрунтуються на ідеї активного, індивідуалізованого, дослідницького та технологічно підтриманого засвоєння знань. Це не лише підвищує якість підготовки майбутніх фахівців, але й формує у студентів цифрові компетентності, критичне мислення, здатність до прийняття рішень і самостійного навчання. Впровадження таких інструментів у навчальний процес сприяє реалізації цілей сталого розвитку освіти, відповідає викликам цифрової трансформації та відкриває нові можливості для ефективного і доступного навчання.

1.3 Функціональні можливості платформи Moodle для розробки та впровадження віртуальних тренажерів

У сучасній освітній практиці платформи дистанційного навчання стали невід'ємним інструментом забезпечення доступності, гнучкості та індивідуалізації навчального процесу. Однією з найпоширеніших систем управління навчанням (LMS) є Moodle (Modular Object-Oriented Dynamic Learning Environment), яка активно використовується закладами освіти різних рівнів по всьому світу. Moodle є відкритим програмним забезпеченням з широкими можливостями для організації навчального процесу, створення курсів, інтеграції мультимедійного контенту, а також реалізації адаптивних і симуляційних форм навчання, зокрема — віртуальних тренажерів.

Однією з ключових переваг Moodle [7] є модульна архітектура, яка дозволяє розширювати базову функціональність за допомогою додаткових плагінів і інтеграцій. Це створює сприятливі умови для розробки та впровадження віртуальних тренажерів у навчальні курси. Moodle підтримує різноманітні формати контенту: SCORM, H5P, IMS Content Package, що дозволяє вбудовувати віртуальні тренажери, створені в інших середовищах або за допомогою сторонніх інструментів, без втрати інтерактивності.

Особливу роль у контексті віртуального моделювання відіграє підтримка стандарту SCORM (Sharable Content Object Reference Model) [8], який забезпечує сумісність контенту з різними LMS. Це означає, що створений у зовнішньому середовищі віртуальний тренажер може бути легко інтегрований у Moodle як SCORM-пакет, при цьому зберігається можливість фіксації результатів користувача, перегляду статистики, а також забезпечення зворотного зв’язку. Завдяки цьому Moodle може виступати не тільки як платформа для презентації контенту, а як повноцінне середовище для реалізації навчального експерименту з елементами моделювання.

Платформа також забезпечує можливість створення інтерактивних тренажерів за допомогою інструменту H5P [8] — відкритого фреймворку, що дозволяє розробляти динамічний контент без глибокого програмування. За допомогою H5P можна створити інтерактивні презентації, симуляції процесів, задачі на послідовності, а також елементи тренажерів, які вимагають від студентів активного залучення. Прикладом є створення симуляцій роботи нейронної мережі або логіки переходів її станів, де користувач отримує візуальний і функціональний інтерфейс для взаємодії з навчальною моделлю.

Значною перевагою Moodle у контексті розробки тренажерів є підтримка репозиторіїв файлів та середовищ для вбудованого програмування [7]. Наприклад, використання HTML-блоків, вставка JavaScript- або Python-фрагментів (через інтеграцію з Jupyter Notebook або CodeRunner) дозволяє створювати більш гнучкі моделі тренажерів безпосередньо в курсах Moodle. Особливо це актуально у випадках, коли тренажер вимагає нестандартної логіки взаємодії або симуляції

поведінки складних моделей, таких як нейронні мережі або системи з елементами штучного інтелекту.

Ще одним інструментом, що сприяє реалізації функціоналу віртуального тренажера, є використання активностей типу “Завдання” (Assignment) або “Тест” (Quiz), які можуть бути адаптовані для реалізації елементів тренажера — наприклад, для симулювання поетапного виконання розрахунків або сценаріїв, які потребують від користувача певних дій, після яких система надає результат або зворотний зв'язок. Moodle також дозволяє реалізувати логіку умовного відкриття наступних етапів тренажера на основі попередніх дій користувача.

У рамках розробки повноцінного віртуального тренажера важливою функціональністю є відстеження прогресу користувачів, яка реалізується в Moodle через журнали оцінювання, звіти активності та аналітичні інструменти [7]. Наприклад, при використанні віртуального тренажера для моделювання роботи мережі Хопфілда, можна фіксувати кількість спроб, результати розпізнавання образів, рівень відповідності очікуваному результату, час на виконання моделювання тощо. Ці дані можуть бути використані як для індивідуального аналізу, так і для статистичної обробки результатів групи користувачів.

Інтеграція Moodle з іншими сервісами також відкриває широкі можливості для розробки тренажерів. Наприклад [8], платформа підтримує веб-сервіси та API, які дають змогу взаємодіяти з зовнішніми програмними продуктами, базами даних або симуляційними сервісами. Таким чином, у рамках віртуального тренажера можна забезпечити взаємодію з серверними обчислювальними модулями, які здійснюють складне моделювання, а результати подаються в Moodle у зручному для користувача форматі. Це дозволяє реалізувати складні обчислювальні експерименти з використанням Moodle як інтерфейсного рівня.

Практичні приклади впровадження таких тренажерів демонструють успішні кейси в різних галузях: від технічних спеціальностей (моделювання електричних ланцюгів, нейронних мереж, алгоритмів управління) до медичних дисциплін (симуляція клінічних сценаріїв), де Moodle слугує інтеграційною платформою між навчальним контентом та симуляційними інструментами.

Не менш важливим є аспект адаптивності навчання [8]. Moodle дозволяє створювати адаптивні шляхи проходження тренажера, де сценарії дій користувача можуть змінюватися залежно від попередніх рішень. Це важливо для реалізації так званих “розгалужених симуляцій”, де студент проходить навчальний процес не лінійно, а з урахуванням власної стратегії, знань і помилок. Такий підхід значно підвищує ефективність навчання, оскільки дозволяє індивідуалізувати процес моделювання.

Важливо підкреслити, що Moodle є платформою з відкритим кодом, що дає змогу розробникам реалізовувати власні плагіни для побудови тренажерів, які враховують специфіку навчальних дисциплін. Це особливо актуально для вищих навчальних закладів, які мають внутрішні IT-ресурси та можуть створювати авторські рішення, не обмежуючись наявним функціоналом. Наприклад, у рамках розробки тренажера для моделювання роботи мережі Хопфілда може бути реалізовано окремий модуль, який забезпечує побудову нейронної конфігурації, візуалізацію змін стану нейронів у реальному часі, а також функції зміни вхідних параметрів і запуску алгоритму асоціативного відновлення образів.

Узагальнюючи, можна стверджувати, що Moodle не тільки дозволяє ефективно управляти навчальним контентом, а й виступає як потужне середовище для реалізації віртуальних тренажерів. Його гнучка архітектура, підтримка сучасних технологій інтерактивного контенту, наявність інструментів для контролю знань, збору аналітики та розробки власних розширень створюють передумови для створення повноцінних симуляційних середовищ у межах дистанційного навчання. Саме завдяки цьому Moodle набуває все більшого значення у трансформації традиційних підходів до навчання в бік інноваційних, дослідницьких і експериментальних методик, у тому числі й у сфері підготовки фахівців з інформаційних технологій та штучного інтелекту.

1.4 Постановка задачі

За результатами проведеного аналітичного огляду можна зробити висновок про актуальність і перспективність розробки та впровадження віртуальних тренажерів на платформах для дистанційного навчання у закладах вищої освіти. Тому метою кваліфікаційної роботи є розробка віртуального тренажера для моделювання роботи мережі Хопфілда, який може бути інтегрований у наявну платформу дистанційного навчання закладу вищої освіти та слугуватиме ефективним засобом підтримки освітнього процесу з дисципліни «Інтелектуальні інформаційні системи».

Відповідно до поставленої мети сформульовані такі задачі:

1) дослідити теоретичні основи побудови та функціонування рекурентних нейронних мереж, зокрема структуру та принципи роботи мережі Хопфілда як моделі асоціативної пам’яті;
2) проаналізувати особливості навчання рекурентних нейронних мереж та механізми стабілізації їх динаміки, включаючи алгоритми енергетичної мінімізації;
3) обґрунтувати вибір програмного середовища та технологій для реалізації тренажера з урахуванням вимог сумісності з платформою дистанційного навчання, що використовується у ЗВО;
4) розробити технічне завдання на створення програмного продукту, що включає функціональні та нефункціональні вимоги;
5) реалізувати віртуальний тренажер відповідно до поставлених задач, забезпечивши інтерактивність, наочність та можливість експериментального дослідження поведінки мережі Хопфілда;
6) забезпечити можливість інтеграції тренажера в інфраструктуру електронного навчання з метою його практичного використання в освітньому процесі;
7) перевірити працездатність розробленого віртуального тренажеру.

РОЗДІЛ 2

ОСОБЛИВОСТІ РЕКУРЕНТНИХ НЕЙРОННИХ МЕРЕЖ НА ПРИКЛАДІ МЕРЕЖІ ХОПФІЛДА

2.1 Основні поняття і визначення

У межах сучасної теорії штучного інтелекту нейронні мережі відіграють ключову роль як інструменти машинного навчання, здатні моделювати складні нелінійні залежності між вхідними та вихідними даними. Одним із важливих різновидів штучних нейронних мереж є рекурентні нейронні мережі (РНМ), які характеризуються наявністю зворотних зв’язків між нейронами, що забезпечує здатність системи до відображення динамічних процесів та збереження історії попередніх станів.

Рекурентна нейронна мережа (recurrent neural network, RNN) [9-10] — це тип нейронної мережі, структура якої дозволяє зв'язки між елементами мережі не лише у напрямку "вхід–вихід", як у класичних багатошарових перцептронах, а й у напрямку "вихід–вхід", тобто зворотно. Це забезпечує здатність мережі працювати з послідовностями даних та створює передумови для реалізації механізмів пам’яті. На відміну від прямих (feedforward) мереж, рекурентні моделі можуть враховувати контекст попередніх входів при обробці поточного.

Особливо важливе місце серед РНМ посідає мережа Хопфілда (Hopfield Network) [10], яка є одним із перших прикладів динамічної нейронної моделі з асоціативною пам’яттю. Запропонована Джоном Хопфілдом у 1982 році, ця модель дозволяє зберігати та відновлювати шаблони на основі частково спотворених або неповних вхідних даних, що робить її надзвичайно актуальною для задач розпізнавання образів, класифікації та фільтрації інформації.

Мережа Хопфілда представляє собою повнозв’язну симетричну систему з двійковими або аналоговими нейронами, яка характеризується такими властивостями [10]:

- Симетричність зв’язків: для кожної пари нейронів з номерами i та j виконується рівність вагових коефіцієнтів: $w_{ij} = w_{ji}$.
- Відсутність самозв’язків: кожен нейрон не з’єднаний із самим собою, тобто $w_{ii} = 0$.
- Асоціативна пам’ять: мережа здатна зберігати певну кількість стійких станів, які відповідають локальним мінімумам енергетичної функції. При введенні нового збудження система переходить до одного з таких мінімумів, що відповідає найближчому до вхідного сигналу шаблону.

Основу функціонування мережі Хопфілда становить ідея енергетичного ландшафту (енергетичної функції Ляпунова), згідно з якою кожна конфігурація активності нейронів відповідає певному рівню енергії. Мережа еволюціонує до стану з мінімально можливою енергією, що забезпечує стійке функціонування і дозволяє зберігати інформацію у вигляді локальних мінімумів.

Формальною моделлю [9] для дискретної мережі Хопфілда є набір з n нейронів, кожен з яких приймає значення з множини $\{-1, +1\}$. Стан системи описується вектором $S = (s_1, s_2, \ldots, s_n)$, а зміна стану окремого нейрона відбувається за правилом:

$$s_i\left(t+1\right) = \operatorname{sign}\left(\sum_j w_{ij}\, s_j\left(t\right)\right),$$

де *sign* — знакова функція, а w_{ij} — вага з’єднання між нейронами i та j. Всі нейрони оновлюють свої стани або синхронно, або асинхронно, що визначає різні динамічні режими мережі.

Оскільки мережа Хопфілда має властивість енергетичного спаду (енергія системи з часом не зростає), вона обмежена в кількості шаблонів, які можна надійно зберігати без помилок. Для дискретної моделі це число приблизно дорівнює $0.15\,n$, де n — кількість нейронів [9]. Це обмеження стимулювало

подальший розвиток моделей з більшими можливостями пам’яті та складнішою архітектурою, зокрема Болцманівських машин і сучасних глибоких РНМ.

Варто зазначити, що мережа Хопфілда хоча і є простою за архітектурою, залишається актуальною для освітніх і демонстраційних цілей [10], оскільки дозволяє студентам і дослідникам експериментально дослідити властивості динамічних систем, ідеї асоціативного запам’ятовування та аналізу стабільності. Зокрема, інтерактивні віртуальні тренажери, побудовані на базі моделі Хопфілда, дозволяють моделювати її поведінку в реальному часі, спостерігати динаміку зміни станів мережі, вивчати вплив вагових коефіцієнтів на результати, що значно покращує якість навчального процесу.

У подальших підрозділах буде розглянуто математичну модель мережі Хопфілда, її властивості, а також особливості моделювання і візуалізації на практичному рівні у межах розробленого віртуального тренажера.

2.2 Математична модель

Математична модель мережі Хопфілда ґрунтується на ідеях енергетичної оптимізації в динамічних системах та базується на формалізації взаємодії між нейронами за допомогою вагових коефіцієнтів [11-12]. Для розуміння роботи цієї мережі важливо розглянути її основні компоненти: структуру нейронів, правила оновлення станів та функцію енергії, яка характеризує стабільність системи.

Розглянемо мережу Хопфілда, що складається з n двійкових нейронів, стан яких у момент часу t описується вектором

$$s\left(t\right)=\left(s_1\left(t\right),s_2\left(t\right),...,s_n\left(t\right)\right)$$

де кожний елемент $s_i\left(t\right)$ приймає значення із множини $\{-1,+1\}$. Використання двійкової системи значень, хоча й спрощує модель, дозволяє ефективно

моделювати поведінку біологічних нейронних систем та забезпечує просту інтерпретацію станів активації.

