Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра прикладної математики
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Математичні моделі та методи інтелектуального аналізу
повідомлень в соціальних мережах
(тема)

Виконав:
студент _2_ курсу, групи ПМм-18-1
Марченко М.С.
(прізвище, ініціали)
Спеціальність
113 Прикладна математика
(код і повна назва спеціальності)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма
Прикладна математика
(повна назва освітньої програми)
Керівник ___доц. Єсілевський В.С.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ПМ _____ ____Тевяшев А.Д._____
(підпис) (прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

Факультет _інформаційно-аналітичних технологій та менеджменту_____
Кафедра _прикладної математики_____
Рівень вищої освіти _другий (магістерський)_____
Спеціальність _113 Прикладна математика_____
(код і повна назва)
Тип програми _освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)
Освітня програма _Прикладна математика_____
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри ПМ _____
(підпис)
“_____” _____ 2019 р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Марченко Марії Сергіївні_____
(прізвище, ім’я, по батькові)

1. Тема роботи Математичні моделі та методи інтелектуального аналізу повідомлень в соціальних мережах_____
_____
затверджена наказом по університету від 31 жовтня 2019 р. № 1600 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 9 грудня 2019 р.____

3. Вихідні дані до роботи математична модель аналізу настроїв користувачів_____ соціальної мережі Twitter_____
_____
_____
_____
_____
_____
_____

4. Перелік питань, що потрібно опрацювати в роботі_____
1. Системний аналіз проблеми інтелектуального аналізу_____
повідомлень в соціальних мережах_____
2. Вибір і обгрунтування методу розв’язання_____
3. Програмна реалізація_____
4. Результати обчислювального експерименту_____
5. Аналіз можливих застосувань_____

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

1. Актуальність теми роботи
2. Постановка задачі
3. Системний аналіз проблеми
4. Метод чисельного аналізу
5. Результати обчислювального експерименту

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Підбір та вивчення технічної літератури за темою роботи	вересень 2019 р.	Виконано
2	Вибір та обґрунтування методу	жовтень – листопад 2019 р.	Виконано
3	Розробка алгоритму і програми	листопад – грудень 2019 р.	Виконано
4	Проведення аналітичних досліджень та розрахунків	листопад – грудень 2019 р.	Виконано
5	Робота над текстом пояснювальної записки	грудень 2019 р.	Виконано
6	Представлення роботи на рецензію в ЕК	грудень 2019 р.	Виконано

Дата видачі завдання 2 вересня 2019 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Єсілевський В.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 81 с., 38 рис., 13 табл., 18 джерел.

ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ, НАВЧАННЯ БЕЗ УЧИТЕЛЯ, МАШИННЕ НАВЧАННЯ, ЗГОРТКОВА НЕЙРОННА МЕРЕЖА, СОЦІАЛЬНА МЕРЕЖА, ВІДГУК, ТОНАЛЬНІСТЬ.

Об’єктом дослідження є задача інтелектуального аналізу повідомлень у соціальних мережах, яка полягає в пошуку оцінки настроїв на основі коментарів соціальної мережі Twitter.

Мета роботи – побудувати математичну модель, а також дослідити методи інтелектуального аналізу.

Методами дослідження стали – метод обробки текстових даних до векторного виду (Word2Vec) та згорткова нейронна мережа (CNN). Задача класифікації настроїв була вирішена за допомогою алгоритму Рандомного Лісу (RF).

В області класифікації настроїв аналізуються думки або настрої людей. Системи аналізу настроїв застосовуються в соціальних платформах і майже в кожному бізнесі, тому що думки або настрою відображають переконання, вибір і діяльність людей. За допомогою цих систем можна приймати рішення для бізнесу в політичних цілях. Останнім часом велика кількість людей діляться своєю думкою через Інтернет, тому в епоху інформації даний підхід до аналізу думки громадськості вимагає своєї уваги і є достатньо актуальним.

Програмна реалізація була здійснена за допомогою мови програмування Python за підтримки відкритої програмної бібліотеки для машинного навчання TensorFlow.

ABSTRACT

Introductory note: 81 pages, 38 figures, 13 tables, 18 sources.

INTELLECTUAL ANALYSIS, TEACHING WITHOUT TEACHER, MACHINE TRAINING, CONCURRENT NEURAL NETWORK, SOCIAL NETWORK, RESPONSE, TONALITY.

The object of the study is the task of intelligently analyzing messages on social networks, which is to find sentiment estimates based on comments from the social network Twitter.

The purpose of the work is to build a mathematical model and to explore methods of intellectual analysis.

Methods of study have become – the method of processing text data to vector-type (Word2Vec) and convolutional neural network (CNN). The problem of mood classification was solved using the Random Forest (RF) algorithm.

In the area of mood classification, people's thoughts or moods are analyzed. Mood analysis systems are used on social platforms and in almost every business, because thoughts or moods reflect people's beliefs, choices and activities. With these systems, you can make business decisions for political purposes. Recently, a large number of people have been sharing their opinions via the Internet, so in this age of information, this approach to public opinion analysis requires attention and is relevant.

The software was implemented using Python programming language with the support of the open TensorFlow machine learning library.

ЗМІСТ

ВСТУП

Суспільна думка сьогодні є важливим індикатором стану різних систем, оскільки відображає рівень соціальної напруженості. Облік і контроль цього рівня дозволяє вибудовувати стратегічне планування для забезпечення стійкого розвитку різних систем, будь то промислове підприємство, суб'єкт держави України або держава в цілому. У зв'язку з цим, моніторинг настроїв є важливим і актуальним інструментом управління, активно застосовуваним соціально-політичними, фінансово-економічними та громадськими структурами. Активне зростання аудиторії соціальних медіа в мережі Інтернет, таких як соціальні мережі, форуми, блоги та інтернет-ЗМІ, привело до становлення цих ресурсів в якості нового джерела даних про думку і настрої користувачів. Специфіка роботи з такими даними несе в собі цілий ряд переваг і недоліків. До переваг відноситься швидкість доступу до інформації, охоплення аудиторії і спектр висловлювання думок. Однією з головних переваг, як і серйозною перешкодою, є обсяг цих даних. Кількість людей, свідомо чи випадково поповнюючих Інтернет даними, стає дедалі більше. Вже створений колосальний масив даних.

Мільярди публікацій, що залишаються користувачами щомісячно, неможливо обробити вручну при проведенні дослідження суспільної думки. Цей факт висуває на перший план потребу в методах автоматизованого інтелектуального аналізу текстової інформації, що дозволяють за короткий проміжок часу обробити великі обсяги даних і зрозуміти сенс користувацьких повідомлень. Саме розуміння сенсу повідомлень є найбільш важливим і складним елементом автоматизованої обробки. Таким чином, актуальність даної роботи обумовлена необхідністю розвитку методологічного апарату, який дозволив би використовувати великі обсяги публікацій користувачів соціальних мереж для розв’язку комплексу завдань по автоматизації моніторингу настроїв користувачів.

1 СИСТЕМНИЙ АНАЛІЗ ПРОБЛЕМИ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ПОВІДОМЛЕНЬ У СОЦІАЛЬНИХ МЕРЕЖАХ

1.1 Системний аналіз проблеми інтелектуального аналізу повідомлень у соціальних мережах

1.1.1 Вербальна модель системи

Об'єктом дослідження є процес аналізу настроїв користувачів соціальних мереж.

Предметом дослідження є автоматизація аналізу загальної думки на основі застосування методів інтелектуального аналізу повідомлень у соціальних мережах.

Соціальні мережі є новим полем комунікативної взаємодії, нова форма публічності і нова форма колективного обговорення цікавлять певну групу людей індивідуальних, вузько групових і суспільних проблем. Характер публікацій настільки різноманітний, що соціологічні дослідження можуть мати абсолютно різний характер. Це робить соціальні мережі універсальним джерелом інформації.

Метою роботи є побудувати математичної моделі, а також дослідження методів інтелектуального аналізу.

Інтелектуальний аналіз даних являє собою процес виявлення придатних до використання відомостей в великих наборах даних. В інтелектуальному аналізі даних застосовується математичний аналіз для виявлення закономірностей і тенденцій, що існують в даних. Зазвичай такі закономірності не можна виявити при традиційному перегляді даних, оскільки зв'язки занадто складні, або через надмірний обсяг даних.

Ці закономірності і тренди можна зібрати разом і визначити як модель інтелектуального аналізу даних. Моделі інтелектуального аналізу даних можуть застосовуватися до конкретних сценаріїв, а саме:

– прогнозування: оцінка продажів, прогнозування навантаження серверу або часу простою серверу;

– ризик і ймовірність: вибір найбільш придатних замовників для цільової розсилки, визначення точки рівноваги для ризикованих сценаріїв, призначення ймовірностей діагнозами або інших результатів;

– рекомендації: визначення продуктів, які з високою часткою ймовірності можуть бути продані разом, створення рекомендацій;

– пошук послідовностей: аналіз вибору замовників під час здійснення покупок, прогнозування наступного можливого події;

– групування: поділ замовників або подій на кластери пов'язаних елементів, аналіз і прогнозування спільних рис.

Побудова моделі інтелектуального аналізу даних є частиною більш масштабного процесу, в який входять всі завдання, від формулювання питань щодо даних і створення моделі для відповідей на ці питання до розгортання моделі в робочому середовищі.

1.1.2 Морфологічний опис системи

Морфологічний опис задачі інтелектуального аналізу почнемо з розгляду поняття навколишнього середовища.

Навколишнє середовище – сукупність всіх об’єктів за межами границі системи, зміна властивостей яких мають вплив на систему, а також тих об’єктів, чиїх властивості змінюються в результаті поведінки системи.

Модель «чорний ящик» – модель досліджуваної системи, що зосереджена на досліджені реакції системи, як цілого, на зміни зовнішнього середовища. До факторів моделі відносяться моніторинг соціальної мережі та алгоритми машинного навчання, кожен з яких має фіксовану кількість можливих значень, що називаються рівнями. Від зміни рівня одного із входів залежить вихідний стан «чорного ящика», тобто результат роботи моделі – оцінка суспільної думки.

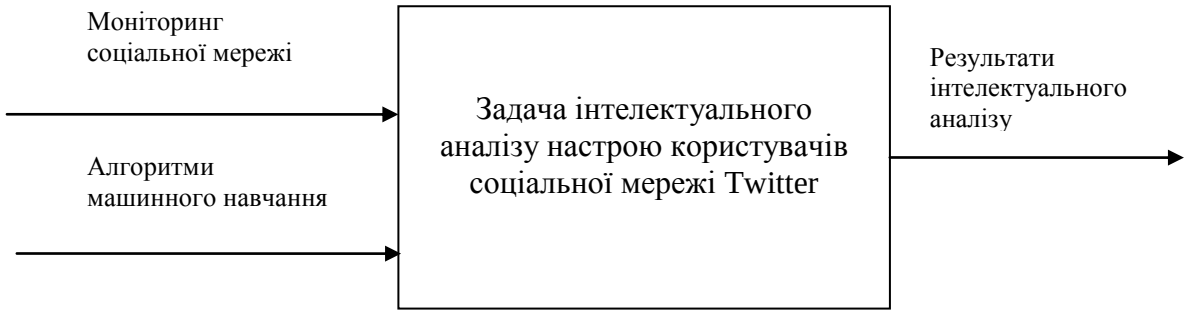

Рисунок 1.1 – Модель системи типу «чорний ящик»

1.1.3 Функціональна модель системи

Контекстна діаграма є вершиною деревовидної структури діаграм і являє собою саме загальний опис системи та її взаємодії з зовнішнім середовищем.

Функціональна модель системи може бути представлена графічно за допомогою контекстної діаграми IDEF0 на рисунках 1.1 – 1.2. У такому випадку система постає у вигляді взаємодіючих функцій, або, інакше кажучи, функціональних блоків.

