КПІ ім. Ігоря Сікорського

МАТЕРІАЛИ ЧЕТВЕРТОЇ ВСЕУКРАЇНСЬКОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ МОЛОДИХ ВЧЕНИХ ТА СТУДЕНТІВ «ІНФОРМАЦІЙНІ СИСТЕМИ ТА ТЕХНОЛОГІЇ УПРАВЛІННЯ»

1978

АСОІУ

24 та 30 квітня
Україна,Київ

Оргкомітет конференції

Голова організаційного комітету: О.А. Павлов – в.о. зав. кафедри АСОІУ, д.т.н., професор.

Члени організаційного комітету:

П.І.П.	Посада
О.А. Павлов	В.о. зав. кафедри АСОІУ, професор
І.В. Стеценко	Професор кафедри АСОІУ
І.П. Муха	Доцент кафедри АСОІУ
О.В. Гавриленко	Доцент кафедри АСОІУ
К.І. Ліщук	Доцент кафедри АСОІУ
О.Г. Жданова	Доцент кафедри АСОІУ
Т.О. Тєлишева	Доцент кафедри АСОІУ
Ю.О. Олійник	Старший викладач кафедри АСОІУ

Члени програмного комітету:

П.І.П.	Посада
В.І. Литвиненко	Професор ХНТУ
Г.В. Рудакова	Професор ХНТУ
І.В. Баклан	Доцент кафедри АСОІУ
М.О. Сперкач	Доцент кафедри АСОІУ
Е.В. Жаріков	Доцент кафедри АСОІУ
Ю.М. Селін	Доцент кафедри АСОІУ
О.С. Жураковська	Доцент кафедри АСОІУ
В.Д. Попенко	Доцент кафедри АСОІУ
М.І. Цюцюра	Доцент кафедри АСОІУ

IV Всеукраїнська науково-практична конференція молодих вчених та студентів «Інформаційні системи та технології управління – ІСТУ-2020». Секція кафедри автоматизованих систем обробки інформації і управління. Матеріали конференції. – Київ. – 2020. 24, 30 квітня 2020р. – 182 с.

У збірник включені тези доповідей, які були представлені на конференції “Інформаційні системи та технології управління – ІСТУ-2020” в секції кафедри автоматизованих систем обробки інформації і управління. В доповідях розглянуті наукові та методичні питання щодо сучасних аспектів інформатики та обчислювальної техніки.

Редакційна колегія:

Гавриленко О.В., к.ф.-м.н., кафедра АСОІУ НТУУ «КПІ ім. Ігоря Сікорського»,
Муравйова І. М., інженер ІІ категорії кафедри АСОІУ

Дизайн титульної сторінки: Майєр З.О.

ЗМІСТ

УДК 004.41

ПЕДОС В. М.

МЕХАНІЗМ ІНТЕГРАЦІЇ СПЕЦІАЛІЗОВАНОГО ОБЛАДНАННЯ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ ПІДПРИЄМСТВА ТОРГІВЛІ

Стаття присвячена засобам автоматизації бізнес-процесів підприємств торгівлі із залученням спеціалізованого обладнання. Розглядається структура типової інформаційної системи підприємства на базі програмного комплексу "1С: Підприємство 8", а також актуалізується проблема інтеграції офіційно непідтримуваного торгівельного обладнання. Наводиться опис розробленого універсального механізму інтеграції стороннього торгівельного обладнання до існуючої інформаційної системи підприємства торгівлі.

АВТОМАТИЗАЦІЯ, ТОРГІВЕЛЬНЕ ОБЛАДНАННЯ, ІНТЕГРАЦІЯ, СКЛАДСЬКИЙ ОБЛІК

The article is devoted to the means of trade enterprises business processes automation with specialized equipment involvement. Provides an overview of a typical enterprise information system structure based on the software complex "1C: Enterprise 8", and actualizes the problem of officially unsupported trading equipment integration. Describes the developed universal mechanism of third-party trade equipment integration into an existing trade enterprise information system.

AUTOMATION, TRADE EQUIPMENT, INTEGRATION, WAREHOUSE ACCOUNT

1. Вступ

За сучасних умов мінливого зовнішнього середовища одним з головних завданням підприємства стає швидке реагування на зміни та оперативне впровадження відповідних заходів в організації і здійсненні власної підприємницької діяльності. Продуктивна робота підприємства торгівлі передбачає чітко розроблені та сплановані бізнес-процеси. Вплив людського фактору на функціонування одного з ключових бізнес-процесів може негативно позначитися на діяльності підприємства в цілому, тому впровадження систем автоматизації таких процесів є основою успішної боротьби в умовах ринку та перспективним напрямком розробки програмного забезпечення.

Під час автоматизації бізнес-процесів людина не виключається повністю з ланцюжка створення додаткової вартості, вона зберігає свою присутність у більшості функціональних областей діяльності підприємства. Автоматизація швидше означає найбільш раціональний розподіл обчислювального та виробничого навантаження між людиною і машиною, баланс якого залежить від конкретного підприємства та цілей автоматизації. Як правило, автоматизуються ключові бізнес-процеси діяльності підприємства: формування замовлень, виконання заявок клієнтів, розробка і запуск нової продукції і т. д., а також інші нескладні, але численні і рутинні процеси.

Автоматизація бізнес-процесів – це широкий клас завдань, що не обмежується рухом і обробкою документів. До їх складу входять різноманітні операції, що виконуються співробітниками. У ході бізнес-процесу можуть оброблятися різні документи і відбуватися взаємодія з зовнішніми IT-системами.

2. Аналіз предметної області

На території країн СНД та на українському ринку зокрема фактичним монополістом у сфері автоматизації бізнес-процесів торгівельних підприємств є продукти компанії 1С. Програмний комплекс "1С:Підприємство 8" включає в себе платформу і прикладні рішення, розроблені на її основі, для автоматизації діяльності організацій і приватних осіб[1]. Сама платформа не є програмним продуктом для використання кінцевими користувачами, які зазвичай працюють з одним з багатьох прикладних рішень (конфігурацій), розроблених на даній платформі. Такий підхід дозволяє автоматизувати різні види діяльності, використовуючи єдину

технологічну платформу.

Однак іноді виникає необхідність не лише автоматизувати процес, використовуючи програмні засоби, а й задіяти спеціалізоване обладнання.

Для ефективної автоматизації складських та торгівельних бізнес-процесів підприємства зазвичай необхідне використання додаткового обладнання, такого як:

- сканери штрих-кодів,
- термінали збору даних,
- інформаційні дисплеї,
- тощо.

Програмний продукт "1С:Підприємство 8" має вбудовані засоби роботи з обмеженим набором торгівельного обладнання, що входить до так званого списку офіційно підтримуваних пристроїв. Але у більшості випадків виникає необхідність інтеграції пристроїв, що офіційно не підтримуються фірмою 1С [2]. Зазвичай взаємодія з таким обладнанням реалізовується шляхом внесення змін до існуючої конфігурації, що створює додаткові ризики та потенційно впливає на стабільність інформаційної системи підприємства, а також ускладнює, а іноді і зовсім унеможливлює автоматичне оновлення конфігурації. Даний підхід потенційного може призводити до порушення функціонування ключових бізнес-процесів підприємства торгівлі.

Тому постає питання розробки універсального підходу, призначеного для інтеграції стороннього торгівельного обладнання з програмним комплексом "1С:Підприємство 8", що з однієї сторони забезпечив би тісну взаємодію з платформою 1С, а з іншої – потребував би внесення мінімальних змін до існуючих конфігурацій.

3. Модель механізму інтеграції спеціалізованого обладнання

Інформаційна система будь-якого торгівельного підприємства, що використовує для автоматизації бізнес-процесів продукт "1С:Підприємство 8", складається з серверу 1С, та клієнтів, під'єднаних до сервера з використанням різноманітних протоколів.

Зважаючи на ризики, пов'язані з інтеграцією офіційно непідтримуваного спеціалізованого обладнання до системи на базі платформи "1С: Підприємство 8", доцільним є реалізація необхідного для роботи торгівельного обладнання функціоналу за межами існуючої інформаційної мережі підприємства та організація каналу зв'язку між розроблюваним програмним забезпеченням та сервером "1С:Підприємство 8".

При таких умовах сервер автоматизації (рис. 1) виступатиме проміжною ланкою між платформою "1С:Підприємство 8" та стороннім торгівельним обладнанням, не вимагаючи внесення жодних змін до існуючої конфігурації, тим самим зводячи до мінімуму ризики порушення функціонування

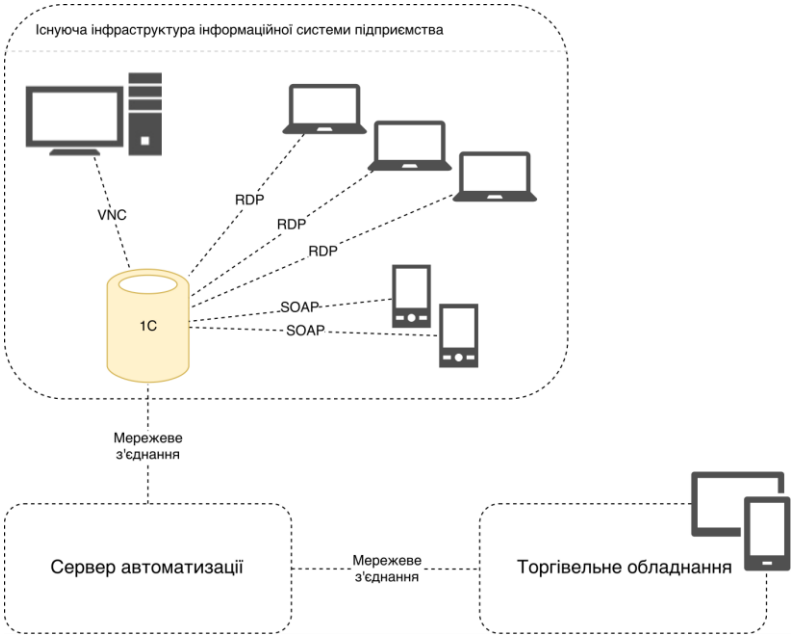


Рис. 1. Концептуальна модель інтеграції торгівельного обладнання

існуючих бізнес-процесів підприємства.

Взаємодія серверу автоматизації як з торгівельним обладнанням, так і з платформою 1С відбуватиметься через мережу.

4. Архітектура сервера автоматизації

У загальному випадку архітектура сервера автоматизації має вигляд наведений на рисунку 2.

Структурно сервер автоматизації

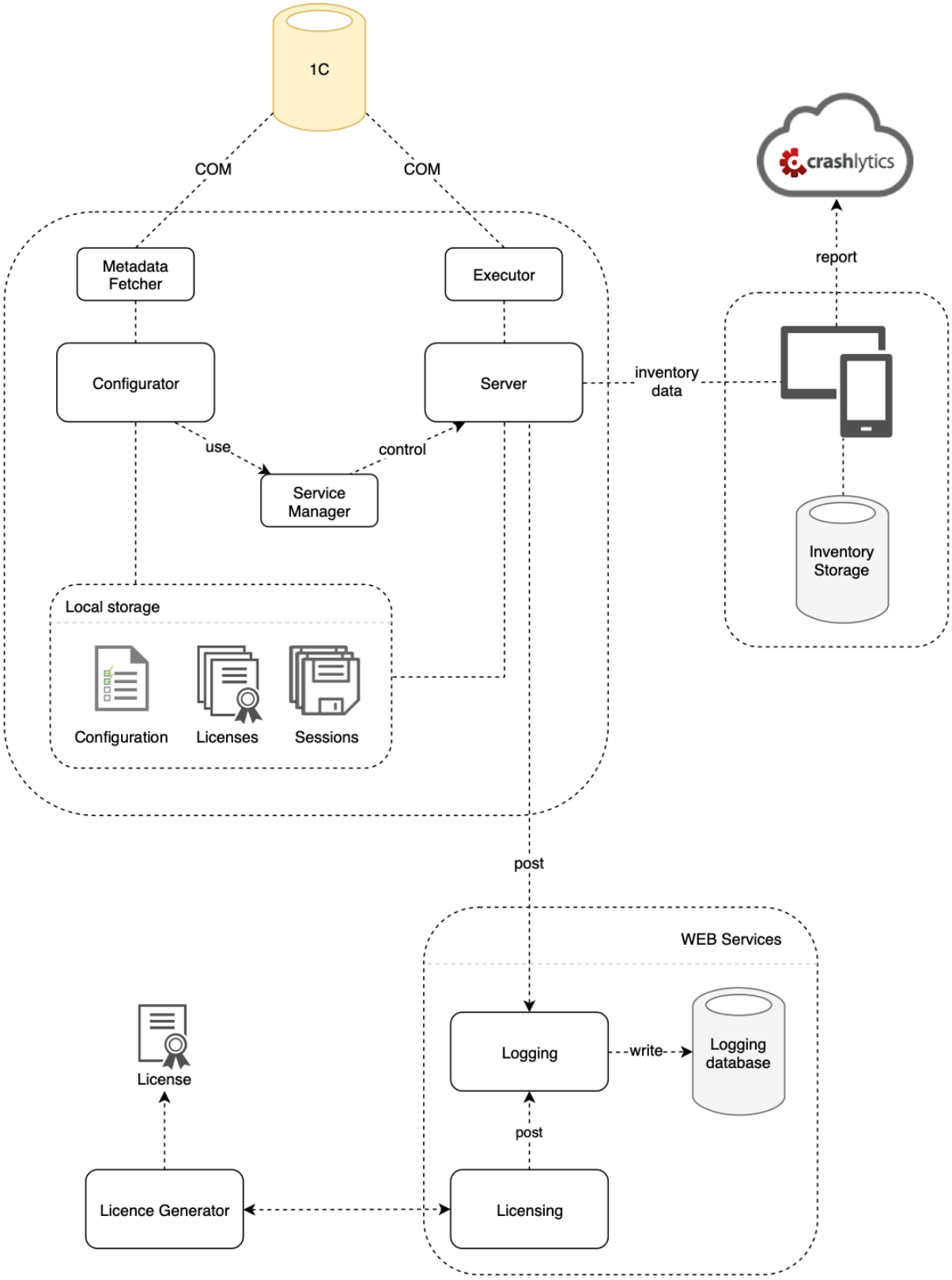


Рис. 2. Архітектура сервера автоматизації

розділений на дві основні складові: безпосередньо сервер автоматизації та додаток для здійснення адміністративних та конфігураційних дій. Сервер автоматизації представлений у вигляді служби Windows. Сервер виступає проміжною ланкою між торгівельним обладнанням та платформою 1С, надаючи з однієї сторони сторонньому обладнанню інтерфейс для роботи за базою даних 1С, а з іншою, забезпечуючи тісну інтеграцію з платформою, не вимагаючи внесення жодних змін до існуючої конфігурації.

Для взаємодії з платформою "1С:Підприємство 8" використовується окремий модуль – "Executor", що представлений окремим виконуваним файлом, та, відповідно, виконується в окремому процесі. Такий підхід дозволяє ізолювати різні сеанси роботи з базою даних 1С (вивантаження та завантаження даних) між собою, забезпечуючи більш ефективне керування ресурсами, виділеними платформою 1С для роботи через COM з'єднання, що є основним механізмом взаємодії з платформою з боку сервера автоматизації. Взаємодія між сервером та модулем "Executor" відбувається через мережеве з'єднання.

Взаємодія торгівельного обладнання та сервера автоматизації реалізується на основі технології віддаленого виклику ApacheThrift [3] через мережеве з'єднання.

Виконання адміністративних та конфігураційних дій здійснюється за допомогою окремого додатку – Конфігуратора, який призначений для управління роботою сервера вцілому, а також для налаштування параметрів окремих сервісів.

Для отримання необхідних у процесі налаштування даних та параметрів з 1С конфігуратор використовує окремий модуль "MetadataFetcher",що діє за схожим до модуля "Executor" принципом та призначений для отримання метаданих конфігурації 1С. Зв'язок між конфігуратором та модулем "MetadataFetcher" реалізований аналогічно до взаємодії між сервером автоматизації та модулем "Executor" – засобами технології віддаленого виклику ApacheThrift.

Для моніторингу стану сервісу та його керуванням використовується модуль "ServiceManager", що надає конфігуратору інтерфейс для керування відповідною службою Windows, а також забезпечує механізм моніторингу стану сервісу в режимі реального часу.

Також передбачений механізм логіювання подій до стандартного журналу подій Windows, а також на віддалений сервер логіювання.

Висновки

Аналіз предметної області в сфері автоматизації бізнес-процесів підприємств торгівлі виявив проблему інтеграції офіційно непідтримуваного торгівельного обладнання з програмним комплексом "1С: Підприємство 8".

Тому було обґрунтовано актуальність розробки універсального механізму інтеграції спеціалізованого обладнання до існуючої інформаційної системи підприємства торгівлі для вирішення даної проблеми.

Розроблений механізм інтеграції стороннього торгівельного обладнання до існуючого програмного комплексу, заснований на міжмережевій взаємодії сервера автоматизації, платформи "1С: Підприємство" та торгівельного обладнання.

Перелік посилань

1. Филиппов Е. В. Настольная книга 1С: Эксперта по технологическим вопросам – М.: 1С-Паблишинг, 2010. – 247 с.
2. Хрусталева Е. Ю. Технологии интеграции 1С:Предприятия – М.: 1С-Паблишинг, 2012. – 358 с.
3. Randy Abernethy. The Programmer's Guide to Apache Thrift. – Manning Publications Company, 2015. – 500 с.

УДК 004.75

ОШИЙКО Я.Р.,
ОЛІЙНИК Ю.О.

ЗАДАЧА ВИЯВЛЕННЯ ДЕЗІНФОРМАЦІЇ В ТЕКСТОВИХ ПОТОКАХ ДАНИХ

У цій статті розглянуто задачу виявлення дезінформації в текстових потоках даних. Сформульовано цілі та актуальність задачі. Проведено аналіз відомих досліджень, проаналізовано їх переваги і недоліки, виділені вимоги. Запропоновано підхід реалізації задачі.

КЛЮЧОВІ СЛОВА: МАШИННЕ НАВЧАННЯ, КЛАСИФІКАЦІЯ, ДЕЗІНФОРМАЦІЯ

This article discusses the problem of detecting misinformation in text data streams. The objectives and relevance of the task are formulated. The known researches are analyzed, their advantages and disadvantages are analyzed, and requirements are highlighted. The approach of task realization is suggested.

KEYWORDS: MACHINE LEARNING, CLASSIFICATION, DISINFORMATION

1. Вступ

Задача визначення елементів дезінформації в різних потоках інформації викликає неабиякий інтерес у дослідників з усього світу. Проведено чимало соціальних досліджень щодо впливу неправдивої інформації на суспільство та реакцію людей на цю інформацію.

Термін "фейкові новини" може застосовуватися до різних форм контенту. Це може бути оманливе відео, змінене зображення, упереджена новина, що не відображає реальності, або публікація в соціальних мережах. Підроблені новини також можна знайти в різних сферах, таких як політика, фінанси, спорт, розваги та інші. Фейковими новинами може бути будь-який контент, який є неправдивим та створеним для того, щоб переконати читача повірити в те, що не відповідає дійсності.

Реальна проблема з'являється тоді, коли люди починають вірити в завідому неправдиву інформацію не перевіряючи її. [1] Тому в сучасному світі з появою соціальних мереж та месенжерів задача класифікації дезінформації вийшла на новий рівень. Ця проблема мала великий вплив на вибори президента в США в 2016 році. Також постійна дезінформація новин про події на Донбасі впливають на велику кількість населення та сприяють неякісній оцінці ситуації.

На сьогодні існує дуже мало інструментів та програмного забезпечення для якісної оцінки правдивості потоку текстових даних. Тому поставлена задача дослідити різні підходи до задачі виявлення дезінформації в текстовій інформації, виявити їх особливості. Також запропонувати реалізацію задачі на прикладі одного з алгоритмів класифікації.

2. Методи комп'ютерної лінгвістики для виявлення елементів дезінформації

Методи комп'ютерної лінгвістики використовують лінгвістичні елементи для того, щоб знайти шаблонипідробки новини та їх для виявлення неправдивої інформації. Дослідження[2] показали, що для створення дезінформації набір тексту використовується за певними правилами, щоб текст здавався правдивим. Підроблений текст має тенденцію до скорочення, передачі меншої кількості інформації, має негативний характер, містить не аналітичну думка, а більш неформальне мислення[2].

Методи комп'ютерної лінгвістикивикористовують ці шаблони, знайдені у тексті для виявлення фальшивих новин [3]. У машинному навчанні та лінгвістичній обробці природної мови слова повинні бути представлені в числовій формі, щоб програмне забезпечення розуміло та застосовувало алгоритми. Далі наведені методи, які дослідники використовували для перетворення неочищеного тексту в векторний вигляд.

2.1 Мішок слів (Bag of Words)

Мішок слів - це представлення тексту у вигляді частоти слів в ньому. Мішок слів схожий на об'єкт JSON, що містить список слів як ключі та частоту слова як значення. Використовуючи одиничні слова та n-грами, що переважають у підроблених виробах, цей підхід може перевірити домінування таких слів у тексті і зробити висновок, чи є матеріал підроблений[1].

Недоліком простого мішка слів є те, що в ньомувисоку частоту містять поширені слова, такі як займенники (я, він, вона, вони), сполучники тощо. Статистичний показник TF-IDF долає цю неефективність шляхом покарання частоти слів, які сильно вживаються в документі. Це забезпечує кращу подання частоти слів у документах, які були скориговані на слова, що трапляються часто.

Обмеженням підходу мішка слів полягає в тому, що цей метод орієнтований на відображенні частоти слів у документі без урахування будь-якої інформації про контекст. Інші методи, наприклад, представлення векторів слів, розглядає контекст слова в документі.

2.2 Синтаксичний аналіз

Оскільки аналізу слів недостатньо для передбачення елементів дезінформації, слід використовувати інші мовні підходи, такі як аналіз синтаксису та граматики. Використовується ймовірність без контексту Граматики (PCFG) для перетворення речень у дерева розбору, що описують структуру речень [4][5]. Синтаксичний аналіз широко використовується для аналізу настрою. Сторонні програмні продукти, такі як Stanford Parser [5] та синтаксис AutoSlog-TS аналізатор [4] часто поєднують з іншими лінгвістичними та мережевими підходами для кращої ефективності [4][6].

2.3 Семантичний аналіз

У більшості випадків правдивість певного тексту можна передбачити, вивчивши коментарі та подібні статті. Якщо більшість подібних статей не відповідають новині, що тестується, найімовірніше, що новина може бути упереджена або неправдва. Аналогічно коментарі до статті можуть бути використані для оцінки того, чи є факти у статті достовірні. Семантичний аналізу - це розширення n-граму моделі синтаксична модель разом із функціями сумісності профілю [6][9]. Основні недоліки такого підходу включають складність автоматичного пошуку подібних статей, перевірки сумісність профілю та облік різних слів, які мають на увазі те ж саме.

3. Мережеві методи виявлення елементів дезінформації

Інтернет містить величезну кількість метаданих, які можна використовувати для прогнозування надійності джерела. Twitter, Facebook та Google має великий мережевий набір даних, пов’язаний з кожним користувачем. Ці дані можуть використовуватися в мережевих підходах.

3.1 Метадані

Мережевий підхід вивчає метадані, наприклад, URL-адресу, автора тексту, вподобання у соціальних мережах тощо і прогнозую чи є джерело надійним. Метадані також використовуються для визначення поведінка сумнівних джерел [7]. Можна комбінувати аналіз метаданих в поєднанні з особливостями, створеними в результаті аналізу настроїв. Це дає можливість машинному навчанню дати більше шаблонів для класифікації, що призводить до кращої моделі.

3.2 Пов’язані дані та перевірка фактів

Правдивість новин можна також визначити, перевіривши факти, згадані в новинах. Проектування мережевого відношення для перевірки фактів є одним із підходів для цього. Перевірка фактів - це лише спосіб використання знань мережі для перевірки фактів. Певні фактичні твердження, що містять дані або відносини можна перевірити, використовуючи надійні джерела. Один з такихметодів- використання мереж знань і загальнодоступних структуровані дані, таких як онтологія DBpedia або Google Relation Extraction Corpus [4]. Хоча перевірка фактів має ряд переваг у правильній класифікації елементів дезінформації, цей метод є дуже затратним по часу.

4. Алгоритми класифікації для виявлення елементів дезінформації

Багато існуючих методів виявлення підроблених новин покладаються на виявлення ознак. Особливість виявлення ознак - це перетворення вихідних даних у

набір даних зі зменшеною кількістю змінних, що містить найбільш важливу інформацію [8]. Наведемо приклади алгоритмів для класифікації потоку текстових даних на наявність елементів дезінформації.

4.1 Наївний Байєсів класифікатор

Це ймовірнісний класифікатор, що використовує теорему Баєса для визначення ймовірності приналежності спостереження (елемента вибірки) до одного з класів при припущенні (наївному) незалежності змінних. Наївну байєсівську модель легко побудувати, без складної ітеративної оцінки параметрів, що робить її особливо корисною для дуже великих наборів даних. Незважаючи на свою простоту, класичний Байєсів класифікатор часто і широко використовується, оскільки перевершує більш складні методи класифікації. Теорема Байєса забезпечує спосіб обчислення задньої ймовірності $P\ (c \mid x)$, з $P\ (c), P\ (x)$ і $P\ (x \mid c)$. Наївний класифікатор Байєса припускає, що вплив значення предиктора (x) на даний клас (c) не залежить від значень інших предикторів. Це припущення називається класовою умовною незалежністю. Формула наївного баєса:

$$P\left(\frac{C}{X}\right) = \frac{P(x|c)P(c)}{P(x)}$$

4.2 Метод стохастичного градієнту

Стохастичний градієнтний спуск (часто скорочено SGD) - це ітеративний метод оптимізації цільової функції з відповідними властивостями гладкості (наприклад, диференційований або субдиференційований). Його називають стохастичним, оскільки метод використовує випадково відібрані (або перетасовані) зразки для оцінки градієнтів, отже, SGD може розглядатися як стохастичне наближення оптимізації градієнтного спуску. [8] Алгоритми стохастичного градієнта (SGD) успішно використовуються для навчання нейронних мереж. [9]

4.3 Метод Random forest

Випадковий ліс (англ. Random forest) — ансамблевий метод машинного навчання для класифікації, регресії та інших завдань, які оперують за допомогою побудови численних дерев прийняття рішень під час тренування моделі і продукують моду для класів (класифікацій) або усереднений прогноз (регресія) побудованих дерев. Недоліком є схильність до перенавчання.

4.4 Метод опорних векторів

В машинному навчанні метод опорних векторів (SVM) - це наглядові моделі навчання з пов'язаними алгоритмами навчання, які аналізують дані, що використовуються для класифікації та регресійного аналізу. Алгоритм навчання SVM будує модель, яка присвоює нові приклади тій чи іншій категорії, роблячи це неімовірнісним бінарним лінійним класифікатором (хоча методи, такі як масштабність Платта, існують для використання SVM в імовірнісному класифікаційному режимі). Це представлення прикладів як точок у просторі, зіставлене так, що приклади окремих категорій поділяються на максимально широкий проміжок, а нові приклади потім відображаються в той самий простір і прогнозується, що вони належать до категорії на основі на яку сторону розриву вони падають.

5. Підхід реалізації задачі

Перед запуском алгоритму відбувається первинна обробка тексту. Первинна обробка тексту необхідна для збільшення точності визначення елементів дезінформації. Обробка тексту поділяється на такі етапи:

1. Приведення слів до нижнього регістру.
2. Стеммінг.
3. Лематизація.
4. Видалення стоп-слів.
5. Нормалізація.
6. Видалення шуму.

Далі ми виконуємо вилучення та вибір ознак, використовуючи бібліотеки мови програмування python разом з алгоритмами, такими як мішок слів (bag of words), n-грам, а потім TF-IDF. Можна використати POS та word2vec для подальшого вилучення ознак. Далі будуються класифікатори для виявлення елементів дезінформації. Усі ознаки, отримані вище, передаються в класифікатори, а потім використовується sklearn з python для реалізації. Вибрані моделі - це наївний Байєсів класифікатор, логістична регресія, метод опорних векторів, метод стохастичного градієнту та

випадковий ліс. Після створення моделей ми порівнюємо значенняF1 кожної моделі, а потім перевіряємо матрицю помилок(confusion matrix). В результаті вибирається модель з найкращою точністю.

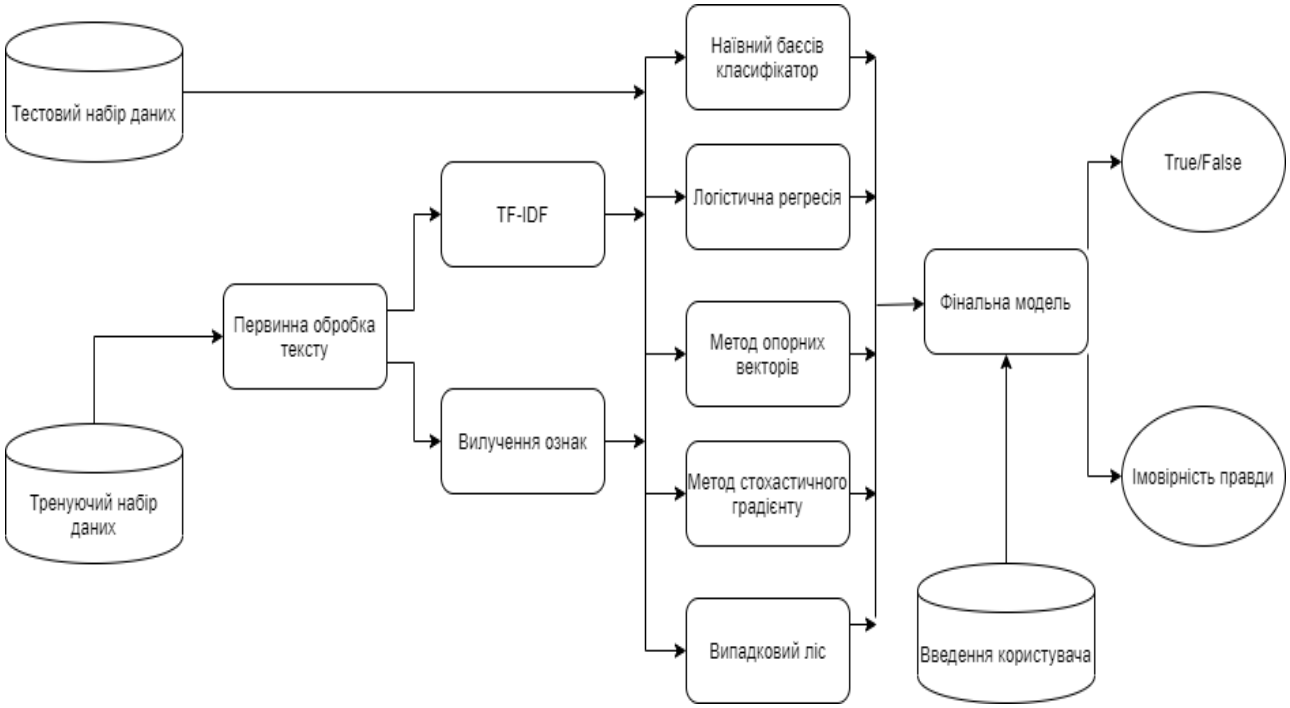


Рис. 1 Застосування методів класифікації для виявлення дезінформації в потоках текстових даних

6. Розповсюдження дезінформації в потоках текстових даних

Наступним кроком в задачі виявлення дезінформації є аналіз її розповсюдження. Так зазвичай фейкові новини перепублікуються іншими новинними сервісами з невеликими змінами. Для знаходження таких перепублікацій пропонується наступний підхід:

1. Збереження до БД новинного контенту з відміткою часу.
2. Знаходження фейкової новини методами, описаними в розділі 5 даної статті.
3. Знаходження новин, з великим коефіцієнтом подібності, що мають відмітки часу більші ніж знайдені новини в п.2.

Такий підхід дозволить знайти шляхи розповсюдження фейкових новин.

Висновки

Виявлення елементів дезінформації в потоках текстових даних – це проблема, яка є викликом, як для науковців, так і для лінгвістів. В сучасному світі ця проблема особливо критична, оскільки з появою соціальних мереж неправдива та неперевірена інформацію впливає на велику кількість людей. Пошук та класифікація цієї інформації зможе допомогти уникнути світових та локальних конфліктів. Для того, щоб розуміти процес формування фейкових новин необхідно проаналізувати правила, за якими вони формуються. Тому наступним етапом є аналіз розповсюдження дезінформації в залежності від часу. На основі отриманого шляху створення дезінформації можна зробити висновки, як протидіяти її поширенню.

Список використаних джерел

1. Media-Rich Fake News Detection: A Survey Shivam B. Parikh and Pradeep K. Atrey Albany Lab for Privacy and Security, College of Engineering and Applied Sciences University at Albany, State University of New York, Albany,NY,USA
2. DSKR Vivek Singh, Rupanjal Dasgupta, and I Ghosh. 2017. Automated fake news detection using linguistic analysis and machine learning. In International Conference on Social Computing,

Behavioral-Cultural Modeling, & Prediction and Behavior Representation in Modeling and Simulation (SBP-BRiMS). 1–3.Sibyl: A system for large scale supervised machine learning / [K. Canini, T. Chandra, E. Ie та ін.]. // Technical Talk. – 2012. – №1. – С. 113.

3. Leo Breiman and Adele Cutler. 2018. [Електронний ресурс] Режим доступу до ресурсу: https://www.stat.berkeley.edu/~breiman/ RandomForests/cc_home.htm

4. Niall J Conroy, Victoria L Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. In Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community. American Society for Information Science, 82.

5. Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In Proceedings of LREC, Vol. 6. Genoa Italy, 449–454.

6. Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Syntactic stylometry for deception detection. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2. Association for Computational Linguistics, 171–175

7. Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. 2010. Who is tweeting on Twitter: human, bot, or cyborg?. In Proceedings of the 26th annual computer security applications conference. ACM, 21–30.Caffe: Convolutional architecture for fast feature embedding / [Y. Jia, E. Shelhamer, J. Donahue та ін.]. // In Proceedings of the 22nd ACMinternational conference on Multimedia. – 2014. – С. 675–678.

8. Mei, Song (2018). "A mean field view of the landscape of two-layer neural networks". Proceedings of the National Academy of Sciences.115(33):E7665–E7671. doi:10.1073/pnas.1806579115. PMC 6099898. PMID 30054315

9. Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop.Neural Networks: Tricks of the trade,LNCS,1524, 1998.]Richtárik P. Distributed coordinate descent method for learning with big data / P. Richtárik, M. Takáč. // The Journal of Machine Learning Research 17. – 2016. – №1. – С. 2657–2681.

10. V. L. Rubin, N. J. Conroy, and Y. Chen, “Towards news veri-fication: Deception detection methods for news discourse,” in Hawaii International Conference on System Sciences, 2015.

11. Y. Chen, N. J. Conroy, and V. L. Rubin, “Misleading online content: Recognizing clickbait as false news,” in Proceedingsof the 2015 ACM on Workshop on Multimodal Deception Detection, pp. 15–19.

12. B. Markines, C. Cattuto, and F. Menczer, “Social spam detection,” in Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web. ACM, 2009, pp. 41–48.

13. N. J. Conroy, V. L. Rubin, and Y. Chen, “Automatic deception detection: Methods for finding fake news,” Proceedings of the Association for Information Science and Technology, vol. 52, no. 1, pp. 1–4, 2015

14. D. S. K. R. Vivek Singh, Rupanjal Dasgupta and I. Ghosh, “Automated fake news detection using linguistic analysis and machine learning,” in International Conference on SocialComputing, Behavioral-Cultural Modeling, & Prediction and Behavior Representation in Modeling and Simulation (SBP-BRiMS), 2017, pp. 1–3. Strategies and principles of distributed machine learning on big data / [E. Xing, Q. Ho, P. Xie та ін.]. // Engineering 2. – 2016. – №2. – С. 179–195.

15. Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In International Conference on Machine Learning. 1188–1196. [12] David M Markowitz and Jeffrey T Hancock. 2014. Linguistic traces of a scientific fraud: The case of Diederik Stapel. PloS one 9, 8 (2014), e105937.

16. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. Learning to Compose Words into Sentences with Reinforcement Learning [Електронний ресурс] / [D. Yogatama, P. Yogatama, C. Dyer та ін.]. – 2016. – Режим доступу до ресурсу: http://arxiv.org/abs/1611.09100.

УДК 004.414

МАЛЯРЧУК К.А.,
ГОБОВ Д.А.

СУЧАСНЕ РОЗУМІННЯ БІЗНЕС-АНАЛІЗУ В ІТ

Розвиток та ускладнення проектів по автоматизації і оптимізації бізнес-процесів призвели до формування нової дисципліни «бізнес-аналіз». Вона є розвитком напрямку інженерії вимог і спрямована на виявлення бізнес-потреб підприємства, визначення найкращого рішення та опису вимог до нього за допомогою спеціалізованих технік та підходів. На поточний момент сформовано декілька зводів знань, що визначають перелік робіт по бізнес-аналізу та є базисом для практичної діяльності бізнес-аналітиків в ІТ-галузі. Дана робота присвячена огляду, аналізу та порівняльному аналізу зводів знань, розробленими провідними міжнародними інституціями організаціями «Міжнародний Інститут Бізнес Аналізу (InternationalInstituteofBusinessAnalysis, IIBA)» та «Інститут Управління Проектами (ProjectManagementInstitute, PMI)» з точки зору робіт по бізнес-аналізу в ІТ-проектах.

КЛЮЧОВІ СЛОВА*: бізнес-аналіз, IIBA, PMI, галузі знань бізнес-аналізу*

The development and complication of projects on automation and optimization of business processes have led to the formation of a new discipline "business analysis". It extends a requirements engineering specialty and is aimed at identifying the business needs of the enterprise, defining the best solution and specifying the requirements for it using specialized techniques and approaches. At present, several bodies of knowledge have been formed that determine the list of business analysis work and are the basis for the business analysts in the IT industry. This paper deals with the review and comparative analysis of the bodies of knowledge developed by the leading international institutions of the International Institute of Business Analysis (IIBA) and the Project Management Institute (PMI) regarding business analysis activities in IT projects.

KEYWORDS: business analysis, IIBA, PMI, business analysis knowledge area

1. Постановка проблеми

Розвиток та ускладнення проектів по автоматизації і оптимізації бізнес-процесів призвели до формування нової дисципліни «бізнес-аналіз». Вона є розвитком напрямку інженерії вимог і спрямована на виявлення бізнес-потреб підприємства, визначення найкращого рішення та опису вимог до нього за допомогою спеціалізованих технік та підходів. Це підтверджується чисельними дослідженнями в галузі керування проектами. Так згідно з [1] найбільш впливовими чинниками, що призвели до невдач проектів по розробці програмного забезпечення були: зміні в пріоритетах організацій (41%), помилки на етапі збору вимог (39%), зміни цілей проекту (36%), неадекватне бачення або мета проекту (30%) тощо. Схожі показники дають і інші галузеві дослідження [2,3]. Всі вищеперераховані проблеми є результатом неправильно організованого процесу роботи бізнес-аналітиків.

Поширення технік, практик та підходів до виконання бізнес-аналізу в проектній діяльності призвело до необхідності визначення переліку робіт, що виконує бізнес-аналітик. Дана робота присвячена огляду, аналізу та порівняльному аналізу зводів знань, розробленими провідними міжнародними інституціями організаціями «Міжнародний Інститут Бізнес Аналізу (InternationalInstituteofBusinessAnalysis, IIBA)» та «Інститут Управління Проектами (Project Management Institute, PMI)» з точки зору робіт по бізнес-аналізу в ІТ-проектах.

2. Рамки бізнес-аналізу

Визначення рамок та безпосередньо сутність бізнес-аналізу можна провести, виходячи з переліку задач, що відносяться до бізнес-аналітичної діяльності. IIBA в [4] виділяє 6 груп задач, якими опікується бізнес-аналітик та які названі галузями знань:

- Планування та Моніторинг бізнес-аналізу (BusinessAnalysisPlanningandMonitoring (BAPM));
- Виявлення та Співпраця (ElicitationandCollaboration(EC));
- Керування Життєвим Циклом Вимог (RequirementsLifeCycleManagement(RLCM));
- Аналіз Стратегії (StrategyAnalysis(SA));
- Аналіз Вимог та Визначення Дизайну (RequirementsAnalysisandDesignDefinition (RADD));
- Оцінка рішення (SolutionEvaluation(SE)).

Взаємозв'язок між галузями знань наведений на Рисунку 1.

При цьому можна виділити базовий цикл: Аналіз стратегії – Аналіз Вимог та Визначення Дизайну – Оцінка рішення, який відповідає трьом фазам проекту: Передпроектне дослідження – Виконання проекту – Оцінка результатів проекту. Задачі з трьох інших галузей знань виконуються протягом всього проекту. Безумовно, методологія виконання проекту буде впливати на часові рамки та інтенсивність виконання робіт з кожної галузі, але в широкому розумінні запропонована модель виконується для кожної з них.IIBA наголошує на тому, що послідовність виконання бізнес-аналітичних задач в кожному проекті може бути різна.

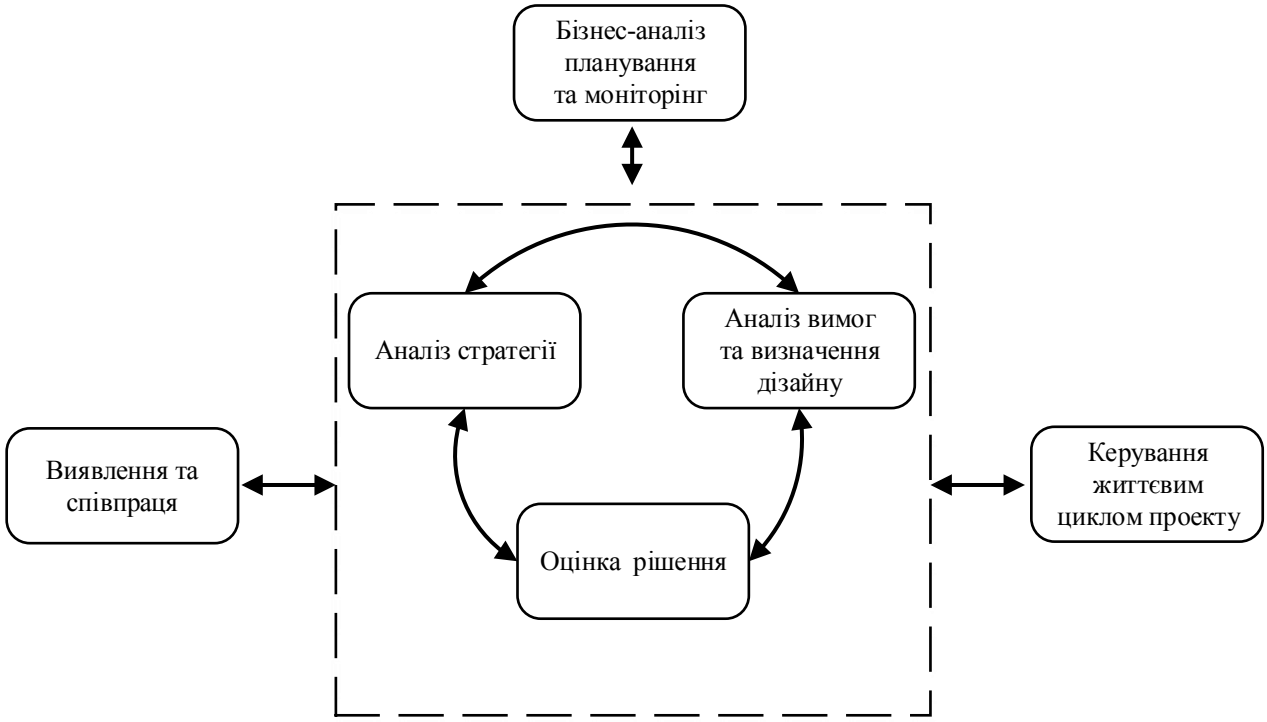


Рис. 1 Галузі знань бізнес-аналізу за IIBA

Але зазвичай, на першому етапі виконуються задачі, що дозволяють визначити поточний стан організації, виявити бізнес-потреби або оцінити ефективність поточного рішення (галузі знань – Аналіз Стратегії та Оцінка Рішення).Крім того, в [4, 5] визначено 5 перспектив бізнес-аналізу:

- Agile;
- Бізнес-аналітика;
- Інформаційні технології;
- Бізнес-архітектура;
- Керування бізнес-процесами.

Перспективи передбачають фокус на певних задачах та методах, специфічних для контексту ініціативи. Вони не є взаємовиключними, більш того, зазвичай один проект передбачає застосування декількох перспектив. Наявність перспективи «Інформаційні технології» підтверджує можливість застосування

запропонованої класифікації бізнес-аналітичних робіт до проектів в галузі IT.

Згідно з [6, 7] діяльність з бізнес-аналізу описується наступними 6 галузями знань:

- Оцінка потреб (NeedsAssessment(NA))
- Залучення зацікавлених сторін (StakeholderEngagement(SE))
- Виявлення (Elicitation(E))
- Аналіз (Analysis(A))
- Трасування та Моніторинг (TraceabilityandMonitoring(TM))
- Оцінка рішення (SolutionEvaluation(SE)).

В цілому їх наповнення відповідає галузям знань, визначених в [4]. Зв'язок між галузями знань наведений на Рисунку 2.

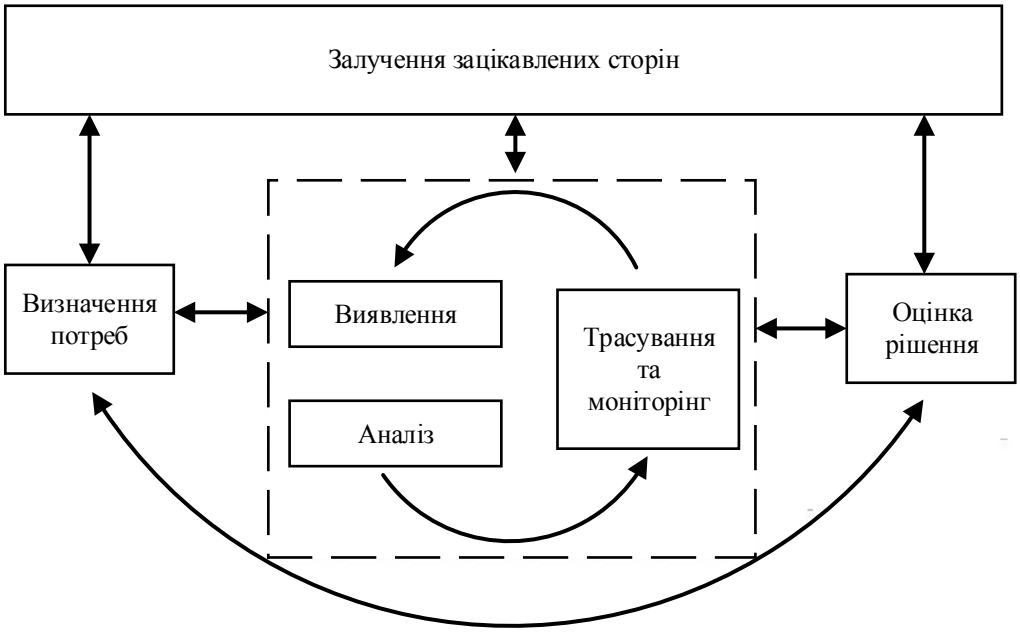


Рис 2. Галузі знань бізнес-аналізу за PMI

Також, PMI, як і IIBA, передбачає модель постійного взаємозв'язку між галузями знань. Ця концепція полягає у постійному процесі аналізу та виявлення вимог, які вдосконалюються завдяки трасуванню і моніторингу, що може призвести до залучення зацікавлених сторін тавизначенню потреб. Цей процес створює додатковий об'єм задач та потребує зусиль від команди розробників та можливо зацікавлених сторін, але допомагає зробити продукт, який буде максимально задовольняти потребам. Порівнюючи галузі знань і виходячи із задач, з яких вони складаються, можна сформулювати відповідності, наведені в Таблиці 1.

Табл. 1. Відповідність галузей знань бізнес-аналізу згідно зводів знань від IIBA та PMI

IIBA	**PMI**
Планування та Моніторинг бізнес-аналізу	Залучення зацікавлених сторін
Виявлення та Співпраця	Виявлення
Керування Життєвим Циклом Вимог	Трасування та Моніторинг
Аналіз Стратегії	Оцінка потреб
Аналіз Вимог та Визначення Дизайну	Аналіз
Оцінка рішення	Оцінка рішення

Висновки

IIBA та PMI розуміють «бізнес-аналіз» як процес із чіткими рамками, цілями та критеріями успішності роботи бізнес-аналітика. Зводи знань з бізнес-аналізу від IIBA та PMI визначають гранці бізнес-аналізу через перелік галузей знань. Перелік галузей знань в обох стандартах співпадає, але IIBAокремо виділяє задачі щодо планування та моніторингу ефективності виконання робіт, тоді як PMIвідносить цю діяльність скоріш до терміну «керування проектом». Наявність перспективи «Інформаційні технології» підтверджує можливість застосування запропонованої класифікації бізнес-аналітичних робіт до проектів в галузі IT. Роботи з бізнес-аналізу можна розбити на 2 групи: кореневий цикл «Аналіз стратегії – Аналіз вимог та визначення дизайну - Оцінка рішення» та супутні «Планування та моніторинг», «Виявлення та співпраця» й «Керування життєвим циклом вимог».

Список літератури

1. Success Rates Rise: Transforming the High Cost of Low Performance // Project Management Institute. – 2017. – P. 29
2. Sanchez O. P. et al. Cost and time project management success factors for information systems development projects //International Journal of Project Management. – 2017. – №. 8. – P. 1608-1626. DOI: 10.1016/j.ijproman.2017.09.007
3. Nelson R. R. IT project management: Infamous failures, classic mistakes, and best practices //MIS Quarterly executive. – 2007. – №. 2.
4. Kevin Brennan A Guide to the Business Analysis Body of Knowledge Version 3.0 // International Institute of Business Analysis – 2015. – P. 514
5. IIBA Global Business Analysis Core Standard A Companion to A Guide to the Business Anlysis Body of Knowledge (BABOK® Guide) Version 3 // International Institute of Business Analysis – 2017. – P. 42
6. Project Management Institute. The PMI Guide to BUSINESS ANALYSIS // Project Management Institute – 2017. – P.444
7. PMI Business Analysis for Practitioners: A Practice Guide //Project Management Institute – 2015. – P. 206

УДК 004.021

КОТЛЯР І. С.
ПОПЕНКО В. Д.

ТЕХНОЛОГІЯ ТОГІВ У РЕАЛЬНОМУ ЧАСІ В ІНТЕРНЕТ-МАРКЕТИНГУ

Великий об’єм даних усіх користувачів Інтернету є досить не структурованим та надто неформальним для аналізу традиційними методами, натомість, технічна можливість обробки та зберігання таких даних виникла відносно нещодавно. Упорядковані дані користувачів стають ціним товаром на ринку інтернет-реклами. Одним з останніх трендів на ринку є технологія алгоритмічної закупівлі і продажу реклами – Real-timebidding (RTB), або аукціон у реальному часі. В доповіді розглянуто особливості технології RTB та її сучасне становище.
Ключові слова.Реклама в Інтернеті, аукціон другої ціни, real-timebidding.

A large amount of data from all Internet users is rather unstructured and too informal for traditional methods of analysis, instead the technical ability to process and store such data has emerged relatively recently. Organized user data is a valuable commodity in the online advertising market. One of the latest trends in the market is the algorithmic buying and selling of advertising – Real-time bidding (RTB), or real-time auction. The report looks at the features of RTB technology and its current status.
Keywords. Online advertising, second-rate auction, real-time bidding.

1.Вступ

Традиційні медіа ресурси змушені проводити спеціальні дослідження, спрямовані на вивчення власної аудиторії. Користувачі Інтернету самі виставляють інформацію про себе у відкритий доступ, але ці дані необхідно збирати, аналізувати та зберігати. Упорядкувавши ці дані, з'являється можливість орієнтуватися на користувачів з певними інтересами, рівнем доходу та освіти, розрізняти користувачів за віковими та статевими групами. Такий підхід дозволяє показувати рекламу певній аудиторії, що значно може покращити результат на відміну від «гарантованих контрактів» з майданчиками, які відвідують і люди, не зацікавлені в пропонованому товарі чи послузі. Рішенням такої проблеми стала поява алгоритмічних закупок рекламних оголошень. Такий підхід називають програматік (programmatic). Це сукупність методів закупівлі рекламних оголошень в Інтернеті з використанням автоматизованих систем і інструментів для прийняття рішень про угоду на основі даних про користувачів. Програматік дозволяє купувати не розміщення на сайті, а контакт з конкретним користувачем, який входить в цільову аудиторію рекламодавця. Найбільшу популярність придбала технологія продажу і покупки реклами на основі аукціону в реальному часі – Real-timeBidding (RTB). Оскільки інші технології не користувалися попитом, зараз єдиною технологією залишається Real-TimeBidding і тому поняття програматік і RTBє майжетотожними.

2. Real-TimeBidding

Real-TimeBidding (RTB) – технологія продажу і покупки показів реклами в Інтернеті на основі автоматизованого аукціону, що дозволяє рекламодавцеві розміщувати рекламу не на конкретному майданчику, а демонструвати її на різних майданчиках тільки для спеціально відібраної по певнихфільтрах (таргетингу) аудиторії. Ціна рекламного контракту в ході аукціону встановлюється в автоматичному режимі, що дозволяє встановити оптимальний баланс між інтересами власників сайтів, які бажають отримати максимальний дохід від показу реклами, і рекламодавців, зацікавлених у максимальному ефекті від рекламної кампанії.RTB використовує автоматизовані системи (роботів) і алгоритми для прийняття рішень про угоду без участі людини на основі соціально-демографічних і поведінкових даних про користувачів, що є в розпорядженні як майданчика, так і рекламодавця. Такий підхід дозволяє поліпшити таргетинг і фактично перейти від закупівлі місць і показів до закупівлі цільової аудиторії.[3]

Передумовою створення технології RTB стала поява перших рекламних мереж (AdNetworks). Вони агрегують рекламний інвентар(місце для розміщення реклами) безлічі інтернет-майданчиків, тим самим спрощуючи роботу рекламодавцю, позбавляючи його від необхідності підписувати договір з кожним окремо. Наступним етапом розвитку рекламних мереж стали рекламні біржі (Adexchanges), на які майданчики виставляли доступний для продажу інвентар. Рішення про покупку реклами приймалося автоматично в режимі реального часу на підставі інформації про моделі поведінки конкретного користувача, часу доби, пристрої, з якого відбувався вихід в мережу, і позиції рекламного оголошення на сайті. Така система відрізнялася від рекламних мереж набагато більшою прозорістю: рекламодавець точно знав, за що і скільки платить. Відповіддю на постійно зростаючу кількість рекламних бірж і обсягів рекламних місць на майданчиках стали системи DSP – платформи попиту. Вони допомагали рекламодавцям вибирати найбільш привабливі пропозиції. Але такі платформи задовольняли потреби тільки з боку рекламодавців. Трохи пізніше виникла аналогічна система – SSP або платформа пропозиції – для майданчиків. Вона дозволила в автоматичному режимі продавати місця під рекламу за максимальною ціною. За допомогою SSP майданчики виходили на безліч рекламодавців. Незважаючи на автоматизацію процесу пошуку і покупки рекламних місць, покупка інвентарю на різних біржах була витратним і неефективним процесом: не було стандартизованого інтерфейсу налаштування рекламної кампанії, для кожної платформи її

доводилося налаштовувати «з нуля»; реклама демонструвалася одному і тому ж цільовому користувачеві кілька разів через різні канали - і плата також стягувалася багаторазово; такий стан справ ускладнював ведення загальної аналітики. Для вирішення цих проблем компанія IAB (InteractiveAdvertisingBureau) розробила протокол OpenRTB. Новий протокол об'єднав не лише біржі, DSP і SSP, але також додав до них систему обробки масивів даних DMP, інструменти аналітики і спеціальні інтерфейси для простого налаштування рекламних кампаній TradingDesk. Технологія була зустрінута позитивно, і OpenRTB отримав подальший розвиток. Таким чином, протокол OpenRTB дозволяє стандартизувати зв'язок між постачальником (SSP) та покупцем (DSP). Метою якого є полегшення інтеграції між сторонами.[7]

3. Послідовність процесу RTB

Торги відбуваються наступним чином:у момент, коли користувач заходить на сайт, підключений до RTB, сайт за допомогою SSP миттєво повідомляє системі про готовність показати рекламу. Також він передає свої технічні характеристики (формат доступного рекламного місця, адреса сторінки і т. п.) і анонімні дані про користувача, які дозволяють віднести користувача до певної соціально-демографічні групи, визначити його інтереси і навіть рівень доходу. Таким чином, рекламодавцю пропонується не покупка рекламного місця, а показ оголошення конкретним користувачам в виділеної цільової аудиторії.

Отримавши ці дані, RTB-система передає їх учасникам аукціону. DSP-системи, проаналізувавши отриману інформацію, визначають претендентів на показ, спираючись на вимоги рекламодавців, зазначені при розміщенні реклами, і дані про користувача. Після того, як ставки зроблені, RTB-система вибирає переможця, який отримує право показати свою рекламу. За показ переможець платить не свою ціну, а ставку найближчого суперника плюс мінімальний крок торгів за принципом аукціону другої ціни (аукціону Вікрі)[2].

Зазвичай OpenRTB розглядається як взаємодія між SSP (Supplysideplatform, продавець (постачальник) рекламних місць) та DSP (DemandSidePlatform, покупець рекламних місць). OpenRTB дозволяє стандартизувати відповіді (bidresponses) DSP на запити (bidrequests) SSP. В результаті роботи SSP отримає декілька відповідей і визначить переможця на основі певних правил аукціону. Система OpenRTB дозволяє повідомляти усім покупцям про те, чи буде показана їх реклама та за якою ціною або сповістити про те, що на даний запит їх реклама не буде показана. [4]

Однак сьогодні екосистемастала набагато складнішою. Власники майданчиків навчилися контролювати процес показу реклами, обираючи найбільш вигідні для себе. Також стало звичним мати більше одного посередника в ланцюжку взаємодії між рекламодавцями і майданчиками.

На рисунку 1 зображено класичний варіант взаємодії учасників RTB (Advertisers, Publishers, DSP, SSP та DMP).

0. Рекламодавець створює на платформі DSP рекламу для відображення.

1. Користувач вводить в рядок браузера назву сайту, підключеного до рекламної RTB мережі. Сайт перед завантаженням сторінки відправляє мережі, до якої він підключений, запит на показ банера, а також дані про користувача.

1.1. Якщо мережа збирає дані про користувача, завантажує їх в DMP.

2. SSP виставляє торг - інформацію про сайт і користувача системам рекламодавців, відправляючи запити на всі партнерів DSP.

2.1. SSP відправляє запит на отримання реклами (BidRequest).

2.1.1. DSP, отримавши запит, обробляють його, та фільтруючи усю рекламу, яку мають. За бажанням, якщо є така можливість, додатково відправляє запит у DMP, для більш точного таргетингу.

2.1.2. Якщо є DMP, то DSP отримує додаткову інформацію про користувача для якіснішої фільтрації.

2.2. SSP отримує відповідь (BidResponse) та обробляє інформацію від всіх рекламодавців, які відповіли на запит.

2.3. У разі відсутності підходящої реклами на показ за протоколом відповідь DSP буде пустою.

3. SSPвибирає найкращий варіант та сповіщає про це DSP (WinNotice).

4. Також окремим запитом SSP може сповістити DSP про успішність відображення реклами та зняття коштів.

5. Завантажується сторінка з отриманим оголошенням.

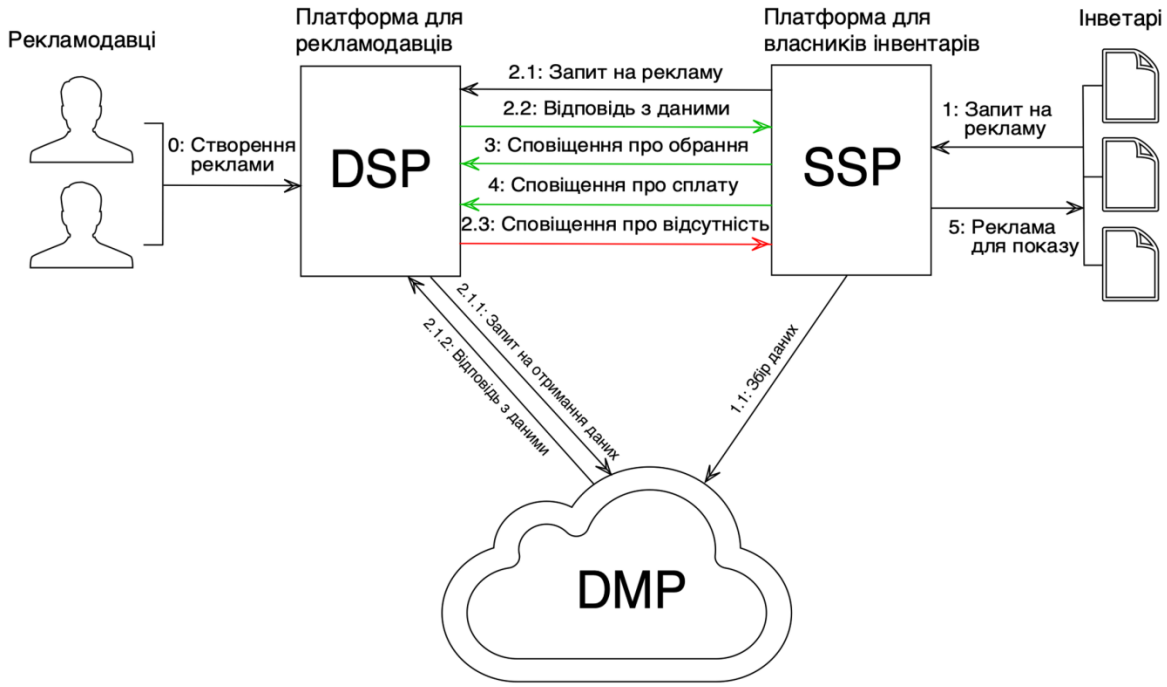


Рис. 1. Схема взаємодії учасників RTB

4. Дослідження математичної моделі аукціону для RTB

В інтернет-рекламі для вибору оголошення, яке буде показуватися на певному місці проводять аукціон. Використовують закритий аукціон двох типів: першої та другої ціни. Інформація про ставки інших учасників невідома і у кожного з них є можливість зробити ставку один раз. Товар (у даному випадку рекламне місце) той, хто запропонував найбільшу ціну, але платить по різному, в залежності від типу аукціону. В аукціоні першої ціни – сплачується найбільша ставка, в аукціоні другої ціни (аукціон Вікрі) – другу найбільшу ціну [2]. Найпопулярнішим серед цих аукціонів в інтернет-рекламі єсаме аукціон другої ціни.[6]

Більшість платформ стягують комісію. Як правило, цю комісію перераховує покупець при здійсненні операції, її розмір обчислюється як певний відсоток від ціни угоди (ціна, зазначена в ставці, збільшена на вартість комісії).

Для будь-якого набору товарів на аукціоні $M = \{t_1, \dots, t_m\}$ та набору учасників $N = \{b_1, \dots, b_n\}$, нехай V_N^M – це «суспільна вигода» (суспільство – усі учасники аукціону) від результатів аукціону при певному наборі ставок. Для учасника b_i та товару t_j ставка учасника буде $v_i(t_j)$.

Учасник b_i, ставка якого для товару t_j, а саме $v_i(t_j)$ є найбільшою серед інших гравців, виграє аукціон, але платить $V_{N/\{b_i\}}^M - V_{N/\{b_i\}}^{M/\{t_j\}}$, що дорівнює неотриманому доходу залишившихся учасників від його виграшу (виграш визначається іншими учасниками).[1, 2]

У випадку класичного аукціону другої ціни, в галузі торгів у реальному часі на платформі DSP, може виникнути ситуація, коли один користувач вказує ціну, яка значно перевищує ціну інших користувачів при необмеженому, на певному проміжку часу, бюджеті, і один користувач буде скуповувати всі рекламні місця, у той час як реклама інших користувачів не буде

показана. Інші користувачі відмовляться брати участь у таких аукціонах, що призведе до втрати користувачів та зменшення прибутку платформи, бо користувач, який залишився, почне викуповувати місця за мінімальною ціною.

Рішенням даної проблеми може бути залежність ймовірності виграшу від ставки. На перший погляд може здатись, що при такому підході платформа отримає менше прибутку, але треба враховувати, що аукціон по найбільшій ціні протримається значно менше, ніж з випадковістю.

Для порівняння стратегії було використано формулу ануїтету:

$$P_{max} = R_{max} * \frac{1-(1+d)^{-n}}{d}$$

де d – процентна ставка для певного періоду, n – кількість періодів, R – періодична виплата.[5]

Також було використано формулу безкінечної геометричної прогресіїдля дисконтування доходів від необмеженого в часі аукціону з випадковими переможцями:

$$P_{rand} = \frac{R_{rand}}{1-r}, r = \frac{1}{1+d}$$

Порівняємо $P_{\text{rand}} > P_{max}$ та отримаємо вираз для n, що знати скільки періодів має протриматись аукціон з найбільшою ціною, щоб бути вигіднішим ніж аукціон з випадковою ціною.

5. Результати дослідження

Вхідні дані:

Ціна B = [1.55, 2.00, 0.35, 0.5, 0.35, 1.55, 1.5]

P = [0.1987, 0.2565, 0.449, 0.064, 0.0449, 0.1987, 0. 1923]

За місяць при такій ціні CPM зроблено приблизно 10 000000 показів.

CPM – ціна за 1000 показів одного оголошення.

Результат:

Математичне очікування місячного доходу DSP від аукціонівдля аукціонів:

$$M(x) = \sum_{i=1}^{n} b_i p_i.$$

$$M(x) = (1.55 * 0.1987 + 2 * 0.2565 + 0.35 * 0.0449 + 0.5 * 0.064 + 0.35 * 0.0449 + 1.55 * 0.1987 + 1.5 * 0.1923) * 10000 = 14808{,}5$$

$$R_{rand} = 14808{,}5 * 25\% \approx 3702$$

$$R_{max} = 2 * 10000 * 25\% = 5000$$

У даному випадку 25% – комісія платформи, яку ми намагаємось максимізувати.Підставимо дані в отриману нерівність (при місячній процентній ставці для дисконтування d= 2%):

$$\frac{\ln\left(\frac{5000}{5000-(1+0.02)*3702}\right)}{\ln(1+0.02)}, = \frac{\ln\left(\frac{5000}{1223.96}\right)}{\ln(1.02)} = \frac{1.4073}{0.0198} = 71 \text{ міс.} \approx 6 \text{ р.}$$

Таким чином, аукціону за найбільшою ціною необхідно протриматись близько 6 років, щоб бути вигіднішим за аукціон з випадковою ціною.

Висновок

Real-Timebiddingє досить популярною технологією, яка стандартизуєпроцес обробки та використання даних користувачів Інтернету, що допомагає рекламодавцям та власникам майданчиків взаємодіяти набагато простіше та ефективніше, ніж за класичною моделлю.

Запропонований підхід обирання реклами з певною ймовірністю допоможе платформі зберегти або навіть підвищити прибуток та зберегти лояльність користувачів, що призведе до стрімкого розвитку платформи та буде значною конкурентною перевагою.

Список літератури

1. Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz: «Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords». American Economic Review 97(1)– 2007– P. 242– 259– DOI 10.1257/aer.97.1.242.

2. А.Савватеев. Аукцион второй цены: доказательство теоремы Викри. Московский физико-технический институт.URL: https://ru.coursera.org/lecture/gametheory/auktsion-vtoroi-tsieny-dokazatiel-stvo-tieoriemy-vikri-PKVzZ.

3. Основи RTB від компанії between exchange. URL: http://rtb-media.ru/wiki-auction/.

4. Основні поняття та опис загальноприйнятої архітектури RTB:URL: https://habr.com/ru/post/169267/.

5. Банковское дело: Учебник для вузов. // Под ред. Г. Белоглазовой, Л. Кроливецкой. – 2-е изд. – СПб.: Питер. – 2010. – с. 240. – 400 с.
6. Vickrey, William. Counterspeculation, Auctions, and Competitive Sealed Tenders// The Journal of Finance– 1961 – Vol. 16, № 1. – P. 8 – 37.– DOI 10.1111/j.1540-6261.1961.tb02789.x.
7. Офіційна документація OpenRTB від IAB. URL: https://github.com/InteractiveAdvertisingBureau/openrtb/blob/master/OpenRTBv3.0FINAL.md.

УДК 004.891.3

БУСОВ І.О.
ГАВРИЛЕНКО О.В.

КОМПЛЕКС ЗАДАЧ З АНАЛІЗУ ФОНОКАРДІОГРАФІЧНОГО СИГНАЛУ

Поява великої кількості оцифрованих медичних даних та розвиток машинного навчання дозволяють пришвидшити та зробити більш достовірним процес діагностики хвороб. У даній роботі розглянута розробка алгоритму автоматизованої сегментації та класифікації сигналів фонокардіограми, для визначення наявності аномалій у роботі серця. Для сегментації сигналів використано конволюційну нейронну мережу, для класифікації – багатошаровий перцептрон, що дозволило отримати високі результати у точності роботи даних алгоритмів.

КЛЮЧОВІ СЛОВА: обробка сигналів, нейронні мережі, серцево-судинні захворювання, фонокардіограма

The emergence of large numbers of digitized medical data and developing of machinele arning make it possible to speed up and make the disease diagnosis process more accurate. In this paper development of analgorithm for automated segmentation and classification of phonocardiograms ignalsis considered to determine the presence of abnormalities in the heart. Convolutional neural network was used for signals egmentation and deepneural network for classification, which allowed to obtain high results in the accuracy of the sealgorithms.

KEYWORDS: signalprocessing, neuralnetworks, cardiovasculardisease, phonocardiogram

1. Вступ

Хвороби серця є найрозповсюдженішою причиною смертності у світі, а їх рання діагностика є найважливішим фактором у подальшому лікуванні хвороби. Фонокардіограма є простим та ефективним способом діагностики хвороб серця, бо порушення тонів серця та шуми можуть бути наслідком відхилень у роботі цього органу.

Поява цифрових стетоскопів та мобільних девайсів дає можливість для високоякісного запису звуків серця. Тому за наявності великої кількості записаних даних постає задача їх аналізу. Автоматична сегментація та класифікація тонів серця може значно пришвидшити аналіз фонокардіограми, визначаючи записи, що мають аномальні значення.

У PhysioNetCardiologyChallenge 2016 була запропонована задача розробити алгоритм класифікації записів звуків серця, зібраних з різних клінічних або не клінічних (таких як домашні відвідування) середовищ. Використовуючи дані, надані організаторами та результати, отримані учасниками, у даній роботі досліджується власний підхід до розв’язку поставленої задачі.

2. Постановка задачі та цілі

Фонокардіограма це крива, що зображує частоту та амплітуду звукових коливань, які виникають у результаті діяльності серця. При аналізі кардіограми виділяють 5 основних тонів серця.

- S1 - звук, який виникає внаслідок закриття мітрального та тристулкового

клапану. Виникає на самому початку систолічного періоду та співпадає з R піком на електрокардіограмі.

- S2 - звук, який виникає внаслідок закриття аортального та пульмонального клапану. Виникає наприкінці систоли та початку діастоли.
- S3 - вказує на збільшення об'єму крові всередині шлуночка. Виникає на початку діастоли після S2 звуку та є ознакою порушень у роботі серця
- S4 – виникає, коли кров потрапляє в жорсткий або гіпертрофічний шлуночок. Ознака патології серця.
- Шуми серця – звуки викликані турбулентністю току крові. У деяких випадках також можуть бути ознакою порушень у роботі серця.

Аналізуючи характеристики та наявність цих звуків, можна робити висновки щодо наявності відхилень у роботі серця.

У даній роботі задачею була розробка алгоритму для сегментації запису тонів серця на періоди(S1, систола, S2, діастола) та класифікація цих записів на основі зробленої сегментації. Розроблений алгоритм має на основі лише фонокардіограми(ФКГ) зробити висновок про наявність аномальних явищах у записі тонів серця, тобто постає задача бінарної класифікації на класи нормальні тони серця/аномальні тони серця.

3. Опис алгоритму класифікації

Для класифікації сигналу фонокардіограми необхідно виявити деякі його характерні ознаки, що вимагає сегментації сигналу. Тобто спочатку потрібно створити алгоритм автоматичної сегментації сигналу ФКГ. Для цього була створена нейронна мережа, яка приймає на вхід сигнал, а повертає масив довжиною вхідного сигналу, елементи якого відповідають періодам серцевого циклу.

Розробка системи для класифікації фонокардіограми було розділено на 3 етапи

1. Створення розміченого датасету сегментованих записів фонокардіограми. На основі даних, що включають у себе записи фонокардіограми та координати R піку та кінця T хвилі на електрокардіограмі, розроблено алгоритм сегментації фонокардіограми.
2. Створення алгоритму, заснованого на методах нейронних мереж, автоматичної сегментації записів фонокардіограми, без використання даних з ЕКГ.
3. Отримання необхідних ознак із запису ФКГ та створення алгоритму класифікатора, який визначає наявність чи відсутність аномалій у відповідному сигналі, базуючись на отриманих ознаках.

5. Сегментація фонокардіограми, базуючись на сигналі електрокардіограми

Для тренування алгоритму автоматичної сегментації сигналу ФКГ був необхідний датасет, де б містилися сигнали з необхідною розміткою. Для створення цього датасету спочатку був створений алгоритм, який стандартними засобами теорії обробки сигналів, базуючись на даних електрокардіограми(ЕКГ), яка була записана паралельно до ФПГ, визначав періоди систоли, діастоли та розташування звуків S1, S2. Був розмічений датасет, який використовувався у роботі [1], що містить у собі 792 записи сигналу ФКГ та координати R піків і кінців T хвиль на паралельно записаній ЕКГ. Частота дискретизації сигналу 2000 Гц.

На вхід подається сигнал $x[n] = \{x_1, x_2, \dots, x_n\}$, що відповідає записаній ФКГ, з частотою дискретизації 2000 Гц, координати R піків $R = \{r_1, r_2, \dots, r_m\}$ та координати кінців T хвиль $T = \{t_1, t_2, \dots, t_m\}$. Необхідно отримати сегментацію сигналу на періоду серцевого циклу $y[n] = \{y_1, y_2, \dots, y_n\}$, де $y_i \in \{0, 1, 2, 3\}$.

$y_i = 0$, якщо x_i належить періоду $S1$

$y_i = 1$, якщо x_i належить до систоли

$y_i = 2$, якщо x_i належить періоду $S2$

$y_i = 3$ якщо x_i належить до діастоли

S1 розпочинається одразу після R піку, а середина S2 знаходиться в околі кінця T хвилі [1], як показано на рисунку 1

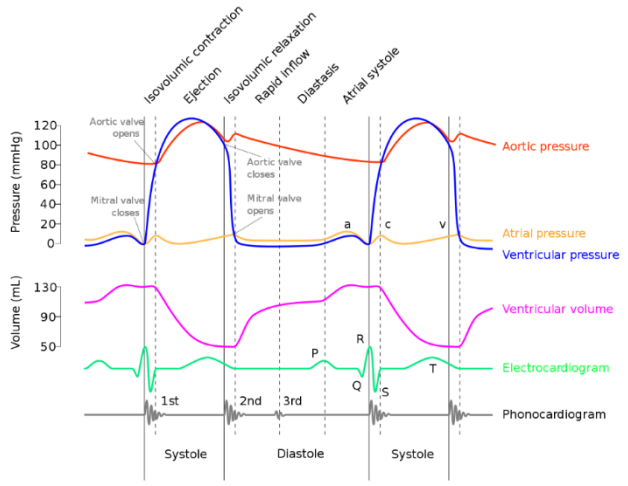


Рис 1. Схема Вітерса, серцевий цикл

Перед визначенням розташування відповідних звуків, сигнал піддається наступній обробці, послідовність якої представлена на рисунку 2

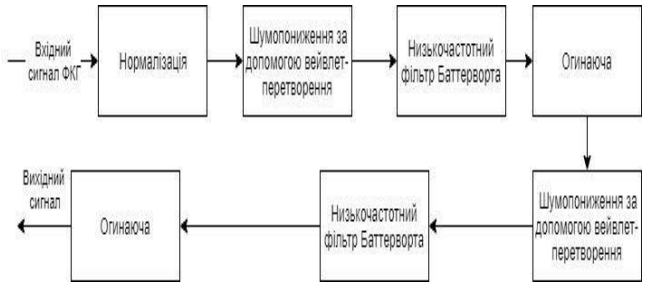


Рис 2. Послідовність обробки сигналу

Позначимо вихідний сигнал як $X[n]$. На вихідному сигналі знаходяться піки $\{X_{p_1}, X_{p_2}, \dots, X_{p_k}\}$, а після цього відбувається сегментація

Визначення розташування S1. Початок j-го S1 визначається як точка j-го R піку. Обирається такий пік вихідного сигналу $X[n]$

Справа від обраного піку обирається перша така точка, значення амплітуди сигналу $X[n]$ у якій менше за деяке значення, що обирається адаптивно для даного піку. Ця точка і буде кінцем S1. Отже інтервал між знайденими крайніми точками позначається як S1.

Визначення розташування S2. В деякому обраному околі точки t_j на сигналі $X[n]$ обирається точка, амплітуда якої є максимальною у цьому околі. Після цього для точок зліва та справа від обраної повторюється процедура аналогічна процедурі пошуку правої границі S1. Таким чином знайдено ліву та праву границі S2, а інтервал між цими границями позначається як S2.

Інтервали між кінцем S1 та початком S2 позначаються як систоли, а між кінцем S2 та початком S1 як діастоли.

6. Створення алгоритму автоматичної сегментації записів фонокардіограми, без використання даних з ЕКГ

Для сегментації сигналу створена нейрона мережа, архітектура якої описана у роботі [2].

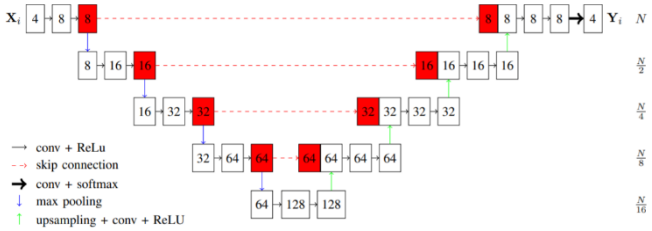


Рис 3. Архітектура конволюційної нейронної мережі, де у прямокутниках вказано кількість фільтрів

Дана конволюційна нейронна мережа заснована на архітектурі Unet, що дає можливість отримувати на виході сегментований сигнал. Такий підхід був обраний через відносну простоту реалізації, відсутність необхідності у складній предобробці сигналу та високу точність.

Спочатку всі сигнали діляться на відрізки довжиною 1 секунда. Останні відрізки кожного з сигналів доповнюються середніми значеннями, якщо вони менші за 1 секунду. Після цього для кожного з отриманих відрізків отримуються гомоморфна обвідна та обвідна Гілберта і вже ці відрізки подаються на вхід до нейронної мережі. На виході нейронна мережа повертає масив довжина якого дорівнює довжині поданого на вхід сигналу і кожен його елемент дорівнює числу яке поставлено у відповідність деякому періоду циклу серця.

7. Створення алгоритму класифікації

Після того як створено алгоритм автоматичної сегментації, з сегментованого сигналу ФКГ можна отримувати більше корисних даних про його особливості. З

кожного сигналу були отримані часові та частотні ознаки.

Часові ознаки. Для наступних показників були взяті значення середнього та середньоквадратичного відхилення: довжина RR інтервалів; довжина S1, довжина S2; довжина систоли; довжина діастоли.

Частотні ознаки. Середнє значення спектру звуків S1 сигналу на частоті f, де f приймає значення $\{10\ Гц, 20\ Гц\ 30\ Гц, \dots, 120\ Гц\}$; середнє значення спектру звуків S2 сигналу на частоті f , де f приймає значення $\{10\ Гц, 20\ Гц\ 30\ Гц, \dots, 120\ Гц\}$

Отже з сигналу отримано 56 ознак, які подаються на вхід до багатошарової нейронної мережі

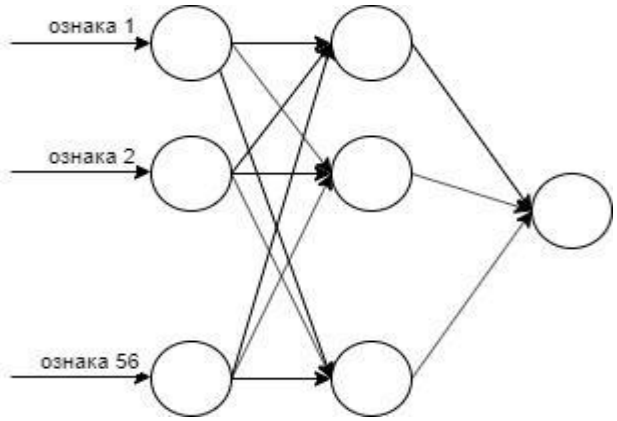


Рис 4. Архітектура нейронної мережі для класифікації

Вихідний нейрон мережі приймає значення ймовірності того, що сигнал ФКГ є аномальним.

Висновки

У даній роботі розроблено алгоритм для автоматичної сегментації та класифікації сигналу ФКГ. Для того щоб отримати більше даних про особливості періодів циклу серця на сигналі спочатку була застосована нейронна мережа, що сегментує сигнал на періоди систоли, діастоли і тонів S1, S2. На основі розміченого сигналу отримувались деякі характеристики роботи серця пацієнта, які обробляються у багатошаровій нейронній мережі, а ця мережа видає на виході ймовірність того, що сигнал ФКГ містить відхилення від норми.

Перелік посилань

[1] David B. Springer, StudentMember, IEEE, LionelTarassenko, SeniorMember, IEEE andGari D. Clifford, SeniorMember, IEEE. LogisticRegression-HSMM-basedHeartSoundSegmentation

[2] Renna, F., Oliveira, J. H., &Coimbra, M. T. (2019). *DeepConvolutionalNeuralNetworksforHeartSoundSegmentation. IEEE JournalofBiomedicalandHealthInformatics, 1–1*. doi:10.1109/jbhi.2019.2894222

[3] CristhianPotes, SamanParvaneh, AsifRahman, BryanConroy. (2016). EnsembleofFeature-basedandDeeplearning-basedClassifiersforDetectionofAbnormalHeartSounds

[4] HongTang, HuamingChen, TingLi, MingjunZhong. (2016). ClassificationofNormal/AbnormalHeartSoundRecordingsbasedonMulti-DomainFeaturesandBackPropagationNeuralNetwork. DOI:10.22489/CinC.2016.171-159

УДК004.75

КУШКА М. О.

