

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет комп'ютерних наук, фізики та математики
Кафедра комп'ютерних наук та програмної інженерії

Формальна верифікація моделей подання даних у системах на базі штучного інтелекту

Кваліфікаційна робота (проект)
на здобуття ступеня вищої освіти “бакалавр”

Виконав: здобувач 4 курсу 441 групи

Спеціальності 121 Інженерія програмного забезпечення

Освітньо-професійної програми

Інженерія програмного забезпечення

Петухов Владислав Михайлович

Керівник: доктор філософії (PhD), доцент

Полторацький Максим Юрійович

Рецензент: наукова співробітниця

Інституту кібернетики ім. В. М. Глушкова

НАН України

Тарасіч Юлія Геннадіївна

ЗМІСТ

ВСТУП	4
РОЗДІЛ 1	7
1.1 Аналіз предметної області систем штучного інтелекту	7
1.2 Поняття та класифікація систем на базі штучного інтелекту	9
1.3 Архітектура сучасних ШІ-систем.....	11
1.4 Поняття та роль моделей подання даних у ШІ-системах	14
1.5 Основні підходи до подання даних у системах штучного інтелекту	16
1.5.1 Логічні моделі подання даних	17
1.5.2 Графові та онтологічні моделі	18
1.5.3 Векторні та статистичні моделі	20
1.6 Проблеми коректності та надійності моделей подання даних	21
1.7 Аналіз помилок і суперечностей у моделях подання даних	22
1.8 Висновки до першого розділу	23
РОЗДІЛ 2	25
2.1 Побудова онтології предметної області.....	25
2.2 Властивості даних, зв'язки та наповнення моделі	27
2.2.1 Властивості даних (Datatype Properties)	27
2.2.2 Зв'язки між класами (Object Properties)	28
2.2.3 Наповнення моделі (екземпляри)	29
2.3 Отримання результатів за допомогою SPARQL та формальна верифікація моделі за допомогою SHACL	33
2.3.1 Використання SPARQL для отримання результатів	33
2.3.2 Використання SHACL для формальної верифікації.....	35
2.4 Висновки до другого розділу.....	37
ВИСНОВКИ	39
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	41

ВСТУП

Стрімкий розвиток систем на базі штучного інтелекту зумовлює зростання вимог до коректності, надійності та формальної обґрунтованості моделей подання даних, що використовуються в таких системах. Дані є ключовим компонентом будь-якої інтелектуальної системи, оскільки саме на їх основі здійснюється навчання моделей, прийняття рішень та формування результатів роботи штучного інтелекту. Помилки або суперечності в моделях подання даних можуть призводити до некоректної поведінки системи, зниження її ефективності або виникнення критичних збоїв.

У зв'язку з цим особливої актуальності набуває застосування методів формальної верифікації, які дозволяють математично довести коректність моделей подання даних, виявити логічні суперечності та забезпечити відповідність заданим вимогам. Формальна верифікація надає можливість аналізувати властивості моделей незалежно від конкретної реалізації та є важливим інструментом підвищення надійності складних інтелектуальних систем.

Актуальність теми дипломної роботи зумовлена необхідністю теоретичного обґрунтування та систематизації підходів до формальної верифікації моделей подання даних у системах на базі штучного інтелекту, а також зростаючим попитом на надійні та перевірені методи забезпечення якості інтелектуальних систем.

Метою дипломної роботи є дослідження теоретичних основ формальної верифікації моделей подання даних у системах штучного інтелекту, а також розробка та перевірка моделі подання даних на основі RDF-графа знань із застосуванням засобів формальної верифікації.

Для досягнення поставленої мети в роботі необхідно вирішити такі завдання:

- проаналізувати основні класи та архітектури систем на базі штучного інтелекту;
- дослідити підходи до подання даних у ШІ-системах та їхні особливості;

- розглянути проблеми коректності та надійності моделей подання даних;
- вивчити теоретичні основи та методи формальної верифікації;
- розробити онтологічну модель предметної області;
- реалізувати RDF-граф знань;
- виконати формальну верифікацію моделі за допомогою SHACL;
- реалізувати отримання результатів за допомогою SPARQL.

Об'єктом дослідження є системи на базі штучного інтелекту.

Предметом дослідження є методи та підходи формальної верифікації моделей подання даних у системах штучного інтелекту.

У роботі застосовуються методи аналізу та узагальнення наукових джерел, формальної логіки, теорії множин, теорії графів, а також методи порівняльного аналізу та формалізації.

Наукова новизна роботи полягає у систематизації теоретичних підходів до формальної верифікації моделей подання даних у системах на базі штучного інтелекту, а також у застосуванні семантичних технологій для побудови та перевірки узгодженості моделей даних.

Практичне значення отриманих результатів полягає у можливості використання розробленої моделі подання даних та механізмів її формальної верифікації при проєктуванні інтелектуальних систем, зокрема систем рекомендацій, що враховують обмеження користувачів та забезпечують безпечний підбір рішень.

Дипломна робота складається зі вступу, двох розділів, висновків та списку використаних джерел.

У першому розділі розглядаються системи штучного інтелекту, їх класифікація, архітектура та підходи до подання даних, а також аналізуються проблеми коректності та узгодженості моделей даних.

Другий розділ присвячений побудові онтологічної моделі предметної області, її представленню у вигляді RDF-графа знань, а також реалізації формальної верифікації за допомогою SHACL і отриманню результатів за допомогою SPARQL.

У висновках наведено основні результати дослідження та окреслено напрями подальших наукових пошуків.

РОЗДІЛ 1

Аналіз систем на базі штучного інтелекту та моделей подання даних

1.1 Аналіз предметної області систем штучного інтелекту

Системи на базі штучного інтелекту (ШІ) сьогодні є одним із найактуальніших напрямів розвитку інформаційних технологій, оскільки вони дозволяють автоматизувати складні процеси аналізу даних та підтримки прийняття рішень. Під такими системами зазвичай розуміють програмно-апаратні комплекси або програмні рішення, які можуть виконувати завдання, пов'язані з інтелектуальною діяльністю людини: розпізнавання об'єктів, аналіз тексту, прогнозування, класифікацію, побудову рекомендацій та виявлення прихованих закономірностей у даних [1 – 10].

Предметна область ШІ-систем є дуже широкою, адже штучний інтелект застосовується у різних сферах діяльності. Наприклад, у медицині такі системи використовуються для аналізу знімків, допомоги в діагностиці або прогнозуванні ризиків захворювань. У фінансовій сфері ШІ застосовують для оцінки кредитних ризиків, виявлення підозрілих транзакцій та побудови торгових стратегій. У промисловості інтелектуальні алгоритми допомагають прогнозувати поломки обладнання, оптимізувати виробничі процеси, а також контролювати якість продукції. В освіті системи на базі ШІ можуть адаптувати матеріал під рівень студента, автоматично перевіряти відповіді та пропонувати персоналізовані рекомендації [1].

Однак варто зазначити, що сучасний штучний інтелект у більшості випадків не є універсальним мисленням або самостійною свідомістю. Найчастіше це сукупність математичних методів і алгоритмів, які працюють на основі даних і здатні навчатися на прикладах. Тобто ефективність ШІ-системи багато в чому залежить від того, наскільки коректно зібрані та підготовлені дані, які використовуються для навчання і тестування моделі. Саме тому останнім часом все більшого значення набуває концепція data-centric AI, де акцент робиться на якості даних, правильному формуванні датасетів та усуненні проблем у вхідній інформації [2].

Для розуміння предметної області важливо виділити основні типи задач, які розв'язують ІІІ-системи. Однією з найпоширеніших є класифікація, тобто віднесення об'єкта до одного з наперед визначених класів. Іншим важливим класом задач є регресія, коли система має спрогнозувати числове значення. Також часто зустрічаються задачі кластеризації, де потрібно автоматично згрупувати об'єкти за схожістю. Окрему роль відіграють задачі ранжування та рекомендацій, де система визначає порядок об'єктів за релевантністю та пропонує користувачу оптимальні варіанти. У сучасних ІІІ-системах активно розвиваються напрямки, пов'язані з комп'ютерним зором та обробкою природної мови, оскільки саме ці сфери забезпечують взаємодію людини з цифровими сервісами на більш природному рівні [1 – 11].

Важливим аспектом предметної області ІІІ є різноманіття типів даних, з якими працюють інтелектуальні системи. У практиці можна виділити структуровані дані (таблиці, реляційні бази даних), напівструктуровані (JSON, XML) та неструктуровані дані (текст, зображення, відео, аудіо). Для кожного типу даних існують свої підходи до обробки та подання. Наприклад, табличні дані часто аналізуються статистичними методами або класичним машинним навчанням, а для текстів і зображень більш ефективними є нейронні мережі та векторні подання. Водночас у багатьох задачах виникає потреба подати інформацію не просто як набір окремих значень, а як систему взаємопов'язаних знань. Саме тоді використовуються онтології та знаннєві графи, які дозволяють формалізувати предметну область через сутності та зв'язки між ними [3].