Спочатку проводиться загальний опис системи. Після опису системи в цілому проводиться розбиття її на великі фрагменти. Цей процес називається функціональної декомпозицією, а діаграми, які описують кожен фрагмент і взаємодія фрагментів, називаються діаграмами декомпозиції.

Після декомпозиції контекстної діаграми проводиться декомпозиція кожного великого фрагмента системи на більш дрібні і так далі, до досягнення потрібного рівня подробиці опису.

Після того, як контекст описаний, проводиться побудова наступних діаграм в ієрархії. Кожна наступна діаграма є більш докладним описом однієї з робіт на указаній вище діаграмі.

Рисунок 1.3 являє собою декомпозицію минулої діаграми на 4 функціональні блоки, між якими послідовні прямі зв’язки. Оскільки задача потребує детального пояснення, то цей блок підлягає декомпозиції, створивши тим самим наступний рівень декомпозиції із 3 функціональними блоками.

IDEF3 є стандартом документування технологічних процесів, що відбуваються на підприємстві, і надає інструментарій для наочного дослідження і моделювання їх сценаріїв. IDEF3 широко застосовується при розробці інформаційних систем. При цьому використовується інструмент візуального моделювання бізнес-процесів. Система описується як упорядкована послідовність подій з одночасним описом об'єктів, що мають відношення до процесу, що моделюється.

Розглянемо опис роботи у стандарті IDEF3 на рисунках 1.6 – 1.8.

Рисунок 1.2 – Контекстна IDEF0 діаграма (рівень A-0)

Рисунок 1.3 – Декомпозиція роботи «Збір даних у соціальній мережі Twitter» (рівень A0)

Розглянемо на рисунках 1.4 – 1.6 діаграми заданих декомпозицій.

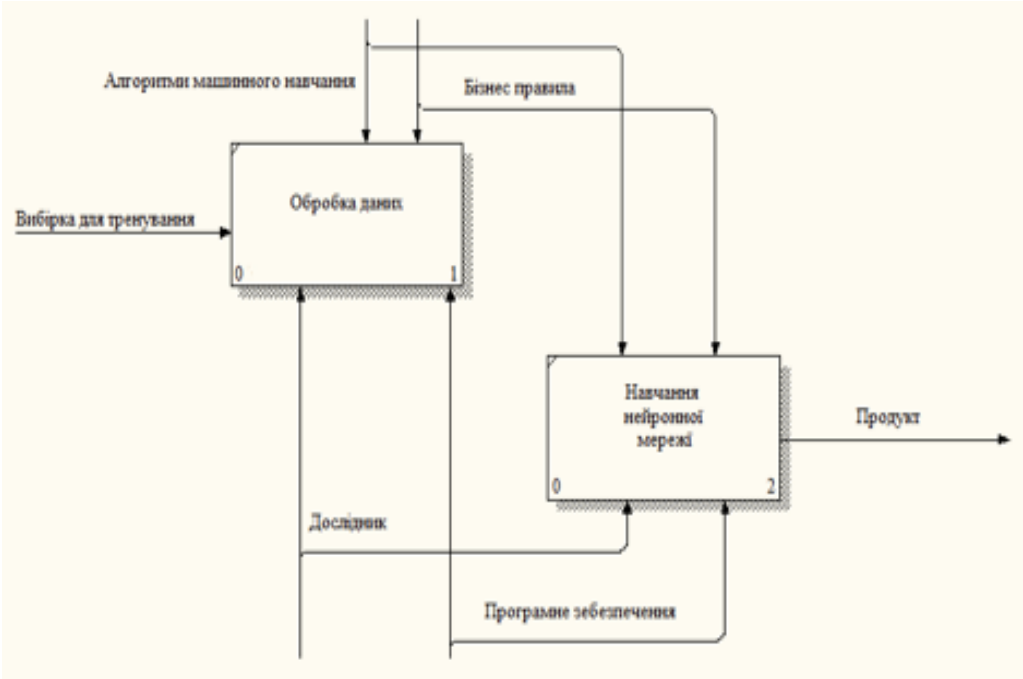

Рисунок 1.4 – Декомпозиція роботи «Написання програмного коду» (рівень А2)

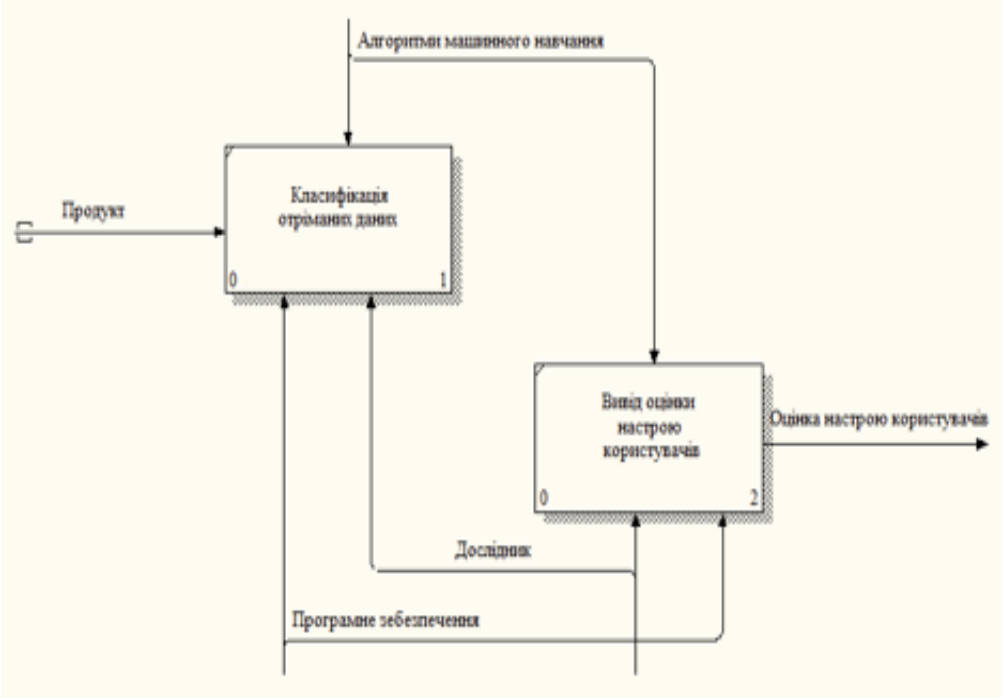

Рисунок 1.5 – Декомпозиція роботи «Обробка даних та вивід результату аналізу настроїв користувачів» (рівень А3)

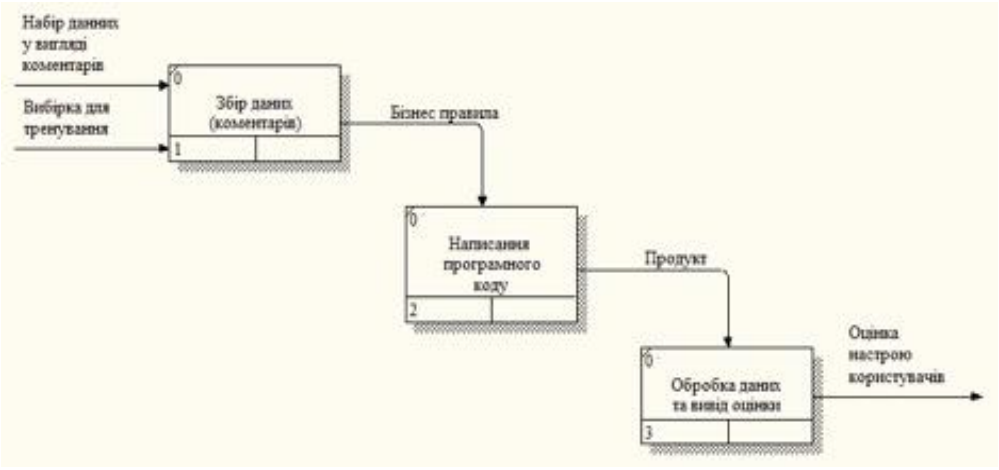

Рисунок 1.6 – Опис роботи «Збір даних у соціальній мережі Twitter» (рівень А0 у нотації IDEF3)

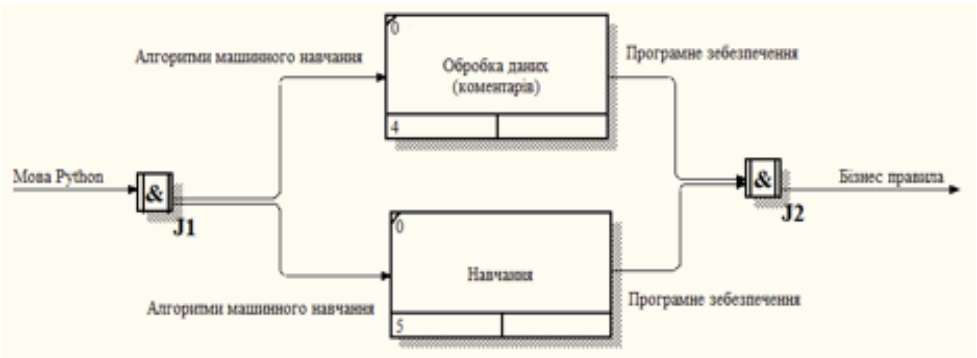

Рисунок 1.7 – Опис роботи «Написання програмного коду» (рівень А2 в нотації IDEF3)

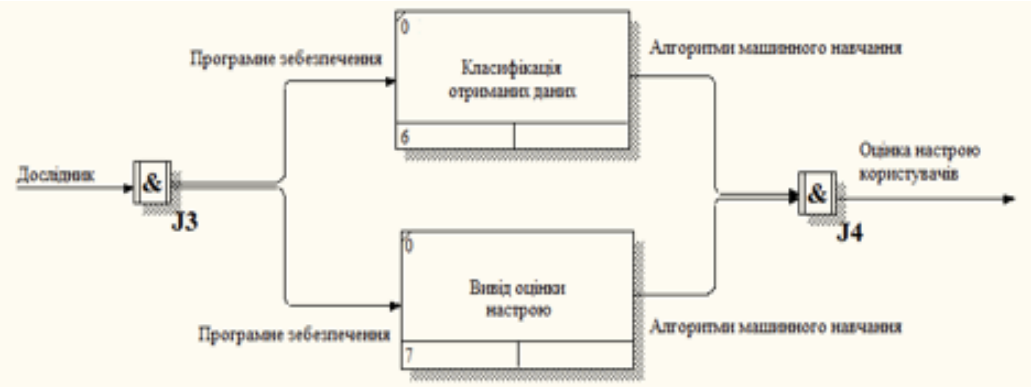

Рисунок 1.8 – Опис роботи «Обробка даних та вивід результату аналізу суспільної думки» (рівень А3 у нотації IDEF3)

1.1.4 Інформаційна модель

Інформаційні моделі відображають різні типи систем об'єктів, в яких реалізуються різні структури взаємодії і взаємозв'язку між елементами системи.

Інструменти методології DFD дозволяють відображати джерела і адресати даних, ідентифікувати процеси і групи даних, що зв'язують в потоки одну функцію з іншого, і ефективно використовуються для опису процесів при впровадженні процесного підходу до управління організацією, так як дозволяє максимально знизити суб'єктивність опису бізнес процесів. Крім того, нотація DFD дозволяє описувати потоки документів і потоки ресурсів.

Діаграма дерева вузлів показує ієрархію робіт в моделі і дозволяє розглянути всю модель цілком, але не показує взаємозв'язку між роботами.

Діаграма дерева вузлів для всіх вузлів моделі показана на рисунку 1.9.

Рисунок 1.9 – Діаграма дерева вузлів

1.2 Аналіз сценаріїв вирішення проблеми інтелектуального аналізу повідомлень у соціальних мережах

1.2.1 Модель аналізу проблеми

Перед тим, як проводити аналіз проблеми інтелектуального аналізу повідомлень у соціальних мережах, скористаємося методом аналізу ієрархій. У ролі критеріїв порівняння методів виступатимуть їх базові характеристики:

– точність результату аналізу (К1);

– складність методу та його програмної реалізації (К2);

– швидкість методу (К3);

– універсальність (К4);

– витрати на розрахунки (К5).

