частині даних), проте ці метрик не використовувались для навчання моделі.

Для оцінки роботи діалогової моделі буде використовуватись внутрішні метрики, так як мануальна оцінку результату є ресурсозатратною.

Для обчислення метрик необхідно побудувати матриці L_1 та L_2. Для матриці L_1 кожен стовбець та рядок представляють собою слово, а $l1_{ij}$ – кількість разів яку слова i та j зустрічались в одному тексті. Для матриці L_2 кожен стовбець та рядок також представляють собою слово, а $l2_{ij}$ – кількість разів яку слова i та j зустрічались в одному реченні.

На основі даних таблиць проводяться дві відповідні кластеризації. Далі для кожного речення згенерованого системою проводяться дві оцінки.

Оцінка цілісності побудови речень – члени речення кластеризується відповідно матриці L_1 та якщо вони потрапляють у різні класи то речення не є цілісним. Для кожного слова необхідно обрати топ n кластерів до яких вони належать, та для речення обраховується мінімальна кількість кластерів у яку вкладаються всі слова. Чим менший цей показник, тим однорідніші речення.

Оцінка однорідності тексту – кожен член речення кластеризується відповідно матриці L_2 - для кожного слова обраховується його степінь приналежності до кластеру. Для кожного речення обраховується приналежність до кластеру як

$$l2 = \max(\frac{\sum_i^n l2_i}{n}), \text{ де}$$

$\sum_i^n l2_i$ – сума приналежностей до кластеру,

n – кількість слів у реченні.

Для тексту обраховується кількість кластерів , які зустрічаються у реченнях . Чим менше кластерів – тим більш однорідний текст.

Висновок

У статті детально описана підготовка даних до навчання моделі, сама модель, її використання та оцінка її роботи. Підготовка даних є не менш важливою ніж сама модель, оскільки за допомогою підготовки можна перетворити нестандартні та специфічні дані до загальних моделей. Також від того, як будуть представлені данні залежить які взаємозв'язки вдасться виявити. Від специфіки даних залежить не тільки їх підготовка, а і оцінка результатів роботи. В даній статті було запропоновано навчити окрему модель для оцінки отриманих результатів, оскільки більш лінійні метрики були б менш інформативними.

Список джерел

1. Гмурман В. Е. Теория вероятностей и математическая статистика //М.: Высшее образование, 2005
2. Pettie Seth, Ramachandran Vijaya An optimal minimum spanning tree algorithm //Journal of the Association for Computing Machinery, 2002, С. 16–34
3. Djauhari M., Gan S. Optimality problem of network topology in stocks market analysis //Physica A: Statistical Mechanics and Its Applications, 2015, С. 108-114
4. Abraham Ittai, Delling Daniel, Goldberg Andrew V., Werneck Renato F. A Hub-Based Labeling Algorithm for Shortest Paths on Road Networks //Symposium on Experimental Algorithms, 2011, С. 230–241

УДК 004.8

КАРПА М. В.,
БАКЛАН І.В.

АРХІТЕКТУРНЕ РІШЕННЯ ДЛЯ АВТОМАТИЧНОГО ПОШУКУ ТЕХНОЛОГІЧНИХ РЕЦЕПТІВ

В даній статті наведено формальне визначення рецептів та інгредієнтів та їх взаємозалежність. Описано спосіб автоматичного визначення рецептів на основі наявних інгредієнтів, використовуючи при цьому експертну систему. Розглянуто програмне середовище для розробки таких систем CLIPS. Представлено приклад мобільних та серверних додатків, що взаємодіють із такою експертною системою.

КЛЮЧОВІ СЛОВА: ЕКСПЕРТНА СИСТЕМА, CLIPS, ІНГРЕДІЄНТ, РЕЦЕПТ

This article provides a formal definition of recipes and ingredients and their interdependence. It contains a description of a method of automatic determination of recipes based on available ingredients, using an expert system. There is also a brief overview of the software environment for the development of such systems called CLIPS. It presents some examples of mobile and server applications that interact with such expert system.
KEYWORDS: EXPERT SYSTEM, CLIPS, INGREDIENT, RECIPE

1. Вступ

В повсякденному житті нерідко виникає потреба у визначенні рецептів на основі наявних інгредієнтів. Очевидним прикладом є кулінарна сфера, де в якості інгредієнтів виступають певні компоненти, з яких можна приготувати деяку страву. Серед них можна навести наступні: мука, цукор, сіль, молоко, тощо. Рецепти ж представляють собою певний набір інструкцій для приготування таких страв. Інший приклад - медицина, де в якості інгредієнтів може виступати список симптомів, а рецепти представляють собою певну послідовність кроків для одужання пацієнта. В даній статті описано використання експертної системи для формування бази знань та автоматичного визначення рецептів на основі певних інгредієнтів.