Сучасні ІІІ-системи часто є складними комплексними рішеннями, які включають не лише модель, але й повний цикл її роботи. У загальному вигляді можна виділити такі етапи функціонування: збір даних, попередня обробка, формування ознак, навчання моделі, оцінювання результатів та інтеграція у прикладну систему [4].

Таким чином, інтелектуальна система залежить не тільки від алгоритму, але й від того, як організовані вхідні дані і як вони представлені. Це можна

описати через підхід “Input – Model – Output”, де вхідна інформація є фундаментом для навчання і роботи моделі, а вихідні результати використовуються у прийнятті рішень чи взаємодії з користувачем [4].

Окрему увагу в сучасній науковій літературі привертають агентні ШІ-системи, які здатні виконувати послідовність дій, взаємодіяти з інструментами та приймати рішення в процесі роботи. Такі системи часто включають декілька модулів: модуль планування, модуль пам'яті, модуль генерації відповіді, а також інтерфейси для взаємодії із зовнішніми джерелами даних або сервісами [5].

Отже, аналіз предметної області систем штучного інтелекту показує, що такі системи є комплексними рішеннями, які залежать від даних, методів їх подання та механізмів обробки. Тому дослідження моделей подання даних і підходів до забезпечення їх коректності є важливою складовою підвищення надійності ШІ-систем.

1.2 Поняття та класифікація систем на базі штучного інтелекту

Поняття систем на базі штучного інтелекту є досить широким, оскільки такі системи можуть суттєво відрізнятися за призначенням, способом роботи з інформацією, ступенем автономності та рівнем складності. У загальному вигляді систему на базі ШІ можна розглядати як сукупність програмних і технічних компонентів, які забезпечують інтелектуальну обробку даних, побудову висновків або формування рішень на основі знань чи навчання на прикладах [1 – 10]. Тобто ключова відмінність таких систем від традиційних програм полягає в тому, що вони не тільки виконують заздалегідь прописаний алгоритм, а можуть адаптуватися до даних і знаходити закономірності, які не були явно задані програмістом.

З практичної точки зору ШІ-системи можна поділити на кілька великих груп залежно від підходів, які використовуються для обробки інформації. Одним із класичних варіантів є символічний (логічний) штучний інтелект, де знання описуються через правила, факти, логічні конструкції, а результат роботи системи отримується шляхом логічного виведення. Такі підходи

історично використовувалися в експертних системах, де механізм прийняття рішення базувався на базі правил типу “якщо–то” та формалізованих знаннях про предметну область [6].

Перевагою символічних систем є їх інтерпретованість, тобто можливість пояснювати, чому саме було прийнято те чи інше рішення. Це є важливою характеристикою для систем підтримки прийняття рішень, де користувач повинен розуміти логіку роботи алгоритму. Разом із тим створення та підтримка баз знань для таких систем часто потребує значних ресурсів і участі експертів предметної області.

Іншим важливим напрямком розвитку є системи на основі машинного навчання, які будують модель поведінки на основі статистичних закономірностей у навчальних даних. Для таких систем характерно, що вони “вчаться” на прикладах і надалі використовують отриману модель для прогнозування або класифікації нових даних [7]. Машинне навчання включає широкий спектр методів — від класичних алгоритмів, таких як лінійна регресія або дерева рішень, до сучасних глибоких нейронних мереж, які здатні працювати з великими обсягами інформації.

У сучасних дослідженнях особливу увагу приділяють глибокому навчанню (deep learning), яке базується на багатошарових нейронних мережах. Такі моделі показують високі результати у задачах комп’ютерного зору, розпізнавання мовлення та обробки природної мови. Однак складність цих моделей часто ускладнює їх інтерпретацію та формальний аналіз, що створює додаткові виклики для забезпечення надійності інтелектуальних систем [11].

Також поширеним є поділ ШІ-систем за типом навчання. Виділяють навчання з учителем, де система отримує правильні відповіді для прикладів; навчання без учителя, де модель шукає структуру в даних без готових міток; а також навчання з підкріпленням, де агент отримує винагороду або штраф за свої дії і поступово навчається оптимальній стратегії поведінки [7]. Останній підхід широко застосовується у задачах керування, робототехніці та ігрових середовищах.

Окрему увагу варто приділити гібридним системам штучного інтелекту, які поєднують різні підходи до обробки інформації. Наприклад, у таких системах нейронні мережі можуть використовуватися для розпізнавання образів або узагальнення даних, тоді як логічні або семантичні моделі забезпечують структуроване представлення знань та логічне виведення [8 – 12]. Подібні підходи дозволяють поєднувати переваги статистичних методів і формалізованих знань, що є перспективним напрямом розвитку сучасного штучного інтелекту.

Ще одним критерієм класифікації є рівень автономності системи. У цьому контексті можна виділити інтелектуальні інформаційні системи, які аналізують дані та надають рекомендації користувачу, а також інтелектуальні агентні системи, які здатні самостійно виконувати послідовність дій і досягати поставленої мети. У сучасних дослідженнях активно розвиваються агентні архітектури, де система може планувати дії, взаємодіяти з інструментами та коригувати свою поведінку залежно від отриманих результатів [5].

Крім того, системи на базі штучного інтелекту можна класифікувати за їх функціональним призначенням. Зокрема, виділяють системи підтримки прийняття рішень, інтелектуальні системи управління, системи аналізу даних, а також інтерактивні системи, такі як чат-боти, голосові асистенти та системи питання–відповіді. Усі ці типи систем використовують різні моделі подання даних та алгоритмічні підходи залежно від специфіки задачі [1].

Узагальнюючи, можна сказати, що системи на базі штучного інтелекту відрізняються не лише за алгоритмами, які вони використовують, але й за способом подання знань та даних. Саме тому при проектуванні та аналізі таких систем важливо враховувати, які моделі подання інформації застосовуються і наскільки вони коректні та узгоджені. Це безпосередньо пов'язано з необхідністю формальної верифікації моделей подання даних, адже помилки або суперечності у представленнях можуть впливати на якість і надійність всієї інтелектуальної системи.

1.3 Архітектура сучасних ІІІ-систем

Архітектура сучасних систем на базі штучного інтелекту (ШІ) є багаторівневою та включає не лише інтелектуальну модель, але й усі компоненти, які забезпечують отримання, обробку та використання даних. На практиці ефективність ШІ-системи залежить не тільки від обраного алгоритму або нейронної мережі, але й від того, наскільки правильно побудований загальний процес роботи системи: від збору інформації до формування кінцевого результату для користувача або іншої програмної підсистеми [1 – 10].

У загальному вигляді архітектуру ШІ-систем можна представити як послідовність основних блоків: вхідні дані → обробка і підготовка → модель ШІ → інтерпретація результатів → вихід і інтеграція. Такий підхід часто описують як структуру “Input – Model – Output”, що підкреслює системний характер роботи штучного інтелекту та залежність результатів від якості та форми вхідної інформації [4]. При цьому кожен із блоків має власні функції та може складатися з декількох модулів.

Першим етапом роботи ШІ-системи є збір і надходження даних, які можуть отримуватися з різних джерел: баз даних організацій, сенсорів, інтернет-ресурсів, користувацьких запитів, текстових документів або мультимедійних файлів. У сучасних умовах дані часто є неоднорідними за структурою, тому виникає необхідність їх перетворення у формат, який можна використовувати для навчання моделі або виконання аналізу [2].

Наступним блоком архітектури є попередня обробка даних (data preprocessing). Вона включає очищення даних, усунення пропусків, нормалізацію, видалення дублікатів, виявлення шуму або очевидних помилок. Саме на цьому етапі часто вирішується, наскільки якісною буде робота всієї системи, адже навіть потужна модель машинного навчання може показувати погані результати, якщо дані є некоректними або суперечливими. У сучасних дослідженнях цей аспект розглядається як один із ключових у межах підходу data-centric AI, де робота з даними вважається таким же важливим елементом, як і вибір алгоритму [2].

Окремо в архітектурі можна виділити модуль представлення даних і формування ознак, який відповідає за перетворення інформації у вигляд, зручний для обробки моделлю. Наприклад, для табличних даних можуть використовуватися числові ознаки, для текстів — векторні подання, а для структурованих знань — графові або онтологічні моделі. Саме вибір способу подання даних впливає на те, які закономірності зможе виявити модель та як точно вона буде працювати [3].

Центральним компонентом архітектури є модель штучного інтелекту, яка може бути реалізована у вигляді алгоритмів машинного навчання, нейронних мереж, логічних правил або гібридних підходів. На цьому рівні система виконує основні обчислення: класифікацію, прогнозування, генерацію відповідей або прийняття рішення. Часто модель навчається на історичних даних і після цього використовується у режимі застосування, коли на нових даних система формує результат. При цьому важливо враховувати, що модель може потребувати періодичного донавчання або оновлення, якщо предметна область змінюється або з'являються нові типи даних [7 – 11].

