Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

магістра
(назва освітньо-кваліфікаційного рівня)

студента	*М'якенького Арсенія Вячеславовича*
	(ПІБ)
академічної групи	*122М-20-1*
	(шифр)
спеціальності	*122 Комп'ютерні науки*
	(код і назва спеціальності)
освітньої програми	*«122 Комп'ютерні науки»*
	(назва освітньої програми)
на тему:	*Обґрунтування добору навчальної методики на підставі аналізу когнітивних моделей у системі дистанційної освіти*

_____*А.В. М'якенький*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
розділів кваліфікаційної роботи				
спеціальний	*Проф. Алексєєв М.О.*			
економічний	*Доц. Касьяненко Л.В.*			
Рецензент				
Нормоконтролер	*Доц. Реута О.В.*			

Дніпро
2022

Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
Завідувач кафедри
Програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 20 ____ Року

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

студенту *122м-20-1* *М'якенькому Арсенію Вячеславовичу*
(група) (прізвище та ініціали)

Тема кваліфікаційної роботи *Обґрунтування добору навчальної методики на підставі аналізу когнітивних моделей у системі дистанційної освіти*

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора НТУ «Дніпровська політехніка» від 10.12.2021 р. № 1036-с

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень – процес формування когнітивних навичок студента під час процесу навчання.

Предмет досліджень – моделі та методи аналізу когнітивних навичок студента та добору навчальних методик.

Мета роботи – підвищення ефективності навчання студента через адаптивне формування навчальних методик на основі аналізу його когнітивної моделі.

Вихідні дані для проведення роботи – теоретичні та експериментальні дослідження, основи моделі оцінки когнітивних навичок студента та методу добору навчальних методик.

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Новизна запропонованих рішень визначається тим, що була розроблений та обґрунтований метод добору навчальних методик, а також модель оцінки когнітивних навичок студента, на підставі аналізу якої робиться висновок щодо необхідності зміни

навчальної методики, що дозволило адаптивно змінювати методику викладання матеріалу під час навчання студента.

Практична цінність результатів полягає в тому, що запропоновані в роботі моделі і методи надають представлення та дозволяють накопичувати когнітивні навички студентів з метою їхнього аналізу, розвитку та можливості до корегування.

4 ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Результати досліджень мають бути подані у вигляді, що дозволяє побачити та оцінити безпосереднє використання запропонованого методу. В результаті роботи повинна бути розроблений метод добору навчальних методик.

5 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	**Строки виконання робіт (початок – кінець**
Аналіз предметної галузі та постановка завдання	12.09.2021-30.09.2021
Аналіз методів дослідження когнітивних процесів	01.10.2021-31.10.2021
Розробка та обґрунтування методу добору навчальних методик для системи дистанційного навчання	01.11.2021-16.12.2021

6 РЕАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ТА ЕФЕКТИВНІСТЬ

Соціальний ефект від реалізації результатів роботи очікується позитивним завдяки розробці методу добору навчальних методик, що дозволяє покращити ефективність навчання та засвоєння студентом матеріалів дисципліни.

Завдання видав _____ _____
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ *М'якенький А.В.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *12.09.2021 р.*

Термін подання кваліфікаційної роботи до ЕК *19.01.2022*

РЕФЕРАТ

Пояснювальна записка: 91 стор., 20 рис., 6 таблиць, 3 додатки, 58 джерел.

Об'єкт дослідження: процес формування когнітивних навичок студента під час процесу навчання.

Предмет дослідження: моделі та методи аналізу когнітивних навичок студента та добору навчальних методик.

Мета роботи: підвищення ефективності навчання студента через адаптивне формування навчальних методик на основі аналізу його когнітивної моделі.

Методи дослідження. Для розв’язання поставлених задач використані методи: класифікації даних, теорії класифікації когнітивних процесів з області когнітивної психології, теорії прийняття рішень, когнітивне моделювання.

Новизна отриманих результатів визначається тим, що була розроблений та обґрунтований метод добору навчальних методик, а також модель оцінки когнітивних навичок студента, на підставі аналізу якої робиться висновок щодо необхідності зміни навчальної методики, що дозволило адаптивно змінювати методику викладання матеріалу під час навчання студента.

Практична цінність результатів полягає в тому, що запропоновані в роботі моделі та методи надають представлення та дозволяють накопичувати когнітивні навички студентів з метою їхнього аналізу, розвитку та можливості до корегування.

Область застосування. Розроблений метод може застосовуватися для добору навчальних методик у системах дистанційного навчання.

Значення роботи та висновки. Розроблений метод дозволяє змінювати методику навчання для окремого студента та покращувати його когнітивні навички для ефективного засвоєння та розуміння навчальних матеріалів, що підтверджується дослідженням, проведеним в даній роботі.

Прогнози щодо розвитку досліджень. Розробка чисельних метрик когнітивних навичок, яка не заснована на експертній оцінці з метою зменшення впливу суб’єктивного погляду експертів.

У розділі «Економіка» проведені розрахунки трудомісткості розробки програмного забезпечення, витрат на створення ПЗ і тривалості його розробки.

Список ключових слів: пізнання, когнітивна архітектура, когнітивне моделювання, когнітивні навички, навчальна методика, випадковий ліс, розумова діяльність, здібність до навчання, викладання.

ABSTRACT

Explanatory note: 91 pages, 20 figures, 6 tables, 3 applications, 58 sources.

Object of research: the process of forming the student's cognitive skills during the learning process.

Subject of research: models and methods of analysis of student's cognitive skills and selection of teaching methods.

Purpose of Master's thesis: improving the effectiveness of student learning through the adaptive formation of educational methods based on the analysis of his cognitive model.

Research methods. Methods Next methods were used to solve set of tasks: data classification, theory of classification of cognitive processes in the field of cognitive psychology, theory of decision making, cognitive modeling.

Originality of research is in developing and substantiating methodology of selection of teaching methods, as well as the model of assessment of cognitive skills of the student, based on the analysis of which the conclusion is made.

Practical value of the results consists of accumulating of cognitive skills of students for their analysis, development and ability to adjust by using models and methods proposed in the work.

Scope of application. The developed methodology can be used for the selection of teaching methods in distance learning systems.

The value of the work and conclusions. The developed methodology allows to change the teaching methods for an individual student and improve his cognitive skills for effective learning and understanding of educational materials, which is confirmed by the research conducted in this work.

Research forecast and development. Development of numerical metrics of cognitive skills, which is not based on expert assessment in order to reduce the impact of subjective views of experts.

In the Economics section we calculated complexity of software development, the cost of creating software and the duration of its development.

Keywords: cognition, cognitive architecture, cognitive modeling, cognitive skills, teaching methods, random forest, mental activity, ability to learn, teaching.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ACT – adaptive control of thought;

4CAPS – cortical capacity-constrained concurrent activation-based production system;

LIDA – learning intelligent distribution agent;

BIC – Bayesian information criterion;

AIC – Akaike information criterion;

JDM – judgment and decision making;

CER – conceptualization experiment reflection;

IBLT – instance-based learning theory;

SDU – situation decision utility;

API – application programming interface;

JSON – JavaScript object notation.

ЗМІСТ

ВСТУП

Процес пізнання належить до розумової діяльності людини, поєднуючи у собі процеси сприйняття, уваги, пам’яті, мислення, розуміння, розв’язання проблем та прийняття рішень та відіграє важливу роль у взаємодії людини та світу, що її оточує. Дослідження цього процесу, а також інших аспектів розумової діяльності людини є темою досліджень когнітивної науки. Когнітивна наука розглядає процес пізнання людини як обчислення інформації та описує підходи та методи до побудови обчислювальних моделей когнітивної діяльності людини.

Дослідження та методи когнітивної науки використовують у різних сферах діяльності, але більшого застосування вони набули у сфері навчання. Когнітивні моделі використовують з метою опису когнітивних процесів під час розумової діяльності студентів при розв’язанні задач та вивченні нового матеріалу. Вони допомагають охарактеризувати знання та навички, які студенти набувають під час різних етапів навчання, та полегшити пояснення та прогнозування результатів студентів. Таким чином, завдання з контролю знань можна розбити на кінцевий набір атрибутів або когнітивних компонентів як когнітивну модель, щоб вона могла зробити результати тестів більш інтерпретованими та значущими.

Аналіз та дослідження результатів когнітивного моделювання дозволяє встановлювати зміни у розумовій та ментальній моделі студента під час навчання та робити висновки щодо ефективності навчальної методики та необхідності її зміни, що зумовлює актуальність обраної теми.

Тому, метою роботи є підвищення ефективності навчання студента через адаптивне формування навчальних методик на основі аналізу його когнітивної моделі.

Для досягнення поставленої мети необхідно вирішити наступні основні завдання:

– дослідити когнітивні процеси, які впливають на процес навчання;

– дослідити методи класифікації когнітивних навичок, що використовуються різними навчальними методиками;

– визначити механізм збору інформації щодо змін когнітивних навичок протягом процесу навчання;

– проаналізувати зібрані відомості та на підставі результатів аналізу обґрунтувати застосування тих чи інших навчальних методик.

Кваліфікаційна робота складається з чотирьох розділів. У першому розділі був проведений аналіз предметної області, а саме описані основні підходи до когнітивного моделювання, наданий розгляд когнітивних архітектур, а також розглянута задача оцінки когнітивних моделей. У другому розділі розглянуті методи дослідження когнітивних процесів у розумовій діяльності людини, а також розглянуті методи класифікації когнітивних навичок людини. Третій розділ містить у собі опис побудованої моделі когнітивних навичок людини та запропонований алгоритм добору навчальних методик з результатами досліджень. У четвертому розділі проведені розрахунки трудомісткості розробки програмного забезпечення, витрат на створення ПЗ і тривалості його розробки.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1. Загальні відомості з предметної галузі

Cognition або пізнання – це процес, який належить до діяльності розуму людини та бере участь у сприйнятті та увазі, пам’яті, навчанні, мисленні та розумінні, розв’язанні проблем та прийнятті рішень. Воно є головною метою досліджень у когнітивній науці – міждисциплінарному напрямі, який поєднує дослідження філософії, психології, комп’ютерних наук, лінгвістики та нейробіології, яка припускає, що пізнання можна розглядати як обчислення інформації [1].

Одним з основних засобів у когнітивній науці для представлення теорій пізнання є когнітивне моделювання. Когнітивне моделювання досліджує сутність пізнання та різноманітних когнітивних функцій людини шляхом визначення відповідних обчислювальних моделей представлень, механізмів та процесів [2].

Когнітивна модель – це модель, яка представляє явні вербальні теорії конкретних когнітивних процесів у вигляді математичного формулювання цих теорій та забезпечує керовану кількісну оцінку когнітивних процесів [3]. Когнітивні моделі можуть прогнозувати те, як багато змінних взаємодіють між собою для формування поведінки об’єкту, що спостерігається під час емпіричних досліджень. Вони допомагають зрозуміти, які взаємопов’язані когнітивні процеси призводять до спостережуваного поведінкового результату.

У процесі когнітивного моделювання використовуються обчислювальні архітектури, основою яких є загальна теорія про архітектуру людського розуму, яку зазвичай називають когнітивною архітектурою [4]. Термін «когнітивна архітектура» запропонував американський науковець у галузі когнітивної психології та штучного інтелекту Аллен Ньюелл, проводячи аналогію з комп’ютерними архітектурами. Когнітивна архітектура – це набір механізмів

людського пізнання, що здійснюються за допомогою процесу здобуття знань людиною, вона заснована на моделюванні того, як саме мозок людини досягає функцій пізнання та розуміння для створення моделі, здатної до розуміння, міркування та розв’язання проблем [5].

Розробка когнітивних архітектур ґрунтується на пам’яті та навчанні. Поєднання цих двох елементів утворює таксономію трьох підходів до побудови когнітивних архітектур (рис. 1.1):

- символьний підхід;
- емерджентний підхід;
- гібридний підхід.

Рис. 1.1. Основні підходи до побудови когнітивних архітектур

Символічний підхід ґрунтується на двох ключових концепціях: представленні та обчисленні. Він описує пізнання, як обчислення, що визначені за допомогою внутрішніх символічних уявлень, які є знаннями системи. Інформація про середовище абстрагується за допомогою сприйняття, представляється за допомогою певної символічної структури даних, оброблюється, а потім використовується для планування та прийняття рішень. Такий підхід називається маніпулюванням символами, різновидом якого вважають людське мислення [6].

У символьному підході символічні репрезентації є описовим продуктом людини, таким чином вони можуть бути безпосередньо зрозумілими та інтерпретованими людиною. Це є недоліком символьного підходу, адже символічні репрезентації знань залежать від програміста та обмежують систему до ідеалізованого опису, який залежить від когнітивних вимог людської діяльності. З іншого боку, при символьному підході увага в когнітивній архітектурі зосереджена на аспектах пізнання які є постійними з часом і відносно незалежними від завдання. Тому перевага систем, створених на основі символьного підходу, полягає в тому, щоб фіксувати статистичні закономірності під час процесу пізнання, наприклад у навчальних даних. Основними проблемами підходу є проблема приземлення символів, яка стосується того, як символи отримують своє значення, та пошук значущих властивостей у великому наборі даних, а потім їх узагальнення.

При емерджентному підході до моделювання когнітивних архітектур процес пізнання розглядається як процес адаптації до навколишнього середовища шляхом збереження власної автономії. Це досягається через процес самоорганізації, завдяки якому системи реагують на навколишнє середовище у режимі реального часу, щоб підтримувати свою працездатність. Процес осмислення взаємодій з середовищем є однією з основ активного підходу до пізнання при емерджентному підході [7].

Репрезентації у такому підході є результатом діяльності когнітивної моделі, її досвідом взаємодії з середовищем. Таким чином сприйняття інформації про середовище є отриманням сенсорних даних, а не процесом абстрагування середовища. Основною моделлю когнітивного навчання є випереджувальне формування навичок, а не набування знань. Це є перевагою емерджентного підходу перед символьним, адже репрезентація інформації не залежить від програміста, а формується при взаємодії моделі з середовищем. Недоліком підходу є складність моделювання через взаємну специфікацію та спільну розробку емерджентних моделей та середовища. На цей час існують лише

загальні фреймворки моделювання емерджентної архітектури, але не існує повністю визначеної моделі пізнання [8].

Гібридний підхід поєднує у собі аспекти символьного та емерджентного підходів до моделювання когнітивних архітектур. Основна ідея підходу полягає у використанні систем штучного інтелекту, уникаючи використання знань, залежних від програміста, та моделі поведінки сприйняття-дія для формування репрезентацій. Як правило, при гібридному підході у когнітивній моделі використовується символічне представлення знань та логічні системи, засновані на правилах, для розуміння знань з метою досягнення цілі та прийняття рішень, водночас використовуючи емерджентні моделі сприйняття на прийняття рішень для дослідження середовища та конструювання нових знань [6].

Таким чином гібридні моделі використовують символьний підхід для представлення знань, але вони формуються самою системою при взаємодії та досліджені середовища, а не через апріорну специфікацію чи програмування. Для цього об’єкти середовища представляються, як унікальні комбінації умов та відповідей, а властивості об’єкта потрібно вивчати за допомогою маніпуляцій з ним. З цього виходить, що здатність гібридних моделей інтерпретувати невідомі об’єкти залежить від її здатності гнучко взаємодіяти з ними.

1.2. Огляд та порівняння когнітивних архітектур

Когнітивні архітектури визначають те, як саме мають бути представлені знання в моделі та механізми навчання, які їх набувають. Вони описують припущення щодо уявлень і механізмів, які лежать в основі пізнання, як правило, включаючи ідеї психології про природу людського розуму.

Аналіз імплементацій когнітивних архітектур виявив найбільш підтримувані продукти, такі як ACT-R, SOAR, 4CAPS, LIDA та DUAL.

Когнітивна архітектура ACT-R зосереджена на модульній декомпозиції пізнання та пропонує теорію, як ці модулі інтегруються для створення процесу когерентного пізнання. Кожен модуль виконує певну когнітивну функцію та

працює незалежно від інших. Декларативний модуль системи відповідає за отримання інформації з довгострокової пам’яті, модуль цілі слідкує за внутрішнім станом моделі при розв’язанні задачі, процедурний модуль координує функціонування інших модулів за допомогою буферів для обміну інформацією (рис. 1.2). ACT-R працює циклічно, на кожному кроці процедурний модуль запитує в інших модулів інформацію, надаючи обмеження у вигляді продукцій, що являють собою правила типу умова-дія. Ці модулі мають бути надані для мінімальної конфігурації моделі [9].

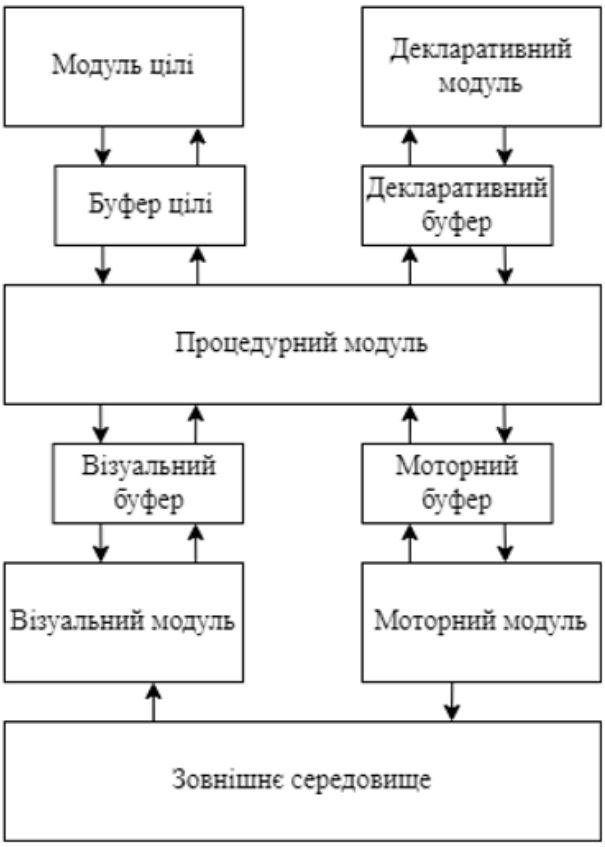

Рис. 1.2. Архітектура ACT-R

SOAR складається з довгострокової пам’яті, яка представлена як набір продукційних правил, та короткочасної пам’яті, яка представлена у вигляді символьної структури графа (рис. 1.3). Короткочасна пам’ять містить у собі оцінку агентом поточної ситуації, отриману на основі сприйняття та використання знань з довгострокової пам’яті. Дія в середовищі відбувається

шляхом створення моторних команд у буфері короткочасної пам’яті. Процедура прийняття рішень вибирає оператори та виявляє безвихідні становища. Робота SOAR складається у зіставленні та запуску продукційних правил. Правила забезпечують гнучке представлення знань при цьому їх умови відповідають поточній ситуації, а їх дії отримують інформацію, що стосується поточної ситуації [10].

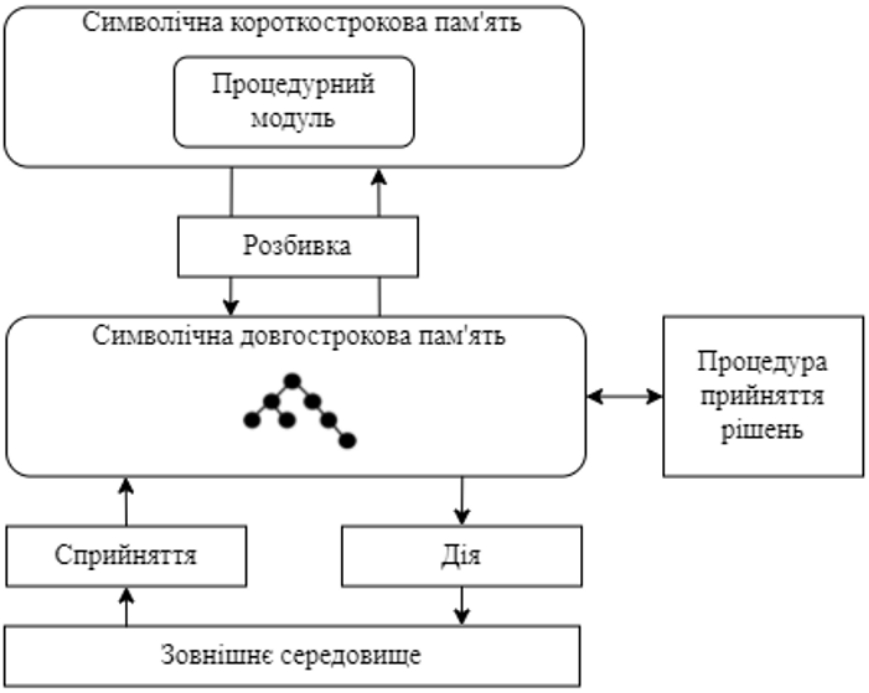

Рис. 1.3. Архітектура SOAR

Архітектура 4CAPS розглядає мислення, як продукт одночасної діяльності багатьох «центрів», які моделюють роботу певних ділянок мозку. Кожен центр – це гібридна система, яка обробляє символічні атрибути. Кожен центр може виконувати кілька когнітивних функцій, і, навпаки, деякі когнітивні функції можуть виконувати більше ніж один центр. Кожен центр має обмежену місткість обчислювальних ресурсів, що обмежує його діяльність. Декларативні знання в архітектурі представлені декларативними елементами пам’яті, які мають набір атрибутів, кожен з яких має символічне або числове значення. Процедурні знання кодуються продукційними правилами типу умова-дія.