Взаємодія між нейронами задається матрицею вагових коефіцієнтів $W = [w_{ij}]$, $i, j=1,2,...,n$, де w_{ij} — вага з'єднання від нейрона j до нейрона i. Важливою властивістю мережі Хопфілда є симетричність матриці ваг:

$$w_{ij} = w_{ji},$$
$$w_{ii} = 0,$$

що виключає самозв'язки нейронів. Симетричність ваг забезпечує існування функції енергії, яка є спадною в процесі оновлення станів, що гарантує збіжність мережі до одного з локальних мінімумів.

Оновлення станів нейронів відбувається за допомогою правил, що імітують динаміку активації:

$$s_i(t+1) = \text{sign}\left(\sum_j w_{ij} x_j(t) - \theta_i\right),$$

де θ_i — поріг активації i-го нейрона, а sign(·) — знакова функція, яка приймає значення +1, якщо аргумент позитивний, і –1 в іншому випадку. Ця формула відображає правило оновлення стану нейрона на основі зваженої суми сигналів, отриманих від інших нейронів мережі.

Мережа може оновлювати стани нейронів у різних режимах: синхронному, де всі нейрони оновлюються одночасно, та асинхронному, коли нейрони оновлюються по черзі. Асинхронне оновлення більш природне для біологічних систем і забезпечує кращу стабільність роботи моделі.

Для аналізу стабільності мережі Хопфілда [12] вводиться поняття енергетичної функції (функції Ляпунова), яка визначається як:

$$E(s) = -\frac{1}{2}\sum_i \sum_j w_{ij} s_i s_j + \sum_i \theta_i s_i ,$$

Ця функція має властивість монотонного зниження або залишання сталої при кожному кроці оновлення стану, що є наслідком симетрії ваг і визначення правил оновлення. Завдяки цій властивості мережа еволюціонує до стійких станів (локальних мінімумів функції енергії), які відповідають запам'ятаним зразкам або паттернам.

Запам'ятовування зразків у мережі Хопфілда реалізується через налаштування матриці ваг за допомогою навчального алгоритму, який, у класичному випадку, базується на правилі Хебба [12]:

$$w_{ij} = \frac{1}{n}\sum_t x_i^t x_j^t ,$$

де $x^t = (x_1^t, x_2^t, ..., x_n^t)$ — t-й зразок із навчальної множини, а n — кількість нейронів. Кожний зразок кодується вектором із значеннями ±1. Правило Хебба передбачає формування ваг як усередненої зовнішньої продукції запам'ятаних шаблонів, що забезпечує асоціативне збереження інформації.

Обмеження кількості зразків, які можуть бути надійно збережені в мережі, пов'язане з її пам'яттю. Емпіричні дослідження показують, що максимальна кількість стабільно збережених паттернів приблизно дорівнює $0.15n$, після чого відбувається погіршення якості відновлення через перехресні взаємодії [11].

Таким чином, математична модель мережі Хопфілда є потужним інструментом для реалізації асоціативної пам'яті і вирішення задач розпізнавання, кластеризації та оптимізації. Вона надає наочний приклад використання принципів енергетичного спаду та динамічних систем у штучних нейронних мережах.

Подальше розглядання моделі включатиме методи її чисельної реалізації, опис алгоритмів моделювання та інтеграції у віртуальні тренажери для навчальних

цілей, що суттєво сприяє глибшому розумінню принципів роботи рекурентних нейронних мереж.

2.3 Особливості моделювання та візуалізації роботи мережі Хопфілда

Моделювання та візуалізація штучних нейронних мереж, зокрема мережі Хопфілда, становить окремий напрям у галузі обчислювального інтелекту та педагогічних інформаційних технологій [13], адже дозволяє не лише досліджувати поведінку моделей, а й ефективно пояснювати її принципи під час навчання. Особливо це актуально в умовах інтеграції технологій доповненої реальності, гнучких візуалізаційних інтерфейсів та платформ дистанційного навчання.

Мережа Хопфілда, як рекурентна структура, характеризується складною динамікою, яка проявляється у зміні станів нейронів під дією внутрішніх зв’язків. Моделювання такої динаміки передбачає реалізацію декількох етапів: ініціалізацію мережі, налаштування вагових коефіцієнтів, оновлення станів згідно з правилом переходу, моніторинг функції енергії та визначення моменту досягнення стабільного стану. Всі ці етапи повинні бути адекватно відображені у візуальному інтерфейсі, що ускладнює реалізацію тренажера, але водночас підвищує його дидактичну цінність.

При розробці моделей часто використовують дискретну часову сітку [13], де кожен такт відповідає одній ітерації оновлення. У візуальному представленні це дозволяє послідовно демонструвати зміну станів окремих нейронів або всього вектору стану. Для цього доцільно використовувати кольорові індикатори або бінарні мітки у вигляді активованих / неактивованих вузлів. Наприклад, значення +1 може відображатися як заповнений колір, а −1 — як порожній, що дає змогу швидко інтерпретувати поточний стан мережі.

У межах навчального тренажера важливо також надати можливість користувачеві задавати вхідні конфігурації, ініціалізувати частково пошкоджені шаблони, щоб спостерігати, як мережа відновлює цілісний патерн. Такий

функціонал дозволяє наочно демонструвати основну функцію асоціативної пам’яті, яку реалізує мережа Хопфілда.

Крім того, одним з ключових елементів візуалізації є граф функції енергії [13]. У кожному такті роботи мережі обчислюється значення енергії відповідно до математичної моделі, і це значення може бути відображене на графіку в режимі реального часу. Зниження функції енергії до певного мінімуму означає досягнення стабільного стану, що підтверджує коректність роботи алгоритму. Наявність такого графіка в інтерфейсі дозволяє користувачам інтерпретувати процеси стабілізації та збіжності з точки зору фізичних аналогій, що сприяє міждисциплінарному навчанню.

Особливу увагу слід приділити способу відображення матриці ваг *W*. Її можна подавати у вигляді інтерактивної таблиці або матричного графа, що дає можливість змінювати значення ваг вручну або автоматично — через реалізацію навчання за правилом Хебба. Взаємозв’язки між нейронами доцільно візуалізувати як зважені ребра між вузлами, при цьому товщина або прозорість лінії може відповідати величині вагового коефіцієнта.

Ще одним важливим аспектом моделювання є підтримка користувацького керування перебігом симуляції. Надання можливості призупинити, відновити або поетапно виконувати ітерації оновлення мережі дозволяє краще розібратися у зміні її внутрішнього стану. У навчальному контексті це створює умови для дослідницької взаємодії, що підвищує якість засвоєння матеріалу.

З технічної точки зору, реалізація такої візуалізації передбачає використання графічних бібліотек або інструментів інтерактивної анімації. Наприклад [13], у середовищі Python популярними є бібліотеки matplotlib, networkx, PyQt, Tkinter або Plotly. У веборієнтованих середовищах доцільним є використання JavaScript-фреймворків (наприклад, D3.js або Three.js), які забезпечують інтерактивність у браузері. У контексті інтеграції з Moodle доцільно використовувати SCORM-модулі або фрейми з вбудованим JavaScript-кодом, що дозволяє запускати симуляцію безпосередньо у вікні курсу.

Таким чином, процес моделювання та візуалізації роботи мережі Хопфілда вимагає врахування як математичних аспектів функціонування рекурентної нейронної мережі, так і особливостей її динаміки у навчальному контексті. Для ефективної розробки віртуального тренажера необхідно забезпечити підтримку узагальнених (ключових) функціональних компонентів, які охоплюють усі етапи взаємодії користувача з моделлю: від побудови структури до аналізу результатів. Кожен із компонентів виконує конкретну методичну та технічну функцію у візуалізації складних внутрішніх процесів:

1. Інтуїтивно зрозумілий графічний інтерфейс для створення, редагування та візуалізації архітектури мережі. Оскільки мережа Хопфілда базується на повнозв'язній структурі, що складається з множини взаємопов'язаних нейронів, надзвичайно важливо надати користувачеві візуальні інструменти для налаштування цієї структури. Можливість графічно додавати або видаляти нейрони, задавати між ними вагові зв'язки, модифікувати параметри у зручному інтерфейсі сприяє глибшому розумінню її топології. Такий підхід не лише формує інтуїтивне уявлення про архітектуру мережі, а й дозволяє зменшити когнітивне навантаження на студента під час взаємодії з абстрактною математичною моделлю.

2. Модуль візуалізації енергетичного ландшафту для відображення енергетичної функції мережі в динаміці. Ключовою характеристикою мережі Хопфілда є її здатність до самостабілізації, що описується через енергетичну функцію. Візуалізація зміни енергії в процесі оновлення станів нейронів дозволяє не лише підтвердити теоретичні положення про спад енергії, але й емпірично перевірити динаміку збіжності мережі до атракторів. Це особливо важливо в освітньому контексті, де абстрактне поняття функціоналу Ляпунова набуває наочності та стає доступним для аналізу. Модуль також дозволяє виявляти «помилкові» стани чи неглибокі локальні мінімуми, що сприяє розвитку критичного мислення у процесі навчання.

3. Інструменти для дослідження динаміки мережі, покрокового моделювання процесу оновлення станів нейронів. У традиційних реалізаціях мережі Хопфілда зміна станів нейронів може бути синхронною або асинхронною. Покрокове

моделювання дає змогу аналізувати, як кожне окреме оновлення впливає на загальний стан системи. Це необхідно для глибшого розуміння принципів локальної взаємодії між нейронами та механізмів, які забезпечують глобальну стабілізацію мережі. Крім того, такий підхід дозволяє досліджувати вплив різних стратегій оновлення (випадкове, фіксоване, за рівнем збудження) та порівнювати їх ефективність.

4. Редактор вхідних образів для зручного введення і маніпулювання вхідними образами. Мережа Хопфілда функціонує як асоціативна пам’ять, здатна згадувати цілі образи на основі часткових або спотворених входів. Тому критично важливо забезпечити інтерфейс для зручного створення, редагування та тестування таких образів. Редактор має дозволяти легко вводити бінарні шаблони, маніпулювати окремими пікселями (нейронами), а також порівнювати вхідні та відновлені образи. Такий інструмент не лише демонструє здатність мережі до корекції помилок, але й дозволяє досліджувати межі її ємності та стабільності.

5. Модуль візуалізації процесу згадування для демонстрації еволюції стану мережі від вхідного образу до стабільного атрактора. Асоціативна згадка — один із центральних механізмів, що демонструють інтелектуальні властивості мережі Хопфілда. Візуалізація процесу згадування дає змогу користувачам простежити шлях еволюції стану нейронної системи від моменту подання часткового образу до досягнення одного з атракторів. Така динаміка є важливою не лише з точки зору розуміння механізму, але й для аналізу випадків, коли згадування відбувається некоректно або мережа входить у цикл. Цей модуль також відкриває можливості для експериментального дослідження феномену суперпозиції та спотворень у згадуваних образах.

6. Інструменти для аналізу атракторів, включаючи ідентифікацію та візуалізацію стабільних станів мережі. Атрактори є кінцевими стабільними станами мережі, до яких вона прямує незалежно від початкового вхідного образу в межах певного басейну притягання. Тренажер має забезпечувати користувача засобами для автоматичного виявлення таких атракторів, порівняння їх із шаблонами навчання та візуалізації результатів у вигляді таблиць, діаграм або

графічних матриць. Це дозволяє проводити кількісний аналіз ефективності навчання мережі, визначати обсяг її пам’яті та відстежувати появу паразитних атракторів. Таким чином, забезпечується повноцінний інструментарій для емпіричної перевірки властивостей асоціативної пам’яті.

Таким чином, можна стверджувати, що для повноцінного моделювання та візуалізації роботи мережі Хопфілда необхідно реалізувати такі компоненти, які не тільки забезпечують математичну коректність моделі, але й адаптують її до вимог педагогіки. Їх реалізація є необхідною умовою для досягнення головної мети – створення інструменту, який одночасно відповідає критеріям наукової обґрунтованості, функціональності та зручності використання в освітньому процесі.

Завдяки поєднанню математичного моделювання та візуальної інтерпретації, користувачі отримують змогу як проводити експерименти, так і глибше розуміти природу асоціативної пам’яті в нейронних мережах, що значною мірою підвищує ефективність засвоєння матеріалу у межах дисциплін, пов’язаних зі штучним інтелектом та когнітивними технологіями.

РОЗДІЛ 3

ІНФОРМАЦІЙНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Вибір середовища для програмної реалізації

Одним із ключових етапів у процесі створення віртуального тренажера для моделювання роботи мережі Хопфілда є вибір відповідного середовища програмної реалізації. Цей вибір визначається не лише технічними вимогами до функціональності майбутнього програмного засобу, а й специфікою його подальшого впровадження в навчальний процес, зокрема інтеграцією з платформою дистанційного навчання Moodle, яка широко використовується у вищих закладах освіти.

Moodle [7-8] є відкритою системою управління навчанням (LMS), що підтримує розширення функціональності за рахунок додаткових плагінів, SCORM-модулів, HTML-блоків та інтеграції з зовнішніми вебдодатками через API або вбудовані фрейми. З огляду на це, програмний продукт, який передбачається створити, має відповідати кільком критеріям: бути платформонезалежним, працювати в браузері без необхідності встановлення додаткового ПЗ, мати інтерактивний інтерфейс користувача, підтримувати динамічну графіку та реагувати на дії користувача в реальному часі.

На основі цих вимог було обґрунтовано вибір веборієнтованого підходу до реалізації тренажера, із застосуванням JavaScript як основної мови програмування. Вибір саме цієї мови пояснюється її універсальністю у створенні клієнтських вебдодатків, високою інтерактивністю, широким набором бібліотек і фреймворків для візуалізації даних та простотою інтеграції в HTML-сторінки Moodle.

