Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження та застосування знання-орієнтованого підходу до розробки рекомендаційних систем
(тема)

Виконав:
студент 2 курсу, групи СШІм-22-3
Жукотинський Т.Г.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)
Керівник проф. Рябова Н.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____» _____ 20 ____ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Жукотинському Тимуру Геннадійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та застосування знання-орієнтованого підходу до розробки рекомендаційних систем

затверджена наказом університету від 1 квітня 20 24 р. № 260Ст
2. Термін подання студентом роботи до екзаменаційної комісії 6 червня 20 24 р.
3. Вихідні дані до роботи Огляд та аналіз методів побудови рекомендаційних систем, заснованих на знаннях. Розробка прототипу рекомендаційної системи, заснованої на знаннях у предметній області формування тактиктичного плану на футбольний матч. Перелік використовуваних програмних засобів: ОС Ubuntu, середовище розробки PhpStorm, середовище Postman. Технічне забезпечення: ноутбук зі встановленим програмним середовищем PhpStorm

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Теоретичні дослідження та проектування
3) Характеристика предметної галузі
4) Розробка прототипу рекомендаційної системи

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	01.04.2024	виконано
2	Аналіз предметної галузі	02.04.2024 – 04.04.2024	виконано
3	Постановка задач дослідження	05.04.2024 – 07.04.2024	виконано
4	Характеристика предметної області	08.04.2024 – 10.04.2024	виконано
5	Проблематика предметної області	11.04.2024 – 13.04.2024	виконано
6	Аналіз методів побудови РС	14.04.2024 – 15.04.2024	виконано
7	Аналіз Constrained-Based методу	16.04.2024 – 18.04.2024	виконано
8	Аналіз Case-Based методу	19.04.2024 – 23.04.2024	виконано
9	Огляд використаного середовища	24.04.2024 – 28.04.2024	виконано
10	Реалізація методів рекомендації	29.04.2024 – 07.05.2024	виконано
11	Перевірка роботи системи	08.05.2024 – 10.05.2024	виконано
12	Написання пояснювальної записки	11.05.2024 – 20.05.2024	виконано
13	Підготовка презентації	21.05.2024 – 26.05.2024	виконано
14	Попередній захист	28.05.2024	виконано
15	Захист перед ЕК	06.06.2024	

Дата видачі завдання 1 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Рябова Н.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 90 с., 29 рис., 10 формул, 2 дод., 21 джерело.

МЕТОДИ ІНЖЕНЕРІЇ ЗНАНЬ, МЕТОДИ ПОДАННЯ ЗНАНЬ, РЕКОМЕНДАЦІЙНІ СИСТЕМИ, CASE-BASED REASONING, CONSTRAINT-BASED RECOMMENDATIONS.

Об’єкт дослідження – рекомендаційні системи, засновані на знаннях.

Предмет дослідження – методи побудови рекомендаційних систем, заснованих на знаннях.

Мета роботи – аналіз та застосування методів побудови рекомендаційних систем, заснованих на знаннях в предметній області кулінарних рецептів.

Методи дослідження – методи системного аналізу, методи машинного навчання, методи інженерії знань, аналіз рекомендаційних систем, побудова прототипу рекомендаційної системи, заснованої на знаннях за допомогою інтегрованого середовища розробки PhpStorm, Postman, мови програмування PHP на базі фреймворку Laravel.

В результаті проведених досліджень, вирішено задачу порівняння та аналізу різноманітних методів побудови рекомендаційних систем, заснованих на знаннях, створено прототип. Дана розробка може бути перспективною до подальшого розвитку у використанні у футбольному світі.

ABSTRACT

Master’s thesis contains: 90 pp., 29 fig., 2 ann., 21 references.

CASE-BASED REASONING, CONSTRAINT-BASED RECOMMENDATIONS, KNOWLEDGE ENGINEERING, KNOWLEDGE PRESENTATION, RECOMMENDER SYSTEMS.

The object of research is recommendation systems based on knowledge.

The subject of research is methods of building recommendation systems based on knowledge.

The purpose of the work is the analysis and application of methods of building recommendation systems based on knowledge in the subject area of culinary concepts.

Research methods – methods of system analysis, machine learning methods, methods of knowledge engineering, analysis of recommender systems, construction of a prototype of a recommendation system based on knowledge using an integrated environment such as PhpStorm, Postman, PHP programming language based on the Laravel framework.

As a result of the research, the task of comparing and analyzing various methods of building recommendation systems based on knowledge was solved, and a prototype was created. This sample may be promising for further development in use in the football world.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CB – Case-Based – на основі вмісту;

CBR – Case Based Reasoning – міркування на основі прецедентів;

CF – Collaborative Filtering – колаборативна фільтрація;

CKB – Constraint Knowledge Base – база знань обмежень;

IDE – Integrated Development Environment – середовище розробки;

KBRS – Knowledge-Based Recommender System – рекомендаційна система, заснована на знаннях;

PHP-ML – PHP Machine Learning – бібліотека машинного навчання на мові PHP;

RS – Recommender System – рекомендаційна система.

ВСТУП

В сучасному світі, коли автоматизація набирає достатніх обертів, все більше функцій підлягають переведенню у цифровий світ. Інтелектуальний аналіз даних зараз актуальна галузь застосування для нових ІТ проектів.

Інтелектуальний аналіз даних – це напрям машинного навчання та застосування математичних методів для обробки даних. У симбіозі предметів аналіз даних та машинне навчання, де складову становлять алгоритми обробки та можливість видавати результат на основі досвіду, отримана дуже важлива сфера для вирішення проблем людини.

На основі інтелектуального аналізу даних стають можливими розробки автоматизованих рекомендаційних систем в котрі інтегровані будь-які алгоритми отримання та обробки даних. Прикладом простої рекомендаційної системи є проходження тестів, де на основі їх виводиться результат користувачу. Тільки на відміну від простого опитування та заздалегідь запрограмованих відповідей, є можливість створення гнучкої системи, котра буде використовувати зібрані дані та вчитись на основі їх видавати результат.

Машинне навчання, як частина штучного інтелекту вже не є новітніми технологіями, кожен день вони збагачуються новими методами, що більш характерні для предметної області кваліфікаційної роботи. Наприклад обробка зображень, коли ми проходимо капчу від Google, аби зберегти сайт від атаки ботів. Штучний інтелект глибоко використовується в життєдіяльності людства задля забезпечення точності та виключення людського фактору. Навіть туристичний літак керується за допомоги автопілота, або таксі без водія. Це тільки маленька частина сфер життя, куди проник штучний інтелект.

Одна з можливостей інтелектуального аналізу даних це класифікація. Класифікація – це метод знаходження відношення одного об’єкту до класу об’єктів даних виходячи з попереднього набору інформації. Класифікаційне

прогностичне моделювання – завдання наближення функції відображення (f) від вхідних змінних (x) до дискретних вихідних змінних (y).

Наприклад, рекомендаційна система на основі опитування можна визначити як вирішення задачі класифікації. Це класифікація на декілька класів даних. Класифікатор використовує початковий пул навчальних даних, аби зробити висновки, за яких умов шуканий екземпляр можна віднести до одного з класів. Навчальними даними виступають заздалегідь підготовлений набір відповідей з попередньою класифікацією за результатом (рекомендацією). Підготований класифікатор, що навчений на прикладах готовий до визначення результату за опитуванням людини.

Існує багато застосувань для класифікації у багатьох сферах, таких як схвалення кредитів, медична діагностика, цільовий маркетинг тощо.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз предметної галузі

Національна збірна з футболу – двадцять дві людини, які з'їжджаються з усіх точок світу, щоб з гордістю виступити в головній команді країни. Це водночас складна та проста футбольна команда. Є широкий вибір найкращих гравців, щоб представляти збірну в десятку матчів за рік, і в той же час надто обмежені терміни, щоб підготувати команду до виступу [6]. Ціна помилки велика, тому що регулярні турніри, як Євро або Чемпіонат Світу проходять раз на чотири роки. Для того, щоб команда регулярно давала результати, потрібно багато аналізувати та наперед продумати план дій, щоб перемогти невелику групу суперників в певному проміжку часу.

В порівнянні з клубним футболом, календар матчів нерегулярний: посеред сезону, в міжсезоння, на тиждень, на місяць. Навіть для найкращих гравців немає гарантії, що вони будуть грати в матчах. Через те, що матчі між країнами відбуваються групами: одразу кілька матчів за одну сесію ігор, є можливість дуже довгих подорожей: зіграти домашній матч в Києві, потім два гостьових в Португалії та на Кіпрі. Виконання інших завдань на полі через те, що на одну позицію перенасичення гравців, а на іншій недостатньо якості. Повне завантаження емоцій, від чого може бути, як підвищена нервовість, так і емоційний запал. Нерідко трапляються випадки, що за клуб футболіст грає на одному рівні, а за національну збірну зовсім на іншому.

Тобто футболісти виходять зі звичного кола подій та переміщуються у стресовий період, де потрібно викластись на повну, швидко познайомитись з командою, тренером та спробувати інший футбол. Здебільшого футболісти грають не тільки в різних клубах, а й в різних країнах, де стиль футболу також має особливості: в Німеччині швидкий вертикальний футбол, в Італії більш захисна модель, тощо.

На рисунку 1.1 наведено показовий приклад збірної України, де стартовий склад зібраний з футболістів що грають в 3 країнах та 8 клубах, де тільки 4 одноклубники з Донецького Шахтаря. Виходячи з географії та статусу клубів, виділяється і стиль гри, до якого призвичаїлись гравці, і відповідальність, яку вони несуть в кожному матчі.

Рисунок 1.1 – Стартовий склад збірної з урахуванням клубів

Шахтар є гегемоном на внутрішній арені, але погано витримує навіть середню інтенсивність і відбивається від дійсно сильних команд в єврокубках. Команда намагається грати в атакуючий футбол, але не може зрівнятись індивідуально талантом при грі з командами з інших чемпіонатів де підвищена конкуренція та інтенсивність. Тому все точиться на персоналіях, де в захисті головним є Матвієнко, а в атаці все вирішується через дирижера атак Судакова, інші ж більше виконують спроби підходу до воріт та пошуку моментів.

Англійські клуби перш за все мають високу інтенсивність: більше 40 ігор за сезон, а разом з єврокубками більше 50. Внезалежності від задач клубу: виграти чемпіонат чи залишитись в АПЛ на наступний сезон, в чемпіонаті спостерігається високий рівень конкуренції, від чого залежить і загальна фізична підготовка і розвиток гравця для того, щоб стабільно займати стартове місце в складі. Тому 4 гравці, що грають в АПЛ дуже знадобляться будь-якій футбольній збірній, щоб дати більше швидкості та техніки, а також не боятись демонструвати особисті вміння.

Іспанські клуби, де грають двоє футболістів зі стартового складу, та ще двоє на лаві запасних, також виділяються інтенсивністю та конкуренцією, на додачу того, що там більш значущою частиною команди є півзахист. Саме від ядра команди будуються усі атакувальні та захисні дії. Також ставка робиться на талановитих виконавців, і дуже важливо в тренувальному процесі розробити кілька дійових зв’язок, що зможуть постійно штурмувати ворота супротивника, виснажуючи захисників.

Отже, зібрати з такого різнобарв’я команду за обмежений термін це досить амбітна задача, яка потребує і тренерського досвіду, і постійної аналітики футболістів, щоб сформувати найкращу версію команди до моменту виходу на поле. Проте на відміну від клубів, тренерський штаб не має багато ресурсів і більшість обов’язків складається на асистентів тренера. Коли в клубному футболі існують окремі відділи скаутів, аналітиків, тренерів різного порядку, що можуть пропрацьовувати індивідуальний

розвиток футболіста, в штабі національної збірної всього кілька людей, яким потрібно вже використати готові навички футболістів.

На рисунку 1.2 наведено приблизну структуру тренерського штабу, де головну роль дирижера команди виконує головний тренер, далі усі процеси реалізовують асистенти, і локально ще допрацьовують більш вузькоспеціалізовані тренери.

Рисунок 1.2 – Структура тренерського штабу збірної

Між ними проходить постійна комунікація в дві сторони, спочатку збір інформації до головного тренера, потім від нього до спеціалістів [7].

Наприклад в Україні головний тренер Сергій Ребров, має іспанський штаб асистентів, та двох українців: Гліба Платова та Рустама Худжамова.

Роль головного тренера – прийняти усю аналітику від асистентів: аналіз супротивника, календар ігор, пропоноване фізичне навантаження для гравців, тощо. На основі усіх даних потрібно сформувати:

- малюнок гри – в якому стилі буде грати команда та за допомоги яких тактичних комбінацій шукати моменти в грі;
- формування складу – календар не завжди є простим, тому варто грамотно врахувати сили і виставляти оптимальні склади, задля досягнення цілей;
- завдання до тренерів – роздати завдання асистентам, щоб вони виконали задачу по організації тренувального процесу та тактичних занять, виходячи з того, який футболіст відіграє роль на полі;
- керування процесів – під час підготовки збірної до гри основним обмеженням виступає час, тому тренер описує процеси та роздає колегам на виконання, за яким потрібно спостерігати.

Роль асистентів та інших тренерів більш локальна і повинна повністю підпорядковуватись вимогам головного тренера. Насправді вони є наглядачами за футболістами, а коучу передають інформацію про кожного з них. Також часто спостерігається той варіант, що може бути друга людина в штабі – перший асистент головного тренера, який набирається від нього досвіду, але має можливість напряму впливати на гру, пропонуючи свої варіанти тактики, вступати в дискусію з коучем та разом з колегами формувати список замін під час гри. Здебільшого асистенти виконують роль головного наглядача над своєю зоною відповідальності: напад, захист, півзахист. Їм потрібно правильно передати думки головного тренера команді, та награти зв’язки, комбінації. Тренер з фізичної підготовки відповідальний за те, щоб привести футболіста в оптимальну форму перед матчем.

1.2 Основні поняття та визначення

1.2.1 Рекомендаційні системи та знання-орієнтований підхід

Рекомендаційна система – інформаційна технологія, що реалізує взаємодію між користувачем і системою на основі запитів та релевантних відповідей. Користувач формує і відправляє питання до системи, яка містить велику кількість шарів фільтрації, аналізує дані і видає персоналізовані рекомендації.

Так як рекомендаційні системи разом з іншими технологіями IT стрімко увійшли в життя людей, їх можна зустріти в широкому колі вжитку. Вони можуть швидко та ефективно надавати персоналізовану інформацію до користувача, що підходить до його вимог. Їх можна легко і нескінченно розширювати, даючи користувачу більшу точність відповіді, покриваючи усі можливі варіанти.

Але маючи на собі багаж великої кількості варіантів і ходів, запрограмованих всередині системи, вони можуть давати значну похибку, плутаючи людей недостовірною інформацією. Від цього можемо зробити висновок, що рекомендаційні системи також мають свій оптимальний розмір, щоб не бути перевантаженими схожими об’єктами, та не змушувати користувача проводити години в процесі заповнення форм [1]. Таким чином можемо визначити умовно дві групи рекомендаційних систем: загальні та спеціалізовані.

Загальні РС мають на меті охопити якомога більший шар інформації з певної сфери знань та видавати максимально абстрактні відповіді, що мало перетинаються між собою. Або реалізують систему, що видає конкретні відповіді у простому форматі з обмеженою кількістю об’єктів, як наприклад Акінатор, що намагається вгадати відому людину чи персонажа. В загальному реалізація таких систем базується на алгоритмах класифікації, деревах, чи фільтруванні інформації.

Спеціалізовані РС мають на меті не просто дати відповідь, що схожа на те, що очікує користувач, а в повній мірі задовольнити його потреби і аргументувати свій вибір. Такі системи можуть мати досить складний формат збору даних, на який потрібно витратити час та максимально чітко описати вимоги, але впротивагу цьому дають досить обширну відповідь, що можна порівняти зі звітом від спеціаліста по аналізу даних. Подібні системи можна спостерігати у медицині, аерофлоті, космічних дослідженнях де діджиталізація набуває найвищого рівня. Для таких систем доволі важко зробити універсальні алгоритми, що будуть відпрацьовувати на великих запитах даних. Тому здебільшого для спеціалізованих РС застосовується експертний, знання-орієнтований підхід.

Як показано на рисунку 1.3, рекомендаційні системи, що використовують знання експертів, мають на меті дати людині рекомендацію на основі її конкретних, але простих запитів.

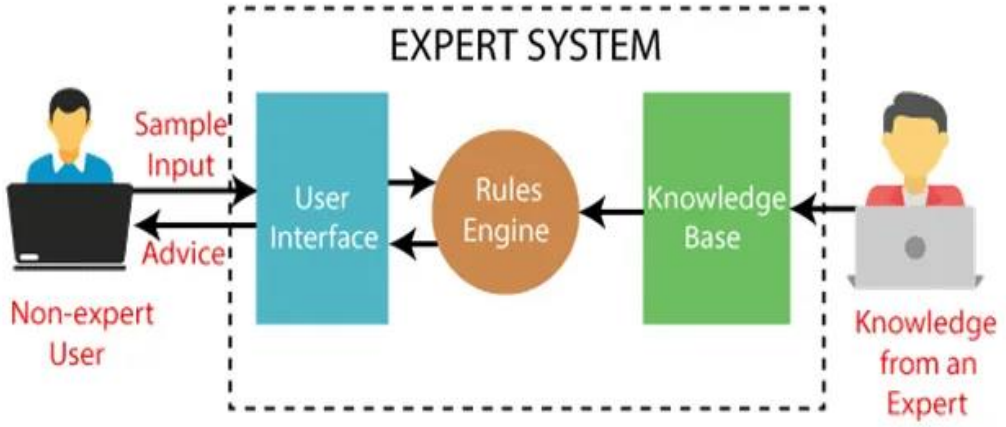

Рисунок 1.3 – Схема роботи експертної рекомендаційної системи

Проте експертні системи мають недолік в людському факторі, і для сфер діяльності, де помилка може коштувати життя, або великих фінансових втрат, неможливо покладатись виключно на експертні знання, і поліпшеною версією цьому є знання-орієнтовані рекомендаційні системи.