Множина, з якої буде прийняте рішення, складається із наступних альтернатив:

– штучний інтелект (А1);

– визначення області (А2);

– пошук за шаблоном (А3);

– обробка ознак (А4).

Таким чином, ієрархічна структура задачі складається із трьох рівнів:

– нульового – вибір методу розв’язання задачі математичного моделювання динамічних об’єктів;

– першого – критерії порівняння методів;

– другого – множина альтернатив.

Далі побудуємо матрицю попарних порівнянь для кожного рівня ієрархії, скориставшись відповідною шкалою Т. Сааті. Ця шкала надає можливість поставити у відповідність ступеням переваги одного порівняльного об’єкта над іншим деяке число. На основі цих даних знаходяться вектори локальних пріоритетів, індекси узгодженості та відношення узгодженості. Останні два допомагають визначити міру узгодженості результатів, отриманих експертним шляхом.

Допустимим вважається значення відношення узгодженості $\leq 0,2$. Більші значення свідчать про нелогічність суджень та спонукають до перегляду даних.

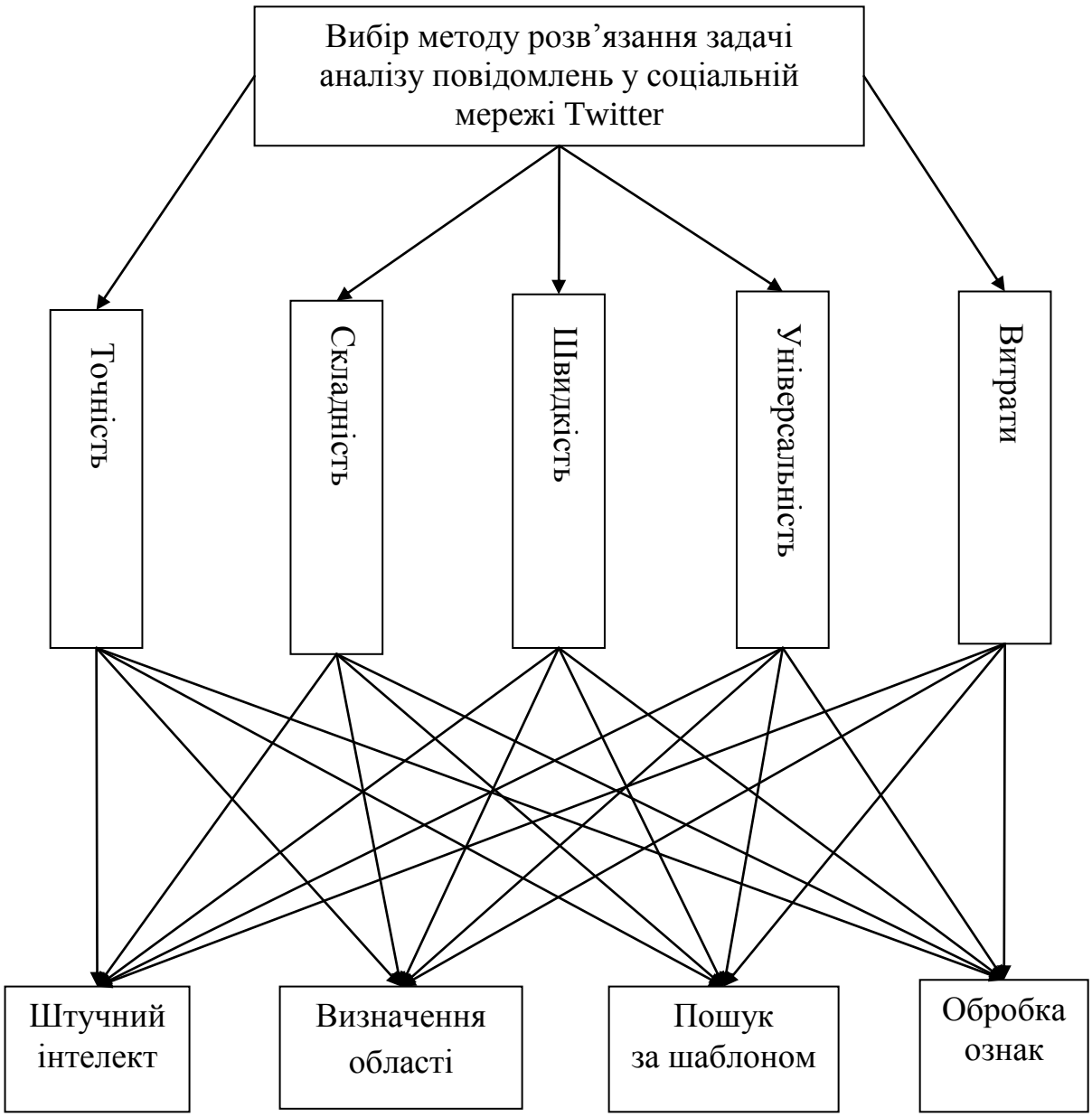

Рисунок 1.10 – Ієрархічна структура задачі вибору методу

$$\text{Індекс узгодженості (ІУ)} = \frac{5,461-5}{5-1} = 0,115.$$

$$\text{Відносна узгодженість (ВУ)} = \frac{0,115}{1,12} = 0,103.$$

Вектор локальних пріоритетів відносно проблеми вибору набуває вигляду:

$$\vec{p}^{K} = (0,406;\ 0,188;\ 0,295;\ 0,082;\ 0,029).$$

З отриманих проміжних результатів можна зробити висновок, що матриця попарних порівнянь заповнена правильно, тобто жодне з тверджень не суперечить загальній логіці моделі, а дані узгоджені між собою на допустимому рівні. Розглянемо матрицю попарних порівнянь першого рівня у вигляді таблиці 1.1.

Таблиця 1.1 – Матриця попарних порівнянь першого рівня

Номер п/п	К1	К2	К3	К4	К5	Власний вектор	Вектор пріоритетів
К1	1	3	2	5	7	2,914	0,406
К2	1/3	1	1/3	5	8	1,348	0,188
К3	1/2	3	1	4	7	2,112	0,295
К4	1/5	1/5	1/4	1	7	0,588	0,082
К5	1/7	1/8	1/7	1/7	1	0,205	0,029

Аналогічним способом необхідно провести порівняльний аналіз альтернатив відносно кожного з критеріїв. Таким чином ми отримаємо дані, що задовольняють індексу узгодженості та відносній узгодженості. Матриці попарних порівнянь представлені у вигляді таблиць 1.2 – 1.6.

Індекс узгодженості (ІУ) $= \dfrac{4,061-4}{4-1} = 0,02.$

Відносна узгодженість (ВУ) $= \dfrac{0,02}{0,9} = 0,022.$

Вектор локальних пріоритетів: $\vec{p}_1^{\,A} = (0,053;\ 0,12;\ 0,31;\ 0,517)$.

Таблиця 1.2 – Матриця попарних порівнянь першого критерію

К1	А1	А2	А3	А4	Власний вектор	Вектор пріоритетів
А1	1	$1/3$	$1/6$	$1/7$	0,298	0,053
А2	3	1	$1/3$	$1/5$	0,669	0,12
А3	6	3	1	$1/2$	1,732	0,31
А4	7	5	2	1	2,893	0,517

Індекс узгодженості (ІУ) $= \dfrac{4,157-4}{4-1} = 0,052$.

Відносна узгодженість (ВУ) $= \dfrac{0,052}{0,9} = 0,058$.

Вектор локальних пріоритетів: $\vec{p}_2^{\,A} = (0,525;\ 0,14;\ 0,291;\ 0,044)$.

Таблиця 1.3 – Матриця попарних порівнянь другого критерію

К2	А1	А2	А3	А4	Власний вектор	Вектор пріоритетів
А1	1	5	2	8	2,99	0,525
А2	$1/5$	1	$1/3$	6	0,795	0,14
А3	$1/2$	3	1	5	1,655	0,291
А4	$1/8$	$1/6$	$1/5$	1	0,254	0,044

Індекс узгодженості (ІУ) $= \dfrac{4,152-4}{4-1} = 0,051$.

Відносна узгодженість (ВУ) $= \frac{0,051}{0,9} = 0,056$.

Вектор локальних пріоритетів: $\vec{p}_3^A = (0,525;\ 0,138;\ 0,284;\ 0,053)$.

Таблиця 1.4 – Матриця попарних порівнянь третього критерію

К3	А1	А2	А3	А4	Власний вектор	Вектор пріоритетів
А1	1	5	2	7	2,893	0,525
А2	$1/5$	1	$1/3$	5	0,76	0,138
А3	$1/2$	3	1	4	1,565	0,284
А4	$1/7$	$1/5$	$1/4$	1	0,291	0,053

Індекс узгодженості (ІУ) $= \frac{4,152-4}{4-1} = 0,051$.

Відносна узгодженість (ВУ) $= \frac{0,051}{0,9} = 0,056$.

Вектор локальних пріоритетів: $\vec{p}_3^A = (0,525;\ 0,138;\ 0,284;\ 0,053)$.

Таблиця 1.5 – Матриця попарних порівнянь четвертого критерію

К4	А1	А2	А3	А4	Власний вектор	Вектор пріоритетів
А1	1	3	2	5	2,34	0,477
А2	6	1	2	4	1,278	0,319
А3	7	2	1	3	0,931	0,19
А4	$1/5$	$1/2$	$1/2$	1	0,359	0,074

Індекс узгодженості (ІУ) $= \dfrac{4,064-4}{4-1} = 0,021.$

Відносна узгодженість (ВУ) $= \dfrac{0,021}{0,9} = 0,024.$

Вектор локальних пріоритетів: $\vec{p}_4^{\,A} = (0,058;\ 0,307;\ 0,452;\ 0,183).$

Таблиця 1.6 – Матриця попарних порівнянь п’ятого критерію

К5	А1	А2	А3	А4	Власний вектор	Вектор пріоритетів
А1	1	$1/6$	$1/7$	$1/3$	0,298	0,058
А2	$1/3$	1	$1/2$	2	1,565	0,307
А3	$1/2$	$1/2$	1	2	2,3	0,452
А4	3	$1/4$	$1/3$	1	0,931	0,183

Індекс узгодженості (ІУ) $= \dfrac{4,447-4}{4-1} = 0,149.$

Відносна узгодженість (ВУ) $= \dfrac{0,149}{0,9} = 0,166.$

Вектор локальних пріоритетів: $\vec{p}_5^{\,A} = (0,477;\ 0,319;\ 0,19;\ 0,074).$

На основі отриманих результатів розрахуємо вектор глобальних пріоритетів та відповідні показники узгодженості. Зазначимо, що спосіб розрахунку індексу узгодженості для всієї ієрархії відрізняється від попередніх випадків – це сума індексу для першого рівня ієрархії та скалярного добутку вектора локальних пріоритетів критеріїв з вектором індексів узгодженості відповідного критерію.

Таблиця 1.7 – Кінцеві результати задачі вибору методу розв’язання

Критерії / Альтернативи	К1	К2	К3	К4	К5	Вектор пріоритетів
А1	0,053	0,525	0,525	0,058	0,477	0,293
А2	0,12	0,14	0,138	0,307	0,319	0,15
А3	0,31	0,291	0,284	0,452	0,19	0,307
А4	0,517	0,044	0,053	0,183	0,074	0,25

Індекс узгодженості (ІУ) $= 0{,}154$.

Відносна узгодженість (ВУ) $= \dfrac{0{,}154}{1{,}12 + 0{,}9} = 0{,}076$.

Як бачимо з таблиці 1.7 усі дані відповідають нормам узгодженості, а максимальна компонента вектора глобальних пріоритетів відповідає третій альтернативі, тобто спосіб, яким буде розв’язуватися задача, належить до навчання класифікатора.