СТВОРЕННЯ МЕСЕНДЖЕРУ ДЛЯ ОПЛАТИ КОНСУЛЬТАЦІЙНИХ ПОСЛУГ В КРИПТОВАЛЮТІ

На сьогоднішній день блокчейн-технологія та пов'язані з нею розробки, наприклад, смарт-контракти, набувають усе більшого розповсюдження. Основними перевагами цієї технології є надійність та незмінність. Завдяки цьому програми (смарт-контракти), що працюють на блокчейні завжди виконуються відповідно до свого програмного коду, який неможливо змінити після того, як відповідний застосунок було опубліковано до блокчейну. Більш того, блокчейн дозволяє зменшити сумніви, що могли б виникнути щодо роботи смарт-контракту, оскільки зазвичай його код публічно доступний усім бажаючим, а тому будь-хто може

переконатися, що смарт-контракт не містить жодних прихованих властивостей та йому можна повністю довіряти. У цій статті показано, як можна застосовувати блокчейн у сфері консультаційних послуг, щоб повністю ліквідувати будь-яку недовіру між особою, що надає консультації та їх замовником.

Ключові слова: блокчейн, смарт-контракт, мікротранзакція, Ефіріум, консультаційні послуги

Nowadays blockchain technology and its derivations such as smart contracts became more and more usual in our day-to-day activities. The main advantages of this technology are reliability and immutability. Based on the fact that computer programs working on the blockchain are always executed in the way they were programmed, it is virtually impossible to change somehow program code once deployed to the blockchain. Moreover, the technology allows reducing doubts about how smart-contract executes because usually program's source code is publicly available, thus everybody can check that this smart-contract does not have any undesired features, and therefore can trust it. This paper shows the possibilities of application of blockchain technology to the area of consulting services to eliminate problems with trust between an employer and a consultant.

Keywords: blockchain, smart-contract, microtransaction, Ethereum, consulting services

1. Вступ

Сфера консультаційних послуг онлайн, як і будь-яка інша, має свої проблеми. Проте однією з найбільших серед них є недовіра людей один до одного: завжди є консультанти, які не є експертами у галузі, хоча за таких себе видають, або ж замовники, які прагнуть отримати консультацію, не заплативши за неї. Незважаючи на те, що блокчейн-технологія є новою (на момент написання статті їй лише трохи більше десяти років), вона може бути використана у сфері надання консультаційних послуг онлайн для розв'язання проблем, які в іншому випадку вирішити було б неможливо.

Блокчейн, або, як його ще називають, технологія розподіленого реєстру (DLT) – це набір пов'язаних між собою блоків (записів реєстру), який постійно збільшується. Кожен з таких блоків містить хеш попереднього блоку (таким чином повністю гарантуючи цілісність системи), набір транзакцій (що були додані до блокчейну) і часову відмітку створення блоку. Завдяки цим особливостям DLT являє собою безпечний та "прозорий" реєстр, який піддається перевірці. Ця технологія дозволяє її користувачам не лише записувати, але й розповсюджувати стан будь-якої власної системи розподіленою мережею [**Error! Reference source not found.**]).

Безпека блокчейну базується на факті, що кожну транзакцію перевіряє велика кількість ніяк не пов'язаних між собою користувачів – майнерів. Технологія надає повний доступ до усіх транзакцій у мережі, що були здійснені з самого початку створення блокчейну, та дозволяє кожному бажаючому їх проглядати та перевіряти в будь-який час [**Error! Reference source not found.**].

Першочергово задуманий як основа для криптовалюти, на сьогоднішній день блокчейн має широкий спектр застосування, починаючи з юриспруденції і закінчуючи будівництвом.

2. Принципи, на яких збудовано проект

2.1. Смарт-контракти

В основі проєкту покладено смарт-контракти і мікротранзакції. Смарт-контрактами називають програми (застосунки), що виконуються у блокчейні, який захищає їх від будь-яких атак, спрямованих на зміну програмного коду, а тому такі програми (застосунки) завжди виконуються відповідно до того, як вони були написані. А це, в свою чергу, дозволяє зберігати баланс користувачів у звичайній змінній без сумнівів щодо цілісності та релевантності даних [***Error! Reference source not found.***]. Ця особливість і була

використана у даній розробці. Відповідний смарт-контракт був розроблений і опублікований до ефіріум-блокчейну. Свого часу ефіріум був *створ*ений компанією "Ethereum Foundation". З деталями й особливостями його застосування можна ознайомитись у документації до продукту – "Yellow Paper" [***Error! Reference source not found.***].

2.2. Мікротранзакції (у блокчейні)

Мікротранзакцією *зазвичай* називають транзакцію на відносно невелику суму. Проте у контексті TPP це поняття має дещо ширший зміст. Щоб ґрунтовно його зрозуміти, потрібно спершу мати уявлення про те, як відбувається транзакція.

Кожен користувач мережі має приватний ключ, який є послідовністю випадковим чином згенерованих байтів (наприклад, 256 байтів). Цей ключ користувач зберігає у таємниці від інших. З приватного ключа математичними перетвореннями можна отримати адресу, яка буде ідентифікувати конкретного користувача, та на яку інші користувачі зможуть надсилати криптовалюту. Проте жодними маніпуляціями з адрес*ою* не можна отримати приватний ключ, про що важливо пам'ятати. У момент, коли власник приватного ключа хоче здійснити транзакцію, він "підписує" цим ключем відповідний набір даних, а саме, хто є одержувачем переказу, яка сума платежу тощо. Маючи таку транзакцію будь-хто може перевірити, що вона була дійсно ″підписана″ власником приватного ключа відповідної адреси. Після ″підписання″ транзакція може бути опублікована до блокчейну, де її додають до блоку, після чого вона набуває статусу опублікованої.

З мікротранзакцією все відбувається так само, але без етапу публікації до мережі. Це робиться з метою економії коштів (оскільки для публікації потрібно заплатити комісію майнерам для її додавання) та часу, адже публікація займає певний час (в Ефіріумі приблизно 20 секунд залежно від розміру комісії). Кожна наступна мікротранзакція робиться на суму

$$S_{last} = S_{prev} + S_{curr},$$

де S_{prev} – сума попереднього неопублікованого мікроплатежу (або 0, якщо такий платіж відсутній), а S_{curr} – сума поточного мікроплатежу. Таким чином, можна здійснювати безліч мікротранзакцій практично миттєво без жодної комісії, а коли S_{last} стає достатньо великою з точки зору одержувача коштів (у нашому випадку особи, яка надає консультацію), він публікує останній платіж (на суму S_{last}) до блокчейну, таким чином закріплюючи своє право на володіння коштами у розподіленому реєстрі [***Error! Reference source not found.***].

3. Технічні деталі проєкту

Мною був розроблений мобільний додаток для платформи iOS у вигляді месенджеру. Цей додаток призначений передусім для осіб, які зацікавлені в наданні консультаційних послуг, та їх замовників. *Проте р*озроблений застосунок містить увесь необхідний функціонал, потрібний для того, щоб бути використаним як звичайний месенджер, що використовується для спілкування. Процес відбувається наступним чином. Під час консультації особа, яка отримує послугу, платить за повідомлення від особи, що їх надає. Оплата відбувається автоматично у вигляді мікротранзакцій щоразу, коли повідомлення читається замовником. Консультант у будь-який момент може опублікувати останній платіж до блокчейну, а тому він може не хвилюватися з приводу того, що його праця не буде належним чином оплачена. У свою чергу замовник, у разі невдоволеності якістю отриманих послуг, може видалити чат з таким консультантом, тим самим переставши йому платити. Більш того, система була спроєктована таким чином, що приватний ключ користувача залишається у нього на мобільному пристрої, і розроблена *мною* система його не зберігає. Завдяки цьому користувачі застосунку можуть не довіряти не тільки один одному, а й системі, причому це не завадить їм повністю безпечно нею користуватися.

Висновки

Новизна проєкту полягає у використанні технології блокчейн та смарт-контрактів у сфері консультаційних послуг, де наразі не існує жодних схожих розробок. Основною

перевагою розробленого мною програмного продукту є те, що він дозволяє усунути проблему з довірою в сфері консультацій, а це може стати суттєвим поштовхом до пришвидшення розвитку онлайн-консультування та розширити ринок надання послуг, оскільки консультант більше не буде "прив'язаний" до свого фізичного місцезнаходження, а зможе проводити консультації для будь-якого замовника з будь-якої країни світу.

Перелік посилань

1. Bitcoin Whitepaper / Накомото Сатоші, 2009 – с. 2-5.
2. What are Cryptocurrency Miners? How does Cryptocurrency Mining work? [Електронний ресурс] – Режим доступу до ресурсу: https://www.ethos.io/what-are-miners-cryptocurrency-mining.
3. Розумний контракт [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Розумний_контракт.
4. The "Yellow Paper": Ethereum’s formal specification [Електронний ресурс] – Режим доступу до ресурсу: https://ethereum.github.io/yellowpaper/paper.pdf.
5. How Blockchain Is Improving Micropayments Capabilities [Електронний ресурс] – Режим доступу до ресурсу: https://komodoplatform.com/micropayments/.

УДК004.8

БЛІНКОВ Є.М.

ТЕХНОЛОГІЇ АНАЛІЗУ ТОНАЛЬНОСТІ ТЕКСТІВ

В даній статті виконується порівняльний огляд технологій аналізу тональності текстів, а саме: наївний Баєсів класифікатор та рекурентна нейронна мережа з довгою короткочасною пам’яттю.Також, пояснюється мета проведення такого аналізу, і з яких кроків він повинен складатися.Крім цього, даються рекомендації щодо того, як слід проводити дослідження ефективності обох алгоритмів, щоб виявити з них найкращий

КЛЮЧОВІ СЛОВА: ТОНАЛЬНІСТЬ, НАЇВНИЙ БАЄСІВ КЛАСИФІКАТОР,РЕКУРЕНТНА НЕЙРОННА МЕРЕЖА,АНАЛІЗ,ВЕКТОРНЕ ПРЕДСТАВЛЕННЯ СЛОВА.

This article provides a comparative overview of sentimental analysis of texts technologies, namely: a naive Bayes classifier and a recurrent neural network with long short-term memory. They are quite often used to analyze text tonality. It also explains the purpose of such an analysis and what steps it should take. In addition, recommendations are given on how to perform a research of the performance of both algorithms to identify the best ones.

KEYWORDS: TONALITY, NAIVE BAYES CLASSIFIER, RECURRENT NEURAL NETWORK, ANALYSIS, WORD EMBEDDING.

1. Вступ

На сьогоднішній день існує численна кількість різних компаній, які пропонують людям свої продукти або послуги. Для забезпечення успішного функціонування виробництва потрібно постійно проводити моніторинг ситуації на ринку, щоб задовольнити всі потреби покупців.

Тому, успішне просування товару на ринку потребує багатьох зусиль спеціалістів різного роду котрі, крім проведення аналізу фінансової ситуації, повинні ще й збирати відгуки клієнтів про товар на різних сайтах соціальних мереж або інтернет-магазинах, щоб оцінити ставлення до нього.

Зі зростанням обсягу інформації, яку потрібно проаналізувати та обробити, аналітики зі сфериDataScienceдля встановлення характеру повідомлень почали використовувати аналіз тональності тексту. Ця технологіяможе допомогти визначити які емоції людина передала у своєму текстовому повідомленні.

Проте, досить часто такий аналіз зводиться лише до оцінки позитивного чи негативного відношення людей до якогось

продукту або до обрахунку певних метрик [1].

Таким чином, основна проблема таких методів дослідження попиту на продукцію полягає в тому, що вони не враховують контекст повідомлень користувачів, яке напряму відображає ставлення покупців щодо даного товару або до пов'язаних з ним змін.

З іншого боку, з приходом нових та набагато потужніших алгоритмів здатність штучного інтелекту аналізувати текст значно покращилась. Так, використання нейронних мереж в поєднанні з глибинним навчанням, і різних методів обробки та підготовки даних перетворило технологію аналізу тональності повідомлень в соціальних мережах на зручний та ефективний інструмент, який значно спрощує роботу аналітика.

2. Визначення аналізу тональності тексту

Аналіз тональності - це процес визначення того, чи є фрагмент тексту позитивним, негативним або нейтральним. Технологія аналізу тональності текстівпоєднує методи обробки природної мови (ОПР) і машинного навчання для визначення зважених оцінок настроїв суб'єктам, тем, тем і категоріям в реченні або фразі [2].

Процес аналізу тональності тексту можна сформувати за допомогою наступних кроків:

Крок 1: Збір даних

Перед початком аналізупотрібно сформувати масив даних котрий повинен бути досить великим для кращої ефективності. Зазвичай дані збираються з популярних соціальних мереж, наприклад, з Twitter, які надають розробникам прикладний програмний інтерфейс (ППІ) для доступу до власних ресурсів та інструментарію [3].

Крок 2: Очищення тексту

Засоби очищення тексту дозволять нам підготувати дані до аналізу, видаливши стоп-слова (а, і, або, але, як, що ...), розділові знаки (коми, крапки). Ці інструменти дозволять досліднику відфільтрувати всю інформацію, яка є неважливою для аналізу [3].

Крок 3: Сентимент-аналіз

На цьому етапі використовуються спеціально підібрані алгоритми для аналізу тональності зібраних даних. Як було зазначено раніше, найбільш поширеною класифікацією емоційної складової тексту є спектр між «позитивним» і «негативним». Проте, більш досконалі технології можуть також визначати більш складні почуття, такі як гнів, смуток і так далі. Такі алгоритми використовуютьпідготовлені словники слів, кожне з яких відповідає певній емоції, щоб отримати змогу точно класифікувати те чи інше повідомлення [3].

Крок 4: Оцінка результатів

В результаті сентимент-аналізу ми отримуємо класифікацію тексту, ймовірність якої була визначена обраним алгоритмом як найбільша. Крім цього, в деяких випадках в тексті можуть виділятися окремі фрагменти, щоб показати до яких класів вони належать, а також, щоб зрозуміти яким же чином був класифікований весь текст.

3. Застосування наївного Баєсівого класифікатора

Основною ідеєю наївного Баєсівого класифікатора є обчислення ймовірностей присвоєння тексту якогось класу, використовуючи спільні ймовірності слів і класів.

Сформулюємо теорему Баєса за допомогою співвідношення (1):

$$P(C_k \mid x_1,\ldots,x_n) = \frac{P(C_k)P(x_1,\ldots,x_n \mid C_k)}{P(x_1,\ldots,x_n)} \quad , \tag{1}$$

де C_k - клас тексту, а $(x_1,\ldots,x_n)$ - це його залежний вектор ознак [4].

Згідно до «наївних» припущень щодо умовної незалежності, для даного класу C_k кожна ознака вектору x_i умовно незалежна від будь-якої іншої ознаки x_j при $i \neq j$ [4].

$$P(x_i \mid C_k, x_1,\ldots,x_n) = P(x_i \mid C_k) \tag{2}$$

Таким чином, відношення (2) може бути спрощено до:

$$P(x_i \mid C_k, x_1,\ldots,x_n) = \frac{P(C_k)\prod_{i=1}^{n} P(x_i \mid C_k)}{P(x_1,\ldots,x_n)} \tag{3}$$

Оскільки $P(x_1,\ldots,x_n)$ є константою, то якщо значення змінних ознак відомі, можна використовувати правило класифікації (4) з логарифмічними значеннями ймовірностей для уникнення арифметичного переповнення:

$$\hat{y} = \arg\max_k (\ln P(C_k) + \sum_{i=1}^{n} \ln P(x_i \mid C_k)) \tag{4}$$

Поліноміальний розподіл параметризується вектором $\theta_k = (\theta_1, \ldots, \theta_n)$ для кожного класу C_k ,де n - це кількість ознак, а θ_{ki} - ймовірність $P(x_i \mid C_k)$ ознаки i з'являється у зразку, котрий належить до класу C_k [4].

Параметри θ_k оцінюються відносним підрахунком частотиза допомогою співвідношення (4):

$$\hat{\theta}_{ki} = \frac{N_{ki} + \alpha}{N_k + \alpha n}, \qquad (5)$$

де N_{ki} - це число разів, коли об'єкт i з'являється у прикладі класу k в навчальному наборі T , а N_k - це загальна кількість всіх об'єктів класу C_k [4]. Для запобігання появі нульових ймовірностей, коли якесь слово відсутнє у навчальній вибірці, застосовується згладжування Лапласа, яке у формулі (5) позначається змінною α , яка зазвичай дорівнює 1

Зрештою, формула (4) набуває наступного вигляду:

$$\hat{y} = \arg\max_k (\ln P(C_k) + \sum_{i=1}^{n} \ln \frac{N_{ki} + \alpha}{N_k + \alpha n}) \qquad (5)$$

Таким чином, згідно формули (5) для всіх класів обчислюються їх ймовірності. Після цього, текст класифікується тим класом, який матиме найбільшу ймовірність.

4. Застосування рекурентних нейронних мереж

Рекурентні нейронні мережі - це тип нейронних мереж, де входиприхованих шарів в поточний момент часу залежать від попередніх виходів прихованого шару [5].

Оскільки часова послідовність в РНН зростає, то значення вагових коефіцієнтів можуть безконтрольно збільшуватися або зменшуватися до нуля. Щоб впоратися з проблемою зникаючого градієнтапри навчанні звичайних РНН, було запропонована технологія довгої короткочасної пам'яті (ДКЧП), яка призначалася для вивчення довгострокової залежності на тривалому часовому проміжку. Таким чином, за допомогою ДКЧП до вхідних і вихідних блоків або «вентилів» додається ще й «Забувальний вентиль». Зазвичай, нейронні мережі з такою архітектурою використовуються для прогнозування часових рядів і розпізнавання рукописного тексту [5].

Для обробки природної мови є корисним проаналізуватикількість входження кожного словау документі. Найпростіше це зробити за допомогою унітарного кодування, яке дозволяє представитивходження кожного слова у документі у вигляді бінарного вектору. Серед різних моделей векторних представлень слів найбільш популярними з них є Word2Vec і GloVe [5].

Після представлення кожного слова відповідним йому вектором, навченим згідно моделі Word2Vec, слова $\{T_1, \ldots, T_n\}$ вводяться у ДКЧП один за одним, як показано на рис.1

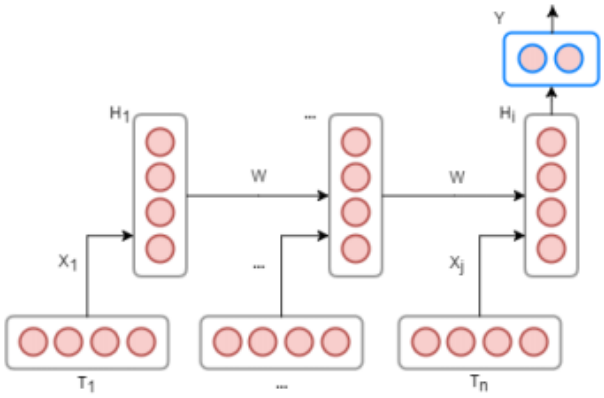


Рис.1.Основна ідея ДКЧП [5]

На Рис. 1 кожна частинка T_i спершу перетворюється у відповідний вхідний вектор x_i , використовуючи модель Word2Vec, а потім входить у ДКЧП один за одним. Далі, в кожен момент часу j вихід W прихованого шару H_i буде переданий назад у прихований шар разом із наступним входом x_{j+1} у наступному моменті часу $j+1$. Зрештою, останній вихід W_n «згодовується» вихідному шару [5].

Щоб діяти за правилами послідовного введення ДКЧП ми спочатку конвертуємо текст у тривимірну матрицю M (X, Y, Z), де X - розмірність моделі вкладення слів Word2Vec, Y - кількість слів у тексті, а Z - кількість текстів. Кількість нейронів у вхідному шарі відповідає розміру моделі Word2Vec, а число нейронів у вихідному шарі дорівнює кількості класів. За допомогоюзворотного поширення у часі відбувається коригування вагового коефіцієнту реберприхованого шару в кожен момент часу. Після завершення кількох

періодів навчаннянейронної мережі ми отримуємоготову модель для класифікації тексту [5].

Після навчання моделі кожен текст з тестової вибірки проходить таку ж саму процедуру, як і елементи навчальної вибірки. Далі, всі процеси, які відображені на Рис. 1 повторюються, але цього разу ваги ребер не оновлюються [5].Потім, отримані результати класифікації текстів можна порівняти із реальними, щоб визначити точність готової моделі.

Висновки

Отже, в даній статті були виконано порівняльний огляд технологійаналізу тональності текстів, а саме: наївний Баєсів класифікатор та рекурентна нейронна мережа з довгою короткочасною пам'яттю. Ці застосування дають змогу визначати класифікацію тексту, якщо перед цим їх натренувати на навчальній вибірці, в якій містяться тексти вже із заздалегідь встановленою класифікацією. Крім цього, незважаючи на те, що дані технології призначені для виконання одного завдання, спосіб його виконання у них абсолютно відрізняється. Таким чином, їх ефективність в більшій мірі залежить від розмірів та вмісту навчальних та тестових вибірок.

Тож, для успішного проведення аналізу тональності тексту потрібно ретельно дослідити роботу кожного з алгоритмів при різних розмірах вхідних даних, щоб застосувати той алгоритм, точність якого є найбільшою.

Список літератури

1. Gupta S. SentimentAnalysis: Concept, AnalysisandApplications [Електронний ресурс] / ShashankGupta. – 2018. – Режим доступу до ресурсу: https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17.
2. SentimentAnalysisExplained [Електронний ресурс] – Режим доступу до ресурсу: https://www.lexalytics.com/technology/sentiment-analysis.
3. Grinvald B. WHAT IS SENTIMENT ANALYSIS? [Електронний ресурс] / BoazGrinvald. – 2019. – Режим доступу до ресурсу: https://www.revuze.it/blog/what-is-sentiment-analysis/.
4. Smetanin S. Sentiment Analysis of Tweets using Multinomial Naive Bayes [Електроннийресурс] / Sergey Smetanin. – 2018. – Режимдоступудоресурсу: https://towardsdatascience.com/sentiment-analysis-of-tweets-using-multinomial-naive-bayes-1009ed24276b.
5. An LSTM Approach to Short Text Sentiment Classification with Word Embeddings [Електроннийресурс] / W.Jenq-Haur, L. Ting-Wei, L. Xiong, W. Long. – 2018. – Режимдоступудоресурсу: https://www.aclweb.org/anthology/O18-1021.pdf

УДК 004.02

АВРАМЕНКО К.А.
ЖДАНОВА О.Г.

ПРО ДВОКРИТЕРІАЛЬНУ ЗАДАЧУ СКЛАДАННЯ РОЗКЛАДУ ВИКОНАННЯ РОБІТ З РІЗНИМИ ДИРЕКТИВНИМИ СТРОКАМИ ІДЕНТИЧНИМИ МАШИНАМИ

Робота присвячена задачі складання розкладу виконання робіт з різними директивними строками ідентичними машинами. За критерії оптимізації взято мінімізацію сумарної тривалості виконання робіт при нульовому запізненні усіх робіт. Розроблено наближений алгоритм, який створено на основі результатів близьких задач. В ході експериментів досліджено умови, за яких розроблений алгоритм будує допустимий розклад (розклад без запізнень), для цього запропоновано показник жорсткості директивних строків.

КЛЮЧОВІ СЛОВА: ДИРЕКТИВНІ СТРОКИ, НУЛЬОВЕ ЗАПІЗНЕННЯ, СУМАРНА ТРИВАЛІСТЬ ВИКОНАННЯ, СКЛАДАННЯ РОЗКЛАДУ.

The work is devoted to the task of compiling a schedule of work with different due dates in identical machines. Minimization of the total duration of tasks execution with zero delay areused as optimization criteria. An approximate algorithm was created based on the results of close problems. During the experiments, the conditions under which the developed algorithm builds an admissible schedule (timetable without delay) were investigated.

KEYWORDS: DUE DATES, ZERO DELAY, TOTAL DURATION, SCHEDULING.

1. Вступ

Найцінніший ресурс – час, його раціональне використання може не тільки зберегти гроші компанії, а й збільшити її конкурентоспроможність в ринкових умовах. Саме тому задачі, пов'язані зі складанням розкладів, впорядкуванням робіт з метою покращення загального часу виконання все частіше знаходять застосування на підприємствах. Приклади таких задач на практиці можна знайти у багатьох сферах: розподіл задач на великих виробництвах, розроблення розкладів прибуття та відправлення для аеропортів, вокзалів і т.д.

Вкрай важливим показником є дотримання встановлених строків виконання. Від'ємне часове зміщення, яке називають запізненням, може принести компанії збитки, або, навіть, втрату замовлень. Врахування часових обмежень зі сторони компанії-виконавця може стати додатковою задачею складання розкладів.

Також, всім відомий показник виробництв – коефіцієнт використання машин. Коли мова йде про великі підприємства мають місця простої, які спричиняють додаткові витрати. Аби зменшити час небажаних затримок в роботі, при розробці розкладу необхідно враховувати сумарну тривалість виконання задач та прагнути до мінімізації цього чинника. Це дозволить навантажувати всі машини рівномірно, покращити коефіцієнт утилізації.

2. Постановка задачі

Дано: m машин (машини або ідентичні, або пропорційні).

Характеристики робіт:

- p_j – тривалість виконання роботи;
- d_j – директивний строк виконання роботи;
- C_{ij} –момент часу, колиробота j завершує своє виконання на машинії.

Характеристики розкладу:

- C_{max} –момент часу, коли завершилось виконання останньої роботи в системі;
- T_j– час запізнення роботи j.

Необхідно скласти такий розклад, у якому:

1) максимальне запізнення дорівнює нулю, тобто усі роботи не запізнюються;

2) сумарний час завершення виконання робіт мінімальний.

Згідно нотації Грехема, маємо задачу:

$$Pm|d_j|\,max\,T_j = 0\,, \sum C_j.$$

3. Результати близьких задач

Задача $1|d_j|\,max\,T_j = 0\,, \sum C_j$

Алгоритм побудови розкладу базується на наступних теоремах.

Правило найбільш ранньої дати завершення (EarliestDueDate– Теорема Джексона). Кожного разу, коли машина звільняється, робота з найбільш раннім директивним строкомвиконується наступною. Це правило мінімізує максимальну затримку серед робіт, які чекають виконання. Насправді, в одній машині з n роботами, доступних в початковий момент часу, правило EDD мінімізує максимальне запізненнях [1][3].

Нехай впорядкування задач відповідно з директивним строком призводить до того, що значення максимуму запізнення є нульовим. Для цієї задачі може існувати іншийрозклад,приякомумаксимумзапізнювання такождорівнюєнулю,але якийзрядупричинкраще впорядкованийвідповідно здирективними строками.Якщопорядзмінімізацієюмаксимальногочасовогозміщенняпрагнутимінімізувати ісереднютривалістьпроходження,товпорядкуванняповинноздійснюватисяу відповідності з наступноютеоремою.

Теорема Сміта. Якщо для системи $n|1$ існує таке впорядкування, що максимальне

запізнювання робіт дорівнює 0, то існує і впорядкування з роботою K в останній позиції, яке зберігає нульове максимальне запізнювання і одночасно мінімізує середню тривалість проходження тоді і тільки тоді, коли:

а) $d_K \geq \sum_{i=1}^{n} p_i$ (робота виконується останньою, якщо це не призводить до її запізнювання),

б) $p_K \geq p_i$для всіх i таких, що

$$d_i \geq \sum_{j=1}^{n} p_j$$

(її тривалість максимальна серед всіх робіт, виконання яких в останню чергу приводить до запізнювання).[2][4]

Алгоритм А1 побудови оптимального розкладу

Крок 1. Розрахувати момент часу, коли завершиться виконання останньої роботи в розкладі

$$\mathrm{C}_{max} = \sum p_j$$

*Крок 2.*Ініціалізувати кількість нерозподілених робіт n як загальна кількість робіт.

Крок 3. Ініціалізувати момент завершення поточної роботи C як C_{max}.

Крок 4. Поки є нерозподілені роботи:

Крок 4.1. Знайти роботи, для яких виконується:

$$d_j \geq \mathrm{C}$$

Крок 4.2. Серед знайдених робіт обрати роботу з найбільшою тривалістю виконанняp_j.

Крок 4.3. Призначити обрану роботу nan-те місце в розкладі.

Крок 4.4. Перерахувати $n = n - 1$.

Крок 4.5. Перерахувати $\mathrm{C} = \mathrm{C} - p_j$.

Крок 5. Розрахувати максимальний час запізнення:

$$maxT_j = \max(T_1, \dots T_n),$$
$$\text{де } T_j = \max(C_j - d_j, 0).$$

Кінець алгоритму.

Задача $Pm||C_{max}$

А тепер розглянемо цю ж задачу для кількості машин $m > 1$, які є ідентичними. При цьому кожна робота може бути виконана будь-якою машиною.

Алгоритм А2 побудови оптимального розкладу

Крок 1. Перенумерувати роботи за неспаданнямтривалостейвиконання робіт.

$$p_1 \leq p_2 \leq \dots \leq p_n.$$

Крок 2. Поки є нерозподілені роботи:

*Крок 2.1.*Для всіх машин по черзі: призначити машині поточну роботу.

Крок 3. Розрахувати C_{max}.

Кінець алгоритму.

Даний розклад можна покращити з огляду на максимальний час завершення робіт, попарно переставляючи роботи одного рівня між машинами.

4. Розробка алгоритмудля задачі$Pm|d_j|\,maxT_j = 0\ ,\sum C_j$

Теоретично мінімальний час, за які б усі машини могли виконати всі завдання [5], становить:

$$C^* = \left\lceil \frac{\sum p}{m} \right\rceil.$$

Необхідна (але не достатня) умова того, що можливо побудувати наближений до оптимального розклад за двома критеріями: усі директивні строки повинні бути більшими за ідеальний теоретичний мінімальний час C^*:

$$\forall\ d_j > \mathrm{C}^*$$

Прирозробці алгоритму були використані ідеї алгоритмів *А1* та *А2*, що наведені вище.

Алгоритм А3

Позначимо k_i як поточну кількість призначених робіт на машину i. Початково $k_i = 0$.

*Крок 1.*Ініціалізувати момент завершення поточної роботи C як C^*.

Крок 2. Поки є нерозподілені роботи:

Крок 2.1. Знайти машину i з найбільшим C:

Крок 2.1.1. Знайти роботи, для яких виконується:

$$d_j > \mathrm{C}$$

Крок 2.1.2. Серед знайдених робіт обрати роботу з найбільшою тривалістю виконання p_j.

Крок 2.1.3. Збільшити k_i для поточної машини на 1.

Крок 2.1.4. Поставити роботу на k_i з кінця місце в розкладі для поточної машини.

Крок 2.2. Перерахувати $C = C - p_j$ для поточної машини.

Крок 3. Розрахувати максимальний час запізнення:

$$maxT_j = \max(T_1, \dots T_n),$$

$$\text{де } T_j = \max(C_j - d_j, 0).$$

Кінець алгоритму.

Проілюструємо роботу розробленого алгоритму на прикладі.

Маємо 5 паралельних машин; 50 робіт, які мають такі характеристики, наведені в таблиці 1.

Табл. 1. Характеристики робіт

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
d_j	135	113	95	213	109	179	45	98	156	130	78	203	47	169	82	89	47
p_j	1	10	2	33	2	15	35	17	30	16	16	7	20	10	5	18	18
j	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**
d_j	105	107	161	121	185	206	149	181	192	163	218	115	50	58	208	231	150
p_j	30	10	24	25	25	10	19	30	36	2	7	21	34	4	18	20	25
j	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	
d_j	125	154	214	52	212	170	177	91	223	100	238	168	223	162	109	151	
p_j	26	23	18	6	30	31	15	27	37	16	22	9	5	13	25	33	

Отримали наступний розклад.
Машина1: 31, 38, 7, 44, 32, 33, 35, 22, 43.
Машина2: 5, 47, 19, 23, 48, 41, 16, 24, 21, 50, 26.
Машина3: 15, 28, 30, 10, 37, 29, 36, 9, 4.
Машина4: 1 3, 46, 14, 17, 6, 8, 18, 20, 25, 39.
Машина5: 27, 12, 2, 13, 11, 42, 49, 34, 40, 45.
Запізнення = 0.

Діаграма Ганта зображена на рис.1.

5. Дослідження алгоритму

Дослідимо умову, за якої розроблений алгоритм не можна використати. Ми вже сказали, що всі директивні строки повинні бути більшими за ідеальний теоретичний мінімальний час C^*:

$$\forall\, d_j > C^*.$$

Проведемо наступний експеримент. Для фіксованої кількості машин (50) згенеруємо роботи з «вільними» директивними строками (за яких існує розклад без запізнення).Далі складемо розклад за описаним алгоритмом А3.

Поступово будемо зменшувати всі директивні строки на 1 та перебудовувати розклад, доки не з'являться запізнення робіт.

Якщо з'явилися запізнення, то знайдемо відношення: $\sum d_j / \sum p_j$. Завершуємо експеримент для даного набору вхідних даних.

Проведемо для кожного набору вхідних даних експеримент 200 разів для різної кількості робіт.

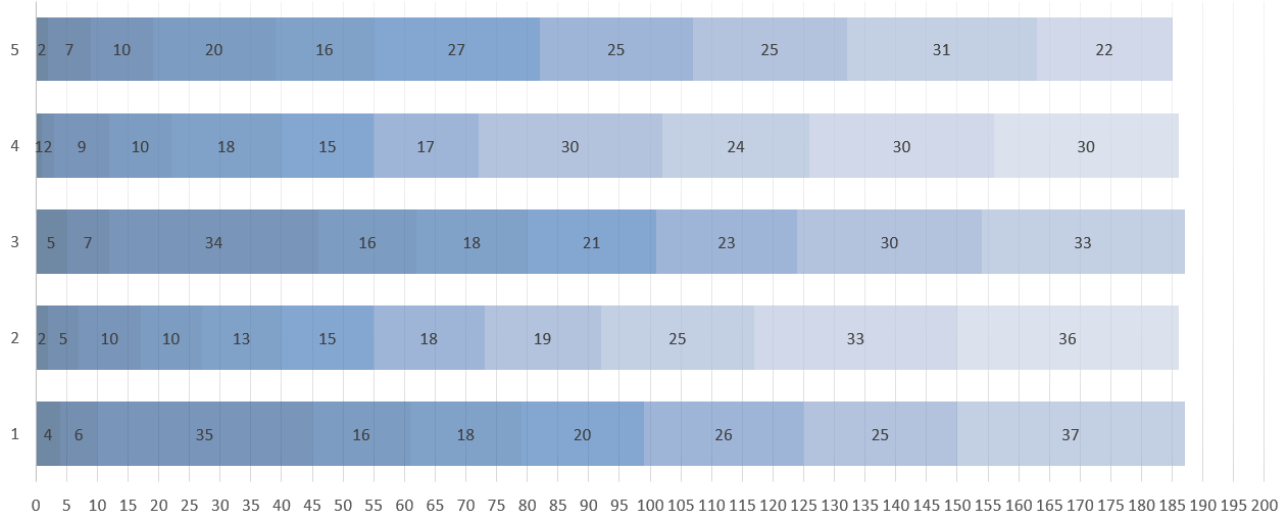


Рис. 1. Діаграма Ганта для отриманого розкладу.

В результаті даного експерименту нас цікавлять мінімальні значення директивних строків, за яких ще немає запізнень, тобто для яких наш алгоритм зможе дати наближенийдо оптимального розклад.

Отримали результати, наведені в таблиці 2. Залежність степені жорсткості директивних строків від кількості робіт графіком зображена на рис2.

Табл. 2. Результати дослідження

Кількість робіт	**Середній min директивний термін**	C^*	C_{max}	**Степінь жорсткості директивних строків** $\sum d_j / \sum p_j$.
200	256,92	257,375	258,245	12,601
250	317,94	318,370	318,940	15,143
300	381,23	381,700	382,200	17,652
350	444,54	444,920	445,260	20,146
400	508,24	508,715	508,980	22,628
450	573,05	573,540	573,785	25,121
500	638,58	638,970	639,195	27,625
550	702,62	703,070	703,225	30,120
600	764,36	764,790	764,855	32,573
650	826,86	827,295	827,350	35,096
700	891,73	892,245	892,265	37,595
750	955,57	955,995	956,030	40,117
800	1018,80	1019,310	1019,315	42,633
850	1085,40	1085,900	1085,910	45,089
900	1145,97	1146,405	1146,410	47,612
950	1211,09	1211,540	1211,555	50,130
1000	1275,68	1276,060	1276,060	52,623

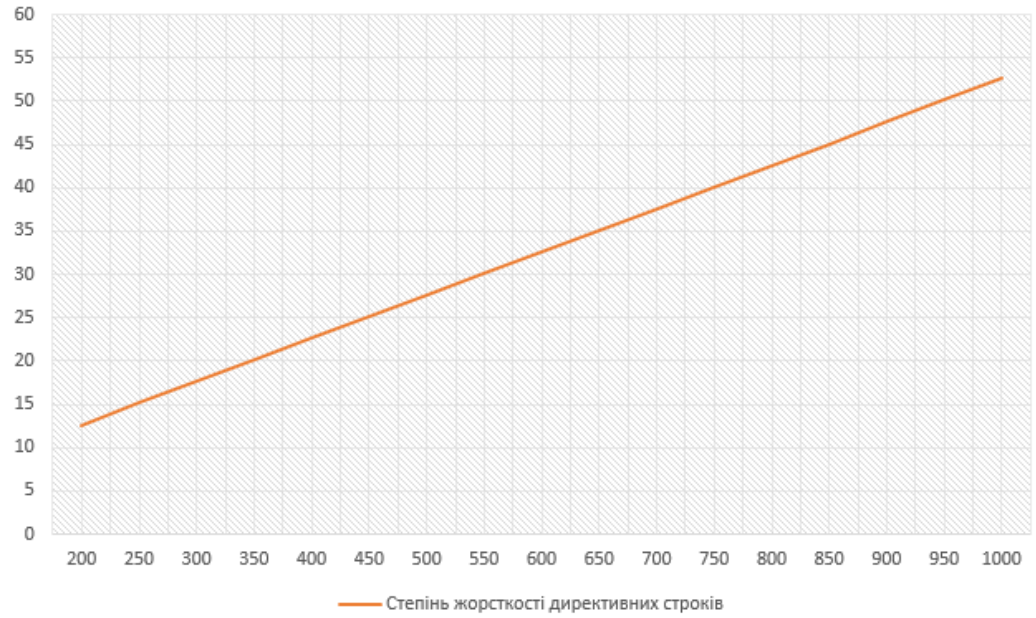


Рис. 2. Графік залежності степені жорсткості директивних термінів від кількості робіт.

Отже, нами запропонована статистично обґрунтована метрика, яка дозволяє з достатньою впевненістю визначити, чи можливо побудувати двокритеріальний розклад для заданоїмножини робіт– степінь жорсткості директивних строків.

В даній статті ми визначили цей показник для кількості робіт 200-1000 та кількості машин 50.

Висновки

В результаті нашого дослідження було розроблено наближений алгоритм побудови розкладу за двома критеріями для ідентичних машин. Також було наведено метрику – ступінь жорсткості директивних строків, яка дозволить оцінити той факт, чи дасть алгоритм допустимийрозклад для заданого набору даних.

Список посилань

1. PinedoMichael L PlaningandScheduling, pringerScienceBusinessMedia, 2009 -532 с.
2. КонвейР.В., Максвелл В.Л., Миллер Л.В. Теориярасписний. – М.: Наука, 1975.
3. J.R. Jackson (1955) “Scheduling a Production Line to Minimize Maximum Tardiness”, Research Report 43, Management Science Research Project, University of California, Los Angeles.
4. W.E. Smith (1956) “Various Optimizers for Single Stage Production”, Naval Research Logistics Quarterly, Vol. 3.
5. Сперкач М.О. Задача визначення максимально пізнього моменту початку виконаннязавданьізспільнимжорсткимдирективнимтерміномпаралельнимипристроямірізноїпродуктивності // Вісник НТУУ “КПІ”. Серія «Інформатика, управління та обчислювальнатехніка». – К.: “ВЕК+”, 2015. – №63 – С.12-18.

УДК 681.3.01

КРАВЦОВ М.В.
КЛИМЕНКО О.М.

КОМЛЕКС ЗАДАЧ ДЛЯ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ПОТРЕБ СПОЖИВАЧІВ

Робота присвячена розробці алгоритмів для збільшення прибутку від продажу товарів. Застосування буде реалізовувати такі функції: пошук суміжних товарів, допомогау створенні реклами, використовуючи цільовий маркетинг. У ході роботи будуть порівнюватися використані алгоритми та їх реалізації.

КЛЮЧОВІ СЛОВА: суміжні товари, цільовий маркетинг, вподобання користувача, персональна реклама.

The work is devoted to the development of algorithms for increasing profits from the sale of goods. The application will implement features such as: search for related products, help create advertising,using targeted marketing. During the work, the algorithms and their implementation will be compared.

KEYWORDS: related products, targeted marketing, user preference,personal advertising.

1. Вступ

У наш час кожна торгова мережа має інтернет магазин, прибуток якого залежить від якісної реклами. В інтернет магазині немає співробітника, який упроцесі живого спілкування із клієнтом може запропонувати супутні товари,продаж яких збільшує прибуток до 10%. Цільовий маркетинг допоможе підлаштуватися під кожного покупця і збільшити обсяг продажів.

2. Machine Learning та маркетинг

Для вирішення задачі пошуку супутніх товарів мивикористовуємокластерний аналіз.

Кластеризація - це метод пошуку закономірностей, який використовується для розбиття сукупностей об’єктів на однорідні групи(кластери). Ціллю кластеризації є отримання нових даних на основі існуючих наборів даних (датасетів).

Застосування алгоритмів Machine Learning дозволить розділити користувачів на групи зі схожими вподобаннями. У роботі будемо порівнювати алгоритми описані у наступних розділах.

3. Хід роботи

Перед застосуванням алгоритмів необхідно виконати підготовку даних. Перш за все треба

сформувати матрицю даних. Для цього чистимо датасети, наприклад,прибираючи інформаційний шум, який не має впливу на поведінку покупця (акційні товари, тара та інше).

Важливо розуміти, що результат кластеризації залежить від проміжку часу,за який були зібрані вхідні дані. Якщо беремо покупки за короткий проміжок часу, то отримаємо кластери з актуальними трендами. Дані за протяжний період дадуть змогу побачити кластери користувачів із стабільною поведінкою(студенти, домогосподарки, пенсіонери та інші).

4. Алгоритм K-means

Це один з найбільш популярних алгоритмів кластеризації. Його мета– мінімізуватисуму середньоквадратичного відхилення кожної точки від її кластера.

$$V = \sum_{i=1}^{k} \sum_{x \in S_i} (x - \mu_i)^2$$

де S_i – і-тий кластер, k- кількість кластерів,μ_i - центр і-го кластеру. На кожній ітерації перераховується центр ваг для кожного кластеру, після чого усі точки на площині діляться на кластери(для кожної точки обирається найближчий центр ваг).

Цей алгоритм має наступні недоліки: необхідно самому задати кількість кластерів, результат роботи алгоритму дуже залежить від вибору початкових центрів ваг, погано працює з об’єктами що належать декільком кластерам або не належать жодному.Останню проблему вирішує модифікована версія алгоритму – c-means. Результатом роботи алгоритму є матриця розміру n*k, де n – кількість всіх користувачів. Кожному користувачу у відповідність ставиться ймовірністьприналежності до певного кластеру.

5. Алгоритм зв’язних компонент

Задачу кластеризації можна також розв’язувати, використовуючи алгоритм зв’язних компонент. У цьому алгоритмі задається параметрR. Якщо відстань між двома точками(усі наші дані ми відобразили у n-вимірному просторі) більша за R, то алгоритм видаляє ребро між ними. Після проходження всього графу з’єднаними залишаються лише точки, які лежать близько один до одної. Головне у цьому алгоритмі підібрати значення R, щоб граф розклався на декілька зв’язних компонент, які в свою чергу і будуть кластерами.

6. Ієрархічні алгоритми

Існують два типи ієрархічних алгоритмів: висхідні і низхідні алгоритми. Низхідні алгоритми працюють за принципом «зверху-вниз».На початку всі об'єкти розміщують в один кластер, який потім розбивається на все більш дрібні кластери. Більш поширені висхідні алгоритми, які на початку роботи розміщують кожен об'єкт в окремий кластер, а потім об'єднують кластери в усе більші, поки всі об'єкти вибірки не будуть міститися в одному кластері.Результати таких алгоритмів зазвичай представляють у вигляді дерева - дендрограмми. Класичний приклад такого дерева - класифікація тварин і рослин.

Висновок

В ході даної роботи будуть застосовані на практиці різні алгоритми кластеризації; порівняні їх складність, точність роботи та масштабування. Нажаль серед цих алгоритмів неможливо вибрати один найкращій і треба проводити багато експериментів, використовуючи різні алгоритми та міняючи їх параметри.

Література

1. Jain A., Murty M., Flynn P.Data Clustering: A Review - ACM Computing Surveys. 1999. Vol. 31, no. 3.

2. Kwok, T., Smith, K., Lozano, S., Taniar, D.Parallel Fuzzy c-Means Clustering for Large Data Sets[Електроннийресурс] – Режимдоступудоресурсу:https://num-meth.srcc.msu.ru/zhurnal/tom_2012/pdf/v13r207.pdf

3. Кластеризация:скажи мнечто ts покупаешь, я скажу тебекто ты [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/company/datawiz/blog/248863/.

4. A tutorial onclusteringalgorithms[Електронний ресурс] – Режим доступу до ресурсу: http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html

УДК 681.3.01

ПРОСКУРКА Д.М.
КЛИМЕНКО О.М.

МОДЕЛІ ПРОГНОЗУВАННЯ НА ФОНДОВОМУРИНКУ

Анотація: Динамічність і складність фондового ринку вимагають відінвесторів знань методів його оцінки, а також навичок в сфері аналізу,проведеного при виборі інвестиційних інструментів дляформування портфелів. У даній статті розглянуті найбільшпоширені моделі прогнозування на фондовому ринку.

Ключові слова: прогнозування, фондовий ринок, регресійнамодель, часовий ряд, технічний аналіз, фундаментальний аналіз.

Summary: The dynamism and complexity of the stock market require investors to know the methods of valuing it, as well as the skills in analyzing the choice of investment tools for portfolio formation. This article examines the most common stock market forecasting models.

Keywords: forecasting, stock market, regression model, time row, technicalanalysis, fundamental analysis.

1. Вступ

Існує постійний масовий інтерес до того, як фондовий ринок веде себе за різних обставин, таких як: рецесії, економічні кризи, війни, санкції та інші події. Ще на початку початку 20 століття тема ринкових настроїв породила новий розрив між тими, хто залишився скептично налаштованим та тими, хто визнав це важливою концепцією майбутнього ринків. З розвитком IT та вивченням Big Data, як інструменту, з’явилося багато більш широких способів дослідити, який саме вплив на прогнозування ринкових цін може бути. Існує велика кількість різноманітних ресурсів даних і те, як вони можуть змінити ставлення до них інвесторів, що беруть участь у ринку. Це призвело до зміщення алгоритмічних парадигм торгівлі, обсяг яких різко зростає з кожним роком. Цей тип торгівлі приваблює тих, хто прагне виграти на покупці, застосовуючи значну кількість різних методик. Однією з найбільш важливихцілей автоматизованої торгівлі є "winamarket", тобто бути максимально точним при прогнозуванні майбутнього прибутку, щоб отримати потенційну винагороду.

2. Моделі прогнозування

Модель прогнозування - формалізований спосіб описудосліджуваного процесу (об'єкта прогнозування), що є основоюдля отримання його майбутніх значень, який спирається навикористання методів прогнозування. Набір методів в кожній моделірізниться, визначає її приналежність до того чи іншого класупрогнозних моделей.

Моделі прогнозування світових цін на фінансових ринкахвідносяться до категорії «надскладних». В їх описі необхідновраховувати взаємозв'язку між великою кількістю чинників. Множинні функції, якими описуються зв'язку та і самі моделі прогнозування в силу відмінностей в розумінні природи об'єктів, що моделюються.Раніше, ще до появи графічних інтерпретацій цін, людипрогнозували динаміку показників фінансового ринку виходячивиключно з фундаментальних факторів, таких як політичніподії, природні явища і т. д. Коли, на протязі деякогочасу, була зібрана певна історична статистика,з'явилася можливість застосування методів графічного і технічногоаналізу. Розглянемо шляхи їх вдосконалення.

3. Метод Саката

Найпершим інструментом для передбачення цін єяпонські свічки і метод Саката. Назва метод отримав від портовогомістечка, де Мунехіса Хонма торгував рисом, вів статистику деннихцін і фіксував динаміку на наступний день. даний методзастосовується до цього дня, як

професійними трейдерами, так іновачками. Також свічкові формації є метою для розпізнаваннянейронною мережею. Багато з свічкових фігур збігаються з формою західнихграфічних моделей. У сукупності вони складають набіргеометричних фігур, який став невід'ємною частиноюіснуючого технічного аналізу.

4. Математичні моделі

Наступним етапом розвитку вивчення цінових діаграм можнавважати спробу звести ринкові флуктуації до математичної моделі.При вивченні створювали велику кількість математичних задач, метоюяких був прорахунок наступних значень числової послідовності,до якої можна звести весь ринок. Відкриті тоді методики заразтакож можна знайти практично в кожному класичному підручнику зтехнічного аналізу: числова мережу Фібоначчі, лінії Ганна і хвиліЕліота. Ці інструменти популярні й донині, за виняткомкоригування під світову фінансову ситуацію.

5. Економетричні моделі

Далі розглянемо неструктурні або, як їх ще називають,економетричні методи, засновані на використанні статистичнихметодів для пошуку статистичних закономірностей, що пояснюютьповедінка змінних моделі, без суттєвої прив'язки до положеньекономічної теорії. З них можна виділити моделі аналізучасових рядів, регресивні моделі і технічний аналіз.

Аналіз часових рядів грунтується на припущенні, щопослідовність значень в ряді даних повторюється черезпевні проміжки часу. Передбачає проведення аналізу наоснові ретроспективних даних.

Через розмаїття методів прогнозування часових рядів,трендових моделей (адитивних, мультиплікативних, змішаних),критеріїв вибору оптимальних показників, виникає необхідністькомбінації прогнозів, які враховують специфіку різних методівпрогнозування. Аналіз часових рядів можна поєднувати зрегресійною моделлю.

Що стосується регресійних моделей, широке поширення вони впрактичних цілях отримали в зв'язку з економічними причинами: принаявності великих сукупностей інформаційних даних дешевше ішвидше буде зробити оцінку об'єктів, що утворюють вибірку і внадалі користуватися цією моделлю для складання прогнозів поданому напрямку, ніж перераховувати кожного разу вихіднідані.

В основі регресійних моделей лежить аналіз, метою якогоє визначення залежності між вихідної змінної ібезліччю зовнішніх факторів, так званих, регресорів. Припобудові моделі вибираються фактори, що впливають на об'єктпрогнозування. Обмежуючим критерієм в даному випадку єте, що додавання в модель більшої кількості чинників невиправдано,коректніше здійснити вибірку з порівняно невеликого числабазових чинників. В процесі тестування визначаються найбільшнадійні і придатні з них. Базові фактори знаходяться вкореляційному зв'язку з обраним функціональним показником. Длярівняння регресії важливо визначити довірчі інтервали,які можливо використовувати в прогнозуванні. У підсумку ми отримуєбагатофакторну регресійну модель для прогнозування.

До побудови моделей даного типу потрібно підходити системно: впроцес повинен бути включений план експерименту, алгоритм виборуструктури моделі, оцінка коефіцієнтів моделі.

Що стосується технічного аналізу, він вивчає, перш за все, саміцінові зміни, заснований на побудові графіків, технічнихіндикаторів, вивченні показників відкритих позицій і обсягу торгівліі передбачає, що в ціні міститься вся наявна на ринкуінформація. Однак при побудові прогнозів виникає ряд проблем.Інституційні фактори не описуються статистично, а значить,неможливо побудувати відповідні кількісні часовіряди.

6. Фундаментальний аналіз

Гіпотеза ефективності ринку стверджує, що вся новаінформація моментально відбивається в ринкових цінах, тому наефективному ринку ніхто не може прогнозувати динаміку ринковихцін, і все ринкові ціни є справедливими для цих активів.

Існує три різновиди ефективного ринку: слабка, середняі сильна фірма ефективності. При слабкій формі, вартість

ринковогоінструменту повністю відображає минулу інформацію, що відноситься донього, перш за все, це динаміка курсової вартості та обсягів торгів.

Середня форма ефективності відображає не тільки минулуінформацію про стан ринкового активу, але і публічну інформації,що знаходиться у вільному доступі (звіти компаній, проведеннязібрань, виплати дивідендів і т. д.)

Сильна форма ефективності крім перерахованих вищечинників відображає також вплив інсайдерської інформації, відомоївузькому колу осіб. Технічні аналітики схиляються до даної форми івважають, що фундаментальний аналіз не допоможе в оцінці майбутньоївартості, так як всі дані, що могли вплинути на неї, вже надаливплив.

Відповідно до теорії випадкових блукань, зміни вартості ціннихпаперів коливаються випадковим чином, тобто ринок практичнонепередбачуваний.

Висновок

В даній статі було розглянуто основні моделі прогнозування на фондовому ринку. Описана проблематика прогнозування та які методи вірішують дані проблеми. Фондовий ринок - ще не розвязана задача і дає можливість досліждувати його та створювати моделі, які покращуватимуть торгівлю на ньому.

Література

1. Stockandbonds [Електронний ресурс] – Режим доступу до ресурсу: https://www.khanacademy.org/economics-finance-domain/core-finance/stock-and-bonds
2. AlgorithmicTrading 101 – Lesson1:TimeSeriesAnalysis [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/the-ocean-trade/the-ocean-x-algo-trading-lesson-1-time-series-analysis-fa3b76f1d4a3
3. Основы фондового рынка [Електронний ресурс] – Режим доступу до ресурсу:https://traders-union.ru/osnovi_fondovogo_rinka/.
4. Машинное обучение: что нужно знать о создании стратегий для торговли на бирже. Часть IV [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/company/iticapital/blog/281603/.
5. Г. МоррисЯпонские свечи. Метод анализа акций и фьючерсов, проверенный временем. – 2012. – 312 с.
6. Эксперимент: создание алгоритма для прогнозирования поведения фондовых индексов [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/en/company/iticapital/blog/278023/

УДК 004.65(004.67)

ФОМІН І. Д.

РОЗРОБКА ІНФОРМАЦІЙНОЙ СИСТЕМИ ДЛЯ КОНТРОЮ СТВОРЕННЯ ТА УПРАВЛІННЯ ПРОЕКТАМИ З ВИКОРИСТАННЯМ RESTAPI

В даній статті буде розгляданий приклад інформаційної системи для контролю створення нових та управління вже існуючих проектів. Ця система надає змогу управлінню будь-яких підприємств слідкувати за продуктивністю співробітників та за виконаним планом.

Інформаційні системи, Управління, RESTAPI, Структури і бази даних, СУБД.

This article will look at an example of an information system to control the creation of new projects and the management of existing ones. This system allows the management of any enterprise to monitor the performance of employees and the plan.

Information Systems, Management, REST APIs, Structures and Databases, DBMS.

1.Вступ

В даній статті буде розгляданий приклад інформаційної системи для контролю створення нових та управління вже існуючих проектів. Організаційні питання стосуються взагалі всіх підприємств, як нових так і старих, і чим більше співробітників працюють над одним проектом або сам проект надто комплексний, тим складніше для управління слідкувати за виконаним планом та складати звіти, бачити хто і на скільки продуктивний. Все це необхідно для контролю створення нових та управління старими проектами.

2. Проектування БД

Для роботи такої інформаційної системи слід розробити базу даних, що буде зберігати учасників системи, а саме: користувачі, проекти, ролі користувачів на кожному проекті, завдання які необхідно на кожному проекті виконати та які користувачі відповідають за які завдання. Звичайно слід додати, що завдання можуть мати таймер часу за який слід їх виконати. Також ролі користувачів на різних проектах можуть бути різні. Завдання можуть мати під-завдання, які мають ті ж самі властивості що і звичайні завдання.

Таким чином в БД існує п’ять таблиць, що не так багато, тому можна скористуватись реляційною БД на мові SQL.Слід виділити, що окрім очевидних зв’язків один-до-багатьох, багато-до-одного, багато-до-багатьох, цій системі необхідний зв’язок багато-до-багатьох-до-багатьох, адже на різних проектах у одного користувача можуть бути різні ролі.

В результаті маємо такі відношення

Task– >User(відношення багато Завдань – один Користувач)

Task–>Task(відношення багато Під задач – одна Задача)

Project<– >Task(відношення багато Проектів – багато Завдань)

Project<– >User<– >Role(відношення багато Проектів – багато Користувачів – багато Ролей)

3.Проектування застосунку

Насамперед слід визначитись як саме люди будуть взаємодіяти з системою, ще і так щоб система мала доступ до спільної БД. Відповідь очевидна – веб-застосунок. З мінусів можливо виділити лиш те, що якщо у користувача не буде інтернету, взаємодіяти з системою він не зможе.Для розробки веб-застосунку, була використана технологія ASP .NET, що дозволяє дуже швидко і зручно підключити БД, та працювати з нею.

Сам застосунок складається з трьох рівнів: DataAccessLayer(DAL), Business Logic Layer (BLL) та Presentation Layer – така архітектура при розробці веб-застосунків зветься трирівнева архітектура. Такий спосіб розподілу обов’язків застосунку дозволяє отримати слабку зв’язність компонентів коду. Наприклад кожен з рівнів має доступ до даних рівня нижче однак не має доступу до рівня вище. Звичайно слід також для слабкою зв’язності реалізувати все через інтерфейси та використовувати DependencyInjection.

3.1. Data Access Layer

Рівень DALскладається з класів сутностей – вони повністю повторюють сутності БД (тобто створюються такі класи: Project, Task, ProjectTask, User, Role, ProjectUserRole), за виключенням того, що зв'язок в коді задається за допомогою посилань, в залежності від типу зв’язку чи на один об’єкт чи на колекцію.

Наприклад клас Usermає властивість-колекцію – Taskstипу Task. В свою чергу клас Taskмає властивість UserIdтипу Integerтаким чином в проекті реалізований зв’язок один до багатьох.

Для зв’язку типу багато для багатьох як і БД створюються сутності-зв’зники, які слугують для зберігання пар ключів сутностей для ідентифікації записів в БД.

Цей рівень також включає в себе класи контексту даних, що репрезентує набори колекцій сутностей.

3.2. Business Logic Layer

BALна сам перед включає в себе DataTransferObjects(DTO) – вони як і сутності DAL повторюють сутності БД, однак кожний такий об’єкт редагується для роботи бізнес логіки, наприклад на сервер прийшов запит на перегляд даних юзеру, для цього буде визваний DTO лише з тими даними, що дозволені до перегляду.

Наступна компонента BAL – сервіси. Сервіси викликають методи в DALі найпростіші сервіси відповідають за додання, зміну чи видалення або перегляду

та передачі користувачу сутностей. Саме сервіси беруть необхідні DTOта виконують бізнес логіку, після чого відправляють на PresentationLayer. Саме на цьому рівні і виконуються всі маніпуляції з даними чи то БД чи то користувача.

3.3. Presentation Layer

Очевидно з назви даного слою, він відповідає за показ даних БД чи оброблених даних BLL та відправку даних користувача на сервер. Слід відмітити, що реалізація цього слою може бути виконана будь яким фронт-енд здатною мовою програмування.

Оскільки в даному прикладі PresentationLayerстворений за допомогою Angular7, основними компонентами будутькомпоненти та модулі – саме вони відповідають за всі елементи фронт-енду.

3.4. Паттерн проектування

Для облегшення роботи з проектом слід дотримуватись патернів проектування, адже вони є самими оптимізованими як для роботи з ними так і загалом.

Для веб застосунку найбільш впевненою кандидатурою є патерн одиниці роботи(Unitofwork). Ідея даного шаблону проектування полягає в створенні транзакцій в роботі з сервером. Тобто кожний користувач при створені запиту на сервер буде створений екземпляр класу UnitOfWork, що містить в собіконтекст БД та додані до DALрепозиторіїв взаємодії з сутностями. В даному прикладі репозиторій був створений для кожної сутності адже не відомо чи знадобляться вони потім. Репозиторії роблять обгортку методів класу контексту даних, щоб створити слабку зв’зність. Працює це таким чином, що в кожний репозиторій ми відправляємо через конструктор екземпляр класу контексту даних і не важливо до якої саме БД він був під’єднаний. Як тільки всі дані будуть взяті з БД, оброблені та відправлені користувачу в екземплярі класу Unit Of Work визивають метод Dispose, що значно швидше звільнює пам’ять серверу.

4.Тестування

Звичайно кожен проект підлягає різним змінам з плином часу, тому проект слід покрити тестами. Тестувати слід сервіси BLL та фронт-енд.

Для тестування сервісів бізнес логіки проекту був використаний nUnit– відкрита середа тестування. Перевіряють на сам перед три основні методи сервісів – додати, змінити, видалити елементи з БД, чи не порушать дані методи цілісності БД.

Слід зауважити, що юніт тестування для веб-застосунків є продуктивним лише при використанні inMemoryDatabase(Встроїної БД у комп’ютер розробника), тобто тести будуть брати дані не з БД а з оперативної пам’ять, що зменшить загрузку БД та звільнить від рисків втрати важливих даних при тестуванні.

Для фронт-енду тестування на сам перед це перевіряти правильність створення запитів на сервер та слідкувати, щоб необхідні компоненти виконували свою роботу правильно.

Для тестування роутів на сервері та правильності відповіді використовується PostMan, що дозволяє формувати та напряму відправляти запити до серверу, імітувати токени та cookie-файли як на localstorageтак і на sessionstorage.

5.Збір даних

Як було написано вище завдання закріпляються за користувачем і кожне таке завдання має свій лічильник часу, однак запускати чи зупиняти його має право лише той учасник за яким це завдання закріплено. Прострочка виставленого для виконання часу привиде до запуску лічильника штрафного часу, який завжди тільки наколюється у учасника. Однак оцінка продуктивності це не тільки накопичений прострочений час, а відношення всього часу роботи до простроченого таким чином чим більше у учасника простроченого часу тим менше у нього рейтинг.

Однак учасник може просто не включати лічильник і виконувати завдання, тому слід прив’язати загальний час роботи в системі контролю створення та управління проектами до системи винагород або заробітної плати.

Висновок

Система контролю створення та управління проектами надзвичайно практична і не складна при проектуванні як і було описано в даній статті, звідси випливає те, що дана система обов’язкова для інтеграцію в будь-який вид підприємськоїдіяльності, адже буде

давати найбільш повну картину щодо продуктивності роботи відповідальних за завдання, та робити висновки як найкраще буде розподілятипрацю та винагороди.

Список літератури

1. Тепляков С.В. “Патерни проектування на платформі .NET” – 2015р.
2. Адам Фримен “ASP.NET Core MVC с примерами на C# для профессионалов. 6-е издание” – 2017р.
3. Json P Smith “Entity Framework Core in Action” – 2018р.

УДК 519.854.2

ІВАНИЦЬКИЙ О.Р.,
ОМЕЛЬЧЕНКО І.О.,
ЖДАНОВА О.Г.

ЗАДАЧА СКЛАДАННЯ РОЗКЛАДУ ВИКОНАННЯ РОБІТ ІДЕНТИЧНИМИ ПАРАЛЕЛЬНИМИМАШИНАМИ З МІНІМІЗАЦІЄЮ СУМАРНОГО ВІДХИЛЕННЯ МОМЕНТІВ ЗАКІНЧЕННЯ ВИКОНАННЯ РОБІТ ВІД ДИРЕКТИВНИХ СТРОКІВ

У роботі розглядається задача складання розкладу виконання робіт паралельними ідентичними машинами при наявності двох множин робіт, кожна з яких має свій спільний директивний строк: у робіт з першої множини - свій директивний строк , у робіт з другої множини - свій.В якості цільової функції обрано мінімізацію сумарного відхилення моментів закінчення виконання робіт від директивних строків. Маючи за основу алгоритм побудови розкладу для системи з одним директивним строком,було розроблено модифікований алгоритм для розв’язання задачі з двома директивними строками.

КАЛЕНДАРНЕ ПЛАНУВАННЯ, СКЛАДАННЯ РОЗКЛАДУ, ДИРЕКТИВНИЙ СТРОК, ПАРАЛЕЛЬНІ МАШИНИ.

The article considers the jobs scheduling problem with identical parallel machines, having two sets of jobs, each having itsown common due date: jobs from the first set have their own due date, jobs from the second set – theirs.The following objective was chosen: minimization of the total deviation of the moments of completion of jobs from due dates. Having the algorithm for building a schedule for systems with only one due date a modified algorithm was developed, which allows us to solve the task with two due dates.

CALENDARSCHEDULING, SCHEDULE BUILDING, DUE DATE, PARALLEL MACHINES.

1. Вступ

Задача, розглянута у роботі, належить до задач теорії розкладів, де основною метою є оптимізація процесу календарного планування. Такого типу задачі широко використовуються на виробництвах для підвищення ефективності побудови плану роботи машин. При умовах, коли є декілька ідентичних машин, або пристроїв, що можуть виконувати роботи, а самі роботи розбиті на дві групи, кожна з яких має свій директивний строк, знайти оптимальний розклад виконання робіт буває важко, або навіть неможливо. Відповідно, потреба в розробці способів, здатних підвищити ефективність календарного планування при заданих умовах, є очевидною. Алгоритм, побудований в даній роботі, пропонує спосіб побудови розкладу з оптимальним, або близьким до оптимального, значенням цільової функції.

2. Постановка задачі

Задано дві множини робіт J_1, що міститьn_1робіт та J_2, яка містить n_2 робіт.Кожна робота має свій час виконання. Є mмашин $(m>1)$. Множина робіт J_1має спільний директивний строкd_1, а множина J_2– директивний строкd_2. Машини працюють паралельно і є взаємозамінними.

Передбачається, що всі роботи множин J_1та J_2надходять одночасно в нульовий

момент часу, процес обслуговування кожної роботи проходить без переривань до завершення. Машини можуть починати свою роботу в ненульовий момент часу.

Необхідно побудувати розклад, що мінімізує сумарне відхилення моментів закінчення виконання робіт від директивних строків.

Дана задача є розширенням задачі, описаній в роботі[1] та частковим випадком задачі з роботи[2]. Відмінністю від [1] є наявність двох директивних строків замість одного, а від [2] – ідентичність машин, що виконують роботи. Окрім того, задачу можна розглядати як розширення інших, подібних за певними умовами. Наприклад, у роботі [3] розглядається задача з мінімізацією суми зваженого відхилення від директивного строку на одній машині, в той час як у роботі [4] автор розглядає лише один аспект задачі – випадок одного, достатньо малого директивного строку.

3. Алгоритм розв'язання задачі з одним директивним строком

Перед розробкою алгоритму для розв'язання задачі варто розглянути простішу постановку – задачу з одним директивним строком. Після успішної реалізації такого алгоритму, його можна розширити на складніші умови. Таким чином буде застосований класичний математичний метод розв'язання складних задач – спрощення до простішого та узагальнення.

Розв'язання задачі з одним директивним строком варто розділити на дві підзадачі в залежності від директивного строку. Якщо він достатньо великий, можливо побудувати ідеальний розклад. На рисунку 1 зображено умовний контур такого розкладу в цьому випадку.

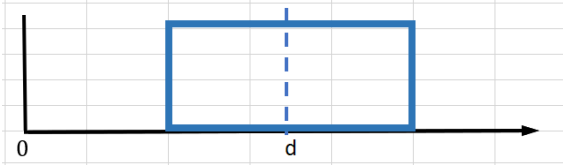


Рис. 1.Умовний контур розкладу у випадку достатньо великого директивного строку

Алгоритм, описаний у роботі [2], гарантує таке розміщення робіт, що досягається мінімально можливе значення цільової функції. Ідея алгоритму полягає в рівномірному розподіленні робіт приблизно однакової тривалості між машинами, а на кожній машині – у рівномірному розташуванні робіт з обох сторін директивного строку. При цьому найдовшим роботам ставляться у відповідність найменші коефіцієнти цільової функції (які залежать від розташування), тобто, щоб при обрахуванні цільової функції роботи входили в неї тим меншу кількість разів, чим вони довші. Таким чином найдовші роботи будуть найбільш віддалені від директивного строку, в той час як коротші роботи будуть скупчуватись навколо нього.

Загальна схема алгоритму побудови розкладу для системи з одним достатньо великим директивним строком (A1):

1. Впорядкувати роботи за незростанням тривалостей;
2. Впорядкувати значення коефіцієнтів ЦФ за зростанням;
3. ЦИКЛ по роботам:
   Знайти для роботи позицію у розкладі, якій відповідає найменший вільний коефіцієнт ЦФ;
   ЯКЩО дана позиція знаходиться перед директивним строком ТА тривалість роботи менша або дорівнює часу, що залишається до директивного строку:
   Призначити роботу на дану позицію;
   ІНАКШЕ:
   Призначити роботу на позицію з найменшим коефіцієнтом цільової функції після директивного строку.

Коли умова достатнього розміру директивного строку не виконується, даний алгоритм уже не дасть оптимального результату. В такому випадку контур розкладу буде виглядати схожим до зображеного на рисунку 2.

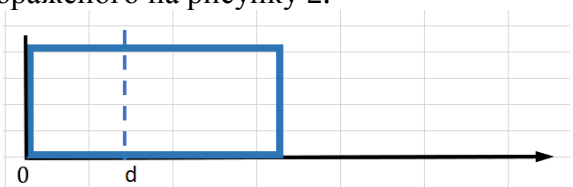


Рис. 2.Умовний контур розкладу увипадку малого директивного строку

В роботі [5] пропонується алгоритм для системи з достатньо малим значенням директивного строку і однією машиною, проте цей алгоритм, за аналогією з попереднім, може бути просто розширений

на систему з декількома машинами. Його ідея полягає в постійній перевірці можливості поставити роботу до директивного строку. Тільки у разі, коли тривалість роботи більша, ніж вільний час, що залишився до директивного строку, робота буде виконуватись після нього.

Загальна схема алгоритму побудови розкладу для системи з одним достатньо малим директивним строком (А2):

1. Впорядкувати роботи за незростанням тривалостей;
2. Впорядкувати значення коефіцієнтів ЦФ за зростанням;
3. ЦИКЛ по роботам:
   ЯКЩО тривалість роботи менша або дорівнює часу, що залишається до директивного строку:
      Призначити роботу на дану позицію;
   ІНАКШЕ:
      Призначити роботу на позицію з найменшим коефіцієнтом цільової функції після директивного строку.

Таким чином, маючи два описані алгоритми, можна будувати розклади для системи з одним директивним строком незалежно від його розташування.

4. Розробка алгоритму складання розкладу для системи з двома директивними строками

Тепер можна розглянути розширену задачу. Її можна розв’язати узагальненням розглянутих вище алгоритмів. При цьому потрібно дивитися не тільки на розташування директивних строків відносно часу надходження робіт (нульового моменту), а й розташування їх відносно одне одного. В такому випадку існує чотири можливі варіанти розташування:

- обидва директивні строки достатньо великі;
- перший директивний строк малий, а другий великий;
- перший директивний строк великий, а другий малий;
- обидва директивні строки малі.

У перших двох випадках можна обійтись описаними вище алгоритмами. При цьому буде побудований оптимальний розклад. У першому випадку, якщо обидва директивні строки достатньо великі відносно нульового моменту та відносно одне одного, як це схематично зображено ня рисунку 3, коли побудовані ідеальні розклади не перетинаються, то для кожної множини робіт застосовується А1.

У випадку, коли перший директивний строк знаходиться дуже близько до початкового моменту, для цієї множини робіт застосовується А2.При цьому, до множини робіт з директивним строком, достатньо сильно віддаленим від першого, застосовується А1. Схематично така ситуація зображена на рисунку 4. Тут побудовані розклади не перетинаються між собою, проте перший директивний строк знаходиться дуже близько до нульового моменту.

У третьому випадкудирективні строки достатньо віддаленівід нульового моменту, проте знаходяться дуже близько один до одного. Така ситуація зображена на рисунку 5. Ідеальні розклади в цьому випадку перетинаються, тому для того, щоб отримати допустимий розклад, деякі роботи потрібно переставити.Для того, щоб побудувати розклад в даній ситуації пропонується наступний алгоритм:

1. Побудувати для кожної з множини робіт розклад за А1;
2. ДОПОКИ сума тривалостей робіт, що знаходяться між директивними строками, більше, ніж проміжок часу між директивними строками:
   Найменшу роботу переставити на інший бік відповідного їй директивного строку.

Таким чином, спочатку будуються «оптимальні»не допустимірозклади, а потім, завдяки мінімальним перестановкам, розклад зводиться до допустимого.

У найскладнішій ситуаціїобидва директивні строки знаходяться близько один до одного та до нульового моменту часу, як це зображено на рисунку 6. Розклади для кожної множини робіт перетинаються між собою, при цьому вони обидва знаходяться занадто близько до нульового моменту, щоб побудувати оптимальний розклад за А1. Ідея алгоритму для цього випадку наступна: спочатку побудувати розклад за А2 для множини робіт з меншим директивним строком, а потім за А2 для множини робіт з більшим директивним строком, взявши за

нульовий момент часу кінець першого розкладу. Таким чином, для першої множини робіт буде побудовано оптимальний розклад, а для другої – з мінімальним відхиленням.

Висновок

В роботі була розглянути задача календарного планування виконання двох множин робіт, кожна з яких має свій спільний директивний строк, в системі з багатьма ідентичними машинами. Оптимізація проводилась у напрямку зменшення сумарного відхилення моментів завершення виконання робіт від відповідних їм директивних строків. Взявши за основу вже відомі алгоритми для побудови розкладів у системі з одним директивним строком, було розроблено модифіковані алгоритми, що дозволяють розширитись на систему з двома директивними строками.

Для перевірки ефективності роботи алгоритмів було проведено серію дослідів, в результаті яких можна зазначити, що при великій кількості робіт алгоритмипоказують непогану якість та прийнятний час виконання. Розроблені алгоритми дозволяють будувати розклади, що, в окремих випадках, дають значення цільової функції, близьке до оптимального. Дані алгоритми є лише одними з багатьох можливих розв'язків задачі, що розглядається. Вони можуть бути покращені та оптимізовані.

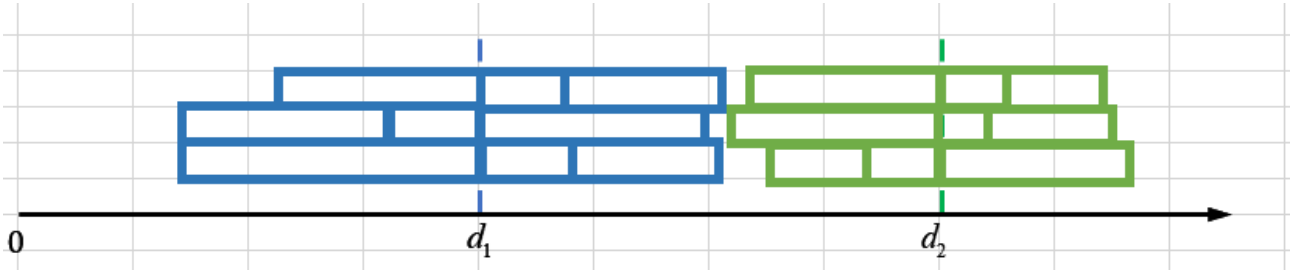


Рис.3.Схематичний приклад побудованого розкладу у системі з директивними строками, достатньо віддаленими один від одного та від нульового моменту

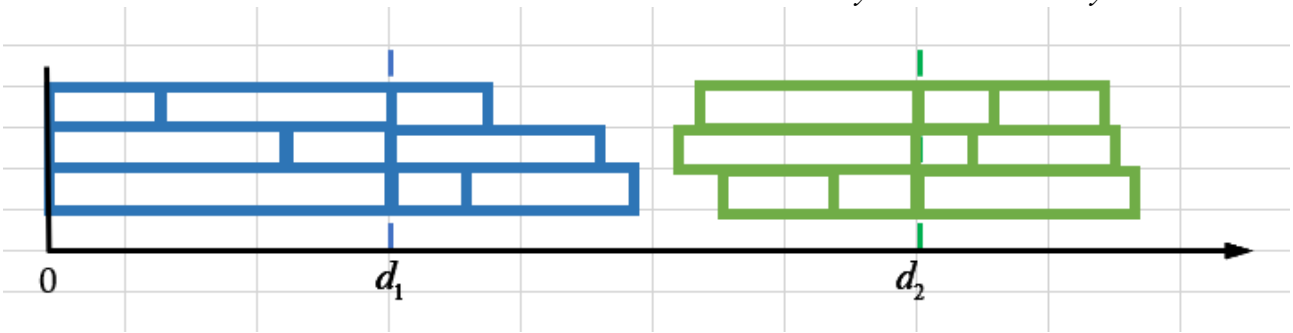


Рис.4.Схематичний приклад побудованого розкладу у системі з директивними строками, менший з яких знаходиться дуже близько до нульового моменту, а більший - достатньо віддалений

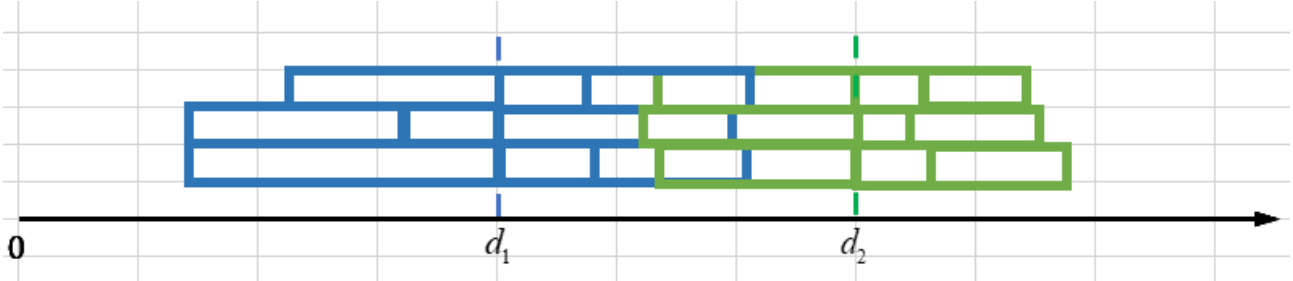


Рис.5.Схематичний приклад побудованого розкладу у системі з директивними строками, які достатньо віддалені від нульового моменту, проте знаходяться дуже близько один до одного

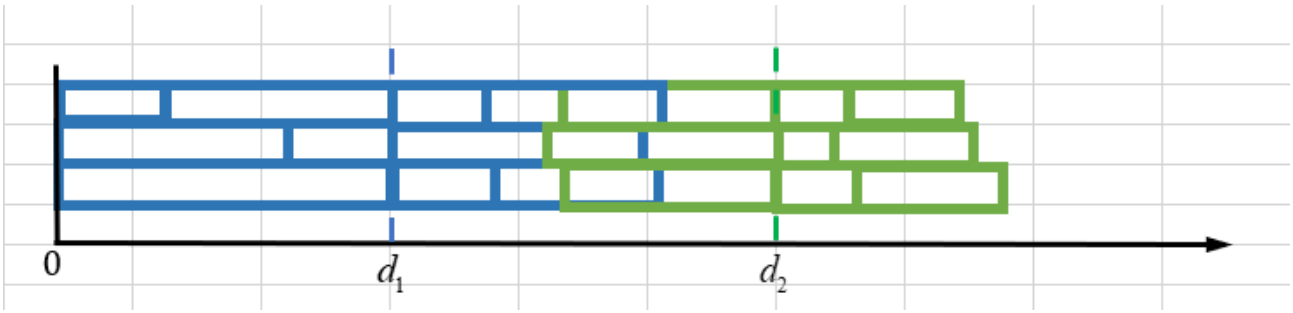


Рис.6.Схематичний приклад побудованого розкладу у системі з директивними строками, які знаходяться дуже близько один до одного та до нульового моменту

Список літератури

1. Година А.В. Задача мінімізації сумарного відхилення від спільного директивного строку при виконанні завдань паралельними пристроями / А. В. Година, О. Г. Жданова, А. О. Маленко, М. О. Сперкач // Науковий огляд. - 2017. -№9(14). - С. 14–32.
2. Година А.В. Задача мінімізації сумарного відхилення моментів завершення від директивних строків при виконанні робіт паралельними пристроями: магістерська дисертація. – Київ: Національний Технічний Університет України «Київський Політехнічний Інститут імені Ігоря Сікорського», 2018. – 116 с.
3. Peter Brucker. Scheduling Algorithms / Peter Brucker. – New York : Springer, 2007. – 378 p.
4. J.A. Hoogeveen, S.L. van de Velde. Scheduling around a small common due date / J.A. Hoogeveen, S.L. van de Velde // European Journal of Operational Research. – 1991. – № 55. – pp. 237-242.
5. Michael L. Pinedo. Scheduling: Theory, Algorithms, and Systems / Michael L. Pinedo. – New York : Springer, 2008. – 662 p.

УДК 004.75

МІРОШНИК О.С.,
ОЛІЙНИК Ю.О.

ЗАДАЧА РОЗПОДІЛЕНОГО МАШИННОГО НАВЧАННЯ

У цій статті розглянуто задачу розподіленого машинного навчання. Сформульовано цілі та актуальність дослідження. Розглянуто обчислювальні аспекти алгоритмів машинного навчання. Описані підходи, які можна використовувати для поставленої задачі, проаналізовано їх переваги і недоліки, виділені вимоги. Описанорозподілене навчання на прикладі одного з алгоритмів машинного навчання.

КЛЮЧОВІ СЛОВА: МАШИННЕ НАВЧАННЯ, РОЗПОДІЛЕНЕ НАВЧАННЯ, ПЕРЕДАЧА ДАНИХ

This article discusses the problem of distributed machine learning. The goals and relevance of the study are formulated. Computational aspects of machine learning algorithms are considered. The approaches that can be used for the task are described, their advantages and disadvantages are analyzed, and requirements are highlighted. Distributed learning is described using one of the machine learning algorithms.

KEYWORDS: MACHINE LEARNING, DISTRIBUTED LEARNING, DATA TRANSMISSION

1. Вступ

Стрімкий розвиток технологій за останнє десятиліттяпризвів до безпрецедентного зростання попиту на штучний інтелект.Це зростання було зумовлене прогресом утехніках машинного навчання та

можливістю використання апаратного прискорення. Задачі машинного навчання все частіше використовуються для аналізу наборів даних там, де традиційні алгоритмічні рішення неможливо застосувати через складність та об'ємність задачі. До прикладів відносять класифікацію та аналіз тексту[1], розпізнавання образів [2], рекомендаційні системи[3] тощо.

Однак, щоб підвищити якість прогнозування та зробити рішення машинного навчання можливими для більш складних застосувань, необхідна значна кількість навчальних даних, оскільки у деяких випадках навчальні дані можуть досягатирозмірув декілька терабайт[4].

Незважаючи на те, що деякі моделі машинного навчання можуть бути навчені наневеликих наборахданих, вхідні данні для навчання моделей, таких як нейронні мережі, зростає експоненційно у залежності від кількості параметрів.

Оскільки попит на обробку та аналіз даних випереджав збільшення обчислювальної потужності техніки, існує потреба в розподілі навантажень навчання серед декількохпристроїв та перетворенні централізованої системинавчання у розподілену. Такі розподілені системи надають ефективну паралелізацію навчального процесу задля створення цілісної моделі.