Знання-орієнтований підхід використовується для побудови рекомендаційних систем на основі експертних знань та правил для генерації відповідей. В цій реалізації увага надається контексту, додатковій інформації, придбаній на попередньому досвіді та актуалізації бази знань. Спеціалізовані РС орієнтовані на вирішення конкретної задачі з різними вхідними факторами, але результат може бути схожим: надати діагноз, запропонувати маршрут перевірки автоматики в міжпланетних станціях, побудувати стратегію піар-кампанії політиків і т.д.

Подібні системи досить ресурсозатратні, тому що потрібно приділяти увагу до актуалізації бази знань, оновленню аналітичних моделей та алгоритмів обробки різних форматів даних. З іншого боку це дає можливість накопичити знання з багатьох джерел, та мати під рукою в потрібний момент сводку необхідної інформації, щоб користувач не витрачав час на хибні роздуми.

1.2.2 Структура знання орієнтованих рекомендаційних систем

Знання – база знань, що складається з набору великої кількості даних, на які посилається програмне забезпечення під час процесу вирішення задач. Важливу роль в рекомендаційних системах носить постійне оновлення знань, що можна реалізувати за допомоги експертних знань, інтернет джерел, літературою, баз даних [4], [5]. На відміну від експертних рекомендаційних систем, чи систем загального плану, знання не можуть бути універсально формалізовані, тому вони представлені у різному форматі: текст, експертні правила, бази даних, онтології і т.д. Тому під час обробки запитів від користувача можуть бути використані наступні технології:

- текстовий аналіз – обробка статей та інших документів;
- семантичний аналіз – для моделювання взаємозв’язків між елементами знань для покращення розуміння контексту;

– машинний аналіз – використання широкого спектру алгоритмів: кластеризація, класифікація, регресія і т.д.

Аналітичні моделі – виконують роль фільтру при зборі інформації, та виведенні рекомендацій. Збираючи інформацію про користувачів та предмети рекомендаційної системи, будуються аналітичні моделі, для створення зв’язків між предметами та користувачами за допомоги різних алгоритмів. Як і база знань, аналітичні моделі мають піддаватись постійному оновленню, щоб забезпечити більшу точність, релевантність рекомендацій.

Контекстуальна обробка – додатковий компонент, що може бути використаний у зв’язці з аналітичною моделлю, для відстежування контексту з боку користувача: врахування місцезнаходження, часу активності, звичок у використанні додатку, тощо.

Програмне забезпечення – рекомендаційна система може бути самостійним застосунком зі своїм бекендом та інтерфейсом користувача, або ж частиною іншої програми, як рекомендації відео на Youtube.

1.3 Огляд основних підходів до формування рекомендацій

При розробці знання-орієнтованих рекомендаційних систем найбільш часто використовуються наступні два відомих підходи:

– constraint-based (базований на обмеженнях) – підхід, що використовує обмеження для генерації відповідей. Обмеження дозволяють допустимий набір комбінацій параметрів об'єктів. Завдяки вказаним користувачем обмеженням, система може підібрати для нього максимально релевантні та цікаві рекомендації;

– case-based (базований на випадках) – підхід, в якому система формує рекомендації, використовуючи метод аналогій, грунтуючись на аналізі схожих сценаріїв або випадків з минулого досвіду. Вона шукає

ситуації, подібні до поточної, й використовує рекомендації, які виявились успішними в аналогічних випадках раніше.

Метод, при якому користувач взаємодіє з рекомендаційною системою, яка збирає інформацію про обмеження. Система також має певний перелік обмежень, на яких формується база знань про об’єкти. Після певного часу взаємодії з системою, користувач може регулярно отримувати рекомендації. Наприклад, користувач використовує букінг для періодичних подорожей. На основі його вподобань, вбудована рекомендаційна система може визначити необхідний тип об’єктів, що потрібна для задоволення користувача, і чим далі він буде користуватись букінгом, тим менше часу йому знадобиться на пошук оптимального рішення [2].

Переваги даного підходу:

− гнучкість генерації рекомендацій – використання не лише стандартного формату обмежень, як для пошуку за фільтрами, але й більш комплексні обмеження, з автоматичною обробкою за порядком важливості критеріїв;

− простота пояснення вибору – система може розкрити користувачу логіку своїх рекомендацій, пояснюючи, які обмеження й параметри вплинули на їх генерацію. Це робить процес вибору більш прозорим і дає користувачам чітке розуміння причин, чому певні варіанти потрапили до списку рекомендацій.

Недоліки Constraint-Based:

− неповна персоналізація – рівень персоналізації може бути не таким високим, як у системах, які глибоко враховують особисті вподобання користувача;

− ефективність до вирішення точкових задач – зростання кількості або складності обмежень може значно ускладнити процес пошуку, роблячи його більш ресурсоємним. Це може призвести до погіршення продуктивності системи та низької релевантності відповідей.

При таких перевагах та недоліках можна виділити перелік предметних областей, де такі рекомендаційні системи будуть максимально ефективними: підбір подарунків, букінг, побудова маршруту по місту, тощо. Тобто головний уклон йде на точкові рішення, де краще за все працюють обмеження. Тому такі системи мають найкращу ефективність, коли вимоги, або обмеження користувача чітко визначенні, їх досить багато і вони специфічні. І все це в рамках невеликої предметної області, де наявний дискретний набір обмежень.

Case-Based підхід робить рекомендаційні системи більш гнучкими, адже вони не обмежені жорсткими правилами. Замість цього, система може адаптувати свої рекомендації, спираючись на аналіз схожих ситуацій, що робить цей підхід особливо ефективним, коли знання важко формалізувати, або користувачі мають унікальні, індивідуальні потреби.

Переваги даного підходу:

− гнучкість та адаптація – система може змінювати свої рекомендації відповідно до нових умов або контексту, а також використовувати різноманітні критерії для порівняння й визначення схожості між сценаріями. Це дозволяє їй більш точно й персоналізовано підбирати рекомендації для кожного користувача, враховуючи його унікальні потреби та вподобання;

− здатність до навчання – можливість навчатися на основі нових сценаріїв та досвіду, додавання нових випадків до бази знань і, як результат покращення якості рекомендацій з часом;

− робота зі складними предметними областями – на відміну від Constraint-Based систем, де потрібні чітко визначені обмеження, в цьому підході система має можливість працювати з більш розмитими знаннями та охоплювати ширший спектр аналізу, використовуючи для вирішення задач попередній успішний досвід в різних варіаціях.

Недоліки Case-Based:

– залежність від даних – так як система генерує рекомендації на основі досвіду схожих сценаріїв, в базі знань має бути достатня кількість даних, щоб отримати релевантну відповідь, самими алгоритмами та універсальним набором даних не обійтись;

– адаптація – система неспроможна сама відловити та зрозуміти зміни у вподобаннях користувачів, тому замість точкової адаптації, система потребує ледь не повного перепису даних про користувача, щоб розуміти які саме сценарії йому потрібні для розгляду.

Виходячи з вищеописаного, Case-Based більше підходить до вирішення глобальних задач, де потрібно дати рекомендацію в яку сторону рухатись, визначити малюнок рішення, а не дати чітко описану відповідь. Тому найкраще для цього підійдуть наступні рекомендаційні системи: система генерації музики, зображень, система визначення контрабандистів в аеропортах, тощо.

Роль Collaborative filtering у знання-орієнтованих рекомендаційних системах полягає у використанні спільної інформації про користувачів для надання персоналізованих рекомендацій. Фільтр формує велику кількість матриць, де одним компонентом є користувач, а іншим одна з характеристик предмету РС, маючи кілька матриць, проходить їх аналіз і видаються найбільш релевантні з рекомендацій. Таким чином він знаходить схожі матриці, щоб рекомендувати нові об'єкти користувачеві на основі їхніх інтересів та уподобань.

Переваги Collaborative filtering:

– швидка адаптація – можливість системи швидко реагувати на зміни у вподобаннях користувачів, та налаштовувати рекомендації до інших користувачів зі схожими матрицями інтересів;

– здатність до виявлення складних взаємозв’язків – маючи багато матриць користувачів по різних критеріях, система має можливість провести аналіз обширного пласту інформації і виявити буквально будь-які зв’язки між групами користувачів за не очевидними складовими;

− здатність працювати з обмеженою інформацією – система використовує лише дані про взаємодію користувачів з об’єктами, без урахування додаткового опису чи набору параметрів.

Недоліки даного підходу:

− невелика щільність даних – якщо кількість користувачів або об'єктів дуже велика, а взаємодія між ними дуже рідка, виникає проблема недостатньої щільності даних, коли важко надати точні рекомендації;

− проблема старту – коли в системі з’являється новий користувач чи об’єкт, потрібно спочатку зібрати інформацію про інтереси користувача, або залученість об’єкта, перед тим, як видавати релевантні рекомендації;

− завелика щільність даних – система може рекомендувати переважно популярні об'єкти, не враховуючи інтересів користувача.

Тобто Collaborative filtering більше підходить для систем де залучена велика кількість користувачів, які активно користуються послугами сервісу, щоб вбудована рекомендаційна система могла збирати дані, та слугувала додатковим чинником, щоб затримати користувача якомога більше в системі. Показовий приклад цьому соціальні мережі та стрімінгові платформи, що будують інформаційну бульбашку користувачу. Проте цей варіант не робочий для рекомендаційної системи, де обмежена кількість користувачів, або вони рідко користуються системою [3].

Роль даного підходу базується на рекомендаціях, що мають схожі характеристики з об’єктами, що вже знаходяться у вжитку користувача. Content-Based не фокусується на історії взаємодії користувачів з об’єктами, а на персоналізованому поданню інформації, виходячи з того, що властивості об’єктів мають спільне підґрунтя [8].

Переваги підходу:

− персоналізація рекомендацій на базі схожості характеристик;

− незалежність від даних інших користувачів;

− врахування контексту під час перевірки схожих об’єктів.

Недоліки Content-Based:

- обмеження до врахування нової інформації;
- одноманітність рекомендацій – якщо користувачі не розширюють коло інтересів, то рекомендації втрачають можливість різноманіття і фокусуються на певному типі об’єктів;
- проблема холодного старту.

Для цього методу підходить будь-яка рекомендаційна система, де є чітка формалізація ключових властивостей, щоб підібрати схожі об’єкти. Як частина платформи, такі системи легко використовуються інтернет магазинами різних категорій: від косметології до книжок.

1.4 Застосування AI технологій у футбольних клубах

В усіх передових топ клубах широко використовуються AI технології. Про них важко дізнатись з відкритих джерел, проте є багато інсайдів з інтерв’ю, або кулуарного спілкування, яке з періодичною частотою виходить в публічний простір. Здебільшого такі новини мають дуже запізнілий характер, і те, що вони описують з великою вірогідністю було запроваджено кілька років тому. Однозначно можна сказати, що більшість футбольних клубів застосовує AI технології для виконання різних задач, як на футбольному полі, так і за його межами.

Реал Мадрид використовує GPS датчики для відстеження своїх гравців на тренуваннях. Основуючись на зібраній інформації, тренери можуть прогнозувати максимально ефективний проміжок часу футболіста на полі. Також вони мають цілу команду технічних спеціалістів, що пише їм софт для відеоаналізу футбольного матчу, що допомагає їм підготуватись до суперника та виконати роботу над помилками. В сукупності з талановитими гравцями та модерними підходами тренерів, цей клуб за останні 10 років 5 разів підіймав кубок ліги чемпіонів та вигравав кілька національних трофеїв.

Навідміну від прямого застосування ШІ в аналізі матчу, Ліверпуль за допомогою рекомендаційних систем працює зі своєю фан базою. Так як усі фанати, та пересічні вболівальники використовують мобільні застосунки від клубу, там під них формується спеціальний розклад для контенту, щоб заволодіти якомога більшою кількістю уваги та підігріти клієнтів на покупку клубного мерчу. Як результат, вболівальники отримують організований фанатський рух, який завжди супроводжує команду на виїзних матчах і приносить довготривалу вигоду для Ліверпуля в економічному та футбольному сенсі.

Манчестер Сіті і Баварія постійно працюють над оновленням інфраструктури та гоняться за сучасними медичними технологіями, щоб забезпечити найкращу підготовку футболістів до матчів. Звісно це все не може працювати без жорсткої зв’язки з сучасними технологіями, зокрема штучний інтелект.

Для порівняння той же Манчестер Юнайтед, в якого за словами Кріштіану Роналду вся інфраструктура та процеси тренування залишились незмінними за останні 15 років, має зовсім інші результати. Незважаючи на дорогий склад в 800 млн. доларів, команда не може показувати стабільну гру і в останні роки матч на матч не приходиться.

Один з найяскравіших прикладів – бельгійський клуб Royale Union Saint-Gilloise, який використовує велику аналітичну базу для того, щоб витягнути перспективного гравця за найменшу ціну з нижчих футбольних ліг. Як і в багатьох середніх клубах не з топ чемпіонатів, мета Сент-Жиллуазу – виховання молодих за найменшу вартість та продаж за великі гроші. Як наслідок – успішні трансфери якісних гравців: Віктора Боніфейса за 20 млн. євро та Деніза Ундава за 7 млн. євро.

Також відомо і про інші клуби, які заробляють здебільшого на перепродажу своїх талантів: Лілль, Ренн, Аякс та інші. Проте їх технології піддаються меншому оголошенню, але в порівнянні з тим, як ефективно працює їх система, однозначно можна сказати, що вони користуються всіма

можливими способами, аби залишатись конкурентноздатними на ринку: медицина, інфраструктура, і AI технології разом з тотальною діджиталізацією не виключення.

Проте саме застосування знання-орієнтованого підходу може бути застосовано у багатьох задачах, виходячи з проблеми, яку вона вирішує. Як було вказано вище, кожен з варіантів знання-орієнтованого підходу виконує різні задачі, тому для кожного з них виділена певна область:

- constraint-based, базуючись на обмеженнях, може підібрати найкращого футболіста на позицію по заданих атрибутам для найкращого виконання установок на полі. Тобто точково підібрати виконавців, які можуть реалізувати тренерський план гри. Наприклад описати усі атрибути для правого вінгера, який має повністю взяти на себе фланг і проривати оборону супротивника, щоб звільняти місце собі і партнерам;
- case-based, виходячи з попередніх ігор суперника, можна взяти досвід, коли він програвав матчі, та знайти схожий малюнок гри, при котрому гравці зможуть повноцінно діяти на полі без втрат якості;
- за допомоги Collaborative Filtering та Content-Based можливо розглянути підходи інших тренерів, що працювали з командою раніше, та які нетипові рішення використовували в складних або патових ситуаціях.

1.5 Постановка задачі

Метою кваліфікаційної роботи є дослідження використання знання-орієнтованих рекомендаційних систем, та застосування їх на практиці, при розробці системи, що допоможе тренерському штабу ефективно побудувати план на гру: тактика, малюнок гри, вибір персоналій згідно зі схемою на тактичній дошці.

В процесі аналізу предметної області було виділено декілька задач, котрі мають бути виконані в ході кваліфікаційної роботи:

- проаналізувати методи що переважають у знання-орієнтованих рекомендаційних системах;
- виділити їх переваги та недоліки;
- провести аналіз предметної області та існуючих аналогів систем у її контексті;
- перевірити ідею поєднання двох методів в рамках однієї глобальної задачі;
- вирішити задачу підготовки датасету для надання рекомендацій по предметній області;
- розробити прототип рекомендаційної системи з використанням методів на основі обмежень та прецедентів.

В результаті буде отримано прототип рекомендаційної системи, що зможе надавати поради тренерському штабу у формуванні плану на гру.

2 ТЕОРЕТИЧНІ ДОСЛІДЖЕННЯ ТА ПРОЕКТУВАННЯ

2.1 Метод надання рекомендацій Constraint-Based

Рекомендаційна система, що використовує Constrained-Based підхід складається з двох головних глобальних елементів:

- база знань, яка складається з набору двох змінних, що описують повний перелік можливих значень та набору трьох обмежень, що слугують правилами для коректної роботи технічної складової;
- технічний движок, що є реалізацією алгоритму видачі основаного на використання правил (обмежень), які були задані користувачем в ході користування програмою, результатом є видача рекомендації з ґрунтовним поясненням обраного рішення.

2.1.1 Структура бази знань

Для цього підходу база знань поділяється на 5 частин:

- customer properties (V_C);
- product properties (V_{PROD});
- constraints (C_R);
- filter conditions (C_F);
- products/allowed instantiations of product properties (C_{PROD}).

V_C – властивості користувача (клієнта) – змінна, що описує повний можливий перелік усіх значень, якими може скористатись користувач, для того, щоб описати свої вимоги до рекомендаційної системи. Наприклад застосунок дає ряд атрибутів, за якими користувач має зробити вибір, або в ході користування програмою ці атрибути будуть заповнені автоматично. В тематиці футболу, в цьому розділі можуть бути накопичені знання про стиль гри тренера та його звички. Приклад атрибуту роль крайніх захисників в атаці наведено у формулі 2.1:

$$rfba_c = (absent, insurance, half - flank, entire\ flank), \quad (2.1)$$

де V_{PROD} – властивості предметів – така сама змінна, як і V_C, але дає повний перелік для об'єктів предметної області, якими маніпулює користувач. В тематиці футболу це можуть бути характеристики гравців, що детально декомпозують профіль навичок кожного. Приклад атрибуту відбору м'яча для центрбека наведено в формулі 2.2:

$$tb_p = (traditional, normal, aggressive), \quad (2.2)$$

де C_R – обмеження – системно визначений набір обмежень у вимогах користувача, що заздалегідь передбачений рекомендаційною системою. В залежності від складності системи та глибини предметної області, може містити великий шар даних. Вони передбачені спеціально для того, щоб спростити процес керування системою, тому що не завжди користувач може прорахувати усі можливі обмеження, за якими він хоче отримати рекомендацію. Приклад обмеження, наведений у формулі 2.3, що широка лінія захисту можлива тільки в схемі з крайніми захисниками:

$$CR_1: ld_c = wide => fb_c = yes, \quad (2.3)$$

де C_F – умови фільтру – як і попередній елемент, цей встановлює взаємозв'язок між значеннями змінних, тільки не в рамках одного V_C, а у зв'язці з V_{PROD}, тобто між властивостями користувача, та предметами. Ці умови також прописані заздалегідь, експертами, що консультують, при написанні технічного рішення. Наприклад, у формулі 2.4 описана ситуація, коли тренер хоче використовувати захисників при виконанні стандартів або кутових, він має виставити гравців не менше 180 сантиметрів:

$$CF_1 : ucbst_c = yes \Rightarrow cbh_p > 179, \quad (2.4)$$

де C_{PROD} – дозволені екземпляри властивостей продуктів, або ж простіше, продукти. Ця частина вміщує в собі перелік предметів, які описані одним обмеженням у вигляді диз'юнктивної нормальної форми, де елементами виступають змінні з V_{PROD}. Приклад визначення футболіста за атрибутами наведений у формулі 2.5:

$$cbh_p > 179 \wedge pzw_p = attacking. \quad (2.5)$$

На рисунку 2.1 видно повну структуру бази знань для рекомендаційної системи. V_C та V_{PROD} є заздалегідь заготовленими переліками усіх можливих атрибутів та їх значень, які будуть запропоновані користувачу на вибір, для формування робочого середовища. C_R, C_F та C_{PROD} – це автоматично створені обмеження, які допомагають клієнту більш гнучко формувати свої запити для отримання відповідей від рекомендаційної системи [15].