1.2.2 Оцінювання вектора пріоритетів незадоволень методом аналізу ієрархій

Для продовження аналізу проблеми, зокрема оцінювання глобального вектора пріоритетів, необхідно побудувати матрицю попарних порівнянь першого рівня, а також і кожної виділеної властивості. Усі матричні дані наведені у відповідних таблицях 1.8 – 1.11.

Упевнившись, що табличні дані не суперечать один одному (ІУ=0,069, ВУ=0,06), знайдемо значення вектора глобальних пріоритетів для кожної властивості і зобразимо їх у вигляді стовпчастої діаграми (рисунок 1.11).

Таблиця 1.8 – Матриця попарних порівнянь першого рівня

Номер п/п	Бажані властивості	Критичні властивості	Небажані властивості	Власний вектор	Вектор пріоритетів
Бажані властивості	1	5	3	2,466	0,631
Критичні властивості	$1/5$	1	$1/4$	0,368	0,094
Небажані властивості	$1/3$	4	1	1,075	0,275

Таблиця 1.9 – Матриця попарних порівнянь бажаних властивостей

Бажані властивості	Точність	Швидкість	Надійність	Власний вектор	Вектор пріоритетів
Точність	1	2	5	2,154	0,57
Швидкість	$1/2$	1	4	1,26	0,333
Надійність	$1/5$	$1/4$	1	0,368	0,097

Таблиця 1.10 – Матриця попарних порівнянь критичних властивостей

Критичні властивості	Складність	Багато-функціональність	Власний вектор	Вектор пріоритетів
Складність	1	1/4	0,5	0,2
Багато-функціональність	4	1	2	0,8

Таблиця 1.11 – Матриця попарних порівнянь небажаних властивостей

Небажані властивості	Границі	Похибка	Витрати	Власний вектор	Вектор пріоритетів
Границі	1	4	1/6	0,322	0,075
Похибка	1/4	1	1/5	1,077	0,252
Витрати	6	5	1	2,885	0,673

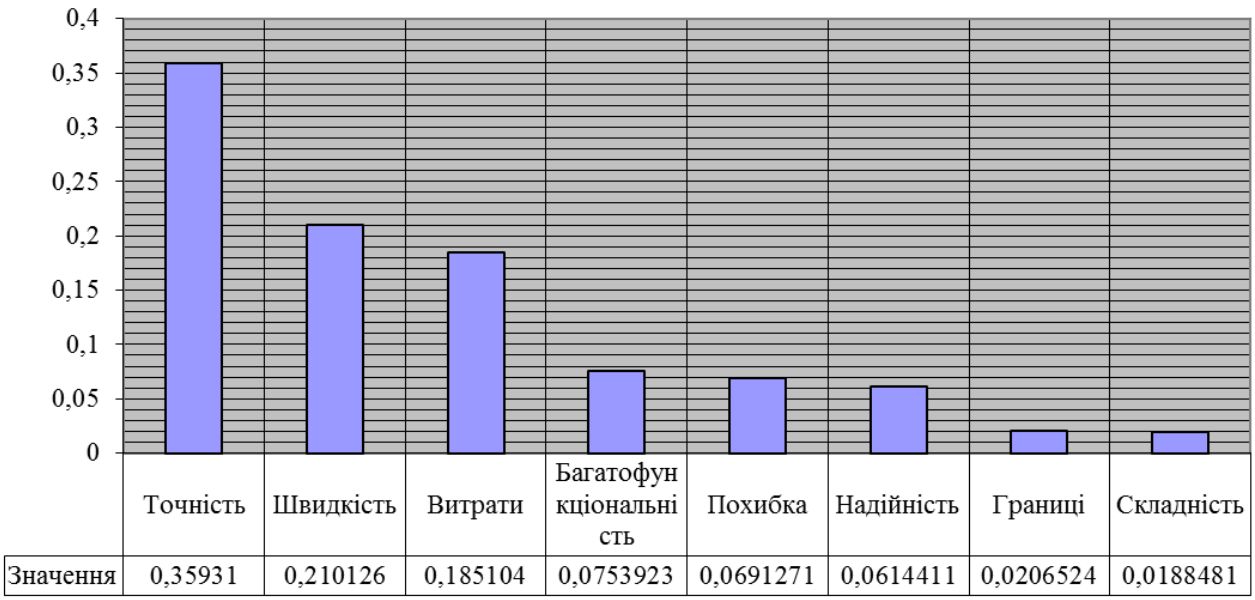

Рисунок 1.11 – Діаграма глобальних пріоритетів

На основі отриманих результатів зробимо висновок, що найвпливовішою трійкою властивостей є збільшення точності результатів, збільшення швидкості обробки даних та проблеми побудови границь області моделювання. У зв’язку із цим зазначимо, що основна увага вирішення поставленої проблеми буде зосереджена на трійці властивостей. Оскільки більша доля припадає на бажані властивості, то і зусилля відповідною мірою будуть направлені на покращення показників цих властивостей. Проте не слід забувати про наявність небажаної складової – певна частина ресурсів буде виділена на усунення великих обчислювальних витрат при моделюванні.

1.3 Змістовна та формальна постановка задачі

1.3.1 Змістовна постановка задачі

Моніторинг соціальних мереж полягає в безперервному відстеження користувальницьких публікацій. Різні онлайнові проекти по-різному організують механізм обміну інформацією між своїми користувачами, але в будь-якому з них дії користувача можна розділити на приватні та публічні. Відстеження доступні дії, носять публічний характер. Прикладом публічних дій є розміщення загальнодоступних повідомлень або залишення коментаря до загальнодоступного джерела. Приватні дії, як наприклад, особисте листування між користувачами, не доступні для моніторингу і тому розгляду не підлягають. Формат переданої користувачами соціальних мереж інформації може бути різним: аудіо, відео, текстові повідомлення, зображення, прикріплені документи різноманітних форматів. Тут і далі в рамках процесу моніторингу соціальних мереж мова буде йти лише про передачу загальнодоступних текстових повідомлень, до яких відносяться публікації, коментарі, твіти / ретвіти (короткі публікації мережі Twitter), статуси, замітки та ін.

Розглянемо задачу аналізу настроїв інтернет-користувачів шляхом відстеження користувальницьких публікацій в соціальних мережах. Як уже зазнача-

лося, ключову роль в цьому процесі відіграє автоматизований інтелектуальний аналіз текстової інформації, заснований на методах глибокого навчання.

Термін «глибоке навчання» був введений в 2006 році і відноситься до алгоритмів машинного навчання, які мають кілька нелінійних рівнів і можуть вивчати ієрархії функцій [1].

Більшість сучасних машинних навчань для отримання хороших результатів спираються на особливості проектування або деякий рівень знань в предметній області. У системах глибокого навчання це не так – замість цього алгоритми можуть автоматично вивчати ієрархії об'єктів, які представляють об'єкти на зростаючих рівнях абстракції. Хоча основні компоненти багатьох алгоритмів глибокого навчання існують вже багато років, в даний час їх популярність зростає з багатьох причин, включаючи досягнення в області обчислювальної потужності, падіння вартості обчислювального обладнання і досягнення в галузі досліджень в області машинного навчання.

Алгоритми глибокого навчання можуть бути класифіковані за їх архітектурі (пряма, зворотна або двонаправленная) і протоколам навчання (чисто контрольований, гібридний або неконтрольований) [2].

Будучи чітко формалізованими математичними моделями, алгоритми глибокого навчання не працюють з текстовою інформацією безпосередньо. Тому нам потрібно представити вхідний текст у вигляді вектора з чисел.

Труднощі аналізу тональності полягають в присутності емоційно збагаченої мови – сленг, багатозначність, невизначеність, сарказм, всі ці чинники вводять в оману не тільки людей, а й комп'ютерів. Слід обрати методологію, яка забезпечить перетворення слів у числовий формат, та дасть змогу зберегти їх семантику для більш точного виявлення настрою.

Тоді вхідними даними для моделей будуть набори числових ознак, які в ході набору нелінійних перетворень дадуть результат у вигляді класифікації.

Для більшої точності виявлення настрою також необхідна попередня обробка тексту. На цій стадії видаляються всі html теги, пунктуації, символи. Також всі числа і посилання в тексті замінюються на теги. Далі в тексті присутні

так звані «стоп слова» – це часті слова в мові, які в основному не несуть ніякої смислове навантаження (наприклад, в англійській мові це такі слова як «the, at, about …»). Та багато інших перетворень. Такий набір методів для вирішення текстових задач називається задачею обробки природної мови (Natural Language Processing).

Таким чином, перши завдання стає задача NLP.

Другим завданням є визначення способів перетворення текстової інформації у векторний вигляд.

Третім завданням є дослідження алгоритмів глибокого навчання, ефективних при обробці текстів на природній мові.

Четвертим, найголовнішим завданням, є складання загальної математичної моделі, що дозволяє вирішувати задачу моніторингу настрою громадськості на основі коментарів у соціальних мережах, конкретно мережі Twitter.

1.3.2 Формальна постановка задачі

Формальна постановка задачі аналізу настрою має вигляд задачі класифікації машинного навчання з учителем.

Дано: N – кінцева множина класів тональності. Так як можливих настроїв 2: «позитивний» і «негативний», тому $N = 2$. Також нам дана тренувальна вибірка спеціально відібрана для аналізу тональності – набір відгуків, кожний з яких складається з наступних полів: id (ID) – унікальний ідентифікатор кожного відгука, s (Sentiment) – тональність відгука, r (Review) – текст відгука. Позначимо вибірку як X^{train}. Класифікатор має наступний вигляд

$$a(x) = \frac{1}{N}\sum_{i=1}^{N} b_i(x).$$

Необхідно:

– для кожного $n = 1,...,N$ побудувати дерево b_n по вибірці X^{train};

– по заданому критерію ми вибираємо кращу ознаку, робимо розбиття в дереві по ньому і так до вичерпання вибірки;

– дерево будується, поки в кожному «листочку» не більше $n_{\min}$ об'єктів або поки не досягнемо певної висоти дерева;

– при кожному розбитті спочатку вибирається m випадкових ознак з n вихідних, і оптимальний розподіл вибірки шукається тільки серед них.

Для задачі класифікації рекомендовано брати $m = \sqrt{n}$, де n – число ознак.

Метрика тональності є двоїчною величиною, тому негативний настрій буде мати значення 0, а позитивний – 1.

1.4 Постановка задач дослідження

Соціальні мережі є новим полем комунікативної взаємодії, нова форма публічності і нова форма колективного обговорення цікавлять певну групу людей індивідуальних, вузько групових і суспільних проблем.

Характер публікацій настільки різноманітний, що соціологічні дослідження можуть мати абсолютно різний характер. Це робить соціальні мережі універсальним джерелом інформації.

Аналізуючи події в Україні кінця 2013 року, в роботі [2] дослідники виявляли ступінь кореляції активності протестно налаштованого населення в онлайн і оффлайн просторах. Показано, що дискурс користувачів в соціальних мережах є прямим відображенням подій реального світу і розвивається відповідно до динаміки протесту безпосередньо «на вулицях», тобто ніж гостріше і активніше проходять дії, тим активніше пишуть про це на просторах соціальних мереж. Автори роблять висновок, що моніторинг соціальних мереж дозволяє робити зрізи суспільних настроїв безпосередньо від учасників подій і проводити аналіз в короткі терміни.

Прикладом дослідних організацій, що використовують соціальні мережі в своїй діяльності, є дослідницький відділ компанії Facebook.

У 2014-му році його співробітники провели масштабний експеримент з управління емоціями користувачів. Для 690 тисяч чоловік навмисно ховалися позитивні або негативні публікації. Це призводило до того, що стрічка повідомлень конкретного користувача полягала або повністю з позитивних повідомлень, або повністю з негативних. Як результат, самі користувачі стали залишати тільки позитивні або негативні повідомлення відповідно, тим самим переймаючи настрій інших користувачів через їх публікації [3].