Призначення когнітивних функцій центрам динамічно змінюється, адаптуючись до доступності ресурсів і до функціональних вимог завдання [11].

LIDA розглядає декілька типів спеціалізованої пам’яті: сенсорну, сенсорно-моторну, перцептивну, епізодичну, декларативну та процедурну. Процес роботи архітектури являє собою серію когнітивних циклів, кожен з яких складається з фаз відчуття, відстежування та дії. У фазі відчуття поточне представлення внутрішнього та зовнішнього середовища системи оновлюється. Вхідні сенсорні дані активують детектори ознак, вихідні дані яких оброблюються перцептивною пам’яттю, де детектори ознак вищого рівня обробляють інформацію далі. Остаточні результати надсилаються в робочу область, де оброблюються декларативною та епізодичною пам’яттю для створення асоціацій. На етапі відстежування дані формуються у коаліції та переміщуються до глобальної робочої області, де кожна коаліція транслюється по всій системі та проходить різні типи навчання. Після трансляції у процедурній пам’яті обираються можливі дії та надсилаються до модуля вибору дій, який вибирає одну дію для виконання процесом [12].

Системи когнітивної архітектури DUAL складаються з великої кількості тісно взаємопоєднаних гібридних агентів, які діють паралельно. Кожен з них може виконувати конкретну задачу та представляти певні знання. Кожен агент складається з двох частин: L-Brain і R-Brain, розроблених відповідно до символічної та емерджентної парадигми (рис. 1.4). Частина L-Brain являє конкретну частину знань реального світу, тоді як частина R-Brain представляє релевантність цього знання для конкретного контексту. Довгострокова пам’ять в архітектурі представлена усіма агентами, а короткострокова пам’ять складається з активних на цей час агентів. Таким чином, продуктивність когнітивної системи виникає як результат роботи та взаємодії активних агентів, де набір активних агентів не визначений заздалегідь для конкретного завдання, а є динамічним і показує конкретний контекст [13].

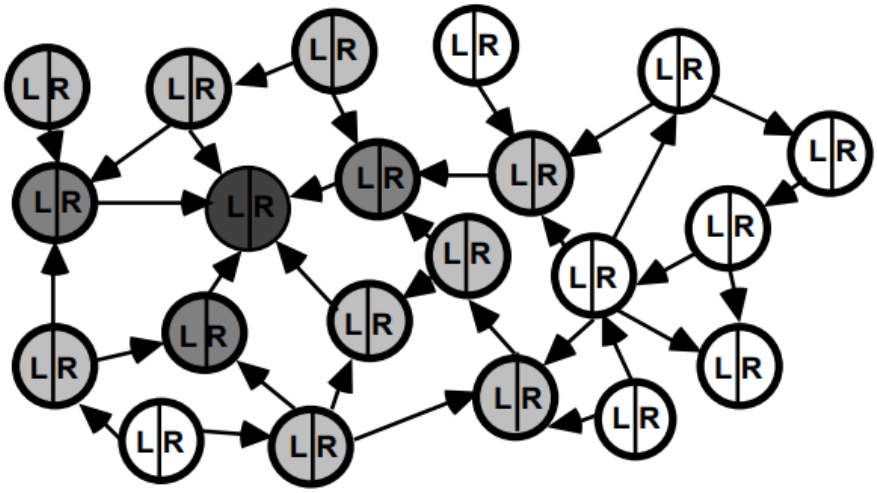

Рис. 1.4. Архітектура DUAL

Для створення поведінки штучних когнітивних агентів, досліджувані архітектури моделюють їх поведінку в термінах когнітивного циклу. Когнітивний цикл складається з набору кроків, які виконуються послідовно або паралельно, що завершується вибором дії, яку має виконати агент. Хоча досі немає узгодження щодо того, якими є ці кроки та як вони взаємодіють у процесі пізнання людини, але є досить точні пропозиції, які описують деякі з цих процесів [14].

Огляд когнітивних архітектур дозволяє порівняти їх за наступними критеріями, які є процесами, що зазвичай беруть участь у когнітивному циклі пізнання людини: представлення цілей, процес сприйняття, механізми навчання та методи розв’язання проблем.

Постановка цілі – це когнітивний процес мозку на рівні сприйняття, який представлений станом, якого агент хоче досягти для виконання дії. Цілі в архітектурах зазвичай представляються у формі стеку або списку, у якому вони постійно конкурують між собою за стан поточної цілі. ACT-R, наприклад, зберігає інформацію про поточну ціль у буфері цілі, тоді як проміжні уявлення зберігаються в модулі цілі [9]. SOAR зберігає цілі у вигляді стану, але якщо процедурні знання не можуть повністю розв’язати проблему, архітектура автоматично розділяє проблему на підпорядковані цілі, і для кожної такої цілі

виконується новий цикл [10]. LIDA представляє цілі у вигляді відчуттів, які впливають на покращення стану когнітивного агента. Порівняльна характеристика методів представлення цілей у когнітивних архітектурах наведена у табл. 1.1.

Таблиця 1.1

Таблиця порівняння когнітивних архітектур за методами представлення цілей

Назва архітектури	Метод представлення цілей
SOAR	Цілі представлені у вигляді стану, з утворенням ієрархії підпорядкованих цілей у разі виникнення тупика
ACT-R	Цілі зберігаються у модулі цілей та доступні продукційному модулю через буфер цілей.
LIDA	Цілі представлені у вигляді набору відчуттів агента.
4CAPS	Для збереження цілей визначений клас цілей у декларативній пам’яті.
DUAL	Цілі представлені активними агентами у пам’яті.

Процес сприйняття охоплює когнітивні процеси, необхідні для організації, ідентифікації та інтерпретації сенсорної інформації для представлення та розуміння навколишнього середовища. Сприйняття у когнітивних архітектурах визначає, як само когнітивний агент отримує інформацію з середовища за допомогою різних модальностей, подібно то того, як людина має доступ до зору, слуху та дотику [15]. Архітектура SOAR отримує дані з навколишнього середовища через перцептивні входи та формує символічні представлення за допомогою модуля кластеризації, які потім зберігаються у короткочасній пам’яті. У архітектурі DUAL агенти отримують інформацію

навколишнього середовища через сенсори, які перетворюють її у символічне представлення. Методи сприйняття інформації в інших архітектурах наведені у табл. 1.2.

Таблиця 1.2

Таблиця порівняння когнітивних архітектур за методами сприйняття

Назва архітектури	Метод сприйняття
SOAR	Інформація о середовищі зберігається в декларативній пам’яті у вигляді символічних представлень.
ACT-R	Реалізує зорові та слухові модулі для сприйняття інформації.
LIDA	Данні отримуються через сенсори у підсистемі сприйняття.
4CAPS	Пропонує структуру планування, яка може представляти переконання не залежно від сприйняття.
DUAL	Середовище сприймається за допомогою сенсорів і перетворюється на символічні структури.

Розв’язання проблеми – це здатність архітектури шукати рішення, коли виникає тупик. SOAR архітектура створює підпорядковану ціль з основної цілі та виконує увесь цикл прийняття рішень у пошуках рішення для цієї цілі. У ACT-R архітектурі проблема вирішується шляхом зіставлення поточного стану зі станом цілі, щоб побачити, яка відстань до цілі лишилася. Порівняння механізмів розв’язку проблем у різних архітектурах наведено у табл. 1.3.

Таблиця 1.3

Таблиця порівняння когнітивних архітектур за методами розв'язання проблеми

Назва архітектури	Метод розв'язання проблеми
SOAR	Створення підпорядкованих цілей з основної цілі
ACT-R	Зіставлення поточного стану зі станом цілі
LIDA	Алгоритм розв'язання незвичних задач (non-routine tasks)
4CAPS	Створення підпорядкованих цілей з основної цілі
DUAL	Активація агентів за допомогою функції активації при символьній обробці даних

Навчання є вирішальним процесом у пізнанні та дуже корисним для штучних агентів, які використовують свої попередні помилки та успіхи, щоб зробити наступну дію. Архітектура SOAR реалізує групування – процес перетворення підпорядкованих цілей, отриманих у результаті розв'язання проблеми, на правила. 4CAPS пропонує механізм навчання у вигляді динамічного формування великомасштабних мереж. Повний огляд механізмів навчання архітектур наведений у табл. 1.4.

Таблиця 1.4.

Таблиця порівняння когнітивних архітектур за методами навчання

Назва архітектури	Механізм навчання
1	2
SOAR	Механізм групування

Продовження таблиці 1.4

1	2
ACT-R	Декларативне навчання, процедурне навчання
LIDA	Перцептивне, епізодичне та процедурне навчання
4CAPS	Організація динамічних великомасштабних мереж
DUAL	Послідовна активація агентів

Кінцевою метою вивчення когнітивної архітектури є створення автоматизованих систем, які можуть краще розуміти людське пізнання, щоб моделювати його або мати змогу до реагування подібно людині. Різні архітектури зосереджуються на різних аспектах людського пізнання та вивчають мозок і поведінку людини в різних вимірах. Кожна з архітектур має свої сильні та слабкі сторони, які роблять її унікальною. Після порівняння був зроблений висновок, що кожна архітектура має певні переваги та певні обмеження, але жодна архітектура не здатна впоратися з більшістю проблем у вивченні та моделюванні процесу пізнання. Розглянуті архітектури пропонують лише загальний підхід до розв’язання проблем у реальному часі, але не містять детальних інструкцій пізнання. Також важливим питанням є оцінка когнітивних моделей, побудованих за когнітивними архітектурами.

1.3. Задача оцінки когнітивних моделей

Когнітивне моделювання – це методологія дослідження, яка розглядає пізнання як обчислювальний процес. Оскільки когнітивне моделювання створює теорії процесу пізнання людського мозку, які можуть бути повними, послідовними та генеративними, необхідний формальний аналіз області, а також емпіричні дослідження не тільки як підстава для моделювання, але і як засіб

перевірки та оцінки цих моделей. Таким чином у когнітивній науці та когнітивному моделюванні важливою проблемою є задача оцінки когнітивних моделей.

При когнітивному моделюванні механізми пізнання формалізують в термінах математичних структур, що містять параметри, які є теоретично значущими. Проблема формалізації та оцінки цих параметрів полягає у тому, що формальні когнітивні моделі виходять за межі вербальних моделей у тому, що когнітивні механізми реалізуються з точки зору математики, і вони виходять за рамки статистичних моделей, оскільки параметри когнітивної моделі піддаються психологічній інтерпретації [16].

У загальному вигляді, процесом аналізу та оцінки когнітивних моделей є відповідь на наступні питання [17]:

1. Як само ми можемо описати процес навчання у термінах когнітивної моделі?
2. Як ми можемо оцінити когнітивну модель з мінімальними затратами ресурсів?
3. Як ми можемо використати інформацію, отриману під час оцінки моделі для її вдосконалення?

Базуючись на цих питаннях, виділяють три ключові елементи під час тестування когнітивних моделей: оцінка параметрів, прогнозування моделі та вибір моделі. Аналіз предметної області дозволив виділити основні підходи до аналізу та оцінки когнітивних моделей: традиційний та байєсівський статистичний [18].

При традиційному підході, оцінка параметрів, прогнозування та вибір моделі здійснюється за допомогою використання оптимізаційних та статистичних методів. Оцінка параметрів когнітивної моделі включає пошук значень параметрів моделі, які мінімізують або максимізують деяку цільову функцію, яка вимірює точність відповідності прогнозів моделі спостережуваним даним. У перших роботах з когнітивного моделювання використовували мінімізацію суми квадратної помилки або середньоквадратичної помилки, а

також максимізацію кореляції між моделлю та даними. Недоліком цих методів було те, що ці цільові функції можуть підходити для оцінки параметрів лінійних статистичних моделей, але вони часто не підходять для нелінійних моделей, що є властивістю багатьох когнітивних моделей. Кращим традиційним підходом до оцінки параметрів у когнітивному моделюванні є оцінка максимальної вірогідності [19]. Мета методу полягає в тому, щоб знайти вектор параметрів, який максимізує вірогідність даних з даними параметрами.

Оптимізація параметрів моделі при традиційному підході виконується за допомогою симплекс-методу. Симплекс-метод – це алгоритм оптимізації задачі лінійного програмування, метою якого є побудова базисних рішень у яких монотонно зменшується лінійний функціонал, до ситуації, коли виконуються необхідні умови локальної оптимальності. Враховуючи початкову точку, або набір вхідних точок, максимізація вірогідності або мінімізація середньоквадратичного відхилення за симплекс методом дає точкову оцінку для найбільш відповідного вектору параметрів. Але метод не бере до уваги міру невизначеності при оцінці параметра. Щоб отримати оцінки невизначеності, необхідно виконати параметричне або непараметричне завантаження, яке дає оцінку довірчого інтервалу навколо оцінки точки максимальної вірогідності [20].

Прогнозування моделі при традиційному підході відбувається з використанням оптимальних параметрів, які були згенеровані методом оцінки параметрів. Прогнози моделей базуються лише на точкових оцінках, ігноруючи будь-яку невизначеність оцінок параметрів, що є потенційним недоліком процесу прогнозування при традиційному підході.

Під час вибору моделі, що конкурують, кількісно порівнюються між собою відповідно до того, наскільки вони відповідають спостережуваним даним. Але вибір моделі «переможця», як моделі з найбільшою вірогідністю не враховує відносну складність або гнучкість моделей, що конкурують. Більш загальна модель математично відповідає так само, якщо не краще, ніж окремий випадок цієї загальної моделі. У цьому випадку при виборі моделі питання полягає не в тому, чи підходить окрема модель гірше, а в тому, чи підходить ця модель значно

гірше, ніж більш загальна модель. У ситуації, коли одна модель не є окремим випадком іншої моделі, модель з більшою кількістю вільних параметрів або з більшою гнучкістю підійде краще, ніж модель з меншою кількістю параметрів та більшими обмеженнями. Тому у цьому випадку питання не полягає у тому, чи підходить складна модель краще, а в тому чи підходить вона краще, навіть якщо ця модель має штраф за свою складність [18]. Традиційні методи, які штрафують міри вірогідності під час порівняння моделей на основі складності, містять у собі байєсівський інформаційний критерій (BIC) [21] та інформаційний критерій Акаіке (AIC) [22]. Ці методи передбачають додавання штрафу на основі кількості вільних параметрів, ігноруючи інші аспекти складності моделі, такі як функціональна форма моделі та розмір простору можливих передбачень.

Альтернативним підходом до оцінки когнітивних моделей є байєсівський підхід, який дозволяє уникнути певні обмеження традиційних підходів. Байєсівський аналіз когнітивної моделі заснований на формулі Байєса (1.1):

$$P(h_i|D) = \frac{P(h_i) * P(D|h_i)}{P(D)}, \quad (1.1)$$

де $P(h_i)$ – ймовірність події h_i;

$P(h_i|D)$ – ймовірність події h_i, за умови, що подія D відбулась;

$P(D|h_i)$ – ймовірність події D, за умови, що подія h_i відбулась;

P(D) – ймовірність події D.

Ця формула використовується для обчислення повного апостеріорного розподілу ймовірностей параметрів моделі. При такому підході вірогідність параметрів визначається когнітивною моделлю так само, як це було б визначено при традиційній оцінці максимальної вірогідності. Попередній розподіл вірогідностей показує суб’єктивне переконання щодо значень параметрів ще до того, як дані будуть розглянуті [23]. Ці переконання можуть бути нечіткими, можуть ґрунтуватися на попередніх даних, які використовувалися для оцінки параметрів або вони можуть ґрунтуватися на значеннях параметрів, які мають

значення за якимось теоретичним, об’єктивним або суб’єктивним критерієм. Формула Байєса також забезпечує міру невизначеності параметра, оскільки обчислює апостеріорний розподіл ймовірностей значень параметрів, що дозволяє використовувати її як рішення для прогнозування моделі, яке враховує цю невизначеність. Байєсівський підхід також дозволяє використовувати засоби для порівняння моделей і вибору моделі, які враховують складність моделі природним, узгодженим і комплексним чином. У той час як традиційні методи оцінки параметрів вимагають пошуку простору параметрів для мінімізації або максимізації певної цільової функції, правило Байєса одразу обчислює вірогідність значень параметрів для моделі з урахуванням спостережуваних даних. Але попри його уявну простоту, навіть для моделей з невеликою складністю, правило Байєса не може бути розв’язане аналітично і вимагає обчислювальної оцінки [23].

Коли обчислювальна модель використовується для оцінки теорій, у загальному вигляді основною метою є узгодженість з даними людини. Але це не завжди є достатнім для доказу теорії. У цьому випадку реалізація когнітивних механізмів з точки зору математики не дозволяє повністю довести теорію когнітивного пізнання мозку людини, оскільки при моделюванні виникає ряд проблем. Марк Халбрюгге у своїй роботі з оцінки когнітивних моделей зазначає наступні проблеми при когнітивному моделюванні [24]:

1. Відповідність теорії. Основна теорія та її реалізація у когнітивній архітектурі можуть розходитися.

2. Нерелевантна специфікація. Більшість моделей вимагають додаткових припущень. Дуже важко сказати, які частини моделі відповідають за її загальну продуктивність, а які є застарілими [25].

3. Засадничі теорії. Особливий випадок проблеми невідповідної специфікації може виникнути, коли архітектура використовує та базується на знаннях з попередніх досліджень, наприклад закон Фіттса [26], про час необхідний для виконання швидкого руху передпліччя. Велика частина відповідності моделі буде заснована саме на цих емпіричних засадничих законах.

4. Інтерпретація. Щоб зрозуміти, як насправді функціонує модель, необхідно подивитися на її код. Оскільки багато спеціалістів з психології не завжди мають професійну освіту з інформатики, це заважає їм брати участь у досліджені когнітивного моделювання.

Когнітивні архітектури не розв'язують ці проблеми повністю, але значною мірою їх послаблюють. Але ще одним важливим питання встає підтвердження або валідація когнітивної архітектури. Андерсон та Лаб'єр запропонували список з дванадцяти критеріїв для оцінки когнітивних архітектур: гнучка поведінка, продуктивність у реальному часі, адаптивна поведінка, велика база знань, динамічна поведінка, інтеграція знань, природна мова, навчання, розвиток, еволюція та реалізація мозку [27]. Марк Халбрюгге наголошує, що ще одним важливим критерієм оцінки та підтвердження когнітивних архітектур є фальсифікованість [24]. Він ставить питання, чи можна взагалі підтвердити когнітивну архітектуру.

Наприклад для архітектури ACT-R, Андерсон та Лаб'єр стверджують, що ACT-R не є «загальною обчислювальною системою, яку можна запрограмувати на будь-що» [27]. Але попри те, що обчислювальна потужність архітектури може бути обмежена, з формальної точки зору така архітектура є завершеною за Тюрінгом. Це твердження вперше було висунуто Векслером [28]. Воно наголошує, що когнітивні архітектури є повними машинами Тюрінга. Це означає, що вони можуть обчислити все, що обчислюється за допомогою такого типу машин. Тому когнітивну архітектуру не можна сфальсифікувати. А отже, оскільки когнітивна архітектура не може бути сфальсифікована, вона не може повною мірою бути доказом когнітивної теорії пізнання [29].

1.4. Постановка задачі

Метою кваліфікаційної роботи є підвищення ефективності навчання студента через адаптивне формування навчальних методик на основі аналізу його когнітивної моделі.

Для досягнення поставленої мети необхідно дослідити когнітивні процеси, які впливають на процес навчання, методи класифікації когнітивних навичок, що використовуються різними навчальними методиками, визначити механізм збору інформації щодо змін когнітивних навичок протягом процесу навчання.

Після проведених досліджень необхідно проаналізувати зібрані відомості та на підставі результатів аналізу обґрунтувати застосування тих чи інших навчальних методик.

1.5. Висновки

У даному розділі був проведений аналіз предметної галузі, а саме було визначено поняття когнітивної моделі та основних підходів до її побудови. Когнітивні моделі надають змогу представити процес пізнання людини у вигляді обчислювальних моделей, а також дозволяють отримати кількісну оцінку когнітивних процесів, які беруть участь у пізнанні.