Поставлена задача: дослідити методи та підходи до розподіленого машинного навчання. Основна задача поділяється на наступні задачі:

1. Дослідження існуючих підходів до розподіленого машинного навчання.
2. Розглянути підхід до розподіленого навчання на прикладі одного з алгоритмів машинного навчання.
3. Формулювання вимог до розподіленого машинного навчання.

2. Обчислювальні аспекти машинного навчання

Незважаючи на велике різноманіття алгоритмів та підходів до машинного навчання, використовувані представлення даних схожі між собою за структурою. Більшу частину обчислень складають операції з векторами, матрицями та тензорами, які у свою чергу є доволі відомими проблемами лінійної алгебри [7][8][9]. Перетворення в лінійній алгебрі зазвичай мають ітеративний характер, що дозволяє розглядати варіант з переходом від однопотокових алгоритмів до паралельних. Такий підхід часто може бути найскладнішим, оскільки він залежить від алгоритму і вимагає чіткого його розуміння.

Як і для інших масштабних обчислювальних завдань, існують два принципово різних тадопоміжні способи прискорення робочих навантажень: додавання більше ресурсів до одного вузла (вертикальне масштабування) та додавання в систему більшої кількості вузлів (горизонтальне масштабування).

Серед рішень щодо вертикального масштабування найпоширенішим методом є додавання програмованих графічних процесорів (GPU), і різні систематичні електронні ортези показали переваги цього (наприклад, [5], [6], [7]). Графічні процесори відрізняються високою кількістю апаратних потоків,що робить їх швидшими для глибинного навчання, ніж звичайний серверний процесор. Коли суттєвий ступінь паралелізму регулюється навантаженням, GPU можуть значно прискорити алгоритми машинного навчання.

3. Розподілений підхід

Розробка узагальненої системи, яка дозволяє ефективно розподіляти машинне навчання, є складною, оскільки кожен алгоритм може різнитися у форматах обміну проміжними результатами і може не мати уніфікованої схеми представлення даних [7][8][9]. Задачу машинного навчання можна розділити на два етапи: навчання та прогнозування.

Етап навчання передбачає підготовку моделі машинного навчання. Задля цього на вхід моделі подається великий набір даних та набір параметрів для обраного алгоритму. У залежності від алгоритму, проміжні результати обчислення можуть поступати знову на вхід до моделі. Кінцевим результатом навчального етапу є навчена (натренована) модель, яку потім можна завантажити у пам'ять та використовувати для прогнозування.

Етаппрогнозуваннявикористовує вже навчену модель для формування прогнозу

(передбачення) на основі нових вхідних даних. На відміну від навчального етапу, етап прогнозування не є таким витратним у обчисленнях та ресурсах оскільки використовує вже натреновані дані та структури. Наприклад, це можуть бути матриці чи вектори.

Етапи навчання та прогнозування не взаємовиключні. Деякі підходи, як-от нейронні мережі, підтримують донавчання, так зване інкрементальне навчання, якеможе поєднуватиобидва етапи та постійно донавчати модель за допомогою нових даних із етапу прогнозування.

Що стосується розподілу, то існує два принциповорізних підходи до розподілу задачта паралелізації на обчислювальних пристроях:паралелізм рівня даних (англ. *Data-Parallelprocessing*) та паралелізм рівня моделі (англ. *Model-Parallelprocessing*) [10][11]. Не виключений випадок комбінування двох підходів.

У підході з паралелізмом даних дані розподіляються між усіма робочими вузлами системи, зазвичай рівномірно. Всі робочі вузли згодом застосовують один і той самий алгоритм до власного набору даних. Модель, яку навчають, доступна для всіх робочих вузлів централізовано або через реплікацію. Такий підхід може бути використаний для більшості алгоритмів машинного навчання.

У підході з паралелізмом моделі точні копії всіх наборів даних обробляються робочими вузлами, які навчають різні моделі, тобто навчають не спільну модель, а незалежну для окремого вузла. При такому підході, спільна модель – це сукупність усіх моделей зібраних з моделей на робочих вузлах.Такий підхід на може застосовуватися абсолютно до всіх алгоритмів машинного навчання, оскільки є випадки коли параметри моделі, як правило, не можуть використовуватися окремо одне від одного.Варіантом вирішення цієї проблеми є підготовка різних екземплярів тієї самої або подібної моделі та об'єднання результатів усіх підготовлених моделей за допомогою таких методів, як збірка (англ. *ensembling*) [12].

4. Алгоритми машинного навчання

У ході навчання моделі алгоритму потрібен зворотній зв’язок (англ. *feedback*), задля того щоб він міг поступово покращувати якість моделі [13]. Існує кілька типів зворотних зв’язків:

1. Навчання з учителем.
2. Навчання без учителя.
3. Напівавтоматичне навчання або часткове навчання.
4. Навчання з підкріпленням.

Навчання з учителем використовує дані навчання, які складаються із вхідних об'єктів (зазвичай векторів) та відповідних бажаних вихідних значень. Алгоритми навчання з вчителем намагаються визначити функцію, яка відображає вхідні дані на потрібний вихідний результат. Потім ця функція може бути застосована до нових вхідних данихзадля прогнозування результату.

Навчання без учителя використовує навчальні дані, що складаються з вхідних об'єктів (як правило, векторів) але без вихідних значень. Такі алгоритми навчання спрямовані на створення функції, якаописує структуру даних та групує неорганізовані вхідні дані. Через те, що дані на вході не розмічені, таким алгоритмам не вистачає чіткої ознаки точності результату.

Напівавтоматичне навчання використовує зазвичай невелику кількість розмічених даних, доповненихпорівняно великою кількістю нерозмічених.

Навчання з підкріпленням використовується для навчання програмногоагента, який повинен вживати дії в певному середовищі на основі своїх спостережень. Зворотній зв'язок покладається на функцію винагороди або витрат, яка оцінюєстани системи.

Незважаючи на різноманітність підходів, обчислення покладаються на обрахунки відповідно до правил лінійної алгебри, що дозволяє нам виконати розподіл навчання між декількома вузлами системи. Розглянемо це на прикладі кластеризації методом k - середніх (англ. *K-MeansClustering*).

Це ітеративний алгоритм машинного навчання без учителя, який розбиваєвхідніданні на k кластери. Відповідно до алгоритму, вибираються випадкові початкові центроїди кластерів і потім кожному вхідному елементу з набору даних присвоюється кластер, центроїд якого до такого елементу є найближчим. Після присвоєння всіх елементів (точок) даних алгоритм визначає нові значення центроїдів шляхом усереднення значень функцій усіх елементів вхідних даних. Алгоритм

повторює ці кроки до завершення заданої кількості ітерацій або до досягнення певних критеріїв конвергенції. Кластеризація таким способом може бути виконана в розподіленому середовищі, використовуючи паралелізм рівня даних, де кожен обчислювальний вузол працює на власній підмножині даних. При такому підході, кожен вузол системи обчислює відстань до всіх поточних центроїдів і визначає найближчий центроїд до кожного локального елемента даних. Далі, для кожного елементу d присвоєного до центроїду c вираховуються суми $sum_c = sum_c + d$та $count_c = count_c + 1$. Після того, як усі вузли системи обробили свої локальні вхідні елементи даних, значення суми та кількості кластерів для всіх вузлів глобальноагрегуються для обчислення нових центроїдних значень.

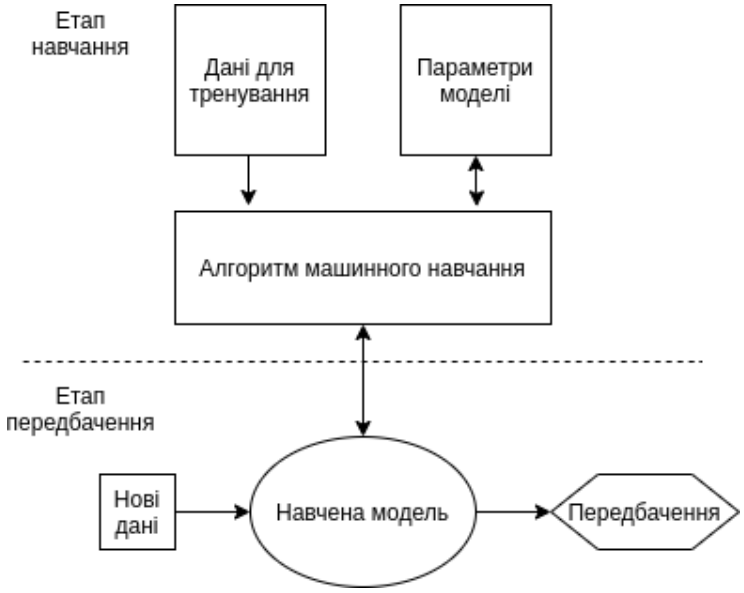


Рис. 3 Загальний огляд машинного навчання. На етапі навчання модель оптимізується за допомогою даних навчання та параметрів моделі. Потім навчена модель розгортається, щоб забезпечити передбачення нових даних, що

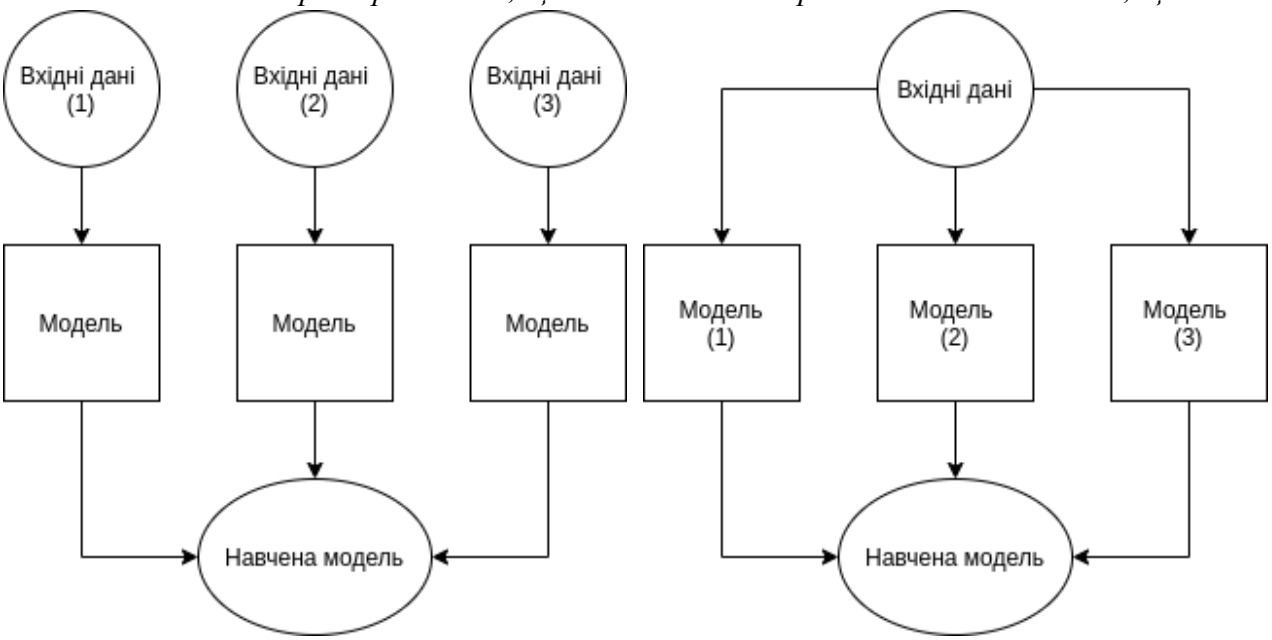


Рис. 2Паралелізм у розподіленому машинному навчанні. Паралелізм рівня даних тренує кілька екземплярів однієї моделі на різних підмножинах навчального набору даних, тоді як паралелізм рівня моделі розподіляє паралельні шляхи єдина модель для декількох вузлів

5. Вимоги до розподіленого навчання

Виконання розподіленого машинного навчання є непростим завданням.Необхідно оброблювати великі набори даних, виконувати їх паралелізацію та розподіл між вузлами системи з урахуванням правильності навчання. Систему ж у свою чергу треба координувати і підтримувати у робочому стані, реагувати на відмови окремих вузлів задля підтримки точності тренувального процесу. Необхідно осмислити загальну архітектуру і підхід до розподілених обчислень та застосувати її для виконання тренування всієї або окремих частин моделі.

Задля ефективної підтримки навчання у системі, що проводить навчання, потрібно мати наступне:

1. Обробку великих наборів даних, підтримку їх цілісності на різних вузлах системи.
2. Відмовостійкість: деякі алгоритми можуть не працювати без проміжних результатів. Відмова окремих вузлів призведе до призупинення навчання, що ж небажаним.
3. Множинні шляхи передачі результатів: використання більшої кількості ресурсів для максимальної пропускної здатності та мінімізації затримок.
4. Якщо це дозволяє алгоритм, то реплікація як вхідних наборів даних, так і моделі на різні вузли задля зменшення часу на обмін даними мережею.

6. Інструменти розподілених обчислень

Проблема обробки великого обсягу даних на кластері з декількома вузлами не обмежується машинним навчанням, а тривалий час вивчалася в розподілених системах та дослідженнях баз даних. Як результат, деякі практичні впровадження використовують основи розподілених систем загального призначення в якості основи для розподіленого машинного навчання.Наразі розроблений цілий ряд систем для підтримки обчислення у розподілених умовах.

Розподілені системи для обробки великої кількості даних значною мірою покладаються на використання декількох дрібних серверів, кожен з яких має відносно невелику ємність зберігання та обчислювальну потужність, а не один дорогий великий сервер. Ця стратегія сталадоступнішою в порівнянні з використанням значно дорожчого спеціалізованого обладнання.

Більша частина існуючих інструментів свої сховища формує з використанням файлової системиGoogle (GFS) або порівнянних реалізаціях [14]. GFS належить і використовується в Google для вирішення всіх потреб пов’язаних зі зберіганнямвеликих об’ємів даних. GFS розбиває дані, які завантажуються в кластер, на шматки, які потім розподіляються між вузлами системиблоками. Частинареплікуються для доступності даних у разі відмов окремих вузлів. Копією GFSз невеликими відмінностями є HDFS, яка входить до інструменту Hadoop.

Незважаючи на те, що архітектура пам’яті зводиться до блокової моделі, існує багато підходів до розподілення задач між вузлами та, власне, проведення обчислення.

MessagePassingInterface (MPI) [22] – інтерфейс взаємодії між вузлами системи за допомогою повідомлень, який призначений для високоефективних розподілених обчислень. MPI пропонує багато примітивів (наприклад, примітиви надсилання, отримання, транслювання та збору повідомлень), що дозволяють користувачам реалізовувати широкий спектр програм, включаючи алгоритми машинного навчання. Однак, завдяки своєму низькому рівню, реалізація багатьох завдань машинного навчання за допомогою MPI часто досить трудомістка і схильна до помилок, оскільки розробники повинні явнокерувати такими аспектами, як розподіл даних та відмово стійкість.

MapReduce [15]. Підхід передбачає наявність декількох фаз і запозичує концепції з функціонального програмування. Спочатку всі дані розділяються на кортежі (так звані ключ-значення пари) під час фази *map*. Це можна порівняти із відображенням функції другого порядку до набору у функціональному програмуванні. Фаза *map* може бути виконана повністю паралельно, оскільки між відображенням функції на два різних значення в наборі немає залежностей

даних. Потім під час фази *shuffle* ці кортежі обмінюються між вузлами і передаються далі. Це вкрай необхідно, оскільки агрегація зазвичай має залежність від даних, і слід забезпечити, щоб усі кортежі, що належать до одного ключа, оброблялися одним і тим же вузлом для правильності. На наступній фазі *reduce*відбувається агрегація кортежів та генерується єдине вихідне значення. Основна перевага такого підходуполягає в тому, що дані можуть бути розподіленими між великою кількістю машин, в той час як завдання кожної з фаз не мають залежності від даних і тому можуть виконуватися повністю паралельно. Ці ж машини можуть бути вузлами в GFS (або подібних), так що замість переміщення даних до коду програми, код програми може бути переміщений до даних для збільшення локальності даних та підвищення ефективності обробки. Найпопулярнішими інструментами, які у своїй основі використовують MapReduceє ApacheHadoop[16]таApacheSpark [17]. У Hadoop всі дії виконуються безпосередньо з файловою системою. У свою чергу, Spark виконує всі дії у оперативній пам'яті, що збільшує швидкість обробки у рази.

ApacheSystemML[18] пропонує декларативну мову високого рівня, яку можна використовувати для реалізації завдань машинного навчання. Ця мова має R-подібний синтаксис і забезпечує безліч вбудованих операторів для виконання матричних операцій. Подані в систему робочі процеси переводяться в завдання MapReduce та оптимізуються, щоб уникнути декількох проходів над вхідними даними.

MLbase[19]– це система, яка дозволяє користувачам виконувати завдання машинного навчання з використанням декларативної мови високого рівня. Оптимізатор системи приймає завдання(наприклад, класифікувати заданий набір даних) і вибирає найкращий алгоритм, параметри та стратегії перевірки. Потім система автоматично виконує завдання в кластері і повертає отриману модель та резюме користувачеві.

7. Проблеми розподіленого машинного навчання

Існує ще ряд проблем, які мають велике значення для довгострокового успіху розподіленого машинного навчання.

Ефективність обчислень. Наразі, більша частина обчислень виконується за допомогою хмарних рішень, де вартість виконання залежить не від завантаженості вузлів системи, а від часу їх роботи [20]. Постає завдання виконання більшої кількості обчислень за той самий проміжок часу або зменшення часу виконання обчислень. Використання апаратного прискорення може пришвидшити виконання всього обчислення.

Відмовостійкість. Не всі системи гарантують продовження навчання у разі відмови одного або декількох вузлів [21]. Можна також зменшитиймовірність виходу з ладу для кожного окремого вузла, але для цього потрібне дуже специфічне обладнання, яке є дорогим і зазвичай не доступний в товарних центрах обробки масштабів або в хмарі. Перехід до асинхронного виконання не є рішенням проблеми оскільки не є універсальним для тренування моделей, яким послідовність виконання важлива.

Портативність. Наразі не існує одного загального формату даних для розповсюдження навчених моделей [6][7]. Таким чином різні системи не можуть донавчати моделі або виконувати роботу з існуючими результатами, що призводить до повторного перенавчання.

Висновки

Розподілене машинне навчання – екосистема з різноманітними рішеннями, які відрізняються архітектурою, алгоритмами, продуктивністю та ефективністю. Доводиться долати деякі фундаментальні проблеми, щоб зробити машинне навчання в першу чергу життєздатним, наприклад, знайти механізми ефективного управління. паралельно обробляти дані, поєднуючи результати, в єдину цілісну модель. Тепер, коли доступні галузеві системи та з огляду на постійно зростаючіапетит до вирішення більш складних проблем з машинним навчанням, розподілене машинне навчання все більше стає нормою, а однопотокові рішення скоріш виняток, подібно до того, як розвивалася обробка даних загалом за останнє

десятиліття. Однак існує ще багато відкритих викликів, які мають вирішальне значення для довгострокового успіху розподіленого машинного навчання.

Список використаних джерел

1. Канівець, Д. В. Математичне та програмне забезпечення класифікації наукових текстів : магістерська дис. : 121 Інженерія програмного забезпечення / Канівець Дмитро Володимирович. - Київ, 2019. - 88 с./
2. Сарнацький, В. В. Математичне та програмне забезпечення для підтримки діагностики онкологічних захворювань за допомогою гібридних методів обробки зображень : магістерська дис. : 121 Інженерія програмного забезпечення / Сарнацький Владислав Віталійович. – Київ, 2019. – 121 с.
3. Koren Y. MatrixFactorizationTechniquesforRecommenderSystems / Y. Koren, R. Bell, C. Volinsky. // Computer. – 2009. – №42. – С. 30–37.
4. Sibyl: A system for large scale supervised machine learning / [K. Canini, T. Chandra, E. Ie та ін.]. // Technical Talk. – 2012. – №1. – С. 113.
5. Raina R. Large-scaledeepunsupervisedlearningusinggraphicsprocessors. InProceedingsofthe 26 thannualinternationalconferenceonmachinelearning / R. Raina, A. Madhavan, A. Y Ng. // ACM. – 2009. – С. 873–880.
6. Theano: A CPU and GPU mathcompilerinPython / [J. Bergstra, O. Breuleux, F. Bastien та ін.]. // In Proc.9th PythoninScienceConf. – 2010. – №1.
7. Caffe: Convolutionalarchitectureforfastfeatureembedding / [Y. Jia, E. Shelhamer, J. Donahue та ін.]. // InProceedingsofthe 22nd ACMinternationalconferenceonMultimedia. – 2014. – С. 675–678.
8. Distributedalgorithmsfortopicmodels / [D. Newman, A. Asuncion, P. Smyth та ін.]. // JournalofMachineLearningResearch. – 2009. – №10. – С. 1801–1828.
9. Richtárik P. Distributedcoordinatedescentmethodforlearningwithbigdata / P. Richtárik, M. Takáč. // TheJournalofMachineLearningResearch 17. – 2016. – №1. – С. 2657–2681.
10. Peteiro-Barral D. A surveyofmethodsfordistributedmachinelearning / D. Peteiro-Barral, B. Guijarro-Berdiñas. // ProgressinArtificialIntelligence 2. – 2013. – №1. – С. 1–11.
11. Strategiesandprinciplesofdistributedmachinelearningonbigdata / [E. Xing, Q. Ho, P. Xie та ін.]. // Engineering 2. – 2016. – №2. – С. 179–195.
12. Ji G. Ensemblelearningbaseddistributedclustering. InPacific-AsiaConferenceonKnowledgeDiscoveryandDataMining / G. Ji, X. Ling. //Springer. – 2007. – С. 312–321.
13. LearningtoComposeWordsintoSentenceswithReinforcementLearning [Електронний ресурс] / [D. Yogatama, P. Yogatama, C. Dyer та ін.]. – 2016. – Режим доступу до ресурсу: http://arxiv.org/abs/1611.09100.
14. Barroso L. Websearchfor a planet: TheGoogleclusterarchitecture / L. Barroso, J. Dean, U. Hölzle. // IEEE micro. – 2003. – №2. – С. 22–28..
15. Dean J. MapReduce: SimplifiedDataProcessingonLargeClusters. InProceedingsofthe 6th ConferenceonSymposiumonOpeartingSystemsDesign&Implementation / J. Dean, S. Ghemawat. // OSDI. – 2004. – №6. – С. 10–10.
16. BigdataanalysisusingApacheHadoop / [J. Nandimath, E. Banerjee, A. Patil та ін.]. // In 2013 IEEE 14th InternationalConferenceonInformationReuse&Integration (IRI). IEEE. – 2013. – С. 700–703.
17. Shanahan J. Largescaledistributeddatascienceusingapachespark. / J. Shanahan, L. Dai. // InProceedingsofthe 21th ACM SIGKDD internationalconferenceonknowledgediscoveryanddatamining. ACM. – 2015. – С. 2323–2324.
18. SystemML [Електронний ресурс] – Режим доступу до ресурсу: https://systemml.apache.org/documentation.
19. MLbase [Електронний ресурс] – Режим доступу до ресурсу: http://mlbase.org/.
20. LargeScaleDistributedDeepNetworks / [J. Dean, G. Corrado, R. Monga та ін.]. // InProceedingsofthe 25th InternationalConferenceonNeuralInformationProcessingSystems. – 2012. – №1. – С. 1223–1231.
21. ExploitingBoundedStalenesstoSpeedUpBigDataAnalytics / [H. Cui, J. Cipar, Q. Ho та ін.]. // In USENIX AnnualTechnicalConference. – 2014. – С. 37–48.
22. Using MPI: portableparallelprogrammingwiththemessage-passinginterface / W.Gropp, W. Gropp, E. Lusk, A. Skjellum. // MIT press. – 1999. – №1.

УДК 681.5.004.4

СМІЩЕНКО Б.О.

СТВОРЕННЯ ПРОГРАМНОГО ДОДАТКУ ДЛЯ КЛАСИФІКАЦІЇ ЗАПИТІВ

У цій статті розглядається прототип додатку, що вирішує проблему класифікації запитів як безпечні та небезпечні. Демонструється використання нейронних мереж задля автоматизації процесу обробки запиту. Реалізацію алгоритмівберемо збібліотекitensorflowта бібліотекиNSL-KDD, що містять приклади запитів, які будуть використані для тренування нейронної мережі.

КЛЮЧОВІ СЛОВА: Нейронні мережі, недостатність прикладів

Inthisarticle is described prototype of application which solves problem of classification requests assecure and dangerous. Neural networks are demonstrated as tool for automation of request processing. Implementation of algorithm is going to be taken from tensorflow and NSL-KDD library as source of training data for neural network.

KEYWORD: neural networks, under-sampling

1. Вступ

У цій статті розглядається прототип додатку, що вирішує проблему класифікації запитів як безпечні та небезпечні. Демонструється використання нейронних мереж задля автоматизації процесу обробки запиту. Реалізацію алгоритмів візьмемо з бібліотеки tensorflowта бібліотекиNSL-KDD, що містять приклади запитів, які будуть використані для тренування нейронної мережі.

2. Актуальність теми дослідження.

Задача підвищення безпеки вхідного трафікує однієюз основних для створення стабільного та безпечного для користувацьких данихсервісу. Одним із засобів забезпечення контролю за величезною кількістю даних, що сформовані безліччюзапитів, є використання програмного забезпечення, спроможного визначати певні види атак. Однак такий підхід є вразливим до нових типів атак, тому доцільно розробити таку систему, яка могла б попереджати та реагувати на можливі новітні загрози. Один з можливих інструментів задля вирішення такого завдання є використання нейронних мереж.

3. Постановка проблеми.

Важливою частиною кожного програмного продукту є забезпечення безпеки як сервісу, так і його клієнтів. Недоліки у безпеці приносять збитки пропорційні кількості необроблених запитів.Тому пропонується розробити систему, модифікація якої буде або автоматичною, або максимально простою.

4. Аналіз останніх досліджень і публікацій.

Незважаючи на те, що кібербезпекастає всебільше невід’ємною частиною будь-якого сервісу, існують загрози, що призводять до дедалі важчих наслідків.При цьому створення програми, що здатна захистити дані клієнтіввід новітніх загроз, ще знаходиться у процесі розробки. Одним з можливих напрямків для посилення безпеки може бути аналіз поведінки користувачів дляпопередження виникнення загроз .

4. Мета статті.

Метою дослідження є створення додатку, що класифікує запити зі сторонніх IP адрес.Також розглядаються засоби підвищення точності визначення типу запитів та вивчення можливості надання системі засобів для самомодифікації.

5. Виклад основного матеріалу.

Задача розпізнавання типів послідовності запитів виглядає наступним чином: на базі існуючої бази даних створюється нейронна мережа, що класифікує запити на потенційно небезпечні та безпечні.

6. Постановка задачі

Маємо множину векторів вхідних змінних $\{X_1, X_2, X_3, X_4, .., X_i, .., X_m\}$ та їх значень $\{Y_1, Y_2, Y_3, Y_4, .., Y_i, .., Y_m\}$, де і – порядковий номер запиту у файлі, М – кількість елементів у файлі, кожне з X_i складається з 41 параметру, які

визначають значення Y_i . Завданням є побудувати модель, що ставить у відповідність векторуXдозначення Y. Значення $Y_i = Y(X_i)$ буде визначати тип та безпечність запиту, що задається 41 параметром. Також однією з проблем даного датасету є неоднорідність кількості прикладів для кожної категорії, що призведе до зниження точності класифікації та, у деяких алгоритмах, до унеможливлення коректних підрахунків. Тобто один із варіантів, через переважаючу кількість прикладів може унеможливити виникнення іншої відповіді.

7. Можливі засоби вирішення наведених задач

Задля приведення кількості прикладів до приблизно однієї величини можемо або скоротити кількість найбільш часто уживаних типів запитів, або додати нові приклади до класів, що мало представлені, але у випадку, коли неможливо отримати нові данні можна спробувати згенерувати нові приклади, базуючись на вже існуючих. Наприклад, мінімізація даних цього класу, де з усіх прикладів, що належать до цього класу, до кожної змінної записуються мінімальні значення кожної із змінних. Таким чином отримуємо також середнє та максимальне значення. Ще одним варіантом може бути незначна модифікація вихідного прикладу, тобто незначна зміна одного або декількох параметрів. Для створення нових прикладів можемо використати бібліотеку imblearn та функцію SMOTE (Synthetic Minority Over-sampling Technique).

Висновки

На прикладі задачі класифікації запитів булипродемонстровані способи модифікації датасетудля підвищення точності прогнозу. Теоретично описано базис для створення нейронної мережі та наведені можливі недоліки при виборі алгоритму класифікації.

Список літератури

1. Matthew K. Thoughtful Machine Learning with python. A test-driven approach. –O`Reilly 2017. – 216 с.

УДК 004.7

ДЗИГА Ю.В.
ХАЛУС О.А.

СИСТЕМА КОНТРОЛЮ І КЕРУВАННЯМ ДОСТУПОМ ДО ІНФОРМАЦІЙНОЇ МЕРЕЖІ

В даній роботі представлений поверхневий огляд того, як працює система контролю і керуванням доступу у інформаційну мережу, а також деякі з її ключових можливостей і особливостей.

КЛЮЧОВІ СЛОВА: безпека, інформаційна мережа, контроль доступу, керування доступом.

This article provides a high-level overview of how Network Access Control solution works and some of the key features and capabilities.

KEYWORDS: security, network, access control, access management.

1. Вступ

Оскільки організаціям зараз доводиться враховувати постійне зростання кількості пристроїв, що отримують доступ до їхніх мереж, та ризики для безпеки, які вони несуть, важливо мати інструменти, що забезпечують видимість мережі, контроль доступу та дотримання пристроями політик безпеки, що є критично необхідним для безпеки їхніх мереж.

Система контролю і керуванням доступом у інформаційну мережу може

заборонити доступ до мережі потенційно небезпечних пристроїв, розмістити їх у карантинній зоні або надати їм лише обмежений доступ до обчислювальних ресурсів, тим самим утримуючи незахищені вузли від зараження мережі.

2. Огляд ключових особливостей

Система поставляється як хмарний сервіс, і це означає, що вона є повністю масштабуєма, не прив'язана до конкретного регіону і не вимагає жодного обладнання. Це також означає те, що поточна версія завжди буде найновішою, з останніми особливостями та можливостями.

Для постійного моніторингу пристроїв, на них встановлюються агенти. Система може використовуватись пристроями як з агентами (agent-based), так і без (agentless), в залежності від ваших потреб та юз-кейсів.

Також до основних особливостей можна віднести контроль доступу з врахуванням ризиків, хмарно-орієнтоване керування бездротових підключень, розміщений у хмарі Radius server та зручне управління користувачами за допомогою груп.

3. Система як хмарний сервіс

Система є дуже легка і зручна у використанні через те, що поставляється як хмарний сервіс. На ній завжди запущене найновіше програмне забезпечення з останніми особливостями та оновленнями безпеки. Оновлення версій проводяться безшовно і поставляються безперервно, покриваючи всі гео-локації. Всі компоненти системи є «чорними ящиками», які не потребують управління. Через те, що це хмарне рішення, система має вбудовану масштабуємість, високу доступність і наявність резервних копій на випадок непередбачуваних обставин.

4. Контроль доступу

Система здійснює контроль доступом у мережу та забезпечує безпечний спосіб передачі даних, використовуючи стандарт IEEE 802.1X, який при правильному використанні, є золотим стандартом для безпечної аутентифікації у мережі. Вона може блокувати потенційно небезпечні пристрої, ізолювати невідповідні кінцеві точки або обмежити доступ до вказаних ресурсів, використовуючи списки контролю доступу (ACL).

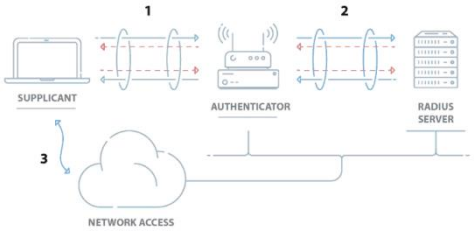


Рис. 1. Компоненти 802.1X

За подібною схемою побудована архітектура нашої системи.

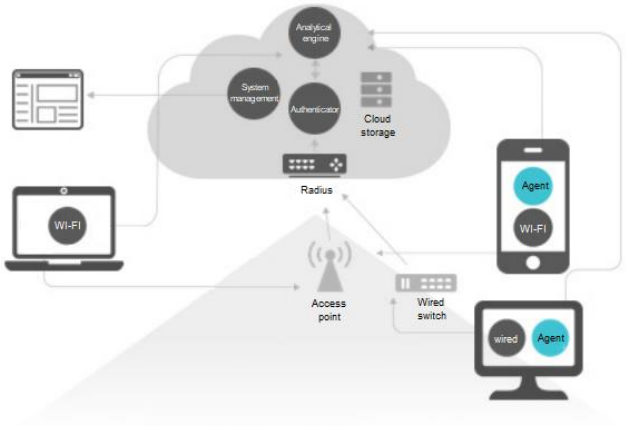


Рис. 2. Архітектура системи

5. Постійний моніторинг пристроїв (агент)

Однією з основних частин системи є агент. Це легкий додаток, який встановлюється на кінцеві пристрої користувачів. Після того як агент був встановлений, його необхідно зареєструвати, ввівши облікові дані користувача. Це всі дії, які вимагаються від кінцевого користувача.

Після того як агент був включений, він починає в режимі реального часу проводити оцінку пристрою і відправляти данні у систему. Такі дані можуть включати: місце знаходження, встановлені програми, антивірус, встановленні оновлення, відкриті порти і т.д.

6. Контроль доступу на основі ризиків

На даний момент є багато подібних систем, які забезпечують керування доступом для пристрою. Також є багато систем, які просто збирають інформацію про різні пристрої. Наша ж система поєднує ці два фактори, і забезпечує моніторинг пристрою в реальному часі та використання цієї інформації для керування доступом.

Адміністратору системи потрібно всього лише створити відповідну політику та визначити атрибути, які можуть нести

потенційну загрозу для мережі, яка керується системою.

Коли пристрій буде підключатись у мережу, система оцінить наскільки він потенційно небезпечний, враховуючи сукупне значення, яке буде обраховано на основі атрибутів певної політики, та вирішить, що з ним робити далі.

7. Групи і динамічний VLAN

Для додаткової зручності у управлінні користувачами, система надає можливість визначати, що для них є дозволено, додаючи їх у групи. Політика на рівні групи включає:

- Визначення членів (користувачі \ пристрої)
- Доступні мережі (дротові \ бездротові)
- Політику безпеки з врахуванням ризиків

Висновок

Традиційні системи контролю доступу до мережі, які встановлюються безпосередньо на апаратне забезпечення користувача (on-premise), мають багато недоліків пов'язаних з необхідністю в утримуванні кваліфікованих кадрів, які зможуть правильно налаштовувати та в подальшому підтримувати систему. Основна перевага нашої системи у простоті, тому що не потрібно жодного додаткового програмного забезпечення, складних конфігурацій чи допомоги висококваліфікованих експертів. Наша система може бути розгорнена та встановлена у лічені хвилини, цим самим ліквідуючи всі недоліки і незручності перераховані вище, та підкреслюючи той факт, що майбутнє стоїть за хмарними технологіями.

Література

1. Що таке NAC (Network Access Control)? [Електронний ресурс] – Режим доступу до ресурсу: https://www.cisco.com/c/en/us/products/security/what-is-network-access-control-nac.html
2. Для чого нам потрібний NAC? [Електронний ресурс] – Режим доступу до ресурсу: https://www.infosecurityeurope.com/__novadocuments/439114?v=636539318028770000
3. Переваги та недоліки NAC [Електронний ресурс] – Режим доступу до ресурсу: https://www.networkworld.com/article/2293857/nac-pros-and-cons.html.
4. Що таке 802.1X? [Електронний ресурс] – Режим доступу до ресурсу: https://www.securew2.com/solutions/802-1x/.
5. Важливість 802.1X для NAC [Електронний ресурс] – Режим доступу до ресурсу: https://www.fortinet.com/content/dam/fortinet/assets/white-papers/wp-understanding-nac.pdf.
6. NAC як сервіс [Електронний ресурс] – Режим доступу до ресурсу: https://www.portnox.com/blog/our-technology/nac-as-a-service-webinar/.

УДК 004.9

КОЗЛОВА О.С.

ОРГАНІЗАЦІЯСИСТЕМИ ДЛЯ РЕКОМЕНДАЦІЇ ПРОДУКТІВ З ДОПОМОГОЮ ПРОГРАМУВАННЯ НА ОСНОВІ ПРАВИЛ

В даній статті розглянуто питання створення інформаційної системи, заснованої на правилах, для управління залишками продуктових магазинів. Сформульовано та описано етапи проектування такої системи, проаналізовано методи реалізації.

Ключові слова: РЕКОМЕНДАЦІЙНА СИСТЕМА, ПРАВИЛА, ЛОГІЧНЕ ПРОГРАМУВАННЯ

This article addresses the issue of creating a rule-based information system for managing grocery store balances. The stages of designing such a system are formulated and the methods of implementation are analyzed.

Keywords: RECOMMENDATION SYSTEM, RULES, LOGICAL PROGRAMMING

1. Вступ

На жаль, без перебільшення можна сказати, що харчові відходи – одна з найбільших проблем, з якою сьогодні стикається людство. Близько 50% всієї їжі, що виробляється у світі щорічно, ніколи не буває вжитою, а натомість викидається на смітники, що несе за собою більше 1 трильйона доларів збитків.

Практика супермаркетів та закладів публічного харчування відповідає за велику кількість харчових відходів, адже багато продуктів так і не доходять до рук споживача, враховуючи їх ціну, стан, необхідність і тому подібне.

Суть даної роботи полягає у створенні програмного продукту, який шляхом програмування на основі правил дозволить індивідуально підбирати для користувачів продукти із магазинів та супермаркетів, які щоденно готуються та мають малий термін придатності, або ж термін придатності яких скоро закінчиться і вони продаються зі знижками. Це продукти, які повинні збуватися магазинами у першу чергу, аби не нести за собою матеріальних збитків для бізнесу та не продукувати нових харчових відходів. Таким чином, розміри харчових відходів можна значно мінімізувати. Аналогів такої програми в Україні поки не існує.

2. Модель даних

Для створення подібної системи необхідна велика база даних продуктів з детальною інформацією про термін придатності, склад та харчову цінність. В Україні відкритої бази продуктів поки що не існує, тому в цілях демонстрації у роботі реалізована інтеграція з відкритою базою даних Міністерства сільського господарства США (USDA)[1].На даний момент у базі зберігаються близько 10 тисяч одиниць продукції.

У результаті інтеграції усі продукти розбиті по ієрархічному дереву категорій, кожен продукт має унікальний код продукту, перші цифри якого відповідають за категоризацію. Наприклад:

55103000 Млинці з фруктами

Перша цифра (5) означає, що продукт знаходиться у категорії «Зернові продукти». Перші дві цифри (55) означають, що продукт знаходиться у підкатегорії «Млинці, вафлі та французькі тости». Перші три цифри (551) означають, продукт знаходиться у підкатегорії «Млинці». Наступні цифри це унікальні цифри продуктового коду. Таким чином, заздалегідь організована категоризація спрощує пошук потрібних продуктів, спрощує генерацію рекомендації за категоріальною ознакою та передбачає легке інтегрування з новою системою продуктової бази.

Складові продуктів – інгредієнти, котрі мають ідентифікатор, назву та ступінь середньостатичної частоти вживання, зберігаються у базу даних у якості окремої сутності та поєднані з продуктами зв’язками багато-до-багатьох. Найбільш рідко вживані інгредієнти пропонуються користувачеві для виключення з раціону споживання.

На даний момент у системі передбачені наступні факти харчової цінності: білки, жири, вуглеводи та загальна калорійність. Інформація про харчову цінність кожного продукту аналізується з інтеграційної продуктової бази та зберігається у окремій таблиці з використанням зв’язку багато-до-багатьох.

Рекомендації будуються на основі індивідуальних переваг кожного користувача, для цього передбачено 11 типів дієт (втрата ваги, палео дієта, середземноморська дієта, кето дієта, дієта для діабетиків, вегетаріанців, веганів, пескатаріанців, їжа без вмісту глютену, уникання молочних продуктів, їжа з низьким вмістом солі) [2].

На рисунку 1 зображена ER (entity relationship) діаграма.

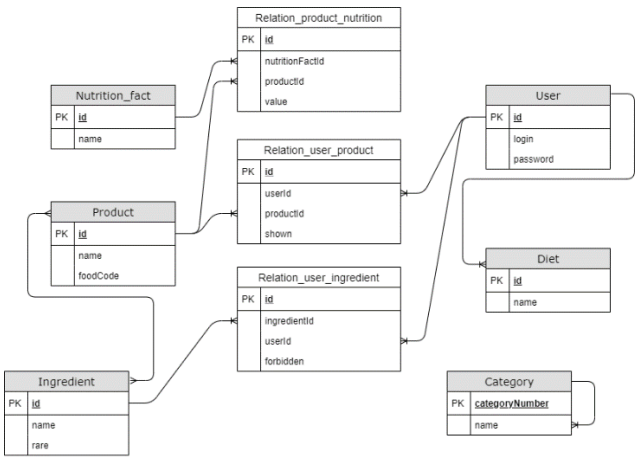


Рис. 1.ER діаграма

3. Аналіз існуючих методів

Інформаційні системи, засновані на правилах, використовуються для того, аби зберігати деяку базу знань та маніпулювати нею для інтерпретації інформації, що там знаходиться. Класичним прикладом такої системи є експертна система, орієнтована на конкретну предметну область, яка використовує правила для того, аби здійснювати вибір.[3]

Переваги систем, заснованих на правилах:

- Модульність та модифікованість. Правила легко закодувати та додати до працюючої системи, не змінюючи інших правил або бізнес-логіки.
- Природність та простота. Формат «якщо..то» є природнім методом для багатьох видів експертизи, здійснених звичайною людиною.
- Інтенсивні знання. Експертна система може бути адаптована для використання у іншій предметній області, для цього варто лише змінити базу знань та правил.

Для великих продуктових систем мови, у яких реалізований механізм логічного програмування (Prolog) не підходять, у них, як правило, відсутні такі пристрої програмного забезпечення як модулі, приховання інформації, повторне використання, що унеможливлює повноцінну реалізацію бізнес-логіки клієнт-серверних систем. Тому було вирішено використовувати об’єктно-орієнтовану мову програмування для реалізації системи правил.

Одним із найбільш поширених методів проектування системи, заснованої на правилах, є поділ її на наступні етапи : Правила, Атрибути, Діаграма відношень атрибутів, Логічний дизайн, Реалізація[4]. Наведемо приклад такого проектування для випадку, коли користувач слідує кето дієті.

3.1 Правила

1) Якщо користувач добавив до списку дієт елемент з ідентифікатором 4, він слідує кето дієті.

2) Якщо дієта – кето дієта, то заборонені категорії – 10, 11, 12.

3) Якщо дієта – кето дієта, то заборонені ключові слова – «хліб», «цукор», «борошно».

4) Якщо продукт не порожній, то виставити категорію продукту, харчову цінність, склад та назву.

5) Якщо список заборонених категорій не порожній і категорія продукту не порожня, добавити до списку операцій – перевірити заборонені категорії.

6) Якщо список ключових слів не порожній і назва продукту не порожня і склад продукту не порожній, додати до списку операцій – перевірити ключові слова.

7) Якщо дієта – кето дієта і харчова цінність продукту не порожня, то додати до списку операцій – перевірити вміст вуглеводів.

8) Якщо операція – перевірити вміст вуглеводів і вміст вуглеводів не перевищує 5%, додати продукт до списку відображених.

9) Якщо операція – перевірити заборонені категорії і категорія продукту не міститься у списку заборонених, додати продукт до списку відображених.

10) Якщо операція – перевірити ключові слова і у назві продукту порожній перетин зі списком ключових слів і у списку складових продуктів порожній перетин зі списком ключових слів, додати продукт до списку відображених.

3.2 Атрибути та діаграма відношень атрибутів

На етапі складання таблиці атрибутів аналізується предметна область і визначаються основні одиниці, що оперуються правилами. Це їх вхідні та вихідні дані, а також атрибути, що використовуються всередині правил.

На діаграмі відношень атрибутів зображені рівні взаємодії між усіма атрибутами. Зі збільшенням рівня, зменшується рівень абстракції взаємодії. Тобто на нульовому рівні ми зазначаємо, що додавання продукту залежить від самого продукту та ідентифікатора дієти користувача, а на останньому рівні можемо бачити усі атрибути, які впливають на цей вибір. Діаграма відношень атрибутів значно спрощує логічний дизайн системи і, як наслідок, сприяє ефективній реалізації.

3.3 Логічний дизайн

Логічний дизайн передбачає побудову так званого дерева рішень, яке показує які правила в якому порядку повинні викликатись, які параметри вони приймають на вхід і який результат видають. Таблиці 1-5

являють собою детальний опис дерева рішень, де правила поділені на логічні групи, пов'язані переходами.

3.4 Реалізація

На даний момент було реалізовано функціонування даної системи, заснованої на правилах, з використанням двигуна правил Drools, де для впорядкування та виклику правил використовується алгоритм Pete [5]. На розмірі масиву продуктів у 10000 та 100 користувачах, час виконання даного алгоритму не перевищує стандартного очікування користувача на відгук додатку в 3 секунди.

Для прямої об'єктної реалізації описаного алгоритму необхідно реалізувати як мінімум наступні класи : Attribute, AttributeType, Rule, RuleEngine та провести так званий декартів добуток множин користувачів та продуктів, аби для кожної пари користувач-продукт викликати створені правила і здійснити вибір на користь або проти деякого продукту. Об'єктна реалізація логічної моделі дає достатньо деталізоване рішення, легке для модифікації розробником. Необхідно порівняти ефективність прямої реалізації двигуна правил з існуючою реазілацією алгоритму Pete і зробити висновки про доцільність використання кожного із способів.

Таблиця 1 – Група 1

Info	Prec	Assert	Ctrl	
I	aDI	aDIE	N	E
1	4	Кето дієта	2.2	1.1

Таблиця 2 – Група 2

Info	Prec	Assert		Ctrl	
I	aDIE	aRC	aRW	N	E
2	Кето дієта	10 11 12	Хліб Цукор Борошно	3.3	2.2

Таблиця 3 – Група 3

Info	Prec	Assert				Ctrl	
I	aPR	aPC	aPN	aPI	aPNU	N	E
3	Продукт	Характеристики продукту				4.4	3.3

Таблиця 4 – Група 4

Info	Prec							Assert	Ctrl	
I	aDIE	aRC	aRW	aPC	aPN	aPI	aPNU	aOP	N	E
4		∅		∅				Перевірка категорій	4.5	4.4
5	Кето дієта						∅	Перевірка вуглеводів	4.6	4.5
6			∅		∅	∅		Перевірка ключових слів	5.7	4.6

Таблиця 5 – Група 5

Info	Prec							Decision	Ctrl	
I	aOP	aRC	aRW	aPC	aPN	aPI	aPNU	aPA	N	E
7	Перевірка вуглеводів						Вуглеводи < 5%	true	5.8	5.7
8	Перевірка категорій	∅		⊄aRC				true	5.9	5.8
9	Перевірка ключових слів		∅		∩aRW= ∅	∩aRW= ∅		true	1.1	5.9

Висновок

Для вирішення актуальної проблеми, що виникає при збуті продуктів мережами супермаркетів, було сформульовано цілі та задачі, проаналізовано існуючі методи проектування та реалізації системи, заснованої на правилах, здійснено вибір на користь двох реалізацій, які варто порівняти, враховуючи швидкість їх роботи, зручність використання та перевикористання, модифікованість та простоту.

Список літератури

1. USDA. FoodDataCentral [Електронний ресурс] / USDA – Режим доступу до ресурсу: fdc.nal.usda.gov.
2. VanRaesS. 10 PopularDietsExplained [Електронний ресурс] / SueVanRaes // ChopraCenter. – 2018. – Режим доступу до ресурсу: https://chopra.com/articles/10-popular-diets-explained.
3. Wu X. Object-Oriented Modeling of Rule-Based Programming / X. Wu, X. Lin. // Intl Conference on Software Engineering and Knowledge Engineering. – 1997.
4. Antoni L. Logical Foundations for Rule-Based Systems / Ligesa Antoni. – Krakow: AGH University of Science and Technology Press, 2006. – 309 с.
5. Kumar N. Rule based programming with Drools / N. Kumar, D. D Patil, D. M.Wadhai. // (IJCSIT) International Journal of Computer Science and Information Technologies. – 2011. – №2. – С. 1121–1126.

УДК 004.94

АЛЕКСИКОВ І. О.
ДОЛІННА Є. О.

СИСТЕМА МОДЕЛЮВАННЯ ЯКОСТІ РОБОТИ ТЕРМОПЛАСТАВТОМАТУ

В даній статті розглянуто можливість оптимізації втрат заводу, зокрема фінансових та втрат якості, на прикладі мінімізації часу, витраченого на технологічне обслуговування термопластавтомата. У роботі описані усі фактори впливу на час технічного обслуговування та сукупність даних, яка має бути проаналізована для подальшої реалізації алгоритмів оптимізації.

ТЕРМОПЛАСТАВТОМАТ, ТЕХНОЛОГІЧНЕ ОБСЛУГОВУВАННЯ, ПРОСТОЇ, КОЕФІЦІЄНТ ЕФЕКТИВНОСТІ ОБЛАДНАННЯ

In this article the possibility of optimization of plant losses, in particular financial and quality losses, is considered, for example minimizing the time spent on technological maintenance of the automatic molding machine. The paper describes all the factors affecting maintenance time and the set of data that must be analyzed to further implementation of optimization algorithms.

INJECTION MOLDING MACHIME, MAINTENANCE, DOWNTIMES, OVERALEQUIPMENT EFFECTIVNESS

1.Вступ

В даній статті розглянуто рішення для оптимізаціїпроцесу роботи заводую. Одним із найбільш затратних процесів є процесс техобслуговування машин на лінії. Технічне обслуговування машини може бути виконано лише під час простою машини, коли вона не виробляє продукцію. На даний момент, існує три варіанти планування простоїв:

1. Reactivemaintenance – робота машини до першої поломки, що призводить до її простою.
2. Preventivemaintenance – робота машини до запланованого простою, що відбувається в визначений час.
3. Predictivemaintenance – робота машини до критичного рівня, кожний простій планується відповідно да стану машини в реальному часі.

Оптимальним варіантом, що дозволить мінімізувати втрати, являється саме Predictivemaintenance.

ЦільPredictivemaintenance [1] не тільки запобігти незапланованим простоям, а також

мінімізувати час, витрачений на заплановані простої.

Так як для побудови моделі, що буде вираховувати потребу технічного втручання (Predictivemaintenance), треба аналізувати дані машини, а у кожної машини свої показники.

2. Термопластавтомат

Існує багато видів машин, що використовуються на заводах, у даному проекті ми будемо розглядати лише Термопластавтома́т (ТПА або IMM - Injectionmoldingmachine) — інжекційно-ливарна автоматизована машина для лиття пластмас під тиском, є універсальним обладнанням для отримання поштучних виробів з пластмас.

Основними параметрами, які мають найбільший вплив на техніко-економічні характеристики термопластавтоматів, є:

- об'єм впорскування за цикл (об'єм виливки);
- об'ємна швидкість упорскування (час упорскування);
- тиск лиття;
- площа литва;
- зусилля замикання і розкриття прес-форми;
- хід рухомої плити, максимальна відстань між плитами, їх жорсткість та швидкохідність;
- пластифікаційна здатність і діапазон температур інжекційного циліндра.

Важливо: перераховані дані не є точно визначеним переліком параметрів, що будуть використовуватись при побудові моделі, надалі данні будуть змінюватись.

Саме зміна даних параметрів, або відхилення їх від норми може сигналізувати про зменшення ефективності машини, а отже в подальшому про її поломку.

3. Коефіцієнт ефективності

Під час виробничого процесу існує потреба оцінки ефективності роботі обладнання, щоб запобігти фінансовим втратам. Загальна ефективність обладнання, надалі OEE (OverallEquipmentEffectiveness) – це основний показник загального догляду за обладнанням, він відображає саме ступінь ефективності використання обладнання. Значення OEE відповідає на наступні питання:

- Наскільки ефективно ви використовуєте своє обладнання?
- Які показники знижують ефективність обладнання?
- На що саме звернути увагу, щоб покращити ефективність?

Для розрахунку OEE необхідно перемножити наступні показники:

A. Доступність обладнання (Availability) - відношення машинного часу до чистого часу роботи.

P. Продуктивність (Performance) – відношення поточного вироблення до запланованого;

Q. Якість (Quality) – відношення кількості якісних деталей до усіх вироблених деталей.

Вимірювання загальної ефективності обладнання передбачає моніторинг того, як функціонує обладнання або протікає процес. Визначення OEE відбувається на основі щоденного фіксування показників, що відображають стан обладнання, що сприяє відкритому підходу до обміну інформацією і запобігає випадкам коли погане функціонування верстатів ставиться в провину конкретного робітника.

4. Простої

Простої - це час, коли машина стоїть без роботи. Якщо існує деяка виробнича лінія, то при простої однієї машини зупиняється вся лінія, а отже зупиняється виробництво, що призводить до значних фінансових втрат.

Простої поділяються на:

- Заплановані – простої, втрати від яких уже прораховані та мінімізовані в прибуток. Приклад: раз в місяць певній машині потрібне технічне обслуговування, яке триває годину. Або ж зміна прес-форм на термопластавтомат машині.

Важливо: заплановані простої також є витратами.

- Незаплановані – (в більшості випадків) це саме поломка обладнання, що призводить до значних втрат, які неможливо прорахувати.

Порівняльна характеристика запланованих та незапланованих простоїв:

Припустимо, простій машини А коштує 1000 умовних одиниць за годину.

Запланований простій, а саме технічна перевірка, раз в місяць на дві години буде коштувати 2*1000 у.о. = 2000у.о., так як це запланована подія, то мінімізовані витрати на виклик команди з технічної підтримки, час, поки технічна підтримка доїде до місця події.

Незапланований простій буде коштувати дорожче. Година на виклик команди технічного обслуговування + година витрачена на дорогу + 2 години роботи, отже виходить 1*1000 + 1*1000 + 2*1000 = 4000 у.о., що перевищує оплату запланованого простою у два рази.

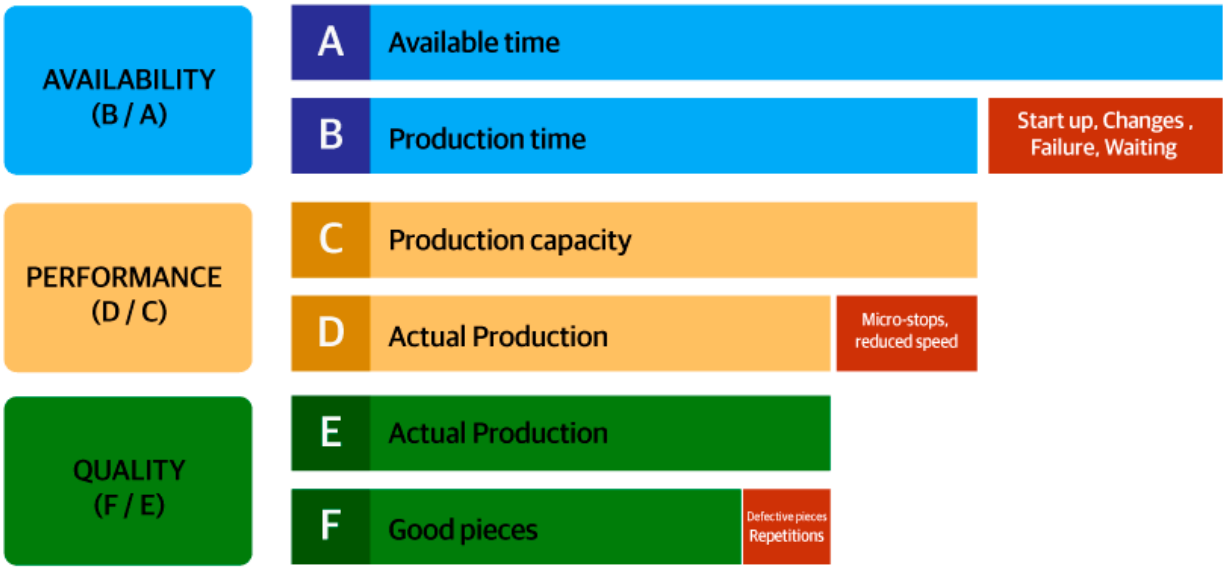


Рис. 3.1 – Графічне зображення складових коефіцієнту ефективності[2]

5. Рішення

Розглянемо впровадження технології прогнозу простоїв[3] (Predictivemaintenance) для машин типу ТПА.

Отже, розібравшись з термінологією даного проекту, можна переходити до більш технічного опису, а саме до архітектури:

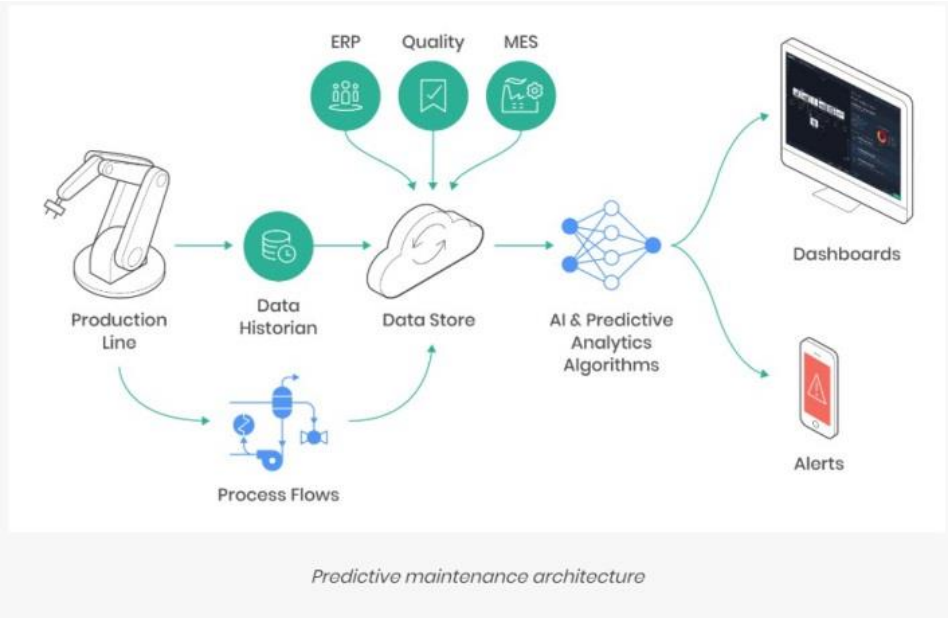


Рис. 5.1 – Архітектурасистеми управління техобслуговування термопластавтомату [4]

Як можна побачити з архітектури, для подальшої розробки програмного продукту потрібно:

1. Зібрати усі потрібні дані для аналізу стану конкретної машини:
    a. Історичні дані стану даної машини впродовж виробництва.
    b. Дані, що відображають стан машини у реальному часі.
2. Структурувати дані для подальшої обробки.
3. Розробити алгоритми для обробки даних, що дозволять нам прогнозувати простої машини, щоб запобігти її поломці
4. Розробити інтерфейс для користувача, що дозволить йому переглядати результати роботи алгоритмів.
5. Порівняти розроблені алгоритми.

6. Алгоритми

Розглянемо задачу регресійного виду[5], тобто, ми маємо деяку періодичність роботи машини, а саме від обслуговування до поломки машини, та ймовірність виходу з ладу даної машини. Нам необхідно, оперуючи історичнимиданимироботи машини за деякий період,спрогнозувати, через який час яка буде поломка.В залежності від нашого прогнозу інженери можуть планувати необхідні заходи, а саме технічне обслуговування машин.

Тобто для конкретної ситуації доцільно використовувати регресійні алгоритми, які мають задачу саме знаходити час,через який передбачається поломка в системі. В рамках цієї задачі необхідно розглянути такі алгоритми, як лінійна регресія, випадковий ліс та інші.

Для того, щоб зменшити кількість помилок, при прогнозуванні наступної поломки на машині, необхідно встановити дозволений в рамках нашої задачі коефіцієнт значимості. Зазвичай він 0.05, тобто ймовірність поломки машини раніще розрахованого терміну буде дорівнювати 0.95. Це значення задається на початку пошуку моделі для прогнозування та всі моделі, що на тестових значеннях не будуть давати достатньої точності не будуть приймати участь у подальшій розробці. Цей коефіцієнт не є остаточним та може змінюватися залежно від подальшої роботи машини. Інженери, що обслуговують дану машину можуть запропонувати збільшити ліміт для більшої точності прогнозування, або навпаки збільшити коефіцієнт.

Висновки

Отже, наша основна ціль, використовуючи алгоритми регресії, передбачити решту корисного терміну експлуатації (RUL(remainingusefullife)) для машини. Це дозволить створити рішення, що може бути проаналізоване, та отримати значення, а саме, скільки часу машина матиме до відмови.Але, навіть після побудування регресійної моделі, необхідно продовжувати роботу над модифікацією моделі, тому що саме під час роботи системи на підприємстві можна дізнатися від інженерів як для них краще, щоб працювала система. Зміни такого роду можуть включати в себе зменшення коефіцієнта значущості, додавання нових правил, які можуть бути побудовані в залежності від деякого періоду (наприклад, підвищення навантаження машини в залежності від дня тижня) або від комплексного значення декількох датчиків, а не одного.

Список літератури

1. R. KeithMobley Introduction To Predictive Maintenance: book. London: Elsevier Science & Technology, 2002. 456 c.
2. INDUSTRY 4.0 PREDICTIVE MAINTENANCE. The Complete Guide. SEEBO: [Електронний ресурс]// Режим доступу:https://www.seebo.com/predictive-maintenance/
3. Mike Barlow Predictive Maintenance: book. Seattle: O'Reilly Media, Inc., 2015.
4. CalculateOEE.SistemasOEE,2017[Електронний ресурс] //

Режим доступу:https://www.sistemasoee.com/en/oee/89-for-dummies/108-calculate-oee

5. Automated Machine Learning Hyperparameter Tuning in Python, 2018[Електронний ресурс] //

Режим доступу: https://towardsdatascience.com/automated-machine-learning-hyperparameter-tuning-in-python-dfda59b72f8a

УДК 004.94

ШАВЕРСЬКИЙ І.О.

ВИКОРИСТАННЯ МЕТОДІВ ЛІНГВІСТИЧНОГО АНАЛІЗУ ДЛЯ ПЕРЕДБАЧЕННЯ РІВНЯ ГЛЮКОЗИ У ДІАБЕТИКІВ

В даній статті буде розглянута тема використання лінгвістичних алгоритмів для прогнозування рівня глюкози в крові у діабетиків. Описані кроки для аналізу даних щодо рівня глюкози і відповідні розрахунки.
КЛЮЧОВІ СЛОВА
Діабет , глюкоза , лінгвістичний алгоритм , методи лінгвістичного аналізу, лінгвістичні шаблони.

In this article, the theme of linguistical gorithms for predicting the level of glucose in the blood of diabetics is highlighted. The steps for analyzing glucosed at and corre sponding calculation sare shown.
KEYWORDS
Diabetes, glucose, linguistic algorithm, methods of linguistic analysis, linguistic templates.

1. Введення

Кількість людей які хворіють діабетом близька до півмільярда людей(приблизно кожна 15 людина в світі). Діабет небезпечний непередбаченими скачками рівня глюкози. В людей без цієї хвороби ці скачки «знешкоджуються» збільшенням вироблення інсуліну підшлунковою залозою, який в свою чергу відповідає за процес обміну енергією в організмі. Але у діабетиків даний гормон в дефіциті і підшлункова залоза може не виробити достатньої кількості щоб врівноважити рівень цукру в крові.[1]Дані показані на рис.1.

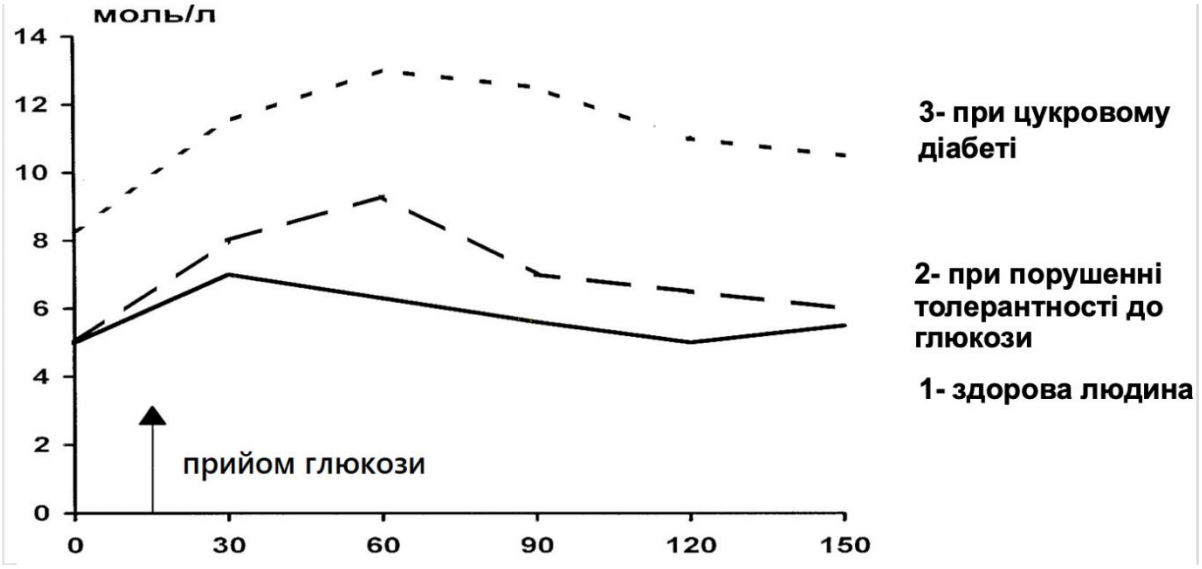


Рис. 1Динаміка рівня глюкози

2. Відомості про використаний алгоритм

Загальновідомим є використання лінгвістичного алгоритму[2] в біоінженерії для швидкого порівняння різних даних таких як ДНК, послідовностей нуклеотидів, пошук інформативних елементів у протеїнах та амінокислотних послідовностях.

Тому було обрано даний метод для пошуку закономірностей в зміні рівня глюкози у діабетиків. Основною перевагою даного методу є можливість довготривалого прогнозу наступних показників шляхом пошуку шаблонів її поведінки вже в зібраних даних.

Абсолютним плюсом лінгвістичного аналізу є виокремлення саме шаблонів поведінки і зручність роботи з ними.

Основні кроки алгоритму наступні:

1. Зчитування даних про виміри глюкози
2. Інтерполяція цих даних, щоб отримати часові ряди
3. Перетворення часових рядів на лінгвістичні ланцюги
4. Пошук критичних значень та запам'ятовування частин ланцюга перед ними
5. За допомогою попередніх кроків «навчаємо» нашу систему
6. Далі при введенні нових значені шукаємо співпадіння с частинами ланцюга, що передували критичним значення

3. Перетворення часових рядів у лінгвістичні.

Основною ідеєю є присвоєння літері певного проміжку значень і потім переведення будь-якого числа в літеру. Варто зазначити що при використанні даного методу необхідно знати можливий діапазон значень для своєчасного обрання алфавіту.

Проблема вибору алфавіту постає у всіх роботах з лінгвістичним аналізом окрім робіт пов'язаних з натуральною мовою.

В нашому випробуванні англійський алфавіт був обраний як оптимальний. Результати переводу деякого ряду значень глюкози в лінгвістичний ряд представлені на рис.2.

Відповіднодо числового ряду «100,119,123,216,211,257,129,239,129,340,67,206,288,77,228,259,256,109,96,200,128,192,263,81,179,88,185,104,86,60»отримуємолінгвістичний ряд FFGKKMGLGRDKODLMMFEKGJNEIEJFED.

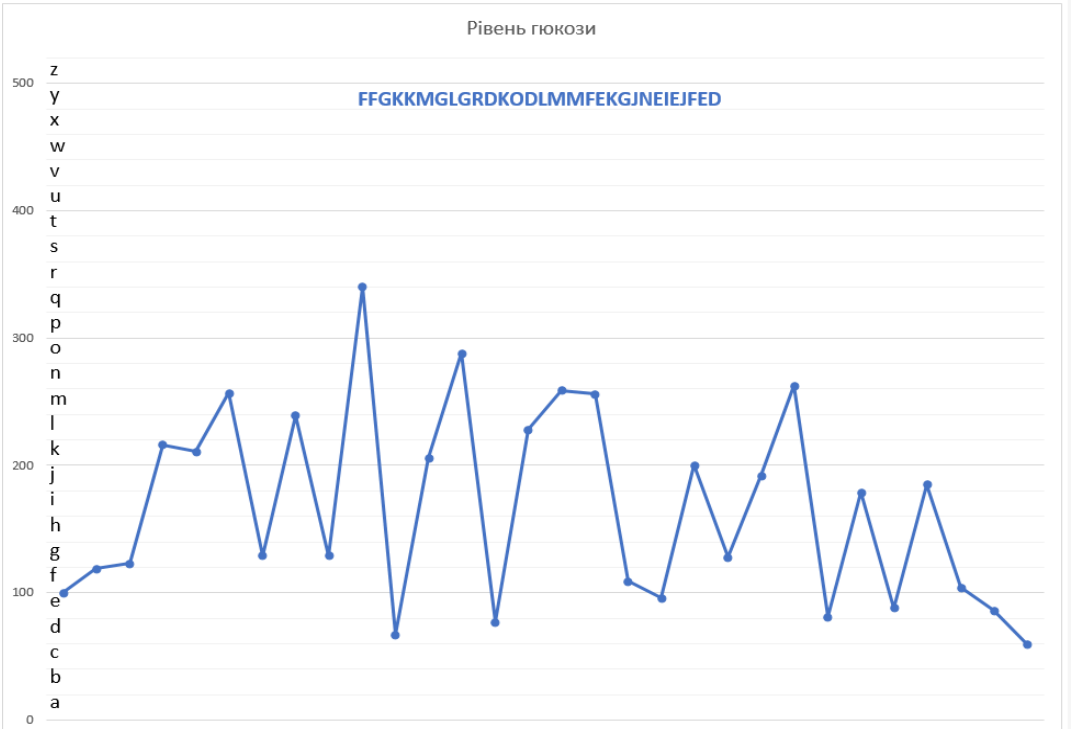


Рис. 2 Рівень глюкози

4. Обробка отриманих лінгвістичних рядів

Наступним кроком при обробці тестових даних (які не були числовим рядом , а були введені людиною)повинна бути інтерполяція – в ході роботи була обрана поліноміальна інтерполяція [3], яка згодом може бути вдосконалена з використанням замість аргументу функції - часу виконання замірів або зміною методів інтерполяції.

Наразі інтерполяція проводиться за допомогою поліномів на відрізках менше 10 проміжків.

Приведемо приклад на перших 9 показниках

Початкові дані приведені на рис.3 а результат на рис. 4

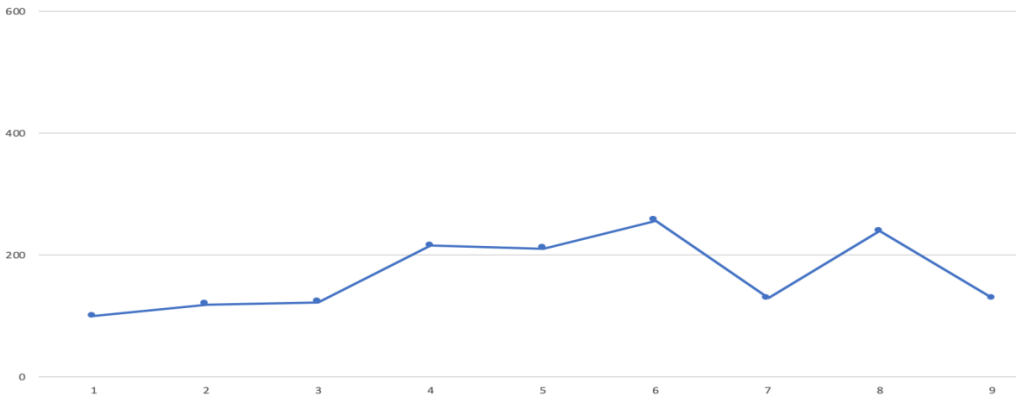


Рис. 3 Початкові дані

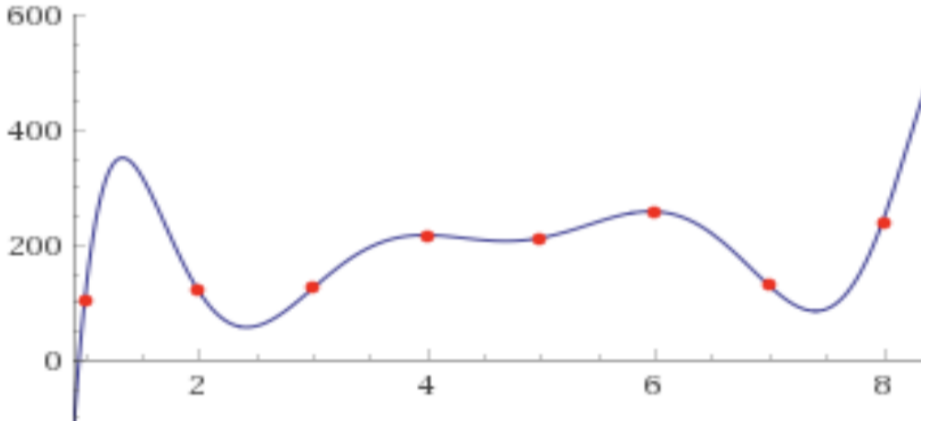


Рис. 4 Інтерполяція за допомогою полігонів

Отримуємо наступний результат:

$$-\frac{7297x^8}{40320}+\frac{71789x^7}{10080}-\frac{37651x^6}{320}+\frac{190891x^5}{180}-\frac{10881493x^4}{1920}+\frac{26253353x^3}{1440}-\frac{343452971x^2}{10080}+\frac{27977777x}{840}-12648$$

Інтерполяція проводиться з метою спрощення роботи з шаблонами та приведенні замірів до одного формату.

4. Алгоритми для порівняння рівня глюкози

Наступним кроком для порівняння отриманих даних і шаблонів поведінки є операція знаходження відстані між даними рядами.

Для цього будемо використовувати відстань Левенштейна.[4]

Відстань Левенштейна - метрика, що вимірює різницю між двома послідовностямисимволів. Вона визначається як мінімальна кількість односимвольних операцій, необхідних для перетворення однієї послідовності символів в іншу.

Основними операціями в даному алгоритмі є вставка, заміна або видалення. Даний алгоритм відомий ще з 60х років минулого сторіччя і є одним з найбільш широко використовуваних. Порівнявши поточні дані поведінки рівня глюкози з тестовими(шаблонними даними) як результат буде взятий той шаблон де відстань Левенштейна є найменшою .

Висновок

На прикладі протестованого набору даних можна зробити висновок , що при більшій кількості зібраних показників та більшій кількості людей які дійсно будуть вносити свої дані – лінгвістичний аналіз може бути корисним в даній області і давати передбачення з достатньою точністю для попередження піків коливань рівня глюкози.

Список літератури

1. Патогенетичне лікування цукрового діабету типу 2 / О. П. Кіхтяк. — К.: Софія-А, 2006. — 161 с.
2. Баклан І. В. Лінгвістичнемоделювання: основи, методи, деякіприкладніаспекти / ІгорВолодимирович Баклан. // Системнітехнології. – 2011. – №3. – С. 10–19.
3. Литвин, О. М. Інтерлінаціяфункцій та деякіїїзастосування/ О. М. Литвин. – Х., Основа, 2002. – 544 с.
4. Levenshtein, Vladimir I. (February 1966). "Binary codes capable of correcting deletions, insertions, and reversals". Soviet Physics Doklady. 10 (8): 707–710

УДК 004.94

ЦИЦИЛЮК А.В.,
ОЛІЙНИК Ю.О.

ВИКОРИСТАННЯ МЕТОДІВ ЛІНГВІСТИЧНОГО АНАЛІЗУ ДЛЯ ПОШУКУ АНОМАЛІЙ В ЕКГ

В даній статті буде розглянуто використання методів лінгвістичного аналізу для знаходження аномалій в ЕКГ(Електрокардіограмі).Будуть приведені поточний результат тестування та загальний алгоритм роботи.

КЛЮЧОВІ СЛОВА: ЕКГ, Електрокардіограма, хвороби серця, серцеві аномалії, методи лінгвістичного аналізу, лінгвістичні шаблони.

This article will discuss the use of linguistic analysis methods to find anomalies in the ECG (Electrocardiogram). The current test result and general algorithm of operation will be presented.

KEYWORDS: ECG, Electrocardiogram, Heart Diseases, Cardiac Anomalies, Linguistic Analysis Methods, Linguistic Templates

1. Введення

В наш час у багатьох люде є проблеми з серцево судинними захворюваннями і аналіз ЕКГ може допомогти в передчасному виявленні цих захворювань.

Як відомо ЕКГ – це визначення показників серцевого ритму , яке відображається на паперовій стрічці і згодом може бути оцінено лікарем. Повноцінний аналіз ЕКГ займає багато часу лікарем і враховуючи людський фактор може бути невірним. Оскільки результати ЕКГ мають шаблонний вигляд (рисунок який виходить в результаті є повторюваним) ми можемо застосувати аналіз на цих даних і маючи дані про аномалію знайти схожі випадки.

Тож аналіз відповідних даних може допомогти лікарям і зменшити вірогідність помилки.

2. Загальні відомості про алгоритм

Методи лінгвістичного аналізу з’явилися в світі програмування ще в 60х роках минулого сторіччя і з того часу не втрачають своєї популярності.

Методи лінгвістичного аналізу набули широкого використання в багатьох сферах. Основною сферою їх використання

являються методи перекладу текстів, перевірки орфографічних помилок.

Але дані методи також популярні і в сфері медицини при аналізі генів,хромосом та білків.

Основний причинроботи алгоритму при аналізі електрокардіограм переведення числового ряду в буквений, де кожному символу відповідає певний числовий проміжок.

Таким чином з числового ряду ми отримуємо символьний і зберігаємо частини з аномаліями. Згодом при пошуку аномалій в реальних даних ми будемо використовувати дані лінгвістичні ряди з аномаліями як шаблон.Алгоритм описано в статті.[1]

Згодом після отримання загальної бази будуть проведені наступні кроки :

1. Пошук точного шаблону відповідності з фрагментами в лінгвістичному ланцюжку(Для чіткого знаходження аномалій).
2. Пошук шаблону аномалій з певною похибкою відстанню. Алгоритми використані для пошуку відстані будуть наведені в пункті 3.
3. Узагальнення сітки. Тобто зменшення або збільшення проміжків між вимірами. (Методами інтерполяції чи іншими)
4. Повторення кроків 1-2
5. Коригування результатів по іншій осі(не часовій, а осі з результатами). Зжимання і розжимання результатів
6. Повторення кроків 1-2

Спираючись на отриманий збіг, фахівець робить висновок про виявлену аномалію серця.

3. Опис алгоритмів для визначення відстані між ланцюгами

Можуть бути використані наступні алгоритми:

Довжина Хемінга - це кількість позицій, на які відповідні символи різні. Іншими словами, вона вимірює мінімальну кількість підстановок, необхідних для зміни одного рядка в інший, або мінімальну кількість помилок, які могли б перетворити один рядок в інший. У більш загальному контексті відстань Хеммінга є однією з кількох рядкових метрик для вимірювання відстані редагування між двома послідовностями. Він названий на честь американського математика Річарда Хеммінга.[2]

Відстань Левенштейна між двома словами є мінімальним числом однозначних редагувань (вставок, вилучень чи підстановок), необхідних для зміни одного слова в інше. Він названий на честь радянського математика Володимира Левенштейна.[3]

Відстань Яро – Вінклера - це рядкова метрика, яка вимірює відстань редагування між двома послідовностями. Це варіант, запропонований у 1990 році Вільямом Е. Вінклером на підставі метрики дистантності Яро.[4]

Відстань Дамерау - Левенштейн (названа на честь Фредеріка Дж. Дамерау та Володимира І. Левенштейна) - це рядкова метрика для вимірювання відстані редагування між двома послідовностями. Неофіційно відстань Дамерау - Левенштейн між двома словами - це мінімальна кількість операцій (що складається з вставок, вилучень або підставок одного символу або переміщення двох суміжних символів), необхідних для зміни одного слова в інше. Відстань Дамерау - Левенштейн відрізняється від класичної відстані Левенштейна тим, що включає транспозиції до своїх допустимих операцій на додаток до трьох класичних операцій редагування одного символу (вставки, видалення та заміни)[5][6][7]

Середня відстань може бути обчислена за формулою

$$Adist = \sum_{i=1}^{4} Dist_i,$$

Де $Dist_i$відстані Хеммінга, Левенштейна, Дамерау – Левенштайна, Яро – Вінклера.

В результаті завдяки комбінації даних методів результати стають достовірнішими і виконання порівняння стає більш точним.

4. Проведення тестів

Загалом є багато важливих програм які базуються на вимірах ЕКГ. Наприклад зараз розробляються системи охорони приміщень – однією з аутентифікаційних методик якого є показання ЕКГ[8]. Також проводиться дослідження сну з використанням таких вимірів.[9]

Також групою китайських вчених було створено програмний продукт за допомогою якого лікарі могли

обмінюватись даними про ЕКГ своїх пацієнтів тим самим ділячись досвідом та спрощуючи взаємодію в команді кардіологів.[10]

Тож завдяки цим та ряду інших важливих досліджень ми отримали достатньо багато тестових даних – доступних для будь-яких досліджень.

За допомогою таких відкритих баз як MIT-BIH Arrhythmia databaseбули проведені дані тестування.

Починаючи з 1980 р. Цей набір даних використовується для базових досліджень серцевої динаміки приблизно в 500 місцях у всьому світі.[11]

Основною складністю при проведенні тестування був вибір алфавіту для його проведення. Оскільки діапазон значень для ЕКГ в тестових наборах був від 800 до 1200 при виборі англійського алфавіту маємо дуже багато схожих ланцюгів і загалом є значна кількість повторюваних літерпоказано на рис 1. Відповідно до цього всі було знайдено аномалії в звичайних значеннях ЕКГ.

В зв’язку з цим була взята послідовність символів Юнікоду 192-292 і результат продемонстрований на рис. 2.