Однак, на додаток до цього, існує ряд нюансів з моніторингу інформації, наприклад, обмеження технічного характеру і специфіка політики конфіденційності. В цілому, беручи до уваги розглянуті тенденції соціологічних досліджень і спектр прикладних експериментів, можна зробити висновок, що відстеження публікацій в соціальних мережах є сьогодні передовим інструментом отримання інформації про настрій користувачів, тобто емоційне забарвлення тексту. Відкритим залишається питання про те, як саме аналізувати інформацію, отриману в ході моніторингу соціальних медіа.

Завдання аналізу емоційного забарвлення текстів зводиться до задачі класифікації. У нашому випадку є набір твітів, кожен з яких потрібно віднести до однієї з трьох категорій: позитивні, нейтральні або негативні.

Іноді класифікація відбувається в два етапи і на обох етапах є бінарної. На

першому відокремлюються суб'єктивні повідомлення від об'єктивних. Об'єктивними в цьому випадку називаються якраз ті, які не несуть емоційного забарвлення і є нейтральними у варіанті з трьома класами. Другий етап ділить суб'єктивні тексти на позитивні і негативні. У випадку з Twitter, де майже всі повідомлення суб'єктивні, а критерії нейтральності можна сформулювати тільки в сенсі «не позитивне» і «не негативна», будемо для простоти розглядати поділ на два класи.

Твіт – це рядок, що складається з не більше ніж 140 символів. Вона може містити спеціальні слова, що починаються з певних знаків: відразу після «@» пишеться ім'я користувача, з яким повідомлення пов'язано або до якої воно звернено, а після «#» знаходиться так званий хештег – слово, яке явно вказує на

зв'язок твіти з об'єктом, який цим словом позначається. Всі твіти створюються користувачами, тому можуть містити помилки, скорочення, особливу пунктуацію і інші способи вираження думки в короткому тексті. У кожного повідомлення в Twitter є час, коли воно опубліковано, і автор. Якщо один твіт є відповіддю на інший, то у першого є ще й посилання на другий, тобто на «батьківський». Ретвіти містять також дані про початковий розміщенні.

Необхідною частиною створення алгоритму аналізу настрою (або будь-якого алгоритму обробки даних) є наявність всеохоплюючого набору чи набору для вивчення, а також тестовий набір даних, який гарантує, що точність алгоритму відповідає стандартам, які ви очікуєте. Це також дозволить вам налаштувати алгоритм і вивести кращі (або більш точні) функції природної мови, які можна витягнути з тексту, що сприятиме посиленню класифікації почуттів, а не використовувати загальний підхід до «слів суті».

2 ВИБІР ТА ОБҐРУНТУВАННЯ МЕТОДУ РОЗВ'ЯЗАННЯ

2.1 Дослідження предметної області

2.1.1 Роль суспільної думки в управліннях систем

Г.І. Козирьов надає наступне визначення громадської думки: «Общественное мнение – это проявление массового сознания, выражающее ценностно-оценочные суждения социальных групп, отдельных общностей и всего общества по отношению к актуальным проблемам и явлениям социальной действительности. Оно может проявляться в виде суждений, оценок, идей, требований, представлений о проблеме, имеющей социальную значимость» [4]. Також наголошується, що громадська думка інформує владу і суспільство про існуючі проблеми і ініціює їх рішення.

Як можна бачити, збір інформації про громадську думку виробляється за допомогою процедури опитування громадян. До 80% всієї емпіричної інформації соціологи отримують методом опитування [5, с. 87]. Розглянутий приклад структури організації проведення збору та аналізу інформації про громадську думку показує, що цей процес має чимало витрат (як організаційного, так і фінансового характеру). Для зниження зазначених витрат, а також отримання ряду інших виграшів в надійності та валідності даних, методи соціологічних досліджень адаптують під інтернет-середовище [6]. Проте, наголошується ряд проблем при проведенні онлайн-досліджень. Ключовим фактором успішного онлайн-опитування є ефективність вибірки. Анонімність користувачів мережі Інтернет розглядається як причина низької ефективності вибірки через неможливість гарантованої ідентифікації користувача.

Варто окремо підкреслити той факт, що людям важливо донести свою думку, внести свій вклад, бути частиною прийняття рішень соціальних проблем. Тут відбувається стик двох потреб. Перша з них – потреба людей, відповідальних за стратегічне управління регіоном, в моніторингу громадської думки. Дру-

га – потреба громадян у виявленні своєї думки, своєї позиції з тих чи інших питань (в тому числі соціальних). Розглянуті раніше методики проведення соціологічних опитувань є хорошим кандидатом на роль однозначного способу задоволення обох потреб, однак, дивлячись на їхні недоліки (витрати «реальних» опитувань і втрата мотивації респондентів в «віртуальних»), виникла необхідність пошуку альтернативних рішень.

На допомогу соціологам прийшли соціальні медіа в мережі інтернет (форуми, блоги, мікроблоги, соціальні мережі). Платформи, спочатку призначені для спілкування інтернет-користувачів між собою досить швидко стали предметом інтересу багатьох дослідників зважаючи на велику кількість інформації, що знаходиться у відкритому доступі.

Зіставляючи між собою потенційну кількість респондентів, можна визначити різницю в кілька порядків між користувачами онлайн-панелей проти мільярдів користувачів соціальних мереж, які вже мають готові до аналізу профілі.

Таким чином, соціальні мережі, через широту охоплення аудиторії і доступності комунікації з респондентами, мають підстави вважатися перспективним джерелом даних для моніторингу та аналізу громадської думки, який, як зазначалося раніше, має велике значення при стратегічному управлінні.

2.1.2 Проблеми та методи аналізу повідомлень

Обробка гігантських об’ємів інформації неможлива вручну, тому отримання цінних знань через автоматизований аналіз великих об’ємів даних сьогодні є вкрай популярним напрямком і тісно пов'язане з такими термінами, як Big Data («великі дані») і Data Mining (в українській мові немає усталеного перекладу для даного терміну, але прийнято розуміти його як «видобуток знань з даних» або «інтелектуальний аналіз даних»).

Слід детально зупинитися на тому, які саме способи роботи з великими даними використовуються сьогодні, в чому їх особливості, переваги та недолі-

ки. Нижче представлені приклади методів, що застосовуються для вирішення завдань, пов'язаних з аналізом великих даних в соціальних медіа:

– підрахунок кількості лайків, репостів, передплатників і інших кількісних характеристик агентів соціальних мереж і їх публікацій;

– підрахунок згадок ключових слів з подальшою угрупованням;

– контент-аналіз користувальницьких повідомлень: класифікація за макро-темами, класифікація по емоційному тону;

– кластеризація публікацій згідно з тематикою.

Завдання інтелектуального аналізу – визначити суб'єкт обговорення (хто говорить), об'єкт (про що говорить) і відношення першого до другого (що саме говорить). Соціальні мережі спрощують рішення перших двох завдань: суб'єктом обговорення є автор публікації, а визначити об'єкт часто, хоча не завжди, допомагає наявність хештегів або тема оригінальної публікації, до якої залишено коментар. Кластеризація або класифікація за заздалегідь виявлених темам також відносяться до задачі визначення об'єкта обговорення.

Так як проблема визначення настрою громадськості через соціальні мережі дуже актуальна на сьогоднішній день, то аналіз текстової інформації уже представлений в багатьох сучасних програмних продуктах, присвячених моніторингу соціальних медіа, наприклад, таких як Brand Analytics, IQBuzz, YouScan. Як правило, емоційної оцінки буває достатньо для базового розуміння ставлення користувача до об’єкту дискусії. А через порівняння простоти і розвинених алгоритмів, даний вид аналізу є найбільш популярним в системах моніторингу соціальних медіа.

Також однією з проблем стає обробка тексту, перетворення його у числовий вигляд. При цьому необхідно забезпечити якомога точне збереження семантичності слів, тобто їх емоційне та смислове навантаження. Слід враховувати, що числа, emoji, символьні смайли («:=)», «XD», «:(» тощо), символьне емоційне передання («!!!!» – наполегливе ствердження, «!?!?!» – здивування) також вимагають обробки та врахування, тому що несуть важливу роль в аналізі настрою.

Підходи, які застосовуються для аналізу текстової інформації на природній мові, поділяють на дві основні групи: інженерно-лінгвістичні методи і методи на основі машинного навчання.

Інженерно-лінгвістичні методи використовують спеціальні, попередньо підготовлені експертами-лінгвістами тональні словники і/або лінгвістичні правила, на основі яких відбувається аналіз текстового фрагмента. Методи машинного навчання, список яких включає в себе (але не обмежується) метод умовних випадкових полів (Condition Random Fields, CRF), метод опорних векторів (Support Vector Machine, SVM), штучні нейронні мережі (Artificial Neural Networks, ANN) та інші. Ця група методів використовує математичні моделі, що дозволяють автоматично визначити оптимальний набір параметрів для вирішення конкретного завдання.

Варто відзначити наявність комбінованих (гібридних) методів, що включають в себе як інженерно-лінгвістичні елементи, так і машинне навчання (рисунок 2.1).

Рисунок 2.1 – Основні групи методів аналізу текстовою інформації

Розглядаючи разом моніторинг соціальних медіа (не тільки для оцінки емоційного забарвлення думок користувачів) і завдання автоматизованого інтелектуального аналізу текстових публікацій, можна стверджувати, що на сьогоднішній день це одне з найбільш актуальних і активно розвиваються прикладних напрямків соціології та комп'ютерної лінгвістики.

2.2 Дослідження методів вирішення задачі аналізу настрою

2.2.1 Представлення текстової інформації у векторному вигляді

Необхідно представити будь-який текст у вигляді вектора з чисел, для цього можна скласти словник з усіма словами, тобто об'єднати всі слова, що зустрічаються у тексті, в один великий словник, або ж використовувати готові словники (Даля або Залізняка), і замінити слова з тексту індексом в словнику. Тобто припустимо ми маємо всього три відгуки з наступними попередньо-обробленими векторами слів:

– [biography, part, feature];

– [film, remember, going];

– [see, cinema, originally].

Об'єднуючи всі слова зі списку в один, ми отримаємо наступний відсортований словник (назвемо його як базис вектор):

[biography, cinema, feature, film, going, originally, part, remember, see].

Замінюючи попередні вектора на індекс слова в словнику, отримуємо наступне:

– [1, 0, 1, 0, 0, 0, 1, 0, 0];

– [0, 0, 0, 1, 1, 0, 0, 1, 0];

– [0, 1, 0, 0, 1, 0, 0, 0, 1].

Проробивши таку роботу для всіх відгуків, ми можемо отримати досить великий список (в нашому прикладі 5000 найчастіше зустрічающихся слів). Ці вектори називаються «векторами властивостей» або ж «features vector». Таким чином ми отримуємо вектора для кожного тестового відгуку, далі ми зможемо порівнювати ці вектора за допомогою стандартних метрик, таких як Евклідову відстань, косинусну відстань тощо. Даний підхід називається «мішок слів» або ж «Bag-Of-Words».

Такий підхід є досить поширеним методом і досить простий в реалізацій, але він носить в собі достатньо недоліків. При порівнянні двох векторів використовується точний збіг слів, при цьому ми втрачаємо важливу інформацію. Однією з таких «втрачених» інформацій є семантика слова. Наприклад, ми з легкістю можемо замінити слово «чорний» словом «темний», так як їх зміст дуже схожий. Такі слова можна назвати семантично схожими словами. До групи таких слів входять синоніми, гіпоніми, гіпероніми тощо.

Суть моделі Word2Vec полягає в наступному – на вхід дається великий обсяг тексту (приблизно 10000 відгуків), на виході ми отримуємо зважений вектор для кожного слова, фіксованої довжини, яка зустрічається в датасеті. Наприклад, для слова men, порівнюючи з усіма словами і сортуючи в порядку спадання, отримаємо наступний результат, представлений у таблиці 2.1.