Зокрема, для реалізації математичної логіки роботи нейронної мережі обрано чистий JavaScript [14] або TypeScript у поєднанні з бібліотеками Math.js [15] для обчислень та D3.js, p5.js або Plotly.js [16] для візуалізації. Вказані бібліотеки дозволяють будувати інтерактивні графи, відображати динаміку змін енергетичної

функції, структуру мережі, а також забезпечують підтримку анімації, необхідної для покрокового відтворення процесу оновлення станів нейронів.

Для забезпечення сумісності з Moodle планується використання формату SCORM 1.2 [17-19], який дозволяє обгорнути вебдодаток у вигляді навчального модуля з можливістю збереження результатів роботи користувача. Альтернативно, реалізація може бути виконана як вбудований HTML5-додаток у ресурсі типу «Сторінка» або «Книга», що дає змогу без додаткової інсталяції запускати тренажер безпосередньо в інтерфейсі Moodle.

З метою підвищення доступності та адаптивності, середовище розробки має підтримувати кросбраузерність і мобільну адаптацію інтерфейсу. Для цього застосовується фреймворк Bootstrap або бібліотека компонентів Tailwind CSS, що дозволяють гнучко налаштовувати вигляд тренажера відповідно до різних розмірів екранів [20].

У процесі розробки також розглядається можливість використання платформи Node.js [21] як серверної частини у разі потреби в збереженні результатів моделювання або створенні багаторазового доступу до попередніх експериментів користувача. Проте у базовій версії реалізація буде повністю клієнтською, що значно спрощує розгортання та забезпечує більшу стабільність роботи на стороні користувача.

Таким чином, вибір середовища для реалізації тренажера обумовлений необхідністю забезпечення гнучкої інтеграції з Moodle, високого рівня інтуїтивної візуалізації, підтримки інтерактивної взаємодії з користувачем та можливістю масштабування функціоналу. Комбінація HTML5, JavaScript та відповідних бібліотек дозволяє створити модуль, який може бути використаний у широкому спектрі освітніх дисциплін, пов’язаних з вивченням нейронних мереж та штучного інтелекту, забезпечуючи сучасний і ефективний інструмент для самостійного або супроводжуваного навчання.

В таблиці 3.1 наведено результати узагальнений аналіз функціональних можливостей тренажера у вигляді таблиці з коментарями щодо можливості реалізації у вебсередовищі з інтеграцією до Moodle.

Таблиця 3.1 – Узагальнений аналіз можливості реалізації у вебсередовищі з інтеграцією до Moodle функціональних можливостей тренажера

№	Функціональна можливість	Можливість реалізації	Коментар щодо реалізації
1	Інтуїтивний графічний інтерфейс для створення, редагування і візуалізації мережі	Так	Реалізується з використанням HTML5, CSS, JavaScript та бібліотек Konva.js, Fabric.js або D3.js.
2	Налаштування кількості нейронів та ваг	Так	Динамічне створення елементів інтерфейсу, з можливістю введення або генерації вагових коефіцієнтів.
3	Модуль візуалізації енергетичного ландшафту	Так	Можна реалізувати через Chart.js, Plotly.js або Three.js з динамічним оновленням графіка енергетичної функції.
4	Покрокове моделювання динаміки, станів та енергії	Так	Використання асинхронного моделювання (`setTimeout`, анімації) з оновленням DOM-елементів.
5	Візуалізація станів мережі та переходів до атракторів	Так	Canvas або HTML-графіка для зображення змін стану у вигляді еволюції вхідного образу.
6	Редактор вхідних образів	Так	Піксельна сітка або форми на canvas; зручне введення у векторному або матричному вигляді.
7	Модуль згадування: від вхідного образу до стабільного атрактора	Так	Поступова демонстрація згортання вхідного образу в атрактор; добре ілюструється через графічний canvas.
8	Аналіз атракторів: виявлення, збереження, візуалізація стабільних станів	Так	Порівняння поточних станів із збереженими; реалізація логіки і візуального відображення результатів аналізу.
9	Інтеграція в Moodle	Так	Можлива як SCORM-пакет або HTML5-модуль із запуском в курсі Moodle без додаткових серверних компонентів.

Джерело: запропоновано автором

У розділі 2.3 «Особливості моделювання та візуалізації роботи мережі Хопфілда» описано шість функціональних модулів тренажера. Ці шість модулів відповідають узагальненим дидактичним та технічним завданням тренажера — тобто логічно згрупованим блокам функціональності, які користувач сприймає як цілісні компоненти. Натомість у табл. 3.1 ці функції розгорнуто більш детально. Тобто деякі з тих самих узагальнених функцій розділено на менші, технологічно

окремі підзадачі, щоб дати більш точну оцінку їх реалізовуваності у Moodle. (табл. 3.2)

Таблиця 3.2 – Деталізована технічна декомпозиція концептуальних функцій тренажера

Узагальнена функція (з розділу 2.3)	Відповідні пункти у таблиці 3.1	Пояснення
1. Інтуїтивно зрозумілий графічний інтерфейс для створення, редагування та візуалізації архітектури мережі	1. Інтуїтивний графічний інтерфейс для створення, редагування і візуалізації мережі	Узагальнена функція охоплює дві різні операції: графічну візуалізацію структури мережі, яка стосується інтерфейсу користувача (переміщення, з'єднання, позначення вузлів), формальне конфігурування параметрів нейронної моделі (кількість нейронів, матриця ваг), що потребує окремих засобів введення та перевірки даних.
	2. Налаштування кількості нейронів та ваг	
2. Модуль візуалізації енергетичного ландшафту для відображення енергетичної функції мережі в динаміці	3. Модуль візуалізації енергетичного ландшафту	Ця функція залишилася автономною, бо відповідає за специфічну форму виведення енергетичної функції (наприклад, у вигляді графіків, теплових карт), яка не перетинається напряму з іншими елементами.
3. Інструменти для дослідження динаміки мережі, покрокового моделювання процесу оновлення станів нейронів	4. Покрокове моделювання динаміки, станів та енергії	Функціонально моделювання (розрахунок станів) і візуалізація динамічного процесу – це різні компоненти: №4 реалізує логіку моделювання, запуску та зміни станів за алгоритмами. №5 відповідає за показ змін – графічне чи анімоване подання переходів
	5. Візуалізація станів мережі та переходів до атракторів	
4. Редактор вхідних образів для зручного введення і маніпулювання вхідними образами	6. Редактор вхідних образів	Ця функція є самодостатньою — її задача полягає у формуванні стартових умов для симуляції мережі. Вона не перетинається функціонально з іншими модулями.
5. Модуль візуалізації процесу згадування для демонстрації еволюції стану мережі від вхідного образу до стабільного атрактора	7. Модуль згадування: від вхідного образу до стабільного атратора	Це цілісний логіко-аналітичний блок, який моделює згадування. Він імітує поведінку мережі після подання деформованого вхідного образу й демонструє її здатність до стабілізації (згадування).
6. Інструменти для аналізу атракторів, включаючи ідентифікацію та візуалізацію стабільних станів мережі	8. Аналіз атракторів: виявлення, збереження, візуалізація стабільних станів	Цей компонент працює з результатами моделювання. Він дозволяє оцінити, чи правильно мережа згадала образ, скільки атракторів існує, в якому вони вигляді та як змінюються залежно від навчання.
-	9. Інтеграція в Moodle	Цей компонент забезпечує реалізацію тренажера у вигляді ресурсу, доступного у системі Moodle.

Джерело: запропоновано автором

В табл. 3.2 функція 9. «Інтеграція в Moodle» не входила в перелік узагальнених. Вона є інфраструктурною і необхідна з огляду на умови розгортання в ЗВО, але не є частиною внутрішньої логіки роботи тренажера.

Таким чином, усі функції, передбачені для тренажера, повністю підтримуються сучасними технологіями веброзробки та можуть бути інтегровані в освітнє середовище Moodle шляхом HTML5/JS-орієнтованої реалізації з адаптацією під SCORM або iframe-інтеграції.

3.2 Алгоритм формування навчальних сценаріїв у тренажері

Кількість і типи навчальних сценаріїв у віртуальному тренажері для моделювання мережі Хопфілда мають відповідати як дидактичним, так і технічним можливостям тренажера. Оптимально це 4–6 базових сценаріїв, які охоплюють різні аспекти роботи з мережею Хопфілда:

1. Сценарій ознайомлення з архітектурою

Мета: Ознайомити користувача з базовою структурою рекурентної мережі Хопфілда: нейрони, з'єднання, матриця ваг.

Завдання:

- Створення мережі з заданою кількістю нейронів
- Ознайомлення з правилами симетрії ваг
- Введення тестових ваг вручну або автоматично

2. Сценарій навчання мережі

Мета: Навчити мережу простих бінарних образів (наприклад, літер, цифр).

Завдання:

- Введення одного або кількох образів
- Генерація матриці ваг методом Хопфілда
- Демонстрація того, як образи «запам’ятовуються»

3. Сценарій згадування образу з шумом

Мета: Показати здатність мережі до корекції помилок і згадування.

Завдання:

- Введення спотвореного образу
- Покрокове оновлення станів
- Демонстрація збіжності до атрактора

4. Сценарій дослідження енергетичного ландшафту

Мета: Пояснити поняття енергетичної функції та її мінімумів.

Завдання:

- Побудова графу енергії у просторі станів
- Аналіз динаміки переходів у напрямку мінімізації енергії

5. Сценарій виявлення обмежень

Мета: Демонструвати межі пам’яті мережі Хопфілда.

Завдання:

- Навчання мережі великої кількості образів
- Перевірка її здатності згадати всі
- Виявлення «помилкових атракторів»

6. Сценарій тестування користувача

Мета: Контроль знань і навичок.

Завдання:

- Подано задачу (наприклад, згадати спотворений образ)
- Користувач повинен вручну або автоматично виконати згадування
- Отримати оцінку або зворотний зв’язок

В таблиці 3.3 наведено узагальнені дані щодо кожного сценарію.

Розглянемо кожний зі сценаріїв детально.

Сценарій 1: Ознайомлення з архітектурою мережі Хопфілда

Мета сценарію - надати користувачеві можливість інтуїтивно дослідити архітектуру мережі Хопфілда, ознайомитися з ключовими її компонентами, змінити базові параметри (кількість нейронів, матрицю ваг), а також візуально спостерігати вплив цих змін на структуру мережі.

Таблиця 3.3 – Узагальнені дані навчальних сценаріїв

Сценарій	Освітнє призначення	Тип активності	Дидактичний результат
Ознайомлення з архітектурою	Теоретичне	Інтерактивне моделювання	Розуміння внутрішньої будови мережі.
Навчання мережі	Практичне	Створення образів	Усвідомлення, як навчання відбувається без вчителя, через кореляцію.
Згадування з шумом	Аналітичне	Дослідження згадування	Розуміння принципу атракторної динаміки.
Енергетичний ландшафт	Теоретико-прикладне	Візуалізація	Пов’язування формальної моделі з інтуїтивним уявленням про енергетичні ями
Межі пам’яті	Аналітичне	Дослідницьке	Усвідомлення обмежень моделі (≈0.15**N* образів для *N* нейронів).
Тестування	Контрольне	Автоматизоване оцінювання	Закріплення знань через активну практику.

Джерело: запропоновано автором

Вхідні дані:

- Початкова конфігурація тренажера (наприклад, 4 нейрони, нульова матриця ваг).
- Інтерфейс для введення параметрів: кількість нейронів, значення ваг.
- Візуальний редактор (граф) для відображення зв’язків між нейронами.

Вихідні дані

- Відображення графа мережі з урахуванням заданих параметрів.
- Оновлена матриця ваг.
- Повідомлення про коректність конфігурації мережі (наприклад, відсутність самозв’язків).

Логіка дій користувача (у форматі списку кроків)

1. Користувач запускає тренажер.
2. Обирає пункт «Створити нову мережу».
3. Уводить кількість нейронів (наприклад, 5).
4. Система генерує граф нейронної мережі з заданою кількістю вершин.
5. Користувач за допомогою графічного редактора встановлює зв’язки між нейронами.
6. У спеціальній панелі він може задати або змінити ваги кожного зв’язку.

7. Програма автоматично оновлює матрицю ваг і перевіряє її симетричність.
8. Візуальний інтерфейс відображає мережу, де товщина або колір ліній відповідає силі ваг.
9. Користувач зберігає конфігурацію або переходить до наступного сценарію.

Псевдокод логіки обробки архітектури

```
function createNetwork(neuronCount):
    initialize matrix W[neuronCount][neuronCount] to 0
    for i from 0 to neuronCount-1:
        for j from 0 to neuronCount-1:
            if i != j:
                W[i][j] = userInputWeight(i, j)
                W[j][i] = W[i][j]  // Ensure symmetry
    return W
```

Приклад взаємодії користувача

Користувач вводить: 4 нейрони.

Тренажер створює: 4 вершини у графі.

Користувач додає зв’язки: між нейронами 1–2, 2–3, 3–4.

Редактор ваг: користувач задає ваги 0.8, -0.4, 1.0 відповідно.

Візуалізація: граф з 4 вершинами, зв’язки підписані вагами, колір відповідає знаку.

Матриця ваг: автоматично оновлена і симетрична

Сценарій 2:

Мета сценарію – забезпечення глибшого розуміння механізмів пам’яті та навчання в рекурентних нейронних мережах, а також підготовка до подальшого дослідження процесу згадування та динаміки переходів до атракторів.