Побудова цієї бази знань виконується технічними спеціаліста і потребує знань не тільки з ШІ, а й з суміжної тематики, тому потребує постійної взаємодії з експертами в предметній області проекту. Для футбольної тематики такими експертами є тренери, та їх команда. В складних системах може існувати додаткове програмне забезпечення, що дозволяє експертам напряму взаємодіяти з базою знань та оновлювати знання. Одним з недоліків використання РС є постійне ручне наповнення, що вимагає додаткових ресурсів, але це компенсується подальшою автоматичною обробкою знань: аналіз, формування рекомендацій з детальним описом та причиною вибору.

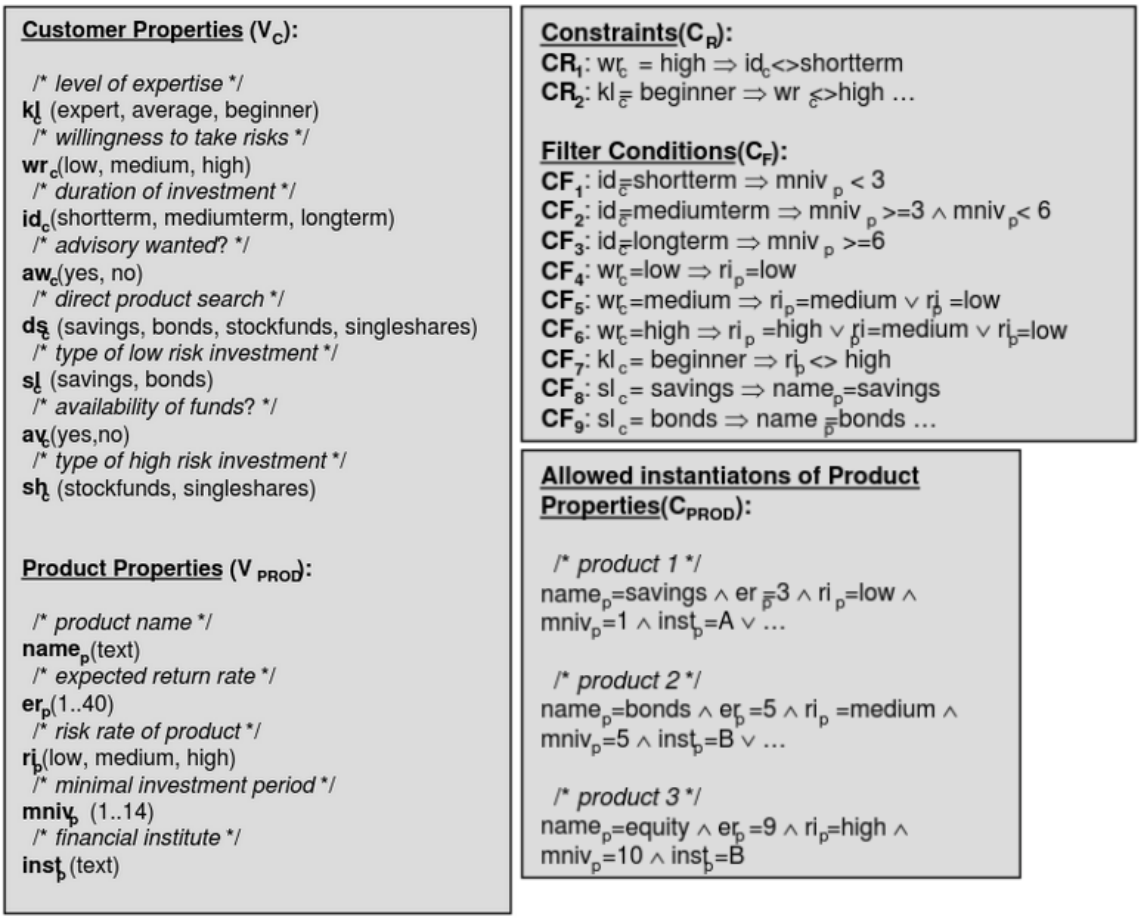

Рисунок 2.1 – Структура бази знань

В додаток до вищеописаної структури бази знань, при вирішенні задачі надання рекомендації, використовується додатковий елемент C_C, який описує вектор атрибутів з переліку V_C, де кожний атрибут має одне значення, з доступних.

2.1.2 Розрахунок рекомендацій

Далі наведено список дефініцій, які в повному обсязі описують як загальний процес розрахунку рекомендацій, так і розбивають його на частини, оглядаючи конкретні формули:

– задачу по розрахунку рекомендацій можна описати, як задачу задоволення обмежень, що описана за формулою 2.6:

$$(V_C, V_{PROD}, C_R \cup C_F \cup C_{PROD} \cup C_C); \quad (2.6)$$

– несуперечлива рекомендація – рекомендація вважається несуперечливою для відповіді РС, що описано за формулою 2.6, якщо кожній змінній у V_C та V_{PROD} призначено значення, яке міститься у запропонованих варіантах;

– процес рекомендації – формула, яку можна описати вектором, представленим у формулі 2.7:

$$(Q, \Sigma, \prod, E, S, F), \quad (2.7)$$

де Q – це скінченний набір станів $\{q_1, q_2, ..., q_j\}$, де кожен стан q_i має змінну $var(q_i) = x_i$ зі скінченною множиною значень, передумови $prec(q_i)$ та післямови $postc(q_i)$, які містять відповідно набори передумов та післямов, а також область значень $dom(x_i)$ для кожної змінної x_i;

Σ – це скінченний набір призначень змінних, де кожне призначення xi = d_{ji} відбувається для змінної xi, яка відповідає стану q_i, і d_{ji} є одним з можливих значень змінної x_i;

$\prod$ – це набір обмежень, які визначають слова приймаються процесом рекомендацій;

E – переходи, що $\subseteq Q \times \prod \times Q$;

$S \subseteq Q$ – це набір початкових станів;

$F \subseteq Q$ – це набір кінцевих станів.

Очевидно, що цей метод підходить більше для вирішення точкових задач, де рекомендація буде будуватись на чітко визначених обмеженнях. При тому вони досить гнучкі у використанні, бо складаються з чотирьох елементів при вирішенні задачі: C_R, C_F, C_{PROD} та C_C. Як наслідок, для цього методу потрібно знайти задачу, інформацію в якій можна в повній мірі розбити на п’ять областей, щоб записати до складових бази знань.

Виходячи з опису Constrained-Based методу, можна зробити висновок, що даний підхід добре підходить для визначення футболіста, якому потрібно зайняти певну позицію на полі. Тренер та його команда можуть декомпозувати свої знання для того, щоб перенести їх до бази знань, після чого використовувати при роботі з системою [14]. Наприклад для визначення атакуючої трійки в схемі 4-3-3. На кожну позицію надати перелік обмежень, яким буде слідувати система при визначенні підходящого кандидата, приклад схеми визначення відповіді наведено на рисунку 2.2, де в залежності від шляху доступні три варіанти.

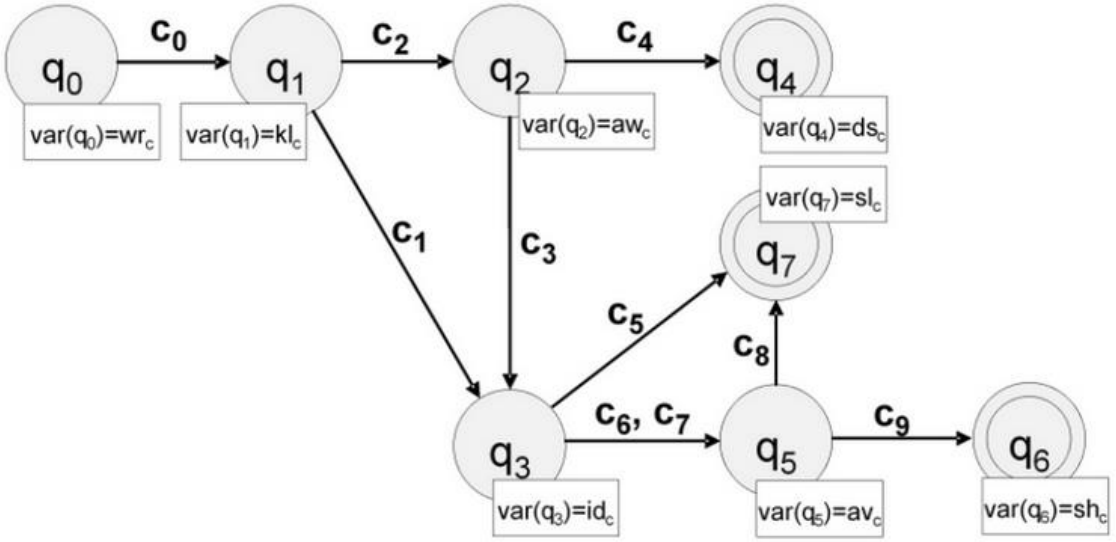

Рисунок 2.2 – Приклад визначення відповіді

2.2 Метод надання рекомендацій Case-Based

Рекомендаційні системи, що використовують цей метод базуються на знаннях, що включають попередній досвід (cases) у вирішенні схожих проблем. Замість того, щоб аналізувати вхідні параметри від користувача за алгоритмом, система шукає співпадіння між запитом користувача та атрибутами попередніх успішних випадках. Цей метод заточений на роботу

з чітко декомпозованою структурною базою знань для того, щоб знайти співпадіння. Приблизну схему роботи наведено на рисунку 2.3.

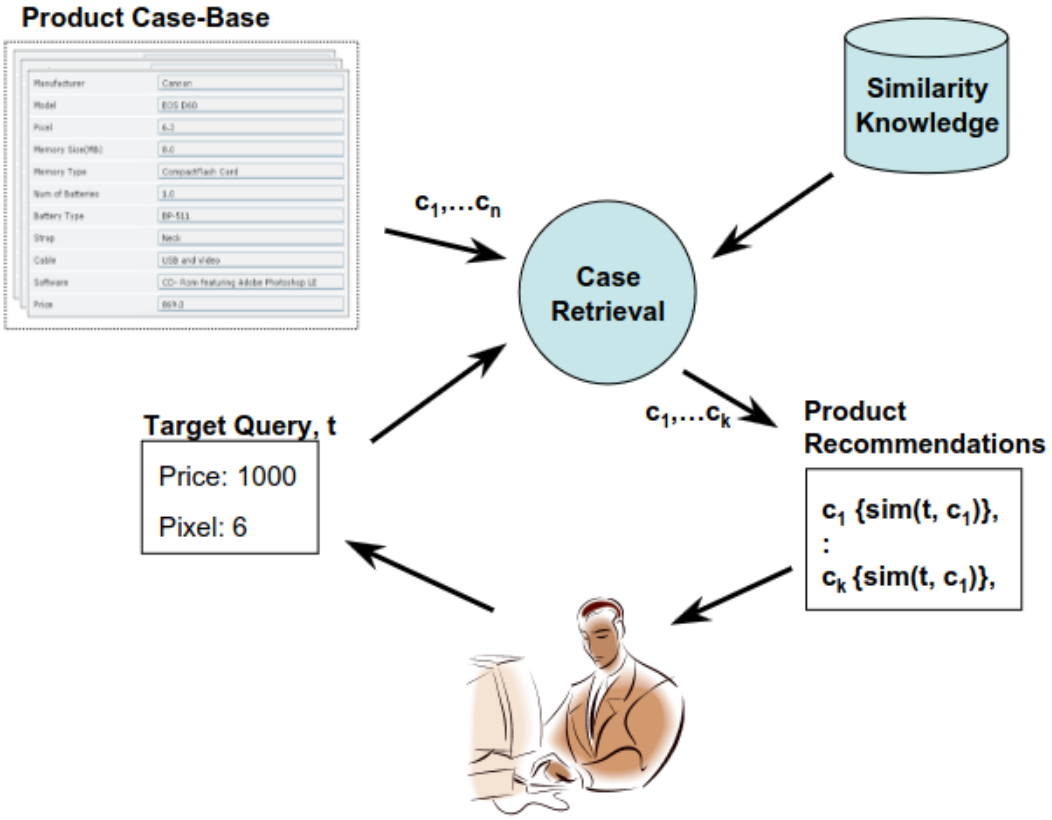

Рисунок 2.3 – Узагальнена схема роботи Case-Based РС

Сам метод Case-Based базується на його попереднику – Case-Based reasoning (CBR). Ранній прототип, що задав напрям розвитку для теперішнього методу, був більш обмежений в реалізації і жорстко покладався на використання бази знань (case base). Тобто під час прийняття рішення, вони враховують саме конкретний попередній досвід, замість того, щоб аналізувати саму форму запиту [10]. Кожний окремий випадок розбивається на кілька частин, з яких виділяється основна інформація, що приведена до уніфікованого вигляду, і саме з цими частинами порівнюється нова ситуація, до якої система намагається підібрати рішення.

Видно, що основними складовими для рекомендаційної системи є база

прецедентів (Case Base) та набір технологій, що займається розрахунком рекомендацій (Similarity Knowledge). На відміну від Constrained-Based, в цьому методі база знань обмежується одним джерелом інформації, всередині якого знання про випадки розбиті на атрибути. Проте в Similarity Knowledge ці ж випадки з Case Base можуть бути перевикористані для побудови дерев для визначення схожості.

2.2.1 База прецедентів

База прецедентів – сховище даних, де усі прецеденти мають однаковий набір атрибутів та їх значень. Не дивлячись на те, що реалізація Case Base є досить простою, вона вимагає від предметної області можливість розбити інформацію на дискретну множину властивостей, і так само обмежений набір значень під кожен атрибут. Проте так само важливо, щоб база прецедентів використовувалась чітко по призначенню, по вирішенню вузькоспеціалізованої або конкретної задачі. Якщо це вибір футболіста на позицію, то не потрібно наповнювати базу додатковими даними про його статистику чи стан здоров’я.

Для прикладу можна заповнити базу прецедентів футболістами, де атрибутами будуть виступати опис їх фізичних характеристик та здібностей. Так як умовна задача досить конкретизована, списку з 10 – 20 пунктів має вистачити: зріст, вага, швидкість, спритність, сила, витривалість, запас енергії, кілометраж, реакція, контроль м'яча, стиль гри.

Після того, як до рекомендаційної системи потрапить запит, similarity knowledge блок буде витягувати дані з бази прецедентів, та ранжуватиме їх за описом, порівнюючи їх із запитом, використовуючи знання про схожість, для виявлення прецеденту, який буде максимально подібним до бажань користувача.

2.2.2 Підхід на основі Similarity Knowledge

Similarity Knowledge – сукупність методів або алгоритмів, які використовуються для визначення ступеня схожості між запитом користувача і прецедентами, які зберігаються в Case Base. Ці знання дозволяють визначити, які випадки з бази прецедентів є найбільш схожими на поточну ситуацію, щоб надати належну рекомендацію.

В простому розумінні є досить багато метрик, за якими можливо встановити схожість між запитом користувача та прецедентами, це можуть бути: евклідова відстань, манхеттенська відстань, коефіцієнт кореляції, косинусна схожість, метрика зваженої суми та інші. Вони використовуються як базовий приклад, від якого можна відштовхнутись, щоб організувати більш персоналізований алгоритм під задачі рекомендаційної системи. Так само як алгоритми, всередині knowledge similarity може зберігатись дерево схожості, яке одразу організує усю базу прецедентів для полегшеного пошуку.

Так як один з найпопулярніших методів для Case-Based РС є метрика зваженої суми, можна розглянути його базовий вигляд та модифікації. Формула 2.8 надає адаптований вигляд цієї метрики для вирішення задачі розрахунку схожості. Функція $sim(t_i, c_i)$, що заміняє простий член x у метриці зваженої суми, є окремою формулою, що вимірює схожість атрибутів між запитом користувача та прецедентом. Для точних значень, як числа, може використовуватись математична формула, що розраховує модульну відстань, а для неточних може використовуватись дерево схожості. При побудові дерева схожості, експерти, що працюють над РС можуть декомпозувати інформацію у дерево, і результатом буде зворотна функція відстані між двома довільними вузлами або відстань від них до найближчого спільного предка.

$$Similarity(t, c) = \frac{\sum_i^n w_i * sim(t_i, c_i)}{\sum_i^n w_i}, \quad (2.8)$$

де t – запит користувача;

c – випадок з бази прецедентів;

w – вага важливості атрибуту;

$sim(t_i, c_i)$ – функція розрахунку схожості між атрибутами.