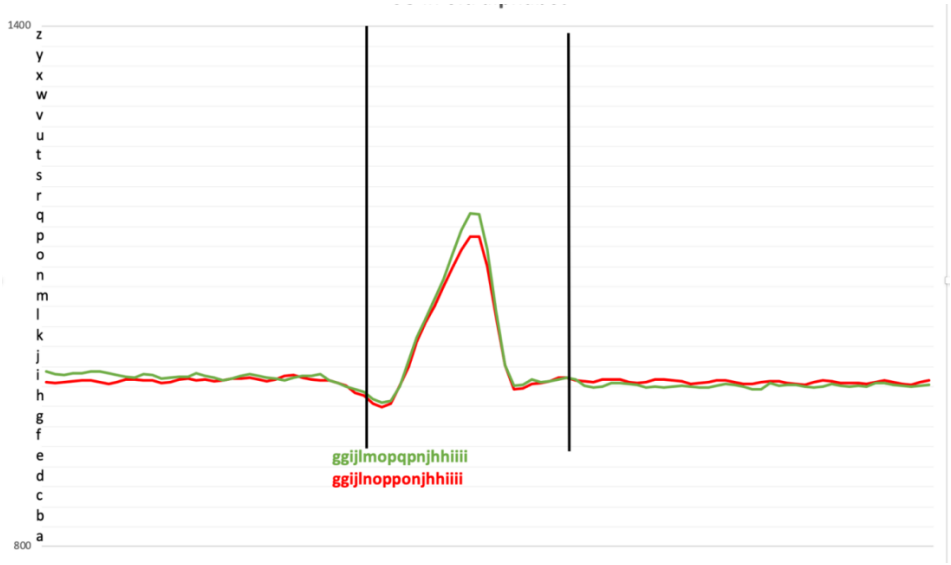


Рис. 5 Електрокардіограма при старому алфавіті

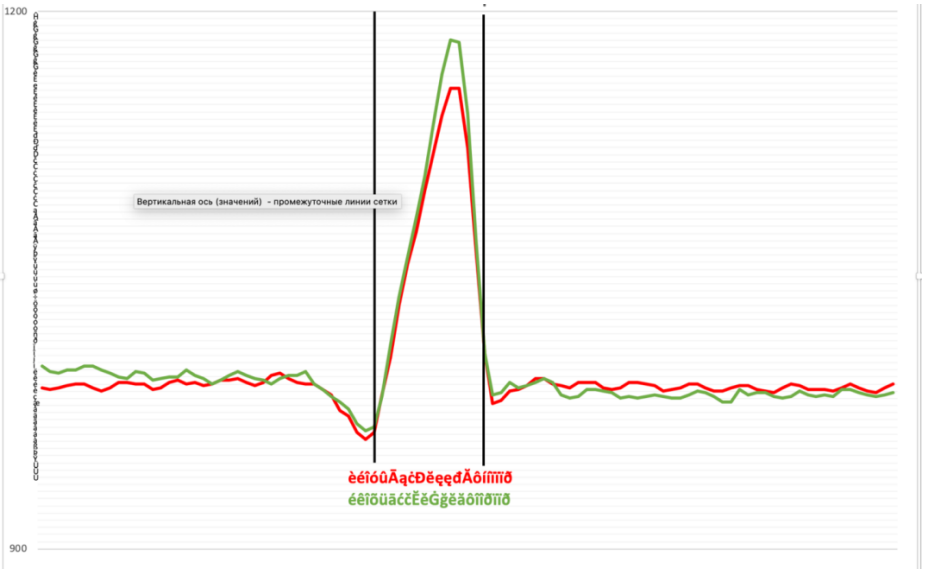


Рис. 6 Електрокардіограма при новому алфавіт

Висновок

Результати тестів довели дієздатність даного алгоритму для знаходження аномалій в електрокардіограмі. Велику роль при роботі з даним методом відіграє науковець,який шляхом постійного тестування і аналізу результатів підбирає потрібні коефіцієнти для алгоритмів та алфавіту.

В нашому випадку при проведених нами тестуваннях можна зробити висновок що звичайних алфавітів для обраної предметної області недостатньо і їх потрібно розширювати.

Також згодом потрібно розширити тестову базу для збільшення кількості шаблонів для аномалії.

Список літератури

1. Baklan, I., Mukha, I., Oliinyk, Y., Lishchuk, K., Nedashkivsky, E., & Gavrilenko, O. (2019, January). Anomalies Detection Approach in Electrocardiogram Analysis Using Linguistic Modeling. In International Conference on Computer Science, Engineering and Education Applications (pp. 513-522).

2. Hamming, R. W. (April 1950). "Error detecting and error correcting codes" (PDF). The Bell System Technical Journal. 29 (2): 147–160. doi:10.1002/j.1538-7305.1950.tb00463.x. ISSN 0005-8580.

3. Влади́мир И. Левенштейн (1965). Двоичные коды с исправлением выпадений, вставок и замещений символов [Binary codes capable of correcting deletions, insertions, and reversals]. Доклады Академий Наук СССР (in Russian). 163 (4): 845–8. Appeared in English as: Levenshtein, Vladimir I. (February 1966). "Binary codes capable of correcting deletions, insertions, and reversals". Soviet Physics Doklady. 10 (8): 707–710. Bibcode:1966SPhD...10..707L.

4. Jaro, M. A. (1989). "Advances in record linkage methodology as applied to the 1985 census of Tampa Florida". Journal of the American Statistical Association. 84 (406): 414–20. doi:10.1080/01621459.1989.10478785.

5. Levenshtein, Vladimir I. (February 1966), "Binary codes capable of correcting deletions, insertions, and reversals", Soviet Physics Doklady, 10 (8): 707–710

6. Damerau, Fred J. (March 1964), "A technique for computer detection and correction of spelling errors", Communications of the ACM, 7 (3): 171–176, doi:10.1145/363958.363994

7. The method used in: Majorek, Karolina A.; Dunin-Horkawicz, Stanisław; et al. (2013), "The RNase H-like superfamily: new members, comparative structural analysis and evolutionary classification", Nucleic Acids Research, 42 (7): 4160–4179, doi:10.1093/nar/gkt1414, PMC 3985635, PMID 24464998.

8. Hsu, C. Y., Hardt, M., & Hardt, M. (2019). Linear Dynamics: Clustering without identifi-cation. arXiv preprint arXiv:1908.01039.

9. Sun,H., Ganglberger, W., Panneerselvam, E., Leone, M. J., Quadri, S. A. Q., Goparaju, B., ... & Westover, M. B. (2019). Sleep Staging from Electrocardiography and Respiration with Deep Learning. arXiv preprint arXiv:1908.11463.

10. Ding, Z., Qiu, S., Guo, Y., Lin, J., Sun, L., Fu, D., ... & Lv, T. (2019). LabelECG: A Web-based Tool for Distributed Electrocardiogram Annotation. arXiv preprint arXiv:1908.06553.

11. P.Sellers, The theory and computation of evolutionary distances: Pattern recognition, J. Algorithms 1 (1980) 359–373

УДК 004.942

ДЖУРА Р.С.
ЖДАНОВА О.Г.

МОДЕЛЮВАННЯ СЛАБКОСТРУКТУРОВАНИХ СИСТЕМ З ВИКОРИСТАННЯМ КОГНІТИВНОГО ПІДХОДУ

У роботіпредставлено огляд існуючих уявлень про нечіткі когнітивні карти та розглянуто підходи у формулюванні нечітких когнітивних карт. Коротко представлено опис та побудову, а також представлено ідеї моделювання нечітких когнітивних карт. Наведено різні типи та математичний опис нечітких когнітивних карт.Показано, яким чином можна

використовувати нечіткі когнітивні карти та метод імпульсного моделювання для аналізу поведінки слабкоструктурованих систем. Наведено приклад практичного застосування нечітких когнітивних карт для вирішення деякої економічної задачі.

Ключові слова: нечіткі когнітивні карти, когнітивне моделювання, імпульсний метод, імпульсне моделювання.

This article examines the existing notion of fuzzycognitive maps and considers some approaches in the formulation of fizzy cognitive maps. Brief description and construction are presented, as well as some ideas of modeling of fizzy cognitive maps are presented. In this area, mainly, the representation, construction and use of fuzzy cognitive maps were studied. In this article is shown how fuzzy cognitive maps and impulse modeling can be used to analyze the behavior of weakly structured systems. An example of the practical use of fuzzy cognitive maps to solve some economic problem.is given.

Keywords: fuzzy cognitive maps, cognitive modeling, impulse method, impulse modeling.

1. Вступ

Слабкоструктуровані системи – це такі системи, взаємозв’язки між компонентами яких важко оцінити звичайними числовими методами, що ускладнює процес їх аналізу та дослідження. Одним із методів, що дозволяє дослідити та проаналізувати поведінку таких систем є метод когнітивного моделювання.

Методика нечітких когнітивних карт – це символічне подання для опису та моделювання складної системи. Нечіткі когнітивні карти описують різні аспекти поведінки складної системи з точки зору концептів (факторів); кожне концепт (фактор) являє собою стан або характеристику системи, і ці концепти взаємодіють один з одним, що відображає динаміку системи. Когнітивні картиілюструють всю систему графічно, що показує причину та наслідки,взаємовплив концептів (факторів), і є простим способом символічного опису моделі системи та поведінки, використовуючи накопичені знання про систему. Нечітка когнітивна карта об'єднує накопичений досвід та знання про роботу системи, використовуючи експертів, які знають роботу системи та її поведінку в різних обставинах. Крім того, нечіткі когнітивні карти використовують методи навчання, які впроваджені в теорію нейромереж, з метою підготовки нечіткої когнітивної карти та вибору відповідних ваг для взаємозв'язків факторів. Політолог Аксельродввів когнітивні карти для представлення соціальних наукових знань і описав методи, які використовуються для прийняття рішень в соціальних та політичних системах[1]. Потім Косько посилив потужність когнітивних карт, враховуючи нечіткі значення концептів когнітивної карти та нечітких ступенів взаємозв'язків між концептами[2-4]. Після цієї піонерської роботи нечіткі когнітивні карти привернули увагу вчених з багатьох областей і використовувалися для вирішення різних наукових проблем.

2. Представлення нечітких когнітивних карт

Нечіткі когнітивні карти –орієнтовані графи з ребрами взаємодії. Вони складаються з вузлів-концептів$C_i (i = 1, \dots, n)$ і взаємозв'язків e_{ij}між концептамиC_i і C_j. Нечіткі когнітивні карти моделюють динамічну складну систему як сукупність концептів та причинно-наслідкових зв'язків між концептами. Когнітивна карта, що зображена на рисунку 1, складається з п'яти вузлів-концептів.

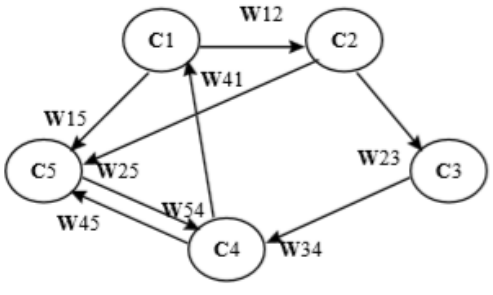


Рис.1

Взаємозв'язокe_{ij} між концептамиC_i і C_jхарактеризується вагою w_{ij}, що описує ступінь впливуодного концепту на інший. Ваги приймають значення в інтервалі[-1,1]. Знак ваги вказує на позитивну причинність $w_{ij} > 0$ між концептамиC_i і C_j, що означає,

що збільшення вартості концепту C_iпризведе до збільшення вартості концепту C_j і зменшення вартості концепту C_iпризведе до зниження вартості концепту C_j. Коли між двома концептами є негативна причинність, то $w_{ij} < 0$; збільшення першого концепту означає зменшення вартості другого концепту та зменшення концептуC_i викликає збільшення вартості C_j. Коли немає взаємозв'язку між концептами, то $w_{ij} = 0$.

Як правило, цінність кожного концепту обчислюється, враховуючи вплив інших концептів на конкретний концепт, застосовуючи наступне правило розрахунку:

$$x_i(t) = f(\sum_{j=1}^{n}{}_{j\neq i} x_j(t-1)w_{ji}), \quad (1)$$

де $x_i(t)$ є значення концепту C_i в час t, $x_j(t-1)$ це значення концепту C_j в час$t-1$, w_{ji}–це вага взаємозв'язку між концептом C_i і концептом C_j, а $f = \frac{1}{1-e^{-\lambda x}}$.

На кожному кроці часу значення для всіх концептів нечіткої когнітивної карти змінюються і перераховуються за рівнянням (1). Правило розрахунку для кожного етапу моделювання нечіткої когнітивної карти включає в себе розрахунок нових значень для всіх концептів. Воно складається з *n*-мірного вектору *X* який містить значення *n*-концептів, та матриці$W = [w_{ij}]_{1\leq i,j\leq n1}$якамістить значення ваг ребер для нечіткої когнітивної карти, де розмірність матриці дорівнює кількості різних концептів, які містить карта, тобто *n*. Таким чином, вектор нового стану*X*у момент часу *t* розраховується наступним чином:

$$X(t) = f(W^T X(t-1)). \quad (2)$$

Як правило, нечіткі когнітивні карти можна навчати, використовуючи алгоритми навчання аналогічно теорії нейронних мереж. Алгоритми навчання, як правило, належать до алгоритмів «навчання без учителя». Під час тренувального періоду ваги ребер нечіткої когнітивної карти змінюються за законом вивчення першого порядку, який базується на кореляційному чи диференційованому законі про навчання Гебба:

$$w'_{ij} = -w_{ij} + x'_{ix'_j}. \quad (3)$$

Так $x'_i x'_j > 0$ якщо значення концептів C_i та C_j рухаються в одному напрямку, та $x'_i x'_j < 0$ якщо значення концептів C_i та C_j рухаються в протилежних напрямках; отже, концепти, які, як правило, є позитивними чи негативними одночасно, матимуть сильні додатні ваги, тоді як ті, що мають тенденцію до протилежного, матимуть сильнівід'ємні ваги.

Інше представлення нечітких когнітивних карт, яке виходить за межі початкового визначення, яке не дозволяє впливати на самі концепти, розроблено Бартом Коско[2-3]. Тепер концепт може враховувати власне минуле значення з масою w_{ii}:

$$x_i(t) = f\left(\sum_{j=1, j\neq i}^{n} x_j(t-1)w_{ji} + w_{ii}x_i(t-1)\right). \quad (4)$$

де x_i– значення концепту C_i в час t, $x_i(t-1)$– значення концептуC_iв час $t-1$, $x_j(t-1)$– значення концепту C_j в час $t-1$, w_{ji}- вага ребра зв'язку міжC_j та C_i, w_{ii}– вага, з якою попереднє значення концепту бере участь у розрахунку нового, а *f*– порогова функція.

Більш стисло можна записати наступним чином:

$$X(t) = f\big(W^T X(t-1)\big), \quad (5)$$

де матриця *W* має ненульові діагональні елементи. Відмітимо, що у рівнянні (2) всі діагональні елементи цієї матриці були нульовими. Приклад нечіткої когнітивної карти, концепти якої враховують своє попереднє значення, зображено нижче на рисунку 2.

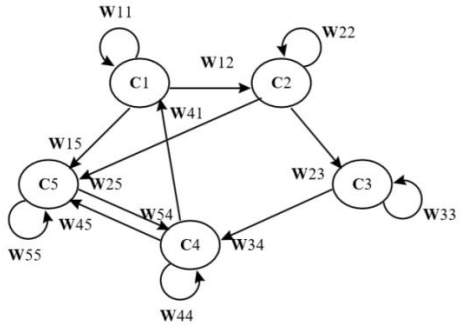


Рис. 2

Цей тип нечітких когнітивних карт повинен розглядатися з точки зору його стабільності. Існує висока ймовірність того, що її буде приведено до граничного циклу або до нестабільного стану, за рахунок того що постійно буде збільшуватися значення

кожного концепту внаслідок впливу, який має кожнийконцепт сам на себе. Нечіткі когнітивні карти такого типу можуть бути корисними для опису поведінки деяких спеціальних систем за певних обставин.

Існує також тип нечітких когнітивних карт, в якому припускається, що кожний концепт має зовнішній вхід, який впливає на нього з певною вагою і береться до уваги під час розрахунку (рис. 3). Цей тип нечітких когнітивних карт дуже схожий з періодичними нейронними мережами[5-6], які використовуються в якості моделей, параметри яких мають відповідати вхідним / вихідним даним, мінімізуючи функцію витрат[7]. Розглядається нечітка когнітивна карта з припущенням про наявність зовнішнього входу до кожного вузла. Значення кожного вузла карти обчислюється за наступним рівнянням:

$$x_i(t+1) = f\left(Ax_i(t) + Bu_i(t)\right). \qquad (6)$$

Для когнітивної карти, зображеної на рисунку 3, виконується:

$$\begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ x_3(t+1) \\ x_4(t+1) \\ x_5(t+1) \end{bmatrix} = f\left(A \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \end{bmatrix} + B \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \\ u_4(t) \\ u_5(t) \end{bmatrix} \right),$$

де $x_i(t)$– це стан концепту C_i в момент часу t, $x_i(t+1)$– це стан концепту C_i на момент часу t + *1*,$u_i(t)$– це значення зовнішнього впливу на концепт C_i в момент часу *t*,*A*– матриця значень взаємовпливів між концептами, *B*–матриця значень зовнішніх впливів на концепти, які мають такі значення:

$$A = \begin{bmatrix} 0 & 0 & 0 & w_{41} & 0 \\ w_{12} & 0 & 0 & 0 & w_{51} \\ 0 & w_{23} & 0 & 0 & 0 \\ 0 & 0 & w_{34} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} w_1 & 0 & 0 & 0 & 0 \\ 0 & w_2 & 0 & 0 & 0 \\ 0 & 0 & w_3 & 0 & 0 \\ 0 & 0 & 0 & w_4 & 0 \\ 0 & 0 & 0 & 0 & w_5 \end{bmatrix}.$$

Новий стан в момент часу*t+1* розраховується із зовнішнім впливом $u_i(t)$ з відповідною вагою w_i і впливом інших концептів.

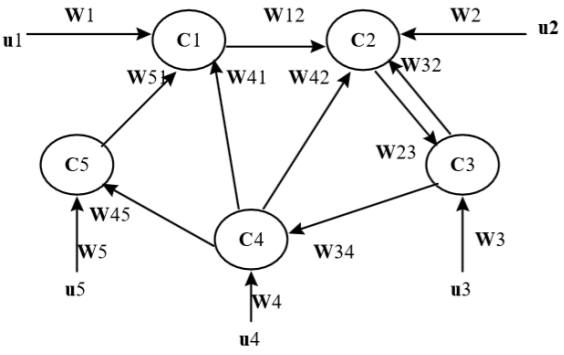


Рис. 3

3. Практичне застосування нечітких когнітивних карт

У наш час метод когнітивного моделювання є розповсюдженим і застосовується досить широко. Він може допомогти у вирішенні багатьох проблем, зокрема в економіці, політиці, освіті, техніці тощо. Для прикладу візьмемо просту економічну задачу, зображену на рисунку 4, яка показує взаємозв'язки між попитом, пропозицією, виробниками та споживачами. Завдяки цьому прикладу стане зрозумілим, як будується нечітка когнітивна карта, як вибираються поняття і як нечітка когнітивна карта моделює та контролює процес.

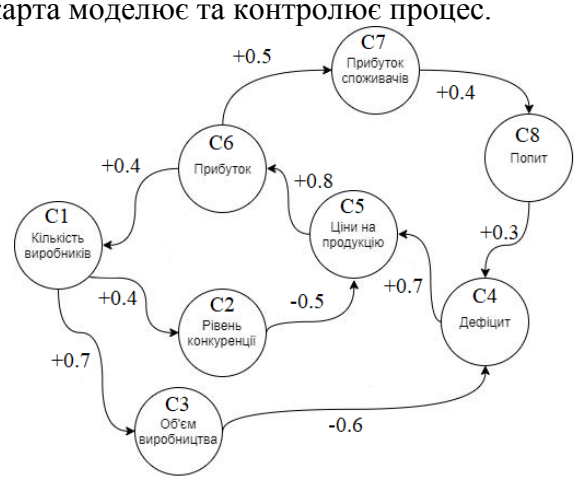


Рис. 4

Взаємозв'язки між концептами в розглянутій системі можна представити у вигляді наступної матриці суміжності:

Табл. 1. Матриця суміжності

	C₁	C₂	C₃	C₄	C₅	C₆	C₇	C₈
C₁	0	0,4	0,7	0	0	0	0	0
C₂	0	0	0	0	-0,5	0	0	0
C₃	0	-0,6	0	0	0	0	0	0
C₄	0	0	0	0	0,7	0	0	0
C₅	0	0	0	0	0	0,8	0	0
C₆	0,4	0	0	0	0	0	0,5	0
C₇	0	0	0	0	0	0	0	0,4
C₈	0	0	0	0,3	0	0	0	0

З'єднання між концептами:

– збільшення (зменшення) кількості виробників призводить до суттєвого збільшення (зменшення) об'єму виробництва;

– збільшення (зменшення) кількості виробників призводить до середнього збільшення (зменшення) рівню конкуренції;

– збільшення (зменшення) об'єму виробництва призводить до суттєвого зменшення (збільшення) дефіциту;

– збільшення (зменшення) рівню конкуренції призводить до середнього зменшення (збільшення) ціни на продукцію;

– збільшення (зменшення) дефіциту призводить до суттєвого збільшення (зменшення) ціни на продукцію;

– збільшення (зменшення) ціни на продукцію призводить до сильного збільшення (зменшення) прибутку;

– збільшення (зменшення) прибутку призводить до середнього збільшення (зменшення) кількості виробників;

– збільшення (зменшення) прибутку призводить до середньогозбільшення (зменшення) прибутку споживачів;

– збільшення (зменшення) прибутку споживачів призводить до середньогозбільшення (зменшення) попиту;

– збільшення (зменшення) попиту призводить до слабкогозбільшення (зменшення) дефіциту.

Значення кожного впливу визначаються експертами, які розробили карту. Експерти спостерігали вплив кожного концепту на інші у реальній експериментальній системі та призначили лінгвістичні значення ваги для кожного з'єднання, які трансформувалися в нечіткі значення.Жоден з концептів не можна очевидно виділити як впливовий, оскільки всі концепти мають як вихідні впливи на інші концепти, так і вхідні впливи з боку інших концептів. Розглянувши дану задачу більш детально, можна дійти висновку що концепти C_3, C_4, C_5 є такими, на які можна вчинити зовнішній, так би мовити «штучний» вплив.

Дану задачу можна описати наступним рівнянням:

$$\begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ x_3(t+1) \\ x_4(t+1) \\ x_5(t+1) \\ x_6(t+1) \\ x_7(t+1) \\ x_8(t+1) \end{bmatrix} = f\left(A \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \\ x_6(t) \\ x_7(t) \\ x_8(t) \end{bmatrix} + B \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \\ u_4(t) \\ u_5(t) \\ u_6(t) \\ u_7(t) \\ u_8(t) \end{bmatrix} \right),$$

де $x_i(t)$– це стан концепту C_i в момент часу t, $x_i(t+1)$– це стан концепту C_i на момент часу $t + 1$,$u_i(t)$– це значення зовнішнього впливу на концепт C_i в момент часу t,A– матриця значень взаємовпливів між концептами, B–матриця значень зовнішніх впливів на концепти, які мають такі значення:

$$A = \begin{bmatrix} 0 & 0.4 & 0.7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.5 & 0 & 0 & 0 \\ 0 & -0.6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.8 & 0 & 0 \\ 0.4 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.4 \\ 0 & 0 & 0 & 0.3 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 00000000 \\ 00000000 \\ 00100000 \\ 00010000 \\ 00001000 \\ 00000000 \\ 00000000 \\ 00000000 \end{bmatrix}.$$

Можна припустити, що зовнішні впливи на концепти нечіткої когнітивної карти виникають внаслідок перетворення вимірювань змінних реальної системи та / або їхніх бажаних значень. У випадку, якщо є менше, ніж n входів, то деякі рядки матриці A будуть нульовими, а керованість нечіткої когнітивної карти повинна бути перевірена відповідно до співвідношення матриць A та B.

4. Імпульсний метод

Одним із методів дослідження поведінки системи за допомогою нечітких когнітивних карт є імпульсний метод. Суть імпульсного методу полягає у збудженні (зміні значення) певного концепту (декількох концептів) протягом певного часу та відслідковуванні змін до яких призводить дане збудження у

інших концептах нечіткої когнітивної карти. Імпульсний метод дає змогу побачити значення будь-якого концепту нечіткої когнітивної карти в будь-який момент часу маючи на вхід початкове значення концептів на момент часу $t = 0$, поточне значення моменту часу t, матрицю взаємовпливів між концептами нечіткої когнітивної карти та величину збудження що відбувається у певних концептах. Зміна величини значення концепту називається імпульсом$P_j(t)$ і задається так:

$$P_j(t) = x_j(t) - x_j(t - 1).$$

Збудження певних концептів призводить до появи в них певного значення імпульсу, що в свою чергу призводить до появи імпульсу у суміжних концептах, на які впливає даний і т.п. В результаті збудження одного концепту нечіткої когнітивної карти призводить до, так би мовити, «ланцюгової реакції». Значення концептів починають змінюватися, що в свою чергу призводить до змін значень залежних концептів. Таким чином загальна картина яку відображає нечітка когнітивна карта може кардинально змінитися через збудження одного або декількох концептів.

Метод імпульсного моделювання дає нам змогу обчислити значення концептів нечіткої когнітивної карти в конкретний момент часу:

$$x_i(t + 1) = x_i(t) + \sum_{j=1}^{n} w_{ij} P_j(t).$$

Імпульсний метод може бути використаний для моделювання стану системи під впливом певних зовнішніх або внутрішніх процесів що призводять до зміни значення в одному або декількох концептах.

Моделювання на графовій моделі проводитьсяпокроково.Ці кроки і називають імпульсами або елементарними збудженнями. Суть даного процесу полягає у наступному: однійабо декільком із вершинам задається значення збудження, яке тягне за собою зміни у значеннях інших концептів нечіткої когнітивної карти, притому посилюючись або затухаючи. Значення у вершинах будуть змінюватися через кожен крок імітації *t*. Вершини, в яких відбувається збудження, зазвичай обираються таким чином, щоб це були впливові концепти, які мають вплив на інші концепти системи в той час, коли на них інші концепти системи не впливають. Також ці концепти обирають з умовою, що вплив на них зовні системи є можливим і не залежить від стану системи, адже в протилежному випадку подібний аналіз не призведе до корисних результатів через неможливість впливу на обрані концепти і досягнення отриманих результатів на практиці

Висновки

У даній роботі представлено різні види нечітких когнітивних карт та розглянуто різні види нечітких когнітивних карт. Деякі з представлених типівнечітких когнітивних карт мають математичне подання, дуже близьке до рекурентних нейронних мереж. Також було розглянуто спосіб аналізу та моделювання поведінки слабкоструктурованоїсистеми з використанням нечітких когнітивних карт та імпульсного моделювання.Крім того було наведено приклад застосування нечітких когнітивних карт для розв’язання проблем,які дозволяють відкрити нові напрямки використання нечітких пізнавальних карт в слабкоструктурованих системах.

Література

1. Axelrod, R., (1976). Structure of Decision: the Cognitive Maps of Political Elites, Princeton University Press, New Jersey.

2. Kosko, B. (1986). "Fuzzy Cognitive Maps," International Journal of Man-Machine Studies, 24, pp. 65 - 75.

3. Kosko, B. (1992). Neural Networks and Fuzzy Systems, Prentice-Hall, Upper Saddle River, New Jersey.
4. Kosko, B. (1997). Fuzzy Engineering, Prentice-Hall, Upper Saddle River, New Jersey.
5. Albertini, F., and E. Sontag (1994). "State Observability in Recurrent Neural Networks," Systems & Control Letters, 22, pp.235-244.
6. Albertini, F., and P.D. Pra. (1995). "Forward Accessibility for Recurrent Neural Networks," IEEE Transactions on Automatic Control, 40, no. 11, pp.1962-1968.
7. Sontag, E., and H. Sussmann. (1997). "Complete Controllability of Continuous-Time Recurrent Neural Networks," Systems & Control Letters, 30, pp.177-183.
8. Stylios, C.D., V. C. Georgopoulos, and P. P. Groumpos (1997a). "The Use of Fuzzy Cognitive Maps in Modeling Systems," ProF c. of 5th IEEE Med. Conf. on Control & Systems, Paphos, Cyprus, paper 67
9. Stylios, C.D., V. C. Georgopoulos, and P.P. Groumpos (1997b). "Introducing the Theory of Fuzzy Cognitive Maps in Distributed Systems," Proc. of the 12th IEEE Int. Symposium on Intelligent Control, Istanbul, pp. 55-60.
10. Stylios, C.D., and P. P. Groumpos (1998). "Mathematical Methodology of Fuzzy Cognitive Map for Complex Systems" Technical Paper LAR-TP-2.
11. Stylios, C.D., and P. P. Groumpos (1999). "Fuzzy Cognitive Maps: A model for Intelligent Supervisory Control Systems," Computers in Industry (accepted for publication.

УДК 004.93(015.7)

ЛІПСЬКА В.О.

СЕРВІС ДЛЯ ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У даній роботі розглянутий концепт сервісу, що розв’язує проблему певної множини задач однієї з важливих галузей реалізації програмного забезпечення - тестування. Були розглянуті вже існуючі продукти, їхні цільові проблеми, які вони вирішують, та висвітлені як переваги, так і типи потреб, які вони не охоплюють. Таким чином був сформований об’єкт дослідження та продемонстровано алгоритм дій сервісу, що розглядається у цій роботі. Ключові слова: програмне забезпечення, тестування, регресивне тестування, пропускна здатність.

This paper discusses the concept of a service that solves the problem of anumber of tasks in one of the important areas of software implementation - testing. Existing products have been considered, their target problems they solve, and the benefits as well as the types of needs they do not cover. In this way, the object of study was formed and the algorithm of the service actions discussed in this work was demonstrated. Keywords: software, testing, regressivetesting, permissioncapacity.

Вступ

Кожен створений продукт розробляється для вирішення певної проблеми чи задач, що являють собою множину, елементи якої об’єднані типовими характеристиками, тому закономірним та логічним кроком після досягнення початку роботи мінімальних функцій стає постановка питання про перевірку на дієздатність програмного застосунку. У розрізі програмного забезпечення важливо щоб результат був не лише коректний, а й задовольняв усі не менш важливі вимоги, такі як швидкість відповіді, навантаження (пропускна здатність), модульність та інші. Також варто пам’ятати, що тестування програмного забезпечення одна з найважливіших складових у розробці та реалізації концепту початкової ідеї та має бути присутня на кожному етапі, починаючи від перших стадій (не має значення яка методологія керування використовується та які ресурси задіяні), тож автоматизація цієї галузі економить неймовірну кількість часу.

Варіативність архітектури проектування програмного забезпечення досить велика, що досить часто зумовлено потребами бізнес-логіки, тому зі зростанням кількості функцій чи акторів в діаграмі прецедентів, тестування може експоненціально зростати за рівнем складності та перетворюватися на ієрархічну систему з вкладеннями та додатковими запитами на кроках без прямої їх потреби при перевірці обраного блоку.

Не зважаючи на те, що вже існує досить багато потужних інструментів, що спрощують процеси роботи при перевірці інженерних рішень, існує ще ряд задач, які вимагають або мануального тестування, або написання своїх проектів чи фреймворків (коли мова йде про об'ємні проекти) для покращення та спрощення процесу перевірки дієздатності. Мануальне тестування багате на ряд недоліків, таких як трудомісткість, людський фактор, важкість контролю, ресурс часу та ліміт доступності (інколи потрібно бути впевненим, що рішення працює щохвилини чи перевіряти щогодини певні задачі в автоматизованому режимі). Також конкурентне програмування в сучасних реаліях займає дуже важливе місце в розробці майже що будь-якого рішення, а такі речі не лише тестувати, а й відслідковувати при створенні складно, тому для аналізу та покращення показників, крім основної мети процесу, потрібні чіткі цифри та результати. Якщо ж мова йде про написання своїх програм, то найчастіше дана задача вирішується шляхом створення власних скриптів, використання сторонніх бібліотек, а саме їх інтеграція в свої (до прикладу, для .NET одна з кращих - "BenchmarkDotNet"[1] розробника Андрія Акіншіна) чи використання інших автоматизованих застосунків.

Аналіз відомих рішень та підходів

1. Postman[2] - безкоштовне популярне зручне рішення, що має ряд переваг, а саме: зручний інтерфейс для задання параметрів та заголовків, збереження колекцій запитів та багато іншого, проте для певної множини задач, яка розглядається в даній роботі, не вистачає послідовного задання запитів з підстановкою динамічних параметрів.

2. Selenium [3]- чудовий проект, який допомагає у тестуванні клієнтської частини, шляхом імітації людської поведінки, проте немає можливості тестування прикладного програмного інтерфейсу.

3. Katalon[4] - потужний гнучкий помічник при створенні та реалізації продуктів, який дозволяє тестувати прикладні програмні інтерфейси, веб-інтерфейси, мобільні додатки, десктоп-додатки, проте немає безкоштовної версії.

4. JMeter[5] - програмне забезпечення з відкритим кодом, яке дає змогу перевіряти програми на пропускну здатність, дає можливість діставати дані з запитів, що необхідні, а також вміє працювати з csv-файлами, але тут також є такі недоліки: досить високий поріг входу, використання занадто великої кількості пам'яті, які навіть спричиняють помилки при роботі.

Таким чином усі вище описані рішення чудові і потужні для свого набору задач і їх дійсно зручно застосовувати для потреб, що вони вирішують, проте предметом дослідження була проблема реалізації такого сервісу, що міг би тестувати прикладні інтерфейси, динамічно підставляючи параметри з одних запитів в інші, та маючи можливість користувачеві формувати з поодиноких запитів модулі чи ланцюги, що пов'язані між собою, які необхідні для тестування системи. Також у цьому сервісі важливою є функція розкладу запусків без втручання користувача та повідомлення за необхідності за обраним каналом зв'язку. Реалізація цього проекту допомагає ще більше автоматизовуватиперевірки та надає можливість легко контролювати стан в будь-який час, пришвидшує регресивне тестування після оновлень чи після деплою на новому оточенні та уникає для цієї множини потреб інтеграції декількох сервісів замість одного.

Алгоритм

Тож ця робота присвячена вирішенню проблеми такої множини потреб: проекти, що використовують отримані результати як параметри в послідуючих діях, прикладом такого може бути ідентифікація користувача за токеном доступу [1] при запиті до прикладного програмного інтерфейсу; що потребують системного тестування [2] за особливим розкладом та миттєвого повідомлення відповідальних за обраним каналом зв'язку; що приділяють увагу регресивному тестуванню [3] (зазвичай контрольні перевірки для оновлення збірок до публікації); що прагнуть перевіряти пропускну здатність та ті що прагнуть до максимальної автоматизації процесу тестування. Також однією з важливих задач була поставлена - модульність, робота має бути легко інтегрована в інші системи за потреби, тому одна із форм представлення, що була обрана - прикладний програмний інтерфейс.

Кожен запит уособлює в собі мікросистему вхідних параметрів, очікуваний стан проекту та вихідний результат. Певні запити важливо поєднувати в послідовні дії та зберігати отримані дані для застосування їх у інших запитах. Таким чином дане рішення надає змогу створювати системні ланцюги станів - запитів - проекту, що тестується, в залежності від вхідних даних та їх динамічної підстановки від отриманих результатів запиту до вхідних параметрів чи заголовків запиту та їх запам'ятовування в межах одного ланцюга. Після одноразового налаштування програмне забезпечення, що тестується, буде перевірятися автоматично за розкладом або за запуском, що визначається в залежності від потреб.

.

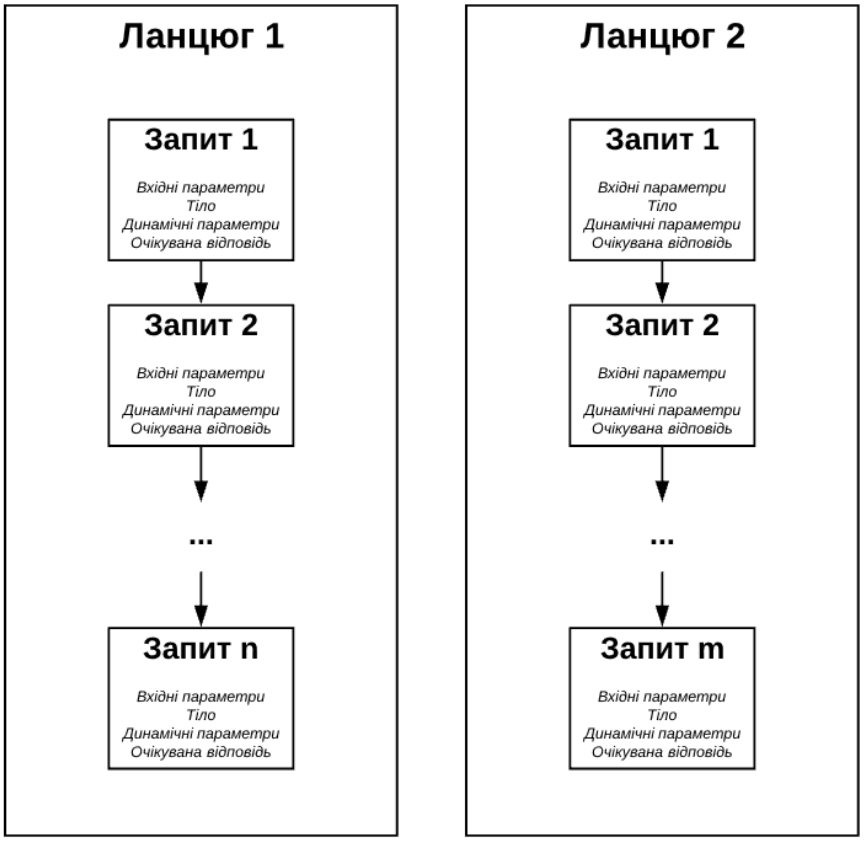


Рис.1. Спрощена схема ланцюгів запитів

Таким чином для застосунку створюються необхідні тести у вигляді ланцюгів запитів, що запускаються за розкладом або миттєво в залежності від потреб в потрібній кількості, виконуючи підстановку динамічних параметрів "на льоту", якщо такі є, вказуючи статичні параметри, заголовки та тіло. Після кожного виконання кроку результат ієрархічного блоку порівнюється з очікуваною відповіддю та кожна відповідь може заповнити динамічні параметри наступної складової тесту, яка вже в свою чергу буде оброблятися під час виконання свого запиту, сформувавши його перед цим. Варто відзначити, що під час процесу збирається статистика та необхідні результати для подальшої обробки користувачем, а також є можливість повідомлення користувача відразу по завершенню.

Такі схеми надають змогу вільного функціонування різних тестів в будь-який час доби, а також надають можливість виконання перевірки на предмет існування чи дієздатності програмного застосунку.

Розглянемо приклад обробки ланцюга: припустимо, що у нас був набір таких параметрів для запиту *A* - *b1, b2, b3,* очікувана відповідь *c1*, а за типом - юніт-тест - просто потрібно перевірити на коректність, за ним слідує запит *B*. Після виконання запиту *A*, відповідь співпала, тому переходимо до запиту *B*, у якого параметри такі: *b4, b5, b6*, а динамічні мають бути з попередньої відповіді *c1*, тому *c1.b7, c1.b8* - також використовується, про що попередньо

вказано в самих налаштуваннях, в той час як очікувана відповідь запиту *B* - c2, потім виконується запит, відповідь перевіряється на предмет коректності і надалі усе аналогічно повторюється до останньої ланки.

Такий алгоритм дозволить застосування гнучкості у перевірці програмного забезпечення, врятує багато часу після першого налаштування, надаватиме певний необхідний обсяг даних за результатами та зможе інформувати користувача про усі випадки в будь-який час за потреби.

Висновки

У ході даної роботи було виявлено, що хоча галузь тестування програмного забезпечення досить широка та добре оснащена інструментами, що спрощують процеси, потреби можуть бути на стільки різними, що потрібно писати власний фреймворк для тестування. Тож зіткнувшись з такою проблемою, прийшла ідея створити своє API, що зможе за розкладом ставити послідовні запити, підставляючи “на льоту” параметри та повідомляючи користувача про проблему.

Перелік посилань

1. AkinshinA. BenchmarkDotNet [Електронний ресурс] / AndreyAkinshin – Режим доступу до ресурсу: https://benchmarkdotnet.org/.
2. Postman[Електронний ресурс] – Режим доступу до ресурсу: https://www.postman.com/.
3. Selenium[Електронний ресурс] – Режим доступу до ресурсу: https://www.selenium.dev/.
4. Katalon[Електронний ресурс] – Режим доступу до ресурсу: https://www.katalon.com/.
5. JMeter [Електронний ресурс] – Режим доступу до ресурсу: https://jmeter.apache.org/.
6. Lock A. ASP.NET Core in Action / Andrew Lock. – United States of America: Manning Publications Co., 2018. – 682 c. – (MANNING SHELTER ISLAND). – (9781617294617; 2147483647).
7. Kshirasagar N. SOFTWARE TESTING AND QUALITY ASSURANCE Theory and Practice [Електроннийресурс] / N. Kshirasagar, T. Priyadarshi // 2. – 2008. – Режимдоступудоресурсу: https://www.softwaretestinggenius.com/download/staqtpsn.pdf.
8. AmmannP. INTRODUCTIONTOSOFTWARETESTING [Електронний ресурс] / P. Ammann, J. Offutt // 1. – 2008. – Режим доступу до ресурсу: http://www.cse.hcmut.edu.vn/~hiep/KiemthuPhanmem/Tailieuthamkhao/Introduction%20to%20Software%20Testing.pdf.

УДК. 519.7.007.004.02

КУХАРЕЦЬ Л.С.

ПОБУДОВА УНІВЕРСАЛЬНОГО АЛГОРИТМУ НА ОСНОВІ МОДЕЛІ ГРУПОВОГО ПЕРЕМІЩЕННЯ РЕЙНОЛЬДСА

У роботі описано три основні правила моделі групового переміщення Рейнольдса (1987): збереження дистанції, гуртування та рівняння. Колективний рух є результатом взаємодії відносно простої поведінки окремих боідів. Сукупний вектор руху кожного об’єкта є сумою векторів, отриманих у результаті застосування основних та додаткових правил моделі. Наведено блок-схеми універсального алгоритму. В залежності від постановки задачі алгоритм моделі можна доповнити. Подане рішення є актуальним для розробки гри як з двовимірним, так і з тривимірним середовищем.

КЛЮЧОВІ СЛОВА: модельРейнольдса, боід, алгоритм, групове переміщення

The paper describes three basic rules of Reynolds' group motion model (1987): separation, alignment, and cohesion. Collective movement is the result of simple individual boids be havior interaction. The aggregate motion vector of each object is the sum of vectors obtained by applying basic and additional rules of the model. The block diagrams of the universal algorith mare given. The algorithm can be supplemented depend in gon the formulation of the problem. The given solution is relevant for game development in both two-dimensional and three-dimensional environment.

KEYWORDS: Reynolds model, boid, algorithm, group movement

1. Вступ

Рух групами є природним для великої кількості тварин. Наприклад, риби та птахи зазвичай переміщуються зграями, вівці переходять між пасовищами у стаді, комарі формують рій, і навіть мурахи шукають їжу не поодинці. Відтворення подібного руху за допомогою універсального алгоритму стало одним із завдань комп'ютерної анімації та штучного життя [1]. Комп’ютерну модель групового переміщення живих істот було розроблено у 1987 Рейнольдсом [1, 2]. Алгоритм імітує рух птахів та риб. Складна сукупна модель поведінки вибудовується завдяки простим рухам окремих об’єктів групи, так званих боідів. Алгоритм базується на трьох правилах: кожен боід повинен уникати скупчень, має підлаштовувати власні рухи під поведінку сусідів, і водночас мусить не покидати зграю. Пізніше загальний алгоритм було модифіковано для вирішення деяких проблем штучного інтелекту [3]. Наприклад, в моделі керованої поведінки Рейнольдса 1999 року додано можливість рухатись до цілі та уникати перешкоди [2].

Модель боідів часто використовується в комп'ютерній графіці для відтворення руху об'єктів у групі. Наприклад, у 1998 році компанія ValveVideoGame застосувала відповідні алгоритми у відеогрі Half-Life для надання реалістичної поведінки зграї птахоподібних істот. Окрім ігрової індустрії, модель використовується в комп'ютерній анімації для кінофільмів [2]. Яскравий приклад її застосування – у художньому фільмі про Бетмена 1992 року для імітації польоту летючих мишей.

Модель боідів залишається популярною в ігровій індустрії і у наш час, тому розуміння її базових принципів є важливим для розробників у даній сфері.

2. Основні правила моделі

Модель Рейнольдса 1987 року базується на взаємодії боідів [1, 4]. Кожен об’єкт зграї рухається, зважаючи на швидкості та позиції сусідніх об’єктів. Три основні правила алгоритму:

- Збереження дистанції (рис. 1): боіди повинні перебувати один від одного на відстані, що є не менше мінімально можливої, задля уникнення скупчень та запобігання зіткненням.
- Рівняння (рис. 2): боід повинен рівнятися на положення та швидкість сусідніх об’єктів, враховуючи ці показники при розрахунку наступної позиції для імітації слідування.
- Гуртування (рис. 3): боід має тенденцію рухатись до середньої точки серед положень сусідів.

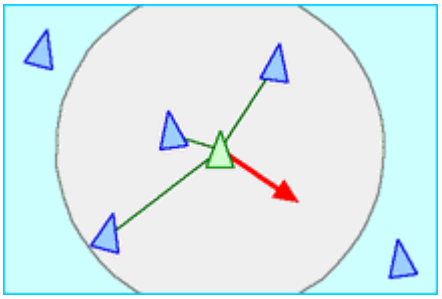


Рис. 1. Правило збереження дистанції

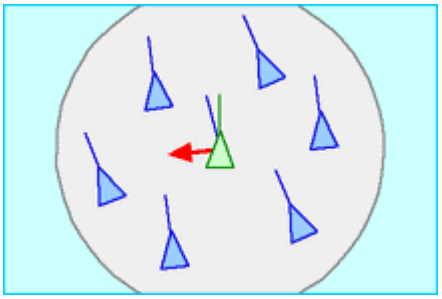


Рис. 2. Правило рівняння

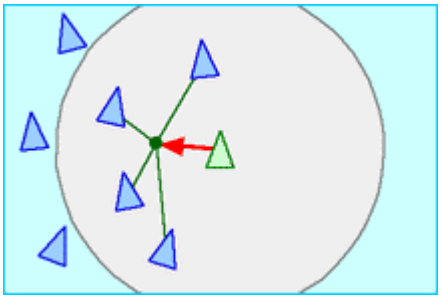


Рис. 3. Правило гуртування

У 1999 році базову модель Рейнольдса було розширено. Правила керованої поведінки стали більш індивідуальними, а боїди отримали можливість слідувати конкретним цілям та адаптуватися до умов середовища.

Деякі покращення моделі:

- Уникнення перешкод (рис. 4): боїди уникають сторонніх об'єктів, що розміщені на їх шляху.
- Слідування за лідером (рис. 5): деякі об’єкти групи починають слідувати за боїдом, що визначений як їх лідер.

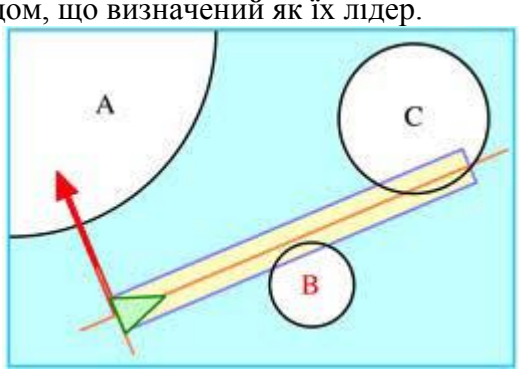


Рис. 4.Боїди оминають перешкоди

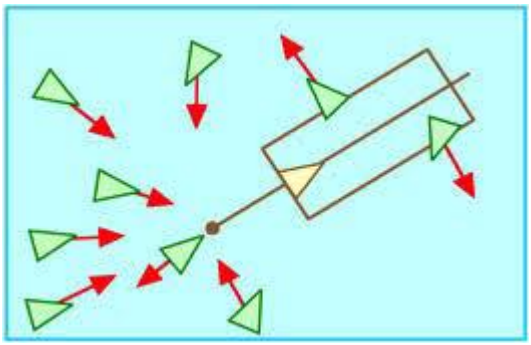


Рис. 5.Боїди слідують за лідером

3. Побудова алгоритму

Нижче наведено загальні правила та псевдокод для реалізації моделі Рейнолдса у грі. Алгоритм однаково добре працює у двовимірному та тривимірному просторах.

В першу чергу необхідно задати початкову позицію кожному боїду. Найкраще розмістити їх за областю видимості, щоб при ініціалізації гри об’єкти не з'являлися нізвідки, а прямували до центру екрану з-за його меж.

З певною періодичністю над кожним боїдом виконуються дії, описані у методі MoveBoids() (рис. 6).

Кожне з правил моделі працює незалежно, тому порядок розрахунку векторів не має значення.

Для правила гуртування використаємо поняття центру мас. Центр мас – це середня позиція всіх боїдів (за аналогією з відповідною фізичною формулою, вважаємо масу всіх боїдів однаковою).

Нехай маємо *N*боїдів –b_1, b_2, ..., b_n. Розташування боїду у даний момент часу позначимо *b.position*.

Розрахуємо центр мас за формулою:

$$c = \frac{\left(\begin{array}{c} b_1.position + b_2.position + \\ +... + b_n.position \end{array}\right)}{N},$$

де позиції є векторами, а N – скалярне число.

Для кожного боїду скоригуємо знаходження центру мас так, щоб нараховувались усі об'єкти окрім даного:

$$c_i = \frac{\left(\begin{array}{c} b_1.position + b_1.position + \\ +b_{i-1}.position + b_{i+1}.position + \\ +... + b_n.position \end{array}\right)}{N-1}.$$

Далі необхідно задати коефіцієнт зміщення до центру k і розрахувати зміщення:

$$\left(c_i - b_1.position\right)k.$$

Псевдокод правила гуртування описано на рис. 7.

Основне завдання правила збереження дистанції – заборонити боїдам стикатися між собою. Псевдокод відповідної функції поданий на рис. 8.

Варто зазначити, що функция діє на обидва боїди, що надто наблизились один до одного.

Алгоритм правила рівняння подібний до алгоритму правила гуртування, але в даному випадку ми працюємо зі швидкостями, а не розташуваннями (рис. 9).

Алгоритм можна доповнювати залежно від умов задачі. Кожне нове правило n розраховує vectorn, що впливає на кінцеву зміну швидкості та положення. Так, можна додати вплив течії чи вітру на рух кожної істоти у зграї, додавши до параметрів боідywind.velocity.

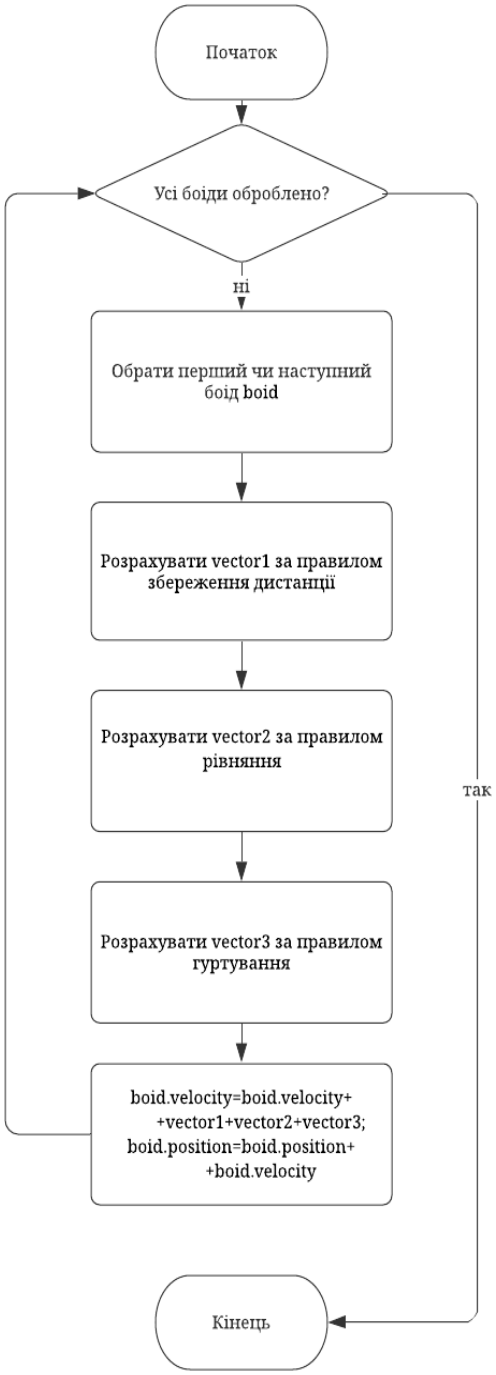


Рис. 6. Схема методу MoveBoids()

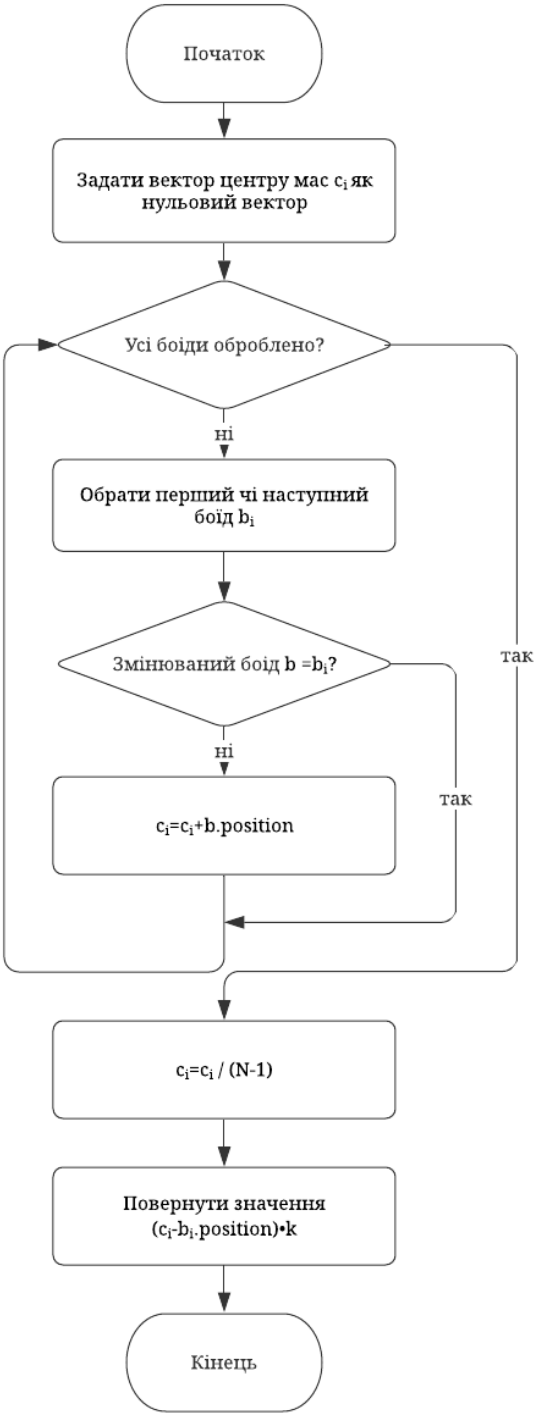


Рис. 7. Схема правила гуртування

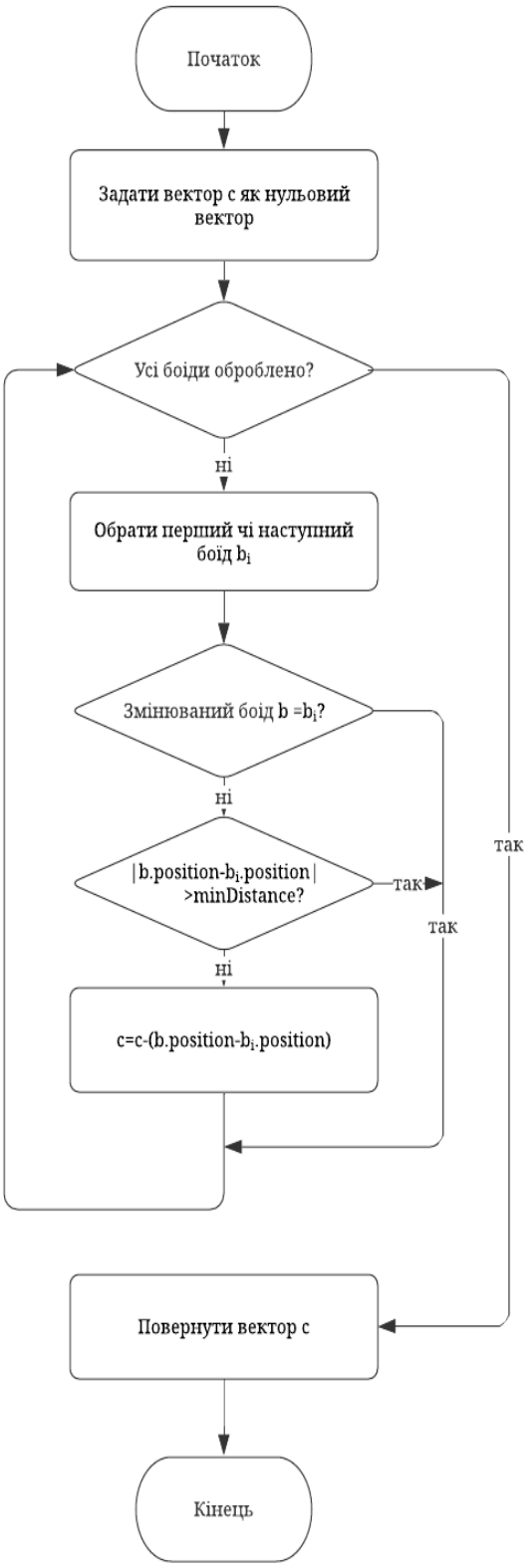


Рис. 8. Схема правила збереження дистанції

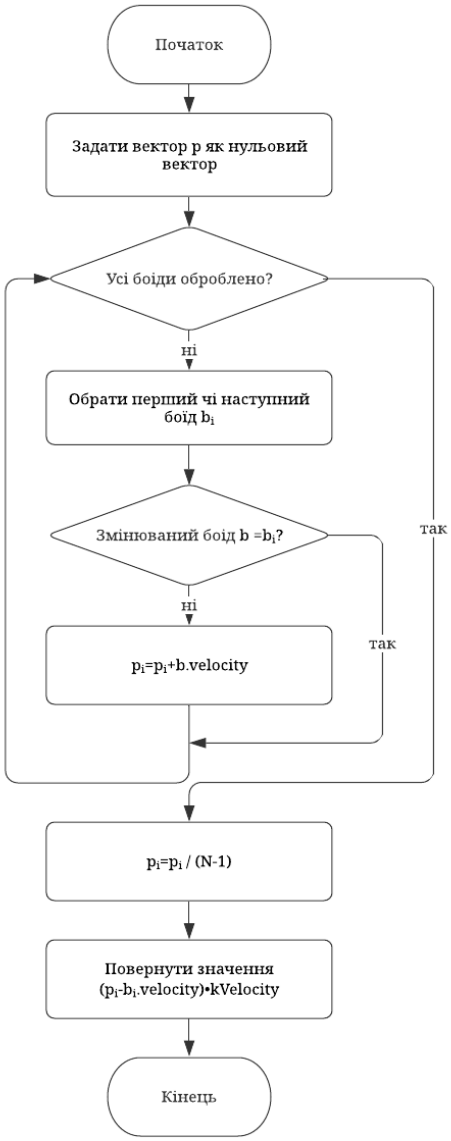


Рис. 9. Схема правила рівняння

Найпростіше функція направленнябоїду до конкретної цілі подано за допомогою блок-схеми на рис. 10:

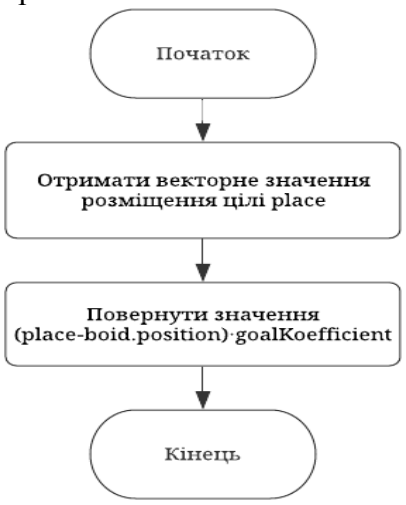


Рис. 10. Схема функції направлення боїду до конкретної цілі

Висновки

З огляду на проаналізовані дані, можна зробити висновок, що модель поведінки Рейнольдса може бути застосована для імітації руху об'єктів у зграї під час розробки 2D та 3D комп'ютерних ігор. Дослідження базових правил моделі показало, що алгоритм легко універсалізується та видозмінюється за потребами розробника.

Список літератури

1. Computergraphics : principlesandpractice / [J. F. Hughes, A. VanDam, M. McGuire та ін.].Boston: Addison-Wesley, 1995. – 1210 с.
2. Reynolds C. Reynold'shomepage [Електронний ресурс]– Режим доступу до ресурсу: http://www.red3d.com/cwr/.
3. HartmanC., BenesB.Autonomousboids //ComputerAnimationandVirtualWorlds. 2006. №17. С. 199–206.
4. Reynolds C. Flocks, Herds, andSchools: A DistributedBehaviouralModel // ComputerGraphics. 1987. №21. С. 25–34.

УДК 004:94: 005:94

ОНУФРІЄВА А.О.
СПЕРКАЧ М.О.
ГОНЧАРОВ К.О.
ПОПЕНКО В.Д.

ПРАКТИЧНЕ ВИКОРИСТАННЯ МЕТОДУ ГІЛОК ТА МЕЖ ДЛЯ ВИРІШЕННЯ ПРОБЛЕМИ ЗНАХОДЖЕННЯ ОПТИМАЛЬНОГО РОЗКЛАДУ ПРИ ПІДГОТОВЦІ ДО ІСПИТУ

В даній статті описано практичне використання методугілок та меж для знаходження оптимального розкладу підготовки школярів або студентів до іспитів. Використання алгоритму демонструється на прикладі підготовки учнів одинадцятих класів до складання зовнішнього незалежного оцінювання. Також були приведені результати роботи алгоритму, що доводять коректність використаних методів.
КЛЮЧОВІ СЛОВА: Метод гілок та меж, зовнішнє незалежне оцінювання, розклад

This article describes the practicaluse of the branch and bound method to find the optimals chedule for preparing students for exams.The use of algorithm is exemplified by the preparation of eleventh-grade students for external in dependent assessment. There sults of the algorithm prove the correctuse of methods.
KEYWORDS: Branch and Boundary Method, External Independent Evaluation, Schedule

1. Вступ

В даній статті описано практичне використання методу гілок та междлявирішення ускладненої«задачі про рюкзак» для знаходження оптимального розкладу підготовки школярів або студентів до іспитів. Використання алгоритмів демонструється на прикладі підготовки учнів одинадцятих класів до складання зовнішнього незалежного оцінювання. Також були приведені результати роботи алгоритмів, що доводять коректність використаних методів.

Всі теми мають якийсь час, що потрібен для вивчення, та кожна тема має свою важливість, тобто можлива така ситуація, що викладач вважатиме деякі теми не досить важливими для вивчення, наприклад такі, що можуть бути у білеті з дуже малою вірогідністю.Також деякі теми неможливо вивчити, якщо не вивчені попередні теми, для урахування таких залежностей

використовується матриця передувань. Кожний викладач складає для себе план, за яким він буде проводити навчання. Значення важливості тем та часу їх вивчення є ключовими елементами у подальшому дослідженні.

2. Метод гілок та меж

Метод гілок і меж є варіацією методу повного перебору з тією різницею, що ми виключаємо свідомо неоптимальні гілки дерева повного перебору. Як і метод повного перебору, він дозволяє знайти оптимальне рішення і тому відноситься до точних алгоритмів.

Оригінальний алгоритм[1], запропонований Пітером Колесаром (англ. PeterKolesar) в 1967 році, пропонує впорядкувати предмети по їх питомої вартості (стосовно важливості до ваги) і будувати дерево повного перебору. Його поліпшення полягає в тому, що в процесі побудови дерева, для кожного вузла ми оцінюємо верхню межу вартості рішення, і продовжуємо будувати дерево тільки для вузла з максимальною оцінкою. Коли максимальна верхня межа виявляється в листі дерева, алгоритм закінчує свою роботу.

Здатність методу гілок і меж зменшувати кількість варіантів перебору сильно спирається на вхідні дані. Його доцільно застосовувати лише в тому випадку, коли питомі вартості предметів відрізняються значно.

Метод гілок та меж має приблизну складність $O(2^N)$

3. Постановка задачі

Школа займається підготовкою учнів 10-11х класів до вступних іспитів. Кожного року видається список тем A1, A2, A3, …,An ($A_i i \in \{1; n\}$), що виносяться на іспит (n-кількість тем), та дата проведення цього іспиту, за якою вираховується час до іспиту(T). По кожній темі виставляється час на її вивчення t_i, та важливість вивчення теми p_i. У деяких тем є послідовність, яку не можна порушувати, але в будь-яке місце цієї послідовності можна додати тему.

Потрібно знайти такий порядок тем, при якому кількість не пройденого матеріалу буде мінімальна.

4. Опис методу

Як відомо, в методі гілок і меж є два моменти алгоритмізації[2], що визначаються специфікою завдання: розбиття вихідної множини комбінацій на підмножини з подальшим вибором підмножини для чергового розбиття і обчислення нижніх (верхніх) меж (оцінок) значень функції, що оптимізується, на підмножинах. Розбиття множини на підмножини називається розгалуженням, а вибір підмножини для розбиття - його стратегією. Оцінка вершини останнього ярусу - рекорд - являє собою поточне значення функції, що оптимізується, яке далі порівнюється з оцінками вершин попередніх ярусів, в результаті чого неперспективні для розгалуження підмножини відсіваються, а перспективні розбиваються, доповнюючи дерево рішень.

Алгоритм завершує роботу тоді, коли будуть порівняні з рекордом всі поточні оцінки вершин. Алгоритм реалізує розбиття кожної чергової множини на дві підмножини, вибір підмножини для розгалуження здійснюють за максимальною (мінімальною) оцінкою функції, що оптимізується, оціночні завдання формуються так, що більш точно обчислюються оцінки.

Що стосується даної задачі, то процес розбиття чергової множини здійснюється на дві підмножини, перша з яких містить комбінації компонент вектора з $x_i = 0$, друга - з $x_i = 1$. Стратегія розгалуження полягає в тому, щоб обирати чергову підмножину для розбиття по максимальній оцінці верхньої межі функції, що оптимізується. В результаті отримуємо бінарне дерево пошуку рішень, зображене на рис. 1.

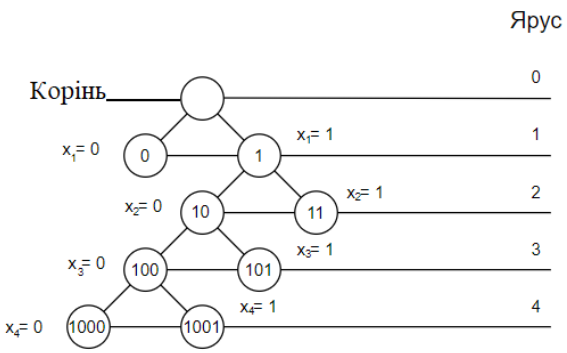


Рис. 1. Бінарне дерево пошуку рішень

Для обчислення оцінок верхніх меж функції, що оптимізується на підмножинах формулюється наступне оціночне завдання[3]: знайти

$$Q = \max \sum_i p_i \cdot x_i$$

За умов:

$$\sum_i t_i \cdot x_i \le T$$

$$x_i = \{0; 1\}$$

$$C_{ij} = \begin{cases} 1, \text{якщо є зв'язок між темами} \\ \text{та важливий порядок їх вивчення} \\ 0, \text{інакше} \end{cases}$$

Для знаходження усіх допустимих розв'язків слід перенумерувати список тем відповідно до матриці передувань так, що для C_{ij}=1, i < j. Таким чином перша вершина не матиме попередників.

Тоді оціночні завдання для вершини r+1 повинні бути сформульовані так: знайти

$$Q_{r+1} = \max(Q_r + \sum_{i=r+1}^{n} p_i x_i)$$

За умов:

$$\sum_{i=1}^{r+1} t_i x_i \le T$$

$$C_{ij}(x_j - x_i) \ge 0,$$

де $i, j \in [1; n]$ та $i \ne j$

5. Схема алгоритму розв'язання задачі

Додаткові змінні:

Q_i//значення критерію на поточному кроці,

Q_k// максимальне значення критерію на поточному кроці,

R // значення рекорду,

X={$x_1, \dots, x_n$}//вектор тем,

T_i // час використаний на усі обрані теми, включаючи поточну.

Вхід: C_{ij},p_i, t_i, T

Вихід: x, L// Сумарний час для вивчення переліку тем

Крок 1. Покласти i = 0 , рекорд R = 0.

Крок 2. Покласти i=i+1 .

Крок 3. Покласти x_i=0 ,

визначити оцінку Q_iпри x_i=0 та запам'ятатиQ_k.

Крок 4. Покласти x_i=1, визначити оцінку Q_i , визначити T_i, при x_i=1.

Крок 5. Обрати вершину для галуження з більшою оцінкою Q_i.

Крок 6. Якщо C_{ij} (x_j-x_i)<0 перейти на крок 11, інакше перейти на крок 7.

Крок 7 Якщо T_i≤T перейти на крок 8, інакше перейти на крок 11.

Крок 8. Якщо Q_i≤R, перейти на крок 11, інакше перейти на кроку 9.

Крок 9. Якщо i<n, запам'ятати більше Q_i, відповідне x_i , і повернутися до кроку 2; інакше перейти до кроку 10.

Крок10. Покласти R рівним більшому Q_i. Запам'ятати вектор X .

Крок 11. Покласти i = i -1 .

Крок 12. Якщо i =0 , зупинитися; інакше перейти до кроку 13.

Крок 13. Обрати Q_i, яке було запам'ято-ване.

Крок 14. Якщо Q_i≤R, повернутися на крок 12, інакше перейти на крок 3.

6. Описрезультатів

В результатівипробувань та досліджень на різних масивах даних були отримані такі дані:метод гілок та меж на невеликих масивах даних дозволяє прибирати варіанти вибору переліків тем, що заздалегідь не є допу-стимими, що дозволяє нам пришвидшити процес отримання допустимого результату і шуканого розкладу.

При дослідженні великих масивів даних метод гілок та меж за часом працює довше, що є очікуваним та повністю природним, бо збільшується об'єм даних, що повинні бути аналізовані. Разом з тим алгоритм цього методу продовжує спрощувати процес відбору правильної послідовності тем для розкладу.

Для будь-якого об'єму масиву даних, що аналізується, метод гілок та меж видає доволі точний результат за досить прийнятний час, показуючи складність, що була описана у пункті 2.

Висновок

Проблема пошуку кращого списку темзустрічається у кожного учня, у кожного викладача. Учні перед іспитами мають достатньо малу часу на вивчення питань, у викладачів з кожним роком зменшується кількість годин, виділених на предмет. В обох випадках потрібно вивчити/розказати якнайбільше. Саме тому вирішення проблеми є досить актуальним.

Список літератури

1. Канцедал С.А. Костикова М.В.. Конструювання й дослідження алгоритмів рішення задачі про рюкзак. [Текст] / Автомобільний транспорт – Х.: УДК, 2015. – 154 с.
2. Land A.H., andDoig A.G. Anautomatic method of solving discrete programming problems. Econometrica. v28 (1960), pp 497-520.
3. Корбут А.А., Финкельштейн Ю.Ю. Дискретное программирование М. Наука. Гл. ред. физ.-мат. лит. 1969.сс 213-219.
4. Саати Т. Л. Целочисленные методы оптимизации и связанные с ними экстремальные проблемы. — М.: Мир, 1973. — 302 с.

УДК 683.519

МЯГКИЙ М. Ю.,
ТВЕРДОХЛІБ А.І.
КОГАН А.В.

СИСТЕМА АВТОМАТИЗОВАНОГО РОЗГОРТАННЯ РОБОЧОГО СЕРЕДОВИЩА ДЛЯ РОЗРОБНИКІВ ПРОГРМАНОГО ЗАБЕЗПЕЧЕННЯ

В даній статті розглянуто практичне застосування системи автоматизованого розгортання робочого середовища (САРРС).Особливістю системи є те, що її розгортання не залежить від платформи, а використання інтерфейсу взаємодії між Front-End таBack-End частинами (RESTAPI) дозволяє користуватись нею з будь-якого пристрою, а також відкрити документацію по системі при необхідності її розширення. Також САРРС надає змогу в повній мірі реалізувати можливості розробки проекту будь-якої складності, а не є лише sandbox для не складних операцій, як системи аналоги. Використання Dockerдозволяє мати достатню кількість проектів в системі.

САРРС, АВТОМАТИЗАЦІЯ, REST, DOCKER

This article discusses the practical application of the Automated Deployment System (SARRS). The peculiarity of the system is that its deployment is platform independent, and the use of the Front-End and Back-End Parts Interface (REST API) allows it to be accessed from any device, and to open system documentation as needed to expand it. SARRS also enables full implementation of the project development capabilities of any complexity, and is not just a sandbox for non-complex operations like analog systems. Using Docker allows you to have enough projects in the system.
SARRS, AUTOMATION, REST, DOCKER

1. Вступ

Сьогодні спостерігається бурхливий розвиток у сфері інформаційних технологій. Але на ряду з цим прогресом з’явились і нові проблеми, такі як: менеджмент проекту, швидке залучення нових розробників і розгортання ними проекту, що є ключовим фактором до його успішного опанування і стрімкого приєднання до розробки.

Також, будь-яка розробка, у наші дні, має бути доволі гнучкою, саме тому з’явилась необхідність перенесення її зі стаціонарного місця в офісі, на зручний для програміста

девайс, котрий не буде «прив'язаний» до певного місця (мобільність розробника).

Метою створення даної системи є надання розробнику необхідної гнучкості в розробці проектів. Гнучкість полягає в тому, що:

1. Не обов'язково знаходитись за одним фізичним робочим місцем.
2. Розгортання системи, займає менше часу і відбувається практично «в один клік».
3. Для ефективної роботи більше не потрібно, кожному з розробників мати потужні обчислювальні засоби, вся ця потужність залежить від серверу.

Завдяки централізації обчислювального процесу можна досягти децентралізації робочих місць розробників, тобто робоче місце користувача більше не пов'язане з місцем де відбуваються всі обчислювальні операції.

2. Система автоматизованого розгортання робочого середовища

Системи, що дозволяють вести віддалену розробку – ту, яка дозволяє не сидіти на одному місці, а займатись кодуванням з будь-якого зручного девайсу та будь-де; вже існують і користуються попитом, однак, вони не є досконалими, бо мають високу ціну, або ж не мають всіх необхідних функцій для комфортної розробки. Більшість користувачів визнає такі системи як: JDOODLE[1], Ripl.it[2]та інші їм подібні, достатньо зручними, але їх вартість та недостатність не дає можливості користуватись даними ресурсами для повноцінної розробки програмного продукту. А такі системи, як: Ideone[3], OnlineGDB[4] та їх подібні, що є безкоштовними, не є практичними, бо, часто, вони достатньо обмежені в мовах програмування, не мають зручного інтерфейсу чи просто не надають всіх необхідних функціональних можливостей для користувачів. Також, усі вище перелічені системи зберігатимуть ваш проект на своїх закритих ресурсах, а для компаній це не є допустим, так як це збільшує ризики витоку інформації і в них неможливо вести розробку «реальних» проектів навіть середнього рівня складності. Все це робить процес розробкиобмеженим та, або не достатньо зручним. CAPPC надає можливість зробити розробку більш комфортною, враховуючи недоліки систем-аналогів. Ключовою частиною розробки є створення docker контейнерів на сервері, та їх налаштування. Головними перевагами системи є:

1. Віддалена розробка
2. Кросплатформенність
3. Наявність можливості менеджменту задач
4. Наявність вбудованого месенджера
5. Гнучке налаштування робочих середовищ

Розглянемо CAPPC в наступному вигляді: Маємо певний проект p і fфайлів, з яких власне і складається проект. Кількість файлів в проекті на пряму впливає на: пам'ять яку сервер використовуватиме для його зберігання, складність проекту, тобто його швидкодію та можливості редагування іншими користувачами.

Для розв'язку завдання покращення взаємодії між користувачами в системі JDOODLE, наприклад, використовуютьAPI, що базується на RESTAPI. В даному випадку інформацію з сайту (код), буде відправлено на їх компілятор через запити і через них же, користувач отримає результат виконання. Але в даному випадку, їх система потребує завантаження проекту з усіма змінами і не враховує одночасну розробку кількох осіб.

CAPPC пропонує спосіб врахування всіх правок кожного з користувачів збільшення надійності не погіршуючи сам процес роботи.

Для цього проект, буде запускатись на одній операційній системі і зберігатиме свою структуру, до тих пір, доки певна кількість розробника не підтвердять доцільність внесених змін іншим розробником. Після цього, проект буде оновлено і файл зі змінами розробника, буде очищено, тобто, його розмір буде 0байт.

Також система буде працювати з різними компіляторами, тобто, направлятиме код певної мови на відповідний їй компілятор і повертатиме результат обробки користувачу.

3. Постановка задачі

Нехай існує певне підприємство і воно має сервер для запуску проекту і rрозробників. Проект буде складатись з fфайлів, кожен з яких матиме певний розмір smегабайт. Всього на сервері Ммегабайт доступної пам'яті. При зміні проекту, розробником, для нього буде створено файл в котрому будуть відображені

його зміни і даний файл буде займати s` мегабайтпам'яті. Потрібно виконати nзмін проектуm-ою кількістю розробників таким чином з пороговою кількістю розробників для прийняття змін p, щоб сумарний розмір проекту, тобто кількість мегабайт, що він займає, з урахуванням змін, був якомога меншим за загальний розмір доступної пам'яті. $\sum_{i=1}^{f} s_i + \sum_{j=1}^{m} s`_j$ – це сума проекту з урахуванням всіх змін, де $\sum_{i=1}^{f} s_i$ – необхідна кількість пам'яті для зберігання проекту, $\sum_{j=1}^{m} s`_j$ – необхідна кількість пам'яті для зберігання внесених правок, а M -це загальна кількістьпам'яті на сервері.

4. Алгоритм вирішення задачі

Маємо наступну умову задачі:

$$\min z = x_1 + x_2;$$

$$x_1 \le M;$$

$$x_2 \le M - x_1;$$

$$x_1, M > 0;$$

$$x_2 \ge 0;$$

де $x_1 - \sum_{i=1}^{f} s_i$, $x_2 - \sum_{j=1}^{m} s`_j$, M – доступна пам'ять,z – розмір проекту з внесеними змінами і сума розмірів файлів зі змінами.

Також, необхідно врахувати те, що при узгодженні певних змін з, щонайменше ррозробниками, необхідна кількість пам'яті для зберігання змін буде зменшено, а необхідну кількість пам'яті для зберігання проекту буде зменшено, або збільшено, відповідно до внесених змін.

Розв'язання поставленої задачі, слід розділити на декілька підзадач:

1. Знаходження фінального розміру проекту.
2. Знаходження суми розмірів фінальних файлів зі змінами.
3. Сума розв'язків попередніх пунктів.

Для того, щоб розв'язати першу підзадачу, необхідно: перевіряти розмір проекту у файловій системі після прийняття змін.

Для розв'язання другої: перевіряти розмір файлу зі змінами розробника, чиї зміни було визнано доцільними і додано до проекту.

Для розв'язку третьої – підсумувати відповідні розміри.

Загальна схема алгоритму знаходження фінального розміру проекту і суми розмірів фінальних файлів зі змінами:

1. Знаходження значення поточного розміру проекту.

2. Знаходження значення суми розмірів поточних файлів із змінами.

3. ЦИКЛ по кількості змін:

Знаходження значення кількості розробників в системі.

Знаходження значення кількості розробників, яка визнала зміни доцільними.

ЯКЩО було прийнято зміни необхідною кількістю розробників:

Видалення змін з поточного файлу зі змінами.

Оновлення значення розміру проекту.

Перерахунок суми розмірів файлів зі змінами.

Декремент кількості змін.

ІНАКШЕ

ЯКЩО кількість розробників в системі дорівнює 0

Перерахунок суми розмірів файлів зі змінами.

Перервати ЦИКЛ.

Таким чином, маючи описаний алгоритм, можна перейти до заключного етапу: сумування поточного розміру проекту і додаткових файлів із змінами.

З даної реалізації видно, що чим більше змін було прийнято і додано в проект, тим меншою буде z, що і є нашою головною метою.

5. Технічні засоби впровадження рішення

Dockerвикористовує так звані блоки, які формують шари[5].Таким чином,при створенні образів(image) використовуються попередньо завантажені шари, а не створюються знову.Таким чином можна досягти збереження пам'яті на сервері, при цьому не втрачаючи змогу запуску проектів та програм.

CAPPC використовує даний механізм для збереження ресурсів, при віддаленому запуску та розробці проектів. Таким чином економиться час, який працівник може використати на написання коду, замість того щоб встановлювати програмні засоби, а також економиться місце на сервері використовуючи шарову структуру і не створюючи зайві копії.

Висновок

На прикладі існуючих систем-аналогів, таких як JDOODLE та Ripl.it було продемонстровано попит на вирішення проблеми швидкого опанування нових проектів розробниками та її актуальність. Запропонована CAPPC містить переваги описаних систем, а також виправляє основні недоліки, які в них спостерігаються.

Зменшення обсягів використовуваних ресурсів серверу, а також зменшення часу на розробку, так як в системі присутня «гнучкість розробника» здатне наблизити CAPPC до дійсно rapiddevelopmentsystem, зменшуючивитрати бізнесу на одержання додаткової пам'яті на сервері та інтеграцію розробників, або ж їх заміну, у більш швидкий та надійний спосіб.

Зменшення основного робочого ресурсу (об'єму пам'яті, що займає проект і додаткові зміни до нього) досягається завдяки наступним підходам:

- Проект створюється в єдиному екземплярі.
- Кожен з розробників матиме свій файл зі змінами, що дозволяє кільком розробникам вести незалежну один від одного розробку.
- Оновлення проекту відбувається, лише з узгодженням мінімальним пороговим значенням розробників певних змін.

Отже використання розробленої системи забезпечить суттєве покращення в роботірозробників на підприємстві, раціональне використання ресурсів і скорочення затраченого часу на розгортання проекту загалом.

Список літератури

1. Документація до системи JDOODLE [електронний ресурс] URL: https://www.jdoodle.com/terms/
2. Документація до системи Repl.it [електронний ресурс] URL: https://docs.repl.it/misc/free-features
3. Документація до системи Ideone [електронний ресурс] URL: https://ideone.com/legal-gdpr
4. Документація до системи OnlineGDB [електронний ресурс] URL: https://www.onlinegdb.com/faq
5. Документація до системи Docker [електронний ресурс] URL: https://docs.docker.com/storage/storagedriver/

УДК 004.94

ШОЛУДЬКО А. А.
ПОПЕНКО В. Д.

ІНФОРМАЦІЙНА СИСТЕМА АДМІНІСТРУВАННЯ БЮДЖЕТНИХ ПРОГРАМ МІСЦЕВОЇ ВЛАДИ НА ПРИКЛАДІ ДОРОЖНЬОГО БУДІВНИЦТВА

У даній статті розглянута задача про ремонт доріг. Були описані методи її розв'язання, а саме жадібний та генетичний алгоритми.

КЛЮЧОВІ СЛОВА: задача про ремонт доріг, жадібний алгоритм, генетичний алгоритм.