Таблиця 2.1 – Семантично близьких слів до слова «man»

Words	Measures
woman	0,6056
guy	0,4935
boy	0,4893
men	0,4632
person	0,4574
lady	0,4487
himself	0,4288
girl	0,4166
his	0,3853
he	0,3829

Довжина вектора, фіксована довжина слова, задається вручну. За міру близькості обрана косинусна відстань.

В альтернативному підході ми спробуємо замінити кожне слово в списку номером його семантичної групи. У підсумку ми отримаємо щось на кшталт «мішка слів» (Bag-Of-Words), але з більш глибоким змістом, ніж у розглянутомі підході вирішення задачі вище. Для цього використовується технологія Word2Vec від Google. Її можна знайти в пакеті бібліотеки gensim, з вбудованими моделями Word2Vec. Принципи роботи, переваги і алгоритм розглядаються у розділах нижче.

2.2.1.1 Переваги Word2Vec перед іншими підходами

Спростуємо міркування чому технологія Word2Vec краща за Bag-Of-Words.

По-перше, навчання Word2Vec більшої кількості тексту повинно значно підвищити продуктивність. Результати Google засновані на векторах слів, які були вилучені з більш ніж мільярда слів. Зручно, Word2Vec надає функції для завантаження будь-якої заздалегідь навченої моделі, яка виводиться вихідним інструментом Google на C, тому також можливо навчити модель на C і потім імпортувати її в Python.

По-друге, в існуючих прикладах застосування Word2Vec було показано, що методи розподілених векторів слів перевершують моделі Bag-Of-Words. У цій роботі використовується набір даних з соціальної мережі Twitter для отримання найсучасніших результатів на сьогоднішній день.

2.2.1.2 Принципи роботи Word2Vec

Word2Vec, опублікований Google в 2013 році [8], являє собою реалізацію нейронної мережі, яка вивчає розподілені уявлення для слів. Інші глибокі або

рекурентні архітектури нейронних мереж були запропоновані для вивчення уявлень слів до цього, але головною проблемою з ними було тривалий час, необхідний для навчання моделей. Word2Vec швидко вчиться в порівнянні з іншими моделями.

Word2Vec не потребує міток для створення значущих уявлень. Це корисно, тому що більша частина даних в реальному світі не має маркування. Якщо в мережі досить навчальних даних (десятки мільярдів слів), вона створює вектори слів з інтригуючими характеристиками. Слова зі схожим значенням з'являються в кластерах, а кластери розташовуються так, що деякі словосполучення, такі як аналогії, можуть бути відтворені з використанням векторної математики. Відомим прикладом є те, що з добре навченими векторами слова «король - чоловік + жінка = королева». Цей приклад був запропонований Томашем Міколовим у 2013 році, (рисунок 2.2).

Слово «man» відноситься до слова «woman» так само, як слово «uncle» до слова «aunt», що для нас цілком природно і зрозуміло, але в інших моделях домогтися такого ж співвідношення векторів можна тільки за допомогою спеціальних хитрувань. Тут же – це відбувається природно з самого корпусу текстів. До речі, крім семантичних зв'язків, уловлюються і синтаксичні, праворуч показано співвідношення однини і множини.

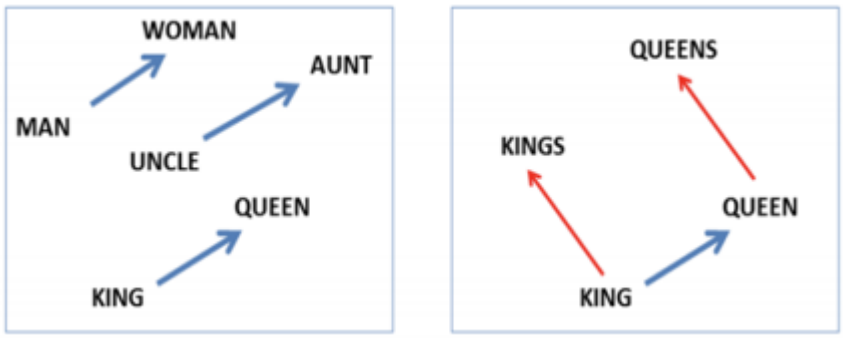

Рисунок 2.2 – Представлення Томаша Міколова принципу роботи Word2Vec

Модель, запропонована Міколовим дуже проста (і тому така гарна) – ми будемо передбачати ймовірність слова по його оточенню (контексту). Тобто ми

будемо вчити такі вектора слів, щоб ймовірність, що привласнюється моделлю слову була близька до ймовірності зустріти це слово в цьому оточенні в реальному тексті.

$$p(w_o|w_c) = \frac{e^{s(w_o,w_c)}}{\sum_{w_i \in V} e^{s(w_i,w_c)}},$$

де w_o – вектор цільового слова;

w_c – деякий вектор контексту, що знайдений з векторів оточуючих необхідне слово інших слів;

$s(w_1, w_2)$ – це функція, яка двум векторам зіставляє одне число.

Приведена формула називається softmax, або «м'який максимум», тобто диференційований. Це потрібно для того, щоб наша модель могла навчитися за допомогою backpropagation, тобто процесу зворотного поширення помилки.

Процес тренування влаштований таким чином: ми беремо послідовно $(2k+1)$ слів, слово в центрі є тим словом, яке повинно бути передбачене. А навколишні слова є контекстом довжини по k з кожного боку. Кожному слову в нашій моделі підтверджено унікальний вектор, який ми змінюємо в процесі навчання нашої моделі.

В цілому, цей підхід називається CBOW (Continuous Bag-Of-Words), continuous тому, що ми згодовуємо нашій моделі послідовно набори слів з тексту.

Також Міколов відразу був запропонований інший підхід – прямо протилежний CBOW, який він назвав Skip-Gram , тобто «словосполучення з пропуском». Ми намагаємося з даного нам слова вгадати його контекст (точніше вектор контексту). В іншому модель не зазнає змін.

За минулий час були запропоновані поліпшення, що стали вже також класичною моделлю Word2Vec. Два найпоширеніших будуть описані нижче.

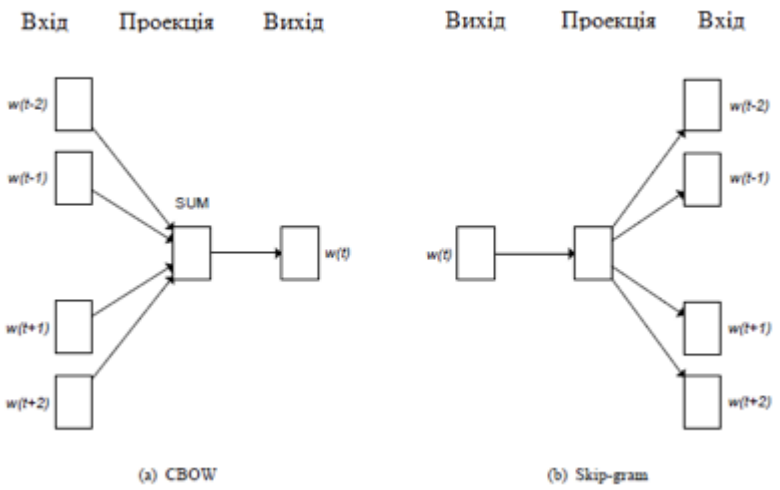

Рисунок 2.3 – Схематичне зображення моделі методу Word2Vec

– Negative Sampling.

У стандартній моделі CBOW, розглянутої вище, ми передбачаємо ймовірності слів і оптимізуємо їх. Функцією для оптимізації (мінімізації в нашому випадку) служить дивергенція Кульбака-Лейблера:

$$KL(p \,\|\, q) = \int p(x) \log \frac{p(x)}{q(x)} dx,$$

де $p(x)$ – розподіл ймовірностей слів, яке ми беремо з корпусу;

$q(x)$ – розподіл, яке породжує наша модель.

Дивергенція – це буквально «розбіжність», наскільки один розподіл не схожий на інший. Оскільки наші розподілу на словах, тобто є дискретними, ми можемо замінити в цій формулі інтеграл на суму:

$$KL(p \,\|\, q) = \sum_{x \in V} p(x) \log \frac{p(x)}{q(x)}.$$

Виявилося так, що оптимізувати цю формулу досить складно. Перш за все через те, що розраховується за допомогою softmax по всьому словнику. (Як ми пам'ятаємо, в англійському зараз близько мільйона слів.) Тут варто відзначити,

що багато слів разом не зустрічаються, як ми вже відзначали вище, тому більша частина обчислень в softmax є надмірною. Було запропоновано елегантний обхідний шлях, який отримав назву Negative Sampling.

Суть цього підходу полягає в тому, що ми максимізували ймовірність зустрічі для потрібного слова в типовому контексті і одночасно мінімізуємо ймовірність зустрічі в нетиповому контексті (тому, який рідко або взагалі не зустрічається). Формулою думка вище записується так:

$$NegS(w_o) = \sum_{i=1, x_i \sim D}^{i=k} -\log(1 + e^{s(x_i, w_o)}) + \sum_{j=1, x_j \sim D'}^{j=l} -\log(1 + e^{-s(x_j, w_o)}),$$

де $s(x, w)$ – точно такий же, що і в оригінальній формулі, а решта дещо відрізняється. Перш за все варто звернути увагу на те, що формулою тепер складається з двох частин: додатної ($s(x, w)$) і від'ємної ($-s(x, w)$).

Додатна частина відповідає за типові контексти, і D тут – це розподіл спільної зустрічальності слова w і інших слів корпусу.

Від'ємна частина – це набір слів, які з нашим цільовим словом зустрічаються рідко. Цей набір породжується з розподілу D', яке на практиці береться як рівномірний по всім словам словника корпусу.

– Hierarchical SoftMax.

Можна зайти з іншого боку – не міняти вихідну формулу, а спробувати порахувати сам softmax більш ефективно. Наприклад, використовуючи бінарне дерево. За всіма словами в словнику будується дерево Хаффмана. В отриманому дереві V слів розташовуються на листі дерева. На рисунку 2.4 подано приклад дерева Хаффмана (Huffman Binary Tree).

На рисунку 2.4 зображений приклад такого бінарного дерева. Жирним виділено шлях від кореня до слова w_2. Довжину шляху позначимо $L(w)$, а j-у вершину на шляху до слова w позначимо через $n(w, j)$. Можна довести, що внутрішніх вершин (не листя) $V1$.

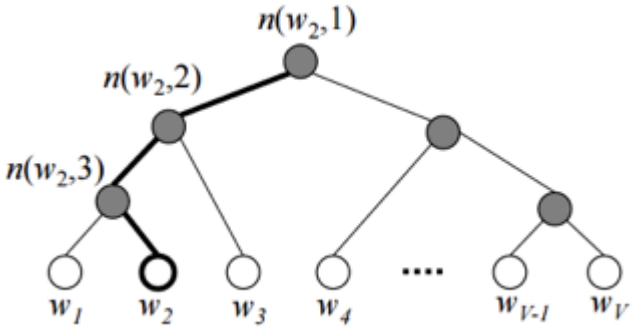

Рисунок 2.4 – Дерево Хаффмана

За допомогою ієрархічного softmax вектора $v_n(w, j)$ передбачається для $V-1$ внутрішніх вершин. А ймовірність того, що слово w буде вихідним словом (в залежності від того, що ми передбачаємо: слово з контексту або задане слово по контексту) обчислюється за формулою:

$$p(w = w_o) = \prod_{j=1}^{L(w)-1} \sigma([n(w, j+1) = lch(n(w, j))] v_{n(w,j)}^T u),$$

де $\sigma(x)$ – функція softmax;

$lch(n)$ – лівий син вершини n.

Слід зазначити, що $[true] = 1$, $[false] = -1$, $u = v_{w_1}$. Якщо використовується метод skip-gram, $u = \frac{1}{h}\sum_{k=1}^{h} v_{w_{1,k}}$, тобто, усереднений вектор контексту, якщо використовується CBOW. Формулу можна інтуїтивно зрозуміти, представивши, що на кожному кроці ми можемо піти наліво або направо з можливостями:

$$p(n, left) = \sigma(v_n^T u),$$

$$p(n, right) = 1 - p(n, left) = 1 - \sigma(v_n^T u) = \sigma(-v_n^T u).$$

Потім на кожному кроці ймовірності перемножуються ($L(w)-1$ кроків) і виходить шукана формула.