Окрім того, що функція $sim(t_i, c_i)$ вимірює схожість атрибутів, для точних значень може використовуватись симетрична та асиметрична метрики. Наприклад, якщо головний тренер шукає схожий випадок, коли в матчі по воротах суперника було завдано 15 ударів, при усіх інших рівних. При симетричній метриці матчі, в яких було завдано 10 и 20 ударів будуть мати однакову цінність. Проте для тренера, який хоче поставити атакувальний варіант гри, матч з більшою кількістю ударів буде більш пріоритетним, ніж з меншою. В такому випадку, замість того, щоб скористатись стандартною формулою по вирахуванню модульної відстані, можна скористатись формулою 2.9.

$$sim(a_t, a_c) = 1 - \frac{|a_t - a_c|}{max(a_t, a_c)}. \quad (2.9)$$

Проте метод Case-Based має одну проблему, яка полягає в тому, що перші запропоновані рекомендації можуть бути досить одноманітні і не даватимуть користувачу повної картини можливостей вибору. Але чим більше рекомендацій дається, тим вони більш різноманітні, що помітно в процесі руху від топ-перших до подальших варіантів по списку. Тобто користувач стикається з тим, що рекомендації з найбільшим рейтингом можуть бути сильно схожі на цільовий запит, але будуть пропонувати ідентичні рішення, що показано на рисунку 2.4 (а). В порівняння наведено рисунок 2.4 (б), на якому використовується суміш схожості і

різноманітності, яка дозволить користувачу ознайомитись з рекомендаціями з усіх боків.

Результат, який показано на рисунку 2.4 (б) можна досягти, якщо вибрати k випадкових рекомендацій з bk , або підійти більш радикально до збереження різноманітності, використовуючи формулу 2.10.

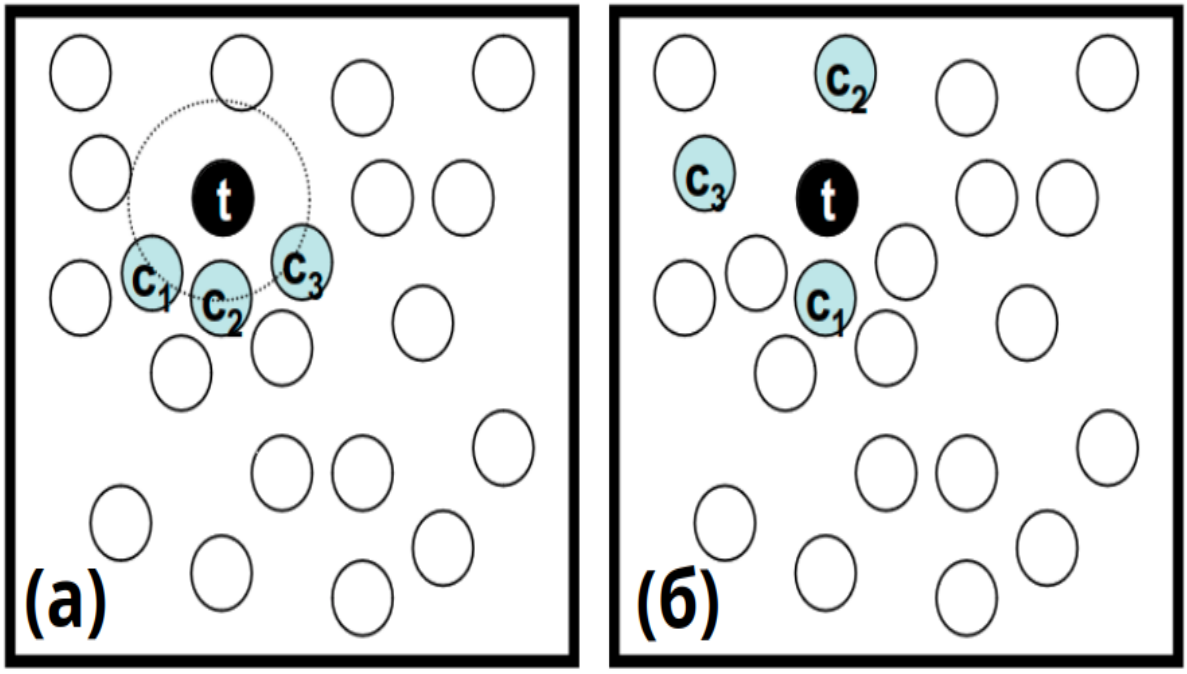

Рисунок 2.4 – Порівняння розрахунку рекомендацій

$$Diversity(c_1, \dots c_n) = \frac{\sum_i^n \sum_j^n (1 - Similarity(c_i, c_j)}{(n * (n-1)) / 2}. \qquad (2.10)$$

Наведені вище формули описують стандартний підхід у роботі Case-Based рекомендаційної системи, де взаємозв'язок між прецедентами та запитом користувача більш прямолінійний [13], [16]. Єдине, що може врегулювати порядок пріоритетності це вага до кожного атрибуту. Проте є інші техніки, які дозволяють використовувати рекомендаційні системи більш гнучко, і не відкидати нерелевантні на перший погляд результати.

2.2.3 Order Case-Based

Техніка, де враховується порядок вимог користувача для формування рекомендацій, на основі упорядкування атрибутів називається Order Case-Based. Схему роботи цього методу наведено на рисунку 2.5.

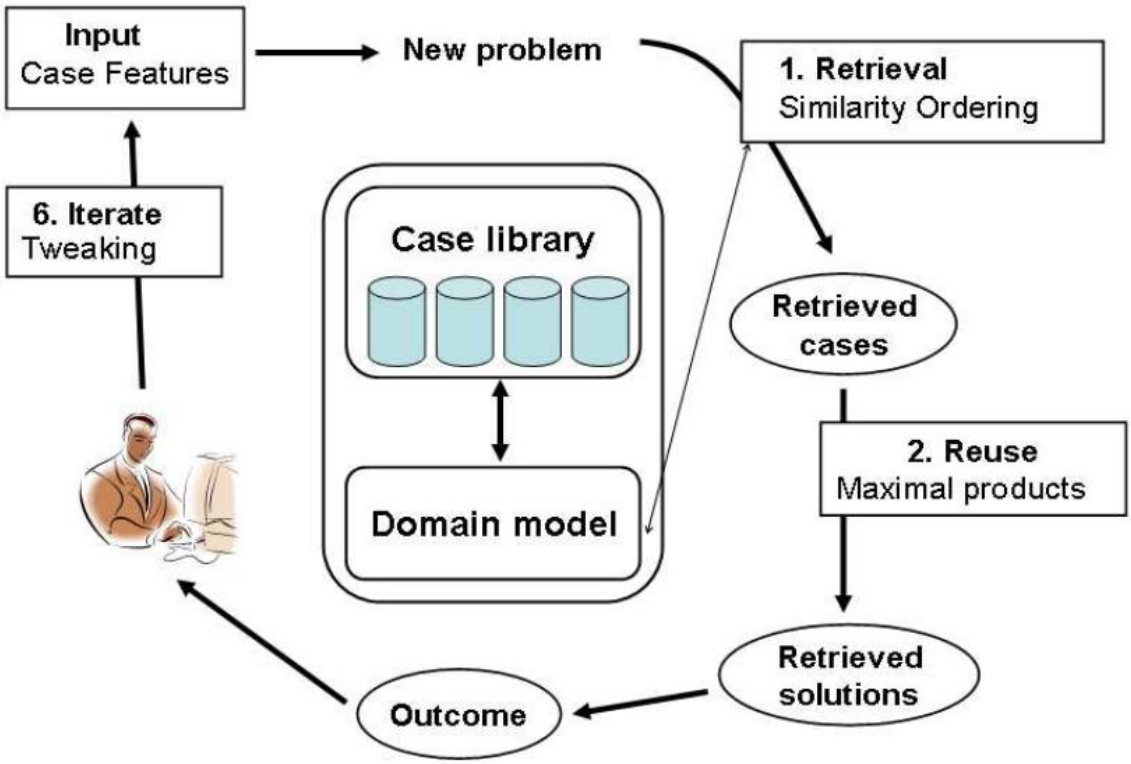

Рисунок 2.5 – Приклад роботи Order Case-Based

Як видно, життєвий цикл роботи користувача з РС принципово відрізняється. Замість одноразового запиту, та отримання рекомендації, користувач постійно взаємодіє з системою, змінюючи вимоги, доки не отримає потрібний результат. Тому найчастіше, такі рекомендаційні системи використовуються на пару з інтерактивністю.

Виходячи з назви цього методу, існує ще одна принципова різниця між ним та стандартною реалізацією Case-Based. Користувач самостійно встановлює порядок в атрибутах, за якими потрібно шукати співпадіння, і в результаті система видає відсортований перелік рекомендацій. Коли в

стандартному варіанті, незважаючи на ваги до кожного атрибуту, користувач не може контролювати порядок важливості атрибутів і ця відповідальність переходить на алгоритм, що обробляє запит [11].

Наприклад, якщо тренер вводить запит: «поставити атакувальний варіант з 20 ударами по воротах, підключенням флангів, з додатковою агресією при стандартних положеннях». В запиті є 4 складові. І виходячи з порядку атрибутів, користувачу будуть запропоновані варіанти, що спочатку будуть розглядати атакувальний варіант гри, потім 20 ударів по воротах і т.д. Проте, крім точної схожості між окремо взятими атрибутами, інші характеристики рекомендацій можуть не так явно співпадати, що дає користувачу більш гнучкий пошук з більшою кількістю варіантів. В аналогічній ситуації зі стандартним Case-Based, система намагатиметься одночасно задовольнити потреби користувача, з можливістю надання одноманітних результатів. В залежності від задачі, яку реалізує система, вибір може впасти на одну з цих систем.

2.2.4 Compromise-Driven Retrieval

Ще один підхід у реалізації Case-Based рекомендаційних систем, це Compromise-Driven Retrieval, який тримає баланс між схожістю та компромісами, які знайшов у базі випадків по запиту користувача. Технічно, цей метод тільки розширює стандартну реалізацію, яка намагається виявити рекомендації за найбільшою кількістю співпадінь між випадком та запитом користувача [12].

На рисунку 2.6 можна побачити повний цикл роботи Compromise-Driven Retrieval. На додачу до стандартного рішення, цей метод накидує пошук на основі компромісу, як обгортку до формули 2.8 та її модифікацій. Після цього відбувається групування результатів по атрибутах, та видача результатів, де система візуально має донести користувачу наявність різних

груп рекомендацій. Далі детально описані кроки, які виконуються в Compromise-Driven Retrieval.

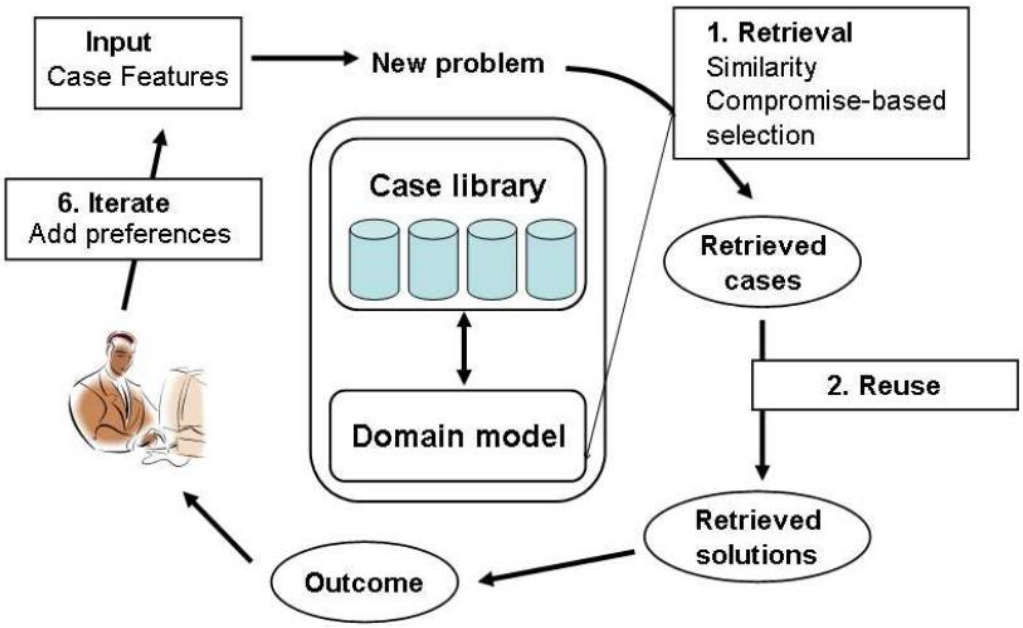

Рисунок 2.6 – Схема роботи Compromise-Driven Retrieval

Алгоритм вибірки будується на наступному: запит користувача, перелік компромісних атрибутів (якщо не вибрано жодного, можна обрати всі, або кілька найголовніших, виділених експертами), кількість можливих компромісів, випадки з Case Base та смітника (threshold). Порядок дій видно на рисунку 2.7, де представлений приклад спрощеного псевдокоду, який вирішує задачу для головного тренера по обранню схеми на матч.

Все починається з того, що користувач вводить запит, де проставляє в кожний атрибут бажане значення. Разом з запитом користувач може відправити список атрибутів, які потрібно виділити в компроміс. Також система може отримати вказівку, який допустимий рівень компромісів для одного випадку, в коді за це відповідає змінна threshold. Далі дані потрапляють до функції «findCompromises». Де до кожного значення атрибута застосовується перевірка на схожість. В даному прикладі використана проста перевірка на повне співпадіння, хоча вона легко може

бути замінена на формулу 2.8 та її модифікації. Якщо атрибут не пройшов перевірку, то цей атрибут додається до масиву компромісів. Цикл цих дій проводиться до кожного атрибуту випадку з Case Base. Далі йде перевірка масиву компромісів, яка визначає, чи не перебільшена їх кількість для кожного випадку. Якщо кількість не більше за змінну threshold, тоді цей випадок переходить на наступну стадію.

```
Function findCompromises(userRequest, recommendedOptions, threshold):
    compromises = empty list

    for each option in recommendedOptions:
        compromise = empty dictionary

        for each attribute, value in userRequest:
            if option[attribute] != value:
                add attribute and its value from option to compromise

        if length of compromise is less than or equal to threshold:
            add compromise to compromises list

    return compromises

# Example usage
userRequest = { "formation": "4-4-2", "tactics": "counter-attack", "defenders": 4, "midfielders": 4, "strikers": 2 }
recommendedOptions = [
    { "formation": "4-4-2", "tactics": "high-press", "defenders": 4, "midfielders": 4, "strikers": 2 },
    { "formation": "4-3-3", "tactics": "counter-attack", "defenders": 4, "midfielders": 3, "strikers": 3 },
    { "formation": "3-5-2", "tactics": "possession", "defenders": 3, "midfielders": 5, "strikers": 2 },
    { "formation": "4-2-3-1", "tactics": "tiki-taka", "defenders": 4, "midfielders": 2, "strikers": 3 },
    { "formation": "5-3-2", "tactics": "park-the-bus", "defenders": 5, "midfielders": 3, "strikers": 2 }
]
threshold = 2

compromises = findCompromises(userRequest, recommendedOptions, threshold)
```

Рисунок 2.7 – Псевдокод вибірки на основі Compromise-Based

Другий крок – групування. Алгоритм групує випадки з однаковими компромісами і створює еталонний набір. Користувач може отримати доступ до нього без змін. Крім того, користувач може вдосконалити свій початковий запит, вибравши один компроміс та додавши вподобання до іншого атрибуту.

2.2.5 Складність у наповненні та оновленні даних в Case-Based

Зрозуміло, що доглядом за базою знань Case Base будуть проводити аналітики, за допомоги консультацій тренерського штабу, однак наповнення та її оновлення є складним завданням, що вимагає значних ресурсів та ретельної уваги [9].

Основна проблематика полягає в наступному:

- збір та кодування даних: щоб система працювала ефективно, кожен кейс повинен бути детально описаний та структурований для цього потрібно збирати інформацію про кожну гру, включаючи всі значущі аспекти: тактику, склади команд, статистику, умови гри тощо;
- актуальність даних: футбольні стратегії та тенденції постійно змінюються, і система повинна відображати ці зміни, забезпечивши постійне оновлення бази знань, щоб включити нові кейси і актуалізувати старі, що є трудомістким процесом;
- аналіз та класифікація: щоб кейси були корисними, їх потрібно правильно класифікувати та аналізувати, що включає в себе визначення ключових факторів, які вплинули на результат гри, і забезпечення того, щоб вони коректно враховані у базі знань, де працює людський фактор;
- інтеграція нових знань: зі збільшенням кількості кейсів зростає і складність їхньої інтеграції, для чого необхідно забезпечити, щоб нові кейси не конфліктували з існуючими і доповнювали їх, покращуючи загальну ефективність системи.

Таким чином, складність наповнення та оновлення бази даних у Case-Based системах полягає у великих обсягах роботи, необхідних для збору, кодування, аналізу та постійного оновлення інформації, щоб забезпечити точність і актуальність рекомендацій.

3 ХАРАКТЕРИСТИКА ПРЕДМЕТНОЇ ОБЛАСТІ

3.1 Цикл підготовки команди до футбольного матчу

Стандартний процес підготовки команди до матчу триває від 4 до 7 днів, коли команда приймає участь в кількох змаганнях одночасно: національний чемпіонат, національний кубок, єврокубки. Все відбувається в чотири етапи:

- відновлення – ключовий етап у збереженні фізичної форми та готовності до наступних ігор. Тут є місце для легких фізичних вправ та масажу, що спрямовані на зниження м'язового напруження і поліпшення кровообігу. Водні процедури використовуються для зменшення запалення і пришвидшення відновлення. Крім того, важливими є процедури з використанням сауни або парної, що допомагають покращити кровообіг і виведення токсинів з організму;
- аналіз власних помилок – сесія тактичних занять, де аналітики клубу показують гравцям їх недопрацювання та помилки, яким буде приділена більша увага під час тренувань;
- аналіз супротивника, під час якого головний тренер разом зі своїм штабом роблять повну характеристику на противника та визначають найкращий варіант тактики гри, персоналії, слабкі місця та взаємодії гравців, які будуть намагатись атакувати ці слабкі місця;
- тренування, як напрацювання потрібних зв’язок, що будуть стимулювати суперника проявити слабину та помилитись біля власних воріт, або створити вихід один на один ще з центру поля.