Вхідні дані:

- Кількість нейронів у мережі
- Множина бінарних вхідних образів (у вигляді векторів, що складаються з елементів {–1, +1} або {0, 1}, залежно від реалізації)
- Вибір алгоритму навчання (наприклад, правило Хебба)
- (Опційно) параметри симетричності чи нормалізації ваг (залежно від реалізації)

Вихідні дані:

- Матриця ваг зв’язків між нейронами (симетрична, з нульовою діагоналлю)
- Візуалізована структура мережі (нейрони, з’єднання, ваги)
- Статистична інформація про кількість збережених образів і їх стійкість
- Повідомлення про успішне навчання або попередження про можливі конфлікти (наприклад, перенавантаження мережі)

Логіка дій користувача — Навчання мережі

1. Запуск тренажера в середовищі Moodle через інтерактивний модуль.
2. Перехід до розділу «Навчання мережі» в інтерфейсі тренажера.
3. Вибір параметрів мережі:
    - Кількість нейронів.
    - Тип оновлення (синхронний / асинхронний — якщо це впливає на тренування).
4. Введення навчальних образів:
    - Користувач малює образи (наприклад, двійкові патерни) у спеціальному редакторі сітки.
    - Може додавати кілька образів.
    - Візуально контролює список збережених навчальних образів.
5. Перевірка коректності образів:
    - Автоматично перевіряється, чи відповідають вони формату (наприклад, довжина вектора, значення).
6. Початок навчання мережі:
    - Користувач натискає кнопку «Навчити мережу».
    - Тренажер застосовує правило Хебба або іншу обрану стратегію для розрахунку матриці ваг.
7. Візуалізація результату навчання:
    - Виводиться матриця ваг (в числовому або графічному вигляді).
    - Відображається кількість збережених патернів.
8. Збереження параметрів навченої мережі:

- Користувач може зберегти поточну конфігурацію ваг.
- Можливе автоматичне збереження в історії сесії.

9. Перехід до тестування або згадування образів:
   - Сценарій завершується, відкриваючи шлях до Сценарію 3 — згадування (еволюції станів).

Сценарій 3: Сценарій згадування образу з шумом

Мета сценарію – дослідити, як мережа Хопфілда розпізнає образи, задані в навчальному наборі, та як відбувається еволюція її станів від вхідного образу до атрактора.

Вхідні дані:

- Кількість нейронів у мережі. Наприклад: 9 (відповідає образу 3×3).
- Навчальні (еталонні) образи подані у вигляді векторів з елементами {-1, +1}. Наприклад: образ "L" → [-1, -1, -1, 1, -1, -1, 1, 1, 1], образ "T" → [1, 1, 1, 0, 1, 0, -1, 1, -1]
- Спотворений вхідний образ для згадування. Наприклад: [-1, -1, -1, 1, -1, -1, 1, -1, 1]
- Режим оновлення станів нейронів (асинхронний або синхронний)
- Максимальна кількість ітерацій.

Вихідні дані:

- Побудована матриця ваг W
- Послідовність станів мережі
- Значення енергетичної функції на кожному кроці
- Фінальний стан (атрактор)
- Оцінка згадування
- Графічне представлення згадування

Ключові дії користувача та логіка системи (алгоритм)

1. Початок сценарію
2. Користувач відкриває інтерфейс тренажера.
3. Користувач задає кількість нейронів (наприклад, 9 для 3×3 піксельного образу).

4. Користувач формує кілька еталонних образів (наприклад, букви "L", "T") через редактор вхідних образів.
5. Система будує матрицю ваг згідно з правилом Хопфілда.
6. Користувач вводить спотворений варіант одного з образів.
7. Система запускає ітеративний процес згадування:
   - Візуалізація поточного стану мережі.
   - Обчислення енергетичної функції.
   - Оновлення станів нейронів (асинхронно або синхронно).
   - Перехід до нового стану.
8. Повторюється, доки мережа не досягне стабільного стану (атрактора).
9. Система порівнює атрактор із відомими еталонними образами.
10. Користувач переглядає результати:
    - Який образ було згадано.
    - Динаміка енергетичної функції.
    - Візуалізація послідовності станів.
11. Кінець сценарію

Псевдокод логіки згадування у мережі Хопфілда

```
// === ІНІЦІАЛІЗАЦІЯ ===
// Кількість нейронів у мережі
N ← розмір образу (наприклад, 9 для 3x3)
// Завантаження навчальних образів
training_patterns ← список векторів образів розміром N
// Побудова матриці ваг за правилом Хопфілда
W ← нульова матриця розміром N×N
for each pattern x in training_patterns:
    W ← W + outer_product(x, x)
end for
// Обнулення діагоналі (немає самозв'язків)
for i from 1 to N:
    W[i][i] ← 0
end for
// Вхідний (спотворений) образ
S ← вхідний_вектор // розмір N
// Встановити параметри
```

```
mode ← "asynchronous" or "synchronous"
max_iterations ← за замовчуванням 100
iteration ← 0
history ← []
// === ДИНАМІКА ОНОВЛЕННЯ СТАНІВ ===
repeat
    previous_state ← S.copy()
    history.append((S, energy(W, S)))
    if mode == "synchronous":
        new_S ← sign(W × S)  // Векторне оновлення
        S ← new_S
    else:
        for i from 1 to N:
            h ← sum_over_j(W[i][j] * S[j])  // Потенціал нейрона i
            S[i] ← sign(h)
        end for
    end if
    iteration ← iteration + 1
until S == previous_state or iteration >= max_iterations
// === ВИВІД РЕЗУЛЬТАТІВ ===
// Вивести фінальний стан S, порівняти його з навчальними образами,
побудувати граф зміни енергії, Візуалізувати послідовність переходів
```

де `outer_product(x, x)` – формує вклад образу x у матрицю зв'язків,

`sign()` – функція активації:

$$sign\left(x\right)=\begin{cases}+1, & якщо\ x>0;\\ -1, & якщо\ x<0;\\ 0, & якщо\ x=0;\end{cases}$$

`energy(W, S)` – функція енергії:

$$enegry\left(W,s\right)=-\frac{1}{2}\sum_{i,j}W_{ij}s_is_j\ .$$

Логіка дій користувача

1. Запуск тренажера з головного меню або відповідного модуля в курсі Moodle.
2. Перехід до модуля «Навчання та згадування» у тренажері.
3. Введення навчальних образів (одного або кількох) за допомогою редактора образів:
   - Обирає розмір сітки (наприклад, 3×3, 5×5).
   - Вводить вектори образів у вигляді клітинок (чорне/біле або +1/–1).
   - Зберігає образи до списку навчальних.
4. Автоматичне або вручну ініційоване обчислення матриці ваг для мережі на основі введених образів (тренажер може це зробити автоматично).
5. Введення вхідного (спотвореного) образу, який частково або повністю не збігається з навчальними.
6. Вибір режиму згадування:
   - Синхронне чи асинхронне оновлення нейронів.
   - Покроковий або автоматичний режим.
7. Запуск процесу згадування:
   - Візуально спостерігає зміну станів нейронів на сітці.
   - Переглядає значення енергії на кожному кроці.
8. Завершення еволюції при досягненні стабільного стану або після заданої кількості ітерацій.
9. Порівняння отриманого атрактора зі збереженими навчальними образами:
   - Відображення результатів (збіг/незбіг).
   - Виведення коментарів (наприклад: «згадано образ №2»).
10. Повторне тестування:
    - Можна змінити образ або додати нові та повторити згадування.

Сценарій 4: Дослідження енергетичного ландшафту мережі Хопфілда

Метою є наочне розуміння процесів збіжності мережі до атракторів, характеру мінімумів енергетичної функції та оцінки стійкості збережених образів.

Вхідні дані:

- Матриця ваг зв’язків між нейронами (результат попереднього навчання)
- Початковий вхідний образ (може бути повністю або частково пошкодженим)
- Параметри візуалізації (наприклад, режим покрокової динаміки або повна симуляція)
- (Додатково) вибір розміру області для побудови ландшафту (наприклад, 2D-проекція у просторі образів)

Вихідні дані:

- Графічна візуалізація енергетичного ландшафту: діаграма зміни енергії в динаміці
- Граф еволюції станів мережі (стан → наступний стан) з нанесеними енергетичними значеннями
- Таблиця або графік зі значеннями енергії на кожному кроці оновлення нейронів
- Ідентифіковані локальні мінімуми (атрактори)
- (Опційно) анімація процесу збіжності

Логіка дій користувача (кроки):

1. Відкрити модуль "Дослідження енергетичного ландшафту"
2. Завантажити або вибрати раніше збережену натреновану мережу
3. Ввести початковий образ або вибрати один із шаблонів
4. Визначити параметри моделювання (режим симуляції, глибину дослідження)
5. Запустити симуляцію зміни станів мережі
6. Спостерігати зміну енергетичних значень у реальному часі
7. Аналізувати отриманий ландшафт (відзначити точки локальних мінімумів)
8. Повторити експеримент з іншим вхідним образом за потреби
9. Зберегти результати аналізу або експортувати графіки для звіту

Сценарій 5: Виявлення обмежень пам’яті мережі Хопфілда

Мета сценарію: Надати користувачу можливість дослідити обмеження асоціативної пам’яті мережі Хопфілда, зокрема з’ясувати, яка максимальна кількість образів може бути коректно збережена і відновлена, а також продемонструвати виникнення помилкових атракторів при перевантаженні мережі.

Вхідні дані:

- Задана кількість нейронів у мережі
- Набір бінарних вхідних образів для навчання (розмірність повинна відповідати кількості нейронів)
- Початкові образи для перевірки згадування (можуть бути пошкодженими)
- Параметри тестування: кількість повторень, варіанти шуму

Вихідні дані:

- Інформація про те, які образи мережа успішно згадує, а які – ні
- Перелік і візуалізація «помилкових атракторів» (станів, які не відповідають жодному з навчальних образів)
- Таблиця зі статистикою успішності згадування залежно від кількості образів
- Візуалізація зміни точності згадування при збільшенні навантаження (наприклад, графік точність / кількість образів)
- (Додатково) повідомлення або звіт про перевищення теоретичної межі збереження (0.15 * N)

Логіка дій користувача (кроки):

1. Запустити модуль "Виявлення обмежень пам’яті"
2. Встановити кількість нейронів у мережі (наприклад, N = 20)
3. Вибрати кількість навчальних образів (починати з 1 і збільшувати до критичної межі)
4. Ввести або згенерувати набір бінарних навчальних образів
5. Запустити навчання мережі на цьому наборі
6. Ввести або вибрати тестові образи (можуть бути як ідеальними, так і з шумом)

7. Запустити процедуру згадування для кожного з них
8. Зафіксувати успішні випадки згадування та випадки переходу до помилкових атракторів
9. Провести серію експериментів із поступовим збільшенням кількості образів
10. Ознайомитися з підсумковими графіками та висновками про обмеження мережі

Сценарій 6: Тестування користувача

Мета сценарію:

Забезпечити можливість оцінювання рівня засвоєння користувачем теоретичних знань і практичних навичок роботи з мережею Хопфілда через виконання контрольного завдання з подальшим автоматизованим або напівручним аналізом результатів.

Вхідні дані:

- Випадково або вручну згенероване завдання (наприклад, частково спотворений образ, який потрібно згадати)
- Навчена мережа (збережений набір ваг і образів, які використовувалися під час навчання)
- Обраний режим перевірки (автоматичний / ручний / напівручний)
- (Опційно) критерії оцінювання (точність згадування, кількість кроків до збіжності, відповідність атрактору, наявність помилок)

Вихідні дані:

- Стан згаданої мережі (кінцевий атрактор)
- Порівняння результату з еталонним образом
- Автоматичне формування оцінки або надання якісного зворотного зв’язку (коментарі, вказівки на помилки, рекомендації щодо покращення)
- Статистика результатів (кількість правильних / неправильних дій, час виконання, кількість спроб)

Логіка дій користувача (поетапно):

1. Увійти до модуля "Тестування користувача"
2. Вибрати тип завдання (наприклад, згадати частково пошкоджений образ)
3. Отримати або згенерувати вхідний образ для згадування (автоматично або вручну)
4. Виконати моделювання згадування:
   - вручну задаючи оновлення станів нейронів крок за кроком
   - або запустити автоматичний процес згадування
5. Перевірити результат: тренажер порівнює отриманий стан з еталонним образом
6. Отримати зворотний зв’язок:
   - текстове повідомлення з оцінкою (наприклад, «образ згадано повністю / частково / невірно»)
   - підказки або пояснення щодо помилок, якщо такі були
   - числова оцінка або бал
7. При необхідності — повторити спробу або перейти до нового завдання

Таким чином, розробка 6 навчальних сценаріїв для віртуального тренажера моделювання мережі Хопфілда дозволяє повністю охопити ключові етапи взаємодії з рекурентною нейронною мережею — від ознайомлення з її архітектурою до дослідження обмежень пам’яті та контролю засвоєного матеріалу. Така структура відповідає як дидактичним цілям навчального процесу (поступове ускладнення завдань, активне застосування знань, формувальне оцінювання), так і технічним можливостям середовища реалізації (інтерактивність, візуалізація, сценарна логіка). Крім того, ці сценарії створюють підґрунтя для гнучкої адаптації тренажера під різні освітні рівні — від ознайомчих курсів до дослідницьких лабораторних практикумів.

3.3 Короткий опис програмної реалізації

Програмна реалізація віртуального тренажера для моделювання мережі Хопфілда здійснена з використанням вебтехнологій HTML, CSS та JavaScript.

Компоненти тренажера розроблено таким чином, щоб забезпечити їх гнучку інтеграцію в систему дистанційного навчання Moodle за допомогою SCORM-модуля або вставки у форматі HTML-коду до навчального ресурсу типу «Сторінка» або «Книга». Особливу увагу приділено модульності реалізації функціоналу, відповідно до шести основних навчальних сценаріїв, кожен з яких забезпечується відповідним набором функцій.

Сценарій 1: Ознайомлення з архітектурою мережі

Призначення функцій:

Користувач має змогу створити мережу з довільною кількістю нейронів, редагувати з’єднання між ними та задавати ваги вручну або автоматично.

Основні функції:

- createNeuralNetwork(n): створює повнозв’язну симетричну матрицю ваг n × n;
- updateWeight(i, j, value): оновлює вагу між нейронами i та j;
- renderNetworkGraph(): візуалізує структуру мережі у вигляді графу.