Аналіз та порівняння архітектур для побудови когнітивних моделей дозволив виявити, що, по-перше, не існує правдивої методики оцінки когнітивних моделей, що є проблемою у цій галузі науки, по-друге, найпоширеніша архітектура для побудови когнітивних моделей ACT-R використовує заскладний та непрактичний інструментарій для такого роду моделювання. Тому виникає проблема, по-перше, створення відповідного інструментарію когнітивного моделювання, а по-друге, виникає необхідність у розробці механізму оцінки або порівняльної оцінки стану когнітивної системи студента до використання щодо нього певної навчальної методики та після.

РОЗДІЛ 2
МЕТОДИ ДОСЛІДЖЕННЯ КОГНІТИВНИХ ПРОЦЕСІВ

2.1. Когнітивні процеси

Когнітивні процеси є однією з найбільш досліджуваних тем у когнітивній науці. Розуміння роботи мозку важливо для побудови моделей, які допоможуть точно моделювати процес пізнання людини та аналізувати його.

Когнітивні процеси – це процеси, які ґрунтуються на мозковій діяльності, які необхідні для отримання знань, маніпулювання інформацією та міркування для вирішення будь-яких завдань [30]. Ці процеси виникають у розумі людини під час отримання стимулу з навколишнього середовища та до моменту явної поведінкової реакції на цей стимул [31].

Можна виділити наступні основні когнітивні процеси розумової діяльності людини [30]:

– сприйняття – це здатність розпізнавати та інтерпретувати сенсорні подразники навколишнього середовища;

– увага – це здатність підтримувати концентрацію на певному об’єкті, дії чи думці;

– пам’ять – це здатність зберігати, накопичувати та відновлювати знання, уміння та навички. Виділяють два основні типи пам’яті: короткочасна (обмежена пам'ять) і довготривала пам'ять (необмежене зберігання);

– мовлення – це навички, що дозволяють нам інтерпретувати звуки в слова і створювати словесні результати;

– візуальна та просторова обробка – це здатність обробляти вхідні візуальні стимули, що отримані за допомогою сприйняття, розуміти просторові відносини між об’єктами, а також візуалізувати образи;

– розв’язання проблеми – це процес пошуку рішень у важких або складних ситуаціях;

– прийняття рішень – це здатність приймати рішення на основі розв’язання проблем, неповної інформації та емоцій.

Таким чином, когнітивні процеси є основою розумної діяльності людини, а також є важливими у пізнанні людини. Вони доповнюють один одного, щоб ефективно функціонувати та визначають успішність результатів пізнання.

Важливим питанням у дослідженні розумових процесів є питання побудови моделей когнітивного пізнання людини. Дослідників цікавить не лише механістичне функціонування когнітивних процесів, а й те, як ці процеси пов’язані один з одним, як вони пов’язані з результатами поза експериментів та біологічними основами когнітивних процесів у мозку, а також як ці процеси змінюються для різних вікових груп [32, 33]. Моделювання повного циклу пізнання людини з урахуванням усіх когнітивних процесів є дуже комплексним завданням, оскільки вимагає великої кількості експериментальних даних та моделювання не тільки кожного процесу, а й взаємозв’язків між ними. Тому у даній роботі було вирішено обрати конкретний когнітивний процес для побудови моделі для аналізу, наприклад процес прийняття рішень, а саме прийняття рішень студентом під час виконання тестів.

2.2. Процес прийняття рішень як один з когнітивних процесів

Прийняття рішень – це високорівневий когнітивний процес, заснований на процесах сприйняття, пам’яті та уваги. Ситуації, які виникають у реальному житті, вимагають прийняття ряду рішень, які залежать від попереднього досвіду людини.

Традиційний підхід до формування стратегії прийняття рішень полягає в порівнянні судження або рішення зі стандартом або «еталоном». Порівняння дає змогу оцінити конкретне судження як «добре» чи «погане» у порівнянні з еталоном. Чіткі правила, отримані з економіки (теорія очікуваної корисності) чи математики (теорія ймовірності), що представлені у нормативних моделях, які забезпечують ці стандарти, використовуються для перевірки прогнозів щодо

поведінки людей у прийнятті рішень. Але у реальному житті процес прийняття рішень зазвичай є динамічним, тобто на прийняття рішень може впливати багато факторів, таких як зміна навколишнього середовища або попередній досвід людини, її пам’ять.

На відміну від традиційних досліджень прийняття рішень, які акцентуються на нормативних моделях та відхиленнях від передбачень цих моделей, когнітивна наука концептуалізує людей як «обробників інформації», які використовують процеси сприйняття, пам’яті, категоризації, розв'язання проблем тощо. Багато процесів, описаних у когнітивних теоріях, подібні до процесів, які беруть участь у прийнятті рішень, на основі яких ґрунтується моделювання процесу прийняття рішень.

Наприклад американський психолог Ульрих Найсер представив ідею, що розумний організм працює в циклі сприйняття-дія: органи почуттів отримують інформацію з навколишнього середовища, розум виконує обчислення на основі цієї інформації, а результати цих обчислень використовуються для вибору дій [34]. Ключовою характеристикою цієї ідеї є те, що існує обмеження на те, скільки інформації можна обробити, і, отже, організм повинен бути селективним у тому, на що він звертає увагу в навколишньому середовищі [35].

Взаємодія між увагою і пам’яттю також є фундаментальною для процесу прийняття рішень у когнітивному моделюванні. Для представлення того, як різні форми інформації зберігаються в тимчасовому сховищі пам'яті у когнітивній науці представлено поняття робочої пам'яті. Вона відповідає за розподіл уваги когнітивного агента на різні завдання обробки, такі як контрольована думка, необхідна для розв’язання проблем, прийняття рішень, міркування тощо [36]. Завдання, з якими стикалися багато разів у минулому, стають простими у виконанні або розв’язанні, оскільки відповідні дії чи рішення можна отримати з пам’яті, і, отже, продуктивність менше залежить від активної уваги. У традиційній «парадигмі азартних ігор» JDM (judgement and decision making) [37], засновані на досвіді зміни в когнітивних процесах не враховуються [38].

Іншим важливим аспектом обробки інформації є те, що організми наділені здатністю адаптивно змінювати свою поведінку залежно від попереднього досвіду та змін навколишнього середовища, тобто вчитися. Люди здатні вивчати непередбачуваність подій і дій. Це здатність є фундаментальною для виживання в мінливому середовищі. Розуміння причинно-наслідкової структури світу, отримане завдяки цьому процесу навчання, також сприяє причинно-наслідковим міркуванням та індукції, що своєю чергою може призвести до розвитку категоризації.

Категоризація – це розумова операція, яка групує об’єкти за схожими ознаками, це фундаментальна здатність, яка дозволяє нам організувати свої знання, реагувати належним чином і робити корисні прогнози щодо властивостей «речей», які ми зустрічаємо у світі [39, 40].

У роботі Пітера Жюслена та Генріка Олсона було зроблено глибоке спостереження, що дослідження категоризації та багатоатрибутних суджень часто ставлять одні й ті ж основні запитання (наприклад, як ви оцінюєте, чи є людина другом чи ворогом) і часто використовують однакові засадничі теорії. При формуванні нових категорій з заданого набору без чітких інструкцій, спочатку виділяють ознаки, що відрізняють різні елементи. Після цього формуються гіпотези про відповідні ознаки, які перевіряються шляхом прийняття послідовних рішень. Послідовні рішення щодо категоризації підкріплюються зворотним зв’язком, який вказує, чи було рішення правильним чи ні [41].

Таким чином когнітивна наука дозволяє розглядати процес прийняття рішень, як динамічний, описуючи механізми моделювання факторів, що впливають на рішення людини. На рис. 2.1 представлено схематичне співвідношення між процесом прийняття рішень і видами когнітивних моделей, які лежать в його основі.

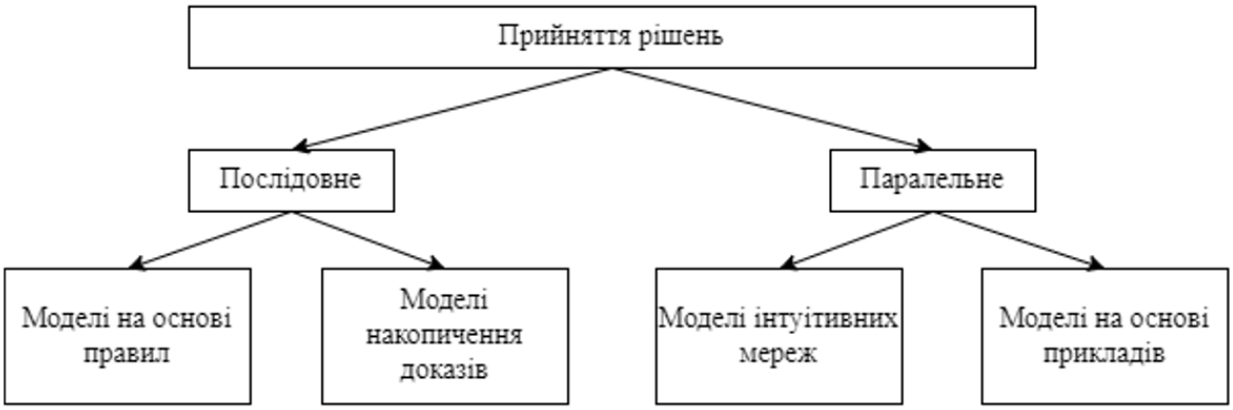

Рис. 2.1. Схема когнітивних моделей процесу прийняття рішень

2.3. Когнітивні підходи до прийняття рішень

Одним з підходів до дослідження динамічного прийняття рішень є комплексне пізнання [42]. Цей підхід досліджує, як різні психічні процеси впливають на планування дій, розв'язання проблем і прийняття рішень. Термін «психічні процеси в комплексному пізнанні» включає не лише когнітивні, а й мотиваційні аспекти. Натуралістичне дослідження з прийняття рішень розглядає, як рішення приймаються «в дикій природі». Реальні рішення, прийняті людьми з певним досвідом, досліджуються в контексті обмеженого часу, суперечливих цілей, умов, що динамічно змінюються, та джерел інформації різної надійності.

Робота Ансона Лі та Камбіз Маані пропонує інший погляд на динамічне прийняття рішень. Вони розглядають цей процес як безперервний цикл оновлення ментальної моделі. Вони описують цей процес за допомогою циклу CER. CER розшифровується як концептуалізація, експеримент та рефлексія [43]. Концептуалізація – це розуміння ситуації та моделювання результату потенційних рішень і пов'язаних з ними дій. Таким чином, особа, яка приймає рішення, порівнює дану ситуацію з пов'язаною інформацією у своїй ментальній моделі та інтегрує нову інформацію, отриману з середовища, для вироблення набору рішень. Під час експериментів рішення, розроблені на основі ментальної моделі особи, яка приймає рішення, динамічно перевіряються у реальному світі. На фазі рефлексії оброблюється зворотний зв'язок, отриманий від результатів

експериментальної фази. Якщо очікуваний результат досягнутий (наприклад, позитивний відгук), початкові рішення залишаються стійкими. Якщо, однак, результат є неочікуваним (наприклад, негативний відгук) або якщо отримані результати відрізняються від очікуваного, особа, яка приймає рішення, оновлює свою ментальну модель. Для цього він або вона вирішується на альтернативні дії, такі як пошук нових джерел інформації для прийняття кращих рішень.

При динамічному прийнятті рішень, рішення не є фіксованими, а змінюються за допомогою інформації, що надходить. Враховуються не лише окремі аспекти прийняття рішень, такі як увага, а й фактори середовища, які дають зворотний зв'язок про дію. Вард Едвардс описує три аспекти, які визначають динамічне прийняття рішень. По-перше, необхідно виконати ряд дій для досягнення певної мети з часом. По-друге, дії залежать один від одного. Таким чином, на рішення впливають попередні дії. По-третє, в результаті цих дій відбуваються зміни в навколишньому середовищі, але також зміни можуть відбуватися випадково, що є найскладнішим аспектом для дослідження [44].

Таким чином, для вивчення динамічного прийняття рішень, необхідно промоделювати наступні характеристики: послідовне прийняття рішень зі зворотним зв'язком, багатофункціональні стимули та перемикання призначень категорій. Щоб визначити, як люди вивчають приналежність ознак у динамічному середовищі, і дослідити, як виникають стратегії прийняття рішень зі складністю, що зростає, спочатку необхідно розробити підхід до моделювання, що стосується цих аспектів. Такі моделі можна використовувати для вдосконалення навчальних методик людини у когнітивному процесі прийняття рішень на індивідуальному рівні.

2.4. Прийняття рішень на основі прикладів

Найбільш поширеною моделлю динамічного прийняття рішень у когнітивній науці є теорія навчання на основі прикладів або instance based learning theory (IBLT) [45]. Прийняття рішень на основі прикладів в основному

означає, що рішення про те, яку дію обрати приймається з використанням накопиченого досвіду. Спостереження в реальних, складних ситуаціях (наприклад, бойові дії та гасіння пожеж) підтверджують ідею про те, що в умовах стресу, невизначеності або перевантаження завдань люди приймають рішення переважно на основі досвіду [46]. IBLT описує, що під час прийняття рішень, в пам’яті людини накопичуються приклади поведінки. IBLT описує когнітивно вірогідний процес прийняття рішень, за допомогою якого люди отримують та оновлюють приклади у пам’яті та навчаються.

В IBLT приклад визначається як триплет із ситуації, рішення та корисності (SDU). Ситуація описується як набір сигналів середовища; рішення являє собою набір дій, застосованих до ситуації; а корисність – це оцінка правильності рішення в цій конкретній ситуації.

В теорії також описані наступні механізми навчання людини при прийнятті рішень:

- знання на основі прикладів – механізм накопичення знань у вигляді екземплярів, що містять триплет SDU;
- пошук на основі розпізнавання – механізм пошуку знань в пам'яті відповідно до подібності між ситуацією, що оцінюється, і прикладами, що зберігаються;
- адаптація стратегій – механізм адаптації від евристичних рішень до рішень на основі прикладів відповідно до кількості практики в динамічній задачі;
- необхідність – механізм контролю продовження альтернативного пошуку;
- оновлення зворотного зв’язку – механізм оновлення корисності знань у пам’яті та отримання причинно-наслідкових характеристик результатів дій.

Ці механізми виникають у контексті процесу прийняття рішень, як показано на рис. 2.2.

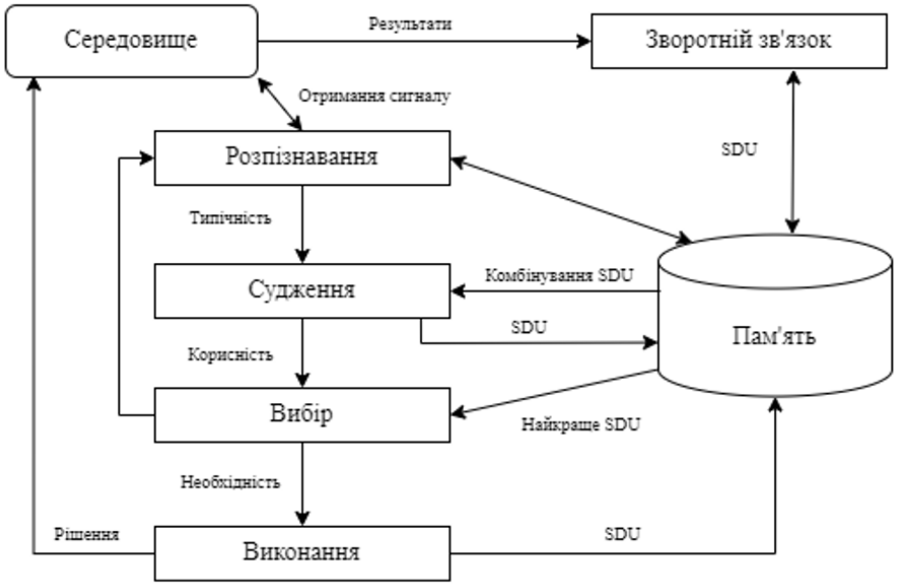

Рис. 2.2. Механізм прийняття рішень у моделі IBLT

Основними кроками в процесі прийняття рішень, запропонованими IBLT, є розпізнавання, судження, вибір і зворотний зв’язок. Прийняття рішень починається з пошуку альтернатив і класифікації цих альтернатив як типових чи нетипових. Ситуація є типовою, якщо є спогади про подібні ситуації, тоді як нетипові ситуації оцінюються за допомогою евристичного або сукупного значення корисності з минулого досвіду. Далі виникає вибір: шукати більше альтернатив або обрати поточну найкращу альтернативу. Результат вибору визначається «рівнем прагнення» особи [47]. Більше альтернатив оцінюється, поки особа, яка приймає рішення, «незадоволена» поточною найкращою альтернативою. Основним визначальним фактором «задоволеності» є час, що залишився, для прийняття рішення, тому якщо часу не лишилося, людина обирає поточну найкращу альтернативу. Після обрання пам'ять про рішення змінюється. Триплети SDU накопичуються в міру того, як виникає більше альтернатив і більше ситуацій прийняття рішень. У певний момент надається зворотний зв’язок із зазначенням результатів попередніх рішень.

Етап розпізнавання відповідає за пошук альтернатив у навколишньому середовищі. IBLT наголошує, що здібності до розпізнавання змінюються з часом від евристичного прийняття рішень до прямого пошуку рішення, відповідно

поточній ситуації. Недосвідчені особи, які приймають рішення, виконують неорганізований і майже випадковий пошук альтернатив, але з накопиченням досвіду вони розпізнають ситуацію і знаходять відповідне рішення. Таким чином особи, які приймають рішення, вчаться зосереджуватися на факторах, що мають відношення до завдання, і вчаться ігнорувати нерелевантні фактори.

Пошук альтернатив у IBLT здійснюється на основі подібності. У IBLT подібність є функцією відношення між ситуаціями, що визначаються атрибутами завдання. IBLT передбачає, що особи, які приймають рішення, розпізнають ситуацію типовою, коли ознаки попередніх ситуацій збігаються з ознаками поточної ситуації.

Наступним етапом після розпізнавання є судження. На ньому особа, що приймає рішення оцінює можливі рішення та дії для поточної ситуації та їх корисність. IBLT пропонує дві процедури судження: для нетипових ситуацій особи, які приймають рішення, використовують евристики, тоді як для типових ситуацій вони використовують попередні знання. IBLT також пропонує процес адаптації стратегій прийняття рішень від евристичних суджень до суджень на основі прикладів.

На етапі вибору особа, що приймає рішення, визначає найкращий спосіб дій. Раціональні теорії вибору припускають, що всі альтернативи, що мають відношення до вибору, відомі, точні, послідовні та стабільні, тоді як найкраща з усіх альтернатив визначається тим, хто приймає рішення [48]. Але при динамічному прийнятті рішень, усі альтернативи невідомі, оскільки нові можуть з’являтися з часом. IBLT пропонує оцінювати альтернативи одну за одною, і після кожної оцінки робиться вибір між пошуком решти альтернатив і виконанням поточної найкращої альтернативи. IBLT передбачає, що оцінка решти альтернатив визначається рівнем потреб особи, яка приймає рішення. Рівень необхідності вимірюється суб’єктивними або об’єктивними факторами, такими як перевага від вибору альтернативи або час, що залишився.

Етап зворотного зв’язку забезпечує особу, що приймає рішення, знаннями для виявлення результатів прийняття рішення. У IBLT ці знання

використовуються для уточнення триплетів SDU у пам’яті людини. Коли SDU створюється, значення його корисності є прогнозуванням результату дії в поточних умовах. При отриманні зворотного зв’язку, значення корисності оновлюється в кращий або гірший бік в залежності від результату.

Значна частина труднощів, пов’язаних з розробкою точних моделей прийняття рішень людиною, пов’язана з тим фактом, що велика частина знань, отриманих експертами у ситуаціях прийняття рішень є неявною. Саме тому при моделюванні процесу прийняття рішень, необхідно коректно класифікувати когнітивні навички, які беруть участь у цьому процесі.

2.5. Класифікація когнітивних навичок людини

2.5.1. Таксономія Блума

Для моделювання процесу прийняття рішень, необхідно зібрати достатню інформацію щодо тих когнітивних навичок, які людина використовує під час цього процесу.

У 1956 році Бенджамін Блум, намагаючись звести до єдиної системи набір розрізнених цілей і завдань, створив теорію класифікації когнітивних процесів розумової діяльності людини, яка називається таксономія Блума [49].

Таксономія Блума містить шість категорій розумових процесів, починаючи від процесів нижчого порядку, які вимагають меншої когнітивної обробки, до процесів вищого порядку, які потребують глибшого навчання та більшого ступеня когнітивної обробки. Таксономія Блума являє собою ієрархію, яка складається з наступних процесів: знання (knowledge), осмислення (comprehension), застосування (application), аналіз (analysis), оцінка (evaluation) та синтез (synthesis) (рис. 2.3).