При використанні простого softmax для підрахунку ймовірності слова, доводилося обчислювати нормується суму за всіма словами з словника, потрібно $O(V)$ операцій. Тепер же ймовірність слова можна обчислити за допомогою послідовних обчислень, які вимагають $O(\log(V))$.

2.2.1.3 Алгоритм роботи Word2Vec

На основі викладеного вище можемо скласти алгоритм роботи техніки Word2Vec.

Крок 1. Читається корпус, і розраховується частота зустрічі кожного слова в корпусі (тобто кількість разів, коли слово зустрілося в корпусі – і так для кожного слова).

Крок 2. Масив слів сортується за частотою (слова зберігаються в хеш-таблиці), і видаляються рідкісні слова (їх ще називають гапакс).

Крок 3. Будується дерево Хаффмана. Дерево Хаффмана часто застосовується для кодування словника – це значно знижує обчислювальну і тимчасову складність алгоритму.

Крок 4. З корпусу читається так зване субречення (sub-sentence) і проводиться субсемплірування найбільш частотних слів (sub-sampling).

Субречення – це якийсь базовий елемент корпусу, зазвичай – просто речення, але це може бути і абзац, наприклад, або навіть ціла стаття.

Субсемплірування – це процес вилучення найбільш частотних слів з аналізу, що прискорює процес навчання алгоритму і сприяє значному збільшенню якості моделі, яку отримуємо.

Крок 5. По субреченню проходимо вікном (розмір вікна задається алгоритмом як параметр). В даному випадку під вікном мається на увазі максимальна дистанція між поточним і передбачуваним словом в реченні. Тобто, якщо вікно дорівнює трьом, то для речення «Я люблю згорткові нейронні мережі» аналіз (тобто застосування алгоритмів на базі однієї з архітектур CBOW або Skip-Gram) буде проходить всередині блоку в три слова – для «Я люблю згорткові»,

«згорткові нейронні мережі» і так далі. Вікно за замовчуванням дорівнює п'яти, рекомендованим значенням є десять.

Крок 6. Застосовується нейронна мережа прямого поширення (Feedforward Neural Network) з функцією активації ієрархічний softmax (Hierarchical Softmax) і / або негативне семплірування (Negative Sampling).

Hierarchical softmax краще веде себе при роботі з не дуже частотними словами, але працює повільніше, negative sampling краще працює з частотними словами і любить вектора слів невеликої розмірності (скажімо, 50–100), проте швидше.

Недавня робота зі Стенфорда [12] також застосувала глибоке навчання до аналізу настроїв; їх код доступний на Java. Однак їх підхід, заснований на розборі пропозицій, не може бути застосований простим способом до абзаців довільної довжини.

Розподілені вектори слів є потужними і можуть використовуватися для багатьох додатків, особливо для передбачення і перекладу слів. Тут ми спробуємо застосувати їх до аналізу настроїв.

2.2.2 Згорткова нейронна мережа

Згорткові нейронні мережі (Convolutional Neural Networks, CNN) – різновид штучних нейронних мереж прямого поширення, на основі яких лежить принцип обробки окремими нейронами не всіх вхідних сигналів одночасно (як, наприклад, в персептроном), а окремих областей (локальних рецептивних полів нейрона). Ці ідеї засновані на роботах Хьюбела і Визеля – двох нобелівських лауреатів в області фізіології і медицини, які вивчали процес обробки мозком візуальної інформації. В ході своїх експериментів з сітківкою ока і корою головного мозку кішок, вчені виділили такі важливі положення. По-перше, нейрони різних областей кори головного мозку реагують на порушення різних ділянок сітчатки ока. Такі нейрони були названі простими (simple). По-друге, ці прості нейрони є входами для інших нейронів, з більш складним поведінкою. Ці ней-

рони були названі комплексними (complex). Описані принципи «пошарової» організації обробки сигналів, що надходять з рецепторів сітківки ока, вплинули на подальші дослідження нейронних мереж і принципи глибокого навчання.

Першою згортковою нейронною мережею є неокогнітрон Фукушіма представлений в 1980 році. Архітектура неокогнітрона, яка полягає в реалізації принципів роботи кори головного мозку, описаних Хьюбел і Візелем, поклала початок всьому класу згорткових нейронних мереж.

Ієрархічна структура мережі на кожному шарі включає набір простих і комплексних нейронів (слід зазначити схожість термінології з біологічними клітинами кори головного мозку). Прості нейрони відповідають за розпізнавання конкретного вхідного образу (поданого на вхід мережі або з попереднього шару), в той час як комплексні обробляють виходи простих нейронів того ж шару. Це дозволяє зменшити залежність розпізнавання від зрушень і поворотів розпізнавання образів.

У 1989 році Ян ЛеКун (Yann LeCun) представив багатошарову згорткову нейронну мережу, здатну досить ефективно вирішувати завдання розпізнавання цифр в зображеннях. Майже через десять років ЛеКун з колегами представили мережу LeNet-5, яка перевершила всі існуючі алгоритми в області розпізнавання рукописних символів [11]. Ця розробка стала стандартом серед згорткових нейронних мереж, а розглянутий в статті набір рукописних цифр MNIST (Mixed National Institute of Standards and Technology database) і до цього дня використовується для порівняльної оцінки якості класифікації алгоритмів в області обробки зображень.

2.2.2.1 Принципи роботи CNN

Розглянемо архітектуру згорткової нейронної мережі на прикладі LeNet-5 (рисунок 2.5).

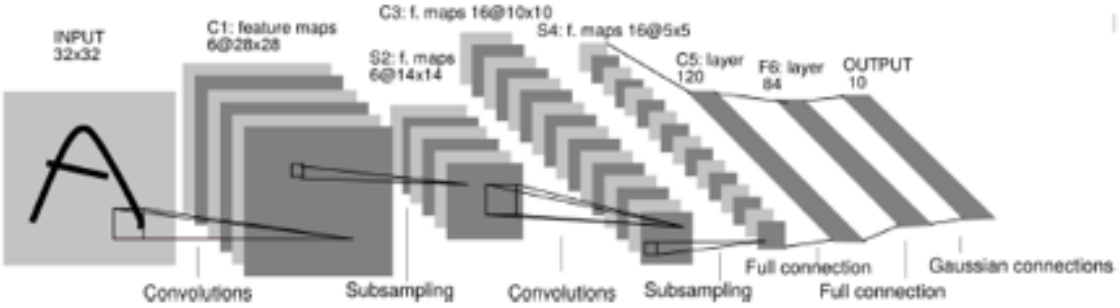

Рисунок 2.5 – Схематичне уявлення архітектури LeNet-5 [11]

Початкове зображення, представлене матрицею пікселів розміром 32x32, обробляється шістьма згортковими фільтрами, що формують шість карт ознак (feature maps) розміром 28x28 пікселів. Кожен елемент карти ознак є проекцією невеликого поля пікселів розміром 5x5 з вихідного зображення. Операція згортки має на увазі можливість перетину полів між собою. Наступна за нею операція підвибірки (subsampling) формує для кожної карти ознак з шару C1 нові карти розмірності 14x14, елементи якої відповідають непересічним проекція полів розміру 2x2. Аналогічні операції згортки і підвибірки проводяться в шарах C3 і S4, після чого повнозв'язні шари формують підсумкові результати класифікації.

Розглянемо архітектуру CNN на прикладі візуалізації від Стенфорду (рисунок 2.6).

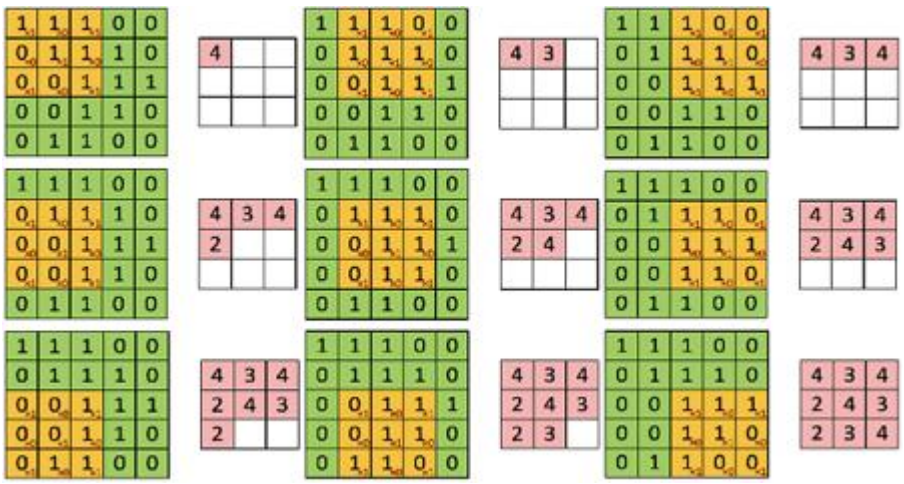

Рисунок 2.6 – Візуалізація принципу CNN

Вікно, яке ми зміщуємо в матриці називається фільтром (в англо-язичному варіанті karnel, filter або feature detector). Фільтр накладається на область великої матриці и кожне значення перемножуються з відповідним йому значенням фільтру (маленькі цифри нижче і правіше від значень матриці). Потім отримане значення складається і отримуємо вихідне («відфільтроване») значення.

Вікно ходить по великій матриці з шагом або stride. Шаг буває горизонтальний або вертикальний.

Розглянемо концепт каналу. Каналами в зображеннях називають базові кольори, наприклад, якщо ми говоримо про розповсюджену схему кодування кольорів RGB (Red, Green, Blue), то припускаємо, що шляхом змішування базових кольорів можна отримати будь-який колір.

Отримуємо зображення, в ньому є канали і по ньому з необхідним шагам ходить фільтр. Кожний фільтр (тобто матриця невеликого розміру) накладається на початкову матрицю одночасно на всі три канали. Результати підсумовуються.

Слід зазначити, що фільтрів в згорткових мережах значно більше. Тобто, існує n фільтрів, що виконують одну й ту ж роботу. Із-за різної ініціалізації фільтруючих матриць, в процесі навчання вони починають звертати увагу на різні деталі (рисунок 2.7).

Розглянувши принцип роботи згорткового шару, звернемо увагу на pooling-шар. Найпростіше пояснити його на прикладі max-pooling. Уявимо, що у згортковому шарі матриця фільтру зафіксована і є одиничною, тобто множення на неї ніяк не впливає на вхідні дані. Замість підсумування всіх результатів множення, обираємо максимальний елемент. Таким чином, обирається з всього вікна піксель з найбільшою інтенсивністю. Такий процес називається max-pooling (рисунок 2.8).