Після отримання результату роботи моделі в архітектурі ШІ-системи часто присутній модуль постобробки та інтерпретації результатів. Він може включати приведення результатів до зрозумілого формату, фільтрацію, ранжування або перетворення прогнозів у конкретні рекомендації. Особливо актуальною є проблема пояснюваності (explainability), коли користувач або розробник має розуміти, чому система прийняла певне рішення. Це є важливим для підвищення довіри до ШІ-систем та для використання їх у критичних сферах, таких як медицина або безпека [1 – 13 – 20].

Фінальним елементом архітектури є інтеграція ШІ-модуля в загальну інформаційну систему. На цьому рівні результат передається у вигляді відповіді користувачу, запису в базу даних, запуску певної дії або передачі до іншого компонента. У сучасних умовах ШІ-системи часто працюють як частина більшого сервісу, тому важливою стає взаємодія між модулями, використання API, забезпечення швидкодії та масштабованості [4 – 19].

Окремо варто виділити розвиток агентних архітектур, де система може виконувати не одну дію, а цілий ланцюг кроків для досягнення мети. Наприклад, агент може аналізувати запит, формувати план, звертатися до зовнішніх інструментів, накопичувати проміжні результати та коригувати поведінку залежно від ситуації. У сучасних роботах *agentic AI*-системи описуються як поєднання модулів пам'яті, планування, інструментальної взаємодії та мовної моделі, що робить їх більш автономними порівняно з класичними ШІ-рішеннями [5].

Таким чином, архітектура сучасних ШІ-систем є комплексною та включає етапи отримання, підготовки, представлення даних, використання моделі, обробки результатів і інтеграції у загальну інфраструктуру. Саме тому при аналізі таких систем важливо враховувати не лише алгоритми, але й моделі подання даних, які використовуються для навчання та логічного обґрунтування роботи ШІ. Це створює основу для подальшого дослідження питань коректності та формальної верифікації моделей подання даних у системах на базі штучного інтелекту.

1.4 Поняття та роль моделей подання даних у ШІ-системах

У сучасних системах на базі штучного інтелекту дані виступають одним із ключових ресурсів, адже саме вони використовуються для навчання моделей, формування висновків і отримання результатів роботи інтелектуальної системи. Проте для ефективного використання даних недостатньо лише їх зберігати або передавати між компонентами системи. Важливим питанням є те, у якому вигляді дані представлені, тобто яка модель подання даних застосовується в конкретній ШІ-системі. Модель подання даних можна визначити як формальний або напівформальний спосіб опису інформації, який задає структуру, формат, зв'язки та правила роботи з даними, необхідні для їх подальшої обробки [1 – 12].

Роль моделей подання даних у ШІ-системах є надзвичайно важливою, оскільки від способу представлення інформації залежить, які саме закономірності буде здатна знайти модель, наскільки повно система зможе

описати предметну область, а також наскільки ефективно дані можуть бути використані для навчання або логічного виведення. Наприклад, одна і та сама інформація може бути подана у вигляді таблиці, набору текстових документів або графової структури, і кожен варіант буде надавати різні можливості для аналізу та різні обмеження. У цьому сенсі модель подання даних виконує функцію посередника між реальними даними та методами штучного інтелекту, які їх аналізують [14].

У широкому розумінні подання даних у ШІ-системах охоплює декілька елементів. По-перше, це структура даних, тобто спосіб організації інформації (наприклад, таблиці, графи, набори фактів, логічні правила). По-друге, це семантика даних, тобто зміст та значення, які пов'язуються з конкретними об'єктами та їх характеристиками. По-третє, це вимоги до якості та узгодженості, оскільки дані можуть містити пропуски, суперечності, помилки або неточності. У разі використання таких даних в інтелектуальній системі вони можуть викликати некоректне навчання або неправильні рішення [2].

В архітектурі сучасних ШІ-систем моделі подання даних застосовуються на кількох рівнях. Перший рівень — це вхідні дані, які надходять до системи з різних джерел. На цьому етапі важливо визначити, у якому форматі інформація буде зчитуватися, оброблятися та зберігатися. Другий рівень — це етап підготовки даних, коли виконується очищення, нормалізація та перетворення інформації у вигляд, придатний для навчання моделі або виконання аналізу. Третій рівень — це представлення даних усередині моделі, де дані можуть бути подані у вигляді числових ознак, векторів або структур знань [4].

У сучасній практиці можна виділити кілька основних причин, чому питання моделей подання даних має вирішальне значення. По-перше, це вплив на якість результатів: неправильне або недостатньо інформативне подання даних може зменшувати точність моделі, навіть якщо алгоритм обрано правильно. По-друге, це вплив на швидкодію та масштабованість: наприклад, графові структури можуть бути корисними для опису зв'язків між сутностями,

але вимагати більше ресурсів для обробки. По-третє, це вплив на інтерпретованість результатів: у деяких випадках логічні або онтологічні моделі дозволяють краще пояснювати рішення системи, ніж суто статистичні векторні підходи [3 – 12 – 20].

Окремо варто зазначити, що в сучасних ШІ-системах активно розвиваються підходи, пов'язані із переходом від простого збереження даних до їх більш змістовного опису. Зокрема, знаннєві граfi є прикладом моделей подання даних, які дозволяють інтегрувати інформацію з різних джерел, формалізувати зв'язки між поняттями та підтримувати виконання запитів і логічного аналізу. У таких моделях дані описуються як набір вершин (сутностей) та ребер (зв'язків), що формує структуровану семантичну основу для подальших ШІ-застосувань [3].

Крім того, у межах сучасних систем часто поєднуються різні моделі подання даних. Наприклад, табличні дані можуть використовуватись для зберігання числових показників, графові структури — для опису взаємозв'язків, а векторні подання — для обробки текстів або зображень. Такий підхід дозволяє забезпечити більш повну картину предметної області, однак ускладнює підтримку узгодженості між різними формами представлення інформації. Це означає, що зростає ймовірність появи суперечностей або помилок, які можуть впливати на коректність функціонування інтелектуальної системи [2].

Отже, моделі подання даних у ШІ-системах є важливим компонентом, який визначає якість, надійність і можливості інтелектуальної системи. Від того, наскільки правильно обрано спосіб представлення даних, залежить не лише ефективність навчання моделей, але й здатність системи працювати з інформацією у складних реальних умовах. Саме тому у подальших підрозділах доцільно розглянути основні підходи до подання даних у ШІ-системах та проблеми, які виникають при забезпеченні їх коректності і узгодженості.

1.5 Основні підходи до подання даних у системах штучного інтелекту

У системах на базі штучного інтелекту дані можуть бути представлені в різних формах залежно від типу задачі, джерел інформації та методів обробки. Спосіб подання даних має важливе значення для ефективності роботи інтелектуальної системи, оскільки саме від структури даних залежить можливість виявлення закономірностей, швидкість обробки інформації та точність отриманих результатів [1].

Моделі подання даних визначають, яким чином інформація про об'єкти предметної області, їх властивості та взаємозв'язки буде представлена у системі. Вибір відповідного підходу впливає не лише на ефективність алгоритмів машинного навчання або логічного виведення, але й на можливість подальшого аналізу та перевірки коректності системи [3].

У сучасних системах штучного інтелекту застосовується кілька основних підходів до подання даних. Найбільш поширеними серед них є логічні моделі, графові та онтологічні моделі, а також статистичні й векторні підходи. Кожен із цих підходів має власні особливості та використовується залежно від характеру задачі та типу інформації, з якою працює система.

Логічні моделі дозволяють описувати знання у формалізованому вигляді за допомогою логічних правил і тверджень. Графові та онтологічні моделі забезпечують представлення складних взаємозв'язків між об'єктами та використовуються для побудови знаннєвих баз і семантичних структур. Векторні та статистичні моделі, у свою чергу, широко застосовуються у сучасних методах машинного навчання для аналізу великих масивів даних [7].

Таким чином, різні підходи до подання даних дозволяють ефективно вирішувати широкий спектр задач у системах штучного інтелекту. Разом з тим використання різних моделей представлення інформації потребує забезпечення їх узгодженості та коректності, що зумовлює необхідність застосування методів формальної верифікації.

1.5.1 Логічні моделі подання даних

Логічні моделі подання даних є одним із найбільш формалізованих підходів у системах штучного інтелекту. У таких моделях знання описуються

за допомогою логічних тверджень, правил та відношень між об'єктами. Найчастіше для цього використовуються формальні мови, що базуються на логіці висловлювань або логіці предикатів. Завдяки цьому можна формально описувати структуру предметної області та виконувати логічний аналіз отриманих знань [1 – 9].

У логічних моделях дані подаються у вигляді фактів і правил. Факти описують властивості об'єктів або подій, тоді як правила визначають залежності між різними твердженнями. Наприклад, правило типу “якщо–то” дозволяє системі формувати нові висновки на основі вже відомих фактів. Подібний підхід широко використовувався у класичних експертних системах, де база знань містила формалізовану інформацію про предметну область, а механізм логічного виведення забезпечував отримання нових знань [6].