2. Формальне визначення понять «інгредієнт» та «рецепт»

Як зазначалось раніше поняття «інгредієнт» та «рецепт» можуть використовуватись в багатьох сферах життєдіяльності, серед яких: кулінарія, медицина, тощо. В деяких галузях не існує таких термінів, проте передбачені інші поняття, що своєю суттю можуть підпадати під визначення термінів «інгредієнт», «рецепт».

В кулінарії інгредієнти представляють собою речовину, яка входить до складу певної суміші (страви), а рецепт – набір інструкцій, що визначають: яким чином приготувати ту чи іншу страву. В сфері будівництва під інгредієнтом можна розуміти набір ресурсів, необхідних для виготовлення того чи іншого будинку, а рецепт – послідовність кроків для побудови даної споруди.

Перш ніж перейти до формального визначення рецепту слід спершу також ознайомитись із визначенням поняття «алгоритм». У інформатиці та математиці алгоритм - це скінченна послідовність чітко визначених інструкцій, які зазвичай використовуються для розв’язання класу конкретних задач або для виконання обчислень.

Отже, рецепт - це деякий алгоритм, для запуску якого потрібні вхідні параметри - підмножина множини інгредієнтів. Множина інгредієнтів та рецептів містить набір підмножин інгредієнтів і рецептів. Під поняттям «підмножина» можна розуміти певну галузь використання. Наприклад: кулінарія, медицина, будівництво, тощо. Інгредієнти однієї підмножини не можуть виступати вхідними параметрами для рецептів іншої підмножини. На рис. 1 представлено схему, що описує підмножини інгредієнтів, рецептів та їх взаємозалежність.

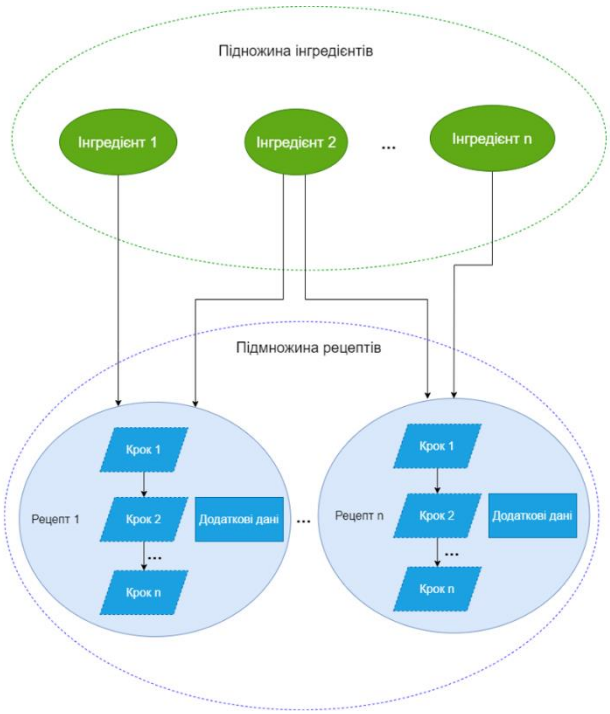

Рис. 1. Залежність між інгредієнтами та рецептами

3. Експертна система

Експертна система - це методологія адаптації алгоритму успішних рішень однієї сфери науково-практичної діяльності в іншу. З поширенням комп'ютерних технологій — це тотожна інтелектуальна комп'ютерна програма, що містить знання й аналітичні здібності одного чи кількох експертів в деякій

галузі застосування і здатна робити логічні висновки на основі цих знань, тим самим забезпечуючи вирішення специфічних завдань (консультування, навчання, діагностування, тестування, проектування тощо) без участі експерта (фахівця в конкретній проблемній галузі). Експертна система відрізняється від інших прикладних програм наявністю таких ознак:

– моделює механізм мислення людини під час розв'язання задач в цій предметній галузі;

– окрім виконання обчислювальних операцій, формує певні висновки, базуючись на тих знаннях, якими вона володіє. Компонент збереження знань прийнято називати базою знань;

– застосовується для предметів реального світу, операції з якими зазвичай вимагають великого досвіду, накопиченого людиною.