In this article the task of repairing roads was considered.Methods of its solution, namely greedy and genetic algorithms,were described.

KEYWORDS:the task of repairing roads, greedy algorithm, genetic algorithm.

1. Вступ

На сьогоднішній день питання ремонту доріг є актуальним. Здається, що це одна з найголовніших проблем нашої країни. Більшість доріг знаходяться в поганому і навіть аварійному стані. На жаль, держава виділяє недостатньо коштів для ремонту доріг. З іншої сторони, якщо профінансувати одночасно всі потреби дорожнього будівництва, ця сума перевищить увесь бюджет. Тому дуже важливо розподілити ці кошти так, щоб люди були максимально задоволені результатом.

У даній статті розглядається задача, яка допомагає визначити, які саме ділянки дороги необхідно відремонтувати, щоб

вкластися в бюджет та мінімізувати час поїздки.

2. Постановка задачі

Дорога ділиться на однакові за розміром ділянки.

$\{1, \dots, n\}$ – скінченна множина номерів ділянок, впорядкованих вздовж дороги.

Ділянку або потрібно ремонтувати, або ні. Кожна ділянка має свою вартість, тобто розмір коштів, які потрібно витратити на її ремонт.

$\{l_1, \dots, l_n\}$ – скінченна множина вартості ділянок. Якщо ділянка не потребує ремонту, вартість її ремонту дорівнює нулю.

Кошти, виділені на ремонт дороги, обмежені. Відремонтовані ділянки складають відремонтовану частину дороги. Чим довша неперервна частина дороги відремонтована, тим більшу швидкість може розвити водій. Чим більша швидкість розвинена, тим менше часу автомобіль знаходиться в дорозі. Швидкість, яку може розвивати автомобіль, обмежена.

Необхідно визначити, які ділянки дороги необхідно відремонтувати, щоб мінімізувати час поїздки.

Сформулюємо математичну постановку задачі:

Сформулюємо цю задачу у вигляді задачі цілочислового лінійного програмування. Введемо булеві змінні:

$x_j = \begin{cases} 1, \text{якщо } j - \text{ту ділянку дороги ремонтуватимуть} \\ 0, \text{інакше} \end{cases}$

і невід'ємніцілочисельнізмінні:

u – максимальний обсяг бюджету, який виділений на ремонт доріг.

a – прискорення автомобіля.

$s_{\text{поч ділянки}}$ – початкова довжина ділянки.

s_j – довжина j-ої ділянки дороги. Оскільки при розташуванні декількох відремонтованих чи невідремонтованих ділянок доріг підряд вони об’єднуються в одну ділянку, то довжина ділянки може змінюватися.

$s_g = 10$ м – відстань, протягом якої здійснюється гальмування.

$V_{max} = 90$ км/год – максимальна швидкість, яку може розвинути автомобіль, з урахуванням Правил дорожнього руху.

$V_{min} = 10$ км/год – мінімальнашвидкість, яку може розвинути автомобіль на дорозі, яка потребує ремонту.

t_g – втрати часу від гальмування.

t_r – втрати часу від розгону.

$t_{p\,j}$ – втрати часу від повільної їзди на j-й ділянці дороги.

$t_{\text{розгону}}$ – час розгону.

$t_{\text{гальмування}}$ – час гальмування.

t_j– загальні втрати часу від поїздки на j-й ділянці дороги.

Тоді в цільової функції мінімізується втрати часу від поїздки:

$$z = \sum_{j=1}^{n} t_j \to min$$

При умовах:

Бюджет, виділений на ремонт доріг, обмежений:

$$\sum_{j=1}^{n} x_j l_j \le u$$

Швидкість, яку може розвивати автомобіль, обмежена:

$V_j \le V_{max}, j \in \{1, \dots, n\}$.

Втрати часу від їзди на j-й ділянці дороги:

$t_j = t_g + t_{p\,j} + t_r, j \in \{1, \dots, n\}$

t_j визначається так.

-t_j=0.

Якщо попередня ділянка нормальна, а j – не відремонтована, додаємо час гальмування.Якщо попередня ділянка не відремонтована, а j – нормальна, додаємо час розгону. Якщо ділянка j не відремонтована, додаємо втрати часу від повільної їзди.

Якщо $j \ne 1$, l_{j-1}>0 та $l_j = 0, t_j = t_j + t_g$.

Якщо $j \ne 1$, l_{j-1} =0 та $l_j > 0, t_j = t_j + t_r$.

Якщо $l_j > 0, t_j = t_j + t_{pj}$.

Час розгону:

$$t_{\text{розгону}} = \frac{V_{max} - V_{min}}{a}.$$

Втрати часу від розгону:

$$t_r = t_{\text{розгону}} - \frac{s}{V_{max}}$$

Час гальмування автомобіля дороги знаходимо з наступної формули:

$$s_g = V_{max} t_{\text{гальмування}} - \frac{a t_{\text{гальмування}}^{\,2}}{2}$$

Втрати часу від гальмування:

$$t_g = t_{\text{гальмування}} - \frac{s_{\text{поч.ділянки}}}{V_{max}}.$$

Втрати часу від повільного руху:

$$t_{p\,j} = \frac{s_j}{V_{min}}, \quad j \in \{1, \dots, n\}$$

3.Принцип оптимальності Беллмана

Принцип Беллмана говорить, що оптимальна стратегія визначається лише станом системи в даний момент і не залежить від того, як ця система прийшла в дану точку.

Розглядаючи дану задачу, ми можемо бачити, що час поїздки автомобіля залежить від початкового розташування автомобіля та розміщення відремонтованих ділянок дороги, оскільки від цього залежить, яку середню швидкість може розвинути автомобіль, час гальмування та розгону. Зокрема, початкова швидкість на ділянці залежить від стану попередньої ділянки.

З цього можемо зробити висновок, що для даної задачі принцип оптимальності Беллмана не виконується. Отже, її не можна розв'язати методами динамічного програмування.

Можемо зробити висновок, що для розв'язання цієї задачі найкраще підходять наближені алгоритми, а саме жадібний та генетичний.

4. Опис можливих алгоритмів розв'язку задачі

Жадібний алгоритм – алгоритм, що полягає в прийнятті локально оптимальних рішень на кожному етапі, при цьому допускаючи, що кінцеве рішення також виявиться оптимальним.[2]

Реалізуючи жадібний алгоритм, ми будемо обирати ті ділянки, які мають найменшу вартість, до тих пір, доки не закінчаться кошти, виділені на ремонт дороги. Після цього треба обчислити кошти, які водії витратять на пальне, враховуючи розташування відремонтованих ділянок.

Генетичний алгоритм – це еволюційний алгоритм, який використовують для вирішення задач оптимізації і моделювання шляхом послідовного підбору, комбінування та варіації шуканих параметрів. Генетичний алгоритм являє собою метод, що відображає природну еволюцію методів вирішення проблем. Генетичні алгоритми обробляють закодовану форму параметрів задачі, здійснюють пошук рішення, виходячи з деякої популяції, використовують тільки цільову функцію, застосовують ймовірнісні правила відбору.[3]

Реалізуючи генетичний алгоритм, ми генеруємо популяцію, тобто розв'язки задачі (вектор x), випадковим чином. Потім виконується мутація кожної особи в популяції: ген, який обирається випадковим чином, змінює своє значення на протилежне. Потім обчислюється вартість ремонту дороги для кожного розв'язку (особи) і відкидаються ті, вартість, яких перевищує вартість виділеного бюджету.Серед нащадків, які залишилися,обирається найкращий розв'язок, тобто той, де час поїздки найменший.

Висновок

У даній роботі розглянута задача про розподіл капіталовкладень на ремонт ділянок доріг. Визначено, які наближені алгоритми можна використати для її розв'язання та яким чином їх можна застосувати в рамках даної задачі.

Планується, що дана задача допоможе визначити, яким чином потрібно ремонтувати дороги, щоб зменшити час поїздки та мінімізувати кількість незадоволених людей.

Список літератури

1. Рівноприскорений рух. Прискорення. Швидкість тіла і пройдений шлях під час рівноприскореного прямолінійного руху. [Електронний ресурс]. – 2019. – Режим доступу доресурсу: https://vseosvita.ua/library/rivnopriskorenij-ruh-priskorenna-svidkist-tila-i-projdenij-slah-pid-cas-rivnopriskorenogo-pramolinijnogo-ruhu-118370.html

2. Жадібні алгоритми. Переваги та недоліки [Електронний ресурс]. – 2016. – Режим доступу доресурсу: https://lektsii.org/8-43572.html

3. Генетичні алгоритми. Ключові поняття і методи реалізації [Електронний ресурс]. – 2018. – Режим доступу доресурсу: http://www.znannya.org/?view=ga_general

УДК 519.854.2

ЛУКОВА О.Ю.
ЖДАНОВА О.Г

ЗАДАЧА СКЛАДАННЯ ПЛАНІВ ІНКАСАЦІЙ ТЕРМІНАЛІВ З ПРИЙОМУ ПЛАТЕЖІВ У НАСЕЛЕННЯ

Робота присвячена задачі складання планів інкасацій терміналів з прийому платежів з метою мінімізації сумарних витрат на інкасацію готівки з терміналів та збитків від можливих вимушених простоїв терміналів. Були розглянуті типи задач маршрутизації транспортних засобів та умови їх використання для поставленої задачі. В роботі розглянуто частковий випадок задачі складання планів інкасацій, який зводиться до задачі комівояжера. Для цієї задачі розроблена модифікація методу найближчого сусіда, яка за рахунок паралельної реалізаціїобчислень дозволила отримати більш точні результати за менший час роботи.

The article is about the task of route planning for cash collection of terminals for receiving payments in order to minimize the total cost of collecting cash from terminals and losses from possible forced downtime of terminals. The types of vehicle routing problems and the conditions of their use for the task were considered. The article deals with the partial case of the problem of route planning which comes down to the Traveling Salesman Problem. For this problem, a modification of the nearest neighbor method was developed, which at the expense of parallel implementation of calculations made it possible to obtain more accurate results in less operating time.

КЛЮЧОВІ СЛОВА: ЗАДАЧА МАРШРУТИЗАЦІЇ, СКЛАДАННЯ ПЛАНУ, ЗАДАЧАКОМІВОЯЖЕРА, МЕТОД НАЙБЛИЖЧОГО СУСІДА

1. Вступ

Задача складання планів інкасації є актуальною на сьогоднішній день і даних щодо її реалізації немає. Головною частиною цієї задачі є задача маршрутизації транспортних засобів (VehicleRoutingProblem – VRP), тобто складання оптимальних або близьких до них планівобслуговування клієнтів. Задача маршрутизації є NP-складною, а отже потребує значних витрат на розрахунки задля забезпечення більш точних результатів. А витрати на транспорт (його обслуговування, паливо) є суттєвими для бізнесу та при зростанні автопарку можуть збільшуватись у декілька разів. Отже, є потреба у побудові алгоритмів, що здатні отримувати наближені до точних результатів замінімально можливий час.

2. Постановка задачі

У місті є n відділень банку та m терміналів. Кожен з терміналів належить тільки одному з відділень, тобто задано вектор p належності терміналу:

$$p_j \in \{1, 2, \dots, n\}, j = 1, 2, \dots, m$$

Відома транспортна мережа$\langle V, E, c, t\rangle$, яка з'єднує термінали, де V- не порожня скінчена множина вершин (терміналів); E-множина ребер:$E = \{(v_i, v_j) \in V \times V\}$,кожному ребру$(v_i, v_j)$ мережі поставлена у відповідність невід'ємна вага (вартість)$c_{v_i v_j}$та час$t_{v_i v_j}$ проходження ребра.

Термінали приймають готівку від користувачів і заповнюються при накопиченні готівки в кількості S (в загальному випадку поріг накопиченнятерміналуjстановить$S_j, j = 1, 2, \dots, m$. Після заповнення термінал припиняє роботу і відключається до моменту вивезення з нього готівки. Такі вимушені простої призводять до зниження рентабельності мережі терміналів через зменшення грошових надходжень в банк через термінали.

Банківські відділення мають у своєму розпорядженні парк інкасаторських машин, які періодично направляються для вивезення готівки зі своїх терміналів. Кожен виїзд інкасаторської машини має певну вартість, яка залежить від типу машини, часу знаходження на маршруті та пройденої відстані.

Відділення збирають статистичні дані щодо інтенсивності заповнення кожного із своїх терміналів, щоденно зберігаючи інформацію про поточний стан (заповнений/незаповнений) та про кількість грошей, що надійшли до терміналу на кінець робочого дня. З урахуванням цієї інформації розрахована середня інтенсивність надходження грошей до кожного терміналу: $a_j, j = 1, 2, \dots, m$ (грн/день). Відповідно цього розраховані величини $b_j, j = 1, 2, \dots, m$, де b_j - збитки терміналу j за одну годину вимушеного простою (грн/год).

Інкасаторські машини можуть працювати тільки у банківські дні. Якщо термінал буде заповнений у банківські вихідні, то він буде відключений («залишиться без роботи та заробітку») до наступної інкасації. А якщо ж вивозити готівку занадто рано, то це збільшує витрати наобслуговування терміналів, призводить до зниження прибутковостімережітерміналів.

Скласти розклад виїзду інкасаторських машини, таким чином щоб мінімізуватисумарні витрати на інкасацію готівки з терміналів та збитки від можливих вимушених простоїв.

3. Огляд типів задач маршрутизації транспортних засобів

Існую велика кількість задач маршрутизації, які відрізняються типами вузлів, транспортних засобів (ТЗ) та обмеженнями. Метою задач маршрутизації є складання планів обслуговування клієнтів з мінімальними витратами.

Задача маршрутизації ТЗ (VehicleRoutingProblem, VRP)

Товар з депо повинен бути доставлений клієнтам у необхідному обсязі. Відома кількість доступних транспортних засобів. Кожен транспортний засіб починає та закінчує маршрут з депо та обслуговує одного або більше клієнтів. Метою є складання планів обслуговування клієнтів для кожного транспортного засобу з мінімальними витратами. Додатково є обмеження, що кожен клієнт обслуговується один раз. Витрати на транспорт розраховуються як вартість проїзду з будь-якої точки у будь-яку іншу точку та вартість виїзду конкретного транспортного засобу на маршрут. Саме цей, класичний, варіант задачі маршрутизації може бути використаний у конкретній постановці задачі.

Задача маршрутизації ТЗ з урахуванням часових вікон(VRP withTimeWindow)

Додаються обмеження на час, коли транспортний засіб може обслуговувати клієнта, час обслуговування та час на переїзди з одного пункті до іншого. У випадку коли транспортний засіб приїздить не у потрібне часове вікнойому потрібно зачекати, або при запізненні пункт вже не буде обслугований.

Цей тип задач може бути використаний коли термінали з такою інтенсивність, що потребують щоденних інкасацій. У реальності термінали не заповнюються так швидко, тому планування можна роботи більш грубо.

Задача маршрутизації з урахуванням завантаженостіТЗ (CapacitatedVRP)

У задачі маршрутизації з урахування завантаженості транспортних засобів до класичної задачі VRP додається обмеження на обсяг вантажів. Тобто на кожному маршруті обсяг вантажів, що перевозиться не повинен перевищувати відповідної величини(вантажопідйомностітранспортного засобу).

Цей тип задачі маршрутизації не може бути використаний для розв’язання даної постановки задачі, так як у конкретній задачі кількість готівки не складає настільки великих розмірів аби перевантажити транспортний засіб. Тому вважаємо, що усі наявні транспортні засоби вміщають будь-яку кількість готівки для перевезень.

Задача маршрутизації ТЗ з поверненням та доставкою товару (VehicleRoutingProblemPickupandDelivery)

У задачі маршрутизації з поверненням та доставкою товару до класичної задачі додається обмеження на обсяг вантажів та клієнти поділяються на пункти прийому та повернення.

Цей тип задачі може бути застосований у випадку обслуговування як терміналів з прийому платежів, так і банкоматів видачі готівки в одному маршруті.

Задача маршрутизації ТЗ з декількома депо (MultipleDepotVRP)

Ця задача характеризується наявністю декількох депо та у кожного депо є свій автопарк, а кожен маршрут починається в одному і тому ж депо. Кожне депо обслуговує тільки своїх клієнтів.

Ця задача також може бути використана у конкретній постановці задачі для декількох відділень банку, проте у випадку банку, коли депо не має великих кількостей вантажів таперевезень, більш раціональним є використання одного автопарку на усі депо і задача розбивається на декілька незалежних підзадач.

Задача маршрутизації ТЗ з більшим горизонтом планування (Periodic VRP)

У задачі планування маршрутів розширюється на період більше одного дня.

Так як інкасація банківських терміналів виконується у межах одного міста, то розширення часу на такі проміжки не є потрібним.

Задача маршрутизації запасів (Inventoryroutingproblems)

Кожен клієнт має норму витрат товару, початкові запаси товару та ємність складу. Депо повинно виконати таку кількість поставок аби забезпечити потреби клієнта.

Цей тип задач може бути використаний при поставках готівки у банкомати.

4. Математична постановка VRP

Виходячи з позначень постановки задачі, маємо класичну задачу маршрутизації, де граф $G = (V, E)$, V – множина вершин($|V| = n$), E- множина ребер $E = \{(v_i, v_j) \in V \times V\}$. Кожному ребру у відповідність поставлена вартість c_{ij}. Індексом $k \in A$позначатимемо відповідний транспортний засіб (де A – множина транспортних засобів). [1]

$$\sum_{k \in A} \sum_{(i,j) \in E} c_{ij} x_{ijk} \rightarrow min \quad (1)$$

$$\sum_{k \in A} \sum_{j \in V} x_{ijk} = 1, \quad \text{для } \forall i \in V \quad (2)$$

$$\sum_{j \in V} x_{0jk} = 1, \quad \text{для } \forall k \in A \quad (3)$$

$$\sum_{j \in V} x_{j0k} = 1, \text{для } \forall k \in A \quad (4)$$

$$\sum_{i \in V} x_{ihk} - \sum_{j \in V} x_{hik} = 0, \quad \text{для } \forall h \in V, \forall k \quad (5)$$

$$x_{ijk} \in \{0,1\}, \forall (i,j) \in V, \forall k \in A. \quad (6)$$

Цільова функція (1) визначає вартість маршрутів для всіх транспортних засобів. Обмеження (2) гарантує, що всі клієнтибудуть відвідані тільки одним транспортним засобом та тільки один раз. Обмеження (3) та (4) гарантують, що транспортний засіб покидає та повертається в депо тільки один раз.Обмеження (5) гарантує, що колитранспортний засіб прибуває до вершини h, то він повинен з неї виїхати.

5. Задача комівояжера

При частковому випадку, коли доступним є лише один транспортний засіб, задача зводиться до пошуку гамільтонового циклу або задачі комівояжера.

У межах міста можна зробити припущення що граф пунктів обслуговування є таким, щов ньому існує гамільнонів цикл.

Задача комівояжера є NP-складною, що говорить про складність знаходження її оптимального розв’язку. Виходячи з цього для її розв’язку необхідновикористовувати наближені абоевристичні методи (методи, які основані на якихось евристиках (правилах), які не гарантують отримання оптимального розв’язку, але дозволяють отримуватидостатньо прийнятні результати за прийнятний час.

6. Метод найближчого сусідадля розв’язаннязадачі комівояжера

Класичний метод найближчого сусіда є жадібним алгоритмом. Ідея алгоритму найближчого сусіда заснована на евристичному правилі: якщо на кожному кроці відвідувати найближчий пункт, то в цілому маршрут також вийде досить непоганий. Алгоритм забезпечує недопущення повторного заїзду до пункту та передчасного повернення в початковий пункт.Для удосконалення результатів алгоритму було введено модифікацію, яка дозволяє покращити отримувані розв’язки без додаткових витрат часу. Це досягається за рахунок паралельної реалізаціїалгоритму.

Якщо першою точкою маршруту обирати кожнуз наданих, або за деяким правилом, то алгоритм буде покривати більшу кількість варіантів маршрутів та зможе знайти кращий

варіант маршруту, ніж при випадковому вибору однієї стартової точки.

Перший пункт обирається зі списку терміналів, що потребуються інкасації.

Загальна схема модифікованого алгоритму найближчого сусіда

1. Отримати список пунктів, які необхідно відвідати.
2. Для кожного пункту спискувиконати:
   2.1. Обрати пункт стартовим.
   2.2. **ДОКИ** залишились невідвідані пункти:
      2.2.1. Перейти до найближчого невідвіданого пункту.
   2.3. Повернутись до стартового пункту.

При такій модифікації складність алгоритму дорівнюєn^2, отже при збільшеннікількості даних час роботи зростає за параболою. Шляхом паралельної реалізації можнашвидше отримувати результати, зберігаючи точність розрахунків, або за той же час отримувати кращі результати .

7. Результати досліджень

В результаті роботи алгоритму були отримані доволі точні результати.На рисунку 1зображеніприклади результатів розв'язання задач розмірності $n = 10$(рис.4а)та $n = 300$ (рис.4б) (вісь абсцис та вісь ординат відображають координати пунктів обслуговування). На рисунку 2 наведені результати порівняння часу роботу послідовної та паралельної реалізацій алгоритму. Як і очікувалось, паралельна реалізаціяалгоритмудозволяє отримувати кращі при тих же витратах часу. Тому для подальшихдосліджень для більш складних VRP доцільно розробляти такі алгоритми які дозволяють робити їх паралельні реалізації.

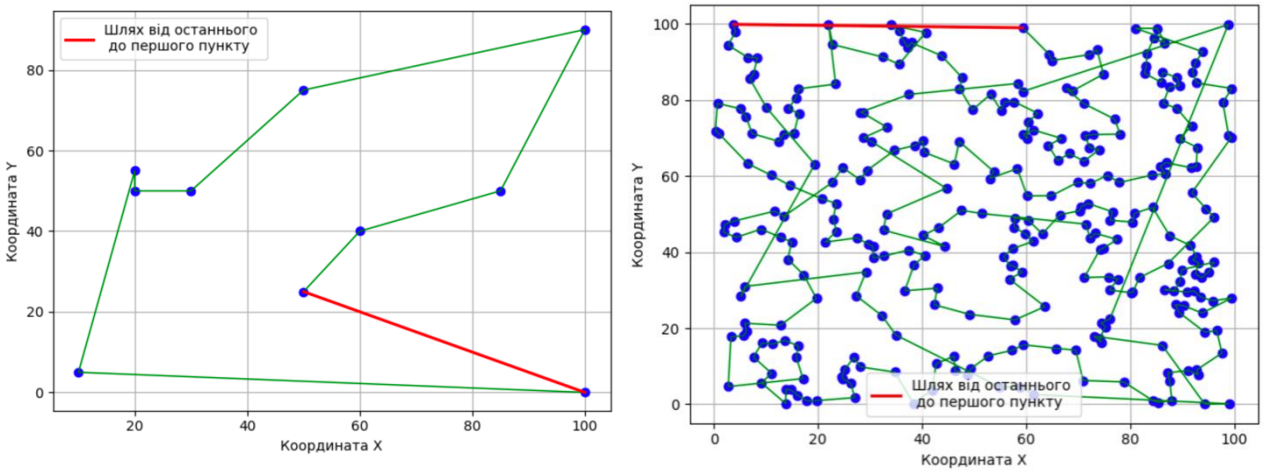
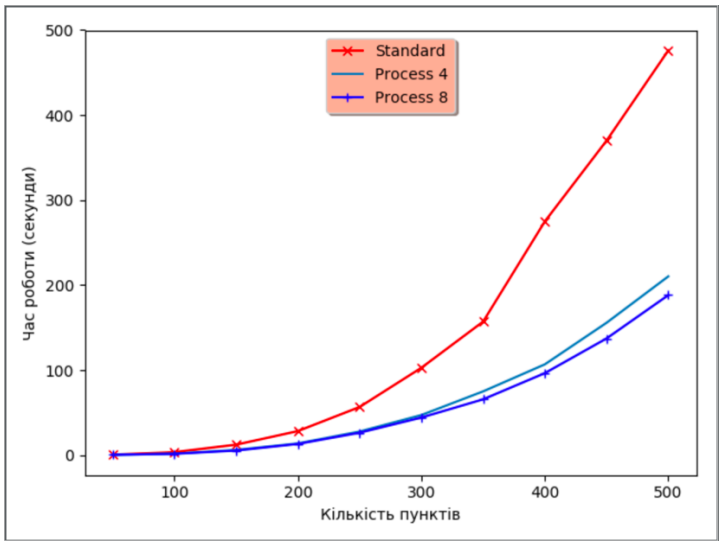


Рис. 1. Приклади результатів роботи



Standart – час роботи послідовної реалізації алгоритмі,
Process4 – час роботи паралельної реалізації на 4 процесах,
Process8 – час роботи паралельної реалізації на 8 процесах.

Рис. 2. Графік порівняння часу роботи

Висновки

Розглянута задача складання планів інкасацій терміналів з прийому платежів, що була зведена до задачі маршрутизації транспортних засобів. Базуючись на методі найближчих сусідів, був розроблений модифікований алгоритм розв'язання задачі комівояжера, що дозволяє знаходити близькі до точних розв'язки задачі. Були розроблені послідовна та паралельна реалізація алгоритму. Для оцінки ефективності були проведені експерименти на малих та великих обсягах даних та порівняні час роботи послідовної та паралельної реалізації. Для розв'язання задачі маршрутизації для більше ніж одного транспортного засобу планується розробка алгоритму бджол та його послідовна та паралельна реалізація.

Список літератури

1. Рассадникова Е.Ю., Коханчиков Л.А. Математическая модель задачи выбора рациональных маршрутов в системе управления транспортировки готовой продукции // Современныепроблемы науки и образования. – 2013. – № 5.
2. Задача коммивояжёра. // https://ru.wikipedia.org/wiki/Задача коммивояжёра
3. Алгоритм_ближайшего_соседа_в_задаче_коммивояжёраhttps://ru.wikipedia.org/wiki/Алгоритм_ближайшего_соседа_в_задаче_коммивояжёра

УДК 338.24

ПРОЦЮК Ю. В.,
ЖАРІКОВ Е. В.

ПІДХІД ДО АНАЛІЗУ І ПРОГНОЗУВАННЯ ІНФОРМАЦІЙНО-КОМУНКАЦІЙНИХ ПРОЦЕСІВ

Розглянуто етапи аналізу процесів в інформаційно-комунікаційних системах тавстановлено можливі варіанти їх класифікації для певних умов використання. Запропоновано підхід до формування ключових показників ефективності процесу.На прикладі центру оброблення викликів показано необхідність застосування відповідних методів для прогнозування плину процесів.

АНАЛІЗ ПРОЦЕСУ, ПРОГНОЗУВАННЯ ПРОЦЕСУ, КЛАСИФІКАЦІЯ ПРОЦЕСІВ, КЛЮЧОВІ ПОКАЗНИКИ ЕФЕКТИВНОСТІ

The stages of analysis of processes in information and communication systems are considered and the possible variants of their classification for certain conditions of use are determined. An approach to the formation of process key performance indicators is proposed. The example of the call center shows the need to use appropriate methods to predict the flow of processes.

PROCESS ANALYSIS, PROCESS FORECASTING, CLASSIFICATION OF PROCESSES, KEY PERFORMANCE INDICATORS

1. Вступ

Аналіз і прогнозування протікання процесів в інформаційно-комунікаційних системах може використовуватися для контролю їх стану і покращення функціонування.

Етапами аналізу процесу є:

- класифікація;
- визначення ключових показників ефективності;
- визначення «вузьких місць» – етапів процесу, що потребують найбільше ресурсів чи власне самих ресурсів, кількість яких є критичною для успішного виконання процесу.

Ґрунтовний аналіз процесу дозволить будувати прогнози, передбачати затримки, зупинкий виявляти їх причини.

2. Класифікація процесів

Для детального аналізу процесу потрібно визначити точну його класифікацію, зокрема, за ступенем автоматизації процеси

можуть бути: ручні, автоматизовані та автоматичні. За характером дій (спрямованістю), на найвищому рівні процеси поділяються на операційні та процеси управління й підтримки[1]. Деталізовану схему класифікації процесів за характером дії наведено на рисунку 1.

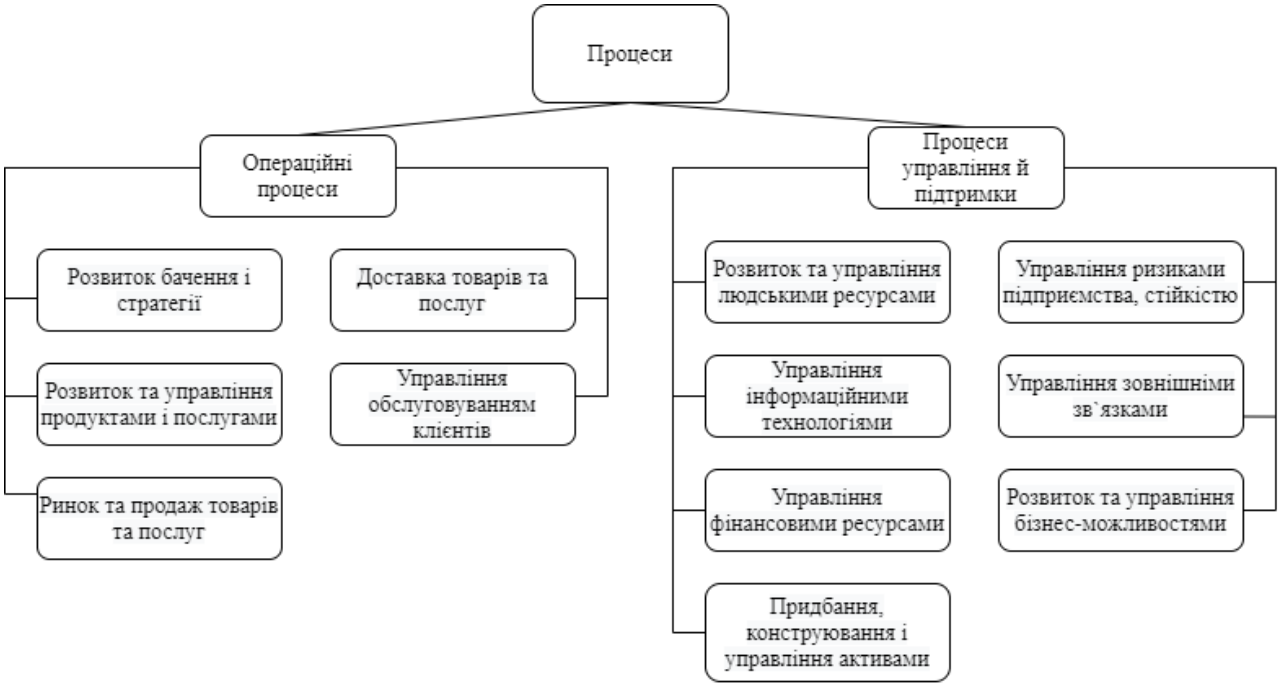


Рис. 1 – Дворівнева класифікація процесів за спрямованістю

За впливом на діяльність процеси можна класифікувати на основні, управлінські, процеси забезпечення та розвитку. За суб`єктами, що взаємодіють, процеси можна розділити на b2b, b2c, c2c, b2g, g2bта інші.

Для прикладу розглянемо процес роботи call-центру. За ступенем автоматизації процес є автоматизованим, оскільки деякі операції виконуються автоматично, інші є ручними, як-от розмова з клієнтом. За характером дії є операційним, а саме управлінням обслуговування клієнтів. За впливом на діяльність є процесом забезпечення, бо сприяє в наданні певного виду послуг. За суб`єктами, що взаємодіють є, як правильно, b2c – підприємство виконує операції з кінцевим споживачем, якщо не вказано інше. Отож, процес роботи call-центру є водночас автоматизованим, управлінням обслуговуванням клієнтів, процесом забезпечення і b2cпроцесом.

Як бачимо, в деяких класифікаціях наявний перетин класів. Тому жодна з наявних класифікацій не дозволяє повністю описати властивості процесу, максимально розкрити його суть, але їх поєднання допомагає в його аналізі.

3. Визначення ключових показників ефективності

Ключові показники ефективності – це ті показники, які зосереджені на аспектах діяльності організації, які є найважливішими для поточного та майбутнього успіху організації[2]. Загальний підхід до формулювання ключових показників ефективності такий:

1. визначити стратегічні цілі;
2. визначити результат, що буде успішним;
3. сформулювати показники ефективності за принципом SMART (конкретні, вимірювані, досяжні, доцільні та обмежені у часі).

Сформульовані показники повинні покривати такі властивості процесу: дієвість, ефективність, час проходження, якість результатів та відповідність нормативним вимогам. Дієвість в контексті процесу означає зв'язок між фактичними та очікуваними результатами. Ефективність є зв'язком між результатами, досягнутими в процесі та ресурсами, що були затрачені та спожиті. Відповідність процесу поділяється на внутрішню та зовнішню. Внутрішня визначає чи досягнуто в результаті

виконання процесу бажаного результату (або ж частка досягнутого результату). Зовнішня відповідність – контролює узгодженість зі стандартами та наявними нормами. Час проходження – очевидний показник, він визначає часові витрати на виконання процесу, враховуючи всі етапи та підпроцеси і їх зв'язки. Якість результатів процесу – чи отримані результати відповідають очікуванням їх споживача, внутрішнім вимогам, в тому числі й бюджетним. Саме ці аспекти й визначають статус процесу, допомагають в прогнозі його результатів і можливих відхилень.

4. Прогнозування процесів

Одним з важливих етапів у прогнозуванні процесів є пошук «вузьких місць». Для реалізації цього пошуку, а також оцінки тривалості процесу пропонується розглядати класичний метод критичного шляху із модифікацією того, що етапом процесу може бути також іпідпроцес. В результаті роботи алгоритму будемо отримувати ланцюг етапів, що є критичним і знатимемо скільки часу витрачатиметься на процес.

Можливим способом прогнозування протікання процесів є побудова дифузійних моделей. Дифузія є процесом, з допомогою якого населення приймає інновації. Відповідними прикладами є фіксований телефонний зв'язок чи сервіси відеоконференцій. Процес поширення характеризується спочатку впровадженням інновації, згодом відбувається повільний ріст, який пропорційний підвищенню ознайомлення з технологією. Ріст пришвидшується до точки, де досягає максимуму за період, а потім сповільнюється, бо населення насичується інноваціями[3]. Дифузійні моделі, що використовуються для прийняття інновацій, завжди є спрощенням процесу, оскільки доступний набір даних, як правило, обмежений і врахування усіх факторів теж є неможливим[4].

Добре вивченим підходом до прогнозування процесів є використання часових рядів. Використання цього апарату може допомогти з плануванням людських ресурсів, що будуть витрачатися під час виконання процесу. Проблема є актуальною, наприклад, в центрах обробки викликів (call-центрах) на короткі, середні та тривалі часові відрізки. Оскільки витрати на персонал складають близько двох третіх експлуатаційних витрат call-центру, то ефективне планування людських ресурсів залежить від точності прогнозування. Відповідні прогнози можуть охоплювати цілий ряд часових відрізків, від кількох місяців до тижня, а можливо і до оновлення щоденних прогнозів в режимі реального часу [5]. Практична цінність точності прогнозування показників плину даних процесів є важливою для прийняття управлінських рішень. Недооцінка частоти викликів приводить до малої кількості персоналу, що може призвести до критичних ситуацій чи значних фінансових втрат. Наслідком переоцінки є також значні фінансові втрати.

Висновки

Розроблено основні етапи аналізу і прогнозування протікання процесів в інформаційно-комунікаційних системах, показано можливі варіанти класифікації процесів із використанням різних підходів (за ступенем автоматизації, за впливом на діяльність організації, глобальний підхід (по суті процесу). Запропоновано підхід до формулювання ключових показників ефективності з урахуванням дієвості, ефективності, часу проходження, якості результатів та відповідності процесу. Запропоновано використовувати метод дифузії та моделі й методи прогнозування часових рядів для застосування при прогнозуванні інформаційно-комунікаційних процесів.

Перелік посилань

1. APQC Process Classification Framework (PCF) – Cross Industry [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.apqc.org/system/files/Cross%20Industry%20v7.2.1_Final.pdf.

2. Parmenter D. Keyper for mance indicators : developing, implementing, andusingwinningKPIs / DavidParmenter. – Hoboken: Wiley, 2015. – 448 с. – (Thirdedition).

3. Meade N. Forecasting in telecommunication sand ICT—A review / N. Meade, I. Towhidul. // InternationalJournalofForecasting. – 2015. – С. 1105–1126.
4. Казанцев С. Ю. Использование диффузионной модели в прогнозировании долей рынка (на примере развития сетей сотовой святи стандартов GSM и cdma2000) / Сергей Юрьевич Казанцев. // Научныетруды: Институтнароднохозяйственногопрогнозирования РАН. – 2005. – С. 248–260.
5. Pesaran M. Forecasting Time Series Subjectto Multiple Structural Breaks / M. Pesaran, D. Pettenuzzo, A. Timmermann. // The Review of Economic Studies. – 2006. – С. 1057–1084.

УДК 004.78

МУСІЄНКО В.С.
КОВТУНЕЦЬ О.В.

МЕТОД КОЛАБОРАТИВНОЇ ФІЛЬТРАЦІЇ НА ОСНОВІ ПОДІБНОСТІ ЕЛЕМЕНТІВ

У даній статті наводиться класифікація методів колаборативної фільтрації, їх короткий опис, порівняння, переваги та недоліки. Також описані основи методу видачі рекомендацій по подібності елементів (Item-based), наведеніприкладироботи алгоритму. КЛЮЧОВІ СЛОВА: колаборативна фільтрація, подібність предметів, система оцінювання, система рекомендацій.

This article describes the classification of collaborative filtering methods, their brief description, comparisons, advantages and disadvantages. Also described are the basics of the method of issuing recommendations for the similarity of elements (Item-based), examples of the algorithm.
KEYWORDS: collaborative filtering, subject similarity, ratingsystem, recommendation system.

1.Вступ

Кожної миті в інтернеті з’являється якась нова інформація, і в цьому безперервному потоці контенту людині може бутискладно знайти те, що цікаве саме їй. Для вирішення подібних проблем були створені рекомендаційні системи – програми, які намагаються передбачити, які об’єкти (фільми, книги, музика, новини тощо) будуть цікаві користувачу. Передбачення грунтуються на певній інформації про кожного конкретного користувача, яка зазвичай міститься в профілі користувача. Такі системи порівнюють однотипні дані від різних людей і вираховують список рекомендацій для конкретного користувача. Одним із методів побудови прогнозів у рекомендаційних системах є колаборативна фільтрація. Саме про неї піде мова у даній статті.

2. Колаборативна фільтрація

Колаборативна фільтрація – це метод, що дає автоматичні прогнози відносно інтересів користувача по зібраній інформації про вподобання більшості користувачів. Його основне припущення полягає в наступному: ті, хто однаково оцінив предмети в минулому, схильні давати схожі оцінки іншим предметам у майбутньому. Наприклад, колаборативна фільтрація по музикальних вподобаннях може передбачити, яка музика сподобається користувачі, маючи список його інтересів. Ці прогнози індивідуальні, хоча використовувана інформація зібрана від багатьох учасників. Вони відрізняються від простішого підходу, який дає середню оцінку для об’єкта.

3. Основні методиколаборативної фільтрації

1) Методи, що використовують аналіз оцінок предметів (Memory-based). Такі методи базується на статистичних способах підбору групи користувачів, близьких до даного користувача. Тут використовуються попередні оцінки, які зробив даний користувач, і аналіз оцінок інших користувачів. Ціметоди ще називають методами найближчих сусідів.
2) Методи, що використовують аналіз моделі даних (Model-based). В цьому випадку спочатку по сукупності оцінок формується

модель інтересів користувачів, товарів і взаємозв'язків між ними, а потім формуються рекомендації на основі отриманої моделі.

3) Методи, що використовують поєднання корисних властивостей попередніх двох груп – гібриднімеотоди.

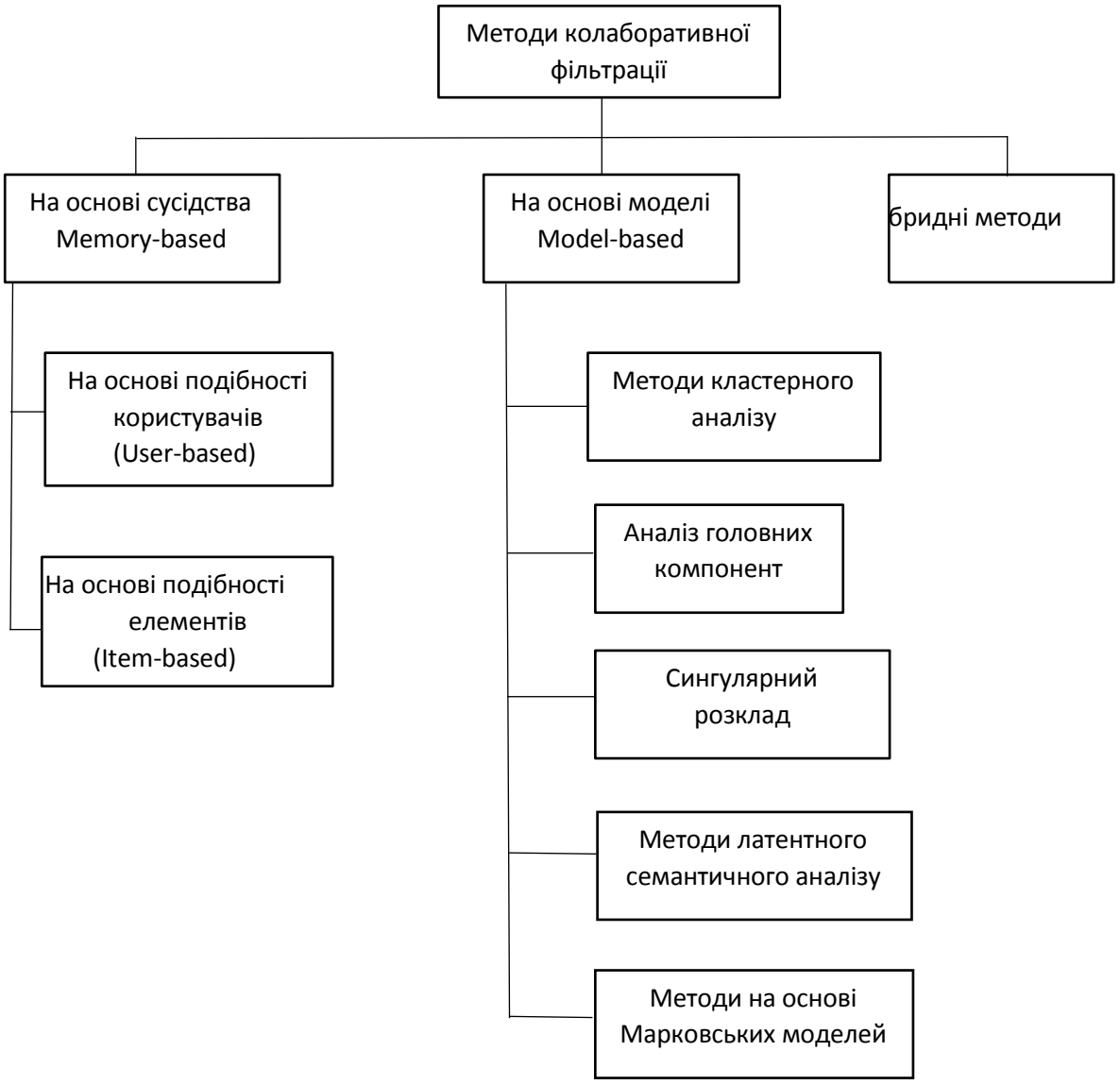


Рис. 1. – Класифікація методів у колаборативній фільтрації

4. Визначення та позначення

U–множина суб'єктів (клієнтів, користувачів: users);

R–множина об'єктів (ресурсів, товарів, предметів: items);

Y – простір описів транзакцій;

$D = (u_i, r_i, y_i)_{i=1}^{m} \in U \times R \times Y$–транзакційні дані;

Агреговані дані:

$F = f_{ur}$— матриця крос-табуляції розміру |U| × |R|, де$f_{ur} = aggr\{(u_i, r_i, y_i) \in \mathrm{D} \mid u_i = u, r_i = r\}$

Задачі:

- Прогнозування незаповнених комірок f_{ur}
- Оцінювання подібності: $p(u, u')$, $p(r, r')$, $p(u, r)$

5. Опис підходів на основі сусідства

- Тривіальна рекомендаційна система:

«Клієнти, які купували товар r_0, також купували $R(r_0)$»

1) $U(r_0) := \{u \in U | f_{ur_0} \neq \emptyset, u \neq u_0\}$ – колаборація;

2) $R(r_0) := \{r \in R | B(r) = \left|\frac{U(r_0) \cap U(r)}{U(r_0) \cup U(r)}\right| > 0\}$;

де $B(r)$ - одна із можливих пір близькості rі r_0;

3) Відсортувати $R(r_0)$ по спаданню $B(r)$, взяти topN;

Недоліки:

- Рекомендації тривіальні (пропонується все найбільш популярне)
- Не враховуються інтереси конкретного користувача u_0
- Проблема «холодного старту» (новий товар нікому не рекомендується)
- Потрібно зберігати всю матрицю F

- Фільтрація по подібності користувачів (user-based)

«Клієнти, схожі на u_0, також купували$R(u_0)$»

1) $U(u_0) := \{u \in U | corr(u, u_0) > \alpha\}$ – колаборація;

де $corr(u, u_0)$ – одна із можливих мір близькості uдо u_0;

2) $R(u_0) := \{r \in R | B(r) = \left|\frac{U(u_0) \cap U(r)}{U(u_0) \cup U(r)}\right| > 0\}$;

де $U(r) := \{u \in U | f_{ur} \neq \emptyset\}$;

3) Відсортувати $r \in R(u_0)$ по спаданню $B(r)$, взяти topN;

Недоліки:

- ~~Рекомендації тривіальні~~
- ~~Не враховуються інтереси конкретного користувача u_0~~
- Проблема «холодного старту»
- Потрібно зберігати всю матрицю F
- Нічого рекомендувати нетиповим/новим користувачам

- Фільтрація по подібності предметів (item-based)

«Разом з об'єктами, які купував клієнт u_0, часто купують $R(u_0)$»

1) $R(u_0) := \{u \in R | \exists r_0 : f_{u_0 r_0} \neq \emptyset, B(r) = corr(r, r_0) > \alpha\}$;

де$corr(r, r_0)$ – одна із можливих мір близькостіrдо r_0;

2) Відсортувати $r \in R(u_0)$ по спаданню $B(r)$, взяти topN;

Недоліки:

- Рекомендації часто тривіальні (немає колаборативності)
- Проблема «холодного старту»
- Потрібно зберігати всю матрицю F
- ~~Нічого рекомендувати нетиповим/новим користувачам~~

6. Метод на основі подібності елементів

Якщо нам доступні оцінки предмета, то з допомогою колаборативної фільтраціїможна передбачити оцінку, яку дасть користувач новому предмету на основі його попередніх оцінок і бази даних оцінок інших користувачів. В цьому випадку метод колаборативної фільтраціїза подібністю елементів передбачає оцінку предмета на основі оцінок іншого предмета, використовуючи найчастіше регресійний аналіз $(f(x) = ax + b)$. Відповідно, якщо змінюється 1000 предметів, то для вивчення можемо отримати до 1000000 лінійних регресійі до 2000000 регресорів. Такий підхід може виявитись неефективним, тому необхідно вибрати пари предметів, для яких відомі оцінки деяких користувачів.

Найкращою альтернативою може бути використання спрощеного предикатора (наприклад, $f(x) = ax + b$):
експериментально показано, що використання такого простого предикатора іноді перевершує регресивний аналіз, маючи при цьому вдвічі менше регресорів. В цього способу низькі вимоги до пам'яті і більша швидкість.

Колаборативна фільтрація за подібністю предметів – це один із багатьох видівколаборативної фільтрації. У випадку використання колаборативної фільтрації заподібністю користувачів аналізуються відношення між користувачами, визначається подібність їх інтересів. Але фільтрація за подібністю предметів потребує менше ресурсів і має більшу ефективність при наявності великої кількості користувачів.

Для визначення подібності користувачів або елементів можна використати такі підходи:

- відстань Евкліда, Хеммінга кореляція Пірсона

- рангова кореляція Спірмена
- коефіцієнт Жаккара
- косинусна схожість

В даній статті ми розглянемо останній із перелічених підходів.

1) Колаборативна фільтрація за предметами на основі статистик покупок

Далеко не завжди у користувачів є можливість виставляти оцінки предметам, тому для колаборативної фільтрації можуть бути доступні тільки двійкові дані (купував користувач предмет чи ні). В таких випадках алгоритми, які залежать від оцінок предметів, є неефективними. Прикладом алгоритму колаборативної фільтрації, який працює з двійковими даними, є алгоритм Item-to-Item. Він розраховує подібність предметів як косинус між векторами покупок в матриці користувачів і предметів:

$$similarity(\vec{A}, \vec{B}) = cos(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| * \|\vec{B}\|}$$

Табл. 1 – приклад матриці покупок

Клієнт	Товар 1	Товар 2	Товар 3
Василь	Купив	Не купив	Купив
Петро	Не купив	Купив	Купив
Олег	Не купив	Купив	Не купив

В цьому випадку косинус між «Товар 1» і «Товар 2» розраховується наступним чином:

$$\frac{(1,0,0) \cdot (0,1,1)}{\|(1,0,0)\| * \|(0,1,1)\|} = 0$$

Між «Товар 1» і «Товар 3»:

$$\frac{(1,0,0) \cdot (1,1,0)}{\|(1,0,0)\| * \|(1,1,0)\|} = \frac{1}{\sqrt{2}} \approx 0.71$$

І між «Товар 2» і «Товар 3»:

$$\frac{(0,1,1) \cdot (1,1,0)}{\|(0,1,1)\| * \|(1,1,0)\|} = \frac{1}{2} = 0.5$$

Таким чином, користувач, що знаходиться на сторінці «Товару 1», отримає «Товар 3» в якості рекомендації; на сторінці «Товару 2» - «Товар 3» і на сторінці «Товару 3» - «Товар 1» (і потім «Товар 2»).

Щоб застосувати алгоритм SlopeOne для заданих n предметів, необхідно розрахувати і

2) Колаборативна фільтрація для предметів з оцінками

Для зменшення ефекту перенавчання, збільшення продуктивності і спрощення застосування було запропоновано сімейство алгоритмів SlopeOne. Головна відмінність від регресійного аналізу відношення рейтингів двох предметів (f(x) = ax+b) поляшає у використанні спрощеної форми регресії усього з одним предикатом (f(x) = x+b). Таким чином, предикатор – це просто середня різниця між оцінками обох предметів. Автори продемонстрували, що такий підхід в деяких випадках більш точний, ніж лінійна регресія і потребує вдвічі менше пам’яті.

Табл. 2 – приклад матриці оцінок

Клієнт	Товар 1	Товар 2	Товар 3
Василь	5	3	2
Петро	3	4	-
Олег	-	2	5

Згідно з цією таблицею, середня різниця в оцінках предмета 1 і 2 рівна $\frac{2+(-1)}{2}$ = 0.5. Таким чином, в середньому предмет 1 оцінюється на 0.5 бали вище, ніж предмет 2. Аналогічно і для предметів 3 і 1: середня різниця в оцінках 3.

Якщо ми спробуємо передбачити оцінку Олега для товару 1, використовуючи його оцінку для товару 2, ми отримаємо 2+0.5 = 2.5. Аналогічно передбачаємо його оцінку для товару 1, використовуючи оцінку, яку він дав товару 3: 5+3 = 8. Оскільки у нас є кілька можливих оцінок (Олег голосував двічі), результуючу оцінку ми отримаємо як зважене середнє. Для вагових коефіцієнтів будемо використовувати кількість користувачів, що дали оцінку товару:

$$\frac{2 \cdot 2.5 + 1 \cdot 8}{2 + 1} = \frac{13}{3} = 4.33$$

зберегти середню різницю і кількість голосів для кожної з n^2 пар предметів.

Висновки

У даній статті було описано основні підходи колаборативної фільтрації та розглянуто приклади використання методу на основі подібності елементів. Кожен із підходів має свої недоліки та переваги, які потрібно враховувати при виборі алгоритму для впровадження його

в систему. Потрібно враховувати такі фактори, як: об’єм доступних ресурсів, допустимий час роботи алгоритму, кількість клієнтів, кількість товарів тощо.

Методи колаборативної фільтрації широко використовуються в комерційних сервісах та соціальних мережах. Основний сценарій використання – це створення рекомендацій відносно цікавої і популярної інформації на основі врахування «голосів» користувачів. Інший популярний сценарій використання полягає у створенні персоналізованих рекомендацій для користувача на основі його попередньої діяльності і даних про інтереси схожих на нього користувачів. Даний спосіб реалізації можна знайти на таких сайтах, як YouTube, Last.fm та Amazon.

Загалом методи колаборативної фільтрації варто широко використовувати через їхню простоту реалізації тадоступніть оцінювання отриманих результатів.

Список літератури

1. Amazon.com, ―Q4 2009 Financial Results,‖ Earnings Report Q4-2009, January 2010.
2. Понизовкин Д.М. Построение оптимального графа связей в системах коллаборативной фильтрации (рус.) // «Программные системы: теория и приложения» : журнал. — 2011. — № 4(8). — С. 107-114. — ISSN 2079-3316.
3. M. vanSetten, S. Pokraev, and J. Koolwaaij, ―Context-aware recommendations in the mobil etourist application compass,‖ Heidelberg, 2004, vol. 3137, pp. 515–548
4. J.S. Breese, D. Heckerman, and C. Kadie, ―Empirical Analysis of Predictive Algorithms for Collaborative Filtering,‖ Proc. 14th Conf. Uncertainty in Artificial Intelligence, July 1998. 113 ISSN 1996-1588 Наукові праці ДонНТУ випуск 2 (18), Серія “Інформатика, кібернетика 2013 та обчислювальна техніка”
5. Xiaoyuan Suand Taghi M. Khoshgoftaar "A Survey of Collaborative Filtering Techniques A Survey of Collaborative Filtering Techniques" // Hindawi Publishing Corporation, Advancesin Artificial Intelligencearchive, USA : 2009. — С. 1-19.
6. G. Adomavicius На пути к новому поколению рекомендационных систем: обзор имеющихся систем и возможные инновации. IEEE Transactionson Knowledge and Data Engineering, Vol. 17, No. 6, Июнь 2005 Электронный ресурс: http://artpragmatica.ru/rs/in/pic/58-870-20061024072441-Toward_the_next_generation_of_recommender_systems.doc

УДК 004.91

МАРЧЕНКОВ І.Д.

МЕТОДИ ОФЛАЙН-РОЗПІЗНАВАННЯ РУКОПИСНОГО ТЕКСТУ НА ПРИКЛАДІ ЗАДАЧІ АВТОМАТИЧНОГО ВИЗНАЧЕННЯ СЕРЕДНЬОГО БАЛУ ДОДАТКУ ДОАТЕСТАТА АБІТУРІЄНТА

В роботі розглянуті проблеми, які виникають при обробці зображень додатків до атестатів абітурієнтів під час вступної кампанії та наведено один із способів офлайн-розпізнавання рукописного тексту.Представлено алгоритм попередньої обробки зображення та можливих методів розпізнавання.

КЛЮЧОВІ СЛОВА: Обробка зображень, машинне навчання, класифікація документів, оптичне розпізнавання символів, розпізнавання рукописного тексту, горизонтальний профіль проекції.

This paper examines the problems that arise when processing images of applicants for admission during the introductory campaign, and provides one way of offline handwriting recognition. An algorithm for image pre-processing and possible recognition methods is presented.

KEY WORDS: Image processing, machine learning, document classification, optical character recognition, handwritingrecognition, horizontal projection profile.

1. Введення

У даній статті розглянуто методи офлайн-розпізнавання рукописного тексту, які використані для автоматичного визначення середнього балу додатку доатестата (далі по тексту – просто додаток) абітурієнта. Розглянуті проблеми, які виникають при обробці зображень додатків абітурієнтів під час вступної кампанії. Наведені приклади зображень додатків і зроблено висновки про можливість автоматичного визначення середнього балу на підставі сканованих копій.

2. Розпізнавання тексту

Переклад зображень рукописного, машинописного та печатного тексту в текстові дані називається розпізнаванням тексту. Задача розпізнавання машинописного тексту має назву оптичного розпізнавання символів (*opticalcharacterrecognition, OCR*)[1]. В даний час існують високоточні системи для розпізнавання машинописних і руко-друкованих текстів (наприклад, FineReader фірми ABBYY). Розпізнавання ж рукописних текстів являється набагато складнішою і на даний момент не вирішеною задачею.Задача розпізнавання рукописного тексту носить назву HWR (*handwritingrecognition*). Існує два класи задач HWR:

- онлайн-розпізнавання – розпізнавання тексту ведеться паралельно із введеннямтексту;
- офлайн-розпізнавання – розпізнавання тексту ведеться на вже синтезованому зображенні.

При онлайн-розпізнаванні процес формування зображення тексту і процес введення в систему розпізнавання суміщені, що дозволяє системі відслідковувати процес креслення символів. Це дає можливість отримувати крім графічної інформації ще й інформацію про структуру вхідних зображень, наприклад, про направлення і швидкість руху пера або про його натиск при написанні символу. На даний момент онлайн-системи розпізнавання широко використовуються на планшетних ПК. У завданні офлайн-розпізнавання, на відміну від попереднього, системі доступна тільки графічна інформація.Вже це робить її значно важче онлайнової. Неповний список проблем, типових при розпізнаванні рукописного тексту офлайн в загальному випадку, включає в себе:

- висока варіативність накреслення символів за розміром, нахилу, набору зіставних частин, зв’язків між ними та ін .;
- орфографічні помилки в тексті;
- специфічні особливості накреслення, чи не що дозволяють впевнено розділяти символи;
- перетин елементів тексту, накладення частин тексту один на одного;
- помарки, плями, виправлення, дефекти носія (паперу), а також артефакти, що виникають при скануванні;
- непаралельність рядків.

На даний момент комерційні пакети HWR можуть впевнено розпізнавати тільки машинозчитувані форми (анкети, заповнені бланки тощо), оскільки при використанні таких структурованих документів і зменшенні діапазону можливих символів, що вводять якість розпізнавання різко підвищується. До цього випадку відносяться розпізнавання поштових адрес при автоматичному сортуванні пошти, підписів на чеках, цифр та ін. Стандартне офлайн-розпізнавання рукописного тексту проводиться за наступною схемою:

1) попередня обробка зображення, виділення області інтересу;
2) сегментація і нормалізація тексту з області інтересу;
3) розпізнавання сегментованого тексту тим чи іншим методом.

3. Постановка проблеми аналізу додатку до атестата абітурієнта

Визначення середнього балу додатку абітурієнта, як складової його загального рейтингу є однією з трудомістких задач при обробці заяви, що безпосередньо впливає на кінцеве віднесення абітурієнта до рейтингу [2]. Велика кількість помилок при визначенні середнього балу додаткуна рівні шкіл (найчастіше пов'язаних з ігноруванням оцінки результатів державної атестації (іспитів)) привела до перекладання основної роботи за визначенням середнього балу на вищі навчальні заклади (ВНЗ). Це вимагає

необхідності автоматичного визначення балу для великих ВНЗ.

Додаток є уніфікованим документом, але при аналізі зображень визначено набір проблем, які ускладнюють можливість його обробки, а саме:

1) ряд абітурієнтів завантажують не додаток, а відмінні від нього документи: власне фото, сам атестат або інші документи;

2) фонове обрамлення – багато хто з завантажених зображень отримані шляхом не сканування документа, а фотографування(рис. 1);

3) розташування декількох документів на одному зображенні (рис. 2);

4)розташування на зображенні – зустрічаються як представлені в книжковій, так і в альбомній орієнтації.

5) обрізка зображення – велика кількість сканованих зображень представлена на аркуші формату А4 в натуральну величину без обрізки.

6) колір – на відміну від сканованих копій, де колір додатку близький до ціан, в залежності від освітлення зйомки, зустрічаються зображення з кольорами близькими до жовтого або червоного (рис. 3).

Таким чином, автоматичне визначення середнього балу додатку абітурієнта неможливо без попередньої обробки і класифікації документів.

Залежно від певних проблем або їх сукупності, послідовність дій для автоматичної обробки зображення і подальшого його розпізнавання може бути вкрай важким алгоритмічним завданням.

4. Попередня обробка зображення

На етапі попередньої обробки зображення виконуються наступні завдання:

1. Підвищення якості зображення (preprocessing), що виконується методами обробки зображень (фільтрація, шумозаглушення і інші) і має своєю метою підвищити якість зображення;

2. Виділення області інтересу на зображенні, що використовує методи аналізу зображень і має на меті позбутися нетекстової інформації (наприклад, плями, помарки, зображення в тексті та інші).

4.1. Препроцесінг

На цьому етапі відбувається очищення зображення від дефектів сканування. Здійснюється це стандартними методами обробки зображень (наприклад, застосуванням різних фільтрів). Важливу роль в препроцесінгу займає т.зв. порогова бінаризація (thresholdbinarization) – переведення зображення в чорно-білий формат з кольорового або відтінків сірого.

4.2. Виділення регіону інтересу

На цьому етапі на бінаризованому зображенні виділяється безпосередньо область, в якій знаходиться текст, що розпізнається, і відкидаються елементи, які не є текстом. До них відносяться такі об'єкти, як ляпи, плями на папері, не видалені в процесі бінаризації, картинки та ін. Для їх видалення можна, наприклад, виділяти компоненти зв'язності на зображенні, обчислювати геометричні признаки і на їх основі класифікувати компоненту зв'язності як частину тексту або дефект, використовуючи методи машинного навчання або евристики.

5. Сегментація і нормалізація тексту

На цьому етапі текст розділяється, або сегментується, на зручні для аналізу складові частини. Найбільш природними процесами на даному етапі є розділення тексту на окремі рядки (сегментація рядків) і поділ рядків на слова (сегментація слів), а також, теоретично, поділ слів на елементарні складові частини. Крім того, на даному етапі проводитьсянормалізація текступриведення виділених складових частин до деякого стандартного виду для зниження варіативності і спрощення розпізнавання.

5.1. Сегментація рядків

Задача сегментації (поділу) рядків в машино-друкованих документах на нинішній час вважається повністю вирішеною [3]. Але в задачі HWR при поділі рядків у загальному випадку виникають складності, що не дозволяють безпосередньо застосовувати алгоритми, придатні для машино-друкованих текстів [4]:

1) рядки не тільки можуть не бути паралельними, а й можуть згинатися;

2) різні рядки можуть бути занадто близькі, а елементи тексту, належати різним рядкам, можуть накладатися один на одного.

Для поділу рядків в додатку можна скористатися методом горизонтальної проекції [5].Цей метод використовує «top-down» підхід і широко використовується в

задачі поділу рядків в машино-друкованих документах. Полягає він в наступному: для зображення рахується так званий горизонтальний профіль проекції (*HPP*, *horizontalprojectionprofile*), який представляє собою суму всіх пікселів зображення вздовж горизонтального направлення, потім на ньому знаходяться локальні мінімуми. Так як розглядається бінарне зображення, на якому пікселі тексту мають значення 1, а пікселі фону 0, ці мінімуми будуть відповідати міжрядковим інтервалам.

5.2. Сегментація слів

На цьому етапі роботи системи розпізнавання виділені рядки тексту поділяються на окремі слова. На відміну від машино-друкованого тексту, в якому відстань між словами більш-менш постійна, а інтервали між символами всередині слова набагато менше, ніж інтервали між словами, в рукописному тексті розмір інтервалів між словами може варіюватися в дуже широких межах.

Можна поставити альтернативну задачу: визначити, чи є пробіл між компонентами зв’язності тексту пропуском між словами або інтервалом між буквами в слові, і вирішувати цю задачу методами класифікації. Оскільки відстань між частинами одного слова зазвичай менше, ніж між словами, поширеність використання так званих метрик розриву (gapmetric): для інтервалу між компонентами зв'язності обчислюється деяка величина і порівнюється з порогом.

5.3. Нормалізація

В силу високої варіативності накреслення слів їх розпізнавання є дуже складним процесом. Нормалізація служить для приведення слова до деякого стандартного виду без значної втрати інформації, необхідної для розпізнавання. Одними з найбільш часто вживаних методів нормалізації є «*slopecorrection*» і «*slantcorrection*».

Slope– це величина (кут) відхилення слова від горизонтальної лінії. *Slant*– величина відхилення тих елементів слова, що повинні бути вертикальними, від, відповідно, вертикалі [6]. Зрозуміло, що кут нахилу слова на його зміст не впливає, але може погіршувати його розпізнавання. Найпростіший метод корекції *slope* полягає у виконанні наступних кроків:

1) Ввести функціонал якості на горизонтальному профілі проекції зображення слова (наприклад, стандартне відхилення), що залежить від кута повороту зображення;
2) максимізувати його на деякому діапазоні кутів.

Фактично метод повторює метод горизонталізації зображення в задачі OCR. Принцип роботи можна побачити на рисунку 4.

6. Розпізнавання слова цілком

У випадку балів в додатку словник слів є достатньо обмеженим (дванадцять різних оцінок), тому можна спробувати розпізнати слово як є - витягти ознаки безпосередньо з усього слова і спробувати вирішити задачу класифікації. Але навіть в цьому випадку легко може виявитися, що ознак занадто багато, і змінюються вони в дуже великому діапазоні, щоб можна було провести впевнену класифікацію або навіть побудувати модель.

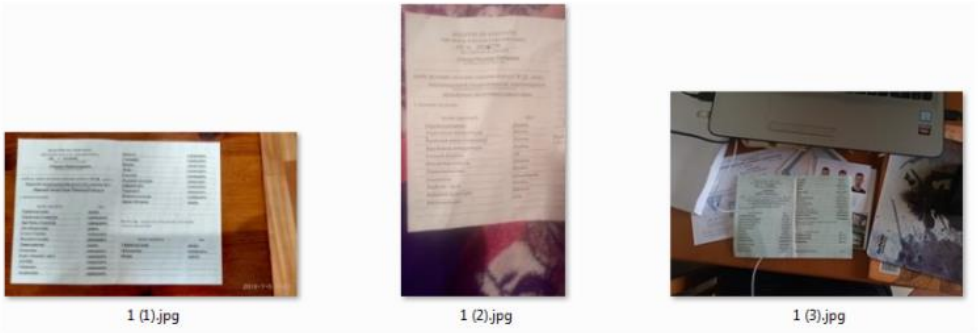


Рисунок 1 – Наявність фону

Рисунок 2 – Зображення з декількома документами

Рисунок 3 – Відтінки зображень

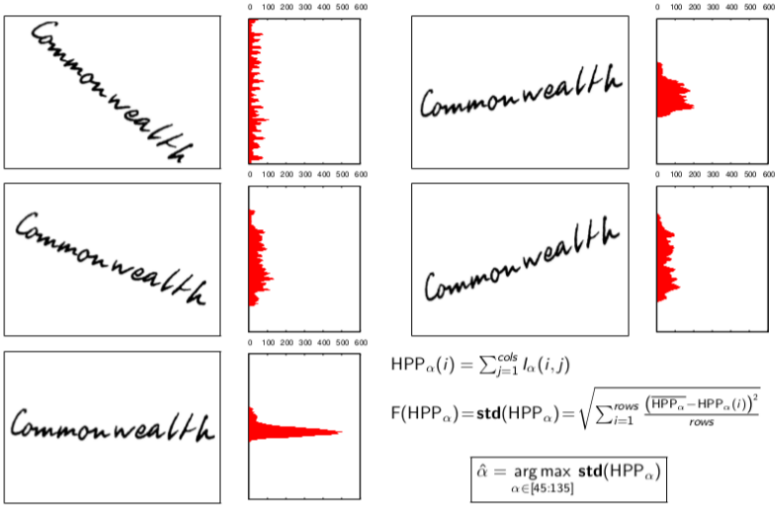


Рисунок 4 – Корекція slope з використанням горизонтального профілю проекції

Висновок

У статті було розглянуто спосіб офлайн-розпізнавання рукописного тексту на прикладі автоматичного визначення середнього балу додатку доатестата абітурієнта. Було виявлено перелік проблем визначення середнього балу, з якими стикаєтьсяВНЗ під час вступної кампанії і підкреслено, що установа має обмежену кількість часу для обробки документів. У наш час не існує ідеального рішення, яке вдало може розпізнавати рукописні тексти, принаймні у відкритому доступі. Особливо проблема гостро стосується української мови.

Для розпізнавання тексту спочатку його потрібно підготувати: перевести зображення в чорно-біле, виділити цільову область, сегментувати рядки, слова, символи, нормалізувати вигляд цих об'єктів, і тільки після цього приступати до розпізнавання.У випадку балів додатку до атестата кількість слів скінчена, тому маємо можливість розглядати кожне слово, як окремий об'єкт. Розпізнавати назви предметів не потрібно, оскільки це призведе до зменшення точності у кращому випадку, у гіршому – до непрацюючої моделі.Для навчання може бути використаний метод опорних векторів, а також багатошарові нейронні мережі.

Список літератури

1. Козлов В. Д. Методы оффлайн-распознавания рукописного текста / Козлов В. Д. // [Електронний ресурс], 2014. – Режим доступу: http://textolog-rgali.ru/userfiles/articles/article2/GK.pdf
2. Ладогубец Т. С. Особенностиавтоматическогоопределениясреднегобаллааттестатаабитуриента при обработкеизображенийаттестатов / Т. С. Ладогубец, П. Л. Литвиненко, Р. И. Сегол, А. Д. Финогенов // Сучасні проблеми моделювання. - 2019. - Вип. 15. - С. 118-127. - Режим доступу: http://nbuv.gov.ua/UJRN/cpm_2019_15_18.
3. Plamondon, R´ejean and Srihari, SargurN.Online and off-line handwritingrecognition: a comprehensive survey // Pattern Analysis and Machine Intelligence,IEEE Transactions on,22, 1, (63–84), 2000.
4. Li, Yi and Zheng, Yefeng and Doermann, David and Jaeger, Stefan.Script-independent text line segmentation in freestyle handwritten documents // PatternAnalysis and Machine Intelligence, IEEE Transactions on,30, 8, (1313–1329), 2008.
5. Likforman-Sulem, Laurence and Zahour, Abderrazak and Taconet, Bruno.Textlinesegmentation of historical documents: a survey // International Journal of DocumentAnalysis and Recognition (IJDAR),9, 2-4, (123–138), 2007.
6. Vinciarelli, Alessandro and Luettin, Juergen.A new normalization technique forcursive handwritten words // Pattern Recognition Letters,22, 9, (1043–1050), 2001.

УДК 001:004.91

МАМОНТОВ В.В.,
БЕРЕЗІНСЬКИЙ Г.В.

АНАЛІЗ НАУКОМЕТРИЧНИХ ДАНИХ ТА СКЛАДАННЯ РЕЙТИНГУ ПІДРОЗДІЛІВ КПІ ІМ. ІГОРЯ СІКОРСЬКОГО НА ОСНОВІ ПУБЛІЦИСТИЧНОЇ ДІЯЛЬНОСТІ ВИКЛАДАЧІВ

В даній роботі розглянуто методи застосування наукометричних даних для аналізу публіцистичної діяльності науковців на основі інформації з всесвітньо відомих ресурсів Google Scholar та Scopus. Представлено вирішення проблеми відсутності можливості спостерігати за показниками публіцистичної активності науковців у режимі реального часу.Побудовано актуальні рейтинги факультетів та кафедр КПІ ім. Ігоря Сікорського.
РЕЙТИНГ, НАУКОМЕТРІЯ, SCOPUS, GOOGLE SCHOLAR, H-ІНДЕКС, ІНДЕКС ХІРША.

This paper examines the methods of using scientometric data to analyze the publicisticactivity of scientists, based on information fromworld-known resourcesGoogle Scholar

and Scopus. Here shown the solution of an absence of possibility to observe the index of publication activity in real-time mode. The actual ratings of faculties and departments ofIgor Sikorsky Kyiv Polytechnic Institute were created.
RATING, SCIENTOMETRICS, SCOPUS, GOOGLE SCHOLAR,H-INDEX, HIRSCH INDEX, HIRSCH NUMBER.

1. Вступ

Проблемою оцінки науковців займається наукометрія – область наукознавства, що займається статистичними дослідженнями структури і динаміки наукової інформації.

Наукове співтовариство дозволяє позиціонувати вчених, дослідницькі центри, наукові організації в локальній та світовій наукових системах. Наукометричний аналіз цих об'єктів дає можливість оцінювати внесок дослідників як виробників інформації в світовий інформаційний масив, вивчати взаємозв'язки між окремими спільнотами.

Аналіз наукових знань дозволяє виявляти області, що швидко розвиваються, віддалені перспективи технологічних проривів, отримувати деякі уявлення про внутрішню структуру наукових досліджень, виявляти ті, що зароджуються, та перспективні напрямки, ухвалювати рішення для підтримки даних напрямків.

Часто наукометричні показники групують наступним чином: кількісні показники активності наукової діяльності (НД), якісні показники впливу публікацій на інформаційний масив і комплексні показники, що враховують кількісні та якісні критерії оцінки НД[1].

Фінансування ЗВО напряму залежить від цих показників і своєчасне отримання даних є запорукою контролю над ситуацією та попередження проблем, які можуть виникнути через недостатню публіцистичну діяльність окремих напрямків та/або підрозділів ЗВО[2].

2. Існуючі рейтинги

Найпопулярнішими ресурсами для агрегації наукових публікацій всіх форматів та дисциплін є Google Scholar, Scopus та Web of Science. Крім, власне, публікацій, кожен ресурс містить профілі активних наукових діячів, які включать у себе наукометричні показники. З-поміж названих систем Google Scholar є єдиною, що дозволяє науковцю зареєструватися власноруч та не вимагає фінансових внесків для початку роботи. На відміну від Scopus та Web of Science, журнали з публікаціями вчених не мають купувати членство у системі, тож Google Scholar містить помітно більшу кількість публікацій.

Наразі для аналізу наукометричних даних в КПІ ім. Ігоря Сікорського існує платформа webometr.kpi.ua, яка використовує зібрані вручну дані для періодичної створення рейтингу кафедр університету, а саме середніх рейтингових показників науково-педагогічних працівників кафедр за напрямами діяльності[3]. Також зазначений ресурс має інформаційний характер та займається створенням рейтингів сайтів кафедр, який базується на відвідуваності доменів, та не має нічого спільного з наукометричними показниками.

Серед недоліків такого підходу варто зазначити неможливість отримати актуальні дані у будь-який момент часу, складність збору даних, а також дуже вузький профіль проаналізованих даних – відсутня можливістьавтоматичного визначення рейтингупідрозділів, отримати інформацію по кожному окремому викладачу.

3. Сучасний спосіб вирішення проблеми

У даній роботібуде проведено аналіз на основі даних, отриманих від парсингу найпопулярніших сервісів для формування статистики публіцистичної діяльності науковців – Google Scholar та Scopus.

Доступ до повних даних для зручного їх аналізу за допомогою API може бути наданий самими ресурсами за підпискою. Проте вартість підписки, наприклад, на Scopus сягала приблизно 140 000$ на рік у 2019 році[4]. Зокрема, дані, що пропонуються сервісами по підписці, у більшості своїй є надлишковими навіть для досить широкого аналізу.

У цій роботі пропонується рішення, яке дозволить аналізувати всібазові наукометричні характеристик на основі даних, що надаються сервісами у публічному доступі. Можливості продукту будуть

показані на прикладі складання рейтингу підрозділів КПІ ім. Ігоря Сікорського.

4. Отримання даних

Для виконання аналізу над даними першочерговою задачею є отримання цих даних. Сервіси захищають інформацію на своїх серверах від DoS-атак, а також запобігають отриманню неліцензійного доступу до приватної інформації, що негативно відображається на складності парсингу. Для захисту даних використовуються наступні методи:

- регулярне оновлення структури сторінки сайту;
- періодичне оновлення зовнішнього вигляду сайту;
- блокування відображення повної інформації для ip-адрес, що не відносяться до учбових закладів;
- блокування ip-адрес при надходженні занадто великої кількості запитів за малий інтервал часу.

Для подолання цих перешкод було зроблено наступне:

- для кожного з сайтів написано надійний парсер, який практично не чутливий до оновлення структури веб-сторінок;
- парсер написано на скриптовій мові PHP, яка не потребує компіляції, та може бути з легкістю перебудований у подальшому при необхідності;
- зчитування даних здійснювалось за допомогою VPN, що імітувало надходження запитів з територіїКПІ ім. Ігоря Сікорського. Це дозволило отримувати повні дані незалежно від ip-адреси, з якої відбувався парсинг;
- для уникнення блокування сайтомвикористовується затримка, яка коливається в межах від 10 до 15 секунд при зчитуванні даних про кожного науковця.

5. Дані

Після успішного зчитування дані розміщуються у базі даних, що має структуру, як вказано на рисунку 1.

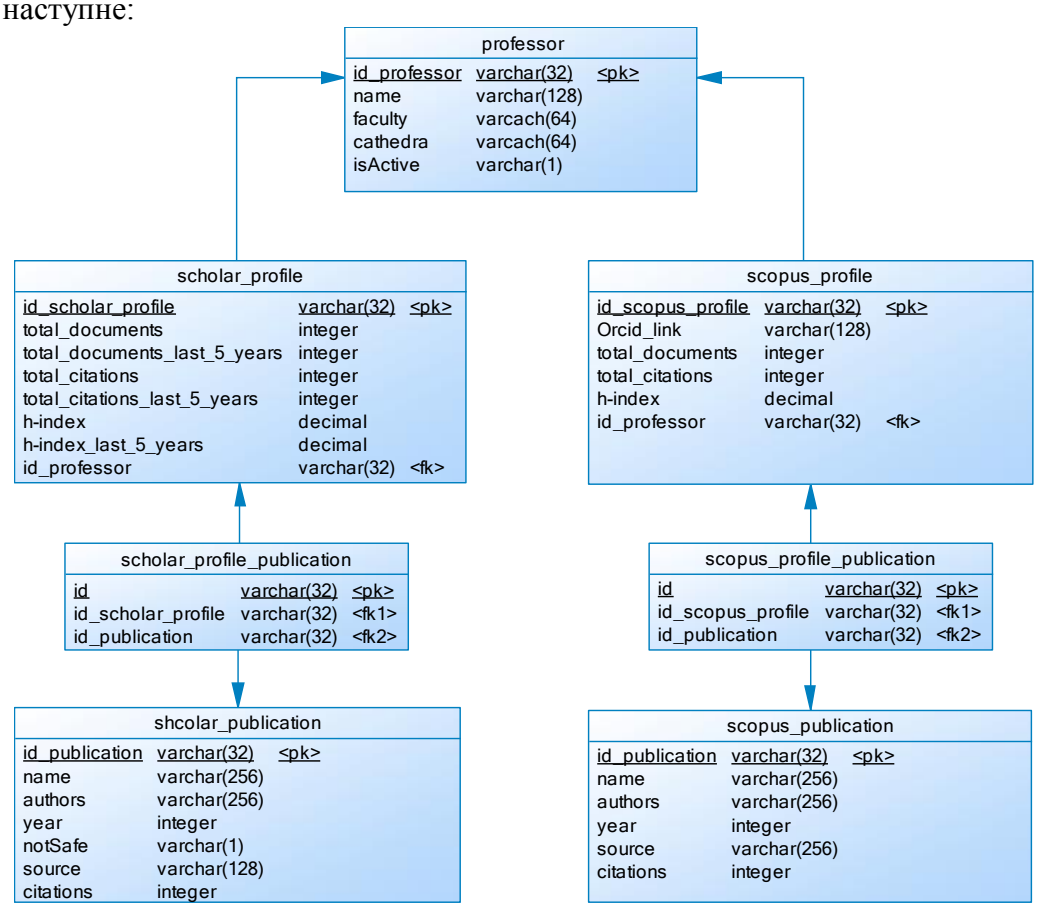


Рис. 1. Структура бази даних викладачів, їх профілів та публікацій

Головною таблицею бази даних є таблиця викладачів. На кожного з викладачів посилається його профільна сайті google.scholar.com та scopus.com. У теорії,

викладач може мати декілька профілів. Проте такі випадки повинні контролюватися адміністраторами і прийматися рішення про злиття та видалення зайвих профілів. Таблиці публікацій та профілів зв'язані зв'язком багато-до-багатьох. Звісно, кожним науковцем-власником профілю може бути видано багато статей. Проте також публікації часто мають понад одного автора, таким чином посилаючись на декілька профілів. Для прикладу – отримані після парсингу дані містили 75 660 наукових статей, взятих з ресурсу Google Scholar. І 10 017 з них мали два або більше співавтори, що належать до викладацького складу КПІ імені Ігоря Сікорського.

Всі унікальні ключі, що використовуються в базі даних, мають стандарт ідентифікації uuid, який призначений для забезпечення унікальності id у середовищі з відсутнім єдиним центром координації [5].

6. Аналіз даних

У отриманій базі даних містяться записи про 2693 викладачі та їх публікації. Загалом за актуальною інформацією у КПІ ім. Ігоря Сікорськогоналічується приблизно 3000 викладачів. Згодом база буде розширюватися до моменту, доки не буде включати актуальну інформацію про абсолютно всіх викладачів. Проте на даний момент достатньо інформації про близько 90% всіх викладачів, що дозволить провести аналіз та зробити адекватні висновки.

Проведемо аналіз публіцистичної діяльності підрозділі КПІ – факультетів та кафедр. Для цього розглянемо викладачів, що належать до зазначених підрозділів. Кількістю публікації підрозділу буде розрахована як сумарна кількість публікацій викладачів у підрозділі, а сумарна кількість цитувань цих публікацій буде розглядатися як загальна кількість посилань підрозділу.

За допомогою описаних даних ми маємо змогу дослідити співвідношення кількості публікацій до кількості посилань для кожного факультету, а також оцінити популярність платформ, що використовуються. Для зручності представимо описані дані у вигляді гістограмна рисунках 2-3.