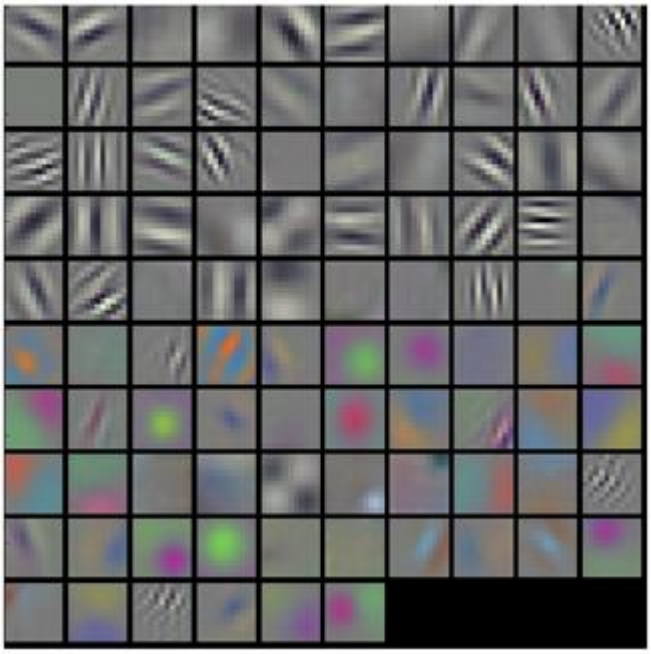

Рисунок 2.7 – Приклад роботи різних фільтрів на одній матриці

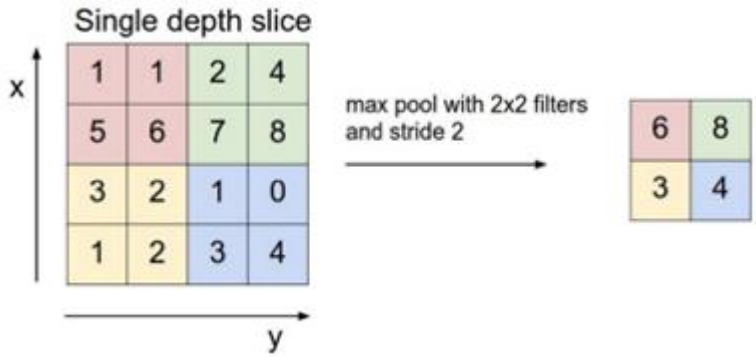

Рисунок 2.8 – Опис процесу max-pooling

Отримавши свій початок в області обробки зображень, згорткові нейронні мережі стали асоціюватися з завданнями розпізнавання образів і комп'ютерного зору, отримавши безліч прикладних застосувань в вітчизняних там зарубіжних розробках.

Проте, CNN-мережі отримали успішний розвиток і в інших областях: обробка звукових сигналів і розпізнавання мови, обробка відео, а також обробка текстових даних.

Розглядаючи більш детально застосування згорткових нейронних мереж для обробки тексту, варто детально зупинитися на тому, як вирішується про-

блема нефіксованої послідовності. Йоон Кім (Yoon Kim) в [12] запропонував наступне рішення. Вхідна послідовність з n слів представляється у вигляді конкатенації n векторних уявлень слів:

$$x_{1:n} = x_1 \oplus x_2 \oplus \cdots \oplus x_n, \tag{2.1}$$

де $\oplus$ – оператор конкатенації. Операція згортки має на увазі використання фільтра $w \in \mathbb{R}^{hk}$, де k – розмірність векторного уявлення, h – розмір вікна. Ознака c_i виходить з вибірки векторних уявлень слів $x_{i:i+h-1}$ наступним чином:

$$c_i = f(w \cdot x_{i:i+h-1} + b), \tag{2.2}$$

де $b \in \mathbb{R}$ – параметр зміщення. Застосовуючи рівняння (2.2) до всіх можливих вибірках слів з (2.1), отримаємо вектор ознак $c = [c_1, c_2, ..., c_{n-h+1}]$. Фінальним кроком для усунення нефіксованим довжини є перетворення вектора c в скаляр $\hat{c}$ шляхом взяття максимального елемента з вектора:

$$\hat{c} = \max\{c\}. \tag{2.3}$$

Ця операція має назву max-over-time polling.

Таким чином, за допомогою операцій (2.2) і (2.3), нефіксована послідовність перетвориться в фіксований скаляр. Зрозуміло, єдине скалярне значення, що представляє одиничну ознаку, не може належним чином відображати всі внутрішні залежності текстового фрагмента. На практиці використовується сукупність безлічі фільтрів при різних значеннях вікна h, що формує вектор або матрицю ознак фіксованого розміру, які згодом можна використовувати з іншими шарами.

Кім у своїй роботі використовував архітектуру, що використовує по 100 карт ознак для кожного із значень 3, 4 і 5 ширини вікна h (рисунок 2.7).

2.2.2.2 CNN та Word2Vec

Використовуючи векторні уявлення, отримані за допомогою алгоритму Word2Vec, Кім показав, що згорткові нейронні мережі можуть ефективно вирішувати завдання обробки текстів на природній мові, зокрема – завдання аналізу настрою користувачів, перевершуючи всі інші алгоритми в деяких тестах [12].

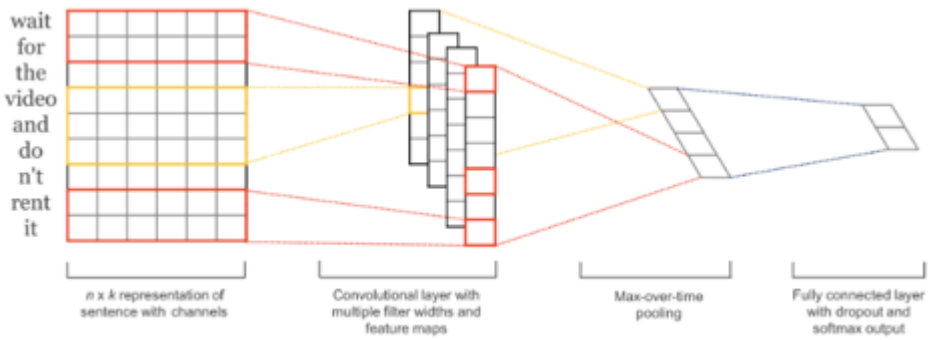

Рисунок 2.9 – Згорткова нейронна мережа Кіма [12]

Слід зазначити, що на рисунку 2.9 матриця для зручності відображення транспонована. Рисунок вище майже повністю пояснює роботу CNN з текстом. Для розуміння як формується матриця, необхідно розуміти термін embedding.

Embedding – це зіставлення точки в якому-небудь багатовимірному просторі об’єкту, в нашому випадку – слово. Найвідомішим прикладом такого embedding є Word2Vec, у якого є семантичні властивості, наприклад

```
word2vec(«king») – word2vec(«man») + word2vec(«woman») ~=
                  ~= word2vec(«queen»).
```

Беремо embedding для кожного слова в тексті і ставимо всі вектора в ряд, отримуючи шукану матрицю.

Далі, говорячи про згортку, можна сказати, що вона ходить по одній осі – по ширині (послідовності токенів в реченні). Тому, щоб розрізнити від стандартної згортки її називають одномірною (1D convolution). Після чого виконується

операція max-over-pooling, що застосовується до всієї послідовності одразу, тобто ширина його вікна дорівнює усій ширині матриці.

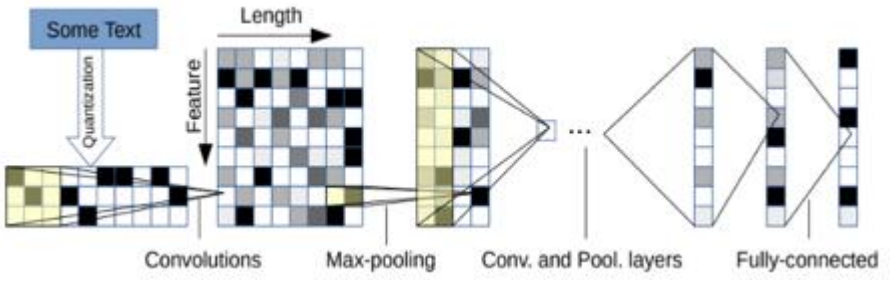

Рисунок 2.10 – Приклад використання CNN для текстів

2.2.2.3 Переваги CNN перед іншими типами мереж

Згорткові нейронні мережі гарні там, де потрібно побачити шматочок або всю послідовність цілком і зробити якийсь висновок з цього. Тобто це завдання, наприклад, детекції спаму, аналізу тональності або вилучення іменованих сутностей.

В роботі [12] Юн Кім (Yoon Kim) показує, що CNN гарні для класифікації пропозицій на різних датасета. Він працює з Word2Vec, але можна і працювати безпосередньо з символами.

В роботі [3] автори класифікують тексти виходячи прямо з букв, вивчаючи embedding для них в процесі навчання. На великих датасета вони показали навіть кращі результати, ніж мережі, які працювали зі словами.

У згорткових нейронних мереж є істотний недолік, в порівнянні з рекурентною нейронною мережею (Recurrent neural network, RNN) - вони можуть працювати тільки зі входом фіксованого розміру (тому що розміри матриць в мережі не можуть змінюватися в процесі роботи). Але автори вищезгаданої роботи [18] змогли вирішити цю проблему. Так що тепер і це обмеження зняте.

В роботі [14] автори класифікують тексти виходячи прямо з символів. Вони використовували набір з 70 символів для того щоб представити кожен символ як one-hot вектор і встановили фіксовану довжину тексту в 1014 символу.

Таким чином, текст представлений бінарною матрицею розміром 70x1014. Мережа не має уявлення про слова і бачить їх як комбінації символів, і інформація про семантичної близькості слів не надано мережі, як у випадках заздалегідь натренованих Word2Vec векторів. Мережа складається з 1d conv, max-pooling шарів і двох fully-connected шарів з дропаутов. На великих датасетах вони показали навіть кращі результати, ніж мережі, які працювали зі словами. До того ж цей підхід помітно спрощує preprocessing крок що може посприяти на його використання на мобільних пристроях.

В іншій роботі [15] автори намагаються поліпшити застосування CNN в NLP використанням напрацювань з комп'ютерного зору. Головні тренди в комп'ютерному зорі останнім часом це збільшення глибини мережі і додавання так званих skip-зв'язків (e.g., ResNet) які пов'язують шари що не є сусідами один з одним. Автори показали що ті ж принципи застосовні і в NLP, вони побудували CNN на основі символів з 16-розмірними embedding, яке вчилися разом з мережею. Вони натренували мережі різної глибини (9, 17, 29 і 49 conv шарів) і експериментували зі skip-зв'язками, щоб з'ясувати, як вони впливають на результат. Вони прийшли до висновку, що збільшення глибини мережі покращує результати на обраних датасета, але продуктивність дуже глибоких мереж (49 шарів) нижче ніж помірно глибоких (29 шарів). Застосування skip-зв'язків привів до поліпшення результатів мережі з 49 шарів, але все одно не перевершив показник мережі з 29 шарами.

Інша важлива особливість CNN в комп'ютерному зорі – це можливість використання ваг мережі натренованої на одному великому датасета (типовий приклад – ImageNet) в інших завданнях комп'ютерного зору. В роботі [16] автори досліджують придатність цих принципів в завданні класифікації текстів за допомогою CNN з word embedding. Вони вивчають, як перенесення тих чи інших частин мережі (embedding, conv шари і fully-connected шари) впливають на результати класифікації на обраних датасетах. Вони приходять до висновку, що в NLP завданнях семантична близькість джерела, на якому заздалегідь тренувалася мережу, грає важливу роль, тобто мережа, натренована на рецензіях до фі-

льмів, буде добре працювати на іншому датасеті з кінорецензій. До того ж вони відзначають, що використання тих же embedding для слів збільшує успіх трансферу і рекомендують не заморожувати шари, а дотреновувати їх на цільовому датасеті.

2.2.3 Рандомний ліс

Рандомний ліс – модель, що складається з безлічі дерев рішень. Замість того, щоб просто усереднювати прогнози різних дерев (така концепція називається просто «ліс»), ця модель використовує дві ключові концепції, які і роблять цей ліс випадковим.

Концепція 1. Випадкова вибірка зразків з набору даних при побудові дерев.

В процесі тренування кожне дерево випадкового лісу вчиться на випадковому зразку з набору даних. Вибірка зразків відбувається з відшкодуванням (в статистиці цей метод називається бутстреппінг, bootstrapping). Це дає можливість повторно використовувати зразки одним і тим же деревом. Хоча кожне дерево може бути високоваріативним по відношенню до певного набору тренувальних даних, навчання дерев на різних наборах зразків дозволяє знизити загальну варіативність лісу, не жертвуючи точністю.

При тестуванні результат виводиться шляхом усереднення прогнозів, отриманих від кожного дерева. Підхід, при якому кожен елемент, що навчається