– має яскраво виражену практичну направленість для застосування в науковій або комерційній сфері;

– записує шлях, який вона проходить для обрання певного рішення. Це дає можливість пояснити, чому запропоновано саме цей розв'язок і довести його обґрунтованість. Користувач має змогу прослідкувати за ходом думок системи та зрозуміти, чому було обране те чи інше рішення.

4. CLIPS

Для створення експертної системи було обрано CLIPS, що з англійської розшифровується як C Language Integrated Production System. CLIPS являє собою програмне середовище для розробки експертних систем.

Основними поняттям в CLIPS є «правило» і «факт». Факт – це певна інформація. Правило – це умовний блок, що виконує якусь дію у тому випадку, коли виконується певна умова. Результатом роботи правила може бути створення нового факту.

5. Приклад використання

При побудові експертної системи було визначено ролі інгредієнтів та рецептів. Інгредієнти представляють собою факти типу «ingredient», а рецепт – це факт типу «recipe». На рис. 2 представлено факт інгредієнту.

```
(assert (ingredient milk))
(assert (ingredient egg))
(assert (ingredient apple))
(assert (ingredient banana))
```

Рис. 2. Правила інгредієнтів

На рисунку 3 зображено шаблон правила рецепту. Як видно рецепт може зберігати не тільки список кроків, але і додаткову необхідну інформацію.

```
(deftemplate recipe
  (slot id)
  (slot shortDescription)
  (slot name)
  (slot logoPath)
  (multislot ingredients)
  (multislot steps)
)
```

Рис. 3. Шаблон правила рецептів

Коли користувач хоче скористатись експертною системою, він передає список назв інгредієнтів в якості вхідних параметрів. На основі них створюються відповідні факти інгредієнтів. Для визначення рецептів використовуються правила, які передбачають наступну умову: якщо існують факти деяких інгредієнтів, тоді створити факт відповідного рецепту. На рисунку 4 представлено приклад такого правила.

```
(defrule mashed_potatoes
  (ingredient "potato")
  (ingredient "salt")
  (ingredient "water")
  (ingredient "milk")
  (ingredient "butter")
  =>
   (assert
    (recipe
      (id "1")
      (name "Mashed potato")
      (shortDescription "Very tasty mashed potatos")
      (logoPath "https://www.recipetineats.com/wp-content/upload
      (ingredients
        "2 pounds baking potatoes, peeled and quartered"
        "2 tablespoons butter"
        "1 cup milk"
        "salt and pepper to taste"
      )
      (steps
        "Bring a pot of salted water to a boil. Add potatoes and
        "In a small saucepan heat butter and milk over low heat
      )
    )
  )
)
```

Рис. 4. Приклад правила

5. Серверний і мобільний додатки

Для представлення використання експертної системи кінцевим користувачем було спроектовано і реалізовано серверний та мобільний додатки.

Серверний додаток представляє собою програмне забезпечення написане, використовуючи мову програмування Go, та містить кінцеву точку POST /recipes, який на вхід приймає список інгредієнтів, а повертає список рецептів. Для визначення рецептів він використовує експертну систему.

Мобільний додаток представляє собою програмне забезпечення, написане використовуючи мову програмування Dart і рушій Flutter. Користувач має змогу задати список параметрів, надіслати запит до веб-серверу та переглянути список і детальну інформацію про рецепти.

На рис. 5 представлено задання користувачем списку інгредієнтів.

Рис. 5. Задання списку інгредієнтів

На рисунку 6 зображено список знайдених рецептів.

На рисунку 7 зображено детальну інформацію про знайдений рецепт.