Однією з головних переваг логічних моделей є можливість формального аналізу та перевірки узгодженості знань. Використання математичної логіки дозволяє перевіряти, чи не містить база знань суперечностей, а також визначати, чи можна отримати певний висновок на основі заданих правил. Крім того, такі моделі забезпечують високий рівень пояснюваності результатів, оскільки процес отримання рішення можна представити як послідовність логічних міркувань [9].

Разом із тим створення великих логічних баз знань може бути складним і потребує значних ресурсів, оскільки знання необхідно формалізувати вручну. Також логічні моделі менш ефективні при роботі з великими масивами неструктурованих даних, де більш доцільно застосовувати статистичні методи машинного навчання [7].

Таким чином, логічні моделі подання даних забезпечують формалізований опис знань та можливість їх математичного аналізу, що робить їх важливим інструментом у задачах побудови інтелектуальних систем та формальної верифікації.

1.5.2 Графові та онтологічні моделі

Графові та онтологічні моделі подання даних широко застосовуються у сучасних системах штучного інтелекту, оскільки дозволяють представляти інформацію у вигляді взаємозв'язків між різними об'єктами. На відміну від традиційних табличних моделей, у яких дані зберігаються у вигляді окремих записів, графові структури дають змогу безпосередньо відображати відношення між сутностями предметної області. Це особливо важливо для систем, що працюють зі складними структурами знань [3].

У графових моделях дані представляються у вигляді графа, який складається з вершин і ребер. Вершини відповідають окремим об'єктам або поняттям, а ребра описують зв'язки між ними. Такий підхід дозволяє ефективно відображати структуру взаємодії між різними елементами інформації та аналізувати залежності між ними.

Одним із найбільш поширених прикладів використання графових моделей є знаннєві графи (knowledge graphs). Вони застосовуються у пошукових системах, системах рекомендацій, а також у задачах обробки природної мови. Знаннєві графи дозволяють інтегрувати інформацію з різних джерел та формувати єдину структуру знань, що полегшує подальший аналіз даних і пошук нових зв'язків між об'єктами [3].

Онтологічні моделі можна розглядати як розширення графових підходів. Вони забезпечують формалізований опис предметної області, який включає класи об'єктів, їх властивості та відношення між ними. Онтології дозволяють будувати структуровані бази знань і використовувати механізми логічного виведення для отримання нових знань на основі вже наявної інформації [12].

Однією з важливих переваг онтологічних моделей є можливість забезпечення семантичної узгодженості даних. Завдяки формальному опису понять і їх взаємозв'язків система може більш точно інтерпретувати інформацію та уникати неоднозначностей при роботі з даними.

Разом із тим графові та онтологічні моделі можуть містити суперечності або некоректні зв'язки між об'єктами. Подібні проблеми можуть виникати при інтеграції даних з різних джерел або при автоматичному формуванні

знаннях баз. Тому для забезпечення коректності таких моделей часто застосовуються формальні методи перевірки, які дозволяють виявляти помилки у структурі знань та забезпечувати їх узгодженість [8].

Таким чином, графові та онтологічні моделі подання даних є важливим інструментом представлення знань у системах штучного інтелекту. Вони дозволяють ефективно описувати складні взаємозв'язки між об'єктами та підтримують семантичний аналіз інформації.

1.5.3 Векторні та статистичні моделі

Векторні та статистичні моделі подання даних є основою більшості сучасних методів машинного навчання. У таких моделях інформація представляється у вигляді числових векторів ознак, які використовуються для навчання алгоритмів та виконання прогнозування [7].

У задачах машинного навчання дані часто подаються у вигляді таблиць, де кожен об'єкт описується набором характеристик. Сукупність значень цих характеристик формує вектор даних, на основі якого алгоритми можуть виконувати різні задачі, зокрема класифікацію, регресію або кластеризацію.

У сучасних системах штучного інтелекту широко застосовуються векторні представлення даних, відомі як *embeddings*. Вони дозволяють представляти складні об'єкти, наприклад тексти або зображення, у вигляді числових векторів у багатовимірному просторі. Це дає можливість застосовувати нейронні мережі та інші алгоритми машинного навчання для аналізу складних типів даних [11].

Перевагою статистичних моделей є здатність ефективно працювати з великими обсягами інформації та автоматично виявляти закономірності у даних. Саме тому такі моделі широко використовуються у задачах аналізу даних, обробки природної мови та системах рекомендацій.

Водночас статистичні підходи мають обмеження, пов'язані з недостатньою пояснюваністю складних моделей. Тому у сучасних дослідженнях активно розвиваються гібридні підходи, які поєднують методи машинного навчання з формальними моделями представлення знань [8 – 16].

Таким чином, векторні та статистичні моделі подання даних відіграють важливу роль у сучасних системах штучного інтелекту, забезпечуючи ефективну обробку великих масивів інформації.

1.6 Проблеми коректності та надійності моделей подання даних

Під час розроблення систем на базі штучного інтелекту важливим завданням є забезпечення коректності та надійності моделей подання даних. Оскільки інтелектуальні системи працюють з великими обсягами інформації та складними структурами знань, помилки у представленні даних можуть призводити до некоректної роботи алгоритмів і зниження точності результатів. Сучасні дослідження також показують, що проблеми якості даних та архітектурні недоліки можуть призводити до накопичення технічного боргу в системах машинного навчання, що ускладнює їх підтримку та знижує надійність [19].

Коректність моделі подання даних означає відповідність структури та змісту інформації встановленим правилам і обмеженням. Якщо модель містить логічні суперечності або некоректні зв'язки між об'єктами, це може порушувати процес логічного виведення або впливати на результати роботи алгоритмів машинного навчання [6].

Надійність моделей подання даних пов'язана зі здатністю системи стабільно працювати навіть у випадку змін у даних або умовах функціонування. Якщо структура даних є нестійкою або недостатньо формалізованою, це може призводити до накопичення помилок та зниження ефективності інтелектуальної системи [7].

Однією з поширених проблем є неповнота даних. У багатьох випадках система не має доступу до всієї необхідної інформації, що ускладнює процес аналізу та прийняття рішень. Неповні або відсутні дані можуть призводити до формування неточних моделей і погіршення якості результатів.

Іншою проблемою є дублювання або надлишковість інформації. У складних інформаційних системах дані часто надходять з різних джерел, що може призводити до появи декількох записів, які описують один і той самий

об'єкт. У графових або онтологічних моделях це може ускладнювати структуру знань і впливати на процес аналізу інформації [3].

Також важливою проблемою є суперечливість даних. Різні джерела можуть містити взаємовиключну інформацію про один і той самий об'єкт або подію. У логічних моделях це призводить до порушення узгодженості бази знань, а у статистичних моделях може знижувати точність алгоритмів машинного навчання.

Таким чином, забезпечення коректності та надійності моделей подання даних є важливою умовою ефективної роботи систем штучного інтелекту. Виявлення та усунення помилок у структурах даних дозволяє підвищити точність алгоритмів і забезпечити стабільність функціонування інтелектуальних систем.

1.7 Аналіз помилок і суперечностей у моделях подання даних

У процесі побудови та використання моделей подання даних у системах штучного інтелекту можуть виникати різні типи помилок, які впливають на коректність функціонування системи. Такі помилки можуть бути пов'язані як зі структурою даних, так і з некоректною інтерпретацією інформації або порушенням логічних обмежень. Їх своєчасне виявлення є важливою умовою забезпечення надійності інтелектуальних систем.

Одним із найбільш поширених типів помилок є логічні суперечності. У логічних моделях подання знань це проявляється у вигляді взаємовиключних тверджень або правил. Наприклад, якщо у базі знань одночасно містяться твердження, що певний об'єкт належить до певного класу і водночас не належить до нього, виникає логічна суперечність. У таких випадках система може формувати некоректні або неоднозначні висновки [9].

У графових та онтологічних моделях суперечності можуть виникати через некоректно визначені зв'язки між об'єктами або порушення ієрархії класів. Наприклад, один і той самий об'єкт може бути віднесений до кількох несумісних категорій. Подібні ситуації часто виникають під час інтеграції

інформації з різних джерел або автоматичного формування знаннєвих графів [3].

Ще одним видом проблем є структурні помилки у даних. Вони можуть проявлятися у вигляді відсутніх атрибутів, неправильних типів даних або некоректних зв'язків між елементами інформації. У системах машинного навчання такі помилки можуть призводити до неправильного формування ознак та погіршення якості навчання моделей [7].

Окремо варто відзначити проблему семантичної неоднозначності. У різних контекстах одна і та сама інформація може інтерпретуватися по-різному, що створює труднощі при об'єднанні даних з різних джерел та побудові узгоджених моделей знань [12].

Для виявлення подібних проблем застосовуються різні методи аналізу моделей даних. До них належать перевірка узгодженості баз знань, аналіз структурних обмежень та використання спеціальних алгоритмів перевірки моделей. Подібні підходи дозволяють автоматично виявляти суперечності та помилки у структурах даних [15 – 17 – 18].