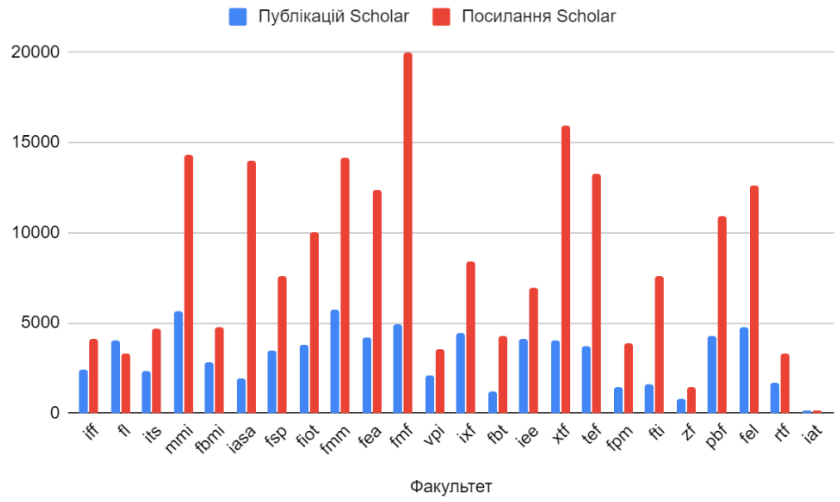


Рис. 2. Співвідношення кількості посилань та кількості публікацій для усіх факультетів у Google Scholar

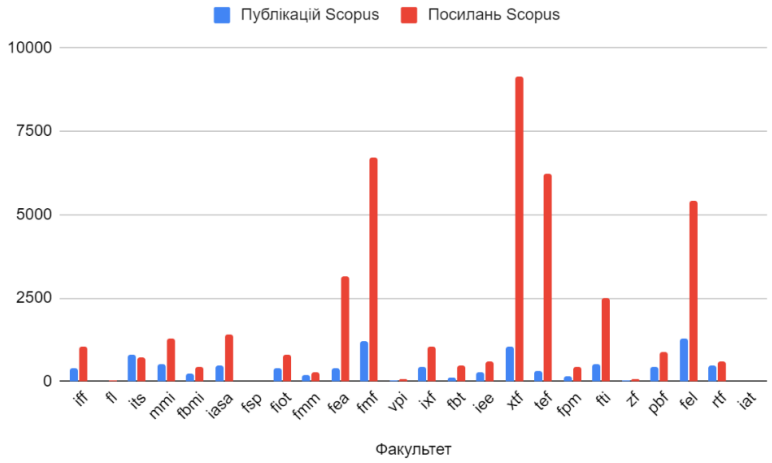


Рис. 3. Співвідношення кількості посилань та кількості публікацій для усіх факультетів у Scopus

Розглянемо отримані гістограми та порівняємо обидва джерела даних. Можна бачити, що кількість наукових робіт,представленихуу Google Scholar, є значно більшою. Таку різку відмінність у показниках діяльності науковців можна пояснити тим, що публікація, яка буде відображена на Scopus, потребує виконання більшого числа формальностей, а також матеріальних витрат, аніж публікація, що увійде до бази Google Scholar. Крім цього, Scopus враховує лише публікації, що були зроблені у журналах, що були попередньо зареєстровані у базі на комерційній основі. В той самий час, Google Scholar враховує публікації у будь-яких відомих йому перевірених джерелах.

Можна помітити, що більша частина факультетів, які займають перші позиції по кількості посилань на публікації на Google Scholar, зберігають позиції і на Scopus. Серед цих факультетів ФММФ, ХТФ, ТЕФ та ФЕЛ. Серед факультетів, що помітно втратили при переході відGoogleScholar на Scopus можна виділити ІПСА (13995 посилань у GoogleScholar проти 1403 у Scopus), ММІ (14291 посилань у Google Scholar проти 1293 у Scopus), ФММ (14147 посилань уGoogle Scholar проти 277 у Scopus) та ПБФ (10905 посилань уGoogle Scholar проти 895 у Scopus). Ця різниця є досить суттєвою та складає від приблизно 10 (ІПСА) до 50 (ФММ) разів.

Цікавим є спостереження: факультет лінгвістики – єдиний факультет за базою Google Scholar, у якому кількість публікацій переважає за кількість посилань на ці ж самі публікації. Результатом цього буде найнижчий показник h-індексу з-поміж всіх факультетів, а саме 1,66.

Детальніше різницю між активністю на цих двох платформах можна побачити на гістограмах, що представлені на рисунках 4 та 5.

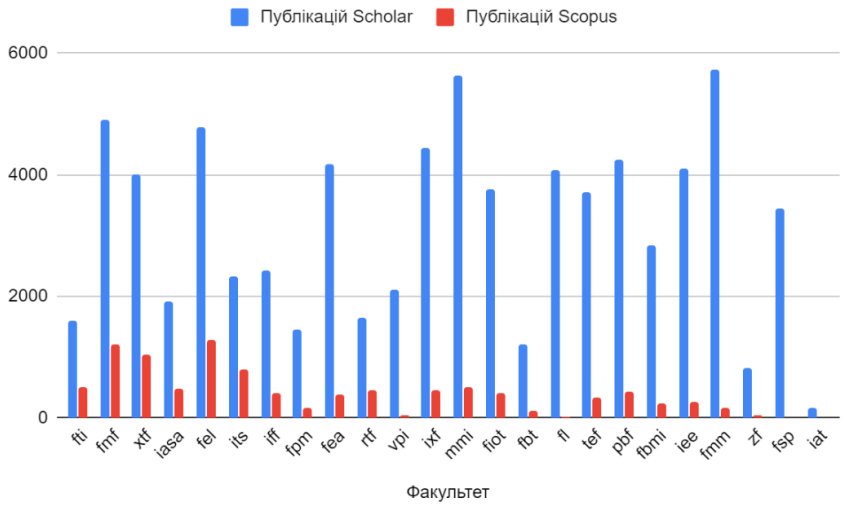


Рис.4. Порівняння кількості публікацій вGoogle Scholar та Scopus

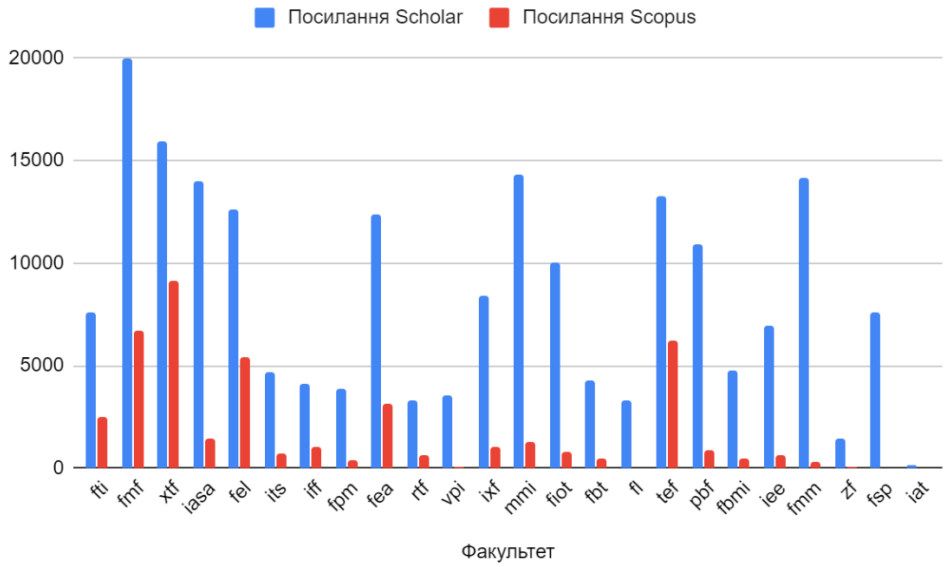


Рис. 5. Порівняння кількості посилань вGoogle Scholar та Scopus

Попри все, основним показником, за якими здійснюється аналіз публіцистичної діяльності викладачів, є h-індекс, або індекс Хірша. Індекс Хірша є кількісною характеристикою продуктивності вченого, заснованої на кількості його публікацій і кількості цитувань цих публікацій. Індекс обчислюється на основі розподілу цитувань робіт даного дослідника. У цей показник закладено наступний зміст: вчений має індекс h, якщо h з його Np статей цитуються як мінімум h раз кожна, в той час як решта (Np-h) статей цитуються не більше, ніж h раз кожна[4].

Для порівняння підрозділів за індексом Хірша необхідно визначити цей індекс для кожного з досліджуваних підрозділів. Індексом Хірша для кожного підрозділу буде середнє значення індексу серед всіх його вчених. На основі середнього h-індексу проведемо оцінку наукової діяльності підрозділу та побудуємо гістограму для відображення h-індексу для всіх факультетів (рисунок 6). Будемо враховувати дані, взяті зі Scopus та Google Scholar.

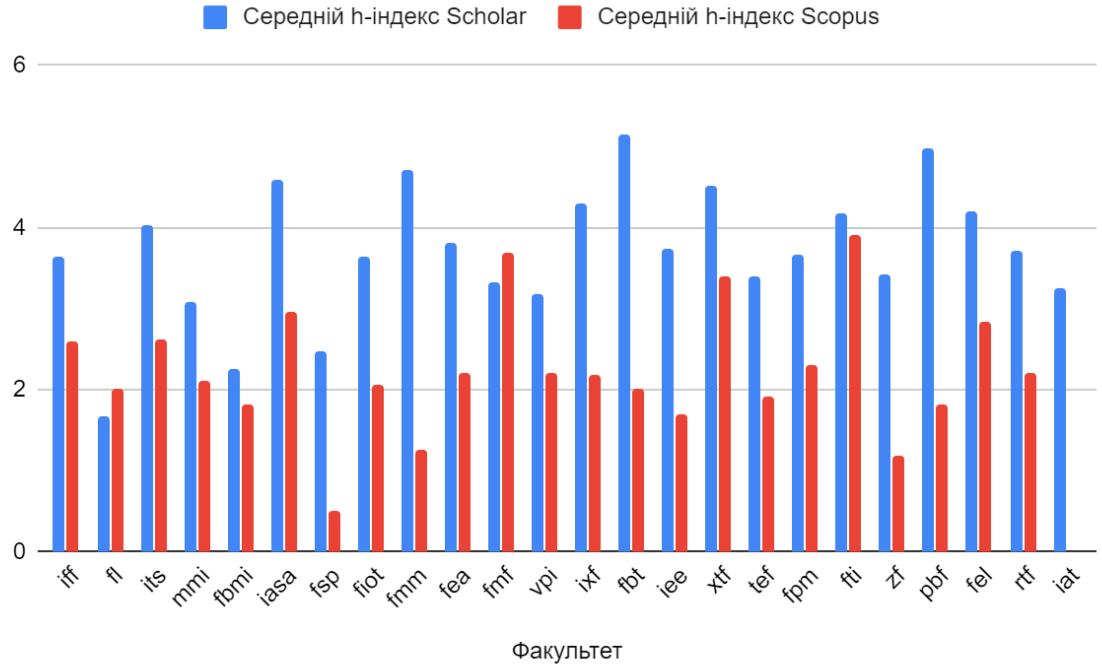


Рис. 6. Середній h-індекс вGoogle Scholar та Scopus

При аналізі даних необхідно пам’ятати, що кількість науковців, що мають акаунт в Google Scholar, значно перевищує таких у Scopus. Також Google Scholar є основним джерелом для оцінки наукометричних показників кафедр, ЗВО та вчених через більшу кількість представлених ресурсів, простішу та дешевшу процедуру публікації. При вирішенні питання фінансування українських ЗВО, ключову роль відіграють наукометричні показники, джерелом яких є саме Google Scholar. Тож подальше дослідження буде проводитися, спираючись на дані, що були отримані з зазначеного джерела.

Всього у Google Scholar було зроблено 202239 посилань на статті, авторами яких є викладачі КПІ. Гістограма, наведена на рисунку 7, зображує кількість посилань у відсотках від загальної кількості, що була зроблена на авторів, які є членами зазначених кафедр.

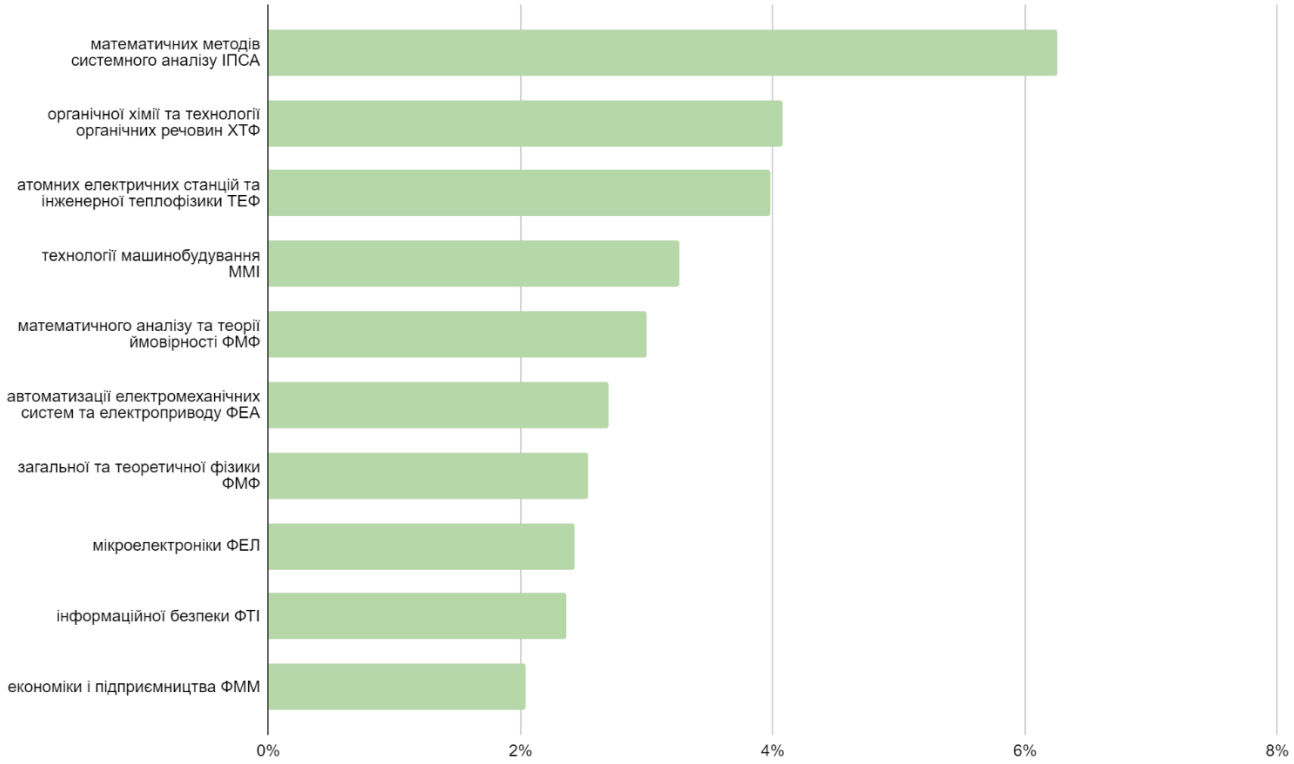


Рис. 7.Топ-10 кафедр з найбільшою кількістю посилань у відсотках

Табл. 1. ***Топ-10 найкращих факультетів за результатами сортування за індексом Хірша***

Факультет	Публікацій Scholar	Посилання Scholar	Середній h-індекс Scholar
fbt	1199	4289	5,153846154
pbf	4245	10905	4,9875
fmm	5730	14147	4,721153846
iasa	1906	13995	4,578947368
xtf	3996	15958	4,52
ixf	4437	8415	4,288888889
fel	4776	12617	4,19047619
fti	1601	7625	4,170212766
its	2333	4662	4,02173913
fea	4168	12347	3,807692308

Оскільки найважливішим наукометричним параметром для одного вченого є його h-індекс, отже найважливішим показником для оцінки наукового підрозділу як сукупності вчених будемо вважати значення середнього h-індексу. Складемо рейтинг топ-10 найкращих факультетів (таблиця 1) та топ-10 найкращих кафедр (таблиця 2) КПІ ім. Ігоря Сікорського за показниками індексу Хірша.

Табл. 2. ***Топ-10 найкращих кафедр за результатами сортування за індексом Хірша***

Кафедра	**Публікацій Scholar**	**Посилань Scholar**	**Середній h-індекс Scholar**
Біоінформатики ФБТ	203	2418	11,33333333
Органічної хімії та технології органічних речовин ХТФ	502	8254	8,461538462
Оптичних та оптико-електронних приладів ПБФ	537	1998	7,375
Виробництва приладів ПБФ	1206	3663	6,789473684
Загальної та теоретичної фізики ФМФ	591	5136	6,666666667
Фізичної хімії ХТФ	564	1264	6,222222222
Фізики енергетичних систем ФТІ	206	958	6,2
Хімічного, полімерного і силікатного машинобудування ІХФ	1078	3181	6
Систем керування літальними апаратами ММІ	71	104	6
Теоретичних основ радіотехніки РТФ	313	1150	5,857142857

Висновок

Результати аналізу показали, що найбільше число цитувань мають факультети прикладного-технічного спрямування. Перші місця за рейтингом по індексу Хірша займають природничо-технічного напрямки. Це стосується в однаковій мірі кафедр і факультетів.

Проведений аналіз є дослідницьким, демонструє можливості програмного рішення, та не обмежується наданими матеріалами. Маючи описану базу даних власник зможе виконувати аналіз публіцистичної діяльності викладачів під потрібним кутом.

Список літератури

1. НАУКОМЕТРИЧНІ ПОКАЗНИКИ [Електронний ресурс] // webometr – Режим доступу до ресурсу: https://webometr.kpi.ua/citation-data.
2. РОЗПОДІЛ ФІНАНСУВАННЯ УНІВЕРСИТЕТІВ [Електронний ресурс] // osvita.ua. – 2020. – Режим доступу до ресурсу: https://osvita.ua/vnz/reform/69813/.
3. СЕРЕДНІ РЕЙТИНГОВІ ПОКАЗНИКИ НАУКОВО-ПЕДАГОГІЧНИХ ПРАЦІВНИКІВ КАФЕДР ЗА НАПРЯМАМИ ДІЯЛЬНОСТІ У 2017/2018 Н.Р. [Електронний ресурс] // webometr. – 2018. – Режим доступу до ресурсу: https://webometr.kpi.ua/2018-rating-teacher.
4. WEB OF SCIENCE VERSUS SCOPUS: JOURNAL COVERAGE OVERLAP ANALYSIS [Електронний ресурс] // Texas A&M University Libraries. – 2019. – Режим доступу до ресурсу: https://oaktrust.library.tamu.edu/bitstream/handle/1969.1/175137/Web%20of%20Science%20versus%20Scopus%20Report%202019.pdf?Sequence=4&isallowed=y.
5. A UNIVERSALLY UNIQUE IDENTIFIER (UUID) URN NAMESPACE [Електронний ресурс] // datapower Technology, Inc.. – 2005. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc4122.

УДК 658.783

ХОМЕНКО О.М.,
ГАВРИЛЕНКО О. В.

ІНФОРМАЦІЙНА СИСТЕМА З ПІДТРИМКИ НАПИСАННЯ ВІРШІВ

У статті розглядається генерація віршів комп'ютером за допомогою нейронних мереж, використовуючи архітектуру transformer,attention. Демонстрація побудови моделі для роботи з віршами, процес обробки вхідних даних, їх структуризація, покращення, визначення головних характеристик(рима, темп, стиль) для роботи й оцінки віршів. Процес навчання мережі та створення сонетів. Аналіз результатів, визначення сильних та вразливих місць у моделі, припущення щодо їх вирішення. Якість віршів, що генеруються, оцінюється за допомогою JaccardSimilarity та за допомогою опитування. Визначення критеріїв для покращення поточного результату.

КЛЮЧОВІ СЛОВА: Нейронні мережі, рекурентні нейронні мережі, LSTM, BERT.

This article examines the generation of poems on a computer using neural networks using the transformer, attention architecture. Demonstration of model construction for work with poems, process of processing of input data, their structuring, improvement, definition of main characteristics (rhyme, tempo, style) for work and evaluation of poems. The process of learning the network and creating sonnets. Analysis of results, identification of strengths and vulnerabilities in the model, assumptions about their solution. The quality of the poems that are generated is assessed using Jaccard Similarity and a survey. Defining criteria for improving the current result.

KEYWORDS: Neural Networks, Recurrent Neural Networks, LSTM, BERT.

1. Вступ

Вірші є невід'ємною частиною літератури. За допомогою віршів можна передавати емоції, думки. Написання віршів є важким процесом – потрібно виконати необхідні умови для створення справжнього шедевру, а саме: зберігати головну ідею, риму і дотримуватися стилю. Стиль тексту - важливий фактор, що визначає його якість, розбірливість та ідентичність. Вченим завжди було цікаво чи зможе комп'ютер згенерувати вірш, який відповідав всім попереднім критеріям. На даний момент будь-які спроби не увінчалися успіхом, але є вже великий прогрес у напрямку створення текстів, накладаючи на них певні обмеження. Першим запропонованим підходом у даній категорії задач були марковські процеси. Але вони представляють достовірно локальні властивості послідовностей, що робить їх не придатними для такого завдання. Неможливо зберегти ідею вірша у великих послідовностях (вірш>3 рядків). Крім того вірші мають рими та метричні обмеження, які додають додаткову складність при формулюванні марковських процесів і ці обмеження потрібно задовольнити на великих послідовностях слів. Як наслідок, більшість підходів до автоматичного генерування текстів (віршів) на основі марковських моделей не задовольняє всі критерії, що обмежує їх використання для практичних застосувань, таких як: генерування текстів машинний переклад [1] або автоматичне узагальнення [2]. Одним з можливих рішень цієї задачі є рекурентні нейронні мережі, які можуть запам'ятовувати великі послідовності тексту, зберігаючи при цьому контекст. Стиль параметризується параметрами низького рівня, такими як довжина рядка. Однак ці системи не справляються зі структурними обмеженнями, що стосуються поезії (рими, метр). З іншого боку, поетичні системи генерують добре сформовані граматично тексти, можна відслідкувати. У системі [3] використана комплексна модель для задоволення кожного критерію: рима, стиль, граматика. Цей підхід застосовується у цій системі на базі рекурентних нейронних мереж і механізму «уваги». Даний підхід використовується для генерування сонетів, у яких є визначені правила щодо написання. Результат можна покращити за допомогою збільшення розміру датасету для навчання, покращити процес створення словника, але

тоді час навчання моделі може вирости в рази і відповідно знадобиться більша обчислювальна потужність. Система [4] використовує генетичні алгоритми для генерування синтаксично правильних текстів, що слідують накладеним шаблоном метрик і передають заданий зміст. Система WASP [5] та її покращення ASPERA [6] створюють поезію. Жодна з цих систем не вирішує проблему стилю, контексту та рими одночасно. Вони вирішують проблеми з римою, і метром, але контекст не завжди зберігається або навпаки. У даній роботі використаємо відносно нову архітектуру нейронних мереж, яку розробив Google – Transformer [7] і механізм «уваги» для генерації віршів.

2. Підготовка даних для тренування

Моделі BERT та GPT-2 вже мають натреновані моделі різних розмірностей, але ці моделі натреновані на загальних текстах з Wikipedia і Reddit. Ці дані є загальними і при генеруванні віршів не буде виконуватися римування і структура вірша. Спочатку для навчального датасету включив існуючі[8], і вірші з онлайн-бібліотеки[9], які були зчитані за допомогою pythonскрипта. Фільтруємо дані, видаляючи знаки пунктуації, лишні пробіли і знаки.

- Підготовка даних для BERT

Кожне речення(рядок) повинно починатися з маркера [CLS], аналогічно, кожна послідовність повинна закінчуватися маркером [SEP]. Це допоможе мережі зрозуміти з якого слова краще починати рядок і яким словом його закінчувати, також генерування хорошої рими. Враховуючи, що вірші різної розмірності тому, щоб забезпечити однакову розмірність рядків, ми фіксуємо максимальну довжину T і додаємо маркер "[PAD]", щоб усі вони були однакового розміру T. Після цього наші дані є готовими для передачі їх в нейронну мережу BERT і розпочати навчання.

- Підготовка даних для GPT-2

Зі сторони може здатися, що роздільник не потрібен для тренування, але його наявність дозволяє моделі вивчити форматування навчальних даних. Наприклад, якщо ми просто хотіли створити вірш довільної форми(кількість складів у кожному рядку буде варіюватися), то ми можете зібрати їх усі разом у файлі без роздільника. Однак, якщо ви хочете створити вірш зі словника слів, які вживає певний автор / вірш певного жанру, то доцільно буде масштабувати масштаб до одного жанру чи автора обмежуючи їх <|startoftext|> і <|endoftext|>.

3. Архітектура моделі

BERT використовує Transformer, механізм уваги, який вивчає контекстні зв'язки між словами (або підсловами) у тексті. У своїй ванільній формі Transformer включає два окремих механізми - encoder, який зчитує введений тексту та decoder, який дає передбачає слова/послідовності для завдання. Оскільки метою BERT є створення мовної моделі, необхідний лише механізм encoder. Детальна робота трансформатора описана в документі Google [10]. На відміну від directional моделей, які читають введення тексту послідовно (зліва направо або справа наліво), encoderTransformer зчитує всю послідовність слів відразу. Тому він вважається двонаправленим, хоча було б точніше сказати, що він не направлений. Ця характеристика дозволяє моделі вивчити контекст слова на основі всього його оточення (зліва та справа від слова). Рисунок нижче - опис encoderTransformer на високому рівні. Вхід - це послідовність лексем/слів, які спочатку перетворюються у вектори, а потім обробляються в нейронній мережі. Вихід - це послідовність векторів розміром n, кожному вектору відповідає вхідний маркер з однаковим індексом.

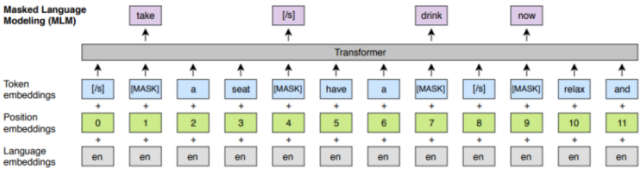


***Рис.1. Обробка вхідних даних перед надсиланням їх в нейронну мережу*[11]**

4. Механізм уваги

Механізм уваги можна описати наступним рівнянням [12].

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Q - матриця, яка містить запит (векторне представлення одного слова в

послідовності), K - всі ключі (векторні представлення всіх слів у послідовності) і V - значення, які знову є векторними представленнями всіх слів в послідовності. Для encoder і decoder, multi-head модулів уваги, V складається з тієї ж послідовності слів, що й Q. Однак для модуля уваги, який враховує послідовності encoder і decoder, V відрізняється від послідовності, представленої Q. Щоб трохи спростити це, ми могли б сказати, що значення V множиться та підсумовуються з деякими вагами a, де наші ваги визначаються [13]:

$$a = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

Це означає, що ваги a визначаються тим, як на кожне слово послідовності (представлене Q) впливають усі інші слова в послідовності (представлені K). Крім того, функція SoftMax застосовується до ваг a для розподілу між 0 і 1. Ці ваги потім застосовуються до всіх слів у послідовності, що вводяться в V (ті ж вектори, що і Q для encoder і decoder, але різні для модуля, який має входи encoder і decoder). Цей механізм уваги можна паралелізувати. Механізм уваги повторюється багаторазово за допомогою лінійних проекцій Q, K і V. Це дозволяє системі вчитися з різних представлень Q, K і V, що вигідно для моделі. Ці лінійні представлення виконуються шляхом множення Q, K і V на вагові матриці W, які вивчаються під час тренінгу. Ці матриці Q, K і V відрізняються для кожного положення модулів уваги в структурі залежно від того, чи перебувають вони в encoder, decoder або між encoder і decoder. multi-head модуль уваги, який з'єднує encoder і decoder, переконується, що вхідна послідовність encoder враховується разом із вхідною послідовністю decoder до заданої позиції.

5. Тренування моделі

Перед подачею послідовностей слів у BERT 15% слів у кожній послідовності замінюються маркером [MASK]. Потім модель намагається передбачити початкове значення слів, що замасковуються, виходячи з контексту, передбаченого іншими, не маскованими, словами в послідовності.Я спробував тренувати цю модель протягом 60 епох і набрав точності 85,23%,

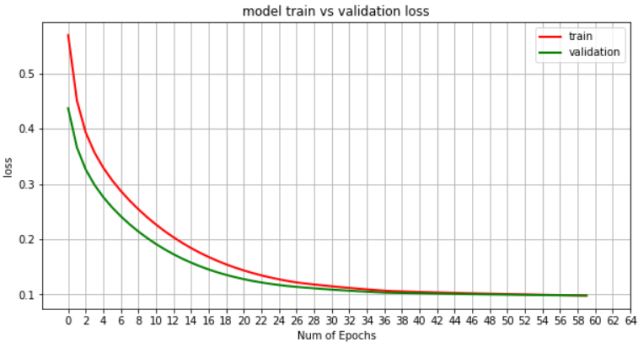


Рис.2. Похибка результатів відносно кількості епох

6. Алгоритм генерування віршів

Користувач вводить рядок вірша, відповідно генерується наступна схема рядка [MASK] [MASK] ... [MASK] [RHYME-WORD], де останнє слово генерується до останнього в попередньому рядку (створюється рима). Модель починає генерувати слова і заміняє відповідні MASK-и. Поки вірш формується вибирається випадкове слово за допомогою метод Монте-Карло для марковських ланцюгів[14] в рядку і програма замінює його на [MASK]. Модель думає, яке слово вставити, бо контекст змінився і підбирає кандидатів на цю позицію. Випадково вибирається слово з набору найбільш ймовірних (найбільш ймовірне слово не завжди найкращий вибір), тому розширимо набір слів до 15. Ця процедура повторюється поки не буде згенеровано всі рядки віршу.

7. Оцінка результатів

- JaccardSimilarity [15]

Індекс схожості Жакарда (іноді його називають коефіцієнтом схожості Жакарда) - це показник схожості для двох наборів даних, діапазон від 0% до 100%. Чим більший відсоток, тим схожіші дві групи.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

JaccardSimilarity =0.00021348995.
Це означає, що програма не заучує певні рядкові комбінації авторів, а генерує свої унікальні вірші.

- Оцінка віршів людьми

Люди підмітили, що вірші легко читати, вони мають гарну риму, але неідеальний стиль(іноді склади з'їжджають і втрачається метрика).

8. Вірш

Theroadtohappinessinthehopesofall,
isbasedonBurmesefolktalecalledAmarball
pairsofbirdsfollowtheflowersintheircall
beingbythebookandsufferingbythefall
Spacetohappinessinthestateofperfectprotocol
thereisonebrightrayshiningthroughthewall
thehopeofthewholeworldisbanal
orroadtogreathappinesswhendeclareduponhall
roadtohappinessasonlystarscanshowstall
roadtosuchversesbethevoiceofrecall
composedbyJohannStrauss II in 1925 cabal

Рис. 3. Приклад створеної поезії

Висновок

У цій роботі ми спробували знайти шляхи вирішення проблеми генерування стилізованого римованого тексту за допомогою архітектурної моделі BERT. Вона краще вміє створювати нові вірші ніж LSTM, це пов'язано з архітектурою, яка допомагає знаходити ключові слова і виділяти їх з контексту. Згенеровані вірші є унікальними: мають риму та стиль, але іноді в них уривається контекст і стиль. Це пов'язано з великою вибіркою віршів, яка складається з віршів різних за стилем і схемою римування. Але вірші гарно читаються і іноді є цілком змістовними. Для покращення результатів можна збільшити датасет для навчання, а також провести навчання на більшій і складнішій моделі GPT-2. Цю модель можна використати для генерування окремих рим, зменшивши довжину вхідного рядка до одиниці і запустити процес підбору рим близьких за можливим контекстом.

Список літератури

1. H. Somers, 'Review article: Example-based machine translation', Ma-chine Translation,14, 113–157, (1999)
2. R. Barzilay, 'Probabilistic approaches for modeling text structure andtheir application to text-to-text generation', in Empirical methods innatural language generation, pp. 1–12. Springer-Verlag, (2010).
3.Jey Han Lau, Trevor Cohn, Timothy Baldwin, Julian Brooke, Adam Hammond, 'Deep-speare: A joint neural model of poetic language, meter and rhyme', in IBM Research Australia, (2018).
4.Xiaoyuan Yi, Ruoyu Li, and Maosong Sun, "Generating chinese classical poems with rnnencoderdecoder," in Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data, 2017, pp. 211–223.
5.Zachary C. Lipton, SharadVikram, and Julian McAuley, "Capturing meaning in product reviews with character-level generative text models," in arXiv preprint, 2015.
6.Chloe Kiddon, Luke Zettlemoyer, and Yejin Choi, "Globally coherent text generation with neural checklist models," In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 329–339, 2016.
7.Ashish Vaswan, Noam Shazeer, NikiParmar, JakobUszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, IlliaPolosukhin "Attention Is All You Need", 2017.
8.Complete poetryfoundation.org dataset[Електронний ресурс] - Режим доступу до ресурсу:https://www.kaggle.com/johnhallman/complete-poetryfoundationorg-dataset#kaggle_poem_dataset.csv
9.Free eBooks - Project Gutenberg [Електронний ресурс] - Режим доступу до ресурсу:https://www.gutenberg.org/
10. Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing [Електронний ресурс] - Режим доступу до ресурсу:https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html
11. BERT: State of the Art NLP Model, Explained [Електронний ресурс] - Режим доступу до ресурсу:https://www.kdnuggets.com/2018/12/bert-sota-nlp-model-explained.html
12. CTRL: A CONDITIONAL TRANSFORMER LANGUAGE MODEL FOR CONTROLLABLE GENERATION NitishShirishKeskar∗ , Bryan McCann∗ , Lav R. Varshney, CaimingXiong, Richard Socher Salesforce Research†
13. How to code The Transformer in PyTorch[Електронний ресурс] - Режим доступу до ресурсу: https://blog.floydhub.com/the-transformer-in-pytorch/
14. MarkovChainMonteCarloinPython[Електронний ресурс] - Режим доступу до ресурсу:https://towardsdatascience.com/markov-chain-monte-carlo-in-python-44f7e609be98
15. IntroductiontoSimilarityMetrics[Електронний ресурс] - Режим доступу до ресурсу:https://medium.com/analytics-vidhya/introduction-to-similarity-metrics-a882361c9be4

УДК 37.09

ФАМ С. Х.
ТЄЛИШЕВА Т. О.

СИСТЕМА ПІДТРИМКИ НАВЧАННЯ СТУДЕНТІВ - УЧАСНИКІВ ЄВРОПЕЙСЬКИХ ПРОГРАМ МОБІЛЬНОСТІ

Розглянуто система підтримки навчання студентів – учасників європейських програм мобільності та запропоновано ефективні способи подачі документів, щоб скорочувати втрати часу на їх оформлення та на вибір курсів для зарахування кредитів після закінчення програми обміну.
ПРОГРАМ МОБІЛЬНОСТІ, КРЕДИТ, ДОКУМЕНТ, ВІДДІЛ МОБІЛЬНОСТІ, КООРДИНАТОР, ІНДИВІДУАЛЬНИЙ НАВЧАЛЬНИЙ ПЛАН, ЄКТС

The system support for student – participant of European program mobility and develop effective ways of applying documents to reduce waste of time. Choosing courses for crediting ECTS after finishing exchange program.
PROGRAM MOBILITY, CREDIT , APPLYING DOCUMENT, OFFICE OF MOBILITY, COORDINATOR, INDIVIDUAL LEARNING PLAN, ECTS

1. Вступ

Еразмус – це програма обмінів студентів, викладачів та науковців країн-членів Євросоюзу. Програм надає можливість навчатися, проходити стажування чи викладати в іншій країні, що бере участь в програмі[1]. Зазвичай, у кожному році відбувається 2 рази конкурс студентів для програми обміну, це весняний семестр і літній семестр. Студенти і викладачі, які хочуть брати участь в програмі мають подати документи у відділ мобільності. Терміни навчання і стажування можуть складати від 3 місяців до 1 року. Щоб брати участь в програмі, кандидат повинен бути громадянином країни. Для навчання за кордоном кандидат повинен володіти англійською мовою на рівні B1, крім цього, участь можуть брати студенти від другого року навчання і студент може взяти участь в програмі тільки 1 раз. За умовами Еразмус студент може отримувати стипендію або вчитися без стипендії, але за навчання в іноземному ВНЗ студент не платить. Сума стипендії залежить від ліміту, який визначається для кожної країни.

Етапи проходження конкурсу є:

- вибір курсів, для навчання за кордону;
- подача документи;
- відбір переможців;
- подача індивідуальний навчальний план.

Послідовність дії наведено на рисунку №1.

2. Вибір курсів

Для початку програми мобільності, студенти мають вибирати для себе курси, в яких вони зацікавлені і хочуть слухати в університеті-партнері. Оскільки навчання в університеті-партнері проходить паралельно з навчанням в їхньому університеті, студенти повинні вибирати такі курси, які можуть бути зараховані в кредити після закінчення програми обміну.

ЄКТС (ECTS) є системою накопичення та трансферу кредитів, що орієнтована на особу, яка навчається, основана на принципах прозорості процесів навчання, викладання та оцінювання. Практичне призначення цієї системи це надати студентам можливість самостійно зробити вибір курсів, які вони хочуть вивчати і досягати успішного результату після навчання. Один з недоліків цієї системи є несумісність з національними / місцевими освітніми програмами[2]. В різних університетах різні навчальні плани для студентів і вони не завжди співпадають. Звідси, для участі в програмі мобільності, студент має вибирати декілька курсів, щоб кількість кредитів відповідала за вимогою університету, крім цього, в списку обраних курсів повинні бути такі курси, які можуть зарахувати після закінчення програмі мобільності. Деталізовану схему за якою проходить вибір кредитів наведено на рисунку 2.

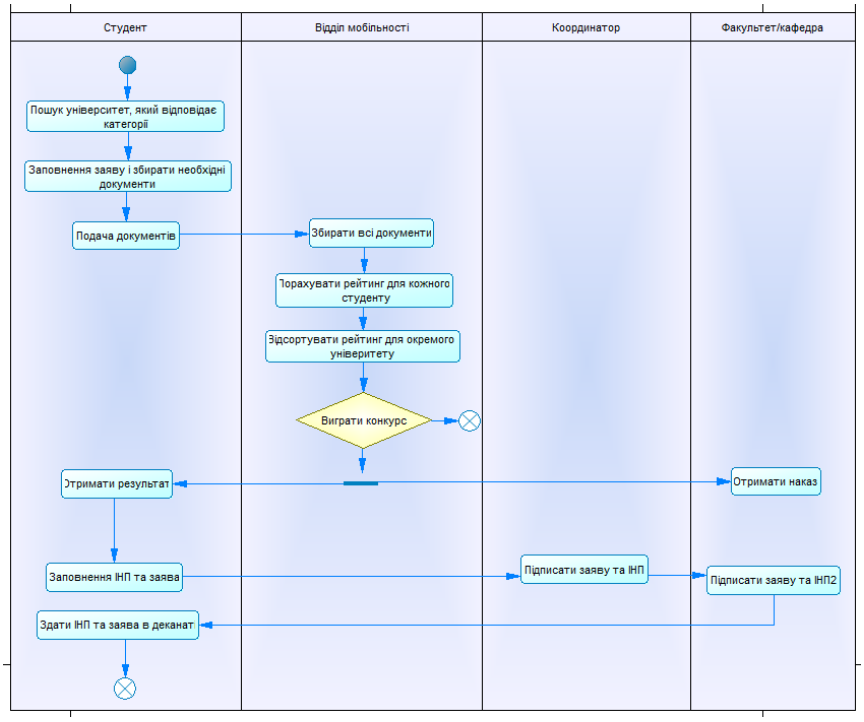


Рис. 1: Послідовність дії вибіру кредитів

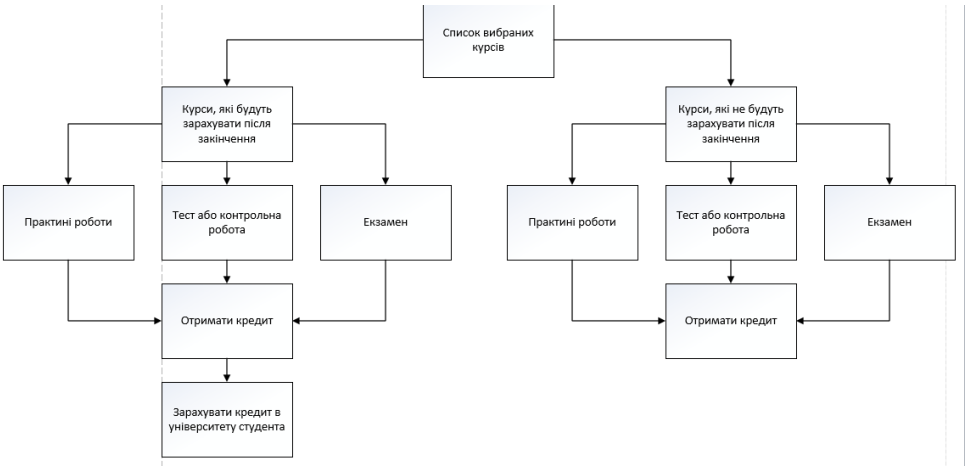


Рис. 2: Процес для отримання кредитів під час програму мобільності

Оскільки, в різні університеті, курси називають по різному. Наприклад, курси для розробки мобільного додатку, в КПІ ім. І. Сікорського називає «Мобільна розробка програмного забезпечення», але в університеті Ройтлінгені, університет-партнер КПІ, який знаходиться в Німеччині, цей курс називається «Mobile Computing». Бачимо ситуація, якщо перекласти назву курса з української мови на англійській мові, вони не співпадають, отже пропонуємо функцію системи, яка порівнює зміста курсів, якщо вони співпадати більше ніж 50%, то цей курс може зарахувати після закінчення програми мобільності і автоматично вставити в Learning Agreement. Так як курси з університету студента не завжди співпадати зі списоком курсів університету-партнера, студент повинен заповнити вручно інші курси, щоб кількість кредитів відповідала за вимогою.

3. Подача документів

Подача документів є одним з важливим кроком в програмі мобільності. Для участі в конкурсі, студент повинен мати копію паспорту для підтвердження громадянства, копію довідки про середній бал, оригінал сертифікату про рівень іноземної мови, яку вимагає університет-партнер, заповнений бланк, де потрібно вказати в якому університеті студент хоче навчатися за

кордоном, публіковані статті, наукова, дослідницька робота[3]. Вимоги для участі в конкурсу є:

- середній бал не нижче 75 балів;
- рівень володіння мови не може бути нижче ніж B1 або B2;
- участь можуть брати студенти від другого року навчання;
- у деякому університеті-партнері вимагають студентів з певного факультету.

Оскільки, цей процес звичайно проходиться в ручному вигляді, це означає що учасник повинен приносити документи у відділ мобільності. Пропонується перевести процес подачі документів в онлайному вигляді, це означає що учасник може подати документи через Інтернет. Цей процес скорочує витрати часу, обидві сторони можуть отримати оригінал , більш безпечно і менше ймовірність втрати документів.

4. Відбір переможців

Після подачі документів, учасник повинен чекати тижні, щоб отримати результати. Для обчислення рейтингу, необхідно врахувати такі пункти:

- середній бал;
- перемоги в олімпіадах;
- публіковані статті;
- наукова-дослідницька робота;
- рівень володіння мови.

Для кожного пункту є певні коефіцієнти. Наприклад, для кожної статті вважає 1 бал, але для кожної наукової роботи вважає 3-5 балів. Так же робиться для середнього балу, студент з високим середнім балом буде мати більше переваги. Крім цього, сертифікат рівня володіння мови також має різні коефієнти. Для англійської мови, найкращий сертифікат є IELTS або TOEFL, але не всі учасники мають достатній час, щоб підготувати і проходити цей тест. І звичайно, учасники буде проходити тест англійської мови в університеті і отримає сертифікат з університету. Коефіцієнти цього сертифікату звичайно менше ніж коефіцієнту сертифікату IELTS або TOEFL. Якщо ваш рівень володіння англійської мови це B2, то сертифікат з університету дає тільки 2 бали, але сертифікат IELTS дає 4-6 балів. Рейтинг учасників рахується як сума всі коефіцієнтів. Як наведено вище, якщо процес подачі документів проходить в онлайному вигляді, то всі ці дані дають можливість відразу обчислити рейтинг для окремого учаснику і створюється таблиця рейтингу для окремих студентів. Тоді час очікування результатів вже не 3-4 тижні, а буде тільки 3-4 дні.

5. Укладання договору і ІНП

Подача заяви на укладання договору на навчання в своєму університеті і на ІНП (індивідуальний навчальний план) є останнім кроком. ІНП повинен включає всі курси, які будуть навчатися в університеті учасника протягом програми мобільності. В заяві та ІНП повинен бути підпис координатора, підпис декана факультету та завідувача кафедри і цей процес також проходить в ручному вигляді. І в деяких ситуаціях, або координатор або декан факультету або завідувач кафедри можуть бути відсутні в університеті і через це витрачається більше часу ніж очікується[4]. Підпис може бути в електронному вигляді, головне - це отримати дозвіл від відділу мобільності.

Висновок

Пропонуємо розробити інформаційну технологію, додаток і веб-сайт , якій буде доступний всім для підтримки процесу оформлення документів: скорочення втрати часу, більш безпечний щодо втрати документів. Для цього необхідно мати узгодження з чотирма сторонами: учасники мобільності, відділ мобільності, координатор з мобільності, факультет/кафедра.

Перелік посилань

1. Програма Еразмус [Електронний ресурс] – Режим доступу до ресурсу : https://uk.wikipedia.org/wiki/Програма_Еразмус
2. Відділ академічної мобільності КПІ ім. І. Сікорського [Електронний ресурс] – Режим доступу до ресурсу : http://mobilnist.kpi.ua
3. Порядок оформлення документів для направлення за кордон [Електронний ресурс] – Режим доступу до ресурсу : http://mobilnist.kpi.ua/processing-documents/

УДК 004.02

СУВОРОВА В. Є.
ІБНУХСЕЙН І.
ЖДАНОВА О. Г.
СПЕРКАЧ М. О.

ЗАДАЧА СКЛАДАННЯ РОЗКЛАДУ ПРОСЛУХОВУВАНЬ ВСТУПНИКІВ ДО ДИТЯЧОГО ХОРУ МЕТОДОМ ПРОМЕНЕВОГО ПОШУКУ

Робота присвячена розв'язуванню задачі складання розкладу прослуховувань вступників до дитячого хору методом променевого пошуку. Задача, що розглядається відноситься до класу NP-повних задач. Розглянуто підходи до розв’язання наведеного класу задач. Сформульовано постановку задачі. Проведено порівняльний аналіз алгоритму променевого пошуку з іншими алгоритмами. Розглянуто один з поширених методів дискретної оптимізації – метод гілок та меж. Наведено алгоритм розв’язання задачі цим методом. Описано метод променевого пошуку та розроблено алгоритм розв’язання задачі складання розкладу прослуховувань вступників до дитячого хору. Проведено дослідження ефективності розроблених алгоритмів.

КЛЮЧОВІ СЛОВА: ТЕОРІЯ РОЗКЛАДІВ, NP-ПОВНА ЗАДАЧА, МЕТОД ГІЛОК ТА МЕЖ, МЕТОД ПРОМЕНЕВОГО ПОШУКУ, РОЗКЛАД ПРОСЛУХОВУВАНЬ

The work is devoted to solving the problem of schedulin gauditions for entrants to the children's choirby the method of Beam Search. The problem under consider at ionbelongs to the classof NP-completeproblems. Approaches to solving this class of problem sare considered. The state ment of the problem is formulated. A comparative analys is of the Beam Sear chalgorithm with other algorithms is carried out. One of the common discrete optimization methods is considered – the Branchand Bound method. An algorithm for solving the problem by this method is presented. The method of Beam Search is described and analgorithm for solving the problem of schedul in gauditions for entrantsto the children's choiris developed. The efficiency of the developed algorithms is investigated.

KEYWORDS: SCHEDULE THEORY, NP-COMPLETE PROBLEM, METHODS OF BRANCHES AND BOUNDS, METHOD OF BEAM SEARCH, AUDITION SCHEDULE.

1. Вступ

При проведенні прослуховувань для бажаючих вступити до дитячого хору, постає задача: скласти оптимальний розклад проведення іспиту для вступників до дитячого хору, враховуючи особливості та рівень підготовки кожної дитини.Тому важливо розробити алгоритм, що за відносно малий період часу зміг би розв’язатидану задачу.

У роботі розглядається задача теорії розкладів, в якій потрібно знайти оптимальну послідовність екзаменування вступників.Кожна дитина подає заявку, у якій вказує наступні дані:

– групу, в яку вона претендує потрапити (новачки, середній рівень, професіонали і т.і.);дана інформація впливає на час перевірки здібностей дитини під час екзаменування;

– деякі персональні дані,власні навички та досягнення, залежно від яких буде встановлено «рейтинг» дитини.

Також у заявки є деякий директивний термін – час, протягом якого дитина повинна бути проекзаменована.

Існують різні підходи до розв’язання такого класу задач. Серед існуючих методів було обрано алгоритм Променевого пошуку – він базується на методі гілок та меж. Метод гілок та меж гарантує точність розв’язку, бо, по суті, реалізує повний перебір підмножин розв’язків, через що страждає його швидкість.

Променевий пошук навпаки, не гарантує точність розв’язків, бо, на відміну від методу Гілок та меж, відсікає значні підмножини

розв’язків на кожному етапі і потім до них вже не повертається. Це робиться якраз для покращення часових характеристик. Тому, в роботі буде реалізовано також і метод Гілок та меж для порівняльного аналізу точності розв’язків та швидкості алгоритму Променевого пошуку.

Отже, метою даної роботи є дослідження ефективності застосування алгоритму Променевого пошуку при складанні розкладу прослуховувань вступників до дитячого хору.

2. Постановка задачі

Потрібно обробитиnзаявок. Кожна заявка має декілька параметрів: фактичний час, що потрібен для обробкизаявкиp_j та бажаний час d_j, до якого ця заявка повинна бути оброблена (залежить від часу подання заявки). Також вказана цінність заявки (штраф за кожну годину затримки обробкизаявки)w_j.

Нехай маємо: $T_j = \max(C_j - d_j; 0)$,

де$C_j = \sum_{i=1}^{j} p_i, j = \overline{1,n}$.

Тоді цільова функція задачі матиме наступний вигляд: $z = \sum_{j=1}^{n} T_j w_j$.

Тобто задача полягає в наступному: визначити порядок, в якому повинні бути виконані роботи, щоб штраф за запізнення був мінімальним.

3. Існуючі методи розв'язання

Задача відноситься до класу NP-повних задач, отже можливі методи вирішення задач наступні:

- метод гілок та меж;
- метод променевого пошуку.

Метод гілок та меж — один з поширених методів дискретної оптимізації [2]. Метод працює на дереві рішень та визначає принципи роботи конкретних алгоритмів пошуку розв'язків, тобто, є мета-алгоритмом.

Метод променевого пошуку[3] – це евристичний алгоритм пошуку, що досліджує граф, розширюючи найперспективніші вузли в обмеженому їх наборі. Променевий пошук є оптимізацією методу гілок та меж, що суттєво знижує його вимоги до необхідної кількості пам’яті. В променевому пошуку лише деяка частина розв’язків зберігаються як кандидати[4, 6].

3.1. Метод гілок та меж

В основі методу гілок та меж лежить ідея послідовного розбиття множини допустимих рішень. На кожному кроці методу елементи розбиття (підмножини) піддаються аналізу - містить дана підмножина оптимальне рішення чи ні. Якщо розглядається задача на мінімум, то перевірка здійснюється шляхом порівняння нижньої оцінки значення цільової функції на даній підмножині з верхньої оцінкою функціоналу. В якості оцінки зверху використовується значення цільової функції на деякому допустимому розв’язку.

Допустиме рішення, що дає найменшу верхню оцінку, називають рекордом. Якщо оцінка знизу цільової функції на даній підмножині не менш оцінки зверху, то підмножина, яка розглядається, не містить рішення краще рекорду і може бути відкинутою. Якщо значення цільової функції на черговому рішенні менше рекордного, то відбувається зміна рекорду. Будемо вважати, що підмножину рішень переглянуто, якщо встановлено, що вона не містить рішення кращого за рекорд.

Якщо переглянуті всі елементи розбиття, алгоритм завершує роботу, а поточний рекорд є оптимальним рішенням. В іншому випадку серед непереглянутих елементів розбиття обирається множина, що є в певному сенсі перспективною. Вона розбивається (розгалужується). Нові підмножини аналізуються за описаною вище схемою. Процес триває до тих пір, поки не будуть переглянуті всі елементи розбиття [5].

Нехай на вхід подається:

L – поточний список підмножин множини допустимих розв’язківX;

$f(x)$ – Цільова функція ($f(x) \to min$);

n– кількість робіт;

p_j – час виконання роботи j;

d_j – час, до якого робота j повинна бути виконана;

w_j–штраф за кожну годину затримки виконання роботи j.

На виході будемо мати:

x^*–послідовність n робіт.

Алгоритм розв’язання задачі

КРОК 1. Ініціалізація: $f(x^*) := +\infty, x^* = \emptyset$
Обираємо зі списку L всю множину X.

КРОК 2. Whileсписок L не порожній **do**

КРОК 2.1.Ділимо множину за правилом $\beta(X^k) = \{X^{k**}\}$,

де $k-$ деяка часткова послідовність робіт на минулому кроці,
$k **= k \cup j$–додаємо до послідовності нову роботу j, що$j = \overline{1,n} \mid \forall\, j \neq K \in k$.
КРОК 2.2.Рахуємо для кожної $X^{k**} f(x^{k**})$.
КРОК 2.3.If$f(x^{k**}) > f(x^{*})$**then**
КРОК 2.3.1.Видаляємо множину розв'язківX^{k**}
endif
КРОК 2.4.If$|k|$==*n* (послідовність з *n* робіт) **then**
КРОК 2.4.1.If$f(x^{k**}) \leq f(x^{*})$**then**
КРОК 2.4.1.1. $x^{*} \coloneqq x^{k**}$
endif
endif
КРОК 2.5.Ifіснують не розглянуті розв'язки **then**
КРОК 2.5.1.Обираємо X^{k**}: $f(x^{k**}) = \min\{f(x^{j}) | X^{j} \in L\}$
elseрозв'язок знайдено
endif
endwhile
Функція $f(x^{k**})$:
КРОК 1.Для обраного розв'язку $f(x^{k**}) = \sum_{j=1}^{|k|} T_j w_j$,
де $T_j = \max(\sum_{i=0}^{j} p_i - d_j; 0)$.

3.2. Метод променевого пошуку

Фільтрований променевий пошук заснований на ідеях методу гілок і меж. Фільтрований променевий пошук – це адаптація методу гілок і меж, в якому на будь-якому рівні оцінюються не всі вузли. Тільки найбільш перспективні вузли на рівні *k* вибираються як вузли для розгалуження. Решта вузлів на цьому рівні відкидаються назавжди. Кількість вузлів, що залишилися, називається шириною променя. Процес оцінки, який визначає, які вузли є перспективними, є найважливішим елементом цього методу. Ретельна оцінка кожного вузла, щоб отримати оцінку потенціалу його нащадків, займає багато часу. Тут необхідно знайти компроміс: наближений прогноз є швидким, але може привести до відмови від хороших рішень, в той час як більш ретельна оцінка може займати занадто багато часу. Ось де вступає фільтр. Для всіх вузлів, згенерованих на рівні *k*, робиться грубе пророкування. На підставі результатів цих грубих прогнозів обирається кілька вузлів для більш ретельної оцінки, а інші вузли відкидаються назавжди. Кількість вузлів, обраних для більш ретельної оцінки, називають шириною фільтра. На підставі результатів більш ретельної оцінки вузлів, які проходять фільтр, підмножина цих вузлів (число, рівне ширині променя, яке не може бути більше ширини фільтра), вибирається з того місця, де генеруються додаткові гілки.

Простий приклад грубого прогнозу полягає в наступному. Обчислюється ЦФ часткового розкладу або будь-яка інша характеристика робіт, які ще потрібно запланувати; на основі цих значень вузли на даному рівні порівнюються один з одним і проводиться загальна оцінка.

Кожен раз, коли вузол повинен пройти ретельну оцінку, все ще не заплановані роботи плануються відповідно зі складовим правилом диспетчеризації. Такий розклад все ще можна генерувати досить швидко, так як цей процес вимагає тільки сортування. Об'єктивне значення такого розкладу є показником перспективності цього вузла. Якщо задіяна дуже велика кількість робіт, вузли можуть бути відфільтровані шляхом вивчення часткового розкладу, який генерується шляхом планування тільки підмножини робіт, що залишилася за правилом диспетчеризації. Цей розширений частковий розклад може бути оцінений, і на основі його значення вузол може бути відкинутий або збережений. Якщо вузол зберігається, його можна проаналізувати ретельніше, запланувавши решту роботи з використанням складеного правила диспетчеризації. Значення цільової функції цього розкладу представляє верхню межу кращого розкладу серед нащадків цього вузла.

Розв'язки в теорії розкладів часто знаходять за допомогою правил диспетчеризації, які є евристиками. Евристики не гарантують оптимального рішення – натомість вони спрямовані на те, щоб знайти відносно хороші рішення за відносно короткий час. Евристика має тенденцію бути досить загальною і може бути легко адаптована до великої кількості задач теорії розкладів. Дослідження правил диспетчеризації діють вже кілька десятиліть, і в літературі вивчається багато різних правил.

Оскільки метод гілок та меж є дуже затратним, важливо мати евристику, яка забезпечує досить гарний графік з прийнятними обчислювальними затратами. Деякі евристики спадають відразу на думку: правило WSPT та MS. Природно шукати евристичне або пріоритетне правило, яке поєднує в собі характеристики цих правил. Евристика ATC(ApparentTardinessCost) – це складене правило диспетчеризації, яке поєднує правило WSPT і правило MS [1]. За правилом MS обчислюється слабкість роботи j в момент часу t, тобто $\max(d_j - p_j - t; 0)$, і обирається робота з мінімальним показником. WSPT правило описується так:

$$\frac{w_j}{p_j} < \frac{w_k}{p_k}$$

Кожен раз, коли машина стає вільною, для кожної роботи, що залишилась, обчислюється індекс ранжирування. Потім обирається робота з індексом найвищого рейтингу для подальшої обробки. Цей індекс ранжування є функцією часу t, при якому машина стала вільною, а також від p_j, w_j та d_j робіт, що залишилися. Індекс визначається як:

$$I_j(x) = \frac{w_j}{p_j} e^{\frac{-\max(d_j - p_j - t; 0)}{K\bar{p}}}$$

де K– параметр масштабування, який можна визначити емпірично;

$\bar{p}$– середнє значення часу обробки робіт, що залишилися.

Якщо K є дуже великим, то правило ATC зводиться до правила WSPT. Якщо K є дуже малим, правило зводиться до правила MS, коли немає прострочених завдань і до правила WSPT для прострочених завдань в іншому випадку. Для розрахунку K нам потрібне значення коефіцієнту R, що визначається як

$$R = \frac{d_{max} - d_{min}}{C_{max}}$$

Тоді:

Якщо $R \leq 0{,}5$, то K = 4,5 + R;

якщо $R \geq 0{,}5$, то K = 6 – 2R.

Нехай на вхід подається:

L– поточний список підмножин множини допустимих розв'язківX;

$f(x)$ – Цільова функція (f(x) → min);

n – кількість робіт;

m – ширина променю;

p_j – час виконання роботи j;

d_j – час, до якого робота j повинна бути виконана;

w_j – штраф за кожну годину затримки виконання роботи j.

На виході будемо мати:

x^*–послідовність n робіт.

Алгоритм розв'язання задачі

КРОК 1. Ініціалізація:$f(x^*) := +\infty, x^* = \emptyset$

Обираємо зі списку L всю множину X

КРОК 2.Whileсписок L не порожній **do**

КРОК 2.1.Ділимо множину за правилом $\beta(X^k) = \{X^{k**}\}$

де $k-$ деяка часткова послідовність робіт на минулому кроці,

$k** = k \cup j$–додаємо до послідовності нову роботу j , що $j = \overline{1,n} \mid \forall j \neq K \in k$.

КРОК 2.2.Розраховуємо правило ***ATC***.

КРОК 2.3.Рахуємо $f(x^{k**})$ для відібраних за ATCрозв'язків.

КРОК 2.4.Видаляємо всі розв'язки крім m перших (у порядку зростання ЦФ).

КРОК 2.5.If$f(x^{k**}) > f(x^*)$

КРОК 2.5.1.Видаляємо множину розв'язківX^{k**}

endif

КРОК 2.6.If$|k|$==n (послідовність з n робіт) **then**

КРОК 2.6.1.If$f(x^{k**}) \leq f(x^*)$**then**

КРОК 2.6.1.1.$x^* := x^{k**}$

endif

endif

КРОК 2.7.Ifіснують не розглянуті розв'язки **then**

КРОК 2.7.1.Обираємо X^{k**}: $f(x^{k**}) = \min\{f(x^j) | X^j \in L\}$

elseрозв'язок знайдено

endif

endwhile

Функція розрахунку правилаATC:

КРОК 1. Whileне переглянуто всі обрані розв'язки x^{k**}**do**

КРОК 1.1.$R = \frac{d_{max} - d_{min}}{C_{max}}$,

КРОК 1.2.If$R \leq 0{,}5$ **then**

КРОК 1.2.1.$K = 4{,}5 + R$

else$K = 6 - 2R$

endif

КРОК 1.3.$I_j(x) = \frac{w_j}{p_j} e^{\frac{-\max(d_j - p_j - t; 0)}{K\bar{p}}}$

endwhile

КРОК 2.$i = 1$,

КРОК 3.Для всіх $i = n$ робіт повертаємо розв'язок, де перша робота i, а далі роботи розташовані по спаданню індексу $I_j(x)$.

Функція $f(x^{k**})$:

КРОК 1.Для обраного розв'язку $f(x^{k**}) = \sum_{j=1}^{n} T_j w_j$,де $T_j = \max(\sum_{i=0}^{j} p_i - d_j; 0)$

4. Дослідження розроблених алгоритмів

Метод гілок та меж є точним алгоритмом, бо він перебирає всі можливі варіанти розв'язків, відкидаючи якість множини розв'язків лише коли вони точно вже є гіршими за рекорд. Променевий алгоритм не гарантує точність розв'язку, але працює набагато швидше. У зв'язку з цим ми можемо дослідити точність роботи алгоритму променевого пошуку, порівнюючи його результати з результатом роботи методу гілок та меж.

Згенерувавши велику кількість ІЗ, та порівнявши результати робіт двох алгоритмів, ми побачили, що розв'язок алгоритмом променевого пошуку співпадає з розв'язком алгоритмом Гілок та меж близько на 95%. А у великій кількості випадків навіть і у 100% випадках. Проілюструємо наші результати графіком. На рисунку1 наведено приклад для 100 індивідуальних задач, коефіцієнт співпадання – 99%.

Отже, алгоритм променевого пошуку, не дивлячись на такий великий відрив за показником часу роботи, дає у більшості випадків розв'язок оптимальний або дуже близький до оптимального.

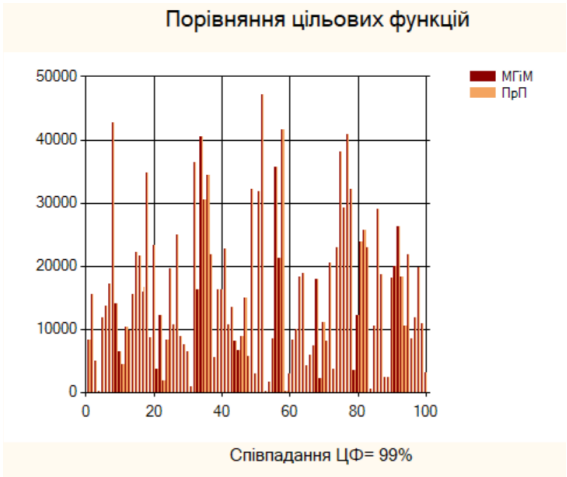


Рис. 1. Графік порівняння ЦФ для 100 ІЗ

Для розмірності 3 час роботи алгоритму гілок та меж трохи менший, але вже для розмірності 4 час роботи алгоритму променевого пошуку з відривом менший. Якщо ми побудуємо графік залежності часу для більших розмірностей (рис. 2), то можемо побачити, що час роботи алгоритму гілок та меж зростає дуже швидко, як ми і передбачали – графік схожий на функцію факторіалу. А час для алгоритму променевого пошуку змінюється прямопропорційно до розмірності задачі.

Результати експериментів співпадають з нашими теоретичними очікуваннями і ми можемо зробити висновок, що променевий пошук працює набагато швидше за метод гілок та меж.

Рис. 2. Графік залежності часу від розмірності задачі

Висновок

Результати експериментів для цих алгоритмів лише підтвердили теоретичні очікування – метод Променевого пошуку у порівнянні з методом гілок та меж працює надзвичайно швидко та, водночас, дає дуже близькі до оптимальних розв'язки. З огляду на це, можна зробити висновок, що при необхідності вирішення задачі теорії розкладів є сенс обрати Променевий пошук як алгоритм розв'язання, якщо ми можемо дозволити собі похибку в розв'язку ~5%. Ця похибка є цілком прийнятною для задачі створення розкладу прослуховувань хору.

У випадку, коли розв'язок повинен бути абсолютно точним, алгоритм Променевого пошуку не дає гарантію точності, і потрібно буде використовувати метод гілок та меж. Але він є надзвичайно повільним, тому цей варіант теж не є оптимальним і є сенс спробувати у майбутньому покращити цей алгоритм не втрачаючи точності розв'язання.

Список посилань

1. Pinedo M. L. Scheduling Theory, Algorithms, and Systems / Michael L. Pinedo., 2008. – (3rd edition).
2. Dakin R. J. A tree-searchalgorithmformixedintegerprogrammingproblems / Dakin. // TheComputerJournal. – 1965. – №8. – С. 250—255.
3. FOLDOC - ComputingDictionary [Електронний ресурс] – Режим доступу до ресурсу: http://foldoc.org.
4. Brucker P. SchedulingAlgorithms / Brucker., 2006.
5. Baptiste P. A BranchandBoundtoMinimizetheNumberofLateJobson a SingleMachinewithReleaseTimeConstraints / P. Baptiste, L. Peridy, E. Pinson. – 2003.
6. Танаєв В. С. Теорія розкладів. Одностадійнісистеми. / В. С. Танаєв, В. С. Гордон, Я. М. Шафранський., 1984. – (Наука).

УДК 004.032.26

ПОРТЯНИЙ І.С.

ВИКОРИСТАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ У ЗАДАЧІ РОЗПІЗНАВАННЯ ОБЛИЧЧЯ

У даній статті розглянуто застосування згорткових нейронних мереж для задачі розпізнавання обличчя. Описані основні методи для розпізнавання обличчя; архітектура згорткової нейронної мережі та функції основних шарів; процес розпізнавання обличчя. Також наведено огляд різних наборів даних для даної задачі.

КЛЮЧОВІ СЛОВА: РОЗПІЗНАВАННЯ ОБЛИЧЬ, ГЛИБОКЕ НАВЧАННЯ, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, ВИДІЛЕННЯ ХАРАКТЕРИСТИК ОБЛИЧЧЯ

This article discusses the use of convolutional neural networks for face recognition. Describedthe basic methods for face recognition; convolutional neural network architecture and core layer functions; face recognition process. An overview of the different datasets is also provided.

KEYWORDS: FACE RECOGNITION, DEEP LEARNING, CONVOLUTIONAL NEURAL NETWORKS, FACE FEATURES EXTRACTING

Вступ

Розпізнавання обличь – це процес знаходження та ідентифікації обличчя за допомогою системи комп'ютерного зору. Розпізнавання обличь використовується в системах безпеки, відеоспостереження, контролю доступу, оплати та навіть у соціальних мережах.

Комп'ютери не "бачать" фотографії та відео, як люди. З точки зору комп'ютера таке зображення – просто набір даних, які сприймаються як форми й інформація про значення кольорів. Вони формують цифрове зображення обличчя.

Системи розпізнавання обличь дуже активно впроваджуються у всьому світі, адже зараз навіть такі пристрої, як ноутбук чи смартфон, які присутні уповсякденному житті дозволять скористатися можливістюданої технології.

Наприклад, у Facebook користувач може увімкнути розпізнавання обличь, після чого створиться шаблон, за допомогою якого соціальна мережа зможе розпізнавати користувача на інших фото, відео та відео-трансляціях. А Китай використовує розпізнавання обличь громадян для виявлення порушників та пошуку злочинців

За останні роки даній сфері було присвячено безліч робіт і досягнуто значних успіхів.Зокрема, завдяки розвитку обчислювальних технологій значно пришвидшився процес обробки та сегментації зображень. Головною

складністю було забезпечення належної роботи системи під дією таких факторів, як освітленість, ракурс, а також вікові зміни.

2. Методи розпізнавання обличь

Вирішувати задачу розпізнавання обличчя можна методами неглибокого(shallowlearning) [3]–[5]та глибокого навчання(deeplearning) [1].

Традиційні методи розпізнавання обличь, засновані на неглибокому навчанністикаються з наступними проблемами: маскування обличчя, зміна пози, фон, різний вираз обличчя. Ці методи вміють визначати та використовувати лише деякі основні характеристикизображення.

На відміну від вищезгаданих методів, алгоритми глибокого навчання досить добре показали себе у виявленні складних структур в багатомірних даних. За допомогою методів глибокого навчання можна отримати більш широкий набір характеристик.

Існують різні підходи у глибокому навчанні, такі як: ConvolutionalNeuralNetworks(CNN–згорткові нейронні мережі), Stacked Autoencoder [6] (автоенкодери) та Deep Belief Network (DBN) [7](Глибинна мережа переконань). Але в основному для задач розпізнавання обличчя використовуються саме згорткові нейронні мережі.

3. Згорткові нейронні мережі(CNN)

CNN – це тип нейронних мереж, в яких використовується методологія згорток для знаходження ознак у вхідних даних задля збільшення кількості характеристик, які можна отримати з зображення. Згорткові нейронні мережі вперше запропоновані французьким вченим Яном Лекуном[8]і вперше використанні для розпізнавання рукописного тексту.CNNдоказали свою ефективність в сфері розпізнавання та класифікації зображень.

Згорткові нейронні мережі це тип нейронних мереж прямого зв'язку, які складаються з багатьох шарів(layer). CNNскладаються з фільтрів(або ядер), які мають свої ваги та параметри. Кожен фільтр приймає деякі вхідні дані та виконує згортку.

Типова архітектура CNN зображена на рисунку 1. Вона містить наступні базові шари: Convolutional Layer, Pooling Layer, ReLU(Rectified Linear Unit) Layer та Fully Connected Layer[9].

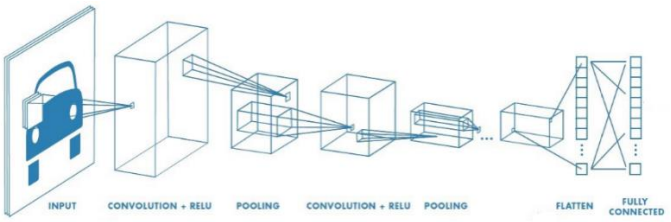


Рис. 1. Типова архітектура згорткової нейронної мережі

Згортковий шар(Convolutional layer) є одним з основних шарів згорткової мережі, який виконує основну масу обчислювальної роботи. Головне призначення цього шару – це витягнути ознаки з вхідних даних, у нашому випадку вхідними даними є картинки. За допомогою згортки зберігаються просторові пропорції між пікселями та визначаються особливості зображення використовуючи маленькі квадрати вхідного зображення. Вхідне зображення «згортається»завдяки застосуванню фільтрів до його частин, за допомогою чого розмірність картинки зменшується й створюється карта ознак (активаційна карта) вихідного зображення, після чого вона передається наступному згортковому шару.

Pooling layer зменшує розмірність кожної активаційної карти, але зберігає найбільш важливу інформацію. Вхідне зображення ділиться на набір прямокутників, які не перетинаються між собою, після чого до кожного з них застосовується понижуюча дискретизація за допомогою нелінійної операції, наприклад знаходження середнього (averagepooling) або максимального(maxpooling). Цей шар забезпечує найкраще узагальнення та найшвидшу збіжність. Він стійкий до спотворень та зазвичай розміщується між згортковимишарами.

ReLU являється нелінійною операцією – це поелементна операція, що означає, що вона застосовується до кожного елементу активаційної матриці. Застосування ReLU замінює всі від'ємні значення в активаційній карті на нуль. Ми можемо визначити цю операцію як $f(x)$, яка на вхід отримуєзначенняелементу активаційної матриціx і визначається наступним чином:

$$f(x) = \max(0, x),$$

де x – значення елементу активаційної матриці.

Останнім шаром є Fully Connected Layer (FCL). ТермінFully Connected Layer означає, що кожен фільтр на попередньому рівні пов'язаний з кожним фільтром на наступному рівні. Цей шар отримує оброблені попередніми шарами ознаки картинки, та на основі отриманих даних класифікує вхідне зображення по різних класах на основі навчального набору даних.

4. Місце CNNy задачі розпізнавання обличь

На рисунку 2 зображено основні етапи процесу розпізнавання обличчя.

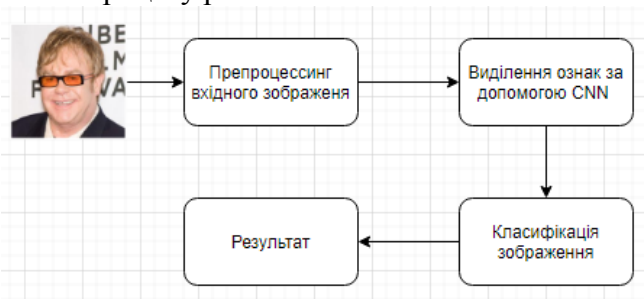


Рис. 2. Діаграма процесу розпізнавання обличчя

З рисунку вище можемо спостерігати наступні основні кроки в процесі розпізнавання обличчя:

1. Обробка вхідного зображення
2. Виділення ознак картинки за допомогою згорткових нейронних мереж
3. Класифікація зображення на основі отриманих ознак.

Для того, щоб CNNпрацювала коректно, вхідне зображення потрібно обробити: зменшити розширення зображення, обрізати фото так, щоб залишилось лише обличчя, зробити зображення чорно-білим(у деяких випадках). Дуже важливим моментом є також те, що всі вхідні зображення повинні бути зведеними до одного розширення.

Саме після того, як виконано препроцессинг зображення, застосовується згорткова нейронна мережа, яка генерує з вхідних даних набір ознак для обличчя.

Маючи готові ознаки ми можемо класифікувати та ідентифікувати вхідне обличчя.

5. Огляд наборів даних

На даний час існує достатня кількість безкоштовних наборів даних обличь. Який саме набір потрібно використовувати залежить від того, яку задачу потрібно вирішити. Нижче наведено різні набори даних, які відрізняються різними властивостями, такими як колір, 2D/3D-дані, фронтальне чи повернуте зображення.

Yale Face Dataset. Цей набір даних з 165 чорно-білих фото 15-ти людей. Для кожної людини є 11 фото з різними унікальними умовами та виразами обличчя: центральне світло, світло зліва,світло справа, в окулярах, без окулярів, нормальний вираз обличчя, щасливий, сумний, здивований та інші. На рисунку 3 наведено приклад даних цього набору.

Рис. 3. Приклад даних для Yale Face Dataset

Labeled Faces in the Wild. Це база даних зображень обличь розроблена для аналізу проблем розпізнавань обличь без конкретних умов щодо картинки. Набір містить більше 13 000 зображень обличь, отриманих з інтернету. На рисунку 4 наведено приклад даних цього набору.

Рис. 4. Приклад даних для Labeled Faces in the Wild

ORL Face Dataset. Цей набір даних містить по 10 унікальних зображень для 40-ка людей. Зображення виконані при різних умовах: слабка освітленість, сильна

освітленість та різні вирази обличчя. Приклад даних на рисунку 5.

Рис. 5. Приклад даних для ORL Face Dataset

У процесі навчання моделі важливо правильно обрати набір даних, на яких модель буде тренуватись в залежності від того, яка задача поставлена.

Якщо говорити про задачу розпізнавання емоцій на обличчі або про задачу автентифікації обличчя, то потрібно обирати набір даних, який містить добре оброблені фото, на яких міститиметься тільки обличчя й нічого зайвого. З вищезгаданих наборів даних для цієї задачі гарним вибором будуть Yale Face Dataset та ORL Face Dataset, так як вони містять лише обрізане обличчя з різними емоціями.

Для задачі знаходження та розпізнавання обличчя на фото краще обирати набір даних, де на зображеннях, окрім обличчя, будуть ще інші об’єкти. В таких умовах нейронна мережа зможе добре знаходити та розпізнавати обличчя серед інших об’єктів на фото.

6. Огляд та порівняння популярних нейронних мереж для розпізнавання обличчя

На даний момент існує достатньо багато видів архітектуризгорткових нейронних мереж, найпопулярніші з них: AlexNet, VGGNet,ResNet та Inception. Розглянемо кожну з цих архітектур.

AlexNet. AlexNet було створено з потреби покращити результати нейронної мережі ImageNet.Це була одна з перших глибоких згорткових мереж, яка досягла значної точності ззначенням 84,7% порівняно з другим найкращим результатом з точністю 73,8.

Нейронна мережа складається з наступних шарів: 11x11, 5x5,3x3, convolutions, maxpooling, dropout, data augmentation, ReLU activations, SGD with momentum. Після кожногоconvolution і fully-connected шару додається активація ReLU.. На рисунку 6 наведено архітектуру даної нейронної мережі [10].

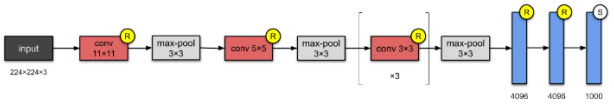


Рис. 6. Архітектура нейронної мережі AlexNet

VGGNet. VGGNet було створено з необхідності зменшення кількості параметрів у convolution-шарах та задля зменшення часу навчання моделі. Існує кілька варіантів VGGNet (VGG16, VGG19 тощо), які відрізняються лише загальною кількістю шарів у мережі. Детальніше розглянемо VGG16.

VGG16 має 13 convolutional і 3 fully-connected шарів, включаючи ReLU. Ця мережа використовує фільтри меншого розміру (2 × 2 і 3 × 3), ніж AlexNet. VGG16 має загалом 138 мільйонів параметрів. На рисунку 7 наведено архітектуру даної нейронної мережі [10].

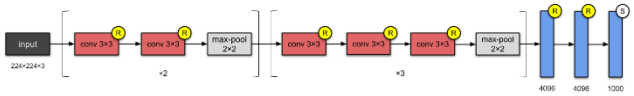


Рис. 7. Архітектура нейронної мережі VGGNet

ResNet. ResNet має близько 11 мільйонів параметрів. Архітектура складається з convolutional шарів з фільтрами розміром 3x3 (так само, як VGGNet), має лише 2 pooling шари на початку та вкінці мережі. На рисунку 8 наведено архітектуру даної нейронної мережі [10].

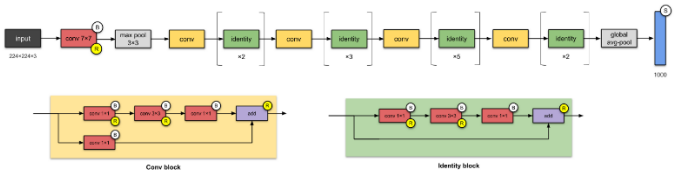


Рис. 8. Архітектура нейронної мережі ResNet

*Inception.*Inception збільшує простір нашої мережі, з якої ми можемо вибрати найкращу мережу після етапу навчання моделі. Замість того, щоб просто заглиблюватися за кількістю шарів, відбувається збільшення в ширину, тобто в межах одного шару реалізовано кілька ядер різного розміру. На рисунку 9 наведено архітектуру даної нейронної мережі[10].

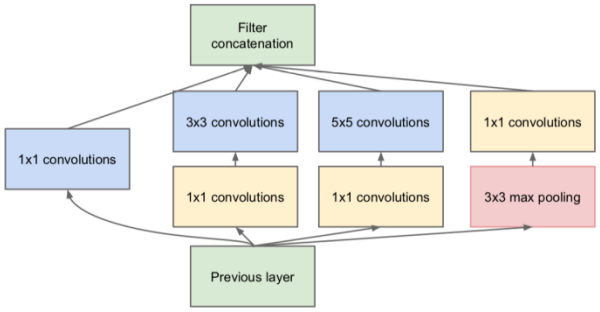


Рис. 9. Архітектура нейронної мережі Inception

Отже, порівняємо розглянуті архітектури. Порівняння приведено в таблиці 1[11].

Таблиця 1.5 – Порівняння розглянутих архітектур

Comparison					
Network	Year	Salient Feature	top5 accuracy	Parameters	FLOP
AlexNet	2012	Deeper	84.70%	62M	1.5B
VGGNet	2014	Fixed-size kernels	92.30%	138M	19.6B
Inception	2014	Wider - Parallel kernels	93.30%	6.4M	2B
ResNet-152	2015	Shortcut connections	95.51%	60.3M	11B

З таблиці бачимо, що найкращу точність має ResNet-152, але водночас вона має достатню велику кількість параметрів, та потребує більше пам’яті та ресурсів для тренування та роботи. Inceptionмає трішки меншу точність, але потребує значно менше ресурсів. Вибір оптимальної для себе архітектури залежить від доступних ресурсів.

Висновки

У даній роботі наведено алгоритм процесу розпізнавання обличчя та вказано роль згорткових нейронних мереж у ньому. Наведено опис згорткових нейронних мереж та їх архітектури.

Виконано огляд різних наборів даних та наведено рекомендації щодо їх використання в залежності від поставленої задачі.

Розглянуто найпопулярніші архітектури згорткових нейронних мереж та наведено короткий опис. Після чого виконано порівняння на основі їх точності та витрат ресурсів.

Список літератури

1. Nikhil B. Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms / Buduma Nikhil., 2017. – 277 с.
2. Himanshu S. Practical Machine Learning and Image Processing: For Facial Recognition, Object Detection, and Pattern Recognition Using Python / Singh Himanshu., 2019. – 165 с.
3. M. a. Turk and A. P. Pentland, “Face Recognition Using Eigenfaces,”Journal of Cognitive Neuroscience, vol. 3, no. 1. pp. 72–86, 1991.
4. J. Wright, a. Y. Yang, a. Ganesh, S. S. Sastry, and Y. Ma, “Robust facerecognition via sparse representation.,” IEEE Trans. Pattern Anal. Mach.Intell., vol. 31, no. 2, pp. 210–227, 2009
5. P. Sukhija, S. Behal, and P. Singh, “Face Recognition System UsingGenetic Algorithm,” in Procedia Computer Science, 2016, vol. 85.
6. R. Xia, J. Deng, B. Schuller, and Y. Liu, “Modeling gender informationfor emotion recognition using Denoising autoencoder,” in ICASSP,IEEE International Conference on Acoustics, Speech and SignalProcessing - Proceedings, 2014, pp. 990–994.
7. G. E. Hinton, S. Osindero, and Y. W. Teh, “A fast learning algorithm fordeep belief nets.,” Neural Comput., vol. 18, no. 7, pp. 1527–54,2006.[38] Y. Bengio, Learning Deep Architectures for AI, vol. 2,no. 1. 2009.
8. Y. LeCun et al., “Backpropagation Applied to Handwritten Zip Code Recognition,” Neural Comput., vol. 1, no. 4, pp. 541–551, Dec. 1989.
9. Rohan T. Convolutional Networks for everyone [Електронний ресурс] / Rohan Thomas. – 2018. – Режим доступу до ресурсу: https://medium.com/@rohanthomas.me/convolutional-networks-for-everyone-1d0699de1a9d.
10. Raimi K. Illustrated: 10 CNN Architectures [Електронний ресурс] / Karim Raimi. – 2019. – Режим доступу до ресурсу: https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d.

11. Aqeel A. Difference between AlexNet, VGGNet, ResNet and Inception [Електронний ресурс] / Aqeel Anwar. – 2019 –Режимдоступудоресурсу: https://towardsdatascience.com/the-w3h-of-alexnet-vggnetresnet-and-inception-7baaaecccc96

УДК 004.7

КАЛІНІЧЕНКО В.С.
ХАЛУС О.А.