Рис. 2.3. Таксономія Блума

Знання є фундаментальним розумовим процесом, найпростішим рівнем класифікації та відповідають здатності людини зберігати певну інформацію, таку як факти або визначення. Знання можна оцінити простими засобами, наприклад, питаннями, які вимагають пошуку інформації. Знання також можуть включати складніші процеси співвідношення і судження, оскільки майже неможливо поставити людину перед проблемою, яка включає точно такі ж стимули або сигнали, які були присутні в іншій ситуації. Але звичайне володіння знаннями не дає доказів його осмислення, що є наступним рівнем у таксономії Блума.

Осмислення інформації людиною виявляється у її здатності перефразовувати її своїми словами, класифікувати її, порівнюючи елементи з іншими подібними сутностями або пояснювати цю інформацію іншим. Осмислення вимагає більшої когнітивної обробки ніж звичайне запам’ятовування інформації. Осмислення дозволяє людині включати знання в

існуючі когнітивні схеми за допомогою яких вони розуміють світ, а також використовувати їх у нових ситуаціях [50].

Третій рівень таксономії Блума застосування визначає уміння використовувати отримані знання у конкретних умовах і нових ситуаціях. Сюди входить застосування правил, методів, уміння розбивати матеріал на складові понять, законів, принципів, теорій.

Рівень аналізу належить до здатності людини розкладати отримані знання на складові частини для розуміння їх організаційної структури. На цьому рівні людина оперує навичками, які є складовими критичного мислення. Аналіз також відповідає за здатність відділяти факти від думок та визначати твердження, на яких будуються аргументи.

Синтез визначається як поєднання елементів і частин для утворення цілого. У таксономії Блума синтез відповідає за здатність людини поєднувати елементи отриманих знань для утворення нових знань. Зазвичай цей процес включає рекомбінацію частин попереднього досвіду з новим матеріалом, реконструйованим у нове та інтегроване ціле [49]. Але результат синтезу не є вільним самовираженням, оскільки загалом розумовий процес людини розглядається в межах, визначених конкретними проблемами та ситуаціями.

Оцінка визначається як процес судження щодо цінності отриманих знань. Вона передбачає використання критеріїв, а також стандартів для оцінки точності, ефективності або задовільності інформації. Судження можуть бути як кількісними, так і якісними, а критеріями можуть бути визначені людиною, або надані. Етап оцінки є останнім процесом у таксономії Блума, оскільки вважається, що він знаходиться на пізній стадії складного процесу, який містить попередні рівні таксономії: знання, осмислення, застосування, аналіз і синтез. Оцінка є останнім етапом не тільки процесу когнітивної поведінки, але також і афективної поведінки, де цінності, симпатія та насолода є основними процесами.

2.5.2. Таксономія Андерсона та Кретвола

Робота Блума розглядає когнітивний процес навчання зі сторони поведінкової психології. З розвитком когнітивної науки та когнітивної психології, таксономія Блума була переглянута у роботі Лоріна Андерсона та Девіда Кретвола [50].

У переглянутій версії три категорії були перейменовані, і всі категорії були виражені як дієслова, а не іменники. Знання було змінено на запам’ятовування (remembering), осмислення стало розумінням (understanding), а синтез було перейменовано на створення (creation). Крім того, етап створення став найвищим рівнем у системі класифікації, перед яким йде етап оцінки. Переглянута версія ієрархії розумових процесів, яка складається з запам’ятовування, розуміння, застосування, аналізу, оцінки та створення представлена на рис. 2.4.

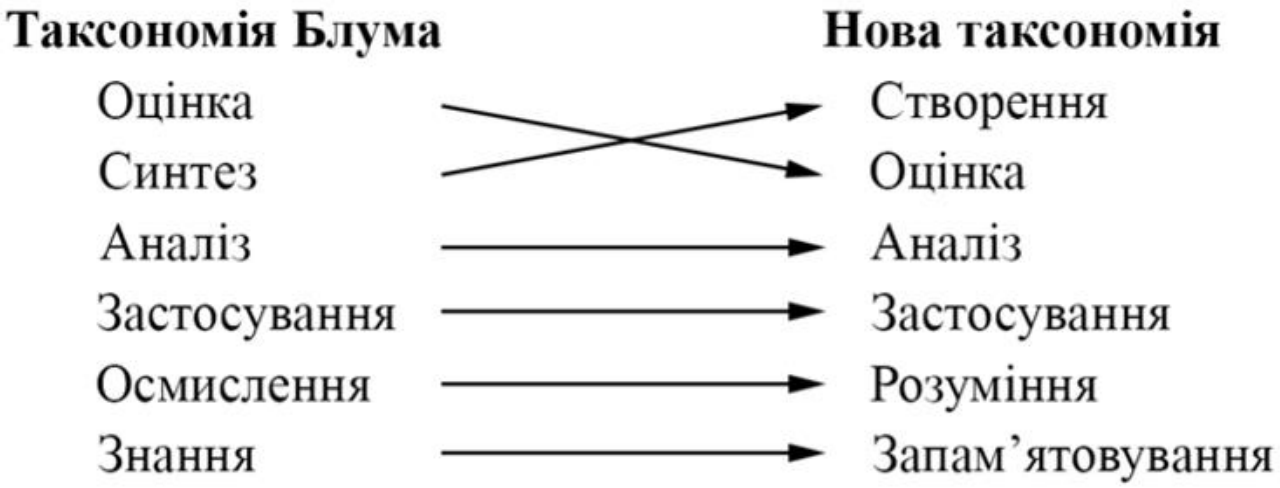

Рис. 2.4. Переглянута таксономія Блума

Рівень запам’ятовування відповідає за розпізнавання або пригадування знань з пам’яті. У цьому процесі пам’ять використовується для створення чи пошуку визначень, фактів або для отримання раніше вивченої інформації. Розумінням отриманої інформації та знань вважається здатність людини конструювати сенс із повідомлень, які надходять у вигляді усної, графічної, або

письмової інформації. Це здатність інтегрувати отриманні знання з існуючими когнітивними структурами. Застосування передбачає використання процедур для виконання вправ або розв'язання задач. Рівень аналізу відповідає за розбиття матеріалів або концепцій на частини, визначення того, як частини пов'язані одна з одною чи як частини пов'язані із загальною структурою чи метою. Визначення рівня оцінки не відрізняється від оригінального визначення у таксономії Блума. Він відповідає за винесення суджень на основі критеріїв і стандартів шляхом перевірки та критики.

Створення або об'єднання елементів у функціональне ціле знаходиться на останньому рівні таксономії. Воно відповідає за реорганізацію елементів у новий шаблон або структуру шляхом створення, планування або виробництва. Створення вимагає по-новому об'єднати частини або синтезувати частини у щось нове та відмінне. Цей процес є найскладнішою розумовою функцією в новій таксономії.

Окрім перегляду основних рівнів розумових процесів у таксономії Блума, Андерсон та Кретвол навели класифікацію відповідних когнітивних навичок, які використовуються людиною на кожному рівні таксономії.

На рівні запам'ятовування виділяють наступні когнітивні навички:

- розпізнавання – віднаходження знань у довготривалій пам'яті, що узгоджуються з поданим матеріалом;
- згадування – отримання відповідних знань з довготривалої пам'яті.

Рівень розуміння відповідає за наступні навички:

- інтерпретація – перехід від однієї форми представлення до іншої;
- знаходження прикладу – пошук конкретного прикладу концепції чи принципу;
- класифікація – визначення належності до категорії;
- підсумовування – виділення загальної теми або основних моментів;
- наведення висновків – створення логічного висновку з представленої інформації;
- порівняння – виявлення відповідності між двома ідеями;

– пояснення – побудова причинно-наслідкової моделі системи.

Рівень застосування:

– виконання – застосування інформації або знань до конкретного знайомого завдання;

– імплементація – застосування інформації або знань до конкретного незнайомого завдання.

Рівень аналізу:

– диференціація – виділення релевантних частин представленого матеріалу від нерелевантних;

– організація – визначення того, як знання взаємодіють між собою;

– виділення властивостей – визначення точки зору, базисів, що лежать в основі матеріалу.

Рівень оцінки:

– перевірка – виявлення невідповідностей або помилок у процесі;

– здатність до критики – виявлення невідповідностей між результатом і зовнішніми критеріями.

Рівень створення:

– продукування гіпотез – висування альтернативних гіпотез на основі критеріїв;

– планування – продумування процедури виконання певного завдання;

– продукування нового.

Таким чином ревізія оригінальної роботи Блума розширює класифікацію когнітивних процесів через опис когнітивних навичок для кожного рівня таксономії, які можна використовувати для побудови когнітивної моделі студента.

2.6. Методи дослідження когнітивних навичок

Методи дослідження когнітивних навичок необхідні для експериментального виділення розумових операцій з метою збору даних про ці

операції, їх аналіз, формулювання гіпотез, перевірки гіпотез, розробки теорій та їх застосування. У когнітивній науці методи дослідження когнітивних навичок для побудови когнітивних моделей можна характеризувати за двома категоріями: методи прямої реєстрації та непрямі методи.

Методи прямої реєстрації характеризуються використанням спеціалізованої апаратури для збору та обробки даних про когнітивні навички у мозку людини. До цієї категорії методів можна віднести експерименти над поведінкою людини та психобіологічні дослідження [51]. Під час експерименту дослідник маніпулює змінними, щоб побачити їх вплив на інші змінні. Наприклад дані про фіксацію очей під час процесу навчання людини відображають увагу та дозволяють дослідити процес зміни уваги, тому їх використовують для дослідження когнітивних процесів [52]. За допомогою психобіологічних досліджень дослідники вивчають взаємозв’язок між когнітивною діяльністю та церебральними подіями та ситуаціями. Для цього використовується сканування мозку людини, щоб встановити де і коли відбуваються когнітивні процеси в мозку [53].

До непрямих методів дослідження когнітивних процесів відносяться комп’ютерне моделювання та дослідження за допомогою штучного інтелекту [51]. Під час комп’ютерного моделювання за результатами ментальної поведінки людини програмуються та навчаються математичні моделі, результати яких апроксимовано відображають реальні когнітивні процеси людини.

Оскільки доступ до спеціалізованої апаратури обмежений, у даній роботі аналіз когнітивних процесів людини буде здійснений шляхом збору відповідних статистичних даних та навчання відповідної математичної моделі. Основним питання є вибір математичної моделі, яка задовольняє потребам поставленої задачі. Аналіз предметної області дозволив виділити основні методи для побудови математичних моделей, такі як нейронні мережі та випадковий ліс.

2.6.1. Нейронні мережі

Нейронні мережі являють собою універсальний механізм обчислень, основна ідея якого полягає в об’єднанні простих елементів для вирішення складних завдань. Ці одиниці називаються нейронами та об’єднуються у шари, які виконують певні ролі. Основна архітектура нейронної мережі представлена на рис. 2.5.

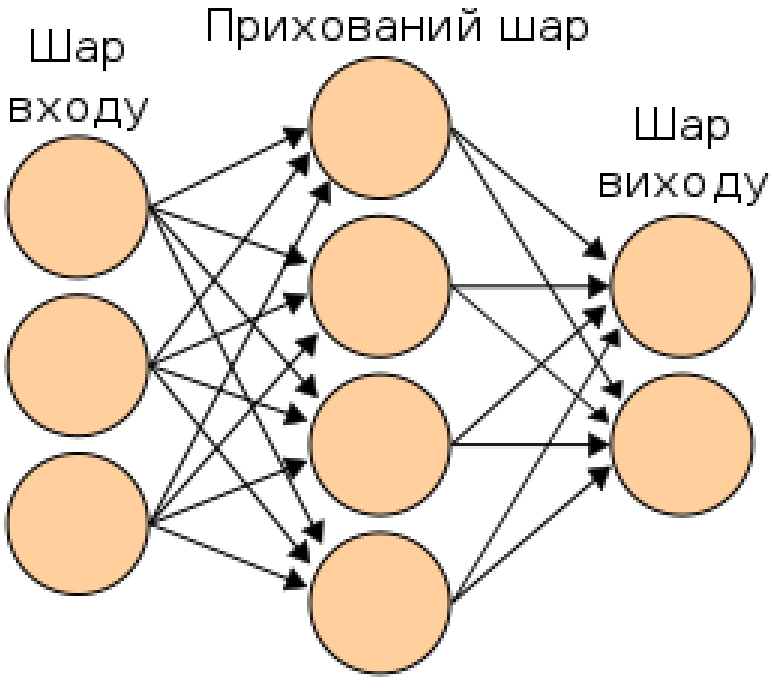

Рис. 2.5. Схема нейронної мережі

Нейронна мережа складається з вхідного та вихідного шарів та одного або кількох прихованих шарів. Кожен нейрон у шарах нейронної мережі має входи та виходи, через які він поєднується з нейронами наступного шару.

Схема функціонування нейронної мережі зображена на рис. 2.6. Кожен нейрон отримує виходи нейронів попереднього шару. Далі вони помножуються на значення ваги зв’язку w_i та підсумовуються. Вихідний результат нейрону визначається функцією активації, яка перетворює суму зважених входів нейрону та передається до нейронів наступного шару або у вихідний шар. Процес

навчання нейронної мережі полягає в пошуку ваг зв’язків між нейронами, які дають бажаний результат. Алгоритми навчання коригують зміною ваг зв’язків між нейронами відповідно до мінімізації відмінностей між реальними значеннями цільових змінних і розрахованими мережею.

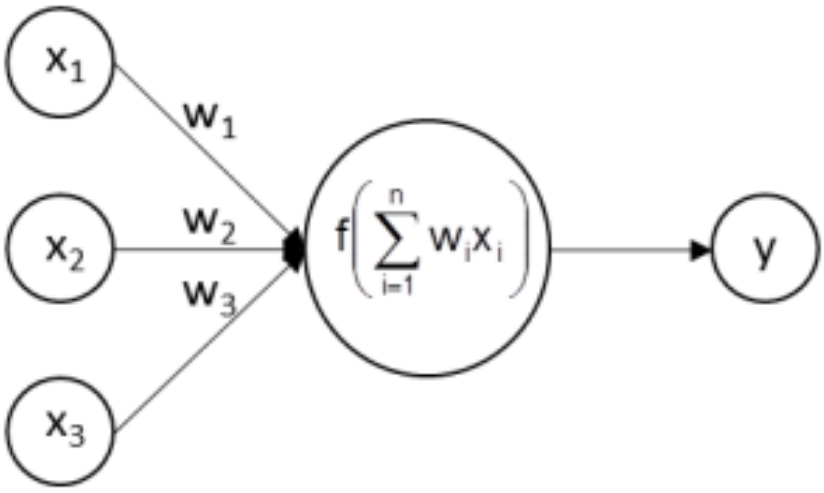

Рис. 2.6. Схема функціонування нейронної мережі

Таким чином, архітектура нейронної мережі надає можливість створювати широкий спектр систем від систем класифікації та регресії до систем розпізнавання образів з великим числом характеристик.

2.6.2. Випадковий ліс

Іншим підходом до задач класифікації та регресії є використання алгоритму сімейства дерев рішень «випадковий ліс». Дерево рішень являє собою класифікацію або модель регресії в деревоподібній структурі. Кожен вузол дерева представляє ознаку з вхідного простору, кожна гілка – рішення, а кожен листок у кінці гілки – відповідне вихідне значення (рис. 2.7).

Рис. 2.7. Дерево рішень

Подібно до нейронних мереж, дерево будується за допомогою процесу навчання з використанням навчальних даних. Процес навчання створює дерево крок за кроком відповідно до важливості вхідних функцій у контексті конкретної програми. Використовуючи навчальні дані, спочатку визначають найбільш важливу ознаку серед усього набору ознак. Відповідно до отриманого значення, навчальні дані розбиваються на підмножини. Для кожної отриманої підмножини визначається друга найважливіша ознака і створюються нові підмножини. Цей процес повторюється для кожної отриманої підмножини, поки не будуть знайдені листкові вузли у всіх гілках дерева.

Алгоритм випадкового лісу будує різні дерева рішень на основі одного і того ж джерела навчальних даних. Кожне дерево рішень створюється з різною, випадково обраною підмножиною навчальних даних і з випадково вибраною підмножиною ознак у кожному вузлі. Таким чином, збудовані дерева рішень не мають доступу до повних навчальних даних. Набір дерев у алгоритмі створює “ансамбль” дерев, на основі якого обчислюється сукупний результат більшістю голосів у разі класифікації або усередненням у разі регресії (рис. 2.8). Це компенсує потенційні помилки окремих дерев у лісі, тому модель менш ймовірно дасть результати, що відрізняються від реальних значень.

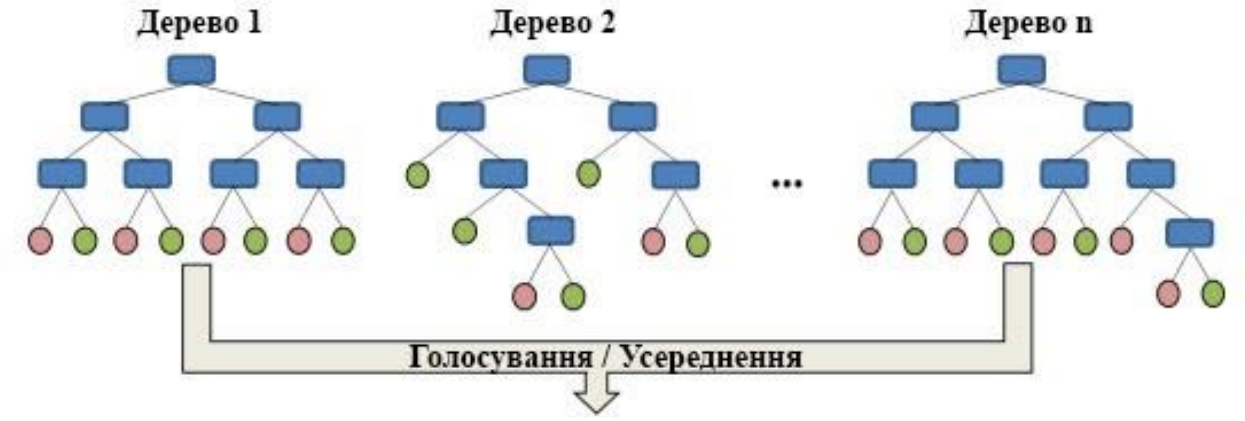

Рис. 2.8. Схема алгоритму випадкового лісу

2.6.3. Порівняння алгоритмів

Нейронні мережі та випадковий ліс описують два різні підходи для створення високоякісних систем класифікації та регресії. На практиці обидва підходи широко й успішно використовуються у різних сферах. Але важливою особливістю машинного навчання є те, що жоден алгоритм не працює найкраще для всіх можливих сценаріїв. Необхідно порівняти ці алгоритми та обрати алгоритм, який відповідає поставленій задачі.

Продуктивність у сенсі якості результату класифікації є важливим фактором для визначення алгоритму. Наприклад алгоритми, які вміють фіксувати лише лінійні зв’язки між даними, не придатні для використання для класифікації даних з великою кількістю нелінійних зв’язків. Нейронні мережі та випадковий ліс мають здатність моделювати як лінійні, так і нелінійні зв’язки, але завдяки своїй гнучкій архітектурі, нейронні мережі мають більший потенціал з боку критерію продуктивності.

Надійність це критерій, який визначає здатність моделі давати задовільні результати не лише з даними, які використовувались для її навчання, але й з новими даними. Така проблема має назву перенавчання. У разі перенавчання модель не забезпечує належної надійності, яка відображає її узагальненість. Для розв’язання цієї проблеми нейронні мережі та випадковий ліс пропонують різні

підходи. У нейронних мережах можна змінювати кількість прихованих шарів та нейронів у цих шарах або використовувати методи оптимізації ваг під час процесу навчання. Випадковий ліс дає можливість налаштувати кількість дерев або максимальний розмір чи глибину окремого дерева. Таким чином, обидва підходи надають можливості для розв'язання проблем із перенавчанням, але нейронним мережам приписують вищу чутливість до вхідних даних, що, як правило, призводить до вищого ризику відхилень у випадку «нетипових» входів.

Витрати часу на навчання моделі також є важливим критерієм при виборі алгоритму. У цьому контексті навчання нейронних мереж є дуже трудомістким. Щоб знайти найкращу модель, необхідно розрахувати та протестувати декілька варіантів. Для цього варіюються різні параметри системи, такі як кількість шарів або нейронів. Для кожної комбінації параметрів необхідно розрахувати та оцінити повну модель. Чим більше існує параметрів, тим більше комбінацій потрібно перевірити, що призводить до значних витрат часу. Крім самого процесу навчання, необхідна також велика підготовча робота, щоб привести вхідні дані в необхідну форму. Вони повинні бути, наприклад, у числовому форматі та нормалізованими.