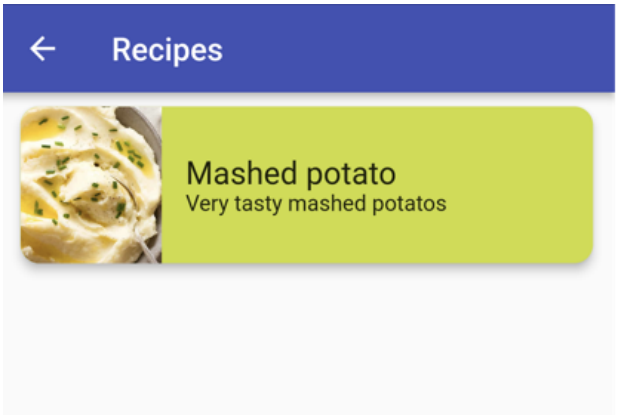

Рис. 6. Знайдені рецепти

Рис. 7. Детальна інформація про рецепти

Висновок

В результаті проведеного дослідження було сформовано формальне визначення понять «інгредієнт» і «рецепт», спроектовано експертну систему, використовуючи для цього програмний рушій CLIPS, а також створено мобільний та серверний додатки для роботи з даною експертною системою. Експертна система складається з фактів і правил. І, як було показано раніше, процес створення нових правил для визначення рецептів є неймовірно простим та зрозумілим. В різних випадках, в залежності від ситуації, такі правила можна

регулювати, додаючи додаткові умови або редагуючи вже існуючі. Окрім інтуїтивно зрозумілого задання правил визначення рецептів на основі наявних інгредієнтів, використання експертної системи також передбачає і отримання усіх унікальних функціональностей, що притаманні лиш цій системі, включаючи: моделювання рішення людиною-експертом, представлення ходу думок, що привели до прийняття того чи іншого рішення, тощо.

Список літератури

1. Jackson P. Introduction To Expert Systems – 1998. – Vol.1.1– P. 542
2. Riley G. CLIPS Reference Manual // Basic Programming Guide – 2017 – Vol.1. – P. 3-18
3. Giarratano J., Riley G. Principles and Programming // Expert Systems – 2005 – Vol.1. – P. 288

УДК 004.02

ШВЕЦЬ Е.Я.,
МУХА І.П.

АВТОМАТИЗАЦІЯ МУЛЬТИПЛАТФОРМЕННОЇ РЕАЛІЗАЦІЇ СЕРВІСІВ РОБОТИ З БАЗОЮ ДАНИХ

В даній статті проведено аналіз існуючого процесу створення мультиплатформенного мобільного застосунку та запропоновано метод автоматизованого створення програмних компонетів, що реалізують сервіси роботи з БД, який передбачає транспіляцію вхідних описів сутностей бази даних та налаштувань (DAO, CRUD, DI), представлених на спеціально розробленій предметно-орієнтованій мові, у SQL- та Kotlin-компоненти загального коду мультиплатформенного мобільного проєкту.
Ключові слова: Первинний мультиплатформенний мобільний проєкт, Kotlin Multiplatfrom Mobile, автоматизація, мультиплатформенний мобільний застосунок.

This article analyzes the existing process of creating a multi-platform mobile application and proposes a method of automated creation of software components that implement database services, which involves translating input descriptions of database entities and settings (DAO, CRUD, DI), presented on a specially designed subject. oriented language, in SQL and Kotlin components of the common code of a multiplatform mobile project.
Keywords: Primary multiplatform mobile project, Kotlin Multiplatfrom Mobile, automation, multiplatform mobile application.

1. Вступ

Останнім часом все більшої популярності у світі мобільної розробки набуває мультиплатформенна розробка мобільних додатків під ОС Android та iOS. Існує ряд технологій, які дозволяють це зробити, зокрема, Xamarin, React Native, Flutter, Kotlin Multiplatform Mobile (KMM) [1]. Найбільш перспективною серед них є KMM, яка має помітні переваги у порівнянні із іншими альтернативними технологіями.

Оскільки версії одного і того ж застосунку для Android та iOS часто мають багато спільного, наприклад, бізнес-логіку застосунку (зокрема, код, який відповідає за автентифікацію, керування даними, аналітику), то для різних платформ цілком природним є використання загального коду, який реалізує відповідний функціонал.

KMM SDK якраз і дозволяє оптимізувати розробку мультиплатформенного мобільного застосунку шляхом одноразового написання блоку універсального коду, а потім спільного використання цього коду на різних платформах. Платформний же код використовується тільки там, де це необхідно — для реалізації інтерфейсу або під час роботи з платформо-залежними API.

Використання KMM хоча і прискорює процес створення мультиплатформенних мобільних застосунків за рахунок спільного