ОПТИМІЗАЦІЯ РОБОТИ З DOM ШЛЯХОМ ІНТЕГРАЦІЇ JAVASCRIPT-ОБ'ЄКТА (VDOM)

В даній роботі представлений поверхневий огляд того, як працює концепція «віртуальний DOM», а також деякі з її ключових можливостей і особливостей.

КЛЮЧОВІ СЛОВА: віртуальний DOM, представлення інтерфейсу, тіньовий DOM, брудна перевірка.

This article provides a high-level overview of how virtual DOM conceptworks and some of the key features and capabilities.

KEYWORDS: virtual DOM, interface view, shadow DOM, dirty check.

1. Вступ

Віртуальний DOM (VDOM) - це концепція програмування, в якій ідеальне чи «віртуальне» представлення інтерфейсу користувача зберігається в пам'яті і синхронізується зі «справжнім» DOM. Цей процес називається узгодженням.

Як результат така техніка дозволяє нам поліпшити продуктивність на клієнтській стороні, уникаючи прямої роботи з DOM шляхом роботи з легким JavaScript-об'єктом, що імітує DOM-дерево.

2. Концепція звичайного DOM

З точки зору об'єктно-орієнтованого програмування, DOM визначає класи, методи та атрибути цих методів для аналізу структури документів та роботи із представленям документів у вигляді дерева. Все це призначено для того, аби надати можливість комп'ютерній програмі доступу та динамічної модифікації структури, змісту та оформлення документа.Через те, що структура документа зображена у вигляді дерева, повний зміст документа аналізується та зберігається в пам'яті комп'ютера. Тому, DOM підходить для використання в програмах, які вимагають багаторазовий доступ до елементів документа в довільному порядку. В разі, якщо треба лише послідовний або одноразовий доступ до елементів документа, рекомендується, для прискорення переробки та зменшення обсягів необхідної пам'яті комп'ютера, застосовувати послідовну модель роботи зі структурованими документами (SAX).

3. Основний механізм роботи віртуального DOM-дерева

Замість того, щоб взаємодіяти з DOM безпосередньо, робота відбувається з його полегшеною копією. Це надає можливість вносити зміни в копію, виходячи з потреб користувача, а після цього застосовувати зміни до реального DOM.

При цьому відбувається порівняння DOM-дерева з його віртуальної копією, визначається різниця і запускається перерисовка того, що було змінено.

Такий підхід працює швидше, бо не включає в себе всі тяжкі частини реального DOM.

Але тільки якщо ми робимо це правильно. Є дві проблеми: коли саме робити повторну перерисовку DOM і як це зробити ефективно.

4. Методи порівняння віртуального DOMзі звичайним

Є два варіанти дізнатися, що дані змінилися:

Перший з них - «dirtychecking» (брудна перевірка) полягає в тому, щоб опитувати дані через регулярні проміжки часу і рекурсивно перевіряти всі значення в структурі даних.

Другий варіант - «observable» (спостережуваний) полягає в спостереженні

за зміною стану. Якщо нічого не змінилося, ми нічого не робимо. Якщо змінилося, ми точно знаємо, що потрібно оновити.

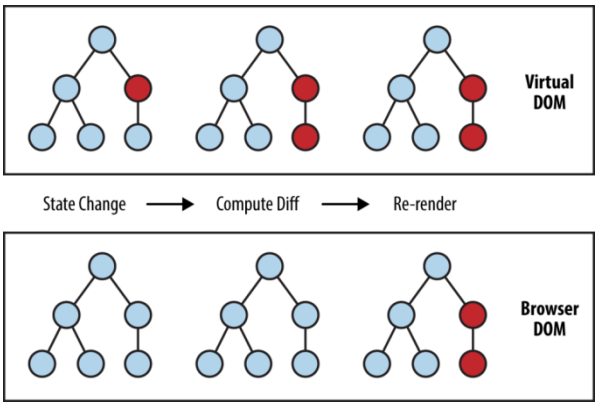


Рис. 2. Схема роботи системи з віртуальним DOM-деревом

5. Швидкодія роботи

Шляхи досягнення швидкодіїроботи з віртуальним DOM-деревом

- Використання ефективних алгоритмів порівняння
- Угруповання операцій читання / запису при роботі з DOM
- Ефективне оновлення тільки під-дерев

Реалізація може виявитися досить складною, але є деякі бібліотеки, які допомагають реалізувати цей підхід.

6. VirtualDOM на прикладі бібліотеки React.js

При зміні двох дерев, React спочатку порівнює два кореневих елемента. Поведінка по-різному залежно від типів кореневих елементів.

Всякий раз, коли кореневі елементи мають різні типи, React буде руйнувати старе дерево і будувати нове дерево з нуля. Перехід від посилання(<a>) до зображення(<img>), або від кнопки(<button>) до контейнеру( ) - будь-який з них призведе до повного відновлення.При зриві дерева старі вузли DOM знищуються. Отримують екземпляри компонентів componentWillUnmount(). При створенні нового дерева нові DOM-вузли вставляються в DOM.Отримують екземплярикомпонентів componentWillMount() і потім componentDidMount(). Будь-який стан, пов'язаний зі старим деревом, втрачається.Будь-які компоненти нижче кореня також будуть демонтовані і знищені. При порівнянні двох елементів React DOM того ж типу React розглядає атрибути обох, зберігає один і той же базовий вузол DOM і оновлює тільки змінені атрибути.

Висновок

Virtual DOM - це техніка та набір бібліотек (алгоритмів), які дозволяють нам поліпшити продуктивність на клієнтській стороні, уникаючи прямої роботи з DOM. Ідея з використанням віртуального DOM відмінна, хоча і не нова –давно відомо, що пряма робота з DOM обходиться дорого. Використовуючи бібліотеки на зразок React, ми можемо підвищити продуктивність додатків і зробити це дуже просто.

Література

1. Об'єктна модель документа: [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0
2. Що таке VirtualDOM [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/256965/
3. ReactJS - Reconciliation (сверка)[Електронний ресурс] – Режим доступу до ресурсу:http://unetway.com/tutorial/reactjs-reconciliation-sverka/

УДК 004.58

КОРОЛЬОВА Л.В.
ХАЛУС О.А.

МОДЕЛЮВАННЯ ЗОБРАЖЕННЯ ОДЯГУ НАДЯГНУТОГО НА ЛЮДИНУ ДЛЯ ОНЛАЙН ПРИМІРОЧНИХ

В даній статті оглядаються проблеми існуючих онлайн примірочних таописано сервіс який дозволяє генерувати модель людини та моделювати те як одяг буде сидіти на людині.Описано основний алгоритм сервісу, вимоги та програмні засоби задіяні для реалізації.

This article overviews the problems of online fitting rooms and describes a service that allows you to generate a modelof human and modelwhich shows how clothing will fit on a person. The basic algorithm of the service, requirements and software involved for implementation are described in the article.

Вступ

В наш час інтернет-магазини одягу є досить популярними. Проте через відсутність можливості фізично приміряти одяг виникає проблема. З боку покупців це небажання брати одяг через те що незрозуміло, чи правильного він розміру і чи буде пасувати. Ті ж хто покупають товари доволі часто їх повертають (в той час як у звичайних магазинів повернення товару складає приблизно 8% для інтернет-магазинів це приблизно 15% - 30%)[5]. Тому сервіс який дозволяє приміряти одяг онлайн знизив би витрати магазинів на повернення товару і викликав більше довіри у покупців.

1. Проблеми онлайн примірочних

Існуючі сервіси використовують як стилізоване[3] так і реальне[4] зображення одягу, протев основному не враховують виміри людини [3],[4]. Це дозволяє перевірити, чи пасують речі одна одній проте не показує, чи підходять речі конкретній людині. Тому однією з основних вимог до додатку є можливість моделювання того як одяг сидить на конкретній людині. Другої проблемою є технологічний процес: необхідно сфотографувати всі кольори моделі і обробити їх так щоб у зображення не було заднього фону. Автоматизація цього процесу значно пришвидшила б внесення одягу у базу даних магазину і знизило б ризик виникнення ситуації коли одяг у магазині є але функція примірки для нього недоступна.

2. Вимоги

Для вирішення зазначених вище проблем онлайн примірочна має відповідати таким функціональним вимогам:

- Можливість створення моделі людини на основі її вимірів
- Можливість створення стилізованого зображення на основі фото одягу
- Можливість створення моделі того, як одяг буде сидіти на людині

3. Завдання розробки

Щоб досягти вищезазначених вимог необхідно виконати наступні завдання.
Створення сервісу що на основі фото реального одягу:

1. визначає його колір/кольори
2. генерує стилізоване зображення одягу
3. надає можливість перефарбувати стилізоване зображення в інший колір

Створення сервісу що на основі вимірів людини та моделі одягу:

1. генерує стилізоване зображення людини з відповідними пропорціями
2. надягає на стилізоване зображення людини стилізоване зображення одягу

Алгоритм моделювання одягнутої на людину речі

Алгоритм попередньої обробки зображення одягу:

1. На вхід подається зображення одягу на однотонному фоні.
2. Видаляється фон зображення.
3. Зображення стилізується.

4. Пікселі зображення кластерезуються для отримання інформації про кольори та їх кількість.
5. Кластеризація зберігається для подальшого перефарбовування моделі.

Алгоритм примірки:

1. На вхід подається: виміри людини, модель одежі, її матеріал, розмір та колір.
2. Генерується модель людини.
3. Модель одягу перефарбовується у заданий колір.
4. Модель одягу деформується відповідно матеріалу та співвідношенню вимірів людини та розміру одягу.

4. Перелік задіяних технологій та програм

Сервіси для обробки фото одягу та створення моделі людини будуть реалізовані на мові програмування Python, а саме будуть задіяні такі бібліотеки як:

- pgmagic[1] для обробки та генерації зображень: видалення фону та змінення картинки;
- sklearn[2] для визначення кольору/кольорів картинки та способу обробки зображення;
- інші побічні бібліотеки для роботи з даними такі як numpy і pandas.

Висновок

Отже в даній статті було розглянуто проблему інтернет-магазинів пов’язану з відсутністю фізичних примірочних, а саме висока частка повернень товарів. Було проведено огляд існуючих примірочних та розглянуто їх проблеми, а саме відсутність моделювання того, як одяг сидить на людині враховуючи її виміри. Також були розглянуті засоби та алгоритм за допомогою яких можна реалізувати визначені вимоги. Як результат розглянутий сервіс матиме такі важливі переваги як врахування параметрів людини та відсутність необхідності фотографувати всі кольори конкретної моделі одягу.

Список літератури

1. Pgmagickdocumentation. [Електронний ресурс] // Режим доступу: https://pgmagick.readthedocs.io/en/latest/
2. Scikit-learn official site. [Електронний ресурс] // Режимдоступу: https://scikit-learn.org/stable/
3. Приклад примірочної без урахування вимірів людини [Електронний ресурс] // Режим доступу: http://www.styleclub.com.ua/model_wardrobe.aspx#modeltop
4. Приклад примірочної з реальними фото одягу [Електронний ресурс] // Режим доступу: https://showroom.onvolga.com/demo/fitting-clothes/
5. Огляд технології від Amazon[Електронний ресурс] // Режим доступу: https://habr.com/ru/company/pochtoy/blog/412023/

УДК004.7

ЛІСОГОР А.Ю.
ХАЛУС О.А.

МОБІЛЬНЕ ЗАСТОСУВАННЯ ЗІ СЧИТУВАННЯ ТЕКСТУ ДЛЯ ЛЮДЕЙ З ВАДАМИ ЗОРУ

В даній роботі описана ідея мобільного застосування зі считування рукописного тексту та його зачитування, що в перспективі може допомогти людям з вадами зору. Висунутіосновнівимоги, переліченіобранітехнологіїдляреалізації.

КЛЮЧОВІ СЛОВА: мобільне застосування, CoreML, зчитування тексту, комп’ютерний зір

This paper describes the idea of a mobile application for scanning and reciting handwritten text that can help people with visual impairments in the future. The basic requirements are listed, the main technologies for implementation are enumerated.

KEYWORDS: mobile application, Core ML, text scanning, OCR

Вступ

Мобільні пристрої в наш час кардинально змінили життя людей з порушеннями зору: завдяки функціям VoiceOver (iOS) і TalkBack (Android) для них доступні всі звичні програми: месенджери і онлайн-банки, а також десятки спеціальних, які визначають номінали купюр, дозволяють дивитися кіно з коментарями і дають можливість питати поради у незнайомих волонтерів в окремій соцмережі. І хоча наш світ досить впевнено крокує в повну цифровізацію, не всі сфери повсякденного життя мають змогу цим похвалитися.

Уявіть, що людина з вадами зору отримує лист, текст якого написано від руки. Це може стати серйозною перешкодою.

На допомогу для вирішення такої проблеми може прийти мобільний пристрій з камерою. Компанія Apple надає широкий спектр технологій для розробника, що дозволяють йому впровадити алгоритми комп'ютерного зору в свій проект (Vision framework) та полегшити класифікацію об'єктів(Core ML).

Метою розробки є створення застосунку, який дозволяє зчитування тексту використовуючи камеру смартфону, надає звуковий супровід основних етапів під час використання застосунку (Наведення на текст, розпізнавання, завершення розпізнавання тощо.), аналізує отриману інформацію та зачитує її.

1. Перелік задіяних технологій та програмних продуктів

1.1. Vision framework - фреймворк розроблений компанією Apple, що надає можливість задіяти високо продуктивний аналіз зображень та технологію комп'ютерного зору до зображень та відео. Дозволяє виявляти штрих-коди, текст, об'єкти і тд.

В Vision наявні 3 основні категорії класів: VNRequest, VNSequence Request Handler, VNObservation. VNRequest - цей абстрактный суперклас описує запроси користувача до Vision (VNDetect Text Rectangles Request, VNDetect Barcodes Request Request та інші).

VNSequence Request Handler - виконує бажану кількість реквестів для зображення.

VNObservation - абстрактный суперклас для результатів аналізу зображення.

Отже, механізм роботи Vision виглядає таким чином, як зображено на рисунку 1.

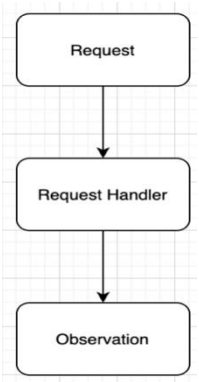


Рис.1. Механізм застосування Vision

1.2. Core ML - декілька років тому компанія Apple представила цей фреймворк, який сильно полегшує роботу з технологіями машиного навчання. В його основі лежить ідея про те, що можна взяти преднавчену модель даних та легко інтегрувати її в свій застосунок. Core ML використовую низькорівневі Metal, Accelerate, BNNS і тому результати обчислень надходять дуже швидко. Ядро підтримує нейронні мережі, generalized linear models, feature engineering, tree ensembles, pipeline models.

1.3. Keras - відкрита нейромережна бібліотека, що створена на мові Python, що дозволяє створювати моделі глибокого навчання. Keras працює за такими принципами:

1. Модульність – модель можна розуміти як послідовність або графік.
2.Мінімалізм – бібліотека забеспечує достатньо для досягнення результату, без надмірностей
3. Розширюваність – нові компоненти легко додавати, дослідження нових ідей можливе

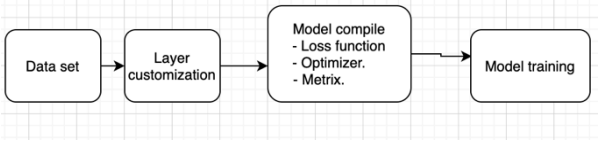


Рис. 2. Схема роботи Keras

1.4 Coremltools – це пакет pythonдля створення, вивчення та тестування моделей у форматі .mlmodel. Зокрема, його можна використовувати для:

1. перетворювати підготовлені моделі в формат .mlmodel

2. перевіряти конверсію використовуючи CoreML.

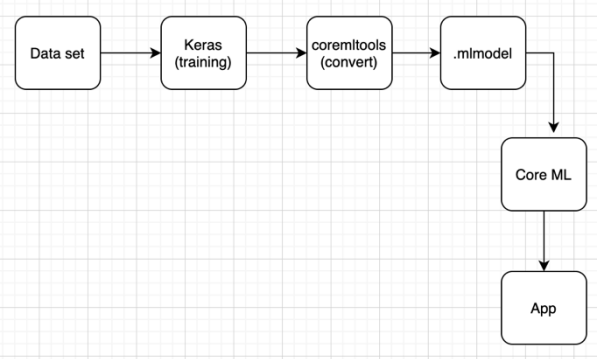


Рис.3. Схема створення .mlmodelдля CoreML

2. Функціональні та нефункціональні вимоги

Застосунок створено на платформі IOS - однієї з найпопулярніших мобільних платформ. Застосунок має
такі функціональні вимоги:
1. Зчитування рукописного тексту на зображенні
2. Класифікування на слова та літери
3. Розпізнання слів та літер
4. Збереження тексту
5. Зачитування тексту
Нефункціональні вимоги:

1. Застосунок має працювати на пристроях з версією IOS 12 та вище.
2. Простий та зрозумілий інтерфейс.

3. Завдання розробки

Для виконання поставлених функціональних вимог має бути створені:

1. Зручний User Interface
2. Модуль для сканування, розпізнавання, та зачитування рукописного тексту.

User Interface має бути побудований із стандартних елементів IOS та бути лаконічним і зрозумілим.
Модуль розпізнавання рукописного тексту повинен базуватися на взаємодії таких фреймворків:

- Vision Framework
- Core ML Framework

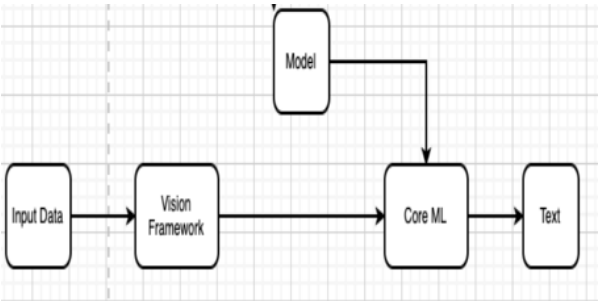


Рис.4. Схема роботи модуля розпізнавання рукописного тексту

Висновок

Отже, в даній статті було розглянуто технології, фреймворки та програми, які існують для обробки, зчитування різних видів інформації(Тексти, Штрих-коди тощо). Були розглянуті такі технічні рішення як: Vision framework, Core ML, Keras, coremltools. Припущено, що програмний продукт при взаємодії з такими технологіями буде здатен вирішити питання отримання інформації з рукописних джерел для людей з вадами зору. Були описані функціональні та нефункціональні вимоги, обрано платформу для створення застосунку.

Список літератури

1. Vision Framework. [Електронний ресурс] // Режим доступу: https://developer.apple.com/documentation/vision
2. Core ML Framework. [Електронний ресурс] // Режим доступу:https://developer.apple.com/documentation/coreml
3. Keras. [Електронний ресурс] // Режим доступу: https://keras.io/getting-started/sequential-model-guide/
4. CoreMLTool [Електронний ресурс] // Режим доступу: https://developer.apple.com/documentation/coreml/converting_trained_models_to_core_ml

УДК 004.7

ПИЛИПЕНКО В.О.
ХАЛУС О.А.

ОПТИМІЗАЦІЯ ЗНАХОДЖЕННЯ ВІДСОТКОВОГО ЗІСТАВЛЕННЯ ТЕКСТУ ШЛЯХОМ ВИКОРИСТАННЯ “ВІДСТАНІ ЛЕВЕНШТЕЙНА”

В даній роботі представлений огляд такого поняття як відстані Левенштейна, а також спосіб її знаходження.

КЛЮЧОВІ СЛОВА: відстані Левенштейна, операція вилучення, операція заміни, операція вставки.

This article provides an overview of the concept of Levenshteindistance and how it is found.

KEYWORDS: Levenshteindistance, removaloperation, replacementoperation, insertionoperation.

1. Вступ

Відстань Левенштейна — метрика, що дозволяє визначити «схожість» двох рядків, тобто мінімальна кількість операцій вилучення, заміни або вставки символа, необхідних для перетворення одного рядка в інший.

2. Визначення

Для розрахунку відстані Левенштейна найчастіше застосовують простий алгоритм, в якому використовується матриця розміром (n + 1) * (m + 1), де n і m - довжини порівнюваних рядків. Окрім цього вартість операцій вилучення, заміни та вставки вважається однаковою. Для конструювання матриці використовується рекурентне рівняння зображене на ***рис. 1.***

$$D_{0,0} = 0$$

$$D_{i,j} = min \begin{cases} D_{i-1,j-1} & +0 \text{ (equal, noch} \\ D_{i-1,j-1} & +1 \text{ (replace)} \\ D_{i-1,j} & +1 \text{ (insert)} \\ D_{i,j-1} & +1 \text{ (delete)} \end{cases}$$

Рис. 1. Рекурентнерівняння для знаходженнявідстані Левенштейна

3. Верхня та нижня межі

Відстань Левенштейна має кілька простих верхніх і нижніх меж. До них належать:

- Різниця розмірів двох рядки.
- Не є більшою довжини найдовшого рядка
- Дорівнює 0, тоді і лише тоді, коли рядки рівні.
- Якщо рядки однакового розміру, відстань Хеммінга є верхньою межею відстані Левенштейна.
- Між двома рядками не більше суми відстаней Левенштейна від третього рядка (нерівність трикутника)

4. Приклад

Для визначення відстані Левенштейна між словами “Sunday” і “Saturday” побудуємо матрицю за формулою описаною раніше. Для наглядності матрицю представимо у вигляді таблиці:

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	4	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

Отже, відстань Левенштейна між словами “Sunday” і “Saturday” = 3.

5. Псевдокод алгоритма

Для розрахунку відстані Левенштейна найчастіше застосовують простий алгоритм, в якому використовується матриця розміром

```
intLevenshteinDistance(
charstr1[1..lenStr1],
char str2[1..lenStr2])
// d таблицякількістьрядків =
lenStr1+1 такількістьстовпців =
lenStr2+1
declareintd[0..lenStr1, 0..lenStr2]
// іта j
використовуютьсядляіндексуванняпозиці
ї у str1 та у str2
declareinti, j, cost

forifrom 0 to lenStr1
d[i, 0] := i
for j from 0 to lenStr2
d[0, j] := j

forifrom 1 to lenStr1
for j from 1 to lenStr2
if str1[i] = str2[j]
thencost := 0//однакові
elsecost := 1   //заміна
d[i, j] := minimum(
               d[i-1,j]+1,//вилучення
d[i,j-1]+1,//вставка
                d[i-1,j-
               )

returnd[lenStr1, lenStr2]
//значеннявідстаніЛевенштайна в
останнійклітинціматриці
```

6. Застосування алгоритму

Основні залузі застосування відстань Левенштейна:

- виправлення помилок в слові (в пошукових системах, базах даних, при введенні тексту, при автоматичному розпізнаванні відсканованого тексту).
- порівняння текстових файлів. Тут ролі «символів» виступають рядки, а ролі «рядків» - файли.
- в біоінформатиці для порівняння генів, хромосом і білків.

Висновок

Відстань Левенштейна — ефективна міра для визначення «схожості» двох рядків. Даний підхід може бути застосований для великих рядків (близько 10^5 символів, тобто фактично для текстів) при одержанні не тільки оцінки «схожості», а й послідовності змін для перекладу одного рядка в іншу.

Література

1. Відстань Левенштейна: [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D1%81%D1%82%D0%B0%D0%BD%D1%8C_%D0%9B%D0%B5%D0%B2%D0%B5%D0%BD%D1%88%D1%82%D0%B5%D0%B9%D0%BD%D0%B0
2. Візуалізація алгоритму Левенштейна [Електронний ресурс] – Режим доступу до ресурсу: http://www-igm.univ-mlv.fr/~lecroq/seqcomp/node2.html
3. Розрахуноквідстаніміж рядками - Левенштейн і Геммінг [Електронний ресурс] – Режим доступу до ресурсу: http://www.cut-the-knot.org/do_you_know/Strings.shtml

УДК 004.93

НОВІЧЕНКО Н.В.

ЗАСТОСУВАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ПРИ КЛАСИФІКАЦІЇ ЗООБРАЖЕНЬ БУДІВЕЛЬ ЗА АРХУТЕКТУРНИМ СТИЛЕМ

З розвитком технологій комп'ютерного зору та постійно зростаючу кількість цифрової документації з'явилася необхідність у автоматизації процесу аналізу інформації. Визначення архітектурних стилів будівель поширена задача в архітектурі при моделюванні та документуванні архітектурної спадщини. В статті розглянуто практичне застосування згорткових нейроннихмереж для вирішення задачі класифікації архітектурних стилів будівель на навчальній вибірці, що включає 25 архітектурних стилів.

МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ, АРХІТЕКТУРНІ СТИЛІ

With the development of a technology computer vision and a constant in crease in the number of digital documents, there is a need for automation in the processof information analyzis. Defining the architectural styles of buildings is a common challenge in architecture when modeling and documenting architectural heritage. The articl edeals with the practical application of convolutional neural networks to solvethe problem of classifying architectural style sofbuildings by training model, which includes 25 architectural styles.

MACHINE LEARNING, NEURAL NETWORKS, IMAGE CLASSIFICATION, ARCHITECTURAL STYLES

1. Вступ

Архітектурний стиль — сукупність основних рис та ознак архітектури певного історичного часу і місця, яка проявляється у функціональних, конструктивних, мистецьких особливостях будов. При визначенні стилю будівля розглядається як набір деталей, кожна зі своїми характеристиками, які в комбінації визначають архітектурний стиль. Кожен стиль відрізняється власними стійкими художніми формами, описує похідну епоху і визначає культурне обличчя епохи та країни у певний період. Наприклад, раціоналізм – напрямок в архітектурі 20 ст., намагається не тільки відобразити в архітектурі ті чи інші життєві процеси, а й активно впливати на ці процеси «винайти» для них нові форми. Характеризується цей стиль естетизацією сучасної техніки (що схоже на техніцизм), пошуки виразності простих геометричних форм, відмова від декору, інтерес до пропорціям, до кольору, до синтезу архітектури з іншими мистецтвами. Зовнішні ознаки – геометрична простота форм, асиметрія, мінімалізм у декору, використання металу, залізобетону та скла. Раціоналізм – спрощеність і стандарт. [1]

Складність визначення архітектурного стилю полягає у схожості основних рис та ознак у декількох архітектурних стилів, а також у можливості існування територіальних особливостей кожного стилю. Необхідність мати можливість визначати стиль для зображень з різних ракурсів та різної якості накладає додаткові вимоги до методів класифікації зображень.

2. Згорткові нейронні мережі для класифікації зображень

Згорткові нейронні мережі (CNN) – це особлива архітектура штучних нейронних мереж, що заснована на механізмах зорової кори, саме тому ця архітектура є найбільш популярною для вирішення задачі класифікації зображень. Алгоритм CNN включає в себе 2 основних процеса: згортка та обєднання. Зображення передається через ряд згорткових, нелінійних об'єднуючих шарів (шарів спрощення) і повністю пов'язаних шарів, а потім генерує вихід. На шар згортки завантажується зображення (матриця з пікселів розміру n × n). Далі вибирається матрицю, що є частиною загальної матриці, яку називають фільтром (або нейроном, або ядром) за наступним принципом:

$$x_{ij}^{(l)} = \sigma(\mathrm{b} + \sum_{r=0}^{n}\sum_{c=0}^{n} W_{r,c}x_{i+r,j+c}^{(l-1)})$$

Ця операція повторюється для кожного фільтру, що отримується переміщенням вбік (або вниз) 1 одиницю, виконуючи подібну операцію. Після проходження фільтра по всіх позиціях отримують матрицю, але меншу, ніж вхідну матрицю.

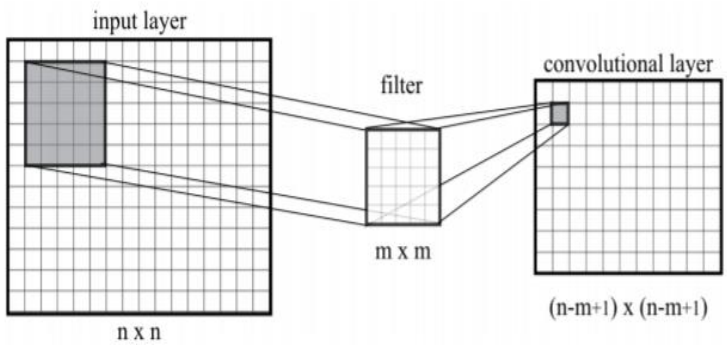


Рис. 1 – Процес згортки

Нейронна мережа складається з декількох згорткових мереж, поєднаних з обєднуючими шарами. Обєднуючі шари завжди йдуть після згорткових та мають функцію активації, що «виділяє» кордони на зображенні і робить матрицю більш «контрастною». Зазвичай використовують стандарну функцію активації наступного вигляду:

$$y(x) = (1 + e^{-x})^{-1},$$

або

$$y(x) = \tanh(x).$$

Ця операція аналогічна то процесу виділення людиною кордонів та меж предметів на зображенні. Процес обєднання(спрощення) зображений на рисунку 2.

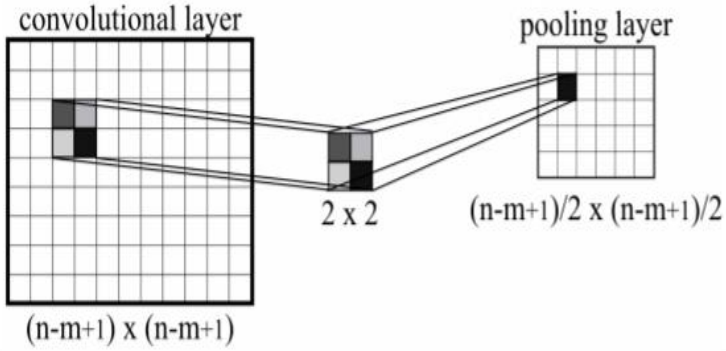


Рис. 2 – Процес обєднання(спрощення)

Обєднуючий шар дозволяє коригувати розміри матриці пікселів – зменшувати розмірність стовбців та рядків.

Після прогону матриці на вище описаних шарах вона готова до аналізу нейронною мережою, тому що зображення достатньо зменшене в розмірах. Матриця потрапляє на «fullyconnectedlayer», де відбувається проекція матриці на вектор вхідного шару [2][3].

Саме така структура дозволяє працювати з зображеннями різних розмірів та якості.

3. Інші методи класифікації зображень

До того, як CNN набули популярності, для класифікації зображень використовували наступні методи: розробники виділяли певні «особливості» на зображеннях – певні об’єкти і саме на цих об’єктах виконувалася класифікація за допомогою таких алгоритмів, як SVM. Деякі алгоритми працювали на основі значень рівнів пікселів, як виділених об’єктах.

CNN можна розглядати як автоматичні екстрактори об'єктів із зображення за допомогою своєї структури. Порівняно з SVM алгоритмом, CNN активно використовує інформацію про сусідні пікселі, що дозволяє більш ефективно спрощувати зображення, чого не дозволяє зробити SVM.

Як приклад, можно розглянути – роботу ГаянеШалунц, ЮллХашимуса і Роберт Саблатинга «Класифікація архітектурного стилю фасаду будівлі» без використання

машинного навчання: алгоритм використовує підхід на основі градієнтних напрямків з використанням SVM на 4 класи для класифікації архітектурного стилю з особливим фокусом на «зворотному процесуальному моделюванні» – використовуючи зображення для створення генеративної процедурної моделі для 3D-графіки [4].

Інший приклад - «Класифікація зображень архітектурної спадщини методами глибокого навчання» виконана Хосе Ламасом, Педро М. Леронесом та іншими, основною метою поставили оцінку застосовності різних методів класифікації зображень архітектурної спадщини. В результаті даної роботи були отримані різні навчені нейронні сітки, спроможні розрізняти різні частини будівель (колони, горгульї, дзвіниці, тощо) [5].

4. Практичне застосування згорткових нейронних мереж (CNN) для класифікації зображень

Для вирішення задачі класифікації було обрано архітектуру згорткових нейронних мереж – ResNet.CNN«витягають» низько-, середньо- тависокорівневі ознаки зображення наскрізним багатошаровим способом та при збільшення кількості шарів можнадосягти визначення більшої кількості «рівнів» ознак. Більшість архітектурCNN (ImageNet) дають покрашення результатів при збільшені кількості шарів, проте при досягненні певного значення результати різко погіршуються. Для уникнення цієї проблеми при необхідності збільшення шарів нейронної мережі, було використанно архітектуру ResNet, яка за рахунок глибокої «залишкової» структури навчання не дає гірші результати при збільшенні кількості шарів.

Результати класифікації зображень будівель з використанням створеного алгоритму наведені на рисунках 4 – 6 у вигляді пар архітектурний стиль – ймовірність.

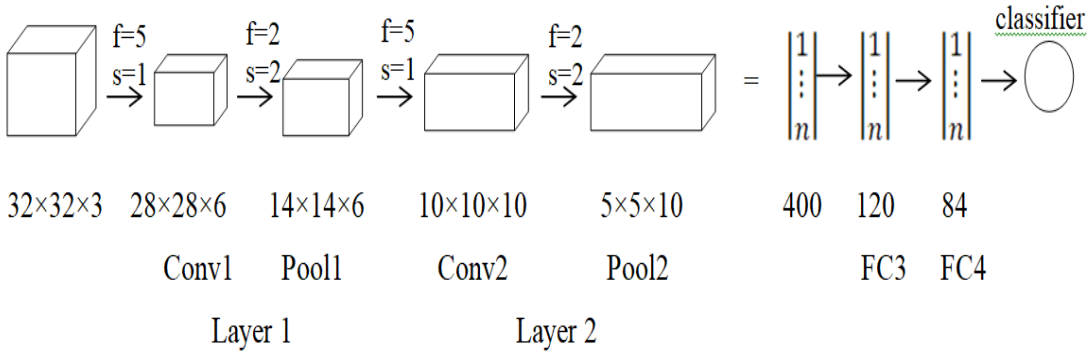
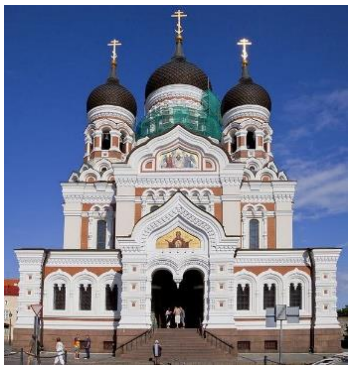


Рис.3 – Спрощена модель CNN

chicagoSchool : 72.9
artDeco : 11.3
artNouveau : 10.8

Рис. 4 – Результати класифікації будівлі архітектурного стилю ChicagoSchool



russianRevival : 68.5
baroque : 15.3
colonial : 11.4

Рис. 5 – Результати класифікації будівлі архітектурного стилю RussianRevival

queenAnne : 71.6
tudorRevival : 7.5
artNouveau : 4.1

Рис.6 – Результати класифікації будівлі архітектурного стилю QueenAnne

Висновок

Було розглянуто варіант вирішення задачі класифікації зображення будівлі за архітектурним стилем з використанням згорткових нейронних мереж та наведено результати класифікації отримані за допомогою створенного алгоритму. Для класифікації, що вимагає аналізу данних, виділення основних «ознак» серед данних доцільно використовувати саме згорткові нейронні мережі через їх особливу архітектуру, що включає процеси згортки та обєднання. Саме така структура дозволяє працювати з зображеннямибудівель з різних ракурсів, різних розмірів та якості. Для виключення погіршення результатів при збільшені глибини нейронної мережі доцільно використовувати архітектуру CNN – ResNet.

Перелік посилань

1. Земляна і. О. Найбільш значимі архітектурні стилі [електронний ресурс] / Ірина Ілександрівна Земляна // класна оцінка. – 2011. – Режим доступу до ресурсу: http://klasnaocinka.com.ua/ru/article/naibolee-znachimie-arkhitekturnie-stili.html.
2. Sorokina K. Image Classification with Convolutional Neural Networks [Електронний ресурс] / Ksenia Sorokina // A Medium Corporation. – 201. – Режим доступу до ресурсу: https://medium.com/@ksusorokina/image-classification-with-convolutional-neural-networks-496815db12a8.
3. Chen W. Convolutional Neural Network for Image Classification / W. Chen, X. Yang.. – (Johns Hopkins University).
4. Shalunts, Gayane, Yll Haxhimusa, and Robert Sablatnig. "Architectural style classification of building facade windows." International Symposium on Visual Computing. Springer Berlin Heidelberg, 2011.
5. Classification of Architectural Heritage Images Using Deep Learning Techniques Jose Llamas 1,* ID , Pedro M. Lerones 1 , RobertoMedina 1 , Eduardo Zalama 2 and Jaime Gómez-García-Bermejo 2 ID https://pdfs.semanticscholar.org/1a1c/4e75c74b715fcc0903a044c4f7aa3d3bbf1c.pdf.

УДК 004.912

ДВОРНИК В. А.

ЗАСТОСУВАННЯ МЕТОДІВ КЛАСТЕРИЗАЦІЇ ДЛЯ ВИЗНАЧЕННЯ СФЕР ДІЯЛЬНОСТІ КАНДИДАТІВ ПРИ ПІДБОРІ КАДРІВ ДЛЯ ІТ-КОМАНІЙ

У даній статті розглянуто проблему підбору кадрів для ІТ-компаній. Розглянуто практичне методів кластеризаціїна прикладі задачі визначеннясфер діяльності кандидатів при підборі кадрів для ІТ-компаній. Приведено постановку задачі.Визначено, до яких моделей зводяться досліджувані проблемні ситуації, та, які методи можуть бути застосовані до розв'язання поставленої задачі. Наведено огляд відомих рішень, також висвітлено переваги та недоліки обраного методу.

КЛЮЧОВІ СЛОВА: КЛАСТЕРИЗАЦІЯ, СФЕРА ДІЯЛЬНОСТІ, КАНДИДАТ, РЕКРУТЕР, ПІДБІР КАДРІВ.

This article addresses the problem of recruiting for IT companies. The practical methods of clustering on the example of determining the areas of activity of candidates in the recruitment of IT companies are considered. The formulation of the problemis given. It is determined to which models the problematic ituation sarereduced and what methods can be applied to solve the problem. Anover view of the knowns olutionsis given, as well as the advantage and disadvantages of the chosen method.

KEYWORDS: CLUSTERIZATION, SPHERE OF ACTIVITY, CANDIDATE, RECRUITMENT, RECRUITER.

1. Вступ

На сьогоднішній день підбір кандидатів для найму з широкого кола кандидатів є основоположним питанням. Традиційними методами є проведення індивідуальних перевірок і різних технічних кваліфікаційних тестів, співбесід та групових обговорень. Виявлення здібностей кандидата за допомогою співбесід є традиційною практикою в процесі найму [1].

Сучасні менеджери з управління персоналом та кадровики повинні обробляти надзвичайно великі обсяги даних: дослідження портфоліо, скринінг соціальних медіа, ідентифікація наборів навичок, а найголовніше – дослідження резюме. Існує необхідність вилучення відповідної інформації з резюме і збереження її в базі даних, щоб опрацьовувати дані стало легше. Крім того, резюме варіюються за форматом і стилем, що ускладнює ведення структурного сховища, яке містити всю необхідну інформацію

Таким чином, дослідження відповідних відомостей з резюме часто є процесом, який займає багато часу та зусиль для фахівців з управління ресурсами та підбору персоналу.

2. Постановка задачі

У задачі визначення сфер діяльності працівників при підборі кадрів для ІТ-компаній в якості вхідної інформації розглядатимуться резюме у текстовому вигляді, в яких буде міститись уся інформація про професійну кар'єру працівника, а також мотиваційні листи, есе та тести з професійної орієнтації з вільними відкритими відповідями.

На виході отримаємо набір професійних сфер діяльності працівників з підібраними до них найкращими резюме, тобто, отримаємо групуваннявхідних даних до певних сфер діяльності.

Для групування та об'єднання вхідних даних пропонується залучити методи кластеризації текстових даних.

Таким чином, нехай X – множина об'єктів, тобто резюме, а Y – множина кластерів, тобто професійних сфер. Задана функція відстані між об'єктами $\rho(x, x')$. Маємо кінцеву навчальну вибірка об'єктів $X^m = \{x_1, \dots, x_m\} \subset X$

Необхідно розбити вибірку на підмножини (кластери), тобто кожному об'єкту $x_i \in X^m$ поставити у відповідність $y_i \in Y$таким чином щоб об'єкти всередині кожного кластера були близькі щодо метрики ρ, а об'єкти з різних кластерів істотно розрізнялися[2].

3. Огляд відомих рішень

Автоматизовану класифікацію резюме за допомогою техніки кластеризації розглядали також професори Сагар Море, БхамареПриянка, Малі Пуджа та КачавеКаляні [3].

Вчені наголошують на тому, що на сьогоднішній день важким завданням для менеджерів з підбору персоналу є завдання

знайти найкращого кандидата який би відповідав усім побажанням та виправдовував усі очікування. Основна проблема полягає на початковому рівні. Традиційний підхід при подачі резюме для отримання роботи полягає в пошуку посади, а потім в направленні резюме на знайдену посаду. Один із методів опрацювання надісланих резюме полягає в фільтрації, але виявлення потенційних кращих резюме проводиться шляхом вивчення кожного резюме, оскільки ці відфільтровані резюме аналогічні один одному. В рамках запропонованого вченими методу вони зосередили увагу на підвищенні ефективності процесу відбору кандидатів. Пропонується метод, що дозволяє відповідним чином визначити особливості та навички у кожному з надісланих резюме.

Цей підхід застосовує ідею кластеризації. На простому рівні кластеризація використовує один або кілька атрибутів в якості основи для ідентифікації кластера. Кластеризація корисна для ідентифікації різної інформації, так як вона корелює з іншими прикладами, так що можна побачити, де подібності та діапазони збігаються.

Вчені поставили перед собою завдання розробити систему, яка дозволить департаменту з людських ресурсів публікувати оголошення про вакансії з вимогами, що пред'являються до посад, а також створити портал для того, щоб кандидат міг завантажити своє резюме, і надати працівникам з підбору персоналу набір з резюме, які підходять для певної вакансії.

Пропоноване вченими рішення використовує методи інтелектуального аналізу даних. Інтелектуальний аналіз даних – це обробка даних і виявлення закономірностей і тенденцій в цій інформації, для прийняття рішення [4]. Метод кластеризації інтелектуального аналізу даних використовується для класифікації та розрахунку. Оскільки кластеризація корисна для ідентифікації різної інформації, так як вона корелює з іншими прикладами, так що можна бачити, де подібності та діапазони збігаються [4].

Система націлена на скорочення часу і зусиль департаменту з людських ресурсів у процесі відбору резюме.

Для кластеризації у системі застосовується алгоритм k-means. Цей алгоритм кластеризує резюме кадидатів у k кластерів.

4. Опис методу розв'язання задачі

Розглянемо методи кластеризації. Кластеризація – це поділ множини вхідних векторів на групи (кластери) за ступенем «схожості» один на одного. Для того, щоб можна було порівнювати два об'єкти, потрібно мати критерій, на підставі котрого і буде відбуватися порівняння. Зазвичай, як правило, таким критерієм є відстань між об'єктами [5].

Алгоритми кластеризації поділяються на дві основні категорії [6]:

1. Чіткі та нечіткі алгоритми
2. Ієрархічні і плоскі

Чіткі алгоритми ставлять у відповідність кожному об'єкту з вибірки номер визначеного кластера, тобто кожен об'єкт належить лише одному кластеру. Нечіткі алгоритми кожному об'єкту ставлять у відповідність набір дійсних значень (чисел), що показують ступінь відношення об'єкта до усіх кластерів. Тобто, кожен об'єкт відноситься до кожного кластеру з певною ймовірністю.

Наступна категорія – це ієрархічні алгоритми, котрі будують не одне розбиття вибірки на непересічні кластери, а систему вкладених розбиттів. Таким чином, на виході отримується дерево кластерів, коренем якого є вся вибірка, а листками – найдрібніші кластери. Плоскі алгоритми будують одне розбиття об'єктів на кластери.

На рисунку 1 показано більш детальну класифікацію алгоритмів кластеризації [7].

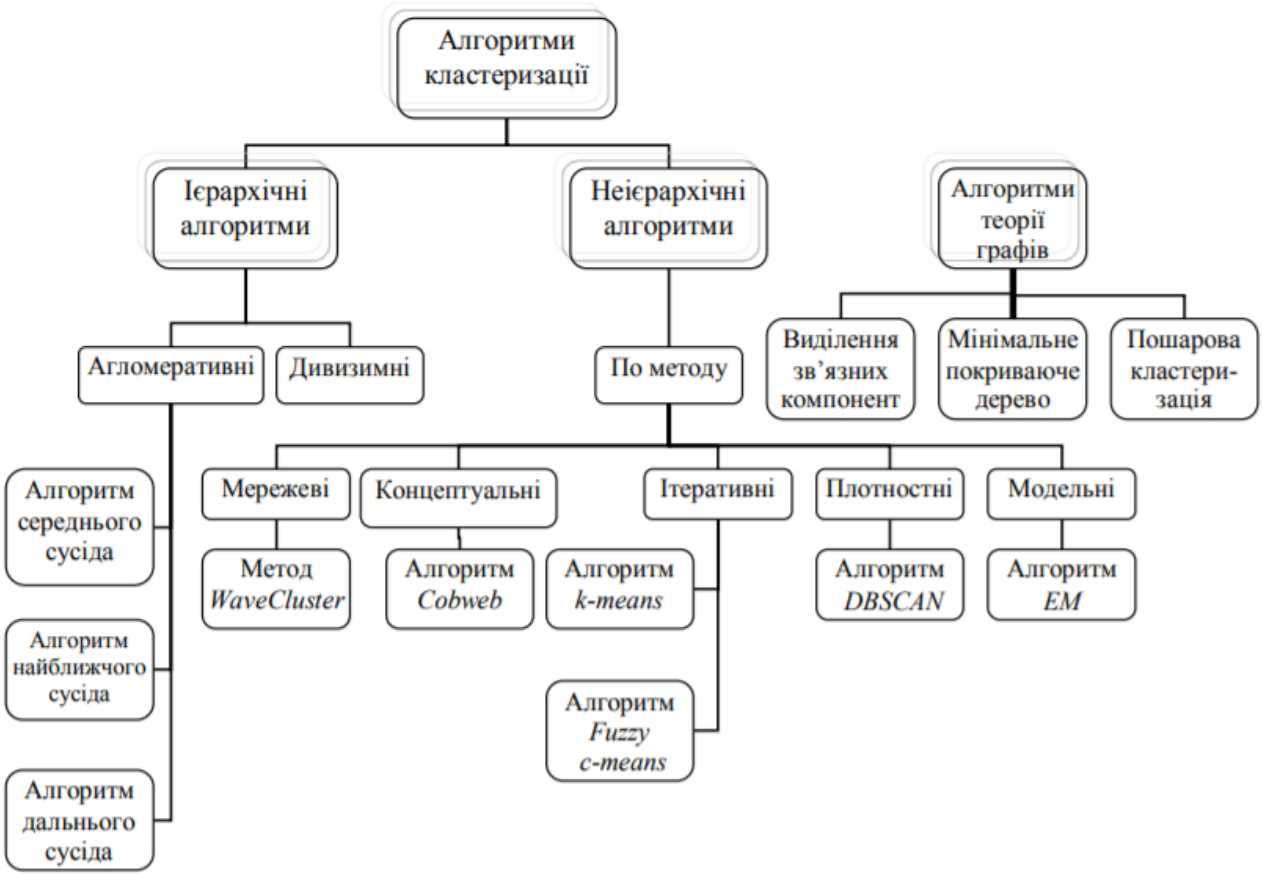


Рис. 1. Класифікація алгоритмів кластеризації

Метод k-середніх (k-means) – це спеціальний алгоритм кластеризації, котрий на вході має масив даних, який ми хочемо згрупувати у кластери, а точніше – в k кластерів.

Вхідними даними в методі k-середніх є тільки матриця. Як правило, формується вона так, щоб кожен рядок представляв окремий приклад (зразок), а кожен стовпець – окрему ознаку або, користуючись термінами з статистики, фактор. Зазвичай говорять, що є N прикладів і D ознак, так що утворюється матриця розмірності $N \times D$.

В алгоритмі методу k-середніх є два основних етапи. Спочатку вибирається k різних центрів кластерів – як правило, це просто випадкові точки в наборі даних. Потім переходять до основного циклу, який також складається з двох етапів. Перший – це вибір, до якого з кластерів належить кожна точка з X. Для цього береться кожен приклад і вибирається кластер, чий центр ближче всього. Спочатку вибираються центри випадковим чином. Другий етап – заново обчислити кожен центр кластера, ґрунтуючись на безлічі точок, які до нього приписані. Для цього беруться всі відповідні приклади і обчислюється їх середнє значення, звідси і назва методу – «метод k-середніх». Все це робиться до тих пір, поки не припиниться зміна в розподілі точок по кластерам або в координатах центрів кластерів [8].

Проте, на жаль, алгоритм k-середніх не справляється із задачею, коли об'єкт не належить жодному кластеру або належить до різних кластерів у однаковій мірі.

З цією проблемою k-means чудово справляється алгоритм c-середніх (c-means). Замість точної відповіді на запитання до якого кластеру відноситься об'єкт, алгоритм визначає ймовірність належності об'єкту до того чи іншого кластеру. Таким чином, твердження вигляду «об'єкт В належить до кластеру 1 з імовірністю 85%, до кластеру 2 – 15%» вірне і набагато зручніше.

Алгоритм c-середніх (c-means) – це модифікація методу k-means. Далі наведено кроки роботи алгоритму [7]:

1. Вибір початкового нечіткого розбиття n об'єктів на k кластерів шляхом вибору матриці належності U розміром $n \times k$.
2. Визначення значення критерію нечіткої похибки із застосуванням матриці

$$E^2(X,U) = \sum_{i=1}^{N}\sum_{k=1}^{K} U_{ik}\left\|x_i^{(k)} - c_k\right\|^2,$$

де c_k – це «центр мас» нечіткого кластера k.

3. Перегрупування (перестановка) об'єктів із метою зменшення значення критерію нечіткої помилки.

4. Перехід до пункту 2 до тих пір, поки зміни матриці U не стануть незначними.

Застосовування алгоритму с-середніх може бути недоцільним, якщо число кластерів заздалегідь невідоме або є необхідність віднесення кожного об'єкту до певного кластеру однозначно.

Для кластеризації також можуть бути застосовані такі алгоритми [6]:

1. Алгоритми, котрі засновані на теорії графів
2. Алгоритм виділення зв'язних компонент
3. Алгоритм мінімального покриваючого дерева
4. Алгоритм пошарової кластеризації

4. Обґрунтування вибору методів дослідження

Розглянуті алгоритми кластеризації достатньо прості, легко реалізуються та показують достатньо високу якість роботи. Незважаючи на це, алгоритми мають і свої недоліки. Наведемо далі переваги та недоліки методу k-середніх [5]:

Переваги:

– простота алгоритму;
– не вимагає обчислення і зберігання матриці відстаней;
– можливість паралелізації;
– лінійна просторова й тимчасова складність.

Недоліки:

– необхідно наперед знати кількість кластерів;
– алгоритм дуже чутливий до вибору початкових центрів кластерів;
– не може впоратись з завданням, коли об'єкт належить до різних кластерів у рівних степенях або не належить ніякому.

Наведемо далі переваги та недоліки методу c-середніх [5]:

Переваги:

– можливість визначення ступеня приналежності елемента до кластеру;
– нечіткість при віднесення об'єкта до кластеру дозволяє включати об'єкти, які знаходяться на границі, в кластери.

Недоліки:

– число кластерів повинно бути відоме заздалегідь;
– комплікативність роботи з об'єктами;
– шукає кластери сферичної форми;
– обчислювальна складність.

Висновки

Застосування методів кластеризаціїдо аналізу резюме та мотиваційних листів кандидатів дає можливість визначити набір професійних сфер діяльності працівників з підібраними до них найкращими резюме, тобто, отримати групування вхідних даних у вигляді резюме до певних сфер діяльності, що показує можливість опановувати певні професії певними кандидатами.

У дослідженнях методурозв'язання було розглянуто відомі рішення від професорів Сагар Море, БхамареПриянка, Малі Пуджа та КачавеКаляні, також розглянуто методи кластеризації та виявлено, що розглянуті алгоритми кластеризації достатньо прості, легко реалізуються та показують високу якість роботи. Наведено переваги та недоліки розглянутих методів кластеризації.

Список використаних джерел

1. Sudhir Bagade. Personality Evaluationand CV Analysisusing Machine Learning Algorithm [Електронний ресурс] // Режим доступу: https://www.researchgate.net/publication/335803493_Personality_Evaluation_and_CV_Analysis_using_Machine_Learning_Algorithm
2. Кластеризація [Електронний ресурс] // Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Кластеризация

УДК 004.8

СМИРНОВ Д.С.

СРАВНИТЕЛЬНЫЙ АНАЛИЗ НЕЙРОННЫХ СЕТЕЙ И ИХ АНСАМБЛЕЙ ДЛЯ ЗАДАЧИ КЛАССИФИКАЦИИ ГОРОДСКИХ ЗВУКОВ

В настоящей статье расмотрено и проанализировано практическое применение методов глубокого обучения и ансамблевых алгоритмов для задачи классификации городских звуков. Были построены и натренированы сверточные нейроные сети и ансамбль на их основе на наборе даных с городскими звуками, посчитана их точность на тестируемой выборке, и был проведен сравнительный анализ.

In this article the practical application of deep learning methods and ensemble algorithms for the task of classifying urban sounds were discussed. The convolutional neural networks and an ensemble based on them were built and trained on a set of data with city sounds, their accuracy on the test sample was calculated, and a comparative analysis was performed.

1. Вступление

Автоматическая классификация звуков окружающей среды является растущей областью исследований с многочисленными практическими применениями. Хотя в смежных областях аудио, таких как речь и музыка, ведется большой объем исследований, работа по классификации звуков окружающей среды сравнительно скудна.

Аналогичным образом, наблюдая последние достижения в области классификации изображений, где сверточные нейронные сети используются для классификации изображений с высокой точностью, возникает вопрос о применимости этих методов в других областях, таких как классификация звука.

Существует множество реальных приложений для этого исследования, таких как:

1. Контентная мультимедийная индексация и поиск
2. Помощь глухим людям в их повседневной деятельности
3. Умные дома, такие как 360-градусная безопасность

В этой работе будет продемонстрировано и проанализировано, как можно применить методы глубокого обучения для классификации звуков окружающей среды, уделяя особое внимание идентификации отдельных городских звуков. Основной идеей является сравнение точности классификации аудиофайлов отдельных моделей сверточных нейроных сетей с ансамблевыми методами на их основе.

2. Общая информация про сверточные нейроные сети и ансамблевые методы

Сверточные нейроные сети [1] - это один из самых успешных алгоритмом для обработки изображений и звуков, который используют для решения как генеративных, так и описательных задач. Сверточные нейроные сети используют систему, очень похожую на многослойный персептрон, который был разработан для снижения требований к обработке. Слои такой сети состоят из входного слоя, выходного слоя и скрытого слоя, который включает в себя несколько сверточных слоев, слои пуллинга, полносвязные слои и слои нормализации. Благодаря такой архитектуре сверточные нейроные сети являются вычислительно ефективны для задач распознавания изображений, где входные данные представляют из себя двухмерную или трехмерную матрицу.

Ансамблевое обучение [2] - это парадигма машинного обучения, в которой несколько моделей (часто называемых «слабыми учениками») обучаются для решения одной и той же проблемы и объединяются для получения лучших результатов. Основная гипотеза состоит в том, что при правильном сочетании слабых моделей мы можем получить более точные и надежные модели.

В большинстве случаев эти базовые модели работают сами по себе не так хорошо, либо потому, что они имеют высокое смещение, либо потому, что у них слишком большая дисперсия, чтобы быть устойчивыми. В итоге, идея ансамблевых методов состоит в том, чтобы попытаться уменьшить смещение и дисперсию таких

слабых учеников, комбинируя несколько из них вместе, чтобы создать сильного ученика (или модель ансамбля), который достигает лучших результатов.

3. Краткое описание проблемы и данных

Проблема классификации городских звуков состоит в способности моделей глубокого обучения определять, содержит ли звуковой образец в машиночитаемом формате (например, в формате WAV) продолжительностью в несколько секунд один из целевых городских звуков. Для этой задачи был найден набор данных под названием Urbansound8K, который содержит 8732 звуковых фрагментов (продолжительностью меньше или равно 4 секундам) городских звуков из 10 классов, а именно:

1. Кондиционер
2. Звук автомобиля
3. Играющие дети
4. Лай собаки
5. Бурение
6. Двигатель на холостом ходу
7. Выстрел пистолета
8. Отбойный молоток
9. Сирена
10. Уличная музыка

Эти звуковые фрагменты представляют собой цифровые аудиофайлы в формате .wav. Звуковые волны оцифровываются путем их дискретизации с дискретными интервалами, известными как частота дискретизации (как правило, 44,1 кГц для звука с качеством CD означает, что выборки берутся 44 100 раз в секунду).

Для того, чтобы использовать те же методы, которые используются для классификации изображений с высокой точностью, все аудиофайлы были трансформированы в визуальное их представление с помощью спектрограмм. Спектрограммы - полезный метод для визуализации спектра частот звука и того, как они меняются в течение очень короткого периода времени. Для этой задачи был использована техника, известная как Mel-Frequency Cepstral Coefficients (MFCC) [3].

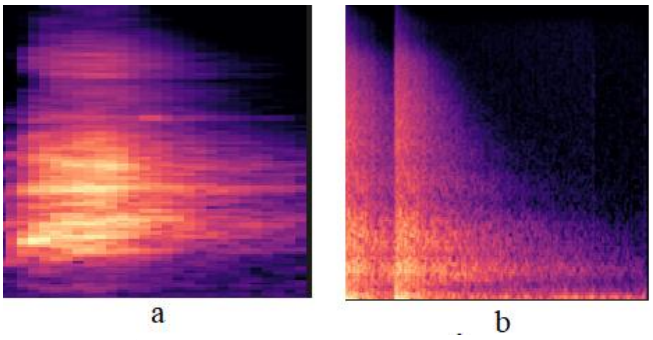


Рис. 1. Примеры спектрограмм: a – лай собаки, b – выстрел пистолета

4. Описание архитектур нейронных сетей

Для сравнительного анализа была построена сверточная нейронная сеть, которая состоит из 3 сверточных слоев с количеством и размерностью ядр соответственно (64, 3, 3), (128, 3, 3) та (64, 3, 3) и из 3 полносвязных слоя с количеством нейронов соответственно 1024, 512 и 10, которые идут после сверточных. После каждого сверточного слоя также присутствует операция подвыборки (max-pooling операция) с ядром размерностью (2, 2). Последний слой нейронной сети представляет из себя слой softmax, который дает вероятности принадлежности к каждому из известных 10 классов для входящего изображения (Рис 2). Стоит также заметить, что в каждом сверточном и в первом полносвязном слоях добавлена функция dropout с параметром 0.25. Она применяется для того, чтоб модель не переобучалась и ее смысл заключается в том, что каждый нейрон в слое с вероятностью 0.25 не участвует во время тренировки на заданной эпохе, то есть обнуляется.

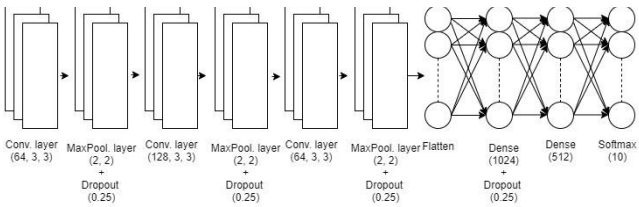


Рис. 2. Архитектура нейронной сети

В качестве ансамбля было взято 5 нейронных сетей, архитектуры каждой из которых аналогичны описаных ранее. Тренируются они отдельно одна от другой. Разница с обычной нейронной сетью заключается в том, что они по разному выдают результаты. Одна нейронная сеть присваивает класс входящему изображению исходя из наибольшей вероятности, которую

она получила в слое softmax. В ансамбле же после того как каждая нейронная сеть дает вероятности принадлежности к каждому классу, эти вероятности суммируются. То есть, например, первая сеть дала вероятность 0.25, что входящее изображение принадлежит к первому классу, и 0.75 - что ко второму. Вторая нейронная сеть в свою очередь выдала 0.5 и 0.5 вероятности соответственно. Просуммировав получаем, что ансамбль дает вероятность в 1.25, что входящее изображение принадлежит к первому классу, и 0.75 - ко второму. Стоить заметить, что результирующие вероятности в данному случае могут быть больше 1, так как они не нормируются. В результате ансамбль отнесет изображение в тот класс, который имеет наибольшую таким образом посчитанную суму (Рис. 4).

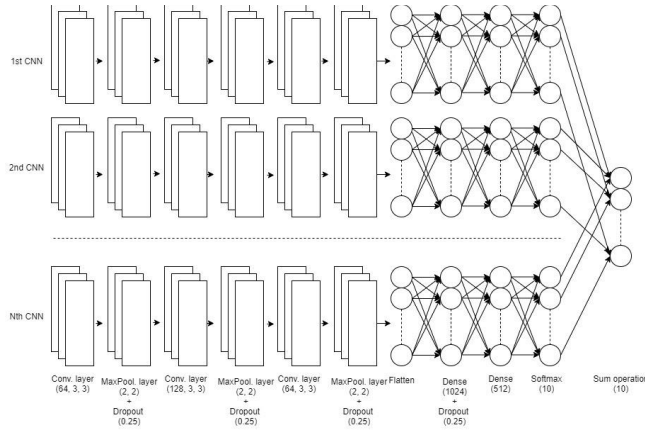


Рис. 3. Архитектура ансамбля нейронных сетей

5. Сравнение метрик

Нейронные сети тренировалась на протяжение 20 епох. В качестве сравнительной метрики [4] была выбрана классическая точность:

$$f = \frac{n\_true}{n}$$

Где n_true = количество правильно классифицированных аудиофайлов и n – общее количество аудиофайлов. Изначальный датасет был разбит на тренировочную и тестовую выборку случайным образом (80% датасета тренировочная, 20% - тестовая). Для одной нейронной сети был использован весь тренировочный датасет, который предварительно был разбит на тренировочный и валидационный (75% - тренировочный, 25% - валидационный). Для нейронных сетей, которые входили в ансамбль, был использован принцип бутстрап агрегации. То есть каждая нейронная сеть случайным образом из общем тренировочной выборки выбирает себе подвыборку (в даном исследовании это 50% от всей тренировочной выборки). Дальше делит ее на тренировочную и валидационную выборки (75%:25%) и обучается.

После того как нейронные сети натренировались, был получена их точность на тестовой выборке. Одна нейронная сеть получила результат приблизительно в 88%. Ансамбль с нейронных сетей показал точность приблизительно в 95%. Следовательно, разница точности этих двух техник составляет 7%.

Заключение

Проанализировав результаты, которые дали отдельная нейронная сеть и ансамбль на их основе, был сделан вывод, что ансамбль нейронных сетей может показать результативность больше в среднем на 7% от классической независимой модели в задаче классификации городских звуков. Это означает, что ансамбль моделей действительно может частично решать проблему большого значения дисперсии во время работы моделей машинного обучения, особенно в глубоких нейронных сетях.

Список литературы

1. An introduction to Convolutional Neural Networks [Електронний ресурс] // researchgate.net. – 2015. – Режим доступу до ресурсу: https://www.researchgate.net/.
2. Deep Neural Network Ensembles [Електронний ресурс] // arXiv.org. – 2019. – Режим доступу до ресурсу: https://arxiv.org/.
3. Mel Frequency Cepstral Coefficients for Music Modeling [Електронний ресурс] // ismir2000 – Режим доступу до ресурсу: ismir2000.ismir.ne.
4. A Review on Evaluation Metrics for Data Classification Evaluations [Електронний ресурс] // researchgate. – 2015. – Режим доступу до ресурсу: https://www.researchgate.net/

УДК 004.91

СЕМЧЕНКО А.О.
КОВТУНЕЦЬ О.В.

ПОБУДОВА І ПРИКЛАДНЕ ЗАСТОСУВАННЯ АПАРАТНИХ КРИПТОГРАФІЧНИХ КЛЮЧІВ

Розглянуто підходи до вирішення задачі безпечного зберігання ключів доступу/авторизації для користувачів інформаційних систем. Особливістю задачі є практичне рішення з низькою вартістю реалізації.

КЛЮЧОВІ СЛОВА: апаратне шифрування, криптографічний ключ, криптографічний токен.

Approaches to solving the problem of secure storage of access/authorization keys for users of information systems are considered. The peculiarity of the problem is a practical solution with low cost of implementation.

KEYWORDS: hardware encryption, cryptographic key, cryptographic token.

1. Вступ

Бурхливий розвиток інформатики та, як наслідок, інформаційних систем, призвів до швидкої цифровізації суспільства. Інформаційні системи активно впроваджуються в великій кількості суспільних процесів:

- електронне голосування
- електронний уряд
- Онлайн-банкінг (традиційний, криптовалюта)
- Отримання державних послуг онлайн

Разом з тим гостро постало питання безпеки інформаційних систем, забезпечення механізмів авторизації і аутентифікації. Очевидно, що з розвитком мережевих технологій історично перша схема аутентифікації на основі паролів втрачає свою актуальність. Основним недоліком пароля є те, що його можна легко викрасти, як через вразливість програмного забезпечення, так і з необережності користувачів. Одночасно виникає проблема безпечного зберігання криптографічних ключів таким чином, щоб їх не могли викрасти. Вищезазначені проблеми можна подолати, використовуючи інструменти асиметричної криптографії та апаратних криптографічних модулів.

Криптографія дає можливість приховувати те, що ми хочемо відправити, переконатися в тому, що ми спілкуємося з чітко встановленою особою, тощо. Зазвичай для того, щоб все це добре працювало, від нас потрібне єдине - тримати в таємниці наші ключі шифрування. Здається, це звучить просто, чи не так? Розберемо способи приховування ключів шифрування:

Перший спосіб – збереження у файлі на робочому столі - старий і перевірений роками спосіб щось зберегти. Проблема в тому, що крім самого користувача, ще багато інших програм мають доступ до цих файлів. На жаль, частина з них може робити не тільки те, для чого безпосередньо призначені, або і взагалі не те, під що вони мімікрують.

Другий спосіб – за допомогою менеджера паролів - по суті таке ж зберігання в файлі, просто тепер цей файл буде зашифрований, і для доступу до нього потрібно знати пароль. Але оскільки менеджер паролів запускається на комп'ютері, то не зашифровані ключі потрапляють в оперативну пам'ять, звідки їх можна вкрасти через вразливості операційної системи.

Третій спосіб – фіксація на паперових носіях («записати на папірець») - дивно, але цей спосіб виглядає безпечнішим за попередні. Ключі не зберігаються на комп'ютері, який потенційно містить небезпечні програми. Кожен раз, коли потрібно використати ключ - його просто вводять з клавіатури. Однак, якщо ваші ключі досить довгі (а ключі повинні бути довгими) - це може стати проблемою використання. Ще однією проблемою можуть стати кейлогери, які з легкістю можуть викрасти як завгодно довгий ключ.

Отже, основна проблема полягає в тому, що ключі або безпосередньо зберігаються на

комп'ютері, або вводяться в комп’ютер за допомогою клавіатури, диска тощо. Як тоді комп'ютер буде шифрувати дані, якщо він не матиме доступу до ключа? Правильна відповідь - ніяк.

Насправді рішення придумано давно. Головна ідея в тому, щоб підключити до комп'ютера спеціальний пристрій, який і буде шифрувати дані, а комп'ютер буде тільки відправляти дані і отримувати результат. Таким способом працюють, наприклад, смарт карти.

2. Постановка задачі

Підключимо до комп'ютера спеціальний пристрій, який буде шифрувати дані, а комп'ютер буде тільки відправляти дані і отримувати результат.

Пристрій буде зібрано на недорогому і досить популярному мікроконтролері серії STM32 з використанням програматора ST-Link v2. Звісно, можна самостійно виготовити друковану плату, однак розумніше використати готове рішення.

Зовні програматор нагадує USB-флеш-накопичувач, що спрощує нашу задачу. Його корпус виготовлений з алюмінію – таким чином ми отримуємо одночасно захист від механічних пошкоджень. Для нашої задачі потребуємо дві одиниці мікроконтролера – одна призначена для переробки в ключ, а друга для завантаження прошивки на першу.

3. Алгоритм реалізації

3.1 Способи зберігання

Способи зберігання криптографічних ключів, що були розглянуті вище можна поділити на 2 категорії: зберігання у незахищеному вигляді та зберігання у зашифрованому вигляді. В залежності від типу носія інформації такі способи поділяються на цифрові та аналогові.

3.2.Компіляція прошивки

Тексти програм прошивки можна знайти в репозиторії проекту GNUK. Процес скачування і переходу в папку проекту опускається.

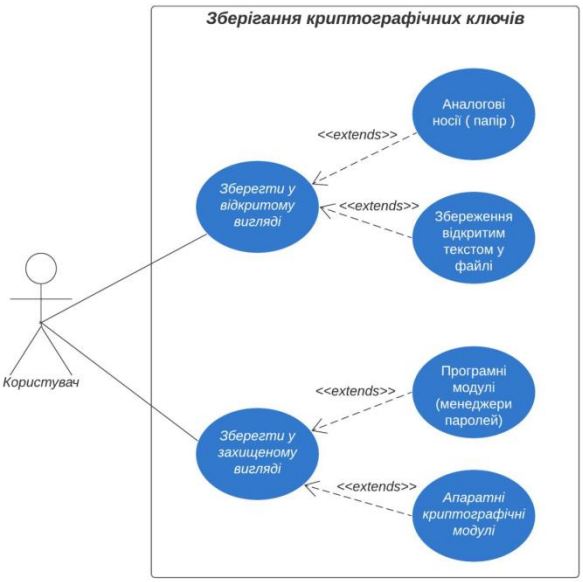


Рис.1 UML use-case діаграма

Для компіляції прошивки необхідно встановити кілька програмних пакетів:

- arm-none-eabi-gcc
- arm-none-eabi-newlib
- openocd

Наприклад, якщо використовувати пакетний менеджер pacman, процес виглядатиме так:

```
$ sudo pacman -S arm-none-eabi-gcc
$ sudo pacman -S arm-none-eabi-newlib
$ sudo pacman -S openocd
```

Для подальшої роботи необхідно запустити конфігураційний скрипт:

```
$ ./configure --vidpid=234b:0000
```

Компіляція прошивки виконується утилітою make:

```
$ make
```

Після компіляції в папці проектц появиться папка build, а в ній файл gnuk.elf — скомпільована прошивка.

3.3.Завантаження прошивки на пристрій

Тепер, коли є готовий файл з прошивкою, потрібно завантажити її на пристрій. Для цього спершу необхідно попрацювати паяльником. Знімаємо корпус із одного із програматорів – він і буде донором. Результат подано на рис. 2.

Зверніть увагу на 4 контакти на платі. До них потрібно припаяти проводи, які будуть підключатися до другого екземпляра програматора. Проводи потрібно підключити відповідно до маркування на платі і контактах програматора.

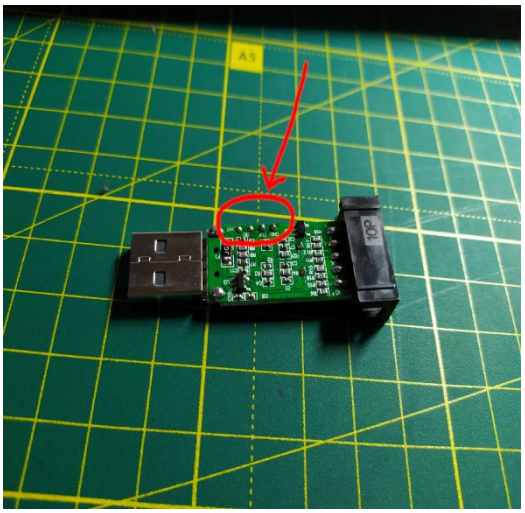


Рис. 2. Програматор без корпусу.

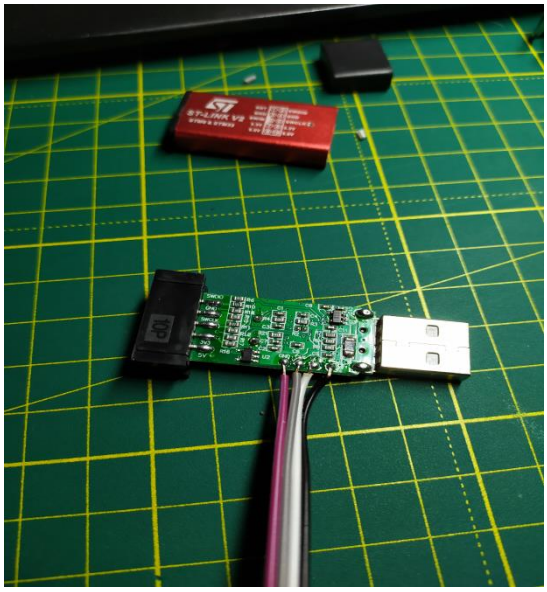


Рис. 3. Програматор із підготовленими проводами.

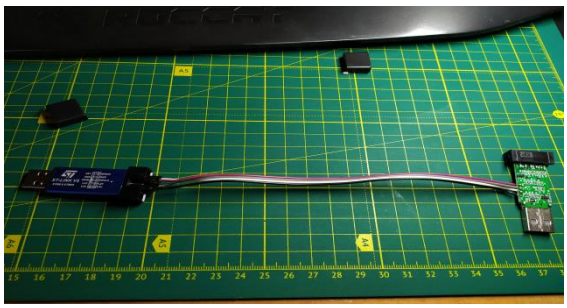


Рис. 4. Підготовлені до завантаження програматори.

Підключимо програматор до комп'ютера, відкриємо термінал і з папки з прошивкою (gnuk.elf) завантажимо прошивку:

```
$ openocd -f interface/stlink-v2.cfg -f target/stm32f1x.cfg -c 'program build/gnuk.elf verify reset exit'
```

Прошивка завантажена на пристрій. Залишився останній крок — заборонити зчитування пам’яті мікроконтролера – щоб перешкодити людині, якій до рук потрапив ключ, витягнути з нього секретну інформацію:

```
openocd -f interface/stlink-v2.cfg -f target/stm32f1x.cfg -c init -c "reset halt" -c "stm32f1x lock 0" -c reset -c exit
```

Тепер пристрій готовий до використання.

3.4 Функції токена

Криптографічний токен має широкий спектр застосування завдяки значущості своїх функцій. Оскільки операції шифрування та підпису даних є основними в криптографії, їх можливо застосувати для вирішення широкого кола прикладних задач.

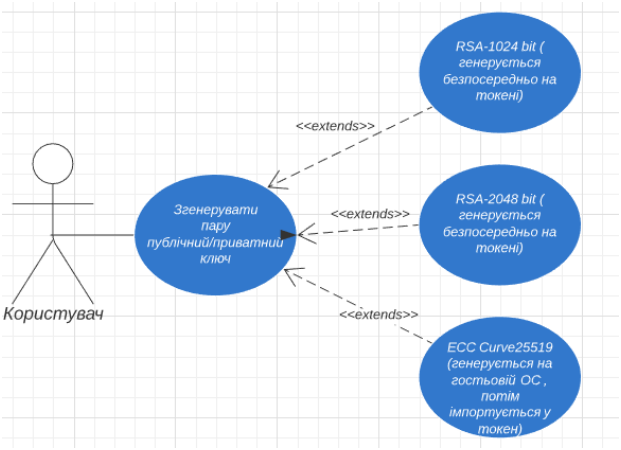


Рис. 5 Use-case діаграма генерації ключів

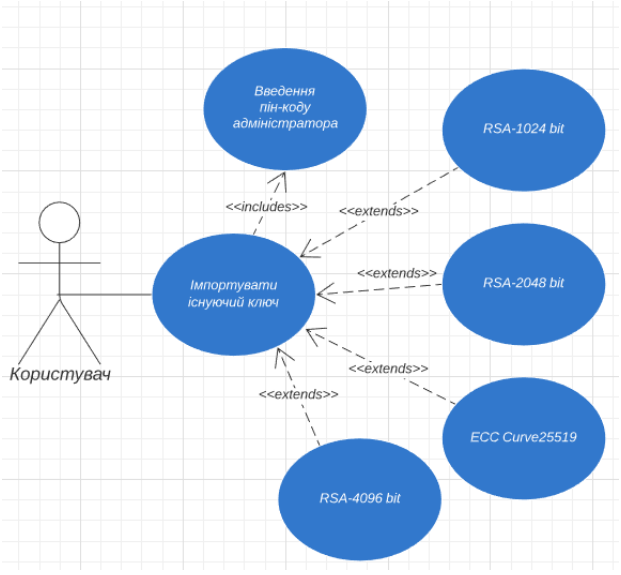


Рис. 6 Use-case діаграма імпорту ключів

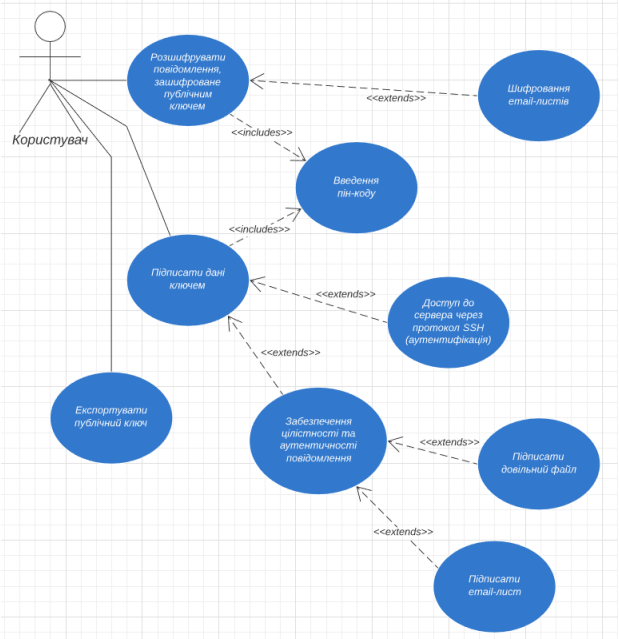


Рис. 7 Use-case діаграма використання токена

3.5.Підготовка ключів у різних сферах застосування

Зібраний пристрій є емулятором OpenPGP смарт-карти. Така карта може зберігати GPG чи SSH ключі. Такий ключ можна використовувати наступним чином:

- Підписувати git коміти
- Зберігати SSH ключі
- Шифрувати і підписувати електронну пошту згідно із стандартом S/MIME
- Входити в ОС із аутентифікацією користувача на основі x509 сертифікатів

Генерування нового ssh-ключа

Важливо розуміти, що ключ генерується на самому пристрої, а не на комп'ютері, тобто такий ключ взагалі ніколи не полишає пристрій.

Запускаємо GNU Privacy Guard (GnuPG або GPG) — відкритий програмний засіб для шифрування та цифрового підписування даних:

$ gpg --card-edit

Відкривається інтерактивний режим gpg, включаємо режим адміністратора відповідною командою:

gpg/card> admin

Запускаємо команду генерування нового ключа:

gpg/card> generate

Під час стандартної процедури генерування ключа gpg буде запропоновано зберегти бекап ключа на диск. Питання безпеки резервних копій ключів лишається за нашими послідовниками.

Надалі в процесі використання будуть потрібні пін-коди. Стандартно в прошивці зафіксовано наступні варіанти пін-кодів (пізніше їх потрібно змінити):

CARD PIN — 123456,

ADMIN PIN — 12345678.

Експорт публічного ключа в ssh-форматі

Отримуємо потрібний публічний ключ з токена, щоб зберегти його на віддаленому сервері:

$ pkcs15-tool --list-keys

Приклад виводу команди:

Using reader with a card: Free Software Initiative of Japan Gnuk (FSIJ-1.2.15-87144751) 00 00
Private RSA Key [Signature key]
Object Flags : [0x03], private, modifiable
Usage : [0x20C], sign, signRecover, nonRepudiation
Access Flags : [0x1D], sensitive, alwaysSensitive, neverExtract, local
ModLength : 2048
Key ref : 0 (0x00)
Native : yes
Auth ID : 01
ID : 01
MD:guid : f3de5f55-d100-4973-d572-40d67e20f033

Тут необхідно звернути увагу на ID ключа, в нашому випадку 01. Тепер експортуємо публічний ключ:

$ pkcs15-tool --read-public-key 01

Копіюємо публічний ключ у файл pub.key:

-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
MIIBCgKCAQEAyzHQIWEApliWYaf0T8jb

Vh2nc5+LklKXeuJFTN3BW2VqdrTw1rpKXi
ANWpi+qbtZhZ2nP3CJX6qoGobXyCOd

/iAiygFlyW4BwTQpnAm81IE9lPzfasOK7SBu
KJ+ZbB4WpuYJRozgtt/gpWzmnWnW

84/CU9Lqbhz95v/C/DImSf6LiwVdmiEj4CUNI
nl5pY4trguDsSfkw1u8gGqSPEsD

ZXtlVRx8iBGi0JR02g9KTL4dDGocUtcTK8W
0eY+BDbQSXfTGCy93v8sEyhdQjHs8

oDiwkvFQ86gYqwL5DJ7U/rFSO3A5X6zmkF
FV8nJZjxB2qfE5aommtXxow4iPml3x

YwIDAQAB

-----END PUBLIC KEY-----

Конвертуємо ключ в ssh-rsa формат:

```
$ ssh-keygen -f pub.key -i -mPKCS8
```

Отримуємо публічний ключ в потрібному форматі:

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABA
QDLMdAhYQCmWJZhp/RPyNtWHadzn4uSU
pd64kVM3cFbZWp2tPDWukpeIA1amL6pu1m
Fnac/cIlfqqgahtfII53+ICLKAWXJbgHBNCmc
CbzUgT2U/N9qw4rtIG4on5lsHham5glGjOC23
+ClbOadadbzj8JT0upuHP3m/8L8MiZJ/ouLBV
2aISPgJQ0ieXmlji2uC4OxJ+TDW7yAapI8Sw
Nle2VVHHyIEaLQlHTaD0pMvh0MahxS1xMr
xbR5j4ENtBJd9MYLL3e/ywTKF1CMezygOL
CS8VDzqBirAvkMntT+sVI7cDlfrOaQUVXycl
mPEHap8Tlqiaa1fGjDiI+aXfFj

Стандартним чином налаштовуємо ssh для входу із заданим ключем — додаємо ключ у файл ~/.ssh/authorized_keys на віддаленому сервері.

Конфігурування ssh

Тепер щоб зайти на сервер, використовуючи токен, треба викликати ssh з ключем -I і передати шлях до драйвера токена. Токен сумісний із одним з стандартних драйверів, тому рекомендується використовувати його

```
$ ssh -I /usr/lib/opensc-pkcs11.so username@remotehost
```

Проте зручніше скористатися config файлом (~/.ssh/config). Додаємо в нього наступні рядки:

```
Host digitalOceanServer
HostName 192.168.0.1
User root
PKCS11Provider /usr/lib/opensc-pkcs11.so
```

Тепер виклик ssh спрощується:

```
$ ssh digitalOceanServer
```

Висновки

Зібраний пристрій можна розглядати як недорогий ключ початкового рівня. Використання такого пристрою суттєво піднімає рівень безпеки користувача.