Виходячи з цього, потрібно детально розглянути пункт аналізу супротивника. Як результат, тренерський штаб отримає тактику, гравців, на яких буде робитись ставка а також буде ясність, які взаємодії між футболістами потрібно тренувати під конкретного ці задачі.

3.2 Аналіз супротивника

Етап аналізу можна поділити на кілька підпунктів:

- перегляд відеоматеріалів, де визначаються основні тенденції в грі клубу в даний період;
- статистичний аналіз, який показує до яких результатів спонукає гра: контроль м’яча, кількість ударів, кількість передач та інше;
- тактичний розбір – детальний аналіз початкової схеми та гри на переходах і додатково ключові зв’язки між гравцями і роль півзахисників у роботі між лініями;
- розбір ключових гравців, які мають найбільший вплив на гру, а також не рідкісні випадки, де гравець виступає своєрідним козирем і може самостійно зробити результат;
- психологічний розбір, який дозволить мати уявлення, наскільки команда готова протистояти, навіть в умовах тотального невезіння і під тиском результату.

Далі по кожному з пунктів наведено детальний опис, який допоможе зрозуміти, як це впливає остаточний результат аналізу профілю супротивника. Загалом, якщо й можливо щось проаналізувати у грі суперника, то потрібно орієнтуватись на останні матчі. В більшості випадків впродовж останніх 8-10 матчів тактика і малюнок гри команди не змінюється, якщо немає значного перекосу у якості суперників.

3.2.1 Аналіз по сумарній ціні

Для прикладу можна взяти одного з майбутніх суперників України у кваліфікації до Євро 2024 – Боснію та Герцеговину. Перед тим як перейти до цих пунктів, в аналітиків є можливість скористатись одним зручним інструментом, Transfermarkt, який допоможе визначити клас команди. Так чи інакше, рушієм усього в світі є бізнес та фінанси, і цей сервіс дає змогу

подивитись на футбольний світ через призму грошей. Наприклад, з рисунку 3.1 видно, що гравці збірної Боснії та Герцеговини сумарно складають трохи більше 122 мільйонів євро.

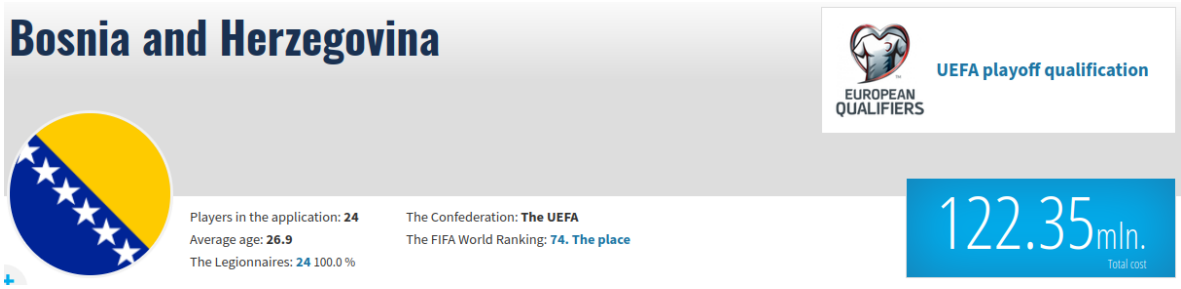

Рисунок 3.1 – Грошова оцінка збірної Боснії та Герцеговини

Якщо взяти наступну сітку розподілу команд, що в більшості випадків має емпіричні докази, то можна побачити до якої категорії команд можна віднести нашого супротивника і саму збірну України:

– до 100 мільйонів – андердоги, команди, які навряд чи можуть претендувати на те, щоб пройти кваліфікацію, здебільшого мають кілька гравців з чемпіонатів нижче середнього рівня, та більша частина перебуває в рамках національних змагань;

– від 100 до 200 мільйонів – команди нижче середнього, які так само, як андердоги здебільшого обмежені кваліфікацією, проте один-два представники стабільно потрапляють до фінального розіграшу інтернаціональних турнірів, мають кілька гравців, які грають в топ-чемпіонатах, проте цього недостатньо, аби постійно робити результат;

– від 200 до 300 мільйонів – команди середнього рівня, які можуть претендувати навіть на 1/8 чи 1/4 фіналу континентальних змагань, складають великий прошарок серед команд в Європі так і по всьому світу;

– від 300 до 500 мільйонів – команди рівня вище середнього, які можуть зібрати як мінімум один склад з гравців, які виступають в топ

чемпіонатах, мають гарну фізичну підготовку, та можуть претендувати на призові місця;

– більше ніж 500 мільйонів – талановиті команди, які ставлять собі задачу у завоюванні трофеїв, окрім Бельгії та Нідерландів, цей прошарок заповнений представниками топ чемпіонатів, або ж чемпіонатами, де активно підростають футболісти для підвищення у класі, що виступають у найкращих клубах.

Виходячи з цього переліку Боснію та Герцеговину можна віднести до категорії команд рівня нижче середнього, а Україну до категорії команд вище середнього з сумарною оцінкою гравців у 350 мільйонів євро.

Незважаючи на те, що даний приклад орієнтований на футбольні збірні, ці ж методи працюють і для клубного футболу. Окрім додаткового аналізу, що наведено на рисунку 3.2, де тренерський штаб може ознайомитись з профілями гравців на клубному рівні, наступні кроки ідентичні для обох ситуацій.

Ці знання дають розуміння того, на що здатні футболісти суперника. Порівнявши гравців обох збірних, результат видно на рисунку 3.2. Основна частина футболістів Боснії та Герцеговини представлена у чемпіонатах Німеччини, Туреччини та Росії. Також ще кілька представників у чемпіонатах середньої ланки. Лише 4 гравці грають в топ клубах найсильніших чемпіонатів. Українська збірна наполовину складається з представників свого ж чемпіонату і додатково має по 4 гравці з топ чемпіонатів Іспанії та Англії. Лише 5 гравців грають в топ клубах найсильніших чемпіонатів.

На рисунку 3.2 в обох блоках у середній колонці під кожну збірну розподілено склад в залежності від того, в якій третині таблиці знаходиться їх команда зараз. В боснійській команді 14 гравців грають в клубах, що претендують на призові місця, 8 гравців, що балансують всередині таблиці та 5 футболістів борються за виживання. Тобто розкид умовних переможців та середняків – 50 на 50. В української команди дві третини футболістів

змагаються зі своїм клубом за призові місця, 5 гравців посередині та три борються за виживання.

Перша колонка дає зрозуміти фізичні властивості футболістів. Очевидно, що в топ чемпіонатах завжди фізичні характеристики будуть на максимумі: витривалість, кілометраж за гру, звичка до інтенсивності та інше. Тобто, окрім вищої якості персональних здібностей, футболісти, що грають в першій шестірці чемпіонатів: Англія, Іспанія, Італія, Німеччина, Франція та Португалія, завжди зможуть виконувати більший обсяг роботи, порівняно з іншими. Виходячи з того, що у Боснії та Герцеговини 5 гравців знаходяться в Німеччині та ще троє у Австрії, загалом 8 людей, що звикли до інтенсивних навантажень. Інші грають в чемпіонатах помірної інтенсивності. Те саме можна сказати і про збірну України, яка має по чотири представника в АПЛ та Ла-Лізі, а всі інші здебільшого зібрані з вітчизняного чемпіонату.

Друга колонка допоможе визначити психологію гравців, до якого розвитку подій вони звикли: домінувати чи грати контратаками, постійно пресингувати і шукати моменти чи сподіватись на кілька шансів у матчі і т.д. Ситуація обох збірних в цьому плані схожа, проте Україна, завдячуючи Шахтарю та Динамо, має більший показник.

Рисунок 3.2 – Порівняння складу збірних

Третя колонка – змішування двох попередніх, де Україна стабільно випереджає суперника на 20%. І маючи всю інформацію, можна сказати, що українська збірна по кожному пункту має перевагу, хай і не дуже велику, проте, так само, якщо подивитись і на клуби, в яких представлені футболісти, можна зібрати цілий склад гравців, що представлені в топ-чемпіонатах, чого не можна сказати про Боснію.

Наступний крок, це більш детально ознайомитись з ціною, яка представлена на тому ж ресурсі TransferMarkt, наведено на рисунку 3.3.

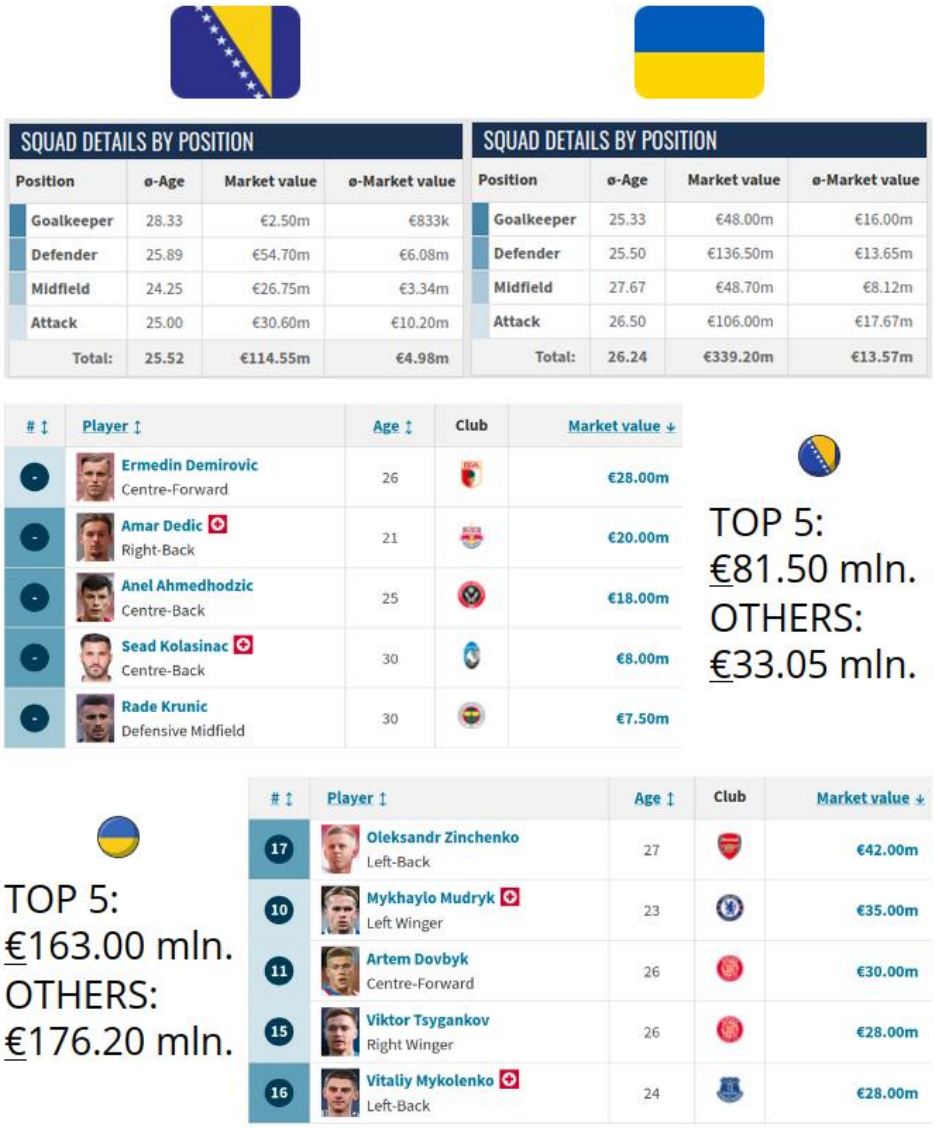

Рисунок 3.3 – Порівняння сумарних цін збірних

Так чи інакше, але в житті, як в і рекомендаційних системах, все зводиться до розуміння простих, дискретних показників. Таким показником є ціна гравців у командів. Можна вважати, що експерти ресурсу TransferMarkt можуть оцінювати футболістів у суб'єктивному ключі, проте усіх однаково, відносно цієї шкали. Виходячи з цього, є можливість визначити якою є справжня ціна команди, віднявши від неї суму найдорожчих гравців. Таким чином можна буде дізнатись справжню середню суму команди, виключивши головних талантів нації.

На даний момент збірна Боснії та Герцеговини оцінюється в 114.55 млн. євро. Україна – 339.20 млн. євро. Топ п'ять боснійської команди оцінюється в 81.50 млн. євро, що становить 71% від усієї ціни збірної. Для України частка ціни топ п'ять складає 48%. Отже, відкинувши топ п'ять, і побачивши реальний порядок цін, можна більш об'єктивно поглянути на самі збірні.

З цього стає очевидно, що Україна володіє набагато сильнішими ресурсами, і також можна побачити як оцінюється кожна лінія в командах. Захист: шість мільйонів проти тринадцяти. Півзахист: три проти восьми. Напад: десять проти сімнадцяти. І загальна середня ціна гравця в боснійській збірній близько 5 млн. євро, а в українській трохи більше 13.50 млн. євро. Очевидно, що слідуючи цим числам, збірна України вважається фаворитом і має нав'язувати свою гру, тому має придумати атакуючий план на гру, бо Боснія та Герцеговина будуть грати від суперника і намагатись шукати свої моменти.

Якщо перенести ці реалії на клубний рівень, тут також впливають показники того, в якій формі перебуває команда, та на якому місці йде. Звісно числа ніколи не дають 100% гарантії, але вони дуже часто правильно орієнтують. Якщо ж ціни суперників приблизно рівні, тоді кожна команда має підготувати як атакувальний, так і захисний плани на гру. Звісно кожен намагається грати сміливо, проте гроші завжди об'єктивно показують справжню силу команди та її шанси на успіх.

3.2.2 Аналіз попередніх ігор суперника

Здебільшого орієнтація на матчі, які відбулись за останні 2 місяці досить точно відображає приблизний орієнтир, якому буде слідувати команда і надалі. Більша частина команд має стабільний малюнок гри, кістяк гравців (від п’яти до восьми) та заготовлені сильні сторони, на які робиться упор в матчі. Звісно є команди, які не можуть похвалитись стабільністю в грі, проте це скоріш за все через те, що керівництво клубу не може досягти бажаного результату, звідки відбуваються постійні зміни і навіть відсутність кістяка, коли стабільно на кожну гру виходить менше п’яти футболістів.

Аналіз попередніх ігор проводиться аналітиками команди за допомоги різних підходів:

- статистично – існує багато сервісів, як WyScout, WhoScored, InStat, які розбивають абсолютно кожний показник на молекули та дають повну статистику дій окремого гравця, його дії з кожним з партнерів та статистику всієї команди, приклад звіту з WyScout наведено у рисунку 3.4;
- перегляд відеоматеріалів – як самостійно, так і за допомоги різних сервісів можливо переглянути тони відео, де ті ж сервіси надають розбивку по матчу на ключові моменти і навіть прописують звіти, чому вважати ці моменти ключовими;
- запит від тренерського штабу – окрім загального рев’ю, аналітики мають завдання від тренерів, що саме потрібно оцінити в майбутнього суперника, щоб розуміти як нівелювати його сильні сторони та максимально впливати на слабкі.

Виходячи з того, який буде суперник, тренерський штаб формує план знизу до гори, тобто з глобального бачення гри до підбору персоналій. А для використання рекомендаційних систем головному тренеру потрібно мати достатній функціонал того, щоб побудувати запит для визначення схеми гри, ядра для розгону атаки, вид захисту, пресингу та підбір футболістів під

ці вимоги. Тобто потрібно сформувати відповідний набір атрибутів, за якими будуть розбиті випадки для Case-Based та обмеження для Constrained-Based. Маючи на руках усі атрибути, для кожного з них, визначити домен значень і запакувати це у пристойний інтерфейс для роботи з рекомендаційною системою.

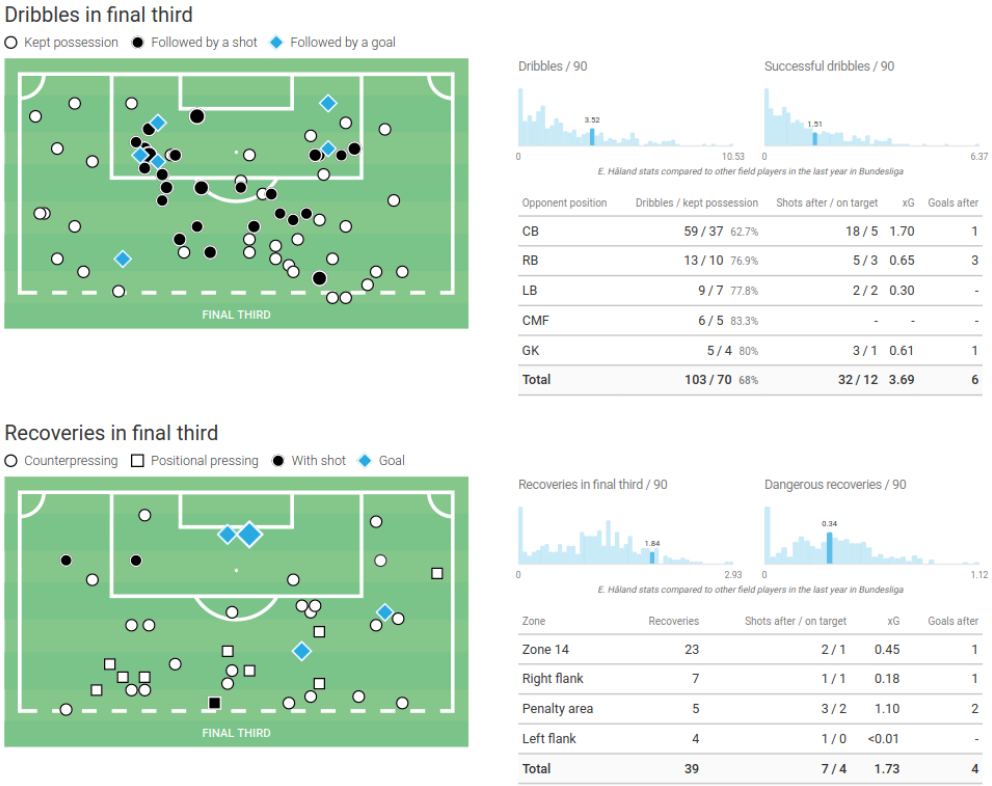

Рисунок 3.4 – Приклад звіту з WyScout

3.2.3 Постановка малюнку гри

Як було зазначено в попередньому розділі, для визначення малюнку гри найкраще підійде варіант використання Case-Based підходу. Задачі які може вирішити цей метод наступні:

– команда має грати проти сильнішого суперника, значить необхідно обрати ту тактику, де акцент буде на одному з видів активної оборони, м’яч буде у суперника і створення моментів через контратаки;

– команда має грати проти рівного суперника, але часто так стається, що легше відбитись від сильнішої команди, або задавити слабшу, а проти рівного противника дуже важко підібрати ключ і все зводиться до того, хто візьме ініціативу;

– команда має багато ігор за короткий час і усіма методами намагається знайти більш оптимальну схему гри для збереження ресурсів.