Таким чином, аналіз помилок і суперечностей у моделях подання даних є важливим етапом розроблення інтелектуальних систем. Своєчасне виявлення проблем у структурах даних дозволяє підвищити надійність системи та забезпечити коректність її роботи. Саме тому у сучасних дослідженнях значна увага приділяється застосуванню формальних методів перевірки моделей, які дозволяють системно аналізувати структуру даних і виявляти потенційні помилки ще на етапі проєктування систем.

1.8 Висновки до першого розділу

У першому розділі було проаналізовано основні особливості систем на базі штучного інтелекту та розглянуто підходи до подання даних, які застосовуються у таких системах. Показано, що ефективність роботи інтелектуальних систем значною мірою залежить від способу організації та структурування інформації.

Було розглянуто класифікацію систем штучного інтелекту та основні принципи побудови їх архітектури. Встановлено, що сучасні ШІ-системи зазвичай включають етапи збору даних, їх попередньої обробки, формування ознак, використання моделей машинного навчання та інтеграції результатів у прикладні інформаційні системи.

Особливу увагу приділено моделям подання даних, які визначають структуру інформації та спосіб її використання у системах штучного інтелекту. Розглянуто основні підходи до подання даних, зокрема логічні, графові та онтологічні, а також векторні та статистичні моделі.

Крім того, проаналізовано проблеми коректності та надійності моделей подання даних, а також типові помилки й суперечності, що можуть виникати під час їх використання. Показано, що некоректна структура даних або суперечливі знання можуть негативно впливати на роботу інтелектуальних систем.

Таким чином, забезпечення коректності моделей подання даних є важливою умовою ефективного функціонування систем штучного інтелекту. Це обумовлює необхідність застосування формальних методів аналізу та перевірки моделей даних. У наступному розділі буде розглянуто теоретичні основи формальної верифікації та математичні підходи, що використовуються для перевірки коректності моделей у системах штучного інтелекту.

РОЗДІЛ 2

Реалізація та верифікація моделі даних на основі RDF та SHACL

2.1 Побудова онтології предметної області

У рамках даної роботи ключовим етапом є побудова онтології предметної області, яка слугує основою для формалізованого подання знань у системі. Онтологія визначає структуру предметної області, основні сутності, їх властивості та взаємозв'язки, що дозволяє представити дані у вигляді семантичного графа.

Предметна область, що розглядається, відповідає системі тренажерного залу, в якій основними елементами є користувачі, тренери, тренування, вправи, цілі та обмеження, пов'язані зі станом здоров'я. Формалізація цих елементів здійснюється за допомогою технологій семантичного вебу, зокрема RDF та RDFS.

Базовим класом онтології є клас **fit:Human**, який представляє узагальнену сутність людини. Цей клас використовується як кореневий для всіх антропоморфних об'єктів системи та містить загальні характеристики, такі як вік і стать. Визначення базового класу наведено нижче:

Лістинг 2.1 – Опис класу Human в RDF-онтології

```
fit:Human a rdfs:Class ;
  rdfs:label "Людина" .
```

Для опису індивідуальних характеристик людини використовуються властивості даних (datatype properties), які дозволяють задавати числові та текстові значення. Зокрема, визначено властивості віку та статі:

Лістинг 2.2 – Опис властивостей даних age та gender для класу Human в RDF-онтології

```
fit:age a owl:DatatypeProperty ;
  rdfs:domain fit:Human ;
  rdfs:range xsd:integer .

fit:gender a owl:DatatypeProperty ;
  rdfs:domain fit:Human ;
  rdfs:range xsd:string .
```

На основі базового класу формуються підкласи, які деталізують ролі учасників системи. Клас **fit:User** представляє користувача, який виконує тренування, тоді як клас **fit:Trainer** описує тренера, який створює програми та проводить заняття:

Лістинг 2.3 – Опис підкласів User та Trainer в RDF-онтології

```
fit:User a rdfs:Class ;
  rdfs:subClassOf fit:Human ;
  rdfs:label "Користувач" .

fit:Trainer a rdfs:Class ;
  rdfs:subClassOf fit:Human ;
  rdfs:label "Тренер" .
```

Важливою особливістю онтології є введення логічних обмежень між класами. Зокрема, класи користувача та тренера визначено як неперетинні:

Лістинг 2.4 – Визначення неперетинності класів Trainer та User в RDF-онтології

```
fit:Trainer owl:disjointWith fit:User .
```

Це означає, що одна й та сама сутність не може одночасно належати до обох класів, що забезпечує логічну узгодженість моделі та запобігає некоректному опису ролей.

Окрім класів, що описують людей, в онтології введено класи, які відображають процеси та об'єкти діяльності у тренажерному залі. До них належать:

- **fit:Training** — тренування як цілісна програма або сесія;
- **fit:Exercise** — окрема фізична вправа;
- **fit:TrainingType** — категорія тренування (наприклад, кардіо або силове).

Їх визначення має вигляд:

Лістинг 2.5 – Опис класів Training, Exercise та TrainingType в RDF-онтології

```
fit:Training a rdfs:Class ;
  rdfs:label "Тренування" .

fit:Exercise a rdfs:Class ;
```

```

rdfs:label "Вправа" .

fit:TrainingType a rdfs:Class ;
  rdfs:label "Вид тренування" .

```

Для врахування індивідуальних потреб користувачів у моделі також введено логічні фільтри у вигляді окремих класів. Клас **fit:Goal** описує цілі тренування, такі як схуднення або набір м'язової маси, тоді як клас **fit:HealthCond** відображає стан здоров'я або обмеження, що можуть впливати на вибір вправ:

Лістинг 2.6 – Опис класів Goal та HealthCond в RDF-онтології

```

fit:Goal a rdfs:Class ;
  rdfs:label "Ціль користувача" .

fit:HealthCond a rdfs:Class ;
  rdfs:label "Стан здоров'я / Обмеження" .

```

Таким чином, побудована онтологія визначає повний набір базових сутностей предметної області та створює основу для подальшого опису зв'язків між ними. Використання ієрархії класів, властивостей та логічних обмежень дозволяє забезпечити структурованість і узгодженість моделі, що є необхідною умовою для подальшого застосування методів формальної верифікації.

2.2 Властивості даних, зв'язки та наповнення моделі

У побудованій онтології поряд із визначенням класів важливу роль відіграють властивості, які задають характеристики сутностей та зв'язки між ними. Саме властивості формують семантичну структуру RDF-графа та забезпечують можливість опису як атрибутів об'єктів, так і взаємодії між ними.

2.2.1 Властивості даних (Datatype Properties)

Властивості даних використовуються для задання атрибутів сутностей у вигляді примітивних значень, таких як числа або рядки. У межах даної моделі визначено дві основні властивості, що характеризують базовий клас людини:

- **age** — визначає вік особи та має тип цілого числа;
- **gender** — визначає стать особи та має текстове значення.

Ці властивості застосовуються до всіх екземплярів класу **fit:Human**, що забезпечує уніфікований опис користувачів і тренерів.

Лістинг 2.7 – Опис базових класів, властивостей та обмежень RDF-онтології

```
fit:Human a rdfs:Class ; rdfs:label "Людина" .

fit:age a owl:DatatypeProperty ; rdfs:domain fit:Human ; rdfs:range
xsd:integer .
fit:gender a owl:DatatypeProperty ; rdfs:domain fit:Human ; rdfs:range
xsd:string .

fit:User a rdfs:Class ; rdfs:subClassOf fit:Human ; rdfs:label "Користувач".
fit:Trainer a rdfs:Class ; rdfs:subClassOf fit:Human ; rdfs:label
"Тренер" .

fit:Trainer owl:disjointWith fit:User .
fit:Goal a rdfs:Class ; rdfs:label "Ціль користувача" .
fit:Training a rdfs:Class ; rdfs:label "Тренування" .
fit:TrainingType a rdfs:Class ; rdfs:label "Вид тренування" .
fit:Exercise a rdfs:Class ; rdfs:label "Вправа" .
fit:HealthCond a rdfs:Class ; rdfs:label "Стан здоров'я / Обмеження".
```

Застосування властивостей даних дозволяє описати індивідуальні характеристики об'єктів без ускладнення структури графа, оскільки значення таких властивостей не потребують створення окремих вузлів.

2.2.2 Зв'язки між класами (Object Properties)

Основу моделі становлять об'єктні властивості, які встановлюють зв'язки між різними класами та формують логічну структуру графа знань.

У моделі визначено такі ключові зв'язки:

- **hasPersonalGoal** (Користувач → Ціль) — визначає індивідуальну мету користувача;
- **hasRestriction** (Користувач → Обмеження) — відображає медичні або фізичні обмеження;
- **hasGoal** (Тренування → Ціль) — встановлює відповідність тренування певній меті;
- **consistsOf** (Тренування → Вправа) — описує склад тренування;
- **isContraindicatedFor** (Вправа → Обмеження) — визначає протипоказання;

- **conductsTraining** (Тренер → Тренування) — вказує відповідального тренера;
- **category** (Тренування → Вид тренування) — класифікує тип навантаження.