Треба розуміти, що ризики компрометації ключа зберігаються. Розглянемо випадок фізичної компрометації ключа. Сам ключ захищений пін-кодом, і після кількох невдалих спроб ключ блокується. Пам’ять самого пристрою захищена від зчитування, тому безпосередньо прочитати збережені у ньому ключі неможливо.

Існують окремі види атак на сам чіп (наприклад препарування і підключення до нього мікроелектродів для вимірювання напруги в будь-якому місці чіпа, в тому числі і пам’яті), але вони досить затратні і складні практично. Отже, оскільки вартість реалізації відносно вартості компрометації достатньо низька, вважаємо, що поставлену задачу виконано.

Список літератури

1. Панасенко С. Алгоритмы шифрования. 2009р.
2. Панасенко С., Ракитин В. (2002 - № 8). Мир ПК.Аппаратные шифраторы. 2002, стор. 77-83.
3. Gnuk implementation of USB cryptographic token for GNU Privacy Guard. URL: http://fsij.org/doc-gnuk/ (дата звернення 20.04.2020).
4. The PKCS #11 URI Scheme. URL: https://tools.ietf.org/html/rfc7512 (дата звернення 20.04.2020).
5. Cryptographic Token Interface Standard. URL: https://cryptsoft.com/pkcs11doc/ (дата звернення 20.04.2020).
6. OASIS PKCS 11 TC Public Documents. URL: https://www.oasis-open.org/committees/documents.php?wg_abbrev=pkcs11 (дата звернення 20.04.2020).
7. OpenPGP. URL: https://www.openpgp.org/ (дата звернення 20.04.2020).
8. GNU Privacy Guard. URL: http://www.gnupg.org/ (дата звернення 20.04.2020).

УДК 004.45

ВІХЛЯЄВ.О.
ХАЛУС О.А.

ТРАНСЛЯЦІЯ UIKIT, APPKIT КОРИСТУВАЦЬКОГО ІНТЕРФЕЙСУ ІЗ XIB У SWIFTUI

В даній роботі представлений огляд такого поняття як рансляція UIKit, AppKit користувацького інтерфейсу із XIB у SwiftUI.

КЛЮЧОВІ СЛОВА: SwiftUI, файли, інтерфейс.

This paper provides an overview of the concept of UIKit, AppKit UI rankings from XIB to SwiftUI.

KEYWORDS:SwiftUI, file, interface.

Вступ

Робота над користувацькими інтерфейсами є дуже важливою частиною розробки сучасних додатків для Apple-систем. Із самого часу появи цих систем і до WWDC2019 для цього використовуютьсяспеціальніфреймворки (UIKit для iOS,Appkit для macOS)

Проблеми UIKit та AppKitUI

Попри багато відмінностей і для iOS і для macOS користувацький інтерфейс будувався у XIB (Storyboard) файлах. Вони являють собою великі (2-3+ тисяч рядків) XML файлискладної структури, що генерується та редагується за допомогою XcodeInterface Builder. Редагувати XIB-и вручну можливо але це є досить складним процесом так як структура дійстно складна, досить легко зробити помилку і навідміну від Swift коду, для XMLXcode не вкаже на помилку і не допоможе її виправити. Більш того при роботі із SVC часто виникають конфлікти.

Перехід на SwiftUI

Із запровадженням SwiftUI на WWDC 2019 зявився новий підхід до побудови користувацького інтерфейсу для Apple систем. SwiftUI - це інстумент який дозволяє будувати застосунки у декларативному стилі, при цьому має повну підтримку звичного імперативного підходу там де це може бути потрібно. Фактично SwiftUI - це DSL на базі Swift. Для того аби від був максимально зручним і інтуєтивно зрозумілим до Swift 5.1 додали:

- *Opaque return types*
- *Omitted return keywords*
- *Function builders*
- *Property wrappers*

Багато нових проектів вже пишуть використовуючи нову технологію, проте для старих проектів перехід може бути доволі складним і довгим. Необхідно буде замінити усі XIB та Storyboard файли на відповідні SwiftUI view і інтегрувати стару логіку додатку. І якщо адаптацію логіки складно якось автоматизуватито трансляцію XIB-ів у SwiftUI структури можна сильно пришвидчити побудуваши конвертер що на вхід приймає .xib / .storyboard файл із старим UI а на виході видає .swift файл із відповідними SwiftUI view.

Опис алгоритму

XIB файли мають деревовидну структуру оскільки представляють відповідно дерево view.Деякі view (nodes)можуть мати child views, деякі (leaves) - ні. Необхідно серілізувати XIB файл у відповідну SwiftUI структуру, зберігаючи такі дані:

- *User-Defined Runtime Attributes*
- *Параметри View (кольори, стилі, шрифти, tintColor)*
- *Subviews*

Для кожної структури даних потрібно визначити stringRepresentation (SwiftUI код що відповідає даній View) - це необхідно для подальшої кодогенерації і збору структури у вихідний код. Елементів наскправді досить багато, тож далі наведено лише невелику частину (для iOSXIB-ів):

1. UILabel

2. UIButton
3. UISegmentedControl
4. UITextField
5. UISlider
6. UISwitch
7. UIActivityIndicatorView

…

Деякі UIKit, AppKit view не мають SwiftUI аналогів. Для таких елементів слід підготувати окремі SwiftUI view структури що обертають собою UIKit / AppKit елементи. Для підтримки крослатформості визначимо окремо iOS та macOS елементи та включатимемо у вихідний код обидва, перевіряючи платформу за домопогою конструкції:

```
#if os(iOS)
 // iOS implementation
 #else
 // macOS implementation
 #endif
```

Генерація SwiftUI коду буде виконуватись шляхом отримання stringRepresentation у Root View, обрахування якої призведе до виклику stringRepresentation усіх вкладених View і їх subviews відповідно. Таким чином отримаємо SwiftUI файл що дублює XIB файл.

Важливі нюанси

XIB файли можуть мати в собі не тільки єдину Root View тож такі випадки також необхідно підтримати алгоритмом.

Storyboard файли можуть мати багато сцен кожна з яких є аналогом окремого XIB файлу - для таких випадків на виході отримаємо набір SwiftUI файлів для кожної сцени.

Окрім трансляції структури вкладеності елементів необхідно врахувати що у XIB-ах на усі view елементи можуть накладатися NSLayoutConstraint-итаких типів:

1. Constraint on element width & height, aspect ratio
2. Constraints on child - parent
3. Constraints between siblings

Для того щоб зменшити кількість додаткової роботи після конвертації слід врахувати і ці обмеження.

Висновок

Маючи рішення що дозволяє автоматизувати трансляцію XIB файлів у SwiftUI код, розробники зможуть пришвидшити перехід до нового підходу побудови UI додатків.Окрім того, SwiftUI код що генерується транслятором є кросплатформовим і такий перехід є також кроком у сторону перевикористання коду і підтримки усього сімейства операційних систем Apple для додатків що існували лише на iOS.

Перелік посилань

1. Apple Documentation - SwiftUI. URL: https://developer.apple.com/documentation/swiftui
2. Apple WWDC 2019 - Introducing SwiftUI. URL: https://developer.apple.com/videos/play/wwdc2019/204/

УДК 004.93(015.7)

ШВЕЦЬ Д. Ю.
ПРОСКУРА С.Л.

АЛГОРИТМИСТВОРЕННЯ ДОПОВНЕНОЇ РЕАЛЬНОСТІ

Розглянуто алгоритми створення доповненої реальності.Побудована математична модель задачі ідентифікації зображень та побудови доповненої реальності. Описаний метод, що дозволяє накладати цифрову інформацію на реальний світ.

Ключові слова: доповнена реальність, 3D-моделі, комп’ютерний зір, SUSAN.

The algorithm for creating augmented reality is considered.A mathematical model of the problem of image identification and augmented reality construction is constructed.The method that allows you to overlay digital information on the real world is described.

Key words: augmented reality, 3D models,computer vision, SUSAN.

Вступ. В останній час активно розвивається така сфера ІТ як доповнена та віртуальна реальності. Відмінність між цими двома поняттями у наступному: віртуальна реальність – це вигаданий, технічно створений світ, який передається людині, у доповненій реальності об'єкти накладаються на реальне оточення. Отже, доповнена реальність проектує будь-яку інформацію на екран пристрою. Дана технологія може бути реалізована за допомогою мобільних додатків, окулярів доповненої реальності, проекційних пристроїв.

Для реалізації доповненої реальності у телефоні необхідно розуміти, що додаток повинен через камеру зчитувати навколишнє середовище, визначаючи розміри та місцезнаходження об'єктів, а також своє положення у цьому просторі. Далі сервіс розміщує віртуальний об'єкт відносно усіх об'єктів у реальному світі, використовуючи координати речей задля гармонійного знаходження його на екрані смартфона.

Постановка проблеми.При розвитку технології доповненої реальності (AR) все більше компаній прагнуть застосувати ці технологіїу своїх цілях. Причинами того є збільшення інтересу користувачів до товарів та, відповідно, кількості продажів. Але, наразі, компаніям, для впровадження доповненої реальності (AR)для своїхпродуктів,необхідно наймати команду розробників, які будуть реалізовувати, впроваджувати та підтримувати цю технологію на певному виробництві, що є не завжди вигідним зі сторони затрат замовника. Тому постає проблема - як зменшити вартість розробки AR для товарів та розробити такий програмний продукт, застосування якого дозволило автоматизувати процес впровадження доповненої реальності на підприємства та у життя людей.

Формулювання цілей статті.Метою цієї роботи є автоматизація процесу впровадження доповненої реальності на підприємствах та у життя людей, мінімізація вартості цієї дії та її спрощення за допомогою мобільного застосування, цілями якого є:

- полегшення створення 3D-моделей;
- оптимізація етапу накладання 3D-моделей на об'єкти;
- полегшення управління доповненою інформацією для усіх товарів.

Для досягнення поставлених цілей, пропонується вирішитинаступнізадачі: створення 3D-моделей; аналіз реального світу через камеру смартфона;накладання моделі на певний об'єкт з навколишнього середовища; вивід товару з доповненою реальністю на ринок.

Постановка задачі. Технології доповненої та віртуальної реальності все більше входять у нашу повсякденність та стають звичними для всіх. Але впровадження та сприйняття їхвсе ще не є зручними та легкими для розуміння людей. На цьому етапі постає задача – розробити систему, яка б посприяла збільшенню кількості використання технологій доповненої реальності як у промисловості, так і у звичайному житті.

Зробивши аналіз доповненої реальності зараз та його тенденції, маємо висновок – найкраще розробляти систему для вирішення описаної задачі у вигляді мобільного додатку. Виділимо основні дії, які дана система повинна дозволяти робити своїм користувачам: вибір готових моделей AR або створення своїх власних, накладання моделей на певні об'єкти або на будь-які предмети на зображення з камери, сканування товарів для отримання інформації у вигляді AR. Також, для впровадження даної системи у підприємства – необхідно мати можливість створити користувача з певним типом, який буде мати змогу накладати доповнену реальність на товари компанії та відправляти їх у мережу. Це робиться для того, аби клієнт, маючи товар компанії з певною міткою для розуміння, що його можна сканувати, міг відкрити додаток та навести камеру на предмет і отримати певну інформацію. Для побудови доповненої реальності на відео потоці буде відбуватись пошук маркеру та відображення 3D-моделі на ньому.

Опис ідентифікації зображень. Визначимо яким чином робиться пошук певних предметів на зображенні. Для цього необхідно звернутись до такого поняття як теорія комп'ютерного зору та його алгоритмів. Дана теорія лежить в основі технології доповненої реальності, адже завдяки неї ми можемо проводити аналіз зображень з камери. Алгоритми комп'ютерного зору дозволяють знаходити

певні об'єкти на зображенні в реальному часі, обробляти та використовувати у подальшому.

Після отримання зображення з камери – постає задача визначити з нього ті предмети, які будуть використовуватися для відображення і взаємодії з доповненою реальністю. Для цього на зображенні виділяються певні особливі точки. Кожна така точка m має округ $o(m)$, який можна відрізнити від округу іншої точки $o(n)$ в деякому іншому округу особливої точки $o_m(m)$.Округ – прямокутник, розміром, наприклад, 5×5. Для визначення описаних точок вводяться такі поняття: детектор та дескриптор. Детектор – метод, який дозволяє дістати усі особливі точки з потоку зображення. Дескриптор – метод, який дозволяє ідентифікувати усі особливі точки з-поміж інших.

Виділяють різні алгоритми для реалізації описаної технології. Деякі з них: Beaudet, Moravec, Förstner, Plessey, Deriche, SUSAN, FAST. Для прикладу опишемо один з них – SUSAN:

1. Центр кругової маски зміщуємо у ядро.
2. Обрахувати, скільки пікселей мають схожу інтенсивність з ядром. Для цього використаємо формулу:
$$c(\vec{r}, \overrightarrow{r_0}) = e^{-(\frac{I(\vec{r})-(\overrightarrow{r_0})}{t})^6}$$
, де $\vec{r}$ – точка усередині маски, $\overrightarrow{r_0}$ – центр ядру, $I(\vec{r})$ – інтенсивність точки, t – поріг інтенсивності.
3. Обрахувати розмір ділянок для отримання зображення з кутами. Використовується формула:
$$R(\overrightarrow{r_0}) = \begin{cases} g - n(\overrightarrow{r_0}), якщо\ n(\overrightarrow{r_0}) < g, \\ 0 \end{cases}$$
, де g – геометричний поріг, $n(\overrightarrow{r_0}) = \sum_{\vec{r}} c(\vec{r}, \overrightarrow{r_0})$.
4. Визначити центри ділянок та їх близькість одне до одного.
5. Обрати особливі точки, визначивши локальні максимуми функції.

Опис побудови доповненої реальності. Після визначення предметів на зображенні камери постає задача побудови 3D простору на ньому. Для цього нам необхідні 2 матриці: внутрішня та зовнішня.

Внутрішня матриця приймає наступні параметри камери: фокальна відстань за двома осями та координат центру фокусу.

$$\begin{bmatrix} f_x & 0\, c_x \\ 0 & f_y c_y \\ 0 & 01 \end{bmatrix}$$

Зовнішня матриця відповідає за модель, яка буде відображуватись, та дає змогу задавати положення даного об'єкту. Елементи матриці відповідають за розтягування, поворот та перенесення.

$$\begin{vmatrix} r_{11} & r_{12} r_{13} & t_1 \\ r_{21} & r_{22} r_{23} & t_2 \\ r_{31} & r_{32} r_{33} & t_3 \end{vmatrix}$$

Розтягування - r_{11},r_{22},r_{33};поворот - r_{12}, r_{13}, r_{21}, r_{23}, r_{31}, r_{32}; перенесення - t_1, t_2, t_3.

Після знаходження значень обох матриць, використовуючи можливості OpenGL, можна відобразити необхідну модель поверх зображення з камери.OpenGL – інтерфейс, який дозволяє малювати тривимірні сцени та об'єкти.

Також при побудові доповненої реальності необхідно пам'ятати, що зображення чи відео потік, які ми отримуємо з камери, – задані у 2Dмножині, а модель AR– у 3D. Тобто, початок координат зсувається до площини об'єкту на зображенні.

Висновки.

Проведені дослідження довелиобгрунтованість підходу до впровадження доповненої реальності на базі комп'ютерного зору. Практичне застосування розглянутої математичної моделі та алгоритмів її розв'язання здійснюється на прикладі інтелектуальної системи впровадження та сприйняття доповненої реальності.

Список літератури

1. И. А. Благовещенский, Н. А. Демьянков, Технологии и алгоритмы для создания дополненной реальности, Модел. и анализ информ. систем, 2013, том 20,номер 2, 129–138
2. B. Smith. SUSAN — A new approach to low level, 1997
3. Introduction To Feature Detection And Matching [Електронний : ресурс] // Режим доступу: https://medium.com/data-breach/introduction-to-feature-detection-and-matching-65e27179885d

УДК 004.02

КОГАН А. В.
ІЛЬЄНКО Р. С.

ЗАДАЧА ВИЗНАЧЕННЯ ТРИВАЛОСТІ ВИКОНАННЯ ЗАВДАННЯ У ІНФОРМАЦІЙНІЙ СИСТЕМІ ПІДТРИМКИ УПРАВЛІННЯ КОМАНДОЮ РОЗРОБНИКІВ

Робота присвячена застосуванню наївного баєсіва класифікатора для визначення тривалості виконання завдання у інформаційній системі підтримки управління командою розробників.

Дана робота є актуальною тому, що наразі проблема менеджменту та керування проектами у сфері інформаційних технологій є однією з головних. Розробники схильні переоцінювати свої можливості від чого страждають поставлені дедлайни та псується репутація перед замовником.

Метою роботи є дослідження наївного баєсіва класифікатора для визначення тривалості виконання завдання у інформаційній системі підтримки управління командою розробників. Було проведено порівняльний аналіз послідовного та паралельного алгоритмів.

Дана задача відноситься до задач класифікації в галузі лінгвістики знаперед заданим набором класів. У роботі розглянуто послідовний алгоритм, для порівняння його з паралельною моделлю. Також було проведено експериментальні дослідження розроблених алгоритмів.

КЛЮЧОВІ СЛОВА: НАЇВНИЙ БАЄСІВ КЛАСИФІКАТОР, ПАРАЛЕЛЬНИЙ АЛГОРИТМ, КОМП’ЮТЕРНА ЛІНГВІСТИКА

The work is devoted to the use of naive Bayes classifier to determine the duration of the task in the information system of team development management support.

Nowthis work is relevant because the problem of IT project management is one of the main ones. Developers tend to overestimate their capabilities, so they suffer deadlines and damage their reputation with the customer.

The purpose of the research is to investigate a naïve classifier to determine the duration of the task in the information system of team development management support.

This task relates to the problems of classification in the field of linguisticswith predefined set of classes. Experimental studies of the developed algorithms were also conducted.

KEYWORDS: NAIVE BAYES CLASSIFIER, PARALLEL ALGORITHM, COMPUTER’S LINGUISTICS.

1. Вступ

Однією із найважливіших задач scrum-майстра є визначення кількості story point які треба надати тій або іншій задачі. Як правило, оцінка тим точніше, чим більше досвіду має спеціаліст. Він розуміє усі тонкощі роботи команди розробників, знає до якого класу важкості можуть відноситись поставленні замовником вимоги, передбачає можливі проблеми. Використовуючи фокус-фактор, з кожним спринтом оцінка часу, що буде витрачене на виконання замовлення, має набувати більшою значущості. Але таких менеджерів доволі важко знайти, також не кожна компанія може собі дозволити високооплачуваного фахівця та зайве робоче місце. Існує досить багато agile застосунків таких як scrumblr або trello, але вони не мають функціоналу передбачення витрат часу на проблему.

Цю задачу можна вирішити за допомогою наївного баєсіва класифікатора. Метою даної роботи є дослідження алгоритму для автоматичного визначення тривалості виконання завдання у інформаційній системі підтримки управління командою розробників.

2. Постановка задачі

Потрібно визначити час виконання певного завдання, тобто виявити до якого класу воно відноситься. Припустимо, що завдання - це документ, а кількість story point - це клас, отже треба визначити приналежність першого до другого. Спочатку будуємо модель класифікатору. Для цього існує навчальна вибірка з D документів, що

постійно наповнюється коригуваннями користувачів. У кожного окремого класу с існує D_c документів. k - кількість унікальних слів у документах навчальної вибірки. Також треба підрахувати W_{ic} (скільки разів і-е слово зустрічається в навчальній вибірці с класу). А $\sum_{j=1}^{k}(W_{jc})$ - загальна кількість слів документу с в навчальній вибірці. Володіючи цими даними для кожного класу визначимо значення:

$$log\frac{D_c}{D}+\sum_{i=1}^{n} log\frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})}$$

Тоді цільова функція задачі матиме наступний вигляд:

$$C_{map}=[log\frac{D_c}{D}+\sum_{i=1}^{n} log\frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})}]$$

Тобто задача полягає в наступному: визначити до якого класу належить певне завдання.

3. Існуючі методи розв'язання

Для визначення поставленої задачі, а саме до якого класу належить певне завдання, пропонується використовувати послідовний наївний баєсів алгоритм, що базується на теоремі Баєса:

$P(c|d)=\frac{P(c)*P(d|c)}{P(d)}$, де

$P(c|d)$- ймовірність, що документ d належить класу с;

$P(d|c)$ - правдоподібність, тобто ймовірність даного значення ознаки при даному класі.

$P(d)$ та $P(c)$ - безумовна ймовірність зустріти документи d та с відповідно в корпусі документів.

В машинному навчанні наївні баєсові класифікатори - це сімейство простих ймовірнісних класифікаторів. Моделі на основі НБА дуже корисні при роботі з великими наборами даних. Метою даного метода є виявлення приналежності документу до певного класу. Так визначається найбільш ймовірний клас:

$$C_{map}=arg\max_{c\,\epsilon\,C}\frac{P(c)*P(d|c)}{P(d)}$$

Таким чином необхідно для кожного класу підрахувати ймовірність та обрати той, який найбільше підходить.

$$C_{map}=arg\max_{c\,\epsilon\,C}[P(c)*P(d|c)]$$

Після припущення умовної незалежності, де умовна ймовірність документу може бути представлена як добуток умовних ймовірностей всіх слів, що входять до цього документу, та вирішенню проблеми арифметичного переповнення, використовуючи властивість логарифма добутку $log(ab)=log(a)+log(b)$, враховуючи проблему невідомих слів разом зі згладжуванням Лапласа, кінцева формула набуває вигляду:

$$C_{map}=[log\frac{D_c}{D}+\sum_{i=1}^{n} log\frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})}]$$

Представимо запропонований алгоритм:

КРОК 1. На вхід подається навчальна вибірка з с кількості класів. В кожному класі с існує D_c документів. Також поступає і документ, клас якого треба знайти.

КРОК 2. Підраховується загальна кількість унікальних слів в усіх класах навчальної вибірки.

КРОК 3. Підраховується кількість документів в усіх класах навчальної вибірки.

КРОК 4. Для кожного класу рахується сумарна кількість слів по всіх документах.

КРОК 4.1. Додається значення логарифмів, де у чисельнику стоїть значення кількості повторень і-го слова з документу, клас якого треба визначити. До цього значення додається 1 (адитивне згладжування). А у знаменнику сума загальної кількості унікальних слів в усіх класах навчальної вибірки та значення кількості слів в цьому класі.

КРОК 4.2. До суми логарифмів додається логарифм, де у чисельнику стоїть кількість документів в цьому класі, а у знаменнику загальна кількість документів.

КРОК 5. Отримуємо деякий набір значень.

КРОК 6. З цього набору максимальне значення належить класу що з найбільшою ймовірністю відповідає даному документу.

4. Розробка паралельного алгоритму

При великих обсягах даних підраховувати синхронно в одному потоці кожний окремий клас може займати доволі великі показники часу. До того ж не використовується усі можливості сучасних багатопроцесорних комп'ютерів.

Вирішити це питання допомагає паралельна модель алгоритму. Загальна схема виглядає наступним чином:

1. Підраховується всі параметри для виконання наївного баєсіва алгоритму.

2. На етапі знаходження належності документу до певного класу відбувається розбиття на потоки. Після чого по цьому класу одразу запускається наступний потік.

3. Системний синхронізатор зупиняє наступне виконання програми доки кожний потік не завершить своє виконання та не відправить необхідні дані у спільний ресурс.

4. З цього ресурсу знаходиться відповідь.

При такій схемі найбільша ефективність досягається коли кількість класів навчальної вибірки дорівнює кількості процесорів у системі. Результати порівняльних експериментів можна побачити на таблиці 1. Для експерименту було взято 10 класів однакової розмірності. Дослідження проводились з різним значенням кількості слів у кожному класі.

Табл. 1 - Порівняння швидкодії алгоритмів (у наносекундах)

К-ть слів	Послідовно	Паралельно	Прискорення
10	1423	34620	0,04110340843
100	23705	37440	0,6331463675
1000	38807	54954	0,7061724351
10000	360087	212068	1,69797895
100000	3099918	2336826	1,326550629
1000000	35943611	22484599	1,598588038
10000000	363140431	242544790	1,497209777

На рисунку 1 видно, що на маленьких об’ємах даних послідовний алгоритм працює швидше за паралельний. Це результат того, що витрати на створення та керування потоками більші ніж витрати на виконання цього завдання послідовно. Але з рисунку 2 можна побачити як паралельна модель набуває сенсу з масивами близькими до 10000 слів.

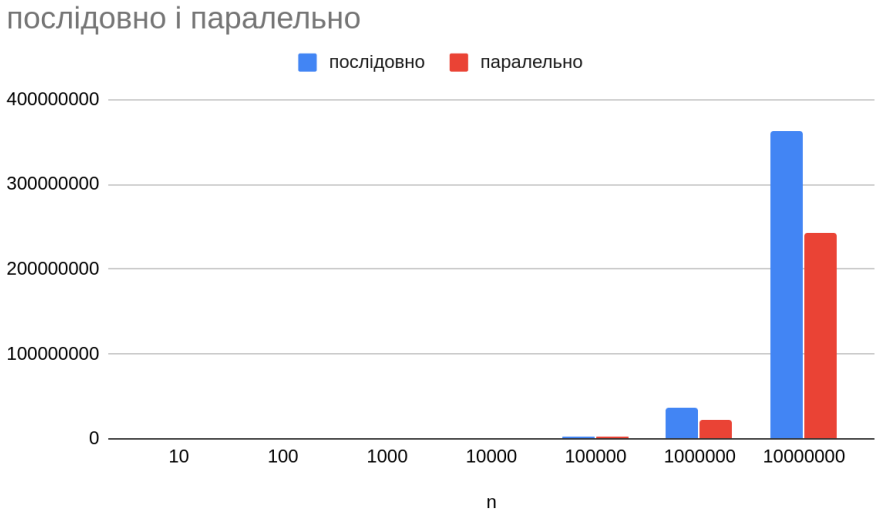


Рис 1. - Графік порівняння швидкодії послідовного та паралельного алгоритму.

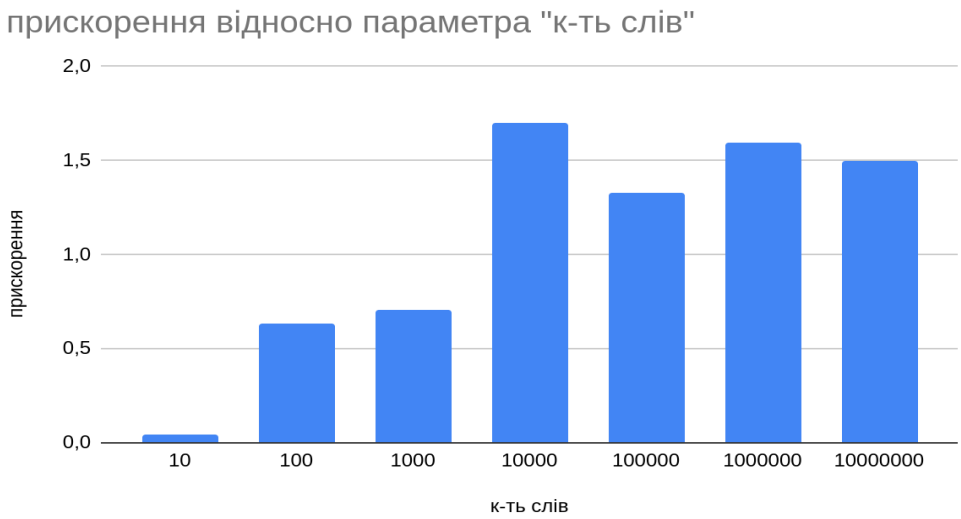


Рис 2. - Візуалізація прискорення на певній кількості слів у класі.

Висновок

Після аналізу наївного баєсіва алгоритму, можна зробити висновок, що даний класифікатор гарно підходить для вирішення задачі визначення тривалості виконання завдання у інформаційній системі підтримки управління командою розробників. Найбільшої результативності він набуває з паралельною схемою роботи при великих об’ємах даних та відповідній кількості процесорів.

Список посилань

1. Domingos, Pedro & Michael Pazzani (1997) «On the optimality of the simple Bayesian classifier under zero-one loss». Machine Learning
2. Rish, Irina. (2001). «An empirical study of the naive Bayes classifier». IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence.
3. Webb, G. I.; Boughton, J.; Wang, Z. (2005). "Not So Naive Bayes: Aggregating One-Dependence Estimators". Machine Learning.
4. Mozina, M.; Demsar, J.; Kattan, M.; Zupan, B. (2004). Nomograms for Visualization of Naive Bayesian Classifier.
5. Maron, M. E. (1961). "Automatic Indexing: An Experimental Inquiry". Journal of the ACM.
6. Minsky, M. (1961). Steps toward Artificial Intelligence.
7. McCallum, A. and Nigam K. «A Comparison of Event Models for Naive Bayes Text Classification».
8. Hand, DJ, & Yu, K. (2001). «Idiot’s Bayes — not so stupid after all? » International Statistical Review.
9. Математические методы обучения по прецедентам (теория обучения машин). К. В. Воронцов.
10. Комп’ютерна лінгвістика : Методичні вказівки до виконання лабораторних робіт для студентів спеціальностей. доц. Фіногенов Олексій Дмитрович.
11. http://scrumblr.ca/
12. https://trello.com/

УДК 004.43

ДУХІН В. О.,
МУХА І. П.

ПРЕДМЕТНО-ОРІЄНТОВАНА МОВА ДЛЯ ОПИСУ СЦЕНАРІЇВАВТОМАТИЗОВАНОГО ТЕСТУВАННЯ

У даній статті розглянуто проблеми реалізації автоматизованого тестування в частині розробки сценаріїв такого тестування на програмній платформі Node.js. Запропоновано та описано нову предметно-орієнтовану мову для опису сценаріїв автоматизованого тестування, яка в порівнянні із методами опису сценаріїв, що пропонують наявні фреймворки тестування для програмної платформи Node.js, відрізняється лаконічністю її сприйняття та написання сценаріїв шляхом застосування синтаксису S-виразів.

ФОРМАЛЬНА МОВА, ФРЕЙМВОРК, S-ВИРАЗ, ГРАМАТИКА, РБНФ

The subject of this article is the implementation issues of test automation in the context of developing test automation scenarios in Node.js. This article proposes and describes a new domain-specific language for describing test automation scenarios, which in comparison with existing methods of test scenarios describing is more laconic by using the syntax of S-expressions.

FORMAL LANGUAGE, FRAMEWORK, S-EXPRESSION, GRAMMAR, EBNF

1. Вступ

Одним з найважливіших етапів життєвого циклу розробки програмного забезпечення є перевірка працездатності написаного програмного коду шляхом його тестування задля контролю якості розробки, попередження можливих дефектів та проведення звітування перед стейкхолдерами.

Розрізняють ручне тестування, що проводиться шляхом моделювання дій користувача, та автоматизоване тестування, яке передбачає використання програмних засобів для виконання сценаріїв тестування й перевірки результатів їх виконання.

За визначенням Міжнародної ради контролю якості програмного забезпечення, автоматизоване тестування – це використання спеціалізованого програмного забезпечення (фреймворків, бібліотек або окремих інструментів) для підтримки процесів тестування, що охоплює організацію сценаріїв тестування, їх виконання та порівняння результатів виконання з певними еталонами або очікуваними результатами [1].

Метою автоматизації тестування є підвищення ефективності процесу тестування шляхом використання допоміжних програмних засобів для виконання сценаріїв тестування, що сприяє скороченню часу на тестування і спрощенню відповідного процесу.

Автоматизоване тестування здебільшого виконується шляхом застосування спеціалізованих фреймворків, що надають засоби, середовище та інструменти для проведення тестування.

Фреймворки для автоматизованого тестування найчастіше є універсальними інструментами та, відповідно докласичної піраміди тестування [2], можуть використовуватисядлятестуваннярізних рівнів архітектури системи. Проте, деякі фреймворки тестування можуть бути орієнтованими лише на певні системи, середовища або рівні тестування.

Незалежно від виду фреймворку, важливою складовою процесу автоматизованого тестування є розробка відповідного сценаріютестування, який би охоплював усі можливі варіанти використання програмного забезпечення, що тестується. Сценарій тестування являє собою опис функціональних та інколи нефункціональних вимог до програмного забезпечення, а рівень його деталізації визначається відповідно до архітектури системи.

У випадку невдалого налаштування фреймворку або надмірної деталізації сценарію тестування,використання автоматизованого тестуванняможе ускладнити процес розробки програмного забезпечення.Тому дослідження ефективних

методів та засобів автоматизації тестування програмного забезпечення, зокрема, розробки сценаріїв автоматизованого тестування, є достатньо актуальною задачею.

2. Підходи до реалізації сценаріївавтоматизованого тестування

Для більшості мов програмування, програмних платформ та стеків програмних технологій існують програмні засоби, що надають можливість створення сценаріїв тестування, їх організації та подальшого проведення автоматизованого тестування.

Більшість таких засобів передбачають, що сценарії автоматизованого тестування повинні бути написаніна тій самій мові програмування, що й вихідний код програмного застосунку.Це зумовлено тим, що мови програмування, як правило, підтримують імперативний або декларативний стиль представлення інструкцій, які є достатньо наочними й зрозумілими. Фреймворки для автоматизованого тестування, своєю чергою, надають низку надбудов для цієї мови програмування і створюють програмне середовище для проведення тестування.

Так, наприклад, для програмної платформи Node.jsзгідно зі звітом «TheStateofJavaScript» [3], найпопулярнішими засобами для автоматизації тестуваннядля програмної платформи Node.js у 2019 році стали:

1. Jest[4] – фреймворк,що є розробкою компанії Facebook, з відкритим програмним кодом, і широким функціоналом, який включає: підтримку тестування через зліпки станів системи, паралельне виконання сценаріїв тестування, можливість ізолювання компонентів системи та інструменти порівняння результатів;
2. Mocha[5]– фреймворк послідовного виконання сценаріїв тестування з відкритим програмним кодом і базовим функціоналом, що активно підтримується розробниками програмної платформи Node.js;
3. Jasmine[6]– один з найперших фреймворків платформи Node.jsз відкритим програмним кодом і з ідеологією розробки, що керована поведінкою;
4. Ava[7]– інструментарій з відкритим програмним кодом і мінімальним функціоналом, необхідним для опису сценаріїв тестування;
5. Cucumber[8]– фреймворк з відкритим програмним кодом, який передбачає використання для опису сценаріїв тестуванняпредметно-орієнтованої мови Gherkin.

Більшість із цих фреймворків (окрім Cucumber) мають прикладний програмний інтерфейс, реалізований мовою програмування JavaScript. Це створює певні проблеми для організації процесу тестування в частині написання сценаріїв, оскільки вимагає від інженерів з тестування та інших стейкхолдерів навичок програмування мовою JavaScript. Окрім того, сценарії у вигляді програмного коду часто є складними для сприйняття та розуміння, а оцінка їх валідності не завжди може бути адекватною.

Вирішенням цих проблем може бути використання предметно-орієнтованої мови програмування для опису сценаріїв тестування.Наприклад, у фреймворку Cucumber для цих цілей використовуєтьсямоваGherkin[9]. Ця мова побудованана конструкціях розмовної мови іявляє собою звичайний текст, що зрозумілим чином описує сценарій тестування: передумови, послідовність кроків, очікувані результатитестування тощо.Від розробника сценарію вимагається лише дотримання структури опису сценарію та використання специфічних ключових слів. Внаслідок простоти мовиGherkin, сценарії тестування цією мовою може розробляти будь-яка людина, незалежно від рівня її знать та вмінь. Це значно спрощує взаємодію програмістів, інженерів з тестування та інших стейкхолдерів в процесі реалізації програмного продукту.

Проте, описаний мовою Gherkin, сценарій тестування не є програмним кодом. Тому виникає необхідність у додатковому перетворенні такого опису сценарію до виду, прийнятного для подальшої обробки інструментальними засобами фреймворку. Cucumber пропонує для цього інструменти парсингу тексту на мові Gherkin тастворення на основі отриманих данихсценаріїв тестування на мові JavaScript.Однак, такі сценарії, як і сценарії тестування, отримані із

використанням інших фреймворків, мають вигляд програмного коду і знову ж таки є незрозумілими для багатьох стейкхолдерів.

На противагу описаному, існує підхід до розробки предметно-орієнтованої мови програмування, що передбачає використання металінгвістичних абстракцій та словникового запасу базової мови програмування [10, 11]. Це дає змогу створити формальну мову, що схожа на звичайну розмовну мову, проте залишається мовою програмування в контексті базової мови.

3. Метод опису сценаріїв автоматизованого тестування

З урахуванням зазначеного вище підходу пропонується створити предметно-орієнтовану мову програмування, що описує сценарій тестування у вигляді S-виразів – нотації для позначення вкладених списків деревоподібної структури, запропонованої Джоном Маккартні [12].Синтаксис S-виразів знайшов широке використання, зокрема, у таких мовах програмування як Lisp, Schemeта WebAssembly.

S-вирази являють собою ієрархічно зв’язану групу атомів (лексем та списків), що представляються відповідно до наступного формату:

$$(x\ y\ z) \equiv \Big(x\ .\ \big(y\ .\ (z\ .\ NIL)\big)\Big),$$

де x, y, z – атоми, NIL – кінець списку.

Внаслідок деревоподібної структури S-вирази дозволяють описувати взаємопов’язані структури, а їх мінімалістичний синтаксис забезпечує легке сприйняття тексту при читанні.

Згідноз стандартом опису сценаріїв тестування [13, 14], у відповідній формальній мовінеобхідно забезпечити можливість описання наступних полів сценарію тестування:

- ідентифікатор сценарію,
- назвасценарію,
- опис сценарію,
- посилання на об’єкт тестування,
- передумови тестування,
- послідовність кроків сценарію,
- постумови,
- додаткові дані для виконання сценарію,
- очікувані результати.

Основним способом створення проблемно-орієнтованих мов, як різновиду формальних мов, є їх породження за допомогою формальних граматик. За ієрархією Чомскі-Шутценберґера [15], мови програмування, як правило, будуються на основі контекстно-вільних граматик, які можуть бути описані наступним чином:

$$G = (N, \Sigma, P, S),$$

де N – скінченна множина термінальних символів (символів алфавіту формальної мови), Σ– скінченна множина нетермінальних символів,якими позначаються синтаксичні конструкції мови,P – скінченна множина правил виводу речень формальної мови,S – початковий символ, що є старшим нетермінальним символом, який визначає клас мовних об’єктів, для опису яких призначена граматика *G*.

Алфавіт термінальних символів Nрозробленої формальної мови являє собою підмножину символів стандартної таблиці ASCIIта включає великі й малі літери латинської абетки (A-Zта a-z), цифри (0-9), а також спеціальні та керівні символи: перенесення рядка, фігурні, квадратні та круглі дужки, символи крапки, коми, дорівнює, крапки з комою та пробілу.

До множини нетермінальних символів відносяться найменування полів стандартного сценарію автоматизованого тестування, що описані вище, та символи опису S-виразів, а саме:

- LETTER – великі та малі літер латинської абетки;
- DIGIT – цифри від 0 до 9;
- CHAR – спеціальні та керівні символи стандартної таблиці ASCII;
- KEYWORD – ключові слова для опису сценарію тестування, а саме:
- description – змістовне описання мети сценарію або цілі тестування;
- entity – посилання на компонент системи, що тестується;
- sourceFile – посилання на файл вихідного коду, де розміщений компонент системи, що тестується;
- doubles – секція об’явлення «двійників» компонент системи, таких як зовнішні залежності, фіктивні функції тощо;

– fakeFunction – об’явлення фіктивної функції як «двійника» компонента системи;
– before та precondition – описання передумов тестування;
– after та postcondition – описання постумов тестування;
– call – виклик компонента системи, що тестується;
– saveReturnAs – збереження результату виклику компоненти системи у змінну;
– cases – секція опису конкретних сценаріїв тестування;
– test – об’явлення конкретного сценарію тестування;
– steps – кроки сценарію, що необхідно виконати для проведення тестування;
– expect – очікувані результати тестування;
– RULE – одиниця опису сценарію тестування, що являє собою відповідне поле у стандарті опису сценаріїв тестування [13, 14];
– RULE_PARAM – параметри та аргументи, що розширюють описання поля сценарію тестування;
– SCENARIO – сукупність полів/правил, що описують вимоги до системи, що тестується.

Усі правила виводу лексем даної формальної мови можуть бути описані із застосуванням розширеної нотації Бекуса-Наура (РБНФ)[16], що наведена у таблиці 1. Початковим нетермінальним символом Sрозробленої формальної мови є символ SCENARIO.

Табл. 1. Правила виводу лексем формальної мови

RULE = KEYWORD, "`", { LETTER | DIGIT | CHAR }, "`", [{ RULE_PARAM }];
RULE_PARAM = "(", [{ RULE | LETTER | DIGIT | CHAR | "," }], ")";
SCENARIO = "scenario", "`", { RULE | LETTER | DIGIT | CHAR } , "` (", [{ RULE, "," }], ");";

Висновки

Більшість наявних підходів до опису сценаріїв автоматизованого тестування передбачають їх написання безпосередньо мовою програмування, яка застосовується у вихідному програмному застосунку, що потребує певної кваліфікації інженерів з тестування. Використання ж деякими фреймворками проблемно-орієнтованих мов, таких якGherkin, для цих цілей також не вирішує проблему, оскільки потребує додаткового опису сценаріїв тестування програмним кодом.

Як розв'язання даної проблеми, розроблена нова предметно-орієнтована мова опису сценаріїв автоматизованого тестування, яка в порівнянні ізметодами опису сценаріїв, що пропонують наявні фреймворки тестуваннядля програмної платформи Node.js, відрізняється лаконічністю її сприйняття та написання сценаріїв шляхом застосуваннясинтаксису S-виразів.

Оскільки розроблена проблемно-орієнтована мова є мовою програмування, то, відповідно, немає потреби у створенні додаткових документів для проведення автоматизованого тестування, що значним чином спрощує процес тестування, а також вирішує проблему взаємодії зацікавлених сторін внаслідок того, що дана мова максимально наближена до звичайної розмовної мови.

Список використаних джерел

1. Standard Glossary of Terms Used in Software Testing [Електронний ресурс] // International Software Testing Qualifications Board. – 2019. – Режим доступу до ресурсу: https://glossary.istqb.org/.
2. Cohn M. Succeeding with Agile. Software Development Using Scrum / Mike Cohn., 2010. – 512 с.
3. TheStateofJavaScript [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://2019.stateofjs.com/testing/.
4. Документація до фреймворку тестування Jest [Електронний ресурс] – Режим доступу до ресурсу: https://jestjs.io/.
5. Документація до фреймворку тестування Mocha [Електронний ресурс] – Режим доступу до ресурсу: https://mochajs.org/.
6. Документація до фреймворку тестування Jasmine [Електронний ресурс] – Режим доступу до ресурсу: https://jasmine.github.io/pages/docs_home.html.

7. Документація до фреймворку тестування Ava [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/avajs/ava.
8. Документація до фреймворку тестування Cucumber [Електронний ресурс] – Режим доступу до ресурсу: https://cucumber.io/docs/cucumber/.
9. GherkinSyntax [Електронний ресурс] – Режим доступу до ресурсу: https://cucumber.io/docs/gherkin/.
10. Abelson H. Structure and Interpretation of Computer Programs / H. Abelson, G. Sussman. – London: The Massachusetts Institute of Technology, 1996. – 883 с. – (Second edition).
11. Fowler M. Domain Specific Languages / M. Fowler, R. Parsons., 2010. – 640 с. – (Signature Series).
12. McCarthy J. Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I [Електронний ресурс] / John McCarthy // MIT Press. – 1960. – Режим доступу до ресурсу: http://www-formal.stanford.edu/jmc/recursive/recursive.html.
13. Куликов С. Тестирование програмного обеспечения. Базовый курс. / Святослав Куликов. – Минск: Четыре четверти, 2017. – 312 с.
14. Best Test Case templates with examples [Електронний ресурс] – Режим доступу до ресурсу: https://geteasyqa.com/qa/best-test-case-templates-examples/.
15. Chomsky N. Syntactic Structures / Noam Chomsky. – Berlin, New York: Mouton de Gruyter, 2002. – 118 с. – (Second edition).
16. ISO/IEC 14977:1996 (Information technology — Syntactic metalanguage — Extended BNF) [Електронний ресурс] – Режим доступу до ресурсу: https://www.iso.org/standard/26153.html.

УДК 004.43

КАДЖАЯ В.М.

АРХІТЕКТУРА МОВИ ПРОГРАМУВАННЯ KUJIRA

У даній статті розглядається архітектура мови програмування Kujira, а так само більш детальне пояснення підходу реалізації.
КЛЮЧОВІ СЛОВА: lark, Flex, lex, LLVM, LALR, Python, C++, SWIG, Tcl, Perl, Guile

This article discusses the architecture of the Kujira programming language, as well as a more detailed explanation of the implementation approach.
KEYWORDS: lark, Flex, lex, LLVM, LALR, Python, C++, SWIG, Tcl, Perl, Guile

Вступ

Відповідно до стандарту ANSI / IEEE 1471 2000 року архітектура програмного забезпечення: «Фундаментальна організація системи, втілена в її компонентах, їх відносинах один з одним і з навколишнім середовищем, а також принципи, що визначають її дизайн і розвиток».[6, 8] Тому компоненти моделюються «протягом усього життєвого циклу розробки та послідовно переглядаються на розгортання та час виконання». [7, 8]

На сьогоднішній день існує дуже багато різноманітних мов програмування кожен з яких має свою перевагу. Але підхід до реалізації в кожній мові однаковий. У кожній мові можна зустріти граматику, лексичний і синтаксичний аналізатор. Деякі мови можуть також використовувати LLVM, де є frontend та backend. Фронтенд перетворює текст програми на мові високого рівня в текст на проміжній мові, оптимізатор виробляє над ним різні оптимізації, а бекенд генерує код цільового процесора (на асемблері або безпосередньо у вигляді бінарного файлу).

1. Основна частина

В нашій ранній опублікованій роботі ми розглянули популярні граматики для мов програмування їх порівняння, переваги і недоліки, а так само використання LALR граматики в мові Kujira.[1]

Розглянемо на прикладі:

```
?start: calc | NAME "=" calc -> assign | value | printval
```

Це запис граматики в формі LALR. ?start**:** це оператор який приймає числа, створює змінні, арифметичні операції і масиви, і вбудовані функції такі як print.

Лексичний аналізатор - це частина компілятора який читає вихідний файл і знаходить токени або лексеми. Прикладом може бути додавання двох чисел. Що він повинен зробити, так це визначити ідентифікатори, оператор присвоювання і

числа, і оператор додавання. Так як використовується lark[4] то не доводилося реалізовувати лексичний аналізатор з нуля. Якщо використовувати LLVM, то для цього потрібно створювати аналізатор використовуючи *lex* або *Flex*.

Так само крім лексичного аналізатора існує синтаксичний аналіз, який входить до програмного забезпечення підтримки мови Kujira. Синтаксичний аналіз - це дерево розбору коду або більш правильно зіставлення послідовності лексем з його формальної граматикою.

Розглянемо приклад.

Нехай вихідний код буде таким

print "hi"

Відповідно дерево коду буде таким як на рис.1:

```
Tree(string, [Token(ESCAPED_STRING, '"hi"')])
```

Рис.1- Синтаксичний аналіз функції

Як можна помітити при парсінгу вихідний код перетворюється в структуру даних, а точніше в дерево. Чому в дерево? Тому що обробка такого дерева відбувається простіше і швидше.

Після всіх аналізів і парсинга вся структура переводиться в **машинний код.**

Семантика мови програмування - це відповідність між синтаксично правильними програмами і діями абстрактного виконавця, тобто це сенс синтаксичних конструкцій. [2] Розглянемо приклад з мови програмування Kujira.

Табл.1 - Семантичне правило

Опис арифметичного правила використовуючи граматику LALR	**Семантичне правило**
?calc: calc "+" prod -> add	E.val := E.val1 + T.val
?calc: calc “-" prod -> sub	E.val := E.val1 - T.val
?calc: calc “*” prod -> mul	E.val := E.val1 * T.val

Так як дана мова відноситься до скриптових мов то переклад з мови програмування в машинну відбувається за допомогою віртуальної машини.

Після реалізації граматики, лексеми і синтаксису можна протестувати мову. Для тестування коду користується кінцевий автомат. Кінцевий автомат - абстрактний автомат, число можливих внутрішніх станів якого звичайно.

Створивши деякі бібліотеки для мови програмування, було вирішено порівняти його з мовою програмування Python. Чому саме на ній, тому що багато бібліотек написані на Python використовують C++. Стандартна бібліотека Kujira також частково написана на C++.

Спершу порівнюються сортування так як це найпростіше з чого можна почати і один з найпопулярніших методів порівняння. Вбудована в мову сортування використовує стандартну STL бібліотеку де є реалізований метод sort (). Тестування проводиться за допомогою мови Python, для власної функції sort () аргументами були вектора. Функція сортування у мові Kujira була написана саме на C++ використовуючи вектори.

Функція сортування:

std::vector<int> arr(std::vector<int> a){…}

Наступним етапом було порівняння швидкості, так як C++ є одним з найшвидших мов програмування, то можна було вважати, що C++ по швидкості випередить Python. Але на подив швидким виявився Python, завдяки алгоритму сортування який він використовує, а саме Timsort. Timsort - це ефективна комбінація декількох алгоритмів, зокрема mergesort і insertsort. Відповідно функція sort Python була в рази швидше ніж Kujira. Тоді було вирішено переписати бібліотеку використовуючи timsort. Результати теж були цікаві.

```
(7.0, 49.0, 73.0, 58.0)
0.008431817000000001
0.00077153900000016
```

Рис.2-Результат

Перший час - час сортування написане на C++, друге на Python. З чим це пов'язано? Це пов'язано з тим що у Python більшість модулів, а також ядро написано на мові C. Також проблемою Kujira у швидкості було те що її архітектура має декілька слоїв, а саме frontend мови використовує Python для парсингу, а backend - C++ для компіляції у машинний код. Ця реалізація може нагадувати інфраструктуру LLVM. Яка зараз доволі часто зустрічається.

Робота з файлами:

Час читання невеликих файлів практично однакова. Але у Python'а є один недолік при читанні файлів - якщо файл містить багато даних то швидкість алгоритму зменшується на відміну від Kujira.

Для реалізації бібліотек використовувався SWIG.[5] SWIG - інструмент розробки програм, що автоматично генерує прив'язки між кодом C / C++ і поширеними мовами сценаріїв, включаючи Tcl, Python, Perl і Guile. SWIG підтримує більшість типів даних C / C++, включаючи покажчики, структури і класи. На відміну від багатьох інших підходів, SWIG використовує декларації ANSI C / C++ і вимагає, щоб користувач практично не вносив змін до базового C-код [3].

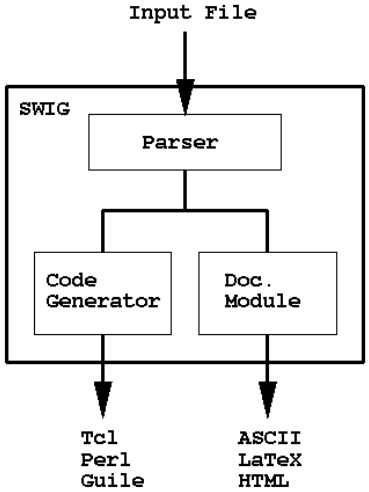


Рис.3- Архитектура SWIG

В основі SWIG лежить уасс для читання вхідного файлу і конвертації в відомі скриптові мови. Вхідним файлом є код написаний на мові C / C ++ який за допомогою уасс парсеру перекладається в один з даних мов Tcl, Python, Perl і Guile.

Висновок

Kujira як мова програмування може в цілому конкурувати з багатьма мовами програмування. Kujira використовує “слоїстий” підхід, де Python використовуєтьс для парсингу, SWIG є проміжним єтапом, вистпає “з’єднувачем” між Python та C++. C++ вже оброблює та виводить результат. Типи і структури даних такі як int, float, hash або set в даній мові це класи які наслідувані від головного класу Object. Таким чином Kujira може нагадати такі мови як Java або Python, де тип і структури наслідуються від одного класу. У подальшому буде оптимізуватися компілятор і додаватися нові функції.

Список літератури

1. Очеретяний О.К., Каджая В.М., Баклан І.В. //Матеріали VIII Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених з автоматичного управління, 2020, с.49.
2. Орлов С.А. Теория и практика языков программирования: Учебник для вузов. Стандарт 3-го поколения**,** 2013, с.182.
3. David M. Beazley, SWIG : An Easy to Use Tool for Integrating Scripting Languages with C and C++ // Presented at the 4th Annual Tcl/Tk Workshop - 1996
4. lark. URL: https://github.com/lark-parser/lark
5. SWIG. URL: http://www.swig.org/
6. IEEE. IEEE Recommended Practice for Architectural Description of Software-intensive Systems. Standard 1471-2000, IEEE, 2000.
7. Object Management Group. Unified Modeling Language: Superstructure, Version 2.0.Technical report, OMG, 2004.

8. Hubert Baumeister, Florian Hacklinger, Rolf Hennicker, Alexander Knapp, Martin Wirsing //A Component Model for Architectural Programming, Electronic Notes in Theoretical Computer Science Volume 160, 2006, c.75-96.

УДК 004-92

ШВЕЦЬ Е.Я.,
ЯЗЕНОК М.С.,
МУХА І.П.

СПЕЦІАЛІЗОВАНИЙ РЕДАКТОР ДЛЯ ПЛАНУВАННЯРОБОЧОГО ПРОСТОРУ ПРИМІЩЕНЬ З ПІДТРИМКОЮ ІНТЕГРАЦІЇ З МОБІЛЬНИМИ ДОДАТКАМИ

У даній статті розглянутареалізація спеціалізованого графічного редактора для створення та редагування плану приміщень закладів сфериHoReCa (ресторани, кафе, бари). Особливістю даного редактора є можливість його інтеграції з мобільними застосуваннями, що є вкрай важливим для реалізації багатьох онлайн-сервісів.

Ключові слова: редактор плану приміщення, android.graphics, 2D графіка.

This article discusses the implementation of a specialized graphic editor for creating and editing a plan for the premises of HoReCa establishments (restaurants, cafes, bars). The feature of this editor is the ability to integrate it with mobile applications, which is extremely important for the implementation of many online services.

Keywords: room plan editor, android.graphics, 2D graphics.

Вступ

На сьогоднішній день важливим трендом у сфері HoReCa є вдосконалення сервісу і якості обслуговування клієнтів.

Одним із основних підходів щодо вирішення цих проблем є створення зручних комунікаційних каналів для спілкування з клієнтами на основі сучасних IT-технологій.

Серед нових методів комунікації, які використовуються у сфері послуг, особливе місце займає онлайн-взаємодія з клієнтами за допомогою мобільних додатків. Такі послуги уже надаються багатьма закладами сфери HoReCa в частині автоматизації процесу формування замовлення, роботизації доставки тощо.

Розповсюдженим різновидом послуг, зокрема у ресторанному бізнесі, що безпосередньо впливає на якість сервісного обслуговування, є можливість бронювання клієнтом столика в онлайн-режимі. Такі послуги вже надаються багатьма закладами сфери «HoReCa» (ресторани, кафе, бари), однак вони не передбачають, наприклад, можливості бронюванні конкретного столика у залі шляхом його безпосереднього вибору на плані приміщення.

Реалізація зазначеної функціональності, зокрема, мобільним додатком потребує наявності спеціалізованого редактора – програми, що дозволяє створювати різноманітні зображення на екрані цифрового пристрою.

Огляд існуючих графічних редакторів

Аналіз існуючих графічних редакторів показав, що багато із них мають необхідні інструменти для планування приміщень, проте усі вони призначені для використання або на локальному комп'ютері користувача, або як веб-застосування. Реалізація функцій таких редакторів не враховує специфіку використання в рамках мобільних додатків, а їхній графічний інтерфейс не орієнтований на роботу із смарт-пристроями, специфікою яких є різноманіття типорозмірів екранів.

Окрім того, специфіка графічних редакторів передбачає підтримку наступних стандартних функцій: створення малюнка з графічних примітивів (точок, ліній, кривих) використання для малювання різних кольорів

і «кисті» (ліній різної ширини і конфігурації), операції з фрагментами малюнка, зміна пропорцій малюнка або його частини тощо. Однак, для редактора, призначеного для роботи з моделями організації робочого простору приміщень в частині побудови схем розміщення окремих посадочних місць та різних функціональних зон таке різноманіття можливостей є надлишковим.

Тому створення спеціалізованого редактора дляпланування приміщення і окремих елементів інтер'єру на екрані смарт-пристрою представляє великий інтерес для широкого кола завдань і є досить актуальною задачею.

Підходи до побудови редактора для планування приміщення

Принципи обробки зображень різними графічними редакторами суттєво відрізняються один від одного. Загалом виділяють два основних підходи до представлення графічної інформації: растровий і векторний.

Растровий підхід передбачає побудову графічних зображень шляхом послідовного зафарбовування окремих пікселів растра екрану. Побудовані таким способом елементи не існують як окремі об'єкти, а сприймаються як сукупність окремих пікселів.

Векторний же підхід ґрунтується на використанні математичних описів зображення як сукупності графічних примітивів (точок, ліній тощо). Побудовані таким способом елементи зображення сприймаються як окремі об'єкти, які можна легко перемістити, видалити, змінити їх розміри. Саме тому векторну графіку часто називають об'єктно-орієнтованою.

Таким чином, зображення у векторному форматі надає широкі можливості для редагування окремих об'єктів, які можна масштабувати, переміщувати, повертати тощо.

Оскільки специфікою редактора для планування робочого простору приміщення є реалізація можливості візуалізації окремих елементів інтер'єру і окремих зон приміщення, то саме векторний формат доцільно взяти за основу при реалізації такого редактора.

Враховуючи необхідність забезпечення онлайн-взаємодії з клієнтами за допомогою мобільних додатків, як одного із сучасних способів реалізації комунікації між закладами індустріїгостинності і клієнтами, такий редактор повинен також підтримувати відповідну інтеграцію з мобільними додатками.

Функціональність спеціалізованого редактора FloorPlanEditor

FloorPlanEditor - це спеціалізований редактор для плануванняробочого простору приміщень різноманітних закладів сфери «HoReCa», зокрема, залів ресторанів, кафе, барів, розроблений з урахуванням зазначених вище підходів. Його основна функціональність передбачає створення схем розміщення посадочних місць і спеціальних зон у залах даних закладів, а також можливість візуального представлення в режимі онлайн поточного статусу столиків (вільний, забронbований тощо).

Даний редактор має наступні спеціальні інструменти, необхідні для зручної роботи по формуванню плану приміщення:

«структурний елемент» - інструмент для візуалізації елементів конструкціїприміщення (стін, вікон, дверей),

«столик» - інструмент для створення столика на плані приміщення,

«іконка» - інструмент, який створює іконку для позначення типу приміщення,

«текст» - інструмент, який використову-ється для виведення назв різних зон приміщення («гардероб», «бар» тощо).

Дані інструменти можна налагоджувати шляхом зміни їх параметрів:

- для столика – номера столика, кількості місць за ним;
- для структурнихелементів– їх типу (вікно, стіна, двері);
- для тексту – розміру шрифту.

Кожен елемент плану можна масштабувати, змінювати його положення та нахил. Також підтримуються функції дублювання елементів (столиків, стін, вікон, дверей, тексту, іконок) та їх видалення.

Всі інструменти відображаються на панелі керування інструментів, тому ними легко користуватися.

Даний редактор підтримує наступні режими роботи:

- режим редагування, який надає можливість менеджеру закладу створювати і редагувати план приміщення (схему розміщення посадочних місць у залах);

- режимбронювання, який надає можливість клієнту переглядати статус столиків («вільний», «зайнятий») в режимі реального часу та обирати столик для бронювання;

- режим обслуговування – спеціальний режим для менеджера закладу, який надає можливість редагувати статус столиків («вільний», «забронюваний», «обслуговується») в режимі реального часу.

Специфікою даної розробки є те, що данийредактор підтримує інтеграцію з середовищами мобільних додатків.

FloorPlanEditor дозволяє серіалізовувати усі дані в спеціальний компактний формат, який підтримує GoogleProtocolBuffers, для більш швидкого обміну інформацією під час синхронізації між окремим мобільними додатками.

Реалізація редактораFloorPlanEditor

Актуальність задачі побудови графічних зображень, зокрема, креслень, планів та схем приміщень на екрані смарт-пристрою привела до появи великої кількості потужних бібліотек для вирішення цієї проблеми.

Платформа Android пропонує різноманітні інтерфейси для двовимірного та тривимірного відображення графіки на смарт-пристроях, які взаємодіють із виробниками графічних драйверів. В Android існує три API для малювання на екрані зображень: Canvas, OpenGLES та Vulkan.OpenGLESта Vulkan надають повний інструментарій для роботи як з 2D, так і з 3D-графікою,Canvas підтримує лише 2D-графіку.

Оскільки, в розроблюваному спеціалізова-ному редакторі не потрібна ні 3D-графіка, ні використання освітлення, ні особливі зміни геометрії, то розробка була реалізована на основі бібліотеки android.graphics.Canvas зі спрощеним способом рендеру 2D-графіки.

Canvasнадає графічні інструменти низького рівня, такі як полотна, кольорові фільтри, точки та прямокутники, які дозволяють безпосередньо керувати малюванням на екрані.

Для відображення в редакторі графічних об’єктів, якими є окремі елементи зображенняна плані, передбачені класи, породжені відабстрактного класу DrawElement. Їх функціональність передбачає зберігання даних відповідного графічного елементу та реалізацію його відображення за допомогою API Canvas з бібліотеки android.graphics.

Кожний об’єкт-елемент зберігає власний набір матриць трансформацій (матрицю переміщення, матрицю обертання та матрицю масштабування). За допомогою цих матриць реалізується зміна (трансформація) зовнішнього вигляду елемента на плані. Для зміни матриць трансформацій в DrawElement визначено спеціальні методи:move, resize, rotate.

Також кожний об’єкт-елемент має початкову форму (набір координат в нетрансформованому просторі), наприклад, чотири точки для чотирикутника (клас android.graphics.RectF). Початкова форма визначається в нащадках абстрактного класу DrawElement.

Під час візуалізації елемента на плані, матриці переміщення та обертання об’єкта застосовуються до Canvas за допомогою наданих бібліотекою методів, а матриця масштабування застосовується безпосередньо до початкової форми об’єкта в нащадках для забезпечення якості зображення. Кожен нащадок реалізує власний спосіб застосування цієї матриці, наприклад, в чотирикутнику виконується трансформування координат чотирьох точок.

Якщо б матриця масштабування застосовува-лася до всього полотна, то графічні елементи втратили би можливість коректного відображення. Так масштабований чотирикутник з 5-ти піксельним контуром відобразився би наступним чином:

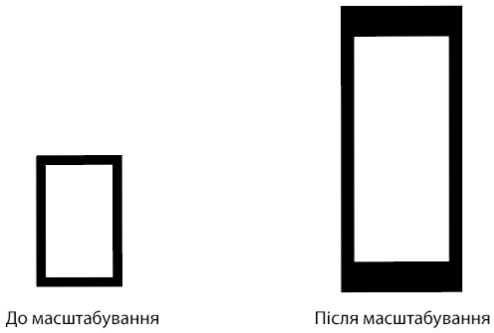


Рис.1. Схема переходу між графічними модами

Для зв’язування спеціалізованого редактора з бібліотечним класом android.view.View, який служить для виведення зображення на екран андроїд-пристрою, передбачено клас DrawingViewProxy, який по патерну Proxy підмінює виклики основних методівView:

onDraw – метода, який викликається при малюванні об’єктів на полотні;

onTouchEvent – метода, який викликається при надходженні подій (жестів на сенсорному екрані) від користувача.

Усі події обробляються за допомогою гнучкої системи інструментів для малювання(drawingmods), яка створена на базі патернуState.

Для реалізації інструментів редактора служить абстрактний клас DrawingMode та його нащадки, де кожний нащадок є реалізацією одного із інструментів. DrawingMode має метод для отримання подій користувача, які надходять з вище визначеного DrawingViewProxy.

Нащадки DrawingMode утворюють патернState. Тобто, під час оброки події один інструментможе змінитися на інший.

Схема переходів між інструментами для малювання(модами) представлена на рис.2.

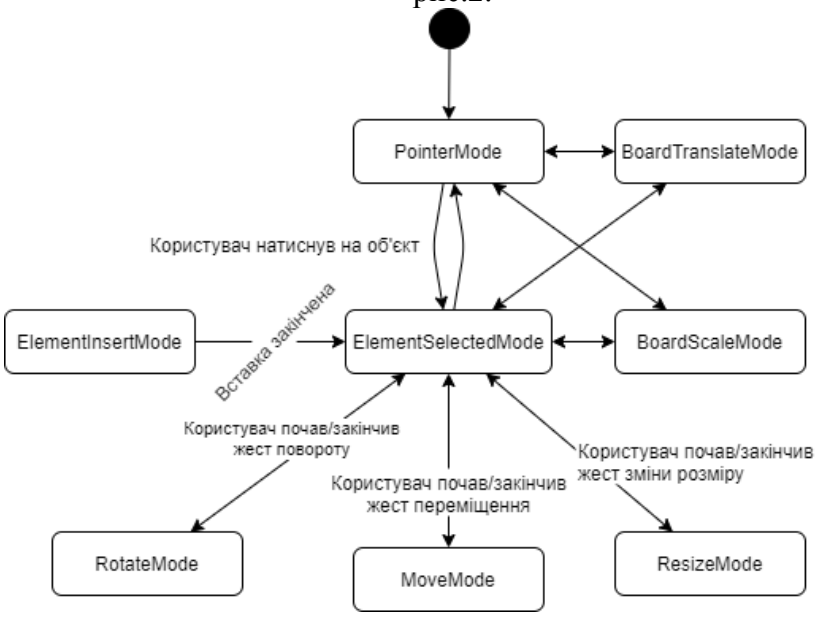


Рис.2. Схема переходу між інструментами

Наприклад, інструментElementSelectedMode змінює поточний інструмент на MoveMode, якщо користувач натиснув на об’єкт і почав жест пересування. MoveMode в свою чергу визначає зсув, створений жестом, і викликає метод DrawElement.move (floatoffsetX, floatoffsetY) у вибраного DrawElement. Після завершення жесту пересування (користувач закінчив натискання на екран), MoveModeзмінюється на ElementSelectedMode.

Для реалізації режимів редактора створено три відповідних класи-фасади (патернFacade) зі спільним батьківським класом:

EditorRestaurantDrawingBoardFacade – для підтримки режиму редагування,

RestaurantReservationDrawingBoardFacade – для підтримки режиму бронювання,

ServicingRestaurantDrawingBoardFacade – для підтримки режиму обслуговування.

Фасади створені для спрощення взаємодії додатків з бібліотеками. Додаткам необхідно лише зв'язати фасад з View, на якому буде виводитися зображення, та встановити необхідні дані (список столів, стін, вікон і т. ін.). Фасад для редагування також надає метод для встановлення обраного інструменту.

За допомогою інтерфейсів фасадівздійснюється обмеження взаємодії додатку з ядром бібліотеки, що забезпечує меншу вірогідність порушити нормальний хід роботи програми.

Для серіалізації даних використовується бібліотека GoogleProtocolBuffers. Основними перевагами цієї бібліотеки є те, що вона переводить дані в дуже компактний формат, що є вкрайважливим для синхронізації додатків із сервером.

На рис.3. наведена структура створеного редактору:

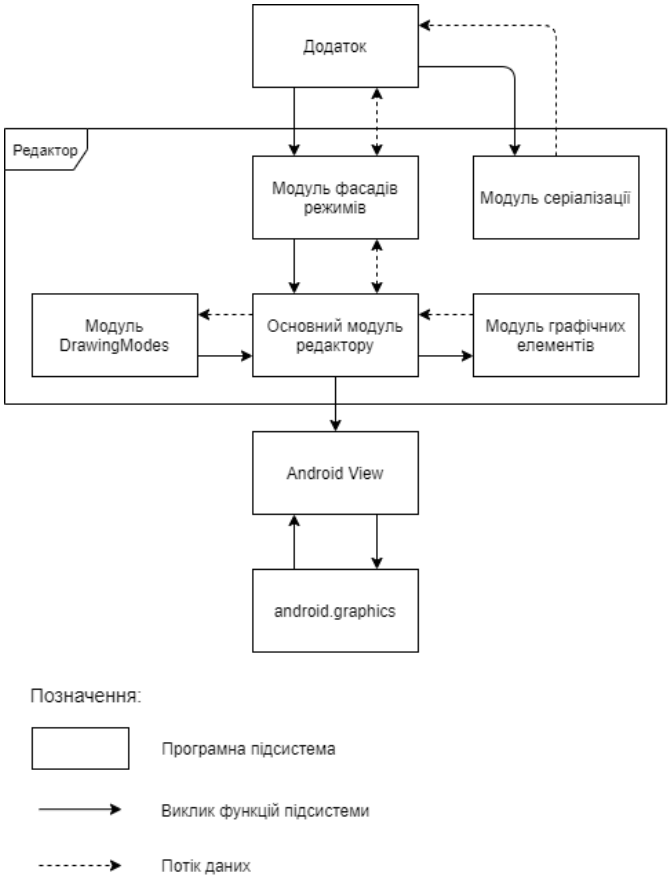


Рис. 3. Структура редактораFloorPlanEditor

Основний модуль, якийзабезпечує взаємодію усіх інших модулів редактору, включає в себе DrawingViewProxy, модуль графічних елементів, який представляє собою ієрархію DrawElement'ів, модуль DrawindModes, якийвключає в себе реалізацію інструментів, та модуль фасадів, який зв'язує утворену бібліотеку редактора з Android-додатками. Також, мобільним додаткам надається можливість використовувати модуль серіалізації для подальшого збереження даних редактора.

Переваги спеціалізованого підходу до створення інструментарію для побудови та редагування плану приміщення засобами мобільного додатку очевидні: створюється реальна можливість отримання досить компактного і простого в експлуатації програмного продукту, що містить лише потрібні для вирішення прикладних задач предметної області функції.

Висновки

На сьогоднішній день існують графічні редактори, які дозволяють відображати і редагувати плани приміщень і окремі елементи інтер'єру, однак їх неможна інтегрувати до мобільних додатків. Тому створення спеціалізованого програмного забезпечення для відображення і редагування графічних елементів на екрані смарт-пристрою представляє великий інтерес для ряду прикладних задач.

Одним із підходів до створення такого програмного забезпечення, зокрема, редактора для планування приміщень, є використання засобів бібліотеки android.graphics.Canvas.

З використанням даної бібліотеки був розроблений спеціалізований редактор FloorPlanEditorдля створення схем розміщення посадочних місць та спеціальних зон у закладах сфериHoReCa (ресторанах, кафе, барах), який має цілий арсенал спеціальних інструментів, що дозволяють створювати плани приміщення настільки складними, наскільки користувачу дозволяють його навички. Особливістю даного редактора є можливість його інтеграції з мобільними застосуваннями, що є вкрай важливим для реалізації багатьох онлайн-сервісів.

Список літератури

1. Vectorgraphicseditor [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Vector_graphics_editor
2. ProAndroidGraphics 1st EditionWallaceJackson – W.: «FloatingBridgePress», 2013
3. Android.graphics [Електронний ресурс]. – Режим доступу: https://developer.android.com/reference/android/graphics/package-summary
4. Graphicsarchitecture [Електронний ресурс]. – Режим доступу: https://source.android.com/devices/graphics/architecture
5. MatricesandTransformationsbyAnthony J. Pettofrezzo – W.: «DoverPublications», 1978

УДК 004.8

ГАСС Л. Е.,
ЗЛАТОКРИЛЕЦЬ М. О.,

ТЕХНОЛОГІЇ РОЗПІЗНАВАННЯПРОДУКТІВ ХАРЧУВАННЯ НА ЗОБРАЖЕННЯХ

У даній статті досліджено різноманітнізасоби та підходи до розпізнавання зображень на базі машинного навчання. Проаналізовано переваги і недоліки використання універсальних платформ для розпізнавання зображень та спеціалізованих моделей на базі TensorFlowв розрізі використання їх для вирішення задач розпізнавання в прикладних програмних додатках. Отримано результати досліджень тапроведено порівняння ефективності використання та ресурсозатратності обох підходів при їх використанні у розробці програмних продуктів.

This article explores the various to ols and approachesto machine-basedimage recognition. The advantages and disadvantages of using Tensor Flow-based universal imagerecognition platform sands pecialized models interms of using the mtosolve recognition problems in application software are analyzed. The results of the research were obtained and a comparison of the efficiency of use and the cost-effectiveness of both approaches in the iruseins of tware development.

Ключові слова: розпізнавання зображень, машинне навчання, TensorFlow.

1. Вступ

Сьогодні одним з найважливіших класів задач є розпізнавання речей та об'єктів на фото та відео. Зараз вже існує багато різноманітних засобів, котрі вже включають в себе десятки тисяч моделей розпізнаних об'єктів. Одним із таких є MLKitвід компанії Google, що пропонує набір даних з більш ніж 60-ма тисячами моделей класів. В той же час окремі компанії та ентузіасти створюють свої

нейронні мережі та TensorFlow-моделі, котрі власноруч тренують та отримують більш точні результати.

Однак результати отримані від готових рішень не є прийнятними. Тому актуальною залишається проблема підвищення точності результатів при розпізнаванні об’єктів зображень при вирішенні прикладних задач, в особливості при розпізнаванні продуктів харчування.

2. Огляд засобів розпізнавання зображень

Великі компанії, такі як Google, Amazon та Microsoft, проводять дослідження в галузі розпізнавання зображень в інтерфейсах прикладного програмування, щоб кожен розробник програмного забезпечення міг використовувати цю технологію в додатках. Вони пропонують наступні API для розпізнавання зображень: GoogleVision API, AmazonRekognition та Microsoft ComputerVision API.

Основні функції, необхідні для розпізнавання об'єктів, є у всіх трьох API.Такі функції, як позначення зображення, чітке виявлення вмісту, виявлення домінуючих кольорів та оптичне розпізнавання символів є найбільш релевантними при аналізі зображень. Крім того, GoogleVision може виявити подібні зображення в Інтернеті, які ми можемо використовувати для виявлення порушень авторських прав та виявлення джерел зображень продуктів харчування, знайдених в Інтернеті. Microsoft ComputerVision API - єдиний API серед трьох, що забезпечує функцію «відеотеги». Отже, якщо програма вимагає розпізнавання об'єктів та аналізу відео, API ComputerVision Microsoft буде накращим вибором.

Якщо наявний невеликий набір зображень (близько 100-150 на один клас), то можна натренувати власну мережу з використанням бібліотекиTensorFlow. TensorFlow – провідна платформа для машинного навчання з відкритим вихідним кодом, що розробляється компанією Google. Вона дозволяє тренувати та розгортати моделі глибокого навчання та традиційного машинного навчання і є однією з найпопулярніших платформ, особливо у мобільних додатках.

Однією з переваг платформи TensorFlow є те, що вона дозволяє тренувати вузькоспеціалізовану нейронну мережу на базі вже готової універсальної мережі нижчого рівня. Із використанням шарів нейронів, що відповідають за розпізнавання графічних примітивів та образів, тренується повністю лише найвищий шар нейронів, що здійснює розпізнавання вже готових об’єктів з конкретної предметної області. Таким чином, можна достатньо швидко і з використанням невеликої кількості ресурсів натренувати мережу, що буде достатньо ефективно розв’язувати покладені на неї задачі. Такий спосіб тренування називають трансферним навчанням.

При тренуванні нейронної мережі методом трансферного навчання,першим етапом є підготовка масиву зображень. Зазвичай достатньо близько 100 зображень для кожного класу об’єктів, що будуть розпізнаватися. Також необхідно підготувати попередньо натреновану модель, верхній шар якої буде перетреновано для конкретної задачі. Приклади таких моделей: MobileNet, Inceptionv1, Inceptionv3 та інші. Після цього починається вторинне тренування моделі: верхній шар нейронів перераховується для підібраного масиву зображень. Завдяки тому, що більша частина моделі вже є натренованою, цей процес не займає багато часу: не більше години на звичайному персональному комп’ютері.

Однією з найважливіших характеристик в процесі машинного навчання є точність натренованої моделі. Платформа TensorFlownадає наступні показники, які дозволяють оцінити якість моделі.

1. Тренувальнаточність – відсоток коректно розпізнаних зображень тренувального набору.

2. Контрольнаточність - відсоток коректно розпізнанихзображень контрольного набору.

3. Перехреснаентропія – величина, яка в данному випадку використовується для оцінки відмінності розподілів випадкових величин. Чим нижчим є цейпоказник, тимточнішою є модель.

За умови добре підібраних тренувального та контрольного наборів зображень в процесі тренування тренувальна точність та контрольна точність поступово підвищуються, а перехресна ентропія зменшується. Також зменшуватись повинна й різниця між тренувальною та контрольною

точністю. Так, модель можна вважати добре натренованою, якщо тренувальна та контрольна точність є більшими за 90%, різниця між ними не перевищує 5%, а перехресна ентропія є в межах 0,5.

3. Порівняння готових засобів розпізнавання зображень

Для застосування в прикладних задачах часто виникає необхідність вибору конкретного засобу, який би забезпечив найвищу ефективність. Далі наведено огляд того, які результати дають різні представлені на ринку APIрозпізнавання зображень при застосуванні їх на фотографіях продуктів харчування.

Дані API порівняно лише на основі їх функцій тегування зображень: для зображень на харчових продуктах це найважливіша особливість, оскільки вона допомагає автоматично класифікувати такі зображення. Нижче наведено результати перевірки вищевказаних API на деяких зображеннях їжі та аналіз тегів, які вони надають (табл. 1-2).

Табл.1. Розпізнавання продукту «Хліб»

API	Найімовірніші помітки
Google Vision API	Хліб(95,7%), їжа(94,3%), тарілка(90,8%), продукт(88,2%)
Amazon Rekognition	Їжа(93,2%), хліб(92,7%), тарілка(90,6%)
Microsoft Computer Vision API	Продукт(96,9%), хліб(95,8%), їжа(92,3%), тарілка(90,6%)

Табл.2. Розпізнавання продукту «Банан»

API	Найімовірніші помітки
Google Vision API	Банан(97,7%), їжа(95,2%), продукт(92,4%)
Amazon Rekognition	Їжа(95,2%), банан(93,7%), продукт(90,6%)
Microsoft Computer Vision API	Банан(92,5%), їжа(90,3%), продукт(88,6%)

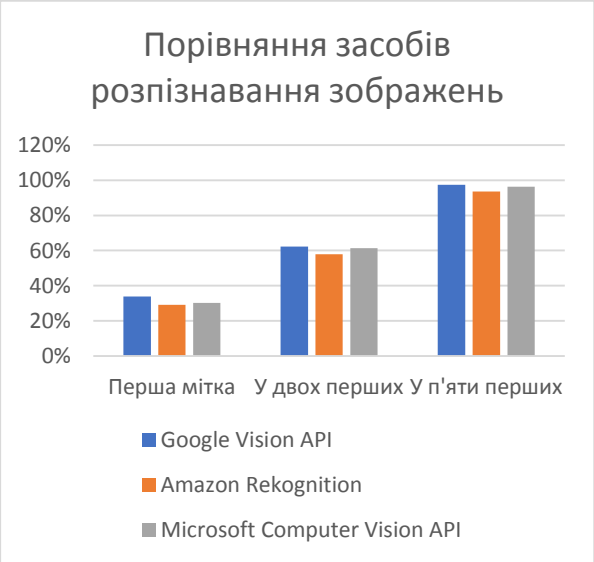


Рис. 1. Діаграма точності розпізнавання моделей

Для того, щоб оцінити точність кожної із моделей, було виконано розпізнавання 3000 продуктів харчуваннякожною з моделей.Зображення продуктів було взято з наборуTensorFlow «food101»[5].Було зафіксовано кількість випадків, коли вірна відповідь була вказана як найімовірніша, коли вірна відповідь містилась серед двох найімовірніших, а також коли вірна відповідь містилась серед п'яти найімовірніших. Результати наведено на графіку (рис. 1).

Таким чином, можна стверджувати, що всі три засоби мають приблизно однакову точність при розпізнаванні продуктів харчування на зображеннях. З незначним відставанням кращі результати дає модель GoogleVisionAPI.

5. Особливості застосування засобів розпізнавання зображень в задачах, пов'язаних з продуктами харчування

При розробці програмного забезпечення можнаприйняти рішення, яку платформу використовувати: універсальну добре натреновану модель, що підтримується сторонніми розробниками, чи власноруч натреновану мережу, спрямовану на вирішення вузького кола задач заданої предметної області.У порівнянні із власноруч розробленою моделлю, універсальні мережі сторонніх розробників не вимагають часу та ресурсів для тренування – їх можна одразу використовувати. В умовах обмежених ресурсів вибір такого підходу може бути єдиним способом додати у програму елементи машинного навчання. Крім того,

через те, що в універсальних моделях має місце підтримка великої кількості предметних областей, деякі класи можуть бути недостатньо деталізовані (наприклад, модель може підтримувати лише клас «собака», без деталізації на окремі породи). Якщо ж модель підтримує багато деталізованих класів, може мати місце недостатня точність моделі.

Однак при роботі з універсальними готовими рішеннями виникає ряд різноманітних труднощів. Одним з них є отримання недостовірних результатів. При рішенні задач, пов'язаних з розпізнаванням продуктів харчування ця проблема є занадто критичною, оскільки отримання вірного результату при роботі з зображеннями продуктів є дуже важливим. Тому виникає задача фільтрації результатів розпізнавання продуктів. При наявності переліку усіх продуктів харчування та результатів від готових засобів від Google, Amazonта Microsoftми отримуємо дві множини даних. Перетином цих множин і буде список продуктів харчування, що були розпізнані за допомогою зазначених вище засобів.

Висновки

В даній статті було проведено аналіз та порівняння існуючих платформ для розпізнавання об'єктів на зображеннях таїх ефективності при застосуванні в прикладних задачах. Крім того, було визначено, що в ході розпізнавання зображень універсальні мережі часто ставлять на перше місце невірну відповідь: відсоток вірних відповідей складає лише 25-35%.Однак серед п'яти найімовірніших варіантів вірна відповідь знаходиться дуже часто:в 93-97% випадків. Як показало дослідження, найкращі результати розпізнавання зображень продуктів харчування продемонструвала модель GoogleVision. Тому оптимальним рішенням єзастосуванняуніверсального засобу розпізнавання зображень GoogleVisionз використанням додаткової фільтрації отриманих результатів. Дану технологію можна успішно застосовувати в задачах розпізнавання продуктів харчування.

Список літератури

1. Умныемобильныепроекты с Tensorflow – М.: ДМК Пресс, 2019. – 384 с.: ил.
2. TensorFlow [Електронний ресурс]. – Режим доступу: https://www.tensorflow.org/learn
3. ML KitforFirebase[Електронний ресурс]. – Режим доступу: https://firebase.google.com/docs/ml-kit
4. Amazon Rekognition[Електронний ресурс]. – Режим доступу:
https://docs.aws.amazon.com/rekognition/latest/customlabels-dg/custom-labels-api-reference.html
5. Food101 [Електронний ресурс]. – Режим доступу:https://www.tensorflow.org/datasets/catalog/food101