Фрагмент коду:

```
function createNeuralNetwork(n) {
  let weights = [];
  for (let i = 0; i < n; i++) {
    weights[i] = [];
    for (let j = 0; j < n; j++) {
      weights[i][j] = i === j ? 0 : 0; // ініціалізація з нуля, без автозв'язків
    }
  }
  return weights;
}
```

Сценарій 2: Навчання мережі

Призначення функцій:

Цей сценарій дозволяє користувачеві ввести бінарні образи та зберегти їх у вигляді матриці ваг за класичним правилом Хопфілда.

Основні функції:

- trainNetwork(patterns): реалізує правило Хопфілда для навчання;
- binarizePattern(input): перетворює введені образи у формат +1/-1;
- updateWeights(patterns): оновлює ваги мережі за всіма образами.

Фрагмент коду:

```
function trainNetwork(patterns) {
  let n = patterns[0].length;
  let weights = Array(n).fill(0).map(() => Array(n).fill(0));
  for (let p of patterns) {
    for (let i = 0; i < n; i++) {
      for (let j = 0; j < n; j++) {
        if (i !== j) weights[i][j] += p[i] * p[j];
      }
    }
  }
  return weights;
}
```

Сценарій 3: Згадування образу з шумом

Призначення функцій

Користувач вводить спотворений образ, після чого тренажер демонструє, як мережа поступово переходить до запам’ятаного стану.

Основні функції:

- recall(inputPattern): запускає ітераційне згадування;
- updateState(state, weights): реалізує асинхронне оновлення станів;
- visualizeRecallProcess(): відображає динаміку змін стану.

Фрагмент коду:

```
function recall(state, weights) {
  let newState = [...state];
  for (let i = 0; i < state.length; i++) {
    let sum = 0;
    for (let j = 0; j < state.length; j++) {
      sum += weights[i][j] * newState[j];
    }
    newState[i] = sum >= 0 ? 1 : -1;
  }
  return newState;
```

```
}
```

Сценарій 4: Дослідження енергетичного ландшафту

Призначення функцій:

Дослідницький модуль візуалізує значення енергетичної функції мережі на кожному кроці.

Основні функції:

- calculateEnergy(state, weights): обчислює енергію мережі;
- logEnergyTrajectory(): будує графік енергетичного ландшафту.

Фрагмент коду:

```
function calculateEnergy(state, weights) {
  let energy = 0;
  for (let i = 0; i < state.length; i++) {
    for (let j = 0; j < state.length; j++) {
      energy -= 0.5 * weights[i][j] * state[i] * state[j];
    }
  }
  return energy;
}
```

Сценарій 5: Виявлення обмежень

Призначення функцій: Користувач перевіряє, як збільшується кількість помилкових атракторів при перенавчанні мережі.

Основні функції:

- evaluateRecallAccuracy(): обчислює відсоток правильних згадувань;
- detectSpuriousAttractors(): виявляє нестабільні чи хибні стани.

Фрагмент коду:

```
function detectSpuriousAttractors(patterns, weights) {
  let attractors = new Set();
  for (let p of patterns) {
    let recalled = recall(p, weights).join("");
    attractors.add(recalled);
  }
  return [...attractors];
}
```

Сценарій 6: Тестування користувача

Призначення функцій:

Модуль тестування надає користувачеві завдання, перевіряє результат і надає зворотний зв’язок.

Основні функції:

- generateTestTask(): генерує тестові дані;
- validateUserAnswer(): перевіряє результат згадування;
- provideFeedback(): надає оцінку дій користувача.

Фрагмент коду:

```
function validateUserAnswer(userInput, expectedOutput) {
  return JSON.stringify(userInput) === JSON.stringify(expectedOutput);
}
```

Усі модулі з’єднуються через головний інтерфейс, який реалізовано у вигляді HTML-сторінки, що взаємодіє з Moodle через SCORM API (APIWrapper.js) або модулі типу LTI. Такий підхід забезпечує повну інтеграцію функціонального контенту тренажера в навчальне середовище ЗВО та дозволяє відстежувати результати взаємодії користувача через журнал Moodle.

3.4 Результати тестування

Методика тестування спрямована на перевірку функціональної придатності електронного навчального засобу у форматі SCORM 1.2 у контексті інтеграції з платформою Moodle. Вона охоплює як технічні аспекти взаємодії з SCORM API, так і функціональність інтерфейсних компонентів, реалізованих за допомогою HTML та JavaScript.

Головною метою тестування є верифікація працездатності кожного зі сценаріїв тренажера, коректності відображення результатів взаємодії користувача в інтерфейсі та правильності збереження даних навчальної сесії в системі управління навчанням Moodle через SCORM API.

Тестове середовище:

- Система управління навчанням: Moodle 4.x з увімкненим SCORM-модулем.

- Операційне середовище: Windows 10.
- Браузери: Google Chrome 125, Mozilla Firefox 118.
- SCORM-пакет: hopfield_scorm_package.zip, сформований відповідно до специфікації SCORM 1.2, з використанням файлів index.html, APIWrapper.js і imsmanifest.xml.
- Завантаження в Moodle: здійснено через функціонал додавання нового елемента курсу типу *SCORM пакет*.

Структура тестування:

1) Функціональне тестування сценаріїв тренажера

Для кожного з шести навчальних сценаріїв (створення мережі, навчання, згадування образу, дослідження енергії, перевірка меж пам’яті, тестування користувача) передбачено послідовне натискання відповідної кнопки в інтерфейсі користувача з подальшою перевіркою:

- наявності повідомлення про виконання дії в інформаційному блоці (div#log);
- відсутності помилок JavaScript у консолі браузера;
- стабільної роботи інтерфейсу при повторному виклику функцій.

2) Тестування SCORM API-взаємодії

Передбачено перевірку коректності ініціалізації SCORM-сесії, передачі даних про результати та завершення сесії через виклики відповідних функцій SCORM API:

- doInitialize() — ініціалізація сесії;
- doSetValue('cmi.core.lesson_status', 'passed') — фіксація статусу проходження;
- doSetValue('cmi.core.score.raw', '100') — передача результату у балах;
- doCommit() — передача даних до LMS;
- doTerminate() — завершення сесії.

Тестування виконувалося за допомогою стандартного SCORM-звіту в Moodle (вкладка «Результати» → «Подробиці спроби»), що дає змогу перевірити, чи були дані про сесію (статус, оцінка) коректно збережені в базі даних Moodle.

3) Критерії успішності проходження тестів наведено в таблиці 3.4

Таблиця 3.4 – Критерії успішного проходження тестів

№	Критерій	Метод перевірки	Очікуваний результат
1	Активність кожної кнопки сценарію	Натискання кнопки, вивід у `#log`	Повідомлення про виконання дії
2	Відсутність JavaScript-помилок	Перегляд консолі розробника	Відсутність повідомлень про помилки
3	Ініціалізація SCORM-сесії	Перевірка журналу SCORM в Moodle	SCORM API успішно ініціалізовано
4	Запис результату (`score.raw = 100`)	Вивід у звітах Moodle	Результат успішно зафіксовано
5	Фіксація статусу сесії (`lesson_status`)	Звіти в Moodle	Статус «пройдено»
6	Завершення сесії	Перевірка виводу `Terminate`	Завершення підтверджено системою

Джерело: запропоновано автором

Інструменти, використані під час тестування

- SCORM-дебаггер у Moodle для моніторингу передачі змінних SCORM.
- JavaScript-консоль браузера (Chrome DevTools, Firefox Developer Tools).
- Форма результатів спроби у Moodle для перегляду детальної SCORM-аналітики.

Оцінка результатів та подальші рекомендації

Проведене тестування SCORM-сумісного віртуального тренажера «Мережа Хопфілда», реалізованого з використанням технологій HTML та JavaScript, дозволило підтвердити його працездатність у середовищі Moodle. Основною метою тестування була перевірка функціонування шести передбачених сценаріїв взаємодії з користувачем, а також коректності ініціалізації та завершення SCORM-сесії, збереження результатів та відповідності базовим вимогам до електронного навчального контенту у форматі SCORM 1.2.

У ході тестування використовувалася тестова інсталяція Moodle версії 4.x із підтримкою SCORM, а сам пакет було завантажено як навчальну діяльність типу SCORM. Кожен із шести сценаріїв – створення мережі, навчання, згадування

образу, дослідження енергетичного ландшафту, перевірка меж пам’яті, а також тестування користувача – активував відповідну функцію, яка викликала повідомлення у вікні логів. Це свідчить про правильну роботу функцій JavaScript, відповідальних за логіку тренажера.

Особливу увагу приділено перевірці роботи SCORM API через реалізацію у файлі APIWrapper.js. Було підтверджено коректне викликання функцій doInitialize(), doSetValue(), doCommit() та doTerminate() при відповідних діях користувача. Усі дані про результат (зокрема, передання оцінки 100 балів та статусу «passed») були успішно зафіксовані в системі керування навчанням Moodle, що підтверджує ефективність взаємодії з SCORM API навіть за умови використання спрощеного API-інтерфейсу.

Функціональність SCORM-механізму була перевірена також шляхом аналізу звітів Moodle, де відображено правильну фіксацію результатів навчання – зокрема, статус проходження та отриману оцінку. Таким чином, тренажер демонструє базову відповідність вимогам SCORM 1.2, а його структура, описана у файлі imsmanifest.xml, успішно сприймається Moodle як SCO-ресурс.

Водночас у процесі тестування було виявлено низку потенційних напрямів для удосконалення. Так, нинішня реалізація не передбачає графічного відображення структури мережі Хопфілда на полотні canvas, не реалізовано інтерфейс введення користувацьких образів, відсутній механізм перевірки правильності згадування образу, а SCORM API має форму заглушки, без повноцінної взаємодії з системою управління навчанням (зокрема, без використання suspend_data). З огляду на це, доцільним є розширення функціоналу тренажера, зокрема шляхом впровадження механізмів візуалізації, збереження стану користувача, перевірки знань та використання повного API SCORM 1.2 або SCORM 2004.

У підсумку, створений прототип SCORM-сумісного тренажера «Мережа Хопфілда» підтвердив базову функціональну придатність для використання в електронному навчанні, а результати тестування свідчать про відповідність

початковим вимогам щодо інтерактивності, структурованості та сумісності з платформою Moodle.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи було розроблено віртуальний тренажер для моделювання роботи мережі Хопфілда з дисципліни «Інтелектуальні інформаційні системи». При цьому було розв’язано такі задачі:

1. Досліджено теоретичні основи побудови та функціонування рекурентних нейронних мереж, структуру та принципи роботи мережі Хопфілда як моделі асоціативної пам'яті.
2. Проведено аналіз особливостей навчання рекурентних нейронних мереж та механізмів стабілізації їх динаміки, включаючи алгоритми енергетичної мінімізації.
3. Здійснено обґрунтування вибору програмного середовища та технологій для реалізації тренажера. При цьому враховувалися вимоги сумісності з існуючою платформою дистанційного навчання ЗВО, що є критично важливим для успішної інтеграції.
4. Сформовано технічне завдання на створення програмного продукту, яке включає детальні функціональні та нефункціональні вимоги.
5. Виконано реалізацію віртуального тренажера відповідно до поставлених задач.
6. Розроблено механізми, що забезпечують можливість інтеграції тренажера в інфраструктуру електронного навчання.
7. Здійснено перевірку працездатності розробленого віртуального тренажера, підтвердивши його функціональність та готовність до використання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Білецький, В. В., Войтович, І. С., Апшай, Ф. В., Теліш, І. С. (2023). Інформаційно-комунікаційні технології в умовах змішаного навчання. *Наукові записки. Серія: Педагогічні науки*, (208), 91-97.

2. Balaban, I., Rienties, B., & Winne, P. H. (2023). Information Communication Technology (ICT) and Education. *Applied Sciences*, 13(22), 12318.

3. Zuhri, R., Arifin, S., & Sari, I. (2023). Information communication technologies education in elementary school: a systematic literature review. *Journal of Education and Learning (EduLearn)*, 17(2), 273-282.

4. Калиновська, О. С., Коляда, М. Г., Полонська, В. В. (2024). Віртуальні тренажери як засіб розвитку професійних компетентностей майбутніх фахівців. *Інформаційні технології і засоби навчання*, *99*(1), 22-34.

5. Pan, Z., Ma, Y., & Chen, G. (2023). Virtual Reality (VR) and Augmented Reality (AR) in Education: A Comprehensive Review. *Journal of Educational Technology & Society*, *26*(3), 112-125.

6. Wang, L., Zhang, X., & Li, Y. (2024). Developing a Virtual Reality Training System for Enhancing Practical Skills in Engineering Education. *International Journal of Engineering Education*, *40*(2), 150-162.

7. Chakraborty, A., & Gupta, A. (2024). Leveraging Moodle for Experiential Learning: Integrating Virtual Reality and Interactive Simulations in Higher Education. *Journal of Applied Learning Technology*, *17*(1), 45-58.

8. Білан, С. В., Кудін, В. О., & Ткачук, В. В. (2022). Використання Moodle для організації практичних занять з елементами віртуальних симуляцій. *Інформаційні технології в освіті*, *(34)*, 108-117.

9. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

10. Романюк, В. А., & Стовпчик, С. Ю. (2023). Рекурентні нейронні мережі: архітектури та застосування. *Проблеми програмованої освіти*, *(1)*, 98-105.

11. Koliushko, N. V., Vovk, T. V., & Shumeiko, I. B. (2023). Моделювання функціонування нейронної мережі Хопфілда в задачах розпізнавання образів.

Вісник Харківського національного університету імені В. Н. Каразіна. Серія: Математичне моделювання. Інформаційні технології. Автоматизовані системи управління, *(51)*, 65-74.

12. Haykin, S. (2009). *Neural Networks and Learning Machines* (3rd ed.). Pearson.

13. Попов, О. В., & Мельник, С. В. (2024). Програмне моделювання та візуалізація динаміки нейронної мережі Хопфілда. *Системні дослідження та інформаційні технології*, *(1)*, 87-95.