Лістинг 2.8 – Опис об'єктних властивостей та зв'язків між класами в RDF-онтології

```
fit:hasPersonalGoal a rdf:Property ;
    rdfs:domain fit:User ;
    rdfs:range fit:Goal ;
    rdfs:comment "Зв'язує конкретного користувача з його поточною фітнес-ціллю" .

fit:hasGoal a rdf:Property ;
    rdfs:domain fit:Training ;
    rdfs:range fit:Goal ;
    rdfs:comment "Поєднує тренування з ціллю (схуднення або маса)" .

fit:category a rdf:Property ;
    rdfs:domain fit:Training ;
    rdfs:range fit:TrainingType ;
    rdfs:comment "Визначає вид тренування (кардіо, силові тощо)" .

fit:consistsOf a rdf:Property ;
    rdfs:domain fit:Training ;
    rdfs:range fit:Exercise ;
    rdfs:comment "Список вправ, що входять у тренування" .

fit:hasRestriction a rdf:Property ;
    rdfs:domain fit:User ;
    rdfs:range fit:HealthCond .

fit:isContraindicatedFor a rdf:Property ;
    rdfs:domain fit:Exercise ;
    rdfs:range fit:HealthCond ;
    rdfs:label "Протипоказано при" .
```

Особливу роль відіграє властивість **isContraindicatedFor**, яка використовується для встановлення зв'язку між вправами та медичними обмеженнями. Саме цей зв'язок є критичним для подальшої реалізації механізмів формальної верифікації, оскільки він дозволяє виявляти небезпечні поєднання.

2.2.3 Наповнення моделі (екземпляри)

Після визначення структури онтології виконується її наповнення конкретними екземплярами, що формують реальний граф знань. Екземпляри представляють конкретних користувачів, вправи, тренування та обмеження.

Зокрема, визначено набір вправ та їх протипоказань:

Лістинг 2.9 – Визначення протипоказань для вправ у RDF-онтології

```
fit:Running fit:isContraindicatedFor fit:KneePain, fit:LumbarHernia .
fit:Burpees fit:isContraindicatedFor fit:KneePain, fit:LumbarHernia .
fit:Deadlift fit:isContraindicatedFor fit:LumbarHernia, fit:Stenosis .
fit:PushUps fit:isContraindicatedFor fit:Stenosis .
```

Ці зв'язки формують базу для перевірки безпеки виконання вправ.

Далі визначено користувачів із їх характеристиками:

Лістинг 2.10 – Опис екземплярів користувачів та їх характеристик у RDF-онтології

```
fit:UserIvan a fit:User ;
  rdfs:label "Іван Петренко" ;
  fit:gender "Чоловік" ;
  fit:age 40 ;
  fit:hasPersonalGoal fit:WeightLoss ;
  fit:hasRestriction fit:KneePain, fit:LumbarHernia .

fit:UserMaria a fit:User ;
  rdfs:label "Марія Сидоренко" ;
  fit:gender "Жінка" ;
  fit:age 28 ;
  fit:hasPersonalGoal fit:MuscleGain ;
  fit:hasRestriction fit:LegPain, fit:Stenosis .
```

Також визначено цілі, типи тренувань та вправи:

Лістинг 2.11 – Опис екземплярів цілей, типів тренувань та вправ у RDF-онтології

```
fit:WeightLoss a fit:Goal ; rdfs:label "Схуднути" .
fit:MuscleGain a fit:Goal ; rdfs:label "Накачатися" .

fit:CardioType a fit:TrainingType ; rdfs:label "Кардіо" .
fit:StrengthType a fit:TrainingType ; rdfs:label "Силове тренування" .
fit:HIITType a fit:TrainingType ; rdfs:label "Високоінтенсивний інтервальний тренінг" .

fit:PushUps a fit:Exercise ; rdfs:label "Віджимання" .
fit:Burpees a fit:Exercise ; rdfs:label "Берпі" .
```

```
fit:Running a fit:Exercise ; rdfs:label "Біг" .
fit:Deadlift a fit:Exercise ; rdfs:label "Станова тяга" .
```

Обмеження користувачів задаються через окремі сутності:

Лістинг 2.12 – Опис екземплярів обмежень стану здоров'я у RDF-онтології

```
fit:KneePain a fit:HealthCond ; rdfs:label "Біль у коліні" .
fit:LegPain a fit:HealthCond ; rdfs:label "Біль у нозі" .
fit:LumbarHernia a fit:HealthCond ; rdfs:label "Грижа поперекового відділу" .
fit:Stenosis a fit:HealthCond ; rdfs:label "Стеноз" .
```

Окремо визначено тренування та їх склад:

Лістинг 2.13 – Опис екземплярів тренувань та їх складу в RDF-онтології

```
fit:FatBurner1 a fit:Training ;
  rdfs:label "Інтенсивне спалювання жиру" ;
  fit:hasGoal fit:WeightLoss ;
  fit:category fit:CardioType ;
  fit:consistsOf fit:Running, fit:Burpees .

fit:MassBuilder1 a fit:Training ;
  rdfs:label "Базове силове тренування" ;
  fit:hasGoal fit:MuscleGain ;
  fit:category fit:StrengthType ;
  fit:consistsOf fit:Deadlift, fit:PushUps .
```

Також враховано роль тренерів у системі:

Лістинг 2.14 – Опис властивості conductsTraining та екземплярів тренерів у RDF-онтології

```
fit:conductsTraining a rdf:Property ;
  rdfs:domain fit:Trainer ;
  rdfs:range fit:Training ;
  rdfs:comment "Показує, яке тренування проводить конкретний тренер" .

fit:TrainerAlex a fit:Trainer ; rdfs:label "Олександр Шварц" ;
fit:conductsTraining fit:MassBuilder1 .
fit:TrainerOlena a fit:Trainer ; rdfs:label "Олена Спортивна" ;
fit:conductsTraining fit:FatBurner1 .
```

Модель також розширено додатковими вправами та їх обмеженнями:

Лістинг 2.15 – Розширення моделі: додаткові вправи, їх протипоказання та включення до складу тренувань

```

fit:Plank a fit:Exercise ; rdfs:label "Планка" .
fit:Elliptical a fit:Exercise ; rdfs:label "Еліптичний тренажер" .
fit:BicepCurl a fit:Exercise ; rdfs:label "Згинання рук на біцепс" .
fit:LegPress a fit:Exercise ; rdfs:label "Жим ногами в тренажері".

fit:Plank fit:isContraindicatedFor fit:LumbarHernia .

fit:LegPress fit:isContraindicatedFor fit:KneePain, fit:Stenosis .
fit:Elliptical fit:isContraindicatedFor fit:LegPain .

fit:FatBurner1 fit:consistsOf fit:Elliptical, fit:Plank .
fit:MassBuilder1 fit:consistsOf fit:BicepCurl, fit:LegPress, fit:Plank .

```

На основі визначених класів, властивостей та екземплярів формується цілісний RDF-граф знань, який відображає всі взаємозв'язки між елементами предметної області.

Граф знань на рисунку 2.1 дозволяє наочно представити структуру моделі, зокрема зв'язки між користувачами, їх цілями, тренуваннями, вправами та обмеженнями. Це прощує аналіз моделі та є основою для подальшої формальної верифікації.