Існують інші варіанти, але найрозповсюдженіші сценарії наведені вище. Перший крок до розуміння в яких умовних вісових категоріях знаходяться команди – цінник. Якщо між командами (без урахування топ п’яти гравців) існує різниця в приблизно 30% від суми, тоді можна сміливо заявляти, що одна з команд має просто більше ресурсів і її футболісти формують сильнішу або слабшу гру. Альтернативна метрика – ціна складу з одинадцяти стартових гравців. У поєднанні буде легше знайти остаточне відношення, щоб не плутатись з підрахунком гравців в командах [17].

Після визначення розстановки сил, тренерський штаб посилає в запит до рекомендаційної системи. Якщо розглянути задачу самого запиту, то є два варіанти:

– сформувати перелік вимог на основі бачення тактики перед матчем, якщо взяти для наглядності наступний масив {стиль_гри = атакувальний, пресинг = при втраті м’яча, застосування_флангів_в_атаці = обидва порівну}, тобто цей масив характеризує, те, як тренер хоче бачити сам процес гри і від того формує задачу для команди;

– перелік вимог по результату, значить, що в базі знань СВ лежить достатня вибірка випадків, які досить чітко описані зі статистичного боку, що дає можливість подивитись на сам результат {кількість_ударів = 15, контроль м’яча = 54%, XG = 1.5}, з такими атрибутами шукаються ігри, де зустрівся приблизно такий результат, і рекомендацією видається план на гру, яка застосувала ця команда.

Очевидно, що найкращим застосуванням буде об’єднання цих двох варіантів, і формування запиту і на основі вимог до тактики, і на основі

результату. Це дасть гнучкість для пошуку в рекомендаційній системі, і бонусом буде мати розширену версію вибірки. Для цього потрібно сформувати повний перелік атрибутів, який зможе в повній мірі описати усі аспекти гри, які зможуть задовольнити вимоги тренерського штабу.

Вигоди, які отримує головний тренер, виконуючи пошук по обох варіантах наступні:

- у випадку пошуку по тактичних вимогах, результатом будуть успішні кейси, коли тактика, яку обрало керівництво клубу на початку гри привело до бажаного результату, маючи у випадках не тільки атрибути а й детальне пояснення, тренерський штаб зможе ознайомитись з деталями, які хитрощі були застосовані командою для досягнення мети;
- для варіанту пошуку по статистиці матчу, результатом будуть успішні кейси, при яких команда змогла досягти цих показників, в цих випадках будуть описані також початкові задумки тренерського штабу, і зміни, які були проведені під час матчу.

Тобто буде два види рекомендацій і тут дуже добре підійде Order Case-Retrieval метод, який буде збирати рекомендації окремо для тактичних вимог і окремо під кінцеву статистику. На виході головний тренер має отримати задовільний звіт, який підкаже: тактику гри, варіант атаки та пресингу, характеристики захисту, план дій при стандартних положеннях, темп і інтенсивність гри, креативність, гра між переходами, та інше, що порахують за необхідне експерти. Отже результатом буде чітко сформоване завдання для команди, будуть розставлені всі фішки на тактичній дошці, приклад чого можна побачити на рисунку 3.5, і залишиться виставити персоналії. Коротко по переліченому вище далі.

Тактика це схема, яка визначає розстановку гравців на полі і є певним показником, який характер гри буде сповідувати команда. Є стандартні схеми, як 4-4-2 з лінією по центру, або 4-3-3 з різною варіацією центру та постановки вінгерів, та частий італійський варіант 3-5-2.

Атака та пресинг характеризують дії команди в побудові атаки: через які фланги будувати атаки, які зв’язки використовувати для виходу гравців у вільний простір, де зустрічати суперника, коли він володіє м’ячем і взагалі який рух команди без м’яча.

Темп та інтенсивність відповідають за те, щоб по-перше самим диктувати правила гри, або ж ламати плани суперника, якщо він сильніший. По-друге, раціональне використання ресурсів, замість того, щоб носитись перший тайм, а на другий вийшла «мертва» команда.

Гра між переходами потрібна для швидкого і ефективного переходу від оборони до атаки. Вона дозволяє команді миттєво використовувати помилки суперника, створюючи небезпечні контратаки. Це важливо, щоб захопити суперника зненацька, коли його оборона ще не організована. Ефективні переходи збільшують шанси на створення гольових моментів, поки суперник не встиг перебудуватися, що підвищує ймовірність на взяття воріт.

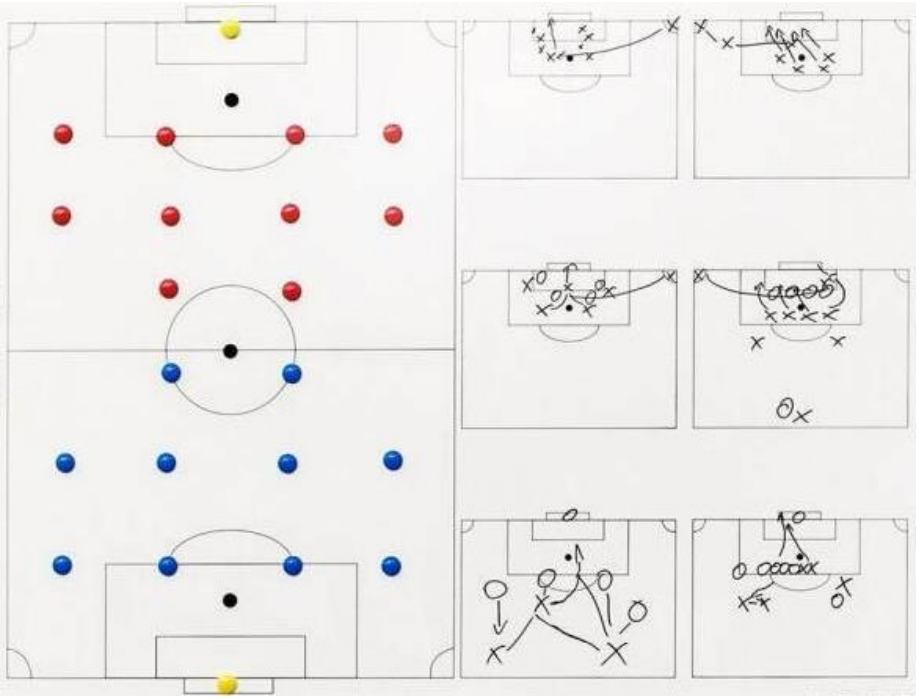

Рисунок 3.5 – Приклад тактичної установки на гру

Так як, чітко категоризувати футбольні техніки дуже важко (чому і обрано Case-Based), можна застосувати власне тлумачення футбольних термінів, яке буде працювати всередині системи. І за допомоги внутрішнього словника з’являється можливість побудувати дискретне значення доменів. Отже необхідним є продумати найголовніші атрибути, які будуть характеризувати глобальний малюнок гри, і за допомоги внутрішнього словника дати всеохоплююче визначення під ці атрибути.

3.2.4 Формування складу

Після того, як затверджено план на гру, задачі, які мають виконати футболісти на полі, потрібно сформувати склад та розставити людей. Так як глобальний малюнок сформувався на папері, то під кожного гравця легше описати більш конкретні вимоги, які від нього очікуються. Окрім персональних характеристик, можна описати його задачі на полі у взаємодії з усією командою, а не гравців на конкретних позиціях.

Так як даними по кожному гравцю легше оперувати через їх достатню точність та не велике розмаїття варіантів, з цією задачею краще впорається Constrained-Based метод, який працює з обмеженнями, та не прив’язаний до попереднього досвіду. Спираючись на формули 2.1 – 2.8, можна ще раз переконатися в тому, що цей метод рекомендацій впорається краще за Case-Based.

І в даному випадку так само наповненням бази знань мають займатись експерти, члени тренерського штабу або аналітичної команди. І результатом їх роботи будуть детальні описи кожного футболіста на мові виключень, якими зможе орудувати алгоритм рекомендаційної системи. Як і для попередньої задачі, при формуванні складу, неможливо буде вказати всі вимоги у запиті, тому потрібно виділити основний перелік обмежень, які найчастіше фігурують, як задача, яку має виконати футболіст.

Використання Constrained-Based підходу у формуванні складу команди на гру має свої виклики та переваги. Основні труднощі включають складність визначення та моделювання обмежень, необхідність регулярного оновлення точних даних про гравців, а також можливість інтеграції з додатковими системами та процесами на перспективу.

3.3 Застосування рекомендаційних систем у побудові плану на гру

Виходячи з постановки задачі, рекомендаційна система має допомогти тренеру побудувати план на гру. Процес складається з двох глобальних етапів: побудова малюнку гри, та формування складу під цю тактику. Система повинна використовувати два методи рекомендаційних систем: Case-Based для вирішення першої задачі та Constrained-Based для вирішення другої.

Проте місія рекомендаційної системи, це не заміна функції, яку виконує тренерський штаб. Це лише розширений погляд на стратегії та неординарні рішення, які можливо використати. Головне полягає в тому, щоб показати тренерському штабу альтернативу, те, що не лежить на поверхні. Додатково, це скасовує проблему авторитетності думки, коли аналітик або тренер звертається до коуча з рекомендаціями, які він може не врахувати, посилаючись на більший досвід і, як наслідок вищу посаду. Поради від рекомендаційної системи, яка не зможе собі присвоїти почесті від можливої перемоги, будуть сприйматись, як аналіз робота без особистості. Тому передбачається, що робота системи буде розділена на два етапи. Між цими етапами, користувач зможе вибрати підходящу стратегію, а також сформувати запит по формуванню складу.

Для прототипу такої рекомендаційної системи, можна використати наступний перелік характеристик для об’єктів бази знань першого етапу. Пункти цього переліку можна роздивлятись, як побудову плану на гру у вигляді піраміди, де нижній блок займає перший пункт, а вершину –

останній. Як зазначалося раніше, визначення цінової категорії допоможе зорієнтувати до якого плану на гру більше придивлятись, тому після грошової оцінки двох команди можна пройтись по наступних пунктах:

- напрямок гри: атака/баланс/захист;
- вид прессингу: високий/середній/низький/ситуативний;
- вид захисту: зональний/персональний/зонно-персональний;
- гра між переходами (фазами);
- темп та інтенсивність;
- створення моментів біля воріт суперника;
- гра при стандартах;
- захист при стандартах суперника;
- дії на перехопленні після стандартів суперника;
- головні зв’язки між гравцями;
- креативність атакуючої лінії;
- схема.

Приблизно за такими характеристиками можна розбити опис кожного випадку, який буде зберігатись у базі знань Case-Based методу. Ці характеристики та їх спеціально обмежений набір значень, що визначений внутрішнім словником системи, будуть лиш якорями, на які можуть спиратись аналітики, щоб інтегрувати матч всередину рекомендаційної системи. Ці ж характеристики будуть доступні для користувача системи для того, щоб зробити запит до РС та отримати відповідь.

Перелік характеристик вище – задумки тренерів до матчу, які вони хотіли реалізувати проти суперника, проте далеко не завжди все вдається так, як задумувалось з самого початку, тому звісно, що у прикріпленому звіті може йтись про певні тактичні зміни. Додатково, щоб забезпечити ширший погляд на ігри суперника, потрібно використати пошук по статистиці, щоб не на теорії, а на практиці роздивитись варіанти того, які успішні прийоми використали команди проти суперника. Так само, як з попереднім список, взято умовні характеристики для прототипу:

- удари по воротах;
- удари в площину воріт;
- удари зі стандартних положень;
- володіння м'ячем;
- кількість успішних пресинг-пасток;
- xg;
- точність передач;
- кількість відборів;
- кількість перехоплень;
- кількість успішного дриблінгу;
- кількість потенційних гольових передач з флангів;
- відношення атак з лівого флангу.

Існує багато джерел, звідки можна взяти статистичну інформацію. І таким чином у експертів є можливість заповнити кейси тими характеристиками, що їм потрібно для їх аналітичної системи.

Окрім цих характеристик, якими описуються матчі, до кожної рекомендації буде прикріплено детальний опис, як текстовий звіт із супутніми відео та фото матеріалами, де матч буде розібраний згідно вимогам головного тренера. І за допомоги вищенаведених списків, буде можливість залучити штучний інтелект до аналізу даних і рекомендації більш цільових звітів головному тренеру.

Маючи на руках рекомендації, тренерський штаб має можливість доналаштувати саму схему гри, і взяти цю схему на озброєння до наступного етапу. Тобто після обраної рекомендації, користувач переходить на наступну сторінку, де може змінити налаштування усіх характеристик. Система збереже цей варіант, а натомість видасть користувачу стандартний набір обмежень по кожному гравцю. Після чого буде сформовано початковий склад на матч.

Для того, щоб сформувати дефолтний склад на матч, експерти РС мають пропрацювати перехідний етап зі значень атрибутів обраного плану

до переліку обмежень, які характеризують гравців. Проте в користувача завжди залишатиметься можливість змінити значення атрибутів і отримати інші рекомендації.

Для того, щоб система могла визначити яких гравців потрібно підібрати, достатньо мати дані про схему, яка визначить кількість гравців в кожній з ліній: атака, півзахист, захист, та їх роль. Наприклад, результат роботи PC Case-Based разом з редагуванням від користувача видав наступне:

- напрямок гри: баланс;
- вид прессингу: середній;
- вид захисту: зонно-персональний;
- гра між переходами (фазами): через півзахисників, що спускаються на лінію нижче, вертикальна передача;
- темп та інтенсивність – висока;
- створення моментів біля воріт суперника: удар по воротах при першій можливості;
- гра при стандартах: пробивання;
- захист при стандартах суперника: зонно-персональний;
- дії на перехопленні після стандартів суперника: контратака всією лінією атаки;
- головні зв’язки між гравцями: трикутники;
- креативність атакуючої лінії: збалансована;
- схема – 4-2-3-1.

Зрозуміло, що неможливо зіставити усі склади на кожен вид плану гри, де при зміні значення одного аргументу будуть розставлені інші персоналії, проте важливо мати кілька орієнтирів, які впливатимуть глобально на склад і точково [18].

Глобальні атрибути визначають кого куди потрібно поставити і тут допомагають по-перше схема, по-друге напрямок гри і додатково вид

пресингу та захисту. Це допоможе зрозуміти гравцями якого типу потрібно заповнити ці безіменні фішки на тактичній дошці.

Початок завжди бере схема. Зі схеми 4-2-3-1 видно, що на полі буде чотири захисники, п’ять півзахисників та один нападник [21]. Як завжди півзахисників найбільше, тому їх теж потрібно розподілити: двоє грають з глибини, значить вони більше задіяні в захисті та побудові атаки; трійка грає вище, допомагає в атаці і бере на себе креативні дії. Також видно, що п’ять гравців більше тримаються центру – це два захисники, два півзахисники, один атакувальний півзахисник та один нападник. Двоє крайніх захисників грають на флангах і до них також підключаються двоє атакувальних півзахисників, які можуть грати і в центрі, і в напів флангах, і біля краю поля.

Наступний крок – визначення ролей, які займають футболісти. Від цих ролей також залежать і інші показники атрибутів, які описують поведінку під час гри. Далі наведено перелік ролей, які можуть займати гравці по позиціях.

Центральні захисники:

− традиційний центральний захисник – захист, вигравання повітряних дуелей, відбір м'яча, виконує основну роль по тому, щоб жорстко зустріти суперника, або тактично сфолити при більшій відстані від штрафного майданчика, часто високі захисники приймають участь в розіграші кутових, щоб допомогти команді забити;

− ліберо – грає зазвичай позаду основної лінії захисту і виконує функції останнього захисника, страхуючи інших захисників, проте класичне розуміння цього слова давно змінено і зараз це останній захисник до воротаря;

− ball-playing defender (захисник, що грає в м'яч) – відзначається добре розвинутими технічними навичками та здатністю розпочинати атаки через точні передачі, може використовуватись, як гравець, що грає між переходами (фазами).

Крайні захисники:

− фулбек – класичний крайній захисник, який зосереджений на обороні, але також підтримує атаки, і бере на себе від 60% до 75% бровки;

− вінгбек – грає ширше і вище, часто у схемі з трьома центральними захисниками, більше залучений в атаки, і може одночасно виконувати роль крайнього захисника, коли схема захисної лінії перебудовується з трьох до п’яти гравців, а також крайнього півзахисника, інколи другого вінгера на фланзі;

− інвертований фулбек – захисник, який рухається в центр поля під час атак, створюючи додаткову опцію в центрі, де скоріше використовується, як додатковий опорний та реагує на переміщення суперників, щоб вчасно закрити коридори.