14. Master Article on Scientific Computation in JavaScript. Scribbler. (2024).

15. Plot Example. Math.js. – Plot Example. Math.js.

16. Using a separate math library for handling p5.js's math operations. GitHub. (2023). https://github.com/processing/p5.js/issues/6527

17. Hossain, A. A., & Rashid, M. H. (2020). Implementing SCORM 1.2 for E-learning Content in Moodle Platform. *International Journal of Computer Science and Network Security (IJCSNS)*, *20*(4), 164-169.

18. Залізняк, Н. В., & Горошко, Т. І. (2021). Аналіз стандартів електронного навчання: від SCORM до xAPI. *Науковий вісник Ужгородського національного університету. Серія: Педагогіка, соціальна робота*, *(2)*, 126-130.

19. Adelphi, R. M., & Rahman, M. M. (2020). Developing an e-learning content management system using SCORM and Learning Management System (LMS). *Journal of Physics: Conference Series*, *1477*(4), 042007.

20. Swapnil, A. S. (2024)Using both Bootstrap and Tailwind CSS in a single React project. - https://dev.to/swapnilahmedshishir/using-both-bootstrap-and-tailwind-css-in-a-single-react-project-2k49

21. Chambers, A. (2024). *Node.js Design Patterns: Master Best Practices for Building Scalable and Robust Node.js Applications*. Packt Publishing.

ДОДАТКИ

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ

На розробку віртуального тренажеру «Мережа Хопфілда» для вибіркового освітнього компоненту «Інтелектуальні інформаційні системи»

Погоджено:

Кандидат технічних наук, доцент кафедри кібернетики та інформатики

Ігор ШЕЛЕХОВ

Суми 2025

1. Призначення й мета створення

1.1 Призначення

Віртуальний тренажер «Мережа Хопфілда» призначений для підтримки інтерактивного навчального процесу в курсах з нейромереж, штучного інтелекту та когнітивного моделювання, що викладаються у вищих навчальних закладах. Тренажер реалізується у форматі SCORM 1.2 та інтегрується з системою управління навчанням Moodle. Він забезпечує студентам можливість наочно вивчити принципи функціонування рекурентної нейронної мережі Хопфілда, здійснити практичне моделювання процесів навчання, згадування, аналізу енергетичного ландшафту та меж пам’яті.

1.2 Мета створення

Метою створення тренажера є розробка електронного навчального засобу, який у формі симульованого середовища дозволяє:

- Наглядно демонструвати архітектуру та динаміку роботи мережі Хопфілда;
- Формувати у студентів розуміння принципів асоціативної пам’яті;
- Забезпечити взаємодію з користувачем через шість навчальних сценаріїв;
- Автоматизувати збирання даних про результати взаємодії з тренажером через SCORM API;
- Підвищити рівень інтерактивності та персоніфікації навчального процесу;
- Реалізувати діагностику навчальних досягнень з можливістю подальшої аналітики в LMS.

1.3 Цільова аудиторія

Цільовою аудиторією віртуального тренажера «Мережа Хопфілда» є здобувачі вищої освіти першого (бакалаврського) та другого (магістерського)

рівнів, які вивчають дисципліни, пов’язані з нейромережевими технологіями, штучним інтелектом, когнітивними системами та математичним моделюванням.

Зокрема, тренажер орієнтований на:

- Студентів спеціальностей 122 «Комп’ютерні науки», 124 «Системний аналіз», 126 «Інформаційні системи та технології», 113 «Прикладна математика», які вивчають архітектури нейронних мереж у рамках навчальних курсів;
- Аспірантів та викладачів для використання тренажера як ілюстративного й експериментального засобу в рамках семінарських та лабораторних занять;
- Самостійних користувачів (інженерів, IT-спеціалістів, дослідників), які прагнуть засвоїти базові поняття та принципи функціонування асоціативних пам’ятей на основі мереж Хопфілда;
- Розробників освітніх програм, які впроваджують елементи змішаного навчання та електронної дидактики у курси з штучного інтелекту.

2. Вимоги до віртуального тренажера

2.1 Вимоги до віртуального тренажера в цілому

2.1.1 Вимоги до структури й функціонування віртуального тренажера

Віртуальний тренажер має реалізовувати інтерактивну модель рекурентної нейронної мережі Хопфілда з підтримкою шести навчальних сценаріїв, кожен з яких відповідає певному аспекту вивчення її роботи. Тренажер повинен мати модульну структуру, забезпечуючи адаптацію до різних рівнів складності та підтримку самостійного, групового й аудиторного навчання. Функціонування тренажера має відбуватись у середовищі LMS Moodle з використанням стандарту SCORM 1.2.

Основними вимогами до функціонування є:

- забезпечення відтворюваності всіх сценаріїв користувацької взаємодії;

- підтримка візуалізації процесів (наприклад, зміна станів мережі, граф енергії, атрактори);
- обробка введених користувачем даних (вектори образів, кількість нейронів тощо);
- фіксація результатів взаємодії у системі управління навчанням через SCORM API.

2.1.2 Вимоги до персоналу

Для ефективного використання віртуального тренажера персонал повинен відповідати таким кваліфікаційним вимогам:

- Користувачі (студенти):
  - володіння базовими навичками роботи з ПК;
  - розуміння фундаментальних понять лінійної алгебри, математичної логіки та основ інформатики.
- Викладачі/тренери:
  - знання теоретичних засад нейронних мереж, зокрема мережі Хопфілда;
  - досвід роботи з Moodle або іншими LMS;
  - вміння інтерпретувати результати симуляцій та коригувати навчальний процес.
- Адміністратори системи:
  - навички інтеграції SCORM-пакетів до середовища Moodle;
  - здатність моніторити працездатність модуля та забезпечувати його оновлення.

2.1.3 Вимоги до збереження віртуального тренажера

Для забезпечення довготривалого та безпечного збереження тренажера мають виконуватись такі вимоги:

- Зберігання SCORM-пакету у централізованому сховищі освітньої платформи (репозиторії Moodle).
- Регулярне резервне копіювання (не рідше одного разу на тиждень).
- Захист SCORM-ресурсів від несанкціонованих змін та втрати даних.
- Контроль сумісності з оновленнями LMS і браузерів.

2.1.4 Вимоги до розмежування доступу

Розмежування прав доступу має здійснюватися на основі ролей у системі Moodle:

- Студент має право запускати тренажер, взаємодіяти зі сценаріями, переглядати результат власного тестування.
- Викладач отримує доступ до звітності, журналів активності студентів, перегляду результатів тестування.
- Адміністратор відповідає за розгортання SCORM-пакету, його оновлення, конфігурацію та підтримку сумісності.

2.2 Структура віртуального тренажера

2.2.1 Навігація

Ліве навігаційне меню з доступом до шести сценаріїв:

- Ознайомлення з архітектурою
- Навчання мережі
- Згадування образу
- Енергетичний ландшафт
- Межі пам’яті
- Тестування користувача

2.2.3 Наповнення віртуального тренажера (контент)

- Інтерактивна панель управління: кнопки запуску сценаріїв.
- Візуалізація: canvas-елемент для графічного відображення структури та динаміки мережі.
- Інформаційна панель: текстова панель з логами дій користувача.
- Тестовий модуль: форма вводу відповідей, автоматична оцінка, передача результатів у SCORM-змінні.

2.3 Вимоги до функціонування системи

2.3.1 Потреби користувача

- Доступ до тренажера через браузер без встановлення додаткового ПЗ.
- Можливість самостійної взаємодії з системою через інтуїтивний інтерфейс.
- Збереження результатів виконання вправ у середовищі Moodle.

2.3.2 Функціональні вимоги

- Ініціалізація SCORM-сеансу.
- Виконання симуляцій відповідно до навчальних сценаріїв.
- Автоматичний запис прогресу та оцінювання.
- Відображення повідомлень користувачеві та логів.

2.3.3 Системні вимоги

- Сумісність з Moodle 3.9 або вище.
- Підтримка стандартів SCORM 1.2.
- Сумісність з браузерами Google Chrome, Mozilla Firefox, Safari, Edge (останні версії).
- Мінімальні апаратні вимоги клієнта:

- RAM: 2 ГБ;
- CPU: 1,5 ГГц;
- Розширення екрану: не менше 1024×768.

2.4 Вимоги до видів забезпечення

2.4.1 Вимоги до інформаційного забезпечення

- Методичні матеріали щодо роботи з тренажером.
- Інструкції користувача для студентів та викладачів.
- Теоретичні довідки щодо кожного сценарію.

2.4.2 Вимоги до лінгвістичного забезпечення

- Повна україномовна локалізація інтерфейсу.
- Дотримання норм наукового та академічного стилю.
- Можливість підготовки альтернативної англомовної версії.

2.4.3 Вимоги до програмного забезпечення

- Основні технології: HTML5, CSS3, JavaScript (ECMAScript 6+).
- SCORM-сумісність реалізується через бібліотеку APIWrapper.js.
- Упакування SCORM-пакету із застосуванням imsmanifest.xml.

3 Склад і зміст робіт зі створення віртуального тренажера

Докладний опис етапів роботи зі створення віртуального тренажера наведено в таблиці А.1.

Таблиця А.1 – Етапи створення віртуального тренажера

№	Склад і зміст робіт	Строк розробки (дні)
1	Формулювання цілей і призначення тренажера	2
2	Визначення цільової аудиторії та її потреб	1
3	Аналіз аналогічних інструментів та дослідження технологічного контексту	2
4	Збір функціональних, освітніх і технічних вимог	3
5	Розробка структури тренажера (сценарії, модулі, навігація)	2
6	Розробка технічного завдання	3
7	Проєктування інтерфейсу користувача (UI/UX макети)	3
8	Узгодження дизайну й навігаційної структури	1
9	Розробка логіки роботи мережі Хопфілда у вигляді окремих сценаріїв (6 модулів)	7
10	Програмна реалізація HTML+JS функціоналу візуалізації	4
11	Реалізація SCORM-фреймворку та підключення APIWrapper.js	2
12	Розробка об'єкта керування навчальним процесом (SCORM API)	2
13	Розробка модулю обробки тестових дій користувача	3
14	Побудова енергетичного графа та візуалізацій динаміки	3
15	Тестування кожного сценарію (модульне тестування)	3
16	Створення manifest.xml та упакування SCORM-пакету	1
17	Тестування сумісності з Moodle (імпорт, логування, SCORM-змінні)	2
18	Виправлення виявлених помилок, перевірка інтеграції	2
19	Написання методики тестування та інструкцій користувача	2
20	Підготовка методичних матеріалів для викладача	2
21	Навчання персоналу та тестове впровадження	2
22	Остаточне розгортання тренажера в LMS Moodle	1
23	Збір зворотного зв’язку від користувачів	2
24	Внесення змін за результатами апробації	2
25	Формування звіту про виконання проєкту	1
	Загальна тривалість робіт	60 днів

Джерело: запропоновано автором

4. Вимоги до складу й змісту робіт із введення вебзастосунку в експлуатацію

1. Підготовчий етап

Мета: забезпечити технічну готовність середовища функціонування вебзастосунку.

Склад робіт:

- Встановлення платформи управління навчанням (LMS), сумісної з SCORM (наприклад, Moodle).
- Перевірка конфігурацій вебсерверу (Apache/Nginx), PHP та бази даних (MySQL/PostgreSQL).
- Розгортання тестового середовища для апробації пакета.

2. Тестове завантаження SCORM-пакету

Мета: перевірка працездатності електронного навчального модуля.

Склад робіт:

- Імпорт SCORM-пакету до LMS.
- Перевірка автоматичної реєстрації навчальних змінних через SCORM API (наприклад, cmi.core.lesson_status, cmi.core.score.raw).
- Тестування інтерфейсу користувача, коректності виводу візуалізацій, роботи сценаріїв.

3. Апробація тренажера з обмеженою групою користувачів

Мета: зібрати емпіричні дані про ефективність і зручність використання.

Склад робіт:

- Надання доступу фокус-групі студентів або викладачів.
- Спостереження за проходженням навчальних сценаріїв.
- Опитування користувачів щодо зручності інтерфейсу, функціоналу й змістовного наповнення.

4. Аналіз результатів апробації та доопрацювання

Мета: забезпечити відповідність застосунку вимогам педагогічної, технічної та ергономічної якості.

Склад робіт:

- Аналіз логів взаємодії користувачів із системою.
- Виправлення виявлених помилок у роботі сценаріїв, логіці оновлення станів мережі.
- Внесення пропозицій щодо удосконалення за відгуками.

5. Підготовка супровідної документації

Мета: забезпечити інструктивне та методичне забезпечення застосунку.

Склад робіт:

- Розробка інструкції користувача (з прикладами вводу/виводу, інтерпретацією результатів).
- Підготовка інструкції адміністратора LMS щодо встановлення та оновлення SCORM-пакету.
- Створення методичних рекомендацій для викладачів.

6. Офіційне введення в експлуатацію

Мета: забезпечити повноцінну доступність і підтримку тренажера для всіх користувачів.

Склад робіт:

- Перенесення SCORM-пакету з тестового до робочого середовища.
- Реєстрація тренажера у структурі електронного навчального курсу.
- Призначення відповідальних осіб за технічне супроводження (адміністратор Moodle, технічний фахівець).
- Розміщення інструктивних матеріалів у відповідних розділах LMS.

7. Постексплуатаційний супровід

Мета: підтримка працездатності системи та її періодичне оновлення.

Склад робіт:

- Регулярне оновлення SCORM-пакету за потреби.
- Моніторинг працездатності, виявлення технічних або методичних збоїв.
- Збір фідбеку від нових груп користувачів.
- Актуалізація контенту відповідно до зміни навчальних програм.

ДОДАТОК Б
ПЛАНУВАННЯ РОБІТ

На рисунку Б.1 зображено Діаграму Ганта, у вигляді календарю.