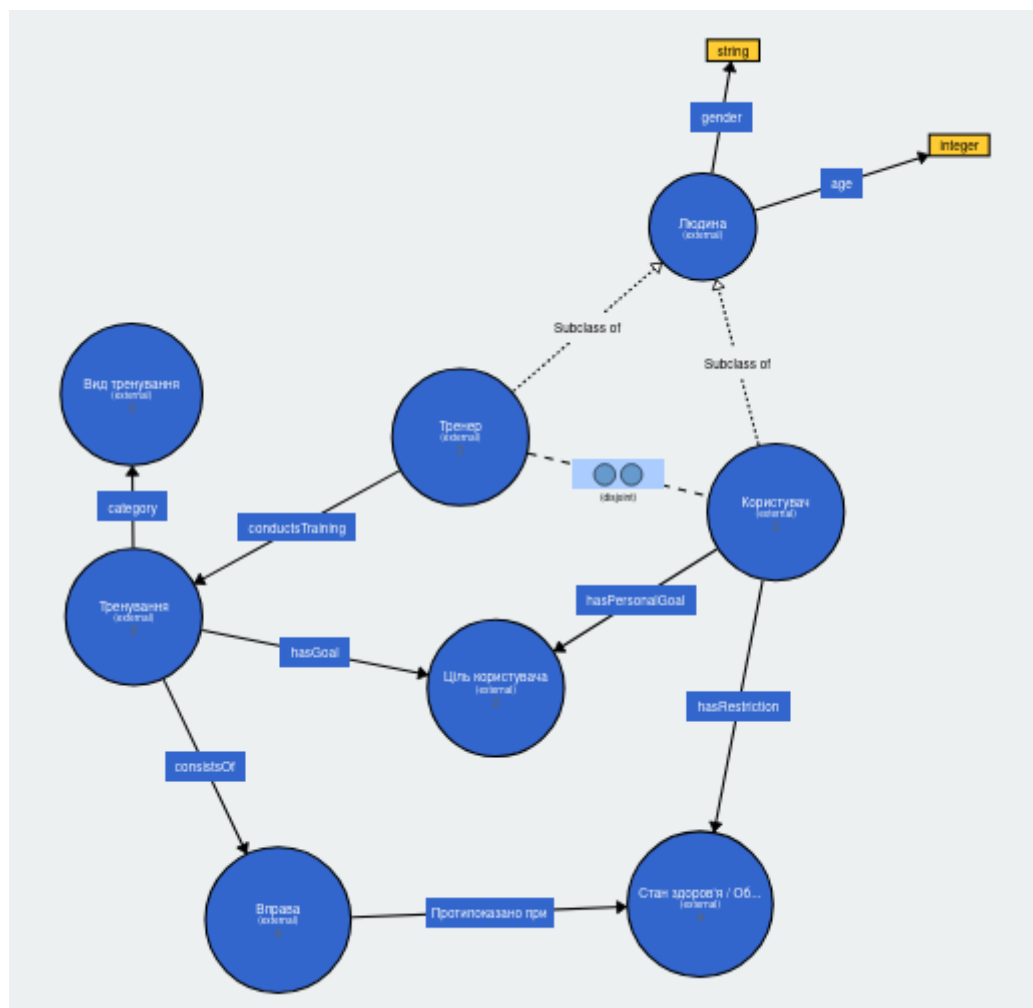


Рисунок 2.1 – Граф знань

Таким чином, у даному підрозділі сформовано повну структуру моделі, яка включає:

- атрибути сутностей (властивості даних);
- логічні зв'язки між об'єктами;
- конкретні екземпляри, що формують реальний граф знань.

Отриманий граф є цілісною моделлю предметної області, яка відображає як структуру даних, так і логіку взаємодії між елементами. Саме ця структура використовується на наступних етапах для виконання запитів та реалізації механізмів формальної верифікації.

2.3 Отримання результатів за допомогою SPARQL та формальна верифікація моделі за допомогою SHACL

Після побудови онтології предметної області та формування RDF-графа знань наступним етапом є практичне використання моделі. Для цього в роботі реалізовано два ключові механізми: отримання даних за допомогою мови запитів SPARQL та перевірка узгодженості моделі за допомогою SHACL. Такий підхід дозволяє не лише працювати з даними, але й забезпечити їх коректність і безпечність.

2.3.1 Використання SPARQL для отримання результатів

Мова SPARQL використовується для виконання запитів до RDF-графа та отримання необхідної інформації на основі заданих умов. У контексті даної роботи SPARQL застосовується для формування рекомендацій вправ для користувачів з урахуванням їхніх цілей та обмежень.

Основна ідея полягає в тому, щоб знайти такі вправи, які:

- відповідають цілі користувача;
- входять до відповідного тренування;
- не суперечать його обмеженням.

Для реалізації цієї логіки використано наступний запит:

Лістинг 2.16 – SPARQL-запит для підбору вправ з урахуванням цілей та обмежень користувача

PREFIX fit: <http://example.org/fitness#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

```
SELECT ?userName ?exerciseLabel
WHERE {
    ?user rdfs:label ?userName ;
        fit:hasPersonalGoal ?goal .

    ?training fit:hasGoal ?goal ;
        fit:consistsOf ?exercise .
    ?exercise rdfs:label ?exerciseLabel .

    FILTER NOT EXISTS {
        ?user fit:hasRestriction ?restriction .
        ?exercise fit:isContraindicatedFor ?restriction .
    }
}
```

У даному запиті спочатку визначається користувач та його ціль тренування. Далі знаходяться тренування, що відповідають цій цілі, а також вправи, які входять до складу цих тренувань. Ключовим елементом є конструкція `FILTER NOT EXISTS`, яка виключає ті вправи, що мають протипоказання, які збігаються з обмеженнями користувача.

Таким чином, запит повертає лише ті вправи, які є безпечними та релевантними для конкретного користувача.

Результати виконання запиту наведено на рисунку 2.2

The screenshot shows the SPARQL Playground interface. On the left, there is a 'Query' tab with a SPARQL query. On the right, there is a 'Results' tab showing the output of the query. The output is displayed in a table format with two columns: 'userName' and 'exerciseLabel'.

Query:

```
PREFIX fit: <http://example.org/fitness#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?userName ?exerciseLabel
WHERE {
    ?user rdfs:label ?userName ;
        fit:hasPersonalGoal ?goal .

    ?training fit:hasGoal ?goal ;
        fit:consistsOf ?exercise .
    ?exercise rdfs:label ?exerciseLabel .

    FILTER NOT EXISTS {
        ?user fit:hasRestriction ?restriction .
        ?exercise fit:isContraindicatedFor ?restriction .
    }
}
```

Results:

userName	exerciseLabel
"Іван Петренко"	"Еліптичний тренажер"
"Марія Сидоренко"	"Згинання рук на біцепс"
"Марія Сидоренко"	"Планка"

Рисунок 2.2 – Результат виконання SPARQL-запиту для підбору безпечних вправ

Отримані результати демонструють, що для кожного користувача сформовано набір вправ, які відповідають його цілі та не суперечать обмеженням за станом здоров'я. Наприклад, для користувача з обмеженнями щодо колін та поперекового відділу автоматично виключаються вправи, що можуть бути небезпечними.

Деталізоване представлення отриманих результатів наведено на рисунку 2.3

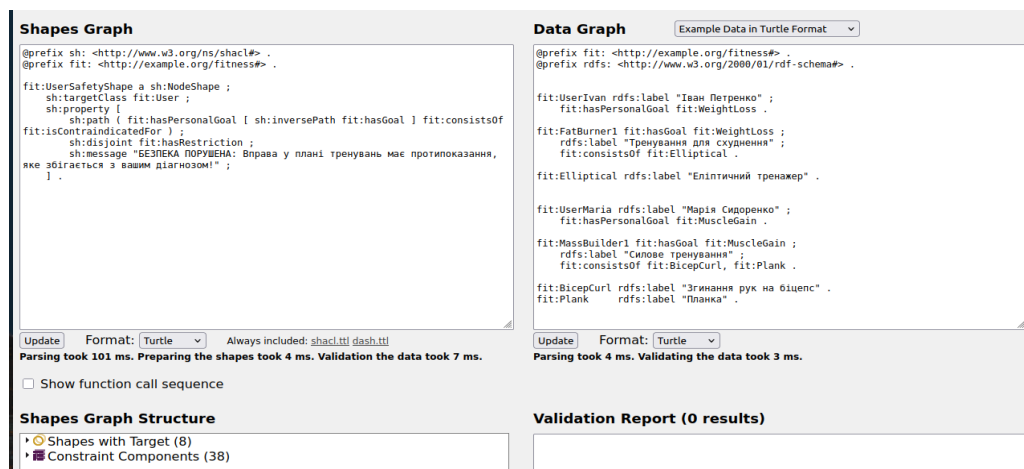


Рисунок 2.3 – Візуалізація результатів виконання SPARQL-запиту

2.3.2 Використання SHACL для формальної верифікації

Після отримання результатів важливим етапом є перевірка узгодженості моделі та виявлення можливих логічних конфліктів. Для цього використовується мова SHACL (Shapes Constraint Language), яка дозволяє задавати обмеження для RDF-графів та автоматично перевіряти їх виконання.

У даній роботі SHACL застосовується для перевірки безпеки тренувань, тобто для виявлення ситуацій, коли користувачу може бути рекомендована вправа, що суперечить його обмеженням.

Для реалізації цієї перевірки визначено наступну SHACL-форму:

Лістинг 2.17 – SHACL-форма для перевірки безпеки тренувань користувача

```

@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix fit: <http://example.org/fitness#> .

fit:UserSafetyShape a sh:NodeShape ;
  sh:targetClass fit:User ;
  sh:property [
    sh:path ( fit:hasPersonalGoal [ sh:inversePath fit:hasGoal ]
fit:consistsOf fit:isContraindicatedFor ) ;
    sh:disjoint fit:hasRestriction ;
    sh:message "БЕЗПЕКА ПОРУШЕНА: Вправа у плані тренувань має
протипоказання, яке збігається з вашим діагнозом!" ;
  ] .

```

У даній формі визначено обмеження для всіх екземплярів класу `fit:User`. Основною особливістю є використання складеного шляху (`sh:path`), який дозволяє виконати навігацію через кілька рівнів графа: від користувача до його цілі, далі до тренування, вправ і їх протипоказань.

Логіка перевірки має вигляд:

Користувач → Ціль → Тренування → Вправа → Протипоказання

Оператор `sh:disjoint` визначає, що множина протипоказань вправ не повинна перетинатися з множиною обмежень користувача. Якщо такий перетин існує, система фіксує порушення та генерує відповідне повідомлення.