Півзахисники, здебільшого представлені трьома або чотирма гравцями на полі, де одна група більше орієнтована до атаки, а друга до захисту:

− центральний півзахисник – гравець, що виконує як оборонні, так і атакуючі функції, охоплює до 80% поля, проте рідко заходить до власної або чужої штрафної, основна роль бути ядром команди, що скеровує розвиток атаки та в залежності від пресингу команди, допомагає повернути м’яч і одразу зробити передачу в атаку;

− опорний півзахисник – гравець, що зосереджується на захисті перед лінією захисників, часто відбирає м'яч та намагається якомога швидше зустріти суперника, щоб ядро півзахисників не відбігало далеко в захист, від чого і назва – опора для півзахисту, також стає додатковим захисником при активній обороні;

− плеймейкер – гравець, що розподіляє м'яч і створює моменти для атакуючої лінії, і сам має задачу зміщатись ближче до штрафного суперника, приймаючи участь в продовженні атаки;

− бокс-ту-бокс – гравець, що поєднує в собі всі вище перелічені властивості півзахисників, вільно переміщується по полю, проте й повинен

мати добре розвинені фізичні якості, які дозволяють виконувати великий обсяг роботи.

Атакуючі півзахисники:

− розігруючий півзахисник – зосереджується на створенні атак та віддачі гольових передач, здебільшого виступає підтримкою для нападаючого, і разом з іншими гравцями намагається звільнити простір для гольового моменту;

− вінгер – швидкий тип гравця, який грає на флангах, головна задача якого створити момент для себе та для команди, знаходячи можливість обіграти захисників, та вибити м’яч на вільний простір;

− підтримуючий нападник (support striker) – гравець, що грає позаду головного нападника, створює та завершує атаки, загалом виконує роль нападника, проте діє з глибини та має навички в розіграші м’яча поряд з іншими півзахисниками.

Нападники:

− центральний нападник – основний бомбардир команди, що зосереджується на забиванні голів;

− фінішер – нападник, що використовує свій інстинкт і швидкість для завершення атак, часто забиває з близької відстані;

− інсайд форвард – гравець, що діє на флангах, але часто зміщується в центр для забивання голів.

Виходячи зі схеми, наведених ролей, виду пресингу та захисту, можна сформувати перелік стандартних обмежень, які будуть надаватись самою системою для вирішення задачі в Constrained-Based методі.

Далі в користувача буде можливість редагувати кожний з параметрів до обмежень по гравцях, і сформувавши бажаний запит відправити до системи. Система пройдеться по кожній з позицій і вирішить для неї найкращого кандидата. Перелік обмежень-атрибутів, які характеризують гравця в базі знань для Constrained-Based методу наведено далі. Звісно це умовний набір атрибутів для демонстрації роботи [20].

Центральні захисники:

- відбори м'яча – середня кількість впродовж сезону;
- гра головою – гра в чужій штрафній, скільки ударів було нанесено по воротах впродовж сезону;
- перехоплення – середня кількість впродовж сезону;
- вибивання м'яча – вибивання з власної штрафної, середня кількість впродовж сезону;
- блокування ударів – середня кількість впродовж сезону;
- швидкість – середня швидкість в кілометрах на годину при ривках в активному захисті.

Крайні захисники:

- відбори м'яча – середня кількість впродовж сезону;
- перехоплення – середня кількість впродовж сезону;
- швидкість – середня швидкість в кілометрах на годину при ривках в активному захисті;
- фізична витривалість – середня кількість зіграних хвилин впродовж сезону;
- гра в нападі – середня кількість успішних підключень до атаки у вигляді предасистів, асистів або спроб закинути м’яч на замикання;
- точність передач – середній відсоток впродовж сезону;
- дриблінг – середня кількість успішних спроб впродовж сезону;
- повернення в оборону – середня кількість вчасних повернень до власної штрафної під час активної фази захисту.

Півзахисники:

- дриблінг – середня кількість успішних спроб впродовж сезону;
- точність передач – середній відсоток впродовж сезону;
- відбори м'яча – середня кількість впродовж сезону;
- перехоплення – середня кількість впродовж сезону;
- успішний build-up – середня кількість успішних дій, з яких була створена атака;

– фізична витривалість – середня кількість зіграних хвилин впродовж сезону;

– швидкість прийняття рішень – середня кількість часу, для того, щоб гравець вийшов з під пресингу, вимірюється в секундах;

– удар по воротах – середня кількість впродовж сезону;

– креативність – середня кількість індивідуальних дій для просування м’яча на третину поля суперника.

Атакуючі півзахисники:

– створення моментів – середня кількість успішних дій, які призвели до удару по воротах;

– точність передач – середній відсоток впродовж сезону;

– дриблінг – середня кількість успішних спроб впродовж сезону;

– удар по воротах – середня кількість впродовж сезону;

– виконання стандартних положень – середня кількість ударів в площину впродовж сезону;

– довгі передачі – середня кількість успішних довгих передач впродовж сезону;

– пенальті – кількість реалізованих пенальті;

– втрата м’яча – середня кількість втрат м’яча в останній третині поля впродовж сезону.

Нападники:

– пенальті – кількість реалізованих пенальті;

– забитих голів – кількість голів, не враховуючи пенальті;

– реалізація моментів – відсоток ударів по воротах до голів;

– xg – очікувана кількість голів впродовж сезону;

– інтуїція нападника – кількість голів з XG менше 0.15;

– креативність – середня кількість успішних дій, що призвели до голу або атаки з XG більше 0.25;

– швидкість – середня швидкість в кілометрах на годину при ривках в активному захисті;

- виграні верхові дуелі – середня кількість впродовж сезону;
- гра при стандартах – середня кількість успішних дій при стандартних положеннях впродовж сезону;
- удари здалеку – середня кількість ударів здалеку в площину воріт впродовж сезону.

Як видно з переліченого вище, в багатьох позиціях є схожі атрибути, тому доречно реалізувати набір гравців в базі знань в рамках одної таблиці. Також спільними атрибутами до кожного гравця будуть:

- вага – вимірюється в кілограмах;
- зріст – вимірюється в сантиметрах;
- позиція на полі;
- роль на позиції;
- суміжні позиції на полі;
- суміжні ролі на позиції.

Після того, як буде запропонований склад, користувач матиме на руках детальну рекомендацію по малюнку гри та персоналіях. Він може використати це для роздумів про гру [19]. Очікується, що рекомендації дадуть йому розгорнуту відповідь, яку він зможе опонувати та представити власний погляд на гру. Але рекомендація буде мати аргументацію свого вибору, на яку і повинен буде звернути увагу користувач, щоб мати впевненість в своєму виборі, який відхиляється від запропонованого рекомендаційною системою.

4 РОЗРОБКА ПРОТОТИПУ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

4.1 Характеристика використаного середовища

Виходячи з контексту задачі, де рекомендаційна система має допомогти тренерському штабу визначитись з тактикою та складом на матч, не потрібно працювати з Big Data. Так чи інакше набір гравців обмежений до 30 в кожній команді, а набір матчів, які можна проаналізувати під час підготовки до суперника можна звести до 10 останніх, де зберігаються актуальні тенденції у використанні задумки тренерського штабу та є помилки, якими може скористатись команда. До того ж рекомендаційна система не носить масовий характер, а використовується обмеженою групою осіб.

Якщо умова обробки Big Data не дуже актуальна, тоді відкривається широкий спектр технологій, на яких можна написати рекомендаційну систему. Для зручності використано мову програмування PHP на базі фреймворку Laravel. Написання Case-Based та Constrained-Based буде відбуватись за допомоги використання бібліотеки PHP-ML, з якої буде використано алгоритм схожості, а інша реалізація буде написана вручну. Зберігання даних буде відбуватись всередині бази даних MySql. Розробка буде відбуватись в IDE PhpStorm.

На виході буде отримана готова рекомендаційна система API, до якої буде можливо звернутись з будь-якого пристрою: desktop, mobile, web та інші. Даний підхід забезпечить кросплатформеність рекомендаційної системи, а також можливість її змінити чи розширити для схожих застосунків. Тестування буде проходити локально на персональному комп’ютері за допомоги Postman.

Laravel – фреймворк, який вміщує в собі уніфіковану архітектуру PHP застосунку і, паралельно з цим, має обширний набір інструментів. Все це зроблено для того, щоб полегшити життя розробникам, звільнити їх від

необхідності кожного разу піклуватись про архітектуру застосунку, робити шаблонні дії, або ж думати над базовими речами у безпеці застосунку. Використовуючи дану технологію, є можливість одразу писати бізнес-логіку і швидко тестувати результати.

PHP-ML – бібліотека для машинного навчання на мові PHP. Вона надає інструменти для виконання різних завдань машинного навчання, таких як класифікація, регресія, кластеризація, обробка даних та інші.

4.2 Приклад формування датасетів

Для рекомендаційної системи по формуванню плану на гру, що використовує Case-Based метод, потрібно використати датасет, що має останні 10 матчів майбутнього суперника. Цей датасет має бути поділений на колонки, які представлені, як атрибути звіту по матчу в розділі 3.3. Так як, задача не типова, знайти подібний датасет серед запропонованих, стандартних немає можливості. Доступу до професійної частини таких платформ як WyScout чи InStat, немає, тому потрібно сформувати подібний датасет самостійно.

Для демонстрації прототипу можна взяти останні 10 ігор Ліверпуля в англійській прем’єр лізі, як зображено на рисунку 4.1, і сформувати рекомендації для Вулверхемптона, який зіграє з ними в останньому турі чемпіонату. Потрібно витягнути наступні дані по матчах: задумки тренера в одну групу атрибутів та статистику матчів в іншу групу.

Перевагою і недоліком підходу, за яким будується рекомендаційна система, це відсутня необхідність у великому наборі даних. Проте користувачам часто доведеться оновлювати дані в базі знань та взаємодіяти з нею, щоб отримати актуальний результат. Приклад готового датасету можна побачити на рисунках 4.2 та 4.3, де усі колонки заповнені даними, згідно того переліку атрибутів, що наведено в розділі 3.3. Дані легко зберігати одразу всередині таблиць MySql.

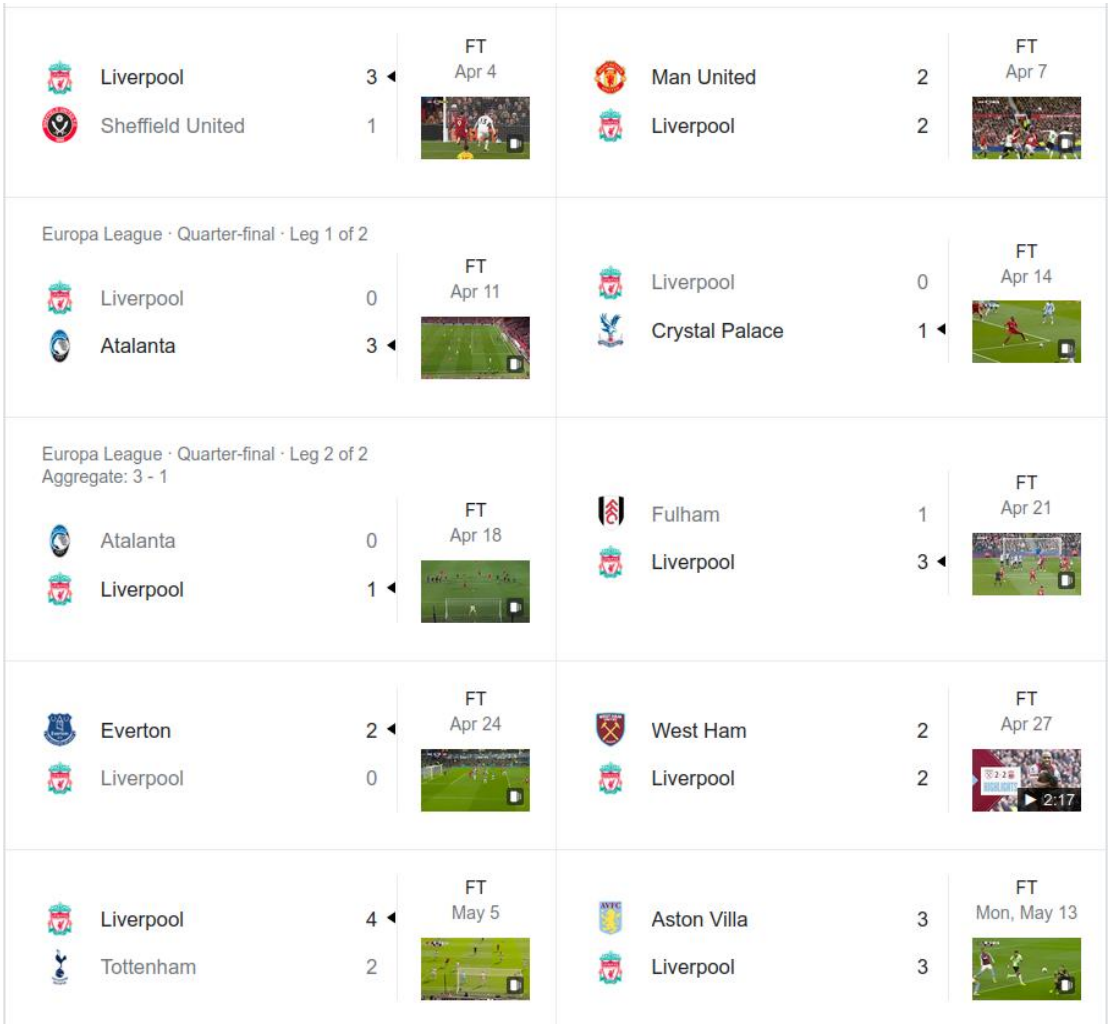

Рисунок 4.1 – Останні 10 ігор Ліверпуля

game...	pre...	defen...	transition_play	tempo...	chance_creation	set_piece_play
defense	medium	zone-personal	through mid-players	low	counter-attack	shooting on goal
balance	medium	zone-personal	through mid-players	high	shooting on goal at t…	shooting on goal
balance	low	zone-personal	long ball	high	counter-attack	tricky ball play
balance	medium	zone-personal	through mid-players	medium	shooting on goal at t…	shooting on goal
defense	low	zone-personal	long ball	high	counter-attack	shooting on goal
balance	medium	zone-personal	long ball	high	counter-attack	tricky ball play
attack	high	zone-personal	through mid-players	high	ball inside box	tricky ball play
balance	low	personal	through flanks	low	counter-attack	shooting on goal

Рисунок 4.2 – Група колонок по плану на гру

shots_on_target	set_p...	possession_p...	successful_pre...	xG	pass_accuracy
3	0	17.00	0	0.50	48.00
5	2	38.00	15	1.90	76.00
5	2	30.00	9	1.40	68.00
5	2	39.00	10	1.10	77.00
6	2	23.00	15	1.20	62.00
8	3	28.00	13	1.60	68.00
5	0	56.00	10	0.30	56.00
4	2	40.00	9	2.00	77.00

Рисунок 4.3 – Група колонок по статистиці матчу

Вище наведені частини однієї таблиці, що відображенням бази знань Case Base. Надалі можна використовувати ці дані для алгоритму рекомендаційної системи.

Наступний етап – виставити склад на гру, використовуючи Constrained-Based підхід. Так само, як для бази знань по звітах матчів, треба сформувати датасет з кожного гравця в команді, де він буде описаний за допомоги обмежень. Перелік обмежень по гравцях, який було наведено в розділі 3.3, можна побачити на рисунку 4.4, де в кожну колонку прописано значення. В перспективі є можливість розділити цю таблицю на кілька, щоб не перевантажувати її колонками, яких зараз в таблиці 39. Тому сама архітектура додатку дає можливість легко форматувати підходи, а також для кожної позиції немає потреби використовувати весь перелік.

tackling...	aerial...	interce...	cle...	blocked...	speed
4	6	5	0	0	29
5	4	6	0	1	31
2	6	4	2	4	28
3	2	5	5	2	34
2	1	3	4	1	33
5	3	2	0	1	31

Рисунок 4.4 – Група колонок по гравцях

4.3 Реалізація методів рекомендації

Перший етап – формування плану на гру, шляхом вибору найкращого варіанту з 10 останніх ігор суперника. В базі знань зберігаються дані про задумки тренерського штабу та статистика в цих матчах. База знань описується однією таблицею, що змістовно розбита на дві групи.

Для даної задачі використовується підхід Order Case Base Retrieval, який дозволяє проводити пошук рекомендацій спочатку за одним параметром, потім за іншим. Проте при формуванні плану на гру, одночасно може бути кілька атрибутів дуже важливими, тому доцільно зробити об’єднання цих атрибутів по змістовному принципу, якраз в ті дві групи, і робити пошук спочатку за одним переліком, а потім за іншим. Таким чином у користувача буде широка вибірка даних, щоб потім самостійно обрати з яким варіантом рухатись далі та налаштувати власні значення для кожного атрибуту.

Проте, навіть маючи Order CBR, не можна розраховувати на повноцінне розуміння системи. Як розглядалось раніше, не може бути, щоб один атрибут розглядався нарівні з іншим. Тому варто ввести систему коефіцієнтів вагів, які будуть пріоритезувати пошук в рамках однієї групи. Ці ваги можуть бути гнучко налаштовані користувачем під кожну задачу, щоб гарантувати актуальність результатів та більш гнучко підходити до вирішення наступного завдання.

Так само доречно згадати про метрику, яка буде вимірювати схожість між запитом користувача та кейсами в базі знань. Виходячи з того, що є атрибути з обмеженим масивом значень, integer та float, варто звернути увагу на кожен з типів полів, щоб не обмежувати методи порівняння.