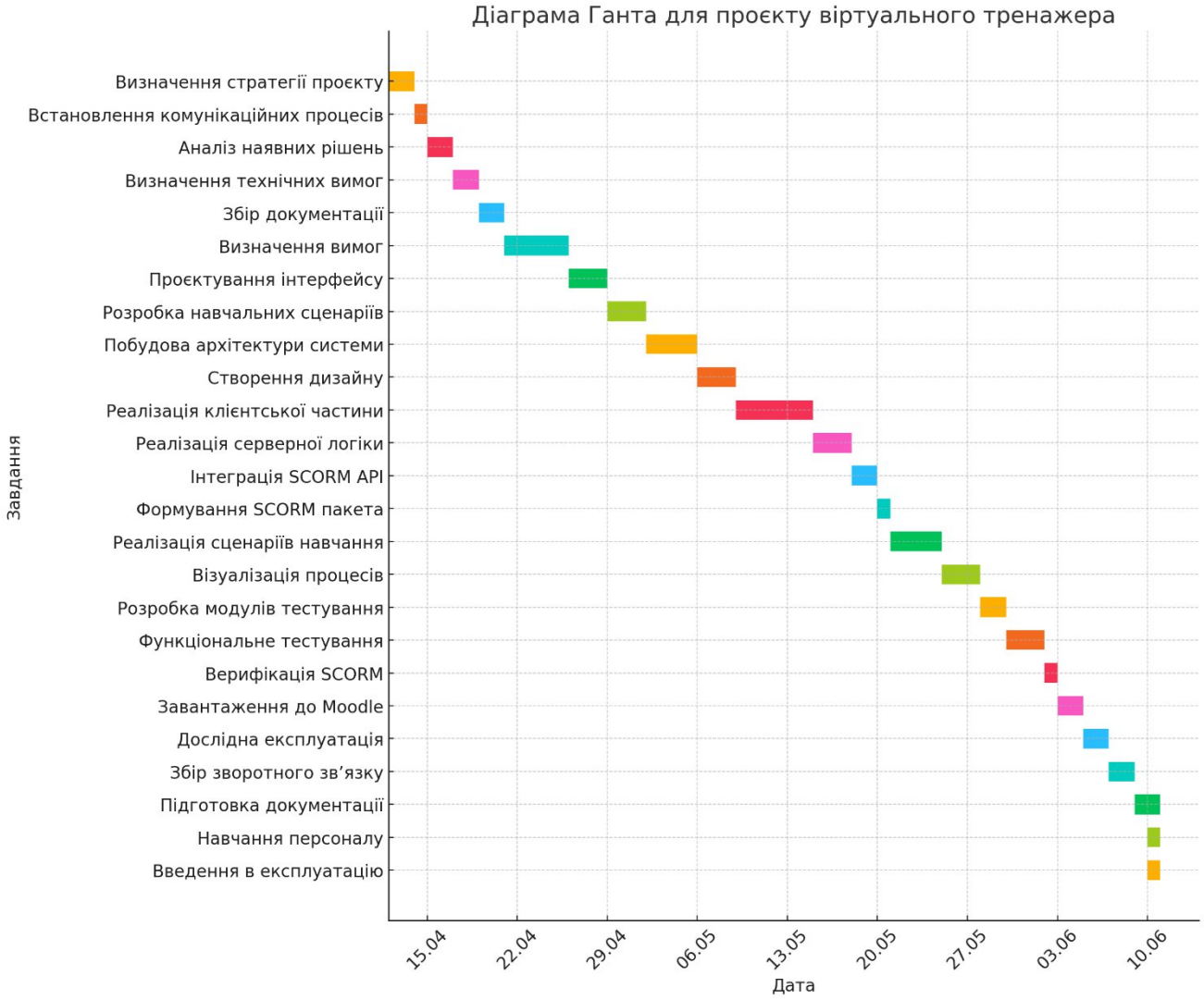


Рисунок Б.1 – Діаграма Ганта

Джерело: запропоновано автором

Реєстр ризиків проекту подао в таблиці Б.1.

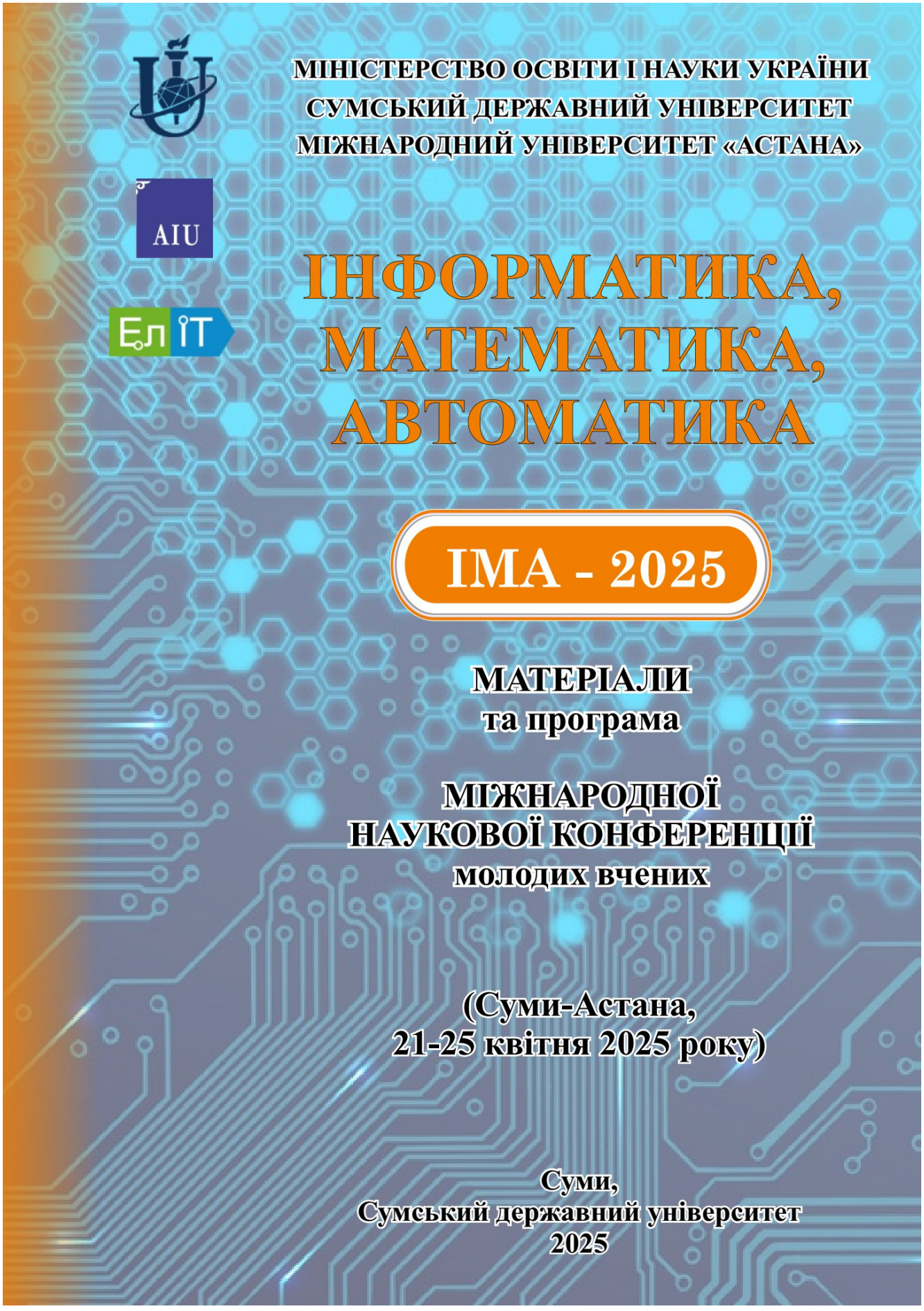
Таблиця Б.1

Реєстр ризиків

Опис ризику	Імовірність виникнення	Вплив на проєкт	Стратегії управління	Відповідальні особи
Затримка в розробці програмних модулів	Середня	Високий	Додаткове планування часу, контроль проміжних результатів, оперативне залучення додаткових розробників	Керівник команди розробників
Недостатня якість SCORM-сумісності	Низька	Високий	Попереднє тестування SCORM API, валідація через SCORM Cloud	Розробник SCORM-модуля
Помилки у візуалізації або взаємодії з користувачем	Середня	Середній	Проведення функціонального тестування, юзабіліті-тестування	UI/UX дизайнер, QA-фахівець
Неправильне визначення вимог або зміна вимог під час реалізації	Середня	Високий	Регулярні консультації з методистами, фіксація вимог у технічному завданні	Аналітик, технічний керівник
Труднощі із завантаженням до Moodle або сумісністю платформи	Низька	Середній	Тестування на демо-версії Moodle, перевірка конфігурації SCORM 1.2 / 2004	Адміністратор LMS
Відсутність або затримка зворотного зв’язку від цільової аудиторії	Середня	Низький	Розсилка запитів на зворотний зв’язок, анкетування, інтеграція автоматичного збирання фідбеку	Менеджер з навчального контенту
Відсутність кваліфі-кованого персоналу для супроводу після запуску	Низька	Середній	Навчання персоналу перед запуском, створення детальної документації	Керівник проєкту, технічний письменник
Втрати даних при формуванні або оновленні SCORM пакету	Низька	Високий	Регулярне резервне копіювання, версіонування вихідних файлів	DevOps-інженер
Невідповідність лінгвістичного оформлення стандартам навчального середовища	Середня	Середній	Залучення філолога, перевірка всього контенту перед публікацією	Методист, редактор
Відсутність часу на повноцінне тестування всіх сценаріїв	Середня	Високий	Пріоритезація тест-кейсів, автоматизація тестування основних функцій	QA-інженер, технічний керівник

Джерело: запропоновано автором

ДОДАТОК В
КОПІЇ ПУБЛІКАЦІЙ


МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
МІЖНАРОДНИЙ УНІВЕРСИТЕТ «АСТАНА»
AIU
ЕлІТ
ІНФОРМАТИКА, МАТЕМАТИКА, АВТОМАТИКА
ІМА - 2025
МАТЕРІАЛИ
та програма
МІЖНАРОДНОЇ НАУКОВОЇ КОНФЕРЕНЦІЇ
молодих вчених
(Суми-Астана,
21-25 квітня 2025 року)
Суми,
Сумський державний університет
2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
МІЖНАРОДНИЙ УНІВЕРСИТЕТ «АСТАНА»

ІНФОРМАТИКА, МАТЕМАТИКА, АВТОМАТИКА

ІМА :: 2025

МАТЕРІАЛИ
та програма

МІЖНАРОДНОЇ НАУКОВОЇ КОНФЕРЕНЦІЇ
молодих учених

(Суми – Астана, 21–25 квітня 2025 року)

Суми
Сумський державний університет
2025

СЕКЦІЯ № 1 «КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА»

Голова секції – канд. фіз.-мат. наук, доцент Шовкопляс О.А.
Секретар секції – канд. техн. наук Прилепа Д.В.

Початок: 23.04.2025 р., 13-00, онлайн,
https://us02web.zoom.us/j/5085263824?pwd=VHI1RjNKUG9TY3hLeFRTUERtYko0UT09
Ідентифікатор конференції: 508 526 3824
Код доступу: 549027

1. Experimental evaluation of a two-level classification model for identifying binary data blocks

   Author: PhD student Boiko M.V.,
   Supervisor – Associate Professor Viacheslav Moskalenko

2. How to remove files from a Git repository

   Author: Student Yevhenii Lytvynenko,
   Supervisor – Associate Professor Kuzikov Borys

3. Factorial Classification Analysis of UAV Geospatial Data Based on Information-Extremal Intelligent Technology

   Authors: PhD Stud. Mokrenko Andrii,
   student Baburov Dmytro,
   Associate Professor Shelehov Igor

4. Methodology and Tools for Interactive Testing Tasks in Base Recurrent Neural Networks

   Authors: student Yuldashova Kumush,
   student Edimaobong Matthew,
   Associate Professor Shelehov Igor

Methodology and Tools for Interactive Testing Tasks in Base Recurrent Neural Networks

Yuldashova Kumush[1], *IST-2101, Student,*
Edimaobong Matthew[2], *IN-15an*, *Student,*
Shelehov Igor[1,2], *Associate Professor*

[1]Department Informatics and Cybernetics of Sumy National Agrarian University, Sumy, Ukraine
[2]Department of Computer Science of Sumy State University, Sumy, Ukraine

Developing complex interactive testing tasks constitutes a crucial element for the practical and comprehensive evaluation of learners' knowledge in recurrent neural networks, specifically deterministic Hopfield networks and stochastic Boltzmann machines [1]. These theses address the methodological and programmatic aspects of designing, implementing, and automatically generating such tasks, which integrate diverse interactive exercises for a holistic assessment of theoretical understanding and practical skills. The methodology for designing complex interactive testing tasks integrates principles from active learning, problem-based learning, and constructivist approaches. asks are developed based on the SOLO (Structure of Observed Learning Outcome) taxonomy [2], which enables the assessment of learners' levels of understanding – from the ability to identify individual aspects (unistructural level) to the integration of knowledge and its application in new contexts (relational and extended abstract levels). Elements of gamification are incorporated to enhance motivation and engagement. The process of interactivity implementation involves the application of web technologies (HTML, CSS, JavaScript) in conjunction with libraries for dynamic visualization (e.g., D3.js, Plotly.js), mathematical computation (MathJax), and capabilities for creating interactive simulations of network behavior. To ensure a comprehensive evaluation of knowledge, the development of tasks encompassing some categories is considered:

1) Identification and Description Tasks: Interactive exercises requiring learners to identify key architectural elements of Hopfield and Boltzmann networks, explain their operational principles and distinctions, and describe fundamental concepts (energy landscape, temperature, states, weights, attractors, probability distributions).