Випадкові ліси вимагають набагато менше підготовки вхідних даних. Вони можуть обробляти двійкові ознаки, категоріальні ознаки, а також числові ознаки, і не потребують їх нормалізації. Випадкові ліси швидко навчаються та оптимізуються відповідно до їхніх параметрів. Таким чином, час навчання випадкового лісу є порівняно низьким з нейронними мережами. Крім того, випадковий ліс можна навчати з відносно невеликою кількістю даних. Нейронним мережам зазвичай потрібно більше даних, щоб досягти того ж рівня точності [54].

Таким чином, після порівняння двох методів, можна обрати відповідну математичну модель для даної роботи. Оскільки обчислювальні потужності для побудови моделі є обмеженими та ми не маємо можливості зібрати досить великий об'єм даних, а також беручи до уваги, що під час дослідження когнітивних процесів цілі експериментів можуть змінюватись і може виникати

необхідність швидко перенавчати модель, використання випадкового лісу є виправданим для даної задачі.

2.7. Висновки

У цьому розділі були розглянуті когнітивні процеси, які виникають при розумовій діяльності людини, а також розглянуті когнітивні підходи до дослідження процесу прийняття рішень. Були розглянуті та порівняні методи класифікації когнітивних навичок людини, у результаті чого для виділення когнітивних навичок студента у системі навчання була обрана ревізія таксономії Блума, яка розроблена Адерсоном та Кретволом, оскільки вона розширює оригінальну таксономію Блума та надає більш детальну класифікацію навичок.

Для дослідження когнітивних навичок був обраний непрямий підхід з використанням комп’ютерних моделей. В результаті аналізу методів для побудови моделей вивчення когнітивних навичок був обраний алгоритм випадкового лісу, оскільки він надає змогу швидко навчати моделі з невеликим набором даних з досить високою точністю, що відповідає вимогам поставленої задачі.

РОЗДІЛ 3
РОЗРОБКА ТА ОБҐРУНТУВАННЯ МЕТОДУ ДОБОРУ НАВЧАЛЬНИХ МЕТОДИК ДЛЯ СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ

3.1. Архітектура системи дистанційного навчання

Як було зазначено, метою даної роботи є представлення методу добору навчальних методик на основі аналізу когнітивної моделі студентів, які навчаються у системі дистанційної освіти. Архітектура системи дистанційної освіти складається з клієнтської, серверної частин та сховища великих даних (рис 3.1).

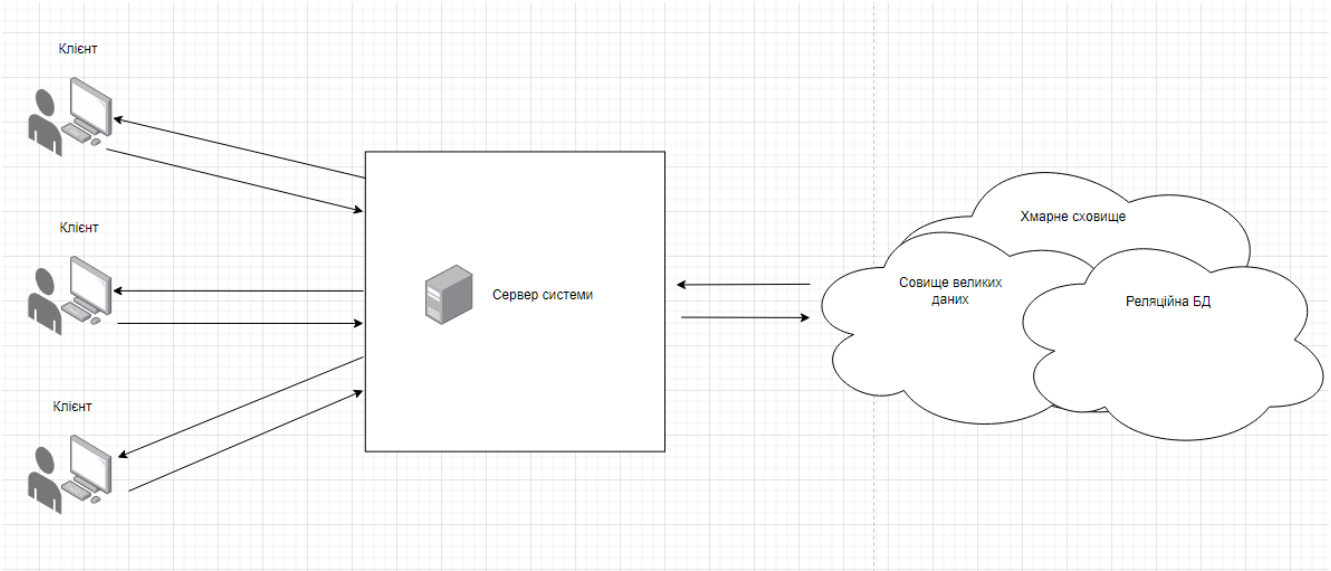

Рис. 3.1. Архітектура системи дистанційної освіти

Клієнтська сторона надає серверу інформацію про переглянуті студентом матеріали, а також відомості про задачі, що виконував студент, такі як кількість спроб, рішення тощо. Сервер системи отримані дані відправляє в сховище великих даних. Воно поділяється на дві частини: власне сховище великих даних та додаткова реляційна база даних, за допомогою якої буде відбуватись аналітика даних. Аналітика впливає на характер віддачі сервером контенту.

До сховища великих даних висуваються наступні вимоги:

– повинне вміти зберігати звичайні дані у вигляді реляційних таблиць або колекцій об’єктів;

– база даних повинна вміти зберігати графи, наприклад у формі матриці суміжності;

– повинна бути можливість вбудовувати код в систему великих даних для того, щоб виконувати аналітичну роботу саме на стороні великих даних, щоб не навантажувати сервер системи.

Процеси когнітивного моделювання та його аналізу з подальшим отриманням результатів для висновків щодо добору навчальних методик повинні відбуватися на стороні великих даних.

Система надає студенту навчальні курси, завдяки яким студент може вивчати окремі дисципліни та теми. Кожен курс має контрольні засоби перевірки знань, такі як тести, за допомогою яких ми можемо доповнювати когнітивну модель студента та робити висновки щодо зміни або добору навчальних методик.

3.2. Опис функціоналу системи

З метою перевірки роботи модуля, що корегує навчальну методику, був розроблений застосунок онлайн системи навчання. Для розробки використовувалися мова програмування Python та фреймворк Django, а також фреймворк Vue.js мови програмування JavaScript. Система має реалізовувати наступний функціонал:

– реєстрація нових користувачів у систему;

– надавати користувачеві список курсів системи;

– надавати користувачеві змогу записатися на курс та мати доступ до навчальних матеріалів курсу;

– надавати користувачеві доступ до завдань курсу та надавати рішення цих завдань;

– збирати статистичні дані про проходження завдань студентами;

– мати інтуїтивний і зрозумілий користувацький інтерфейс.

Користувач системи отримує доступ до навчальних матеріалів через список курсів, який зображений на рис. 3.2.

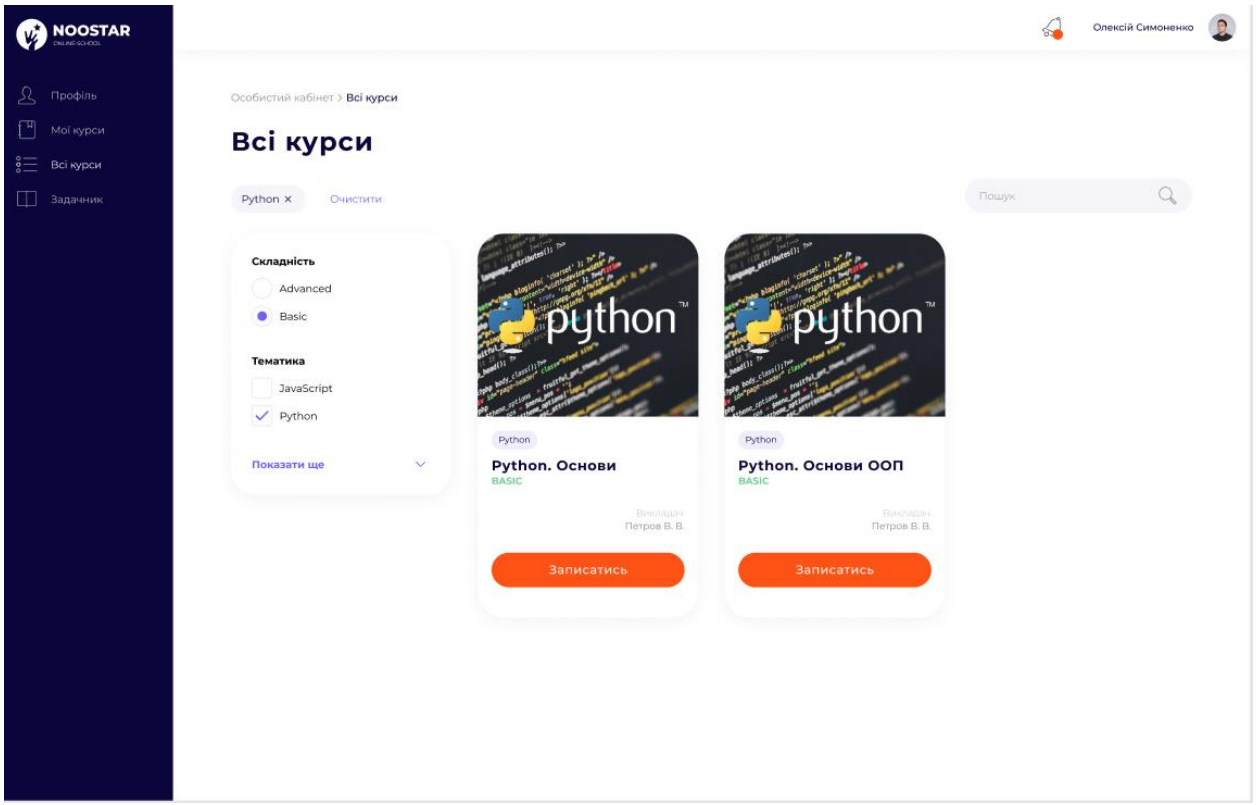

Рис. 3.2. Сторінка «Всі курси»

На сторінці «Всі курси» користувач може переглянути список доступних курсів у системі, їх назву, складність, інформацію про теми курсу та викладача, який веде цей курс, а також зареєструватися на курс. У лівому боці сторінки знаходиться список фільтрів, за якими користувач може уточнювати інформацію про курси. Список курсів можна фільтрувати за складністю та тематикою. За допомогою рядка пошуку у верхній частині сторінки можна шукати курси за назвою.

Для функціонування описаної сторінки було створене API для отримання списку курсів. Запит надається методом GET, відповідь у вигляді списку курсів надходить у форматі JSON. Рядок запиту на сервер виглядає наступним чином:

– /api/1.0/course/all.

Описане API має наступні вхідні параметри:

– params – об’єкт з набором параметрів фільтрації, має тип даних JSON.

Формат вихідних даних:

– courses – об’єкт зі списком даних про курси, має тип даних JSON.

Нижче наведено код реалізації, виконаний з використанням фреймворку Django мови програмування Python.

```
class CourseAPI(generics.GenericAPIView):
   permission_classes = [permissions.AllowAny]
   serializer_class = CourseSerializer
   queryset = Course.objects.all()
   filter_backends = [filters.DjangoFilterBackend]
   filterset_class = CourseFilter
   def get(self, request):
        return Response(
        self.get_serializer(
        self.filter_queryset(
        self.get_queryset()
        ), many=True).data, status=200)
```

Кожен курс надає користувачеві доступ до списку практичних задач з тем, які він охоплює. У лівому боці інтерфейсу користувача знаходиться навігаційне меню, через яке можна отримати доступ до списку задач та іншого функціоналу системи. Інтерфейс списку задач зображений на рис 3.3.

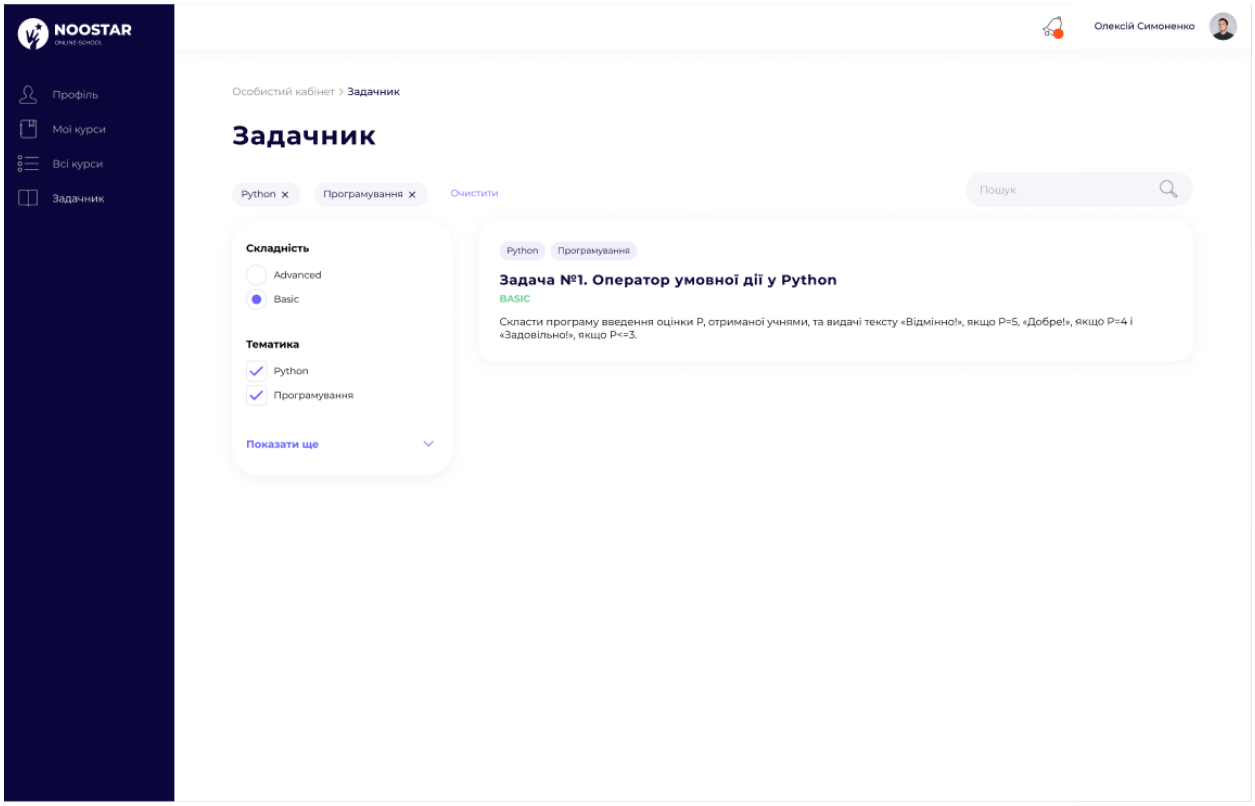

Рис. 3.3. Сторінка «Задачник»

На сторінці «Задачник» можна переглянути список усіх доступних задач, їх назву, складність, тематику та стислу умову задачі. За допомогою рядка пошуку можна шукати конкретні задачі, а за допомогою фільтрації можна відфільтрувати список задач за складністю або тематикою. На цій сторінці користувачеві відображаються задачі з курсів, на які він зареєстрований.

Щоб отримати список задач для сторінки було розроблене API з наступною рядком запиту:

– /api/1.0/task/all.

Описане API має наступні вхідні параметри:

– params – об’єкт з набором параметрів фільтрації, має тип даних JSON.

– is_public – прапорець, що відповідає за видачу списку задач з публічним рівнем доступу, має тип даних boolean.

Формат вихідних даних:

– tasks – об’єкт зі списком даних про задачі, має тип даних JSON.

Код реалізації API виглядає наступним чином:

```
class GetTasksAPI(generics.GenericAPIView):
    serializer_class = TaskSerializer
    queryset = Task.objects.all()
    filter_backends = [filters.DjangoFilterBackend]
    filterset_class = TaskFilter

    def get(self, request):
        if 'is_public' in request.GET:
            value = request.GET['is_public'].upper()
            is_public = True if value == 'TRUE' else False
            if is_public:
                return Response(
                    self.get_serializer(
                    self.get_queryset().filter(
                    is_public=True),many=True
                ).data,status=200)
            return Response(
                self.get_serializer(
                self.get_queryset().filter(
                is_public=False), many=True
                ).data,status=200)
        else:
            return Response(
                self.get_serializer(
                self.filter_queryset(
                self.get_queryset()), many=True
                ).data,status=200)
```

Щоб розпочати розв’язання задачі, користувач може натиснути на картку задачі зі списку, після чого йому відкриється нова сторінка з описом обраної задачі (рис. 3.4).

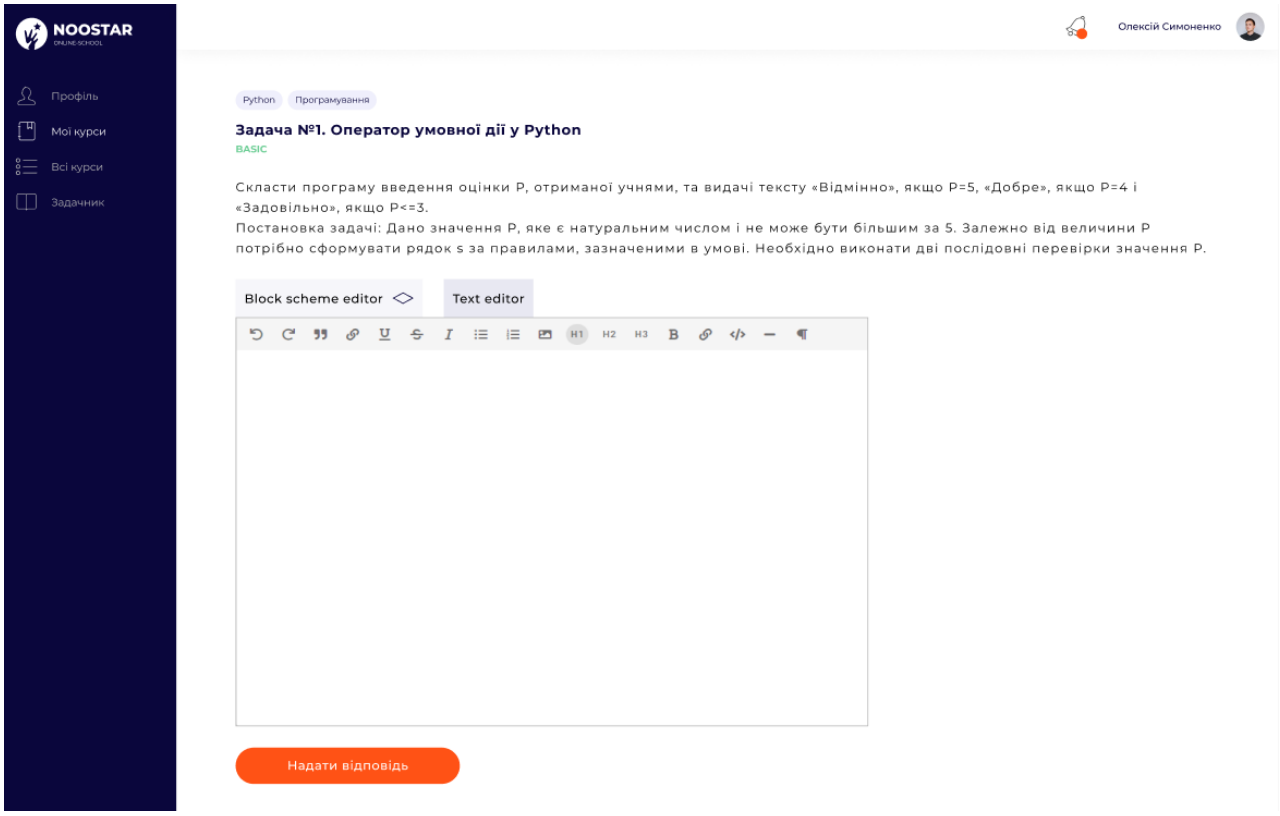

Рис. 3.4. Сторінка задачі

На цій сторінці дублюються дані про назву, складність та тематику задачі, а також надається повна умова задачі з вкладеними медіафайлами за необхідності. Щоб надати відповідь про задачу, користувач повинен написати відповідь у текстовому редакторі, який знаходиться після умови задачі. Щоб відправити відповідь на завдання, користувач повинен натиснути кнопку «надати відповідь», після чого відповідь відправляється на опрацювання сервером.

Відповіді на задачі користувач також може надавати у вигляді блок-схем. Для цього на сторінці задачі користувач має змінити тип редактора, натиснувши на іконку блок-схеми у формі редактора (рис. 3.5).

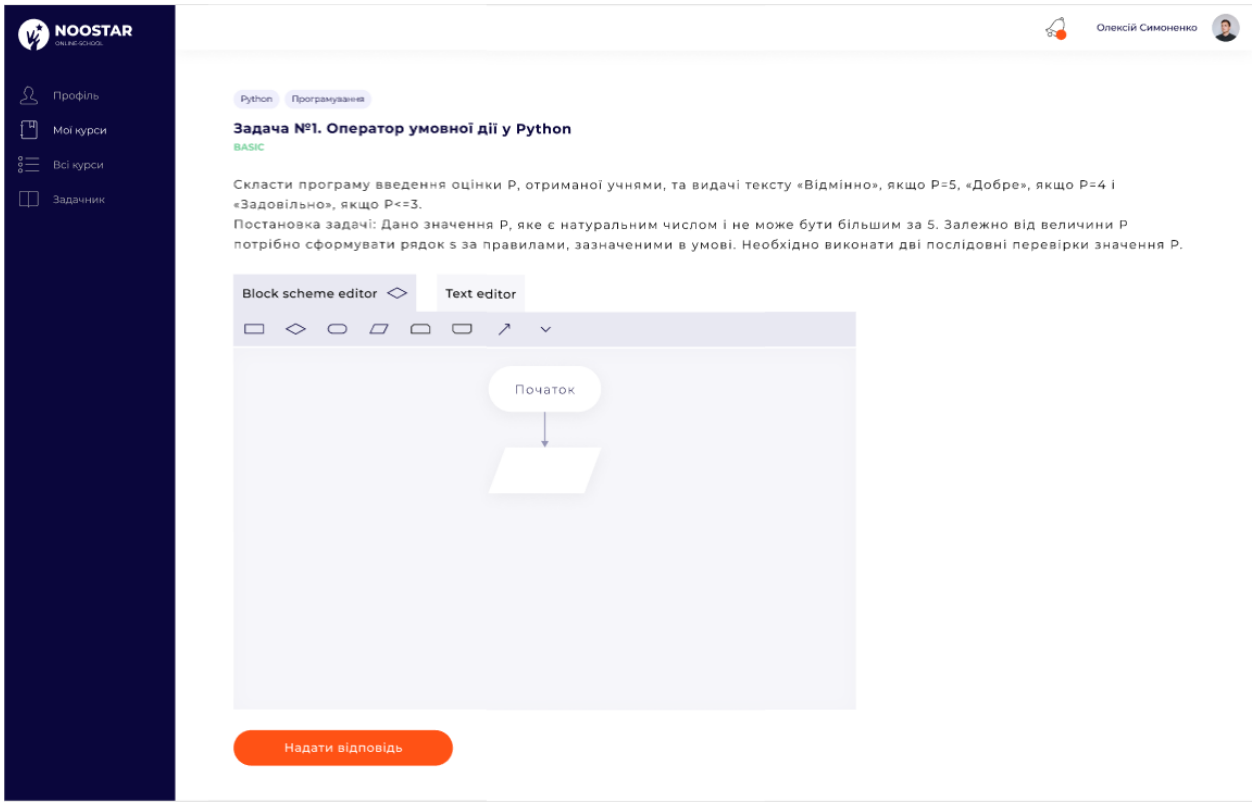

Рис. 3.5. Редактор блок-схеми алгоритму

За допомогою редактора блок-схем користувач має змогу будувати блок-схеми алгоритмів. Для цього у редакторі є палітра, яка знаходиться у верхній частині редактора, у якій можна обрати основні фігури, необхідні для побудови блок-схем. Щоб побудувати фігуру, користувачу необхідно натиснути на її іконку у палітрі, після чого натиснути на вільному місці на канві редактора. Побудовані фігури можна масштабувати та вільно переміщувати.

Щоб відправити відповідь на сервер, використовується API, яке за допомогою метода POST відправляє дані з форми редактора та має наступний рядок виклику:

– /api/1.0/task_result/create.

Описане API має наступні вхідні параметри:

– task_uuid – унікальний ідентифікатор задачі, має тип даних UUID.

– student_uuid – унікальний ідентифікатор студента, має тип даних UUID.

– answer – об’єкт з наданою відповіддю на задачу, має тип даних JSON.

– time – час витрачений на розв’язання задачі, має тип даних bigint.

Формат вихідних даних:

– status – об’єкт з інформацією про результат запиту, має тип даних JSON.

Код реалізації API виглядає наступним чином:

```
class TaskResultAPI(generics.GenericAPIView):
    serializer_class = TaskResultSerializer
    queryset = TaskResult.objects.all()

    def post(self, request):
        if 'task_uuid' not in request.data:
            return createResponseWithStatus(
            'task_uuid is required', 400)
        if 'student_uuid' not in request.data:
            return createResponseWithStatus(
            'student_uuid is required', 400)
        if 'answer' not in request.data:
            return createResponseWithStatus(
            'answer is required', 400)
        if 'time' not in request.data:
            return createResponseWithStatus(
            'time is required', 400)
        try:
            process_answer(
            task_uuid, student_uuid, answer, time)
            return createResponseWithStatus(
            'Task answer successfully processed', 200)
        except Exception as e:
            return createResponseWithStatus(
            f'Something went wrong {e}', 400)
```

3.3. Опис набору даних

Оскільки відсутній доступ до спеціальної апаратури, яка дозволяє досліджувати когнітивні процеси, у даній роботі використовується методологія непрямого дослідження. Цей підхід дозволяє отримати наближені результати, як і при дослідженні зі спеціалізованою апаратурою, але є трудомісткім, оскільки вимагає організації збору статистичних даних.

Аналіз когнітивних навичок студентів системи з подальшим висновком щодо добору навчальних методик здійснюється за допомогою аналізу проходження студентами тестових матеріалів. Для того, щоб тестові матеріали надавали уявлення про когнітивні навички, необхідні для вирішення завдання, був створений набір даних, у якому містяться оцінки когнітивних навичок для кожного завдання, які надані експертами [55, 56]. Окрім уявлення про когнітивні навички для вирішення завдання, нам необхідно мати уявлення про когнітивні навички студента, які необхідні для засвоєння окремих тем, які охоплює дисципліна та знання яких перевіряється тестовими матеріалами. Перелік когнітивних навичок був обраний згідно з таксономією Андерсона та Кретвола. У табл. 3.1 наданий детальний опис наборів даних оцінок когнітивних навичок для тестових завдань та тем, які вони перевіряють.

Таблиця 3.1

Таблиця опису набору даних експертних оцінок когнітивних навичок для тестових завдань та тем

Назва стовпців	Опис
1	2
Task/theme number/name	Номер тестового завдання або назва теми
Expert ID	Унікальний ідентифікатор експерта

Продовження таблиці 3.1

1	2
Recognition, Recalling, Interpreting, Exemplifying, Classifying, Summarizing, Inferring, Comparing, Explaining, Executing, Implementing, Differentiating, Organizing, Attributing, Checking, Critiquing, Generating, Planning, Producing	Експертні оцінки когнітивних навичок розпізнавання, згадування, інтерпретації, знаходження прикладу, класифікації, підсумовування, наведення висновків, порівняння, пояснення, виконання, імплементації, диференціації, організації, виділення властивостей, перевірки, здатності до критики, продукування гіпотез, планування, продукування нового від 0 до 1

Набір даних формувався на основі групи експертів з інформаційних технологій та освіти у складі 10 осіб. Для оцінки були взяті тестові матеріали з теми «імперативна парадигма програмування». Отриманий набір даних надає уявлення про те, як повинні виглядати когнітивні навички людини для успішного виконання кожного з завдань тестових матеріалів та успішного засвоєння тем дисципліни, за думкою обраної групи експертів, завдяки ефекту мудрості натовпу.

Мудрість натовпу – теорія, яка описує, що великі групи людей разом розумніші за окремих експертів, коли справа доходить до розв’язання проблем, прийняття рішень, інновацій та прогнозування. Теорія базується на ідеї, що точка зору окремої людини може бути упередженою, тоді як отримання середнього рівня знання натовпу може призвести до усунення упередженості або шуму, щоб отримати чіткіший і послідовний результат. Докази того, що середня оцінка групи може бути точнішою, ніж оцінки індивідуального експерта, уперше з’являються у роботі Френсіса Гальтона [57], а Джеймс Шуров'єскі підтверджує цю теорію, наводячи приклади з фондових ринків, політичних виборів та

вікторин [58]. Таким чином отримані оцінки експертів були усередненні для отримання еталонних оцінок когнітивних навичок, що мусить мати студент, для кожного з тестових завдань, та тем дисципліни.

Наступним кроком для аналізу необхідно побудувати модель того, як окрема група студентів виконують завдання з тестових матеріалів. З цією метою був розроблений другий набір даних, який містить дані про виконання групою обраних студентів тестових завдань. Ці дані містять інформацію про теми, які перевіряються завданням, результат виконання завдання, кількість спроб на його виконання та когнітивні навички студента за результатами його виконання. Для оцінки когнітивних навичок групою експертів була розроблена система штрафів під час виконання тестових завдань, яка обчислювала оцінки когнітивних навичок студента базуючись на еталонному значені [56]. У табл. 3.2 наданий детальний опис набору даних та його стовпців.

Таблиця 3.2

Таблиця опису набору даних результатів тестування студентів

Назва стовпця	Опис
1	2
Task number	Номер завдання
Student ID	Унікальний ідентифікатор студента
Opportunities	Кількість спроб виконання завдання від 1 до 5
Result	Результат виконання завдання, може набувати значень «0» – не склав, та «1» – склав
Theme 1	Назва першої теми, яку охоплює завдання
Theme 2	Назва другої теми, яку охоплює завдання

Продовження таблиці 3.2

1	2
Theme 3	Назва третьої теми, яку охоплює завдання
Recognition, Recalling, Interpreting, Exemplifying, Classifying, Summarizing, Inferring, Comparing, Explaining, Executing, Implementing, Differentiating, Organizing, Attributing, Checking, Critiquing, Generating, Planning, Producing	Обчислені когнітивні навички студента з результатів виконання завдання відносно еталонних значень з використанням розробленої системи штрафів від 0 до 1.

Результати тестувань були отримані від групи студентів у складі 15 осіб. Тестування проводилися з теми «імперативна парадигма програмування». За отриманими результатами ми можемо аналізувати успішність результатів розв’язання задач, що дає нам уявлення про те, яка оптимальна комбінація когнітивних навичок повинна бути у студента для вирішення кожного з завдань тестових матеріалів.

3.4. Алгоритм добору навчальних методик

Процес навчання студента у системі дистанційного навчання відбувається наступним чином. Студент записується на курс з дисципліни, яка охоплює перелік тем, які студент має засвоїти після його проходження. Курс викладається за навчальною методикою, що може бути як груповою, так і індивідуальною для кожного студента.

Навчальна методика курсу – це набір заходів, таких як лекції, лабораторні, практичні, самостійні завдання з вивчення матеріалів курсу. В ході вивчення курсу за певною методикою студент має набути певні знання з окремих тем та підвищити власні когнітивні навички. Знання з окремих тем формують

онтологію знань студента, яка необхідна для застосування отриманих знань на практиці та, за необхідності, вивчення суміжних курсів. Підвищення когнітивних навичок впливає на здатність студента до критичного мислення, адекватного розуміння навколишнього світу та можливості вивчення інших наук та отримання інших практичних навичок, які не стосуються поточного курсу, але вимагають певного рівня інтелекту.

У певних випадках початково застосована навчальна методика може не давати бажаного освітнього ефекту або, якщо студент навчається швидше та якісніше за очікування викладача, то може виникнути потреба змінити методику викладання. Також може виникнути потреба у з'ясуванні того, які зміни відбуваються з онтологією знань студента та його когнітивними навичками, щоб розуміти як ефективніше змінити навчальну методику. Тому освітня онлайн система має постійно збирати проміжні дані про виконання студентом практичних форм освітнього навантаження та час від часу контролювати поточний стан онтології знань студента та його когнітивних навичок через механізм контрольних заходів, таких як тестування, контрольна робота тощо. Матеріали цих контрольних заходів як і матеріали з відповідного курсу мусять мати оцінку когнітивних навичок та перелік тем, наявність знань про які намагається перевірити система.

Після виконання кожного контрольного заходу, система отримує відомості про успішність чи неуспішність та кількість спроб, за якими вираховує зміни у когнітивних навичках, та фіксує їх за необхідності в онтології знань студента. Але на добір складу навчальної методики безпосередній вплив здійснюють лише когнітивні навички. Тому, якщо порівняти швидкість їх зростання протягом різних періодів вивчення курсу, можна зробити висновок щодо впливу певної навчальної методики на ці навички. Інакшими словами можна адаптивно змінювати навчальну методику і контролювати ефективність її застосування до окремого індивіда, шляхом періодичного діагностування його когнітивних навичок.

3.5. Модель оцінки когнітивних навичок студента

У якості моделі, на підставі якої робиться висновки щодо добору навчальних методик, була побудована та навчена модель класифікації з використанням алгоритму випадкового лісу.

Об’єктом класифікації є результат виконання студентом контрольних або проміжних завдань з дисципліни за окремою навчальною методикою. На виконання завдання студентові надається лімітована кількість спроб, а результат визначається станом «склав» або «не склав». Таким чином вихідними є класи, отримані комбінацією станів завдання «склав» або «не склав» та з якої спроби був отриманий стан завдання.

Ознаками класифікації є обчислені когнітивні навички студентів у результаті виконання завдання та теми, які охоплюються завданнями тестових матеріалів. Кожне завдання у розробленій моделі охоплює три теми з набору тем дисципліни. Даний набір ознак дозволяє класифікувати результати тестування окремих студентів та робити висновки щодо покращення навчальної методики.

Щоб отримати уявлення про початкові когнітивні навички, студенти проходять вступне тестове завдання дисципліни. Отриманий набір даних використовується для навчання моделі. Після цього студенти починають вивчати матеріали дисципліни за стандартною навчальною методикою курсу, за якими надається наступний тестовий проміжний контроль. Результати тестування використовуються для прогнозування успіхів студенту за попередньою моделлю, а після цього порівнюються з реальними результатами. На основі порівняння робиться висновок щодо ефективності обраної методики для кожного студента. У результаті висновків навчальна методика знову змінюється, модель доповнюється отриманими даними і процедура повторюється до моменту задовільних результатів.

Запропонована методика була застосована на наборі даних результатів тестувань студентів. На рис. 3.6 представлений графік прогнозованих успіхів окремого студента у виконанні окремого завдання на заданий набір тем, а на рис.

3.7 представлені реальні результати, отримані студентом під час виконання цього завдання.

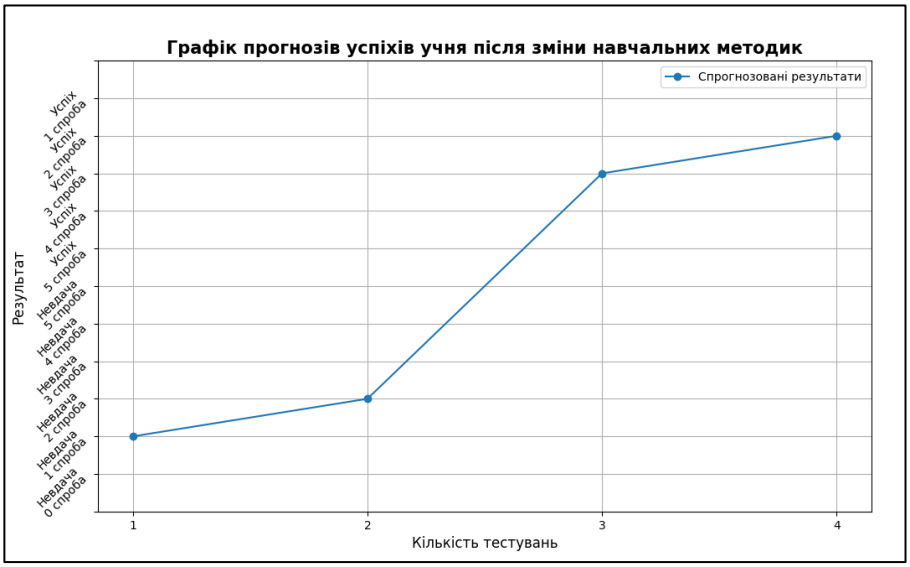

Рис. 3.6. Графік прогнозів успіхів студента після зміни навчальних методик

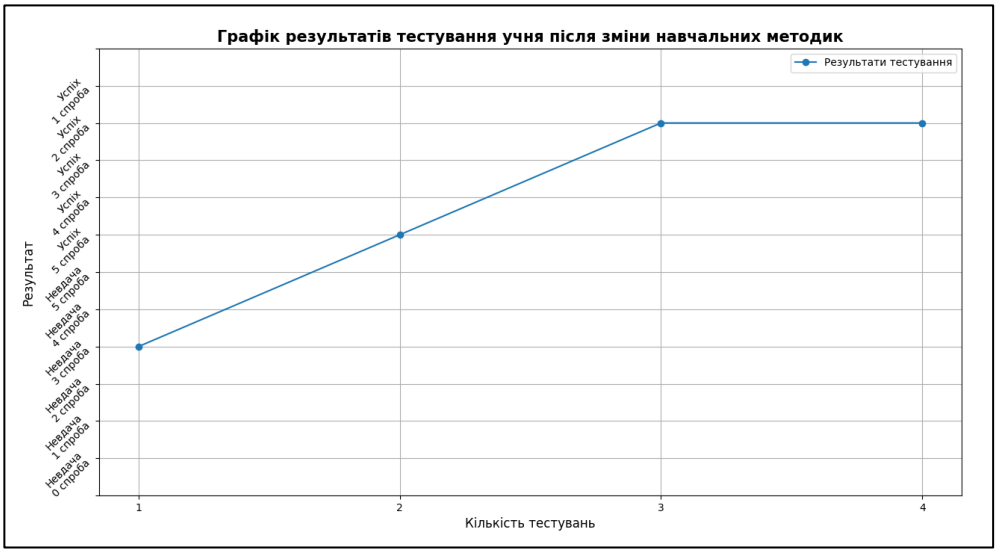

Рис. 3.7. Графік результатів тестування студента після зміни навчальних методик

Після кожного тестування результати моделі порівнювались с реальними результатами та робився висновок щодо покращення навчальної методики.

Тенденцію ефективності добору навчальних методик показує графік порівняння результатів моделі та реальних результатів, зображений на рис. 3.8.

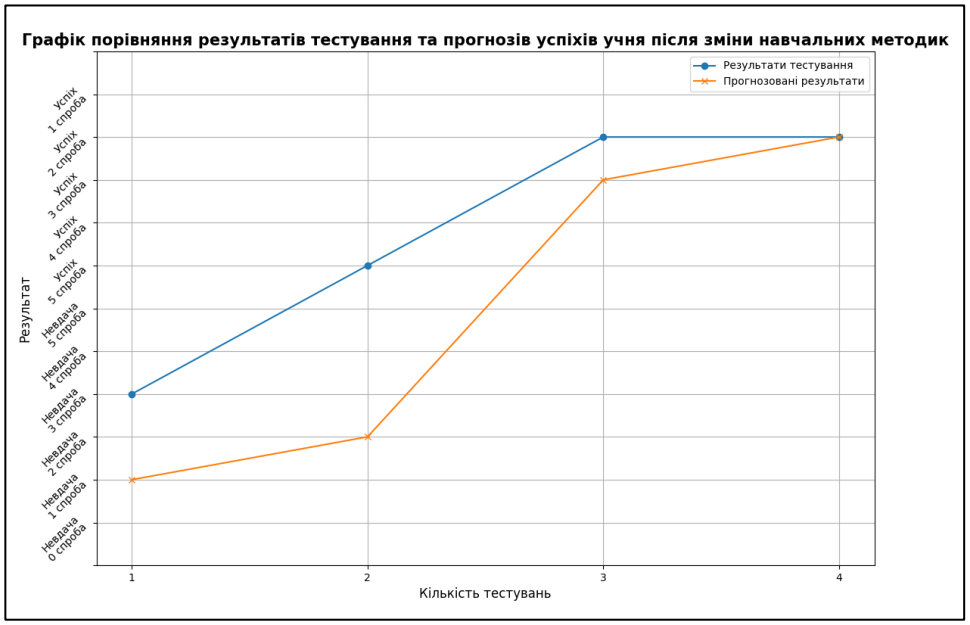

Рис. 3.8. Графік порівняння результатів тестування та прогнозів успіхів студента після зміни навчальних методик

Як можна бачити на рис. 3.4, після першого тестування прогнозований та реальні результати показують невдачу у виконанні студентом поставленого завдання. З цього можна зробити висновок, що початкова методика навчання не є ефективною і потребує змін. Після корегування навчальних методик на наступному проміжному тестуванні за прогнозованими результатами студент все ще не мав скласти завдання успішно, але реальні результати показують, що студент використав усі спроби, але виконав завдання успішно. На основі цього можна зробити висновок, що зміна навчальної методики для цього студента покращила його когнітивні навички та його результат з даних тем. Після доповнення навчальної методики, прогнозований та реальний результат показали успіх з невеликою різницею у спробах, що означає, що модифікована навчальна методика значно покращила опанування матеріалу студентом. Подальші вдосконалення методики не надали результатів покращення. Причиною цього може бути або досягнення межі можливостей когнітивної

системи студента, або відсутність мотивації досягнення максимального результату або комбінація цих факторів.

3.6. Висновки

У даному розділі була описана архітектура системи дистанційного навчання, а також наведений опис розробленого фрагмента освітньої онлайн системи, створеної за допомогою фреймворків Djsngo та Vue.js, для якого була розроблений метод добору навчальних методик.

Був запропонований метод добору навчальних методик на основі аналізу когнітивних навичок студента, обчислених за результатами виконання контрольних заходів дисципліни. З цією метою були описані набір даних з експертними оцінками когнітивних навичок для кожного завдання з контрольних матеріалів та тем, що охоплюють ці завдання, а також набір даних про результати виконання групою студентів тестових завдань з обрахованими оцінками когнітивних навичок на основі системи штрафів.

Отримані набори даних використовувались для навчання моделі прогнозування успіхів студента, на підставі аналізу яких приймалось рішення про зміну або доповнення навчальної методики. Отримані результати показують ефективність запропонованого методу, оскільки вони показують покращення результатів контролю знань студента та відповідно зростання його когнітивних навичок.

РОЗДІЛ 4
ЕКОНОМІЧНИЙ РОЗДІЛ

4.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Вхідні дані:

1. Передбачуване число операторів – 830.
2. Коефіцієнт складності програми – 1,2.
3. Коефіцієнт корекції програми в ході її розробки – 0,2.
4. Годинна заробітна плата програміста, грн/год – 120.
5. Коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі – 1,2.
6. Коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності – 1,1.
7. Вартість машино-години ЕОМ, грн/год – 30.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t=t_o+t_u+t_a+t_n+t_{отл}+t_д, \text{ людино-годин}, \quad (4.1)$$

де t_o – витрати праці на підготовку й опис поставленої задачі (приймається 50);

t_u – витрати праці на дослідження алгоритму рішення задачі;

t_a – витрати праці на розробку блок-схеми алгоритму;

t_n – витрати праці на програмування по готовій блок-схемі;

$t_{отл}$ – витрати праці на налагодження програми на ЕОМ;

$t_д$ – витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q * C * (1 + p), \quad (4.2)$$

де q – передбачуване число операторів;

C – коефіцієнт складності програми;

p – коефіцієнт корекції програми в ході її розробки.

$$Q=840*1,2*(1+0,2) = 1210. \quad (4.3)$$

Витрати праці на вивчення опису задачі t_u визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q * B}{(75 \dots 85) * k}, \text{ людино-годин}, \quad (4.4)$$

де B – коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі;

k – коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.

$$t_u = \frac{1210*1,2}{80*1,1} = 16,5, \text{ людино-годин}. \quad (4.5)$$

Витрати праці на розробку алгоритму рішення задачі:

$$t_a = \frac{Q}{(20 \dots 25) * k}, \text{ людино-годин,} \quad (4.6)$$

$$t_a = \frac{1210}{22*1,1} = 50, \text{ людино-годин.} \quad (4.7)$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20 \dots 25)*k}, \text{ людино-годин.} \quad (4.8)$$

$$t_n = \frac{1210}{22 * 1,1} = 50, \text{ людино-годин.} \quad (4.9)$$

Витрати праці на налагодження програми на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4 \dots 5)*k}, \text{ людино-годин,} \quad (4.10)$$

$$t_{отл} = \frac{1210}{4 * 1,1} = 275, \text{ людино-годин.} \quad (4.11)$$

- за умови комплексного налагодження завдання:

$$t_{отл}^{k} = 1,5 * t_{отл}, \text{ людино-годин.} \quad (4.12)$$

$$t_{отл}^{k} = 1,5*275 = 412,5, \text{ людино-годин.} \quad (4.13)$$

Витрати праці на підготовку документації:

$$t_д = t_{др} + t_{до}, \text{ людино-годин,} \quad (4.14)$$

де $t_{др}$ – трудомісткість підготовки матеріалів і рукопису.

$$t_{др} = \frac{Q}{15..20*k}, \text{ людино-годин.} \quad (4.15)$$

$$t_{др} = \frac{1210}{18*1,1} = 61, \text{ людино-годин.} \quad (4.16)$$

$t_{до}$ – трудомісткість редагування, печатки й оформлення документації.

$$t_{до} = 0,75 * t_{др}, \text{ людино-годин.} \quad (4.17)$$

$$t_{до} = 0,75*61 = 45,75, \text{ людино-годин.} \quad (4.18)$$

$$t_д = 61 + 45,75 = 106,75, \text{ людино-годин.} \quad (4.19)$$

Тепер розрахуємо трудомісткість ПЗ:

$$t = 50 + 16,5 + 50 + 50 + 275 + 106,75 = 548,25, \text{ людино-годин.} \quad (4.20)$$

4.2. Розрахунок витрат на створення програми

Витрати на створення ПЗ $К_{по}$ включають витрати на заробітну плату виконавця програми $З_{зп}$ і витрат машинного часу, необхідного на налагодження програми на ЕОМ.

$$К_{по} = З_{зп} + З_{мв}, \text{ грн.} \quad (4.21)$$

Заробітна плата виконавців визначається за формулою:

$$З_{зп}=t*С_{пр}, грн, \quad (4.22)$$

де t – загальна трудомісткість, людино-годин;

$С_{пр}$ – середня годинна заробітна плата програміста, грн/година.

$$З_{зп}=548,25*120=65790, грн. \quad (4.23)$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{мв}=t_{отл}*С_{мч}, грн, \quad (4.24)$$

де $t_{отл}$ – трудомісткість налагодження програми на ЕОМ, год.

$С_{мч}$ – вартість машино-години ЕОМ, грн/год.

$$З_{мв}=275*30=8250, грн. \quad (4.25)$$

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення інформаційної системи.

$$К_{по}=65790+8250=74040, грн. \quad (4.26)$$

Очікуваний період створення ПЗ:

$$Т=\frac{t}{В_к*F_р}, міс. \quad (4.27)$$

де $В_к$ – число виконавців (дорівнює 1);

$F_р$ – місячний фонд робочого часу (при 40 годинному робочому тижні $F_р$=176 годин).

$$T=\frac{548,25}{1*176}=3,11, \text{ міс.} \quad (4.28)$$

4.3. Маркетингові дослідження ринку збуту розробленого програмного продукту

Оскільки тема кваліфікаційної роботи присвячена дослідженням, пов’язаним з когнітивним моделюванням та дослідженням добору навчальних методик в освітніх онлайн системах, то можна зробити висновок, що робота комерційних перспектив не має, тому маркетингові дослідження проведені не були.

Однак, отримані результати роботи можуть бути використані під час створення систем дистанційної освіти, що дозволить досягти соціального ефекту у вигляді підвищення ефективності навчання та засвоєння матеріалу студентами, а також покращення загальної якості освітніх сервісів та методик викладання за допомогою створення нових, якісних освітніх матеріалів та вдосконалення створених раніше.

4.4. Оцінка економічної ефективності впровадження програмного забезпечення

Оскільки виконана робота має виключно дослідницький характер і не передбачає створення кінцевого продукту, то чисельний розрахунок економічної ефективності виконаний бути не може.

Висновок: у результаті виконання кваліфікаційної роботи був обґрунтований метод добору навчальної методики на основі аналізу когнітивних

моделей студентів. У даному економічному розділі було визначені витрати на створення ПЗ, які склали 74040 грн. і час створення ПЗ – 3 місяці.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи був обґрунтований метод добору навчальної методики на основі аналізу когнітивних навичок студента, обчислених за результатами виконання контрольних заходів з дисципліни. За допомогою цього способу було проведене дослідження впливу адаптивної зміни навчальних методик на результати оцінювання окремих студентів.

Для тестування запропонованого методу добору були наведені набори даних з експертними оцінками когнітивних навичок та тем, які необхідні для виконання контрольних завдань, а також результати виконання студентами тестових завдань з обрахованими оцінками когнітивних навичок на основі системи штрафів. Отримані результати показали ефективність запропонованого методу добору, що виражається в покращенні результатів контролю знань окремих студентів.

Обраний спосіб може використовуватись у системах дистанційної освіти з метою підвищення когнітивних навичок студентів, що своєю чергою підвищить ефективність засвоєння матеріалів дисциплін. Для досягнення поставленої мети були виконані наступні задачі:

– досліджені когнітивні процеси, які впливають на процес навчання;

– досліджені методи класифікації когнітивних навичок, що використовуються різними навчальними методиками;

– визначено механізм збору інформації щодо змін когнітивних навичок протягом процесу навчання;

– обґрунтовано добір навчальних методик на підставі результатів аналізу зібраних відомостей.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Bender A., Beller S. Cognition and cognitive science / A. Bender, S. Beller // Frontiers in Psychology. – 2016. – Vol. 7. – P. 24-30.

2. Sun R. Introduction to computational cognitive modeling / R. Sun // The Cambridge handbook of computational psychology – Cambridge University Press, 2008. – P. 3-19.

3. Frischkorn G.T., Schubert A. L. Cognitive Models in Intelligence Research: Advantages and Recommendations for Their Application / G.T. Frischkorn, A.L. Schubert // Journal of Intelligence. – 2018. – Vol. 6, № 3. – P. 1-22.

4. Strube G. Cognitive modeling: research logic in cognitive science / G. Strube, N. J. Smelser, B. Baltes // International Encyclopedia of the Social and Behavioral Sciences. – Amsterdam, 2001. – P. 2124-2128.

5. Bell C. G., Newell A. Computer structures: readings and examples / C. G. Bell, A. Newell. – New York : McGraw-Hill, 1971. – 668 p.

6. Kolers P. A., Smythe W. E. Symbol manipulation: Alternatives to the computational view of mind / P. A. Kolers, W. E. Smythe // Journal of Verbal Learning and Verbal Behavior. – 1984. – Vol. 23, № 3. – P. 289–314.

7. Goertzel B., Lian R., Arel I. A world survey of artificial brain projects, Part II: Biologically inspired cognitive architectures / B. Goertzel, R. Lian, I. Arel // Neurocomputing. – 2010. – Vol. 74, № 1. – P 30-49.

8. Duch W., Oentaryo R. J., Pasquier M. Cognitive Architectures: Where do we go from here? / W. Duch, R. J. Oentaryo, M. Pasquier // Frontiers in Artificial Intelligence and Applications. – 2008. – Vol. 171. – P. 122-136.

9. Anderson J. R., Schunn C. Implications of the ACT-R learning theory: No magic bullets / J. R. Anderson, C. Schunn // Advances in instructional psychology: Educational design and cognitive science. – 2000. – Vol. 5. – P. 1-33.

10. Larid J. E. Extending the Soar cognitive architecture / J. E. Larid // Frontiers in Artificial Intelligence and Applications. – 2008. – Vol. 171. – P. 224-235.

11. Just M. A., Varma S. The organization of thinking: What functional brain imaging reveals about the neuroarchitecture of complex cognition / M. A. Just, S. Varma // Cognitive, Affective, & Behavioral Neuroscience. – 2007. – Vol. 7, № 3. – P. 153-191.

12. Thorisson K. R., Helgasson H. P. Cognitive Architectures and Autonomy: A Comparative Review / K. R. Thorisson, H. P. Helgasson // Journal of Artificial General Intelligence. – 2012. – Vol. 3. – P. 1-30.

13. Kokinov B. The DUAL Cognitive Architecture: A Hybrid Multi-Agent Approach / B. Kokonov // Proceedings of 11th European Conference on Artificial Intelligence, Amsterdam, 1994. – P. 203-207.

14. Baar B. J., Franklin S. How conscious experience and working memory interact / B. J. Baar, S. Franklin // Trends in Cognitive Sciences. – 2003. – Vol. 7, № 4. P. – 166-172.

15. Langley P., Laird J. E., Rogers S. Cognitive architectures: Research issues and challenges / P. Langley, J. E. Laird, S. Rogers // Cognitive Systems Research. – 2009. – Vol. 10, № 2. – P. 141-160.

16. Lewandowsky S., Farrel S. Computational Modeling in Cognition: Principles and Practice / S. Lewandowsky, S. Farrel. – Thousand Oaks : Sage Publications, 2011. – 359 p.

17. Cen H., Koedinger K., Junke B. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement / H. Cen, K. Koedinger, B. Junke // Lecture Notes in Computer Science. – 2006. – Vol. 4053. – P. 164-175.

18. Annis J., Palmeri T. J. Bayesian statistical approaches to evaluating cognitive models / J. Annis, T. J. Palmeri // WIREs Cognitive Science. – 2018. – Vol. 9, № 2. – P. 1-19.

19. Myung J. I. Tutorial on maximum likelihood estimation / J. I. Myung // Journal of Mathematical Psychology. – 2003. – Vol. 47. – P. 90-100.

20. Wagenmakers E. J., Ratcliff R., Gomez R. Assessing model mimicry using the parametric bootstrap / E. J. Wagenmakers, R. Ratcliff, R. Gomez // Journal of Mathematical Psychology. – 2004. – Vol. 48. – P. 28-50.

21. Schwarz G. Estimating the dimension of a model / G. Schwarz // The Annals of Statistics. – 1978. – Vol. 6, № 2. – P. 461-464.

22. Akaike H. A new look at the statistical model identification / H. Akaike // IEEE Transactions on Automatic Control. – 1974. – Vol. 19. – P. 716-723.

23. Finetti B. Foresight: Its logical laws in subjective sources / B. Finetti, H. Kyburg, H. Smokler // Studies in Subjective Probability. – New York : Wiley, 1964. – P. 93-158.

24. Halbrugge M. Evaluating Cognitive Models and Architectures / M. Halbrugge // Evaluating architectures for intelligence. – Menlo Park, California : AAAI Press, 2007. – P. 27-31.

25. Reitman W. R. Cognition and thought / W. R. Reitman. – New York : Wiley, 1965. – 312 p.

26. Fitts P. M. 1954. The information capacity of the human motor system in controlling the amplitude of movement / P. M. Fitts // Journal of Experimental Psychology. – 1954. – Vol. 47, № 6. – P. 381-391.

27. Anderson J. R., Lebiere C. The Newell Test for a theory of cognition / J. R. Anderson, C. Lebiere // Behavioral and Brain Sciences. – 2003. – Vol. 26, № 5. – P. 587-640.

28. Wexler K. 1978. A review of John R. Anderson's language, memory, and thought / K. Wexler // Cognition. – 1978. – Vol. 6, № 4. – P. 327-351.

29. Tadepalli P. Cognitive architectures have limited explanatory power / P. Tadepalli // Behavioral and Brain Sciences. – 2003. – Vol. 26, № 5. – P. 622-623

30. Kiely K. Cognitive function / K. Kiley, K. M. Michalos // Encyclopedia of Quality of Life and Well-Being Research. – Springer, 2014. – P. 974-978.

31. Smith A. D., Kelly A. Cognitive Processes / A. D. Smith, A. Kelly // The Encyclopedia of Adulthood and Aging. – Hoboken, NJ : John Wiley & Sons, Inc., 2015. – P. 1-4.

32. Salthouse T. Correlates of cognitive change / T. Salthouse // Journal of Experimental Psychology: General. – 2014. – Vol. 143. – P. 1026-1048.

33. Craik F., Salthouse T. Handbook of aging and cognition, 3rd edition / F. Craik, T. Salthouse. – New York : Psychology Press, 2008. – 672 p.

34. Neisser U. Cognitive psychology / U. Neisser. – New York : Appleton-Century-Crofts, 1967. – 351 p.

35. Miller G. A. The magical number seven, plus or minus two: Some limits on our capacity for processing information / G. A. Miller // Psychological Review. – 1956. – Vol. 63, № 2. – P. 81-97.

36. Baddeley A. D., Hitch G. Working Memory / A. D. Baddeley, G. Hitch // Psychology of Learning and Motivation. – 1974. – Vol. 8. – P. 47-89.

37. Goldstein W. M., Hogarth, R. M. (1997). Judgment and decision research: Some historical context / W. M. Goldstein, R. M. Hogarth // Research on judgment and decision making: Currents, connections, and controversies. – Cambridge University Press, 1997. – P. 3-65.

38. Betsch T., Haberstroh S. The routines of decision making / T. Betsch, S. Haberstroh. – New Jersey: Lawrence Erlbaum Associates Publishers, 2004. – 392 p.

39. Bruner J. S., Goodnow J. J., Austin G. A. A study of thinking / J. S. Bruner, J. J. Goodnow, G. A. Austin. – New York: Wiley, 1956. – 330 p.

40. Markman A. B., Ross B. H. Category use and category learning / A. B. Markman, B. H. Ross // Psychological Bulletin. – 2003. – Vol. 129, № 4. – P. 592-613.

41. Karlsson L., Julsin P., Olsson H. Exemplar-based inference in multi-attribute decision making: Contingent, not automatic, strategy shifts? / L. Karlsson, P. Juslin // Judgment and Decision Making. – 2008. – Vol. 3. – P. 244-260.

42. Knauff M., Wolf A. G. Complex cognition: the science of human reasoning, problem-solving, and decision-making / M. Knauff, A. G. Wolf // Cognition Process. – 2010. – Vol. 11. – P. 99-102.

43. Li A., Maani K. Dynamic decision-making, learning and mental models / A. Li, K. Maani // Proceedings of the 29th International Conference of the System Dynamics Society, Washington DC, 24-28 july 2011. – Washington DC, 2011. – P. 1-21.

44. Edwards W. Dynamic Decision Theory and Probabilistic Information Processings / W. Edwards // Human Factors: The Journal of the Human Factors and Ergonomics Society. – 1962. – Vol. 4, № 2. – P. 59-74.

45. Gonzalez C. Instance-based learning in dynamic decision making / C. Gonzalez // Cognitive Science. – 2003. – Vol. 27, № 4. – P. 591-635.

46. Klein G. A., Orasanu J., Calderwood R., Zsambok C. E. Decision making in action: Models and methods / G. A. Klein, J. Orasanu, C. E. Zsambok. – New York : Ablex Publishing, 1993. – 448 p.

47. Simon H. A. Modes of man; social and rational / H. A. Simon. – New York: Wiley, 1957. – 287 p.

48. March J. G. A primer on decision making: how decisions happen / J. G. March. – New York: Free Press, 1994. – 289 p.

49. Bloom B. S. Taxonomy of educational objectives: the classification of educational goals / B. S. Bloom. – New York: Longmans, Green, 1956. – 403 p.

50. Anderson L. W., Krathwohl D. R. A taxonomy for learning, teaching, and assessing: a revision of Bloom's taxonomy of educational objectives / L. W. Anderson, D. R. Krathwohl. – New York : Longmans, 2001. – 352 p.

51. Thomas C. O., Linden J. B. Cognitive Psychology / C. O. Thomas, J. B. Linden // The SAGE Handbook of Qualitative Research in Psychology. – Los Angeles : SAGE, 2008. – 664 p.

52. Kester L., Nievelstein F., Giesbers B. Uncovering cognitive processes: Different techniques that can contribute to cognitive load research and instruction / L. Kester, F. Nievelstein, B. Giesbers // Computers in Human Behavior. – 2009. – Vol. 25 № 2. – P. 325-331.

53. What is cognitive psychology and the general approaches to the study of human cognition. *ECS Blogs*: website. URL: https://blog.soton.ac.uk/comp6044/2011/10/25/what-is-cognitive-psychology-and-the-general-approaches-to-the-study-of-human-cognition/ (дата звернення: 29.07.2021).

54. Ahmad M.W., Mourshed M., Rezgui Y. Trees vs Neurons: Comparison between Random Forest and ANN for high-resolution prediction of building energy consumption / M. W. Ahmad, M. Mourshed, Y. Rezgui // *Energy and Buildings*. – 2017. – Vol. 147. – P. 77–89.

55. Lindsey V. R., Khajah M. M., Mozer M. Automatic Discovery of Cognitive Skills to Improve the Prediction of Student Learning / V. R. Lindsey, M. M. Khajan, M. Mozer // Proceeding of 21nd Conference on Neural Information Processing Systems, Vancouver, 8-10 december 2008. – Vancouver, 2008. – P. 1-10.