Результати виконання SHACL-перевірки наведено на рисунку 2.3.2.

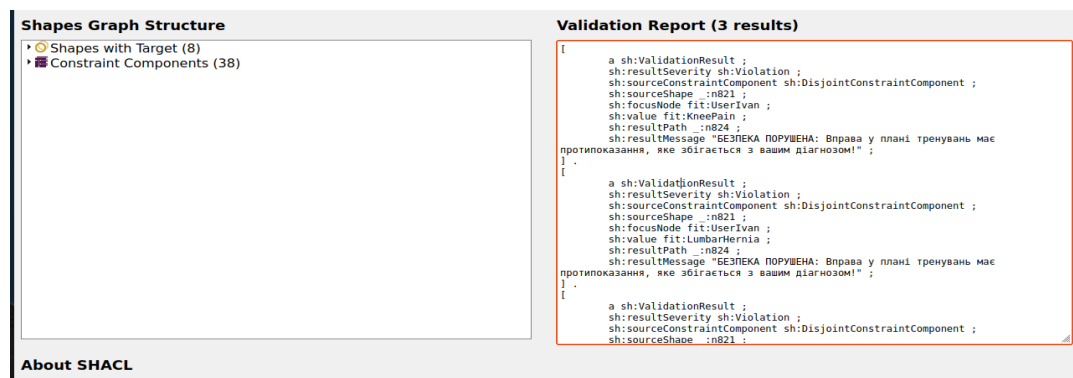


Рисунок 2.4 – Результати формальної верифікації моделі за допомогою SHACL

Аналіз результатів показує, що система успішно виявляє випадки, коли вправи суперечать обмеженням користувачів. Це підтверджує коректність реалізованого механізму верифікації та його придатність для забезпечення безпеки рекомендацій.

Таким чином, у даному підрозділі реалізовано повний цикл роботи з моделлю даних: від отримання інформації за допомогою SPARQL до її формальної перевірки засобами SHACL. Використання SPARQL дозволяє ефективно витягувати релевантні дані з RDF-графа, тоді як SHACL забезпечує контроль узгодженості та виявлення потенційно небезпечних ситуацій.

Поєднання цих технологій забезпечує не лише функціональність системи, але й її надійність, що є критично важливим для інтелектуальних систем, які працюють із рекомендаціями та враховують обмеження користувачів.

2.4 Висновки до другого розділу

У другому розділі було виконано формалізацію моделі подання даних предметної області тренажерного залу з використанням технологій семантичного вебу. Побудовано онтологію, яка включає класи, властивості та екземпляри, що дозволяє представити дані у вигляді RDF-графа знань.

У межах моделі визначено основні сутності предметної області, зокрема користувачів, тренерів, тренування, вправи, цілі та обмеження. Встановлено властивості даних для опису характеристик об'єктів, а також об'єктні властивості, що формують зв'язки між ними та визначають логіку взаємодії елементів системи.

На основі побудованої онтології сформовано граф знань, який відображає структуру предметної області та взаємозв'язки між її компонентами, що створює основу для подальшого аналізу та обробки даних.

У процесі практичної реалізації моделі спочатку було використано мову запитів SPARQL для отримання результатів з RDF-графа. Реалізовані запити дозволили сформулювати рекомендації вправ для користувачів з урахуванням їхніх цілей та обмежень, виключаючи небезпечні варіанти.

На наступному етапі застосовано мову SHACL для формальної верифікації моделі. Реалізовані обмеження дозволили автоматично перевірити узгодженість даних та виявити можливі конфлікти між вправами та обмеженнями користувачів.

Отримані результати підтверджують, що побудована модель забезпечує як можливість отримання релевантних даних, так і контроль їх коректності. Поєднання SPARQL та SHACL дозволяє реалізувати повний цикл роботи з даними — від їх отримання до формальної перевірки.

Таким чином, у другому розділі продемонстровано, що використання семантичних технологій та формальних методів дозволяє забезпечити узгодженість, безпечність і надійність моделей подання даних у системах на базі штучного інтелекту.

ВИСНОВКИ

У дипломній роботі було досліджено теоретичні та практичні аспекти формальної верифікації моделей подання даних у системах на базі штучного інтелекту.

У першому розділі проведено аналіз систем штучного інтелекту, їх класифікації та архітектурних особливостей. Розглянуто основні підходи до подання даних, зокрема логічні, графові та онтологічні моделі. Встановлено, що ефективність функціонування ШІ-систем значною мірою залежить від коректності організації та узгодженості моделей даних. Також проаналізовано проблеми, пов'язані з суперечностями та помилками в моделях подання даних, що можуть призводити до некоректної роботи інтелектуальних систем.

У другому розділі було розроблено онтологічну модель предметної області тренажерного залу з використанням технологій семантичного вебу. Модель реалізовано у форматі RDF-графа знань, що дозволило формалізувати сутності предметної області, їх властивості та взаємозв'язки.

У межах розробленої моделі визначено класи, об'єктні та дататипні властивості, а також сформовано набір екземплярів, що відображають реальні сценарії використання системи. Побудований граф знань забезпечує структуроване та формалізоване представлення даних, придатне для подальшого аналізу.

За допомогою мови SPARQL реалізовано отримання результатів на основі побудованої моделі. Запити дозволяють формувати рекомендації вправ для користувачів з урахуванням їхніх цілей та обмежень, виключаючи варіанти, що суперечать стану здоров'я.

Для перевірки коректності моделі було застосовано мову SHACL, за допомогою якої реалізовано механізм формальної верифікації. Проведена перевірка дозволила виявити конфлікти між вправами та обмеженнями користувачів, що підтверджує здатність моделі автоматично визначати некоректні або небезпечні конфігурації даних.

Отримані результати свідчать про те, що використання формальних методів, зокрема RDF, SPARQL та SHACL, дозволяє забезпечити коректність, узгодженість і надійність моделей подання даних у системах штучного інтелекту.

Таким чином, поставлена мета дипломної роботи досягнута, а всі визначені завдання виконані. Розроблена модель може бути використана як основа для побудови інтелектуальних систем рекомендацій та подальших досліджень у галузі формальної верифікації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Russell S. Artificial Intelligence: A Modern Approach / S. Russell, P. Norvig. — 4th ed. — Pearson, 2021.
2. Mazumder M. Data-Centric Artificial Intelligence: A Survey / M. Mazumder et al. // ACM Computing Surveys. — 2025.
3. Hogan A. Knowledge Graphs / A. Hogan et al. // ACM Computing Surveys. — 2021.
4. Park S. AI System Architecture Design Methodology Based on IMO (Input–Model–Output) / S. Park, J. Lee, J. Lee // Data & Knowledge Engineering. — 2024.
5. Abou Ali M. Agentic AI: A Comprehensive Survey of Architectures, Applications, and Future Directions / M. Abou Ali, F. Dornaika. — 2025.
6. Giarratano J. Expert Systems: Principles and Programming / J. Giarratano, G. Riley. — 4th ed. — Cengage Learning, 2005.
7. Bishop C. Pattern Recognition and Machine Learning / C. M. Bishop. — Springer, 2006.
8. d’Avila Garcez A. Neural-Symbolic Cognitive Reasoning / A. d’Avila Garcez, L. C. Lamb, D. Gabbay. — Springer, 2009.
9. Enderton H. A Mathematical Introduction to Logic / H. B. Enderton. — Academic Press, 2001.
10. Nilsson N. The Quest for Artificial Intelligence / N. J. Nilsson. — Cambridge University Press, 2010.
11. Goodfellow I. Deep Learning / I. Goodfellow, Y. Bengio, A. Courville. — MIT Press, 2016.
12. Brachman R. Knowledge Representation and Reasoning / R. Brachman, H. Levesque. — Morgan Kaufmann, 2004.
13. Huth M. Logic in Computer Science: Modelling and Reasoning about Systems / M. Huth, M. Ryan. — Cambridge University Press, 2004.
14. Jackson D. Software Abstractions: Logic, Language, and Analysis / D. Jackson. — MIT Press, 2012.
15. Wooldridge M. An Introduction to MultiAgent Systems / M. Wooldridge. — Wiley, 2009.
16. Mendelson E. Introduction to Mathematical Logic / E. Mendelson. — CRC Press, 2015.
17. Baader F. Description Logics as Ontology Languages for the Semantic Web / F. Baader, I. Horrocks, U. Sattler. — Springer, 2004.
18. Cormen T. Introduction to Algorithms / T. Cormen, C. Leiserson, R. Rivest, C. Stein. — MIT Press, 2009.
19. Sculley D. Hidden Technical Debt in Machine Learning Systems / D. Sculley et al. // Advances in Neural Information Processing Systems. — 2015. — Vol. 28.

20. Doshi-Velez F. Towards A Rigorous Science of Interpretable Machine Learning / F. Doshi-Velez, B. Kim // arXiv preprint arXiv:1702.08608. — 2017.20. Doshi-Velez F., Kim B. Towards A Rigorous Science of Interpretable Machine Learning. arXiv preprint arXiv:1702.08608. 2017.