2) Modeling and Experimentation Tasks: Interactive simulations enable learners to create and configure network parameters (number of neurons, weights, temperature), input data, and observe the dynamics of their behavior in real-time. This includes investigating memory recall processes, learning algorithms, and the influence of various parameters on network stability and functionality.

3) Analysis and Comparison Tasks: Interactive scenarios presenting learners with the task of comparing the behavior of Hopfield networks and Boltzmann machines in similar problems, analyzing their advantages and disadvantages, and determining the optimal network type for specific applications.

4) Problem-Solving Tasks: Interactive exercises simulating practical application scenarios of recurrent neural networks (e.g., simple optimization problems, associative memory tasks), where learners must apply acquired knowledge to achieve a specific goal by adjusting network parameters and analyzing results.

5) Synthesis and Design Tasks: More complex interactive exercises requiring learners to develop network architectures or learning algorithms for solving specific problems with defined performance criteria.

For the automated generation of such complex tasks, a mechanism for combining different interactive exercises and parameterizing their elements is planned. The generator may randomly select a set of tasks from a pre-designed bank or dynamically create new combinations based on defined criteria of complexity and coverage of learning material.

Utilizing modern web technologies in conjunction with the developed approach to automated task generation will facilitate a practical and personalized learning assessment experience in recurrent neural networks. The integration of diverse interactivity types will enable the verification of factual knowledge and the evaluation of learners' ability to analyze, synthesize, and practically apply acquired knowledge. Future research will focus on the development of intelligent task-generation algorithms adapted to the learners' proficiency levels, as well as on the evaluation of the effectiveness of the comprehensive assessment approach.

1. Salem, Fathi M. Recurrent Neural Networks: From Simple to Gated Architectures. Springer, 2022. 130 p.
2. Biggs, John, et al. Teaching for Quality Learning at University 5e. McGraw-Hill Education, 2022. 440p.

12. Automation of Software-Defined Network Testing Using Python

    Author: Ph.D. student Serhii Shestopalov
13. Task Management and Performance Analysis System for Small and Medium-Sized Businesses

    Author: student Dmytro Shelikhov,
    Supervisor – assistant Olha Shutylieva

14. Концепція та функціональні можливості інтерактивного тренажеру для вивчення рекурентних нейронних мереж Хопфілда

    Автори: здобувач Юлдашова Кумуш,
    доцент Шелехов І.В.

15. Класифікаційне прогнозування результатів телемаркетингової компанії

    Автори: здобувач Михайлюк Олександр,
    доцент Шелехов Ігор

16. Вплив аудиту інформаційної безпеки на захищеність підприємств: Ключові ризики та рекомендації

    Автори: здобувач Бабак А.О.,
    доцент Ободяк В.К.,
    асистент Пугач І.О.

17. Огляд засобів linting (static code analyze) у Python

    Автор: здобувач Білоцерківець Б.Р,
    Керівник – асистент Шутилєва О.В.

18. Модифікація нечіткого Bees Algorithm з розподіленням рою частинок на групи

    Автори: здобувач Плутенко О.Ю.,
    старший викладач Шаповалов С.П.

Концепція та функціональні можливості інтерактивного тренажеру для вивчення рекурентних нейронних мереж Хопфілда

Юлдашова Кумуш[1], *ІСТ-2101, здобувач*;
Шелехов Ігор[1,2], *доцент*

[1]Кафедра кібернетики та інформатики Сумського національного аграрного університету, Суми, Україна
[2] Кафедра комп'ютерних наук Сумського державного університету, Суми, Україна

У зв'язку зі стрімким розвитком і розширенням застосування штучних нейронних мереж, вивчення рекурентних нейронних мереж Хопфілда, як парадигми асоціативної пам'яті та методу оптимізації, мають важливе значення при підготовці здобувачів вищої освіти ІТ-спеціальностей першого (бакалаврського) рівня. Здатність таких мереж до запам'ятовування та відтворення образів, а також динамічна еволюція до стабільних станів (атракторів) робить їх корисними у вирішенні комбінаторних задач, розпізнаванні образів та моделюванні когнітивних процесів. Однак розуміння принципів функціонування рекурентних нейронних мереж Хопфілда ускладнюється абстрактністю математичного апарату та складністю динаміки, що відбувається в мережі. Традиційні навчальні матеріали (підручники, візуалізації, симуляції) часто не надають достатньої наочності та інтерактивності для ефективного засвоєння ключових концепцій. Це створює труднощі для здобувачів, що починають вивчення цієї теми, і для дослідників, які прагнуть інтуїтивно зрозуміти складні процеси мереж.

Метою дослідження є розробка інтерактивного тренажеру, що забезпечить візуально-орієнтоване та експериментальне середовище для вивчення рекурентних нейронних мереж Хопфілда. Створення такого інструменту допоможе покращити розуміння принципів роботи мережі, її архітектури, енергетичної функції, механізмів запам'ятовування та відтворення образів, а також поведінки мережі при різних параметрах і вхідних умовах.

Аналіз наявних навчальних ресурсів показує відсутність інтерактивних інструментів, що дозволяють безпосередньо маніпулювати параметрами мережі та спостерігати за результатами

змін в реальному часі. Існуючі програмні симулятори орієнтовані на дослідження і мають складний інтерфейс, що робить їх малопридатними для початкового навчання.

Рекурентні нейронні мережі Хопфілда — це одношарові мережі, в яких кожен нейрон з'єднаний з усіма іншими, за винятком самого себе. Кожен нейрон має біполярний стан (+1 або -1) і оновлює свій стан асинхронно, використовуючи зважену суму станів інших нейронів та функцію активації (звичайно порогову). Ключове поняття в мережах Хопфілда — енергетична функція, яка зменшується під час еволюції мережі до локального мінімуму, що відповідає стабільному стану (атрактору). Атрактори використовуються для запам'ятовування образів, і мережа еволюціонує до найближчого збереженого атрактора, реалізуючи процес згадування. Розуміння архітектури, принципу оновлення станів, поведінки енергетичної функції та концепції атракторів є важливим для вивчення мереж Хопфілда.

Основні функціональні можливості тренажеру:

1) Інтуїтивно зрозумілий графічний інтерфейс для створення, редагування та візуалізації архітектури мережі, з можливістю налаштування кількості нейронів та ваг.

2) Модуль візуалізації енергетичного ландшафту для відображення енергетичної функції мережі в динаміці, що дозволяє спостерігати процес збіжності до атракторів.

3) Інструменти для дослідження динаміки мережі, покрокового моделювання процесу оновлення станів нейронів, відображення поточного стану мережі та енергетичних значень.

4) Редактор вхідних образів для зручного введення і маніпулювання вхідними образами.

5) Модуль візуалізації процесу згадування для демонстрації еволюції стану мережі від вхідного образу до стабільного атрактора.

6) Інструменти для аналізу атракторів, включаючи ідентифікацію та візуалізацію стабільних станів мережі.

Отже, запропонований інтерактивний тренажер дозволяє здобувачам вищої освіти глибше зрозуміти принципи роботи рекурентних нейронних мереж Хопфілда, сприяючи ефективному засвоєнню теоретичних та практичних аспектів через активну взаємодію з віртуальною моделлю та візуалізацію складних динамічних процесів.

ДОДАТОК Г
ЛІСТИНГИ ПРОГРАМ

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Віртуальний тренажер: Мережа Хопфілда</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
      background-color: #f4f4f4;
    }
    header {
      background-color: #003366;
      color: white;
      padding: 1rem;
      text-align: center;
    }
    main {
      display: grid;
      grid-template-columns: 1fr 2fr;
      gap: 1rem;
      padding: 1rem;
    }
    section, aside {
      background-color: white;
```

```
  border-radius: 8px;
  padding: 1rem;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}
h2 {
  color: #003366;
}
button {
  margin: 0.25rem 0;
  padding: 0.5rem 1rem;
  background-color: #0055a5;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}
button:hover {
  background-color: #003f7f;
}
canvas {
  width: 100%;
  border: 1px solid #ccc;
  background-color: #fff;
}
.input-pattern {
  display: grid;
  grid-template-columns: repeat(auto-fit, 20px);
  gap: 2px;
}
.neuron {
```

```
      width: 20px;
      height: 20px;
      text-align: center;
      background-color: #eee;
      border: 1px solid #ccc;
    }
  </style>
</head>
<body>
  <header>
    <h1>Віртуальний тренажер: Мережа Хопфілда</h1>
  </header>
  <main>
    <aside>
      <h2>Кроки</h2>
      <button onclick="createNetwork()">1. Створити мережу</button>
      <button onclick="trainNetwork()">2. Навчити мережу</button>
      <button onclick="recallPattern()">3. Згадування образу</button>
      <button onclick="showEnergyGraph()">4. Енергетичний граф</button>
      <button onclick="testCapacity()">5. Виявлення меж пам’яті</button>
      <button onclick="userTest()">6. Тестування користувача</button>
    </aside>
    <section>
      <h2>Візуалізація</h2>
      <canvas id="networkCanvas" width="500" height="500"></canvas>
      <div id="patternInput" class="input-pattern"></div>
      <div id="log"></div>
    </section>
  </main>
```

```
<script>
  let networkSize = 9;
  let weights = [];
  let patterns = [];
  let currentPattern = [];

  function createNetwork() {
   weights = Array.from({ length: networkSize }, () => Array(networkSize).fill(0));
   document.getElementById('log').innerText = `Мережу створено (${networkSize} нейронів)`;
  }

  function trainNetwork() {
   // Навчання одного образу: приклад [1, -1, 1, -1, 1, -1, 1, -1, 1]
   patterns = [
    [1, -1, 1, -1, 1, -1, 1, -1, 1]
   ];
   weights = Array.from({ length: networkSize }, (_, i) =>
    Array.from({ length: networkSize }, (_, j) => {
     if (i === j) return 0;
     return patterns.reduce((sum, p) => sum + p[i] * p[j], 0);
    })
   );
   document.getElementById('log').innerText = 'Мережу навчено за методом Хопфілда';
  }

  function recallPattern() {
   currentPattern = [...patterns[0]];
```

```
        currentPattern[2] *= -1; // Додаємо шум
        let prevPattern = [];
        let steps = 0;

        while (!arraysEqual(currentPattern, prevPattern) && steps < 10) {
          prevPattern = [...currentPattern];
          for (let i = 0; i < networkSize; i++) {
            let sum = 0;
            for (let j = 0; j < networkSize; j++) {
              sum += weights[i][j] * currentPattern[j];
            }
            currentPattern[i] = sum >= 0 ? 1 : -1;
          }
          steps++;
        }
        document.getElementById('log').innerText = `Згадано образ за ${steps} кроків`;
      }

      function showEnergyGraph() {
        // Простий приклад виводу енергії
        let energy = -0.5 * currentPattern.reduce((E, vi, i) =>
          E + currentPattern.reduce((sum, vj, j) => sum + weights[i][j] * vi * vj, 0), 0);
        document.getElementById('log').innerText = `Енергія поточного стану: ${energy.toFixed(2)}`;
      }

      function testCapacity() {
        patterns = [];
        for (let i = 0; i < 6; i++) {
```

```
        let pattern = Array.from({ length: networkSize }, () => Math.random() > 0.5 ? 1 : -1);
        patterns.push(pattern);
      }
      weights = Array.from({ length: networkSize }, (_, i) =>
        Array.from({ length: networkSize }, (_, j) => {
          if (i === j) return 0;
          return patterns.reduce((sum, p) => sum + p[i] * p[j], 0);
        })
      );
      document.getElementById('log').innerText = 'Мережу навчено 6 образами. Перевірте якість згадування.';
    }

    function userTest() {
      let testPattern = [...patterns[0]];
      testPattern[1] *= -1;
      testPattern[3] *= -1;
      currentPattern = [...testPattern];
      let prevPattern = [];
      let steps = 0;
      while (!arraysEqual(currentPattern, prevPattern) && steps < 10) {
        prevPattern = [...currentPattern];
        for (let i = 0; i < networkSize; i++) {
          let sum = 0;
          for (let j = 0; j < networkSize; j++) {
            sum += weights[i][j] * currentPattern[j];
          }
          currentPattern[i] = sum >= 0 ? 1 : -1;
        }
```

```
        steps++;
      }
      document.getElementById('log').innerText = `Результат тесту: згадано образ за ${steps} кроків.`;
    }

    function arraysEqual(a, b) {
      return a.length === b.length && a.every((v, i) => v === b[i]);
    }
  </script>
</body>
</html>
```

SCORM-пакет для Moodle

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>Мережа Хопфілда</title>
  <script src="APIWrapper.js"></script>
</head>
<body>
  <h1>Тренажер: Мережа Хопфілда</h1>
  <button onclick="completeLesson()">Завершити урок</button>
  <script>
    window.onload = function () {
      if (typeof doInitialize === 'function') {
        doInitialize();
      }
```

```
    };
    window.onunload = window.onbeforeunload = function () {
      if (typeof doTerminate === 'function') {
        doTerminate();
      }
    };
    function completeLesson() {
      doSetValue("cmi.core.score.raw", "100");
      doSetValue("cmi.core.lesson_status", "passed");
      doCommit();
      alert("Результат відправлено у LMS.");
    }
  </script>
</body>
</html>

<?xml version="1.0" encoding="UTF-8"?>
<manifest identifier="hopfield_simulator" version="1.0"
  xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"
  xmlns:adlcp="http://www.adlnet.org/xsd/adlcp_v1p3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.imsglobal.org/xsd/imscp_v1p1
                      imscp_v1p1.xsd
                      http://www.adlnet.org/xsd/adlcp_v1p3
                      adlcp_v1p3.xsd">

  <organizations default="org1">
    <organization identifier="org1">
      <item identifier="item1" identifierref="res1">
        <title>Мережа Хопфілда</title>
```

```
      </item>
    </organization>
  </organizations>

  <resources>
    <resource identifier="res1" type="webcontent" adlcp:scormType="sco"
href="index.html">
      <file href="index.html"/>
      <file href="APIWrapper.js"/>
    </resource>
  </resources>
</manifest>
```