56. Liley M., Barker T. Students' perceived usefulness of formative feedback for a computer-adaptive test / T. Barker, M. Liley // Electronic Journal of e-Learning. – 2007. – Vol. 5. – P. 31-38.

57. Galton F. Vox Populi / F. Galton // *Nature.* – 1907. – Vol. 75. – P. 450-451.

58. Surowiecki J. The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business, economies, societies, and nations / J. Surowiecki. – New York : Doubleday, 2004. – 336 p.

Додаток А

КОД ПРОГРАМИ

Random_Forest.py

```
import numpy as np
import pandas as pd
import itertools
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import precision_score, recall_score, roc_auc_score, roc_curve,
confusion_matrix
import matplotlib.pyplot as plt
from bayes_opt import BayesianOptimization
from sklearn.model_selection import cross_val_score
from utils import *
from sklearn.model_selection import validation_curve
# завантаження тестових та навчальних даних
train_data = load_csv('testing_ds_1.csv')
test_data = load_csv('testing_ds_2.csv')
# лист ознак класифікатора
features_list = ['Theme 1', 'Theme 2', 'Theme 3', 'Recognition', 'Recalling', 'Interpreting',
'Exemplifying', 'Classifying', 'Summarizing', 'Inferring', 'Comparing', 'Explaining', 'Executing',
'Implementing', 'Differentiating', 'Organizing', 'Attributing', 'Checking', 'Critiquing',
'Generating', 'Planning', 'Producing']
# об'єкт класифікації
target_list = 'Result'
# тренувальні дані для класифікатора
train = train_data[features_list]
train_labels = train_data[target_list]
# тестові дані для класифікатора
test = test_data[features_list]
test_labels = test_data[target_list]
print("Розміри навчальних ознак", train.shape, sep = ":\n")
print("\n")
print("Розміри навчальних об'єктів", train_labels.shape, sep = ":\n")
print("\n")
print("Розміри тестових ознак", test.shape, sep = ":\n")
print("\n")
print("Розміри тестових об'єктів", test_labels.shape, sep = ":\n")
print("\n")
print("Навчальні ознаки", train[:2], sep = ":\n")
print("\n")
print("Навчальні об'єкти", train_labels[:2], sep = ":\n")
# методи оптимізації гіперпараметрів класифікатора
def bayesian_optimization(dataset, function, parameters):
   iterations = 5
   params = {"alpha": 1e-4}
   BO = BayesianOptimization(function, parameters)
   BO.maximize(n_iter=iterations, **params)
   return BO.max
def rfc_optimization(splits):
    def function(n_estimators, max_depth, min_samples_split):
        return cross_val_score(
            RandomForestClassifier(
                n_estimators=int(max(n_estimators, 0)),
                max_depth=int(max(max_depth, 1)),
                min_samples_split=int(max(min_samples_split, 2)),
                n_jobs=-1,
                random_state=42,
                class_weight="balanced"
            ),
            X=test,
            y=test_labels,
            cv=splits,
            scoring="roc_auc",
            n_jobs=-1
        ).mean()
    parameters = {"n_estimators": (10, 1000),
                  "max_depth": (1, 150),
                  "min_samples_split": (2, 10)}
    return function, parameters
def xgb_optimization(cv_splits, eval_set):
    def function(eta, gamma, max_depth):
        return cross_val_score(
            xgb.XGBClassifier(
```

```
                objective="binary:logistic",
                learning_rate=max(eta, 0),
                gamma=max(gamma, 0),
                max_depth=int(max_depth),
                seed=42,
                nthread=-1,
                scale_pos_weight=len(train_labels[train_labels == 0]) /
                                 len(train_labels[train_labels == 1])),
            X=train,
            y=train_labels,
            cv=cv_splits,
            scoring="roc_auc",
            fit_params={
                "early_stopping_rounds": 10,
                "eval_metric": "auc",
                "eval_set": eval_set},
            n_jobs=-1).mean()
    parameters = {"eta": (0.001, 0.4),
                  "gamma": (0, 20),
                  "max_depth": (1, 2000)}
    return function, parameters
# метод дослідження оптимізації параметрів моделі
def optimization(X_train, y_train, X_test, y_test, function, parameters):
    dataset = (X_train, y_train, X_test, y_test)
    cv_splits = 4

    best_solution = bayesian_optimization(dataset, function, parameters)
    params = best_solution["params"]

    model = RandomForestClassifier(
        n_estimators=int(max(params["n_estimators"], 0)),
        max_depth=int(max(params["max_depth"], 1)),
        min_samples_split=int(max(params["min_samples_split"], 2)),
        n_jobs=-1,
        random_state=42,
        class_weight="balanced")
    model.fit(X_train, y_train)
    return model

# метод надає статистичну інформацію про середнє значення вузлів та глубину дерев у класифікаторі
def trees_statistics(model):
    nodes = []
    depths = []
    for tree in model.estimators_:
        nodes.append(tree.tree_.node_count)
        depths.append(tree.tree_.max_depth)
    print(f'Середнє число вузлів {int(np.mean(nodes))}')
    print(f'Середня максимальна глибина {int(np.mean(depths))}'

def build_validation_curve():
    parameter_range = np.arange(1, 10, 1)
    train_score, test_score = validation_curve(RandomForestClassifier(),
                                               train,
                                               train_labels,
                                               param_name="n_neighbors",
                                               param_range=parameter_range,
                                               cv=5,
                                               scoring="accuracy")

    # Обчислення серенього значення та середньоквадратичного відхилення для оцінки навчальних даних
    mean_train = np.mean(train_score, axis=1)
    std_train = np.std(train_score, axis=1)

    # Обчислення серенього значення та середньоквадратичного відхилення для оцінки тестових даних
    mean_test = np.mean(test_score, axis=1)
    std_test = np.std(test_score, axis=1)
    plt.plot(parameter_range, mean_train,
             label="Training Score", color='b')
    plt.plot(parameter_range, mean_test,
             label="Cross Validation Score", color='g')
    plt.title("Validation Curve with KNN Classifier")
    plt.xlabel("Number of Neighbours")
    plt.ylabel("Accuracy")
    plt.tight_layout()
    plt.legend(loc='best')
    plt.show()

# метод повернає прогнозовані результати та актуальні результати виконання завдання для студента
def compare_results_for_task(task_number, student_id, model):
```

```
    student_result = test_data['participant id' == student_id & 'task id' == task_number]
    prediction_result = model.predict(student_result)
    print(prediction_result)
    actual_result = student_result['target']
    return prediction_result, actual_result

# метод надає інформацію про важливість ознак для класифікації даних
def feature_importance(model):
    importances = list(model.feature_importances_)
    feature_importances = [(feature, round(importance, 2)) for feature, importance in
zip(features_list, importances)]
    feature_importances = sorted(feature_importances, key=lambda x: x[1], reverse=True)
    [print('Змінна: {:20} Важливість: {}'.format(*pair)) for pair in feature_importances]
    # Plot the feature importances
    plt.figure()
    plt.title("Feature importances", size=30)
    indices = np.argsort(importances)[::-1]
    std = np.std([model.feature_importances_ for tree in model.estimators_], axis=0)
    plt.bar(range(features_list.shape[1]), importances[indices],
            yerr=std[indices], color='r', align="center")
    plt.xticks(range(features_list.shape[1]), features_list[indices], rotation=20, fontsize=7)
    plt.xlim([-1, features_list.shape[1]])
    plt.show()
    return importances

# метод будує матрицю неточностей моделі
def create_confusion_matrix(test_labels, predictions, classes, normalize=False, title='Матриця
неточностей', color=plt.cm.Oranges):
    cm = confusion_matrix(test_labels, predictions)
    print('Матриця неточностей')
    print(cm)
    plt.figure(figsize=(10, 10))
    plt.imshow(cm, interpolation='nearest', cmap=color)
    plt.title(title, size=24)
    plt.colorbar(aspect=4)
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45, size=14)
    plt.yticks(tick_marks, classes, size=14)
    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    # Labeling the plot
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt), fontsize=20,
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")
    plt.grid(None)
    plt.tight_layout()
    plt.ylabel('True label', size=18)
    plt.xlabel('Predicted label', size=18)
    plt.show()

# метод оцінює побудовану модель, будуючи криву ROC
def evaluate_model(predictions, probs, test_labels):
    baseline = {}
    baseline['recall'] = recall_score(test_labels, [1 for _ in range(len(test_labels))])
    baseline['precision'] = precision_score(test_labels, [1 for _ in range(len(test_labels))])
    baseline['roc'] = 0.5

    results = {}
    results['recall'] = recall_score(test_labels, predictions)
    results['precision'] = precision_score(test_labels, predictions)
    results['roc'] = roc_auc_score(test_labels, probs)
    for metric in ['recall', 'precision', 'roc']:
        print(f'{metric.capitalize()} Baseline: {round(baseline[metric], 2)} Test:
{round(results[metric], 2)}')
    base_fpr, base_tpr, _ = roc_curve(test_labels, [1 for _ in range(len(test_labels))])
    model_fpr, model_tpr, _ = roc_curve(test_labels, probs)
    plt.figure(figsize=(8, 6))
    plt.rcParams['font.size'] = 16
    plt.plot(base_fpr, base_tpr, 'b', label='baseline')
    plt.plot(model_fpr, model_tpr, 'r', label='model')
    plt.legend()
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('ROC Curves')
    plt.show()

# метод побудови графіку порівняння прогнозованихз та реальних результатів
def comparsion_plot(predict, actual):
```

```
    fig = plt.figure()
    ax = fig.add_axes([0.1, 0.1, 0.8, 0.8])  # main axes
    ax.plot([1, 2, 3, 4], actual, '-o', label='Результати тестування')
    ax.plot([1, 2, 3, 4], predict, '-x', label='Прогнозовані результати')
    plt.title('Графік порівняння результатів тестування та прогнозів успіхів учня після зміни
навчальних методик',
              fontsize=15, fontweight='bold')
    plt.xlabel('Кількість тестувань', fontsize=12)
    plt.ylabel('Результат', fontsize=12)
    plt.legend()
    ax.set_xticks([1, 2, 3, 4])
    ax.set_yticks([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
    ax.set_yticklabels(['', 'Невдача\n 0 спроба', 'Невдача\n 1 спроба', \n 2 спроба', 'Невдача\n 3
спроба', 'Невдача\n 4 спроба', 'Невдача\n 5 спроба', 'Успіх\n 5 спроба', 'Успіх\n 4 спроба',
'Успіх\n 3 спроба', 'Успіх\n 2 спроба', 'Успіх\n 1 спроба', ''], rotation=45)
    plt.grid(True)
    plt.show()
# створення класифікатора
model = RandomForestClassifier(n_estimators=100,
                               random_state=50,
                               max_features='sqrt',
                               n_jobs=-1,
                               verbose=1
                               )
# тренування моделі
model.fit(train, train_labels)
# тестування моделі
predictions = model.predict(test)
probs = model.predict_proba(test)[:, 1]
# статистика дерев
trees_statistics(model)
importances = feature_importance(model, train, train_labels)
# evaluate_model(predictions, probs, test_labels)
# побудова матриці помилок
create_confusion_matrix(test_labels, predictions, classes=['Успіх 1 спроба', 'Успіх 2 спроба',
'Успіх 3 спроба', 'Успіх 4 спроба', 'Успіх 5 спроба', 'Невдача 0 спроба', 'Невдача 1 спроба',
'Невдача 2 спроба', 'Невдача 3 спроба', 'Невдача 4 спроба', 'Невдача 5 спроба',
])
# порівняння прогнозів з реальними значеннями
res = "\n".join("{} | {}".format(x, y) for x, y in zip(list(predictions), list(test_labels)))
print (res)
comparsion_plot(list(predictions), list(test_labels))
```

course_view.py

```
import json
import os
import random
import sys
from django.db import transaction, IntegrityError
from django.shortcuts import render
from django.conf import settings
from .serializers import *
from rest_framework import viewsets, permissions, generics
from rest_framework.response import Response
from .models import *
from custom_auth.models import *
from user_space.models import *
from tasks.models import *
from media_files.models import *
from tags.models import *
from datetime import datetime
from courses.permissions import IsOwnerOrReadOnly
from rest_framework import permissions
import base64
from django_filters import rest_framework as filters
import uuid
from rest_framework.parsers import JSONParser, MultiPartParser
from django.core.files.storage import default_storage

class CourseFilter(filters.FilterSet):
    min_price = filters.NumberFilter(field_name="price", lookup_expr='gte')
    max_price = filters.NumberFilter(field_name="price", lookup_expr='lte')
    search = filters.CharFilter(field_name='tags__name', lookup_expr='icontains')

    class Meta:
        model = Course
        fields = ['difficulty']

class CourseAPI(generics.GenericAPIView):
```

```
    permission_classes = [permissions.AllowAny]
    serializer_class = CourseSerializer
    queryset = Course.objects.all()
    filter_backends = [filters.DjangoFilterBackend]
    filterset_class = CourseFilter
    parser_classes = [JSONParser, MultiPartParser]

    def update_table_of_content(self, course, should_table_updated):
        if should_table_updated:
            create_table_of_content(course)

    def get(self, request):
        return Response(self.get_serializer(self.filter_queryset(self.get_queryset()),
many=True).data, status=200)

    def post(self, request):
        if 'difficulty' in request.data and (
                request.data.get('difficulty') is None or request.data.get('difficulty') == ""):
            return Response({'status': 'error', 'message': 'Difficulty is should not be empty'},
status=400)

        if 'price' in request.data and (request.data.get('price') is None or
request.data.get('price') == ""):
            return Response({'status': 'error', 'message': 'Price is should not be empty'},
status=400)

        if 'rating' in request.data and (request.data.get('rating') is None or
request.data.get('rating') == ""):
            return Response({'status': 'error', 'message': 'Rating is should not be empty'},
status=400)

        if 'is_deleted' in request.data and (
                request.data.get('is_deleted') is None or request.data.get('is_deleted') == ""):
            return Response({'status': 'error', 'message': 'is_deleted is should not be empty'},
status=400)

        try:
            with transaction.atomic():
                is_topic_updated = False
                if len(request.data.get("entities_to_delete", [])) != 0:
                    for entity in request.data.get("entities_to_delete"):
                        entity_uuid = entity.get("uuid")
                        type = entity.get("type")

                        if not entity_uuid or not type:
                            raise Exception('Uuid and type are required parameters to delete
entity')

                        model = entities_to_delete_mapping.get(type)

                        if not model:
                            raise Exception(f"Unsupported entity {type}")

                        model.objects.get(uuid=entity_uuid).delete()

                course = Course.objects.get(uuid=request.data.get('uuid'))
                course_data_to_update = {
                    'difficulty': request.data.get('difficulty', course.difficulty),
                    'price': request.data.get('price', course.price),
                    'rating': request.data.get('rating', course.rating),
                    'tags': request.data.get('tags', [tag.uuid for tag in course.tags.all()]),
                    'is_deleted': request.data.get('is_deleted', course.is_deleted),
                    'owner': course.owner.uuid,
                }

                course_serializer = CourseDeserializer(course, data=course_data_to_update)
                course_serializer.is_valid(raise_exception=True)
                course_serializer.save()

                return Response({'status': 'success', 'message': 'Course updated'}, status=200)

        except Exception as e:
            return Response({'status': 'error', 'message': f'Error with course updating: {e}'},
status=400)

    def put(self, request):
        try:
            with transaction.atomic():
                course_data = {
```

```
                    'owner': request.user.uuid,
                    'difficulty': request.data.get('difficulty'),
                    'price': request.data.get('price', 0),
                    'rating': request.data.get('rating', 0),
                    'tags': request.data.get('tags')
                }

                course_deserializer = CourseDeserializer(data=course_data)
                course_deserializer.is_valid(raise_exception=True)
                course = course_deserializer.save()

                transaction.on_commit(lambda: create_table_of_content(course))
        except Exception as e:
            return Response({'status': 'error', 'message': f'Error with course creation {e}'},
status=400)

        return Response({'status': 'success', 'message': 'Course created'}, status=200)
```

task_view.py

```
class TaskFilter(filters.FilterSet):
    search = filters.CharFilter(field_name='tags__tag__name', lookup_expr='icontains')

    class Meta:
        model = Task
        fields = ['difficulty']

class GetTasksAPI(generics.GenericAPIView):
    serializer_class = TaskSerializer
    queryset = Task.objects.all()
    filter_backends = [filters.DjangoFilterBackend]
    filterset_class = TaskFilter

    def get(self, request):
        if 'is_public' in request.GET:
            value = request.GET['is_public'].upper()
            if value == 'TRUE':
                is_public = True
            elif value == 'FALSE':
                is_public = False
            else:
                return Response({'status': 'error', 'message': 'invalid parameter is public'},
status=400)
            if is_public:
                return Response(self.get_serializer(self.get_queryset().filter(is_public=True),
many=True).data,
                                status=200)
            return Response(self.get_serializer(self.get_queryset().filter(is_public=False),
many=True).data,
                            status=200)
        else:
            return Response(self.get_serializer(self.filter_queryset(self.get_queryset()),
many=True).data, status=200)

class TaskResultAPI(generics.GenericAPIView):
    serializer_class = TaskResultSerializer
    queryset = TaskResult.objects.all()

    def post(self, request):
        if 'task_uuid' not in request.data:
            return createResponseWithStatus(
            'task_uuid is required', 400)
        if 'student_uuid' not in request.data:
            return createResponseWithStatus(
            'student_uuid is required', 400)
        if 'answer' not in request.data:
            return createResponseWithStatus(
            'answer is required', 400)
        if 'time' not in request.data:
            return createResponseWithStatus(
            'time is required', 400)
        try:
            process_answer(
            task_uuid, student_uuid, answer, time)
            return createResponseWithStatus(
            'Task answer successfully processed', 200)
        except Exception as e:
            return createResponseWithStatus(
            f'Something went wrong {e}', 400)
```

```
class TaskAPI(generics.GenericAPIView):
    serializer_class = TaskCreationSerializer
    queryset = Task.objects.all()

    def post(self, request):
        try:
            with transaction.atomic():
                if len(request.data.get("entities_to_delete", [])) != 0:
                task = Task.objects.get(uuid=request.data.get('uuid'))
                task_data_to_update = {
                    'is_public': request.data.get('is_public', task.is_public),
                    'course': request.data.get('course', task.course.uuid),
                    'topic': request.data.get('topic', task.topic.uuid),
                    'difficulty': request.data.get('difficulty', task.difficulty),
                    'updated_at': int(datetime.now().timestamp()),
                    'number': request.data.get('number', task.number),
                }
            task_serializer = TaskCreationSerializer(task, data=task_data_to_update)
            task_serializer.is_valid(raise_exception=True)
            task_serializer.save()

            return Response({'status': 'success', 'message': 'Task updated'}, status=200)
        except Exception as e:
            return Response({'status': 'error', 'message': f'Error with tasks updating: {e}'},
status=400)

    def put(self, request):
        if not request.user.is_authenticated:
            return Response("User not authenticated", status=401)
        if 'course' not in request.data:
            return Response({'status': 'error', 'message': 'Course id is required'}, status=400)
        if 'topic' not in request.data:
            return Response({'status': 'error', 'message': 'Topic id is required'}, status=400)
        main_files = 0
        for file in request.data.get('content'):
            if 'is_main' in file and file.get('is_main'):
                main_files += 1
        if main_files == 0 or main_files > 1:
            return Response({'status': 'error', 'message': 'Task must contain only one main file'},
status=400)
        task_data = {
            "is_public": request.data.get("is_public"),
            "course": request.data.get("course"),
            "topic": request.data.get("topic"),
            "difficulty": request.data.get("difficulty"),
            "number": request.data.get("number"),
            "updated_at": int(datetime.now().timestamp())
        }
        try:
            task_serializer = TaskCreationSerializer(data=task_data)
            task_serializer.is_valid(raise_exception=True)
            task = task_serializer.save()
            return Response({'status': 'success', 'message': 'success'}, status=200)
        except Exception as e:
            return Response({'status': 'error', 'message': f'{e}'}, status=400)
```

Додаток Б

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Факультет інформаційних технологій
Кафедра програмного забезпечення комп'ютерних систем

ВІДГУК

Керівника
економічної
частини Касьяненко Л.В., к.е.н., доцента каф. ПЕП та ПУ
(прізвище, ім'я, по батькові, вчене звання, посада, місце роботи)

На кваліфікаційну роботу
студента М'якенького Арсенія Вячеславовича
(прізвище, ім'я, по батькові)

курсу II групи 122м-20-1
спеціальності Комп'ютерні науки
на тему Обґрунтування добору навчальної методики на підставі аналізу когнітивних моделей у системі дистанційної освіти.

.

«___»_____2022 р. _____
(підпис)

Додаток В

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
М’якенький.doc	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
М’якенький.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF.
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму.
Презентація	
Презентація_М’якенький.ppt	Презентація кваліфікаційної роботи.