Результатом виступає файл, що наведено на рисунку 4.5, де можна побачити усю структуру класу, який описує Order Case-Based Retrieval. В класі видно поділ атрибутів на тактичні та статистичні. Публічний метод search відповідає за весь процес пошуку рекомендацій на основі запиту

користувача, що передається в аргументі $request. Вміст файлу search можна побачити на рисунку 4.6. Спочатку в масив коефіцієнтів вагів збираються дані із запиту користувача, потім пошук за кожною з груп, та вивід даних до користувача. Внутрішній процес пошуку можна побачити на рисунку 4.7, де атрибут з кожної групи перебирається разом зі своїм коефіцієнтом ваги і надсилається до підрахунку схожості. На рисунку 4.8 зображено процес порівняння значень атрибутів між запитом користувача та випадками з бази знань. Результатом буде вивід рекомендацій, розподілені на дві групи: тактична та статистична.

```php
class RecommendationService
{
    protected $caseRepository;

    private $tacticalAttributes = [
        'direction', 'pressing', 'defense', 'transitions', 'tempo',
        'chances_creation', 'set_pieces', 'defense_set_pieces',
        'interceptions_after_set_pieces', 'key_player_links',
        'attack_creativity', 'formation'
    ];

    private $statisticalAttributes = [
        'shots', 'shots_on_target', 'set_piece_shots', 'possession',
        'successful_pressing_traps', 'xg', 'pass_accuracy', 'tackles',
        'interceptions', 'successful_dribbles', 'potential_assists_from_flanks',
        'left_flank_attacks_ratio'
    ];

    public function __construct(CaseRepository $caseRepository)
    {
        $this->caseRepository = $caseRepository;
    }

    public function search(Request $request){...}

    private function rankResults($results, $attributes, $request, $weights){...}

    private function calculateSimilarity($caseValue, $requestValue, $weight){...}
}
```

Рисунок 4.5 – Клас Order CBR

```php
public function search(Request $request): array
{
    $weights = [];
    $merged = array_merge($this->tacticalAttributes, $this->statisticalAttributes);
    foreach ($merged as $attribute) {
        $weights[$attribute] = $request->input( key: "weight_{$attribute}",  default: 1);
    }

    $cases = $this->caseRepository->getAllCases();
    $rankedTacticalResults = $this
        ->rankResults($cases, $this->tacticalAttributes, $request, $weights);
    $rankedStatisticalResults = $this
        ->rankResults($cases, $this->statisticalAttributes, $request, $weights);

    return [
        'tactical_results' => $rankedTacticalResults,
        'statistical_results' => $rankedStatisticalResults
    ];
}
```

Рисунок 4.6 – Процес визначення рекомендацій

```php
private function rankResults($results, $attributes, $request, $weights)
{
    foreach ($results as $case) {
        $case->score = 0;
        foreach ($attributes as $attribute) {
            if ($request->has($attribute)) {
                $case->score += $this
                    ->calculateSimilarity(
                        $case->$attribute,
                        $request->input($attribute),
                        $weights[$attribute]);
            }
        }
    }

    return $results->sortByDesc('score')->values();
}
```

Рисунок 4.7 – Процес виконання пошуку

```php
private function calculateSimilarity($caseValue, $requestValue, $weight)
{
    if (is_numeric($caseValue) && is_numeric($requestValue)) {
        $difference = abs( num: $caseValue - $requestValue);
        $maxDifference = max($caseValue, $requestValue);
        return $weight * (1 - $difference / $maxDifference);
    }

    if ($caseValue === $requestValue) {
        return $weight;
    } else {
        return $weight * 0.5;
    }
}
```

Рисунок 4.8 – Виконання процесу порівняння

Наступний етап – формування складу на гру, реалізується за допомоги Constrained-Based методу. Після того, як користувач відредагує потрібний варіант обраного плану на гру, він переходить до розстановки персоналій. Приклад бази знань з наступних 30 правил:

– «якщо напрямок гри атакувальний, то гравці в атаці мають найвищий Удар по воротах»;

– «якщо напрямок гри захисний, то всі гравці мають найвищу Фізичну силу»;

– «якщо гра побудована на високому пресингу, то гравці повинні мати найвищу Швидкість»;

– «якщо гра побудована на середньому пресингу, то гравці повинні мати найвищу Витривалість»;

– «якщо гра побудована на низькому пресингу, то гравці повинні мати найвищу Комунікацію»;

– «якщо гра побудована на ситуативному пресингу, то гравці повинні мати найвищу Фізичну силу»;

– «якщо захист зональний, то захисники мають найвищий Відбір м'яча без порушення правил»;

– «якщо захист персональний, то захисники мають найвищу Фізичну силу»;

– «якщо захист зонно-персональний, то захисники мають найвищу Фізичну витривалість»;

– «якщо гра будується на швидкому переході від захисту до атаки, то гравці повинні мати найвищу Швидкість»;

– «якщо гра будується на контрольованому переході від захисту до атаки, то гравці повинні мати найвищу Точність передач»;

– «якщо темп гри високий, то гравці повинні мати найвищу Фізичну витривалість»;

– «якщо темп гри низький, то гравці повинні мати найвищу Креативність»;

– «якщо інтенсивність гри висока, то гравці повинні мати найвищу Фізичну витривалість»;

– «якщо команда часто створює моменти біля воріт суперника, то нападники повинні мати найвищу Креативність»;

– «якщо команда добре грає при стандартах, то гравці повинні мати найвищу Гру при стандартах»;

– «якщо команда добре захищається при стандартах суперника, то захисники повинні мати найвищу Гру головою»;

– «якщо команда швидко повертається в оборону після стандартів суперника, то гравці повинні мати найвищу Швидкість»;

– «якщо головні зв'язки між гравцями на полі важливі, то гравці повинні мати найвищу Комунікацію»;

– «якщо креативність атакуючої лінії важлива, то нападники повинні мати найвищу Креативність»;

– «якщо команда часто використовує фланги в атаці, то гравці на флангах повинні мати найвищу Точність передач»;

- «якщо команда багато використовує перехоплення, то гравці повинні мати найвищу Відбір м'яча»;
- «якщо команда часто використовує дриблінг, то гравці повинні мати найвищу Дриблінг»;
- «якщо команда грає на контратаках, то гравці повинні мати найвищу Швидкість»;
- «якщо команда грає на тривале володіння м'ячем, то гравці повинні мати найвищу Точність передач»;
- «якщо команда часто пробиває з дальніх дистанцій, то гравці повинні мати найвищу Удар по воротах»;
- «якщо команда часто робить кроси в штрафний майданчик, то гравці повинні мати найвищу Точність передач»;
- «якщо команда грає через центр поля, то гравці повинні мати найвищу Креативність»;
- «якщо команда часто використовує тактичні фоли, то гравці повинні мати найвищий Відбір м'яча без порушення правил»;
- «якщо команда використовує позиційний захист, то гравці повинні мати найвищу Комунікацію»;
- «якщо команда грає в збалансований футбол, то всі гравці повинні мати найвищу Комунікацію»;
- «якщо команда грає з високим пресингом, то гравці повинні мати найвищу Фізичну витривалість»;
- «якщо команда використовує зональний захист, то гравці повинні мати найвищий Відбір м'яча».

Алгоритм пошуку показано на рисунку 4.9, де також застосована система коефіцієнтів вагів, та метрика для визначення схожості між запитом користувача та об’єктами в базі прецедентів. В результаті після 11 прогонів алгоритму, користувач отримає рекомендований склад.

```php
public function recommend()
{
    $players = Player::all();

    $rankedPlayers = $players->map(function ($player) {
        $score = 0;
        foreach ($this->constraints as $attribute => $constraint) {
            $weight = $this->weights[$attribute] ?? 1;

            if (is_numeric($constraint)) {
                // Метрика для числових значень (напр. швидкість)
                $difference = abs( num: $player->$attribute - $constraint);
                $normalizedDifference = 1 - min( values: $difference / max( values: 1, $constraint),  values: 1);
                $score += $normalizedDifference * $weight;
            } else {
                // Метрика для категоріальних значень (напр. зональний захист)
                $score += ($player->$attribute == $constraint ? 1 : 0) * $weight;
            }
        }
        return ['player' => $player, 'score' => $score];
    });

    // Сортування гравців за сумарним рейтингом
    $rankedPlayers = $rankedPlayers->sortByDesc('score')->values();

    return $rankedPlayers;
}
```

Рисунок 4.9 – Процес роботи алгоритму Constraint-Based

4.4 Приклад роботи протипу

Для прикладу можна взяти запит до рекомендаційної системи, який наведено на рисунку 4.10. Для того, щоб система видала результат, не потрібно робити запит по всіх атрибутах одразу, а взяти кілька в кожній групі та виставити пріоритет у коефіцієнтах вагів. Цей запит для Case-Based методу, щоб на основі попередніх ігор суперника знайти схожий варіант, який хоче бачити користувач.

На рисунку 4.11 наведено приклад відповіді, окремо в двох масивах знаходяться результати за різними запитами. В одному з цих масивів повністю розкритий об’єкт моделі CaseBaseReportGame, в атрибутах якого можна спостерігати певні співпадіння із запитом по статистичній групі атрибутів.

```
"tactical_attributes" : {
    "game_direction": {
        "weight": 1,
        "value": "balance"
    },
    "pressing_type": {
        "weight": 3,
        "value": "high"
    },
    "chance_creation": {
        "weight": 2,
        "value": "ball inside box"
    } ,
    "formation": {
        "weight": 2,
        "value": "4-2-3-1"
    }
},
"statistical_attributes" : {
    "shots_on_target": {
        "weight": 2,
        "value": "10"
    },
    "possession_percentage": {
        "weight": 1,
        "value": "50"
    },
    "xg": {
        "weight": 3,
        "value": "1.2"
    }
}
```


Рисунок 4.10 – Приклад запиту до рекомендаційної системи

Рисунок 4.11 – Приклад рекомендації з планом на гру

Наступний етап – формування складу для підібраного плану на матч. Навідміну від попередніх результатів, система буде повертати самих гравців з посиланням на базу знань. Частину запиту користувача наведено на рисунку 4.12. Цю задачу вирішує Constraint-Based метод. Він запускається 11 разів, щоб пройтись по обмеженням для кожного гравця і в кінці видає результат, який зображено на рисунку 4.13. Всередині кожної моделі гравця зберігається особиста інформація та посилання на таблицю, де описані його характеристики, як показано на рисунку 4.14.

Повний програмний код наведено в додатку А.


```
{
    "0" : {
        "position": "DR",
        "speed": "30",
        "attacking_support": "5",
        "stamina": "90",
        "crossing_accuracy": "85"
    },
    "1" : {
        "position": "DC",
        "speed": "28",
        "blocked_shots": "5",
        "stamina": "90",
        "crossing_accuracy": "80"
    },
    "2" : {
        "position": "DC",
        "speed": "30",
        "tackle_without_fouling": "80",
        "stamina": "90",
        "recovery": "10"
    },
    "3" : {
        "position": "DL",
```


Рисунок 4.12 – Приклад запиту

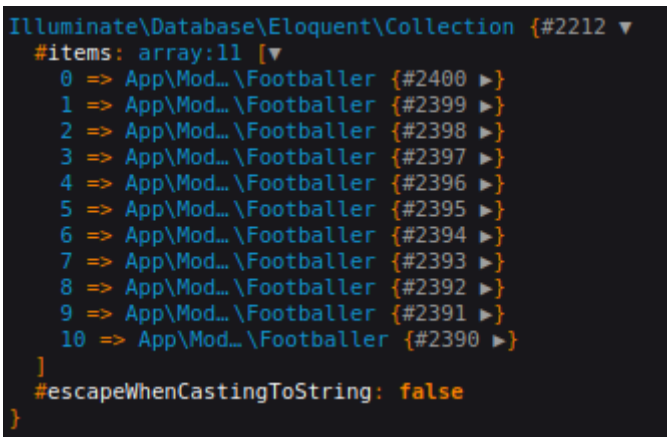

Рисунок 4.13 – Приклад відповіді з рекомендаціями

Рисунок 4.14 – Приклад рекомендації гравця з характеристиками

ВИСНОВКИ

У кваліфікаційній роботі досліджено застосування знання орієнтованих рекомендаційних систем для підбору тактичних планів та складу команди у футболі. Розглянуто загальні принципи розробки рекомендаційних систем та специфіку побудови систем, заснованих на знаннях. Футбол, з його численними змінними та взаємозалежностями, є ідеальною сферою для впровадження таких систем.

Виходячи з того, що аналоги цієї системи можуть бути під NDA в кожному футбольному клубі, потрібно було самостійно проаналізувати предметну область та застосувати знання-орієнтовані підходи для створення рекомендаційної системи, яка допоможе тренерському штабу пройти повний шлях від підбору тактичного плану до формування складу на гру. Для цього досліджено Case-Based та Constraint-Based підходи і, зважаючи на їх переваги та недоліки, підібрано найкращий варіант їх застосування в прототипі рекомендаційної системи.

Проте сам прототип є лише прикладом того, як можна використати рекомендаційну систему. Її можливо розширити та гнучко використовувати аналіз по різних атрибутах. Головна задача, яку вирішує ця система – дати тренерському штабу альтернативний погляд на успіху інших команд, які досягли цього проти майбутнього суперника.

Перший етап роботи полягав у розробці плану гри за допомогою методу Case-Based. Створено базу даних з останніми 10 матчами суперника, де зберігаються дані про клуби, з якими грав опонент, розподілені на дві категорії: стратегічні задуми тренерів і статистика матчів. Це дозволило проводити пошук рекомендацій, порівнюючи поточну ситуацію з аналогічними випадками з минулого.

Другий етап передбачав формування складу команди з використанням методу Constraint-Based. Розроблено систему обмежень, що включає атрибути гравців, такі як відбір м'яча, гра головою, перехоплення,

швидкість, фізична сила тощо. Створено базу знань з 30 правил, що дозволяють тренеру обирати гравців на основі тактичних вимог.

Для покращення алгоритму рекомендацій була впроваджена можливість застосування коефіцієнтів ваги, що дозволяє тренеру задавати пріоритети для різних атрибутів. Це зробило систему більш гнучкою та адаптивною до різних тактичних сценаріїв. Крім того, для кожного атрибута була розроблена метрика схожості: для числових значень використовується нормалізована різниця, а для категоріальних – точне співпадіння або близькість значень.

Рекомендаційну систему написано на PHP на базі фреймворку Laravel із використанням IDE PhpStorm, Postman.

Перевагами рекомендаційної системи є адаптивність, прозорість і модульність. Система досить гнучка і через реалізацію підходів пошуку, є можливість клієнтського налаштування кожного атрибуту, через що до кожного матчу можуть бути індивідуальні налаштування.

Недоліком рекомендаційної системи виступає складність підтримки актуальних даних, залежність від людського фактору і відсутність власної інтерпретації спірних моментів.

Перспективи розвитку безмежні:

- автоматизація збору даних через підключення API сервісів, що надають детальну статистику;
- використання різних метрик і методів пошуку для урізноманітнення рекомендацій;
- розширення повноважень рекомендаційної системи, щоб вона розраховувала наперед більш довгострокові задачі, а не тільки під конкретний матч.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1) Charu C. A. Recommender Systems. New-York : Springer International Publishing, 2016. 518 p.

2) Knowledge-Based Recommendation Systems: A. Survey / S. Bouraga et al. *International Journal of Intelligent Information Technologies*. 2014. Vol. 10, no. 2. P. 1–19.

3) An Integrated Environment for the Development of Knowledge-Based Recommender Applications / A. Felfernig et al. *International Journal of Electronic Commerce*. 2006. Vol. 11, no. 2. P. 11–34.

4) Solving the apparent diversity-accuracy dilemma of recommender systems / T. Zhou et al. *Proceedings of the National Academy of Sciences*. 2010. Vol. 107, no. 10. P. 4511–4515.

5) Towards Diversity in Recommendations using Social Networks / S. Sheth et al. *Department of Computer Science, Columbia University*. 2011. Vol. 2, no. 4. P. 510–531.

6) Football in Brazil: Origin and Evolution of Training Methodologies. *Revista Científica Multidisciplinar Núcleo do Conhecimento*. URL: https://www.nucleodoconhecimento.com.br/education-physics-en/football-in-brazil (дата звернення: 17.04.2024).

7) Gama K. V. 20 Legendary Coaches That Changed World Football. *Bleacher Report*. URL: https://bleacherreport.com/articles/1102600-20-legendary-coaches-that-changed-world-football (дата звернення: 17.04.2024).

8) Distributed representations of words and phrases their compositionality / T. Mikolov et al. *Neural Information Processing Systems Foundation, Inc*. 2013. Vol. 9, no. 4. P. 119–129.

9) System response time and user satisfaction. An experimental study of browser-based applications / J. Hoxmeier et al. *Americas Conference on Information Systems*. 2000. Vol. 1, no. 12. P. 10–13.

10) Kolodner J. Case-based Reasoning. Burlington : Morgan Kaufmann Publishers, 1988. 482 p.

11) Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches / A. Aamodt et al. *AI Communications*. 1994. Vol. 7, no. 1. P. 39–59.

12) Retrieval, reuse, revision and retention in case-based reasoning / R. LOPEZ DE MANTARAS et al. *The Knowledge Engineering Review*. 2005. Vol. 20, no. 3. P. 215–240.

13) Expertclerk: A conversational case-based reasoning tool for developing salesclerk agents in e-commerce webshops / H. Shimazu et al. *Artificial Intelligence Review*. 2002. Vol. 18, no. 3/4. P. 223–244.

14) Improving case representation and case base maintenance in recommender systems / M. Montaner et al. *Proceedings of the 6th European Conference On Case Based Reasoning*, Vol. 6, no. 2. 2002. P. 234–248.

15) Knowledge-based recommender systems. / R. Burke et al. *Encyclopedia of Library and Information Science*. 2000. P. 69-93.

16) On the role of diversity in conversational recommender systems. / L. McGinty *International Conference on Case-Based Reasoning*. 2003. Vol. 5, no. 1. P. 276–290.

17) Passing and Movement Patterns URL: https://www.professionalsoccercoaching.com/soccer-drills-for-passing-and-movement/passing-and-movement-patterns (дата звернення 17.04.2024).

18) Tactical Dynamics | Guide to Football. URL: https://www.guidetofootball.com/tactics/tactical-dynamics (дата звернення 17.04.2024).

19) Ultimate Guide: How to Play as a Winger - The Titans Football Academy. *The Titans Football Academy - Football Academy in Hertfordshire, Buckinghamshire, London and Essex.* URL: https://thetitansfa.com/ultimate-guide-how-to-play-as-a-winger (дата звернення 17.04.2024).

20) Counter-pressing and the gegenpress: football tactics explained - The Coaches' Voice. *The Coaches' Voice*. URL: https://www.coachesvoice.com/cv/counter-pressing-gegenpressing-football-tactics-explained-klopp-guardiola-bielsa-hasenhuttl (дата звернення 17.04.2024).

21) Key Principles of Pressing High In The Final Third. URL: https://footballdna.co.uk/features/pressing-high-in-the-final-third (дата звернення 17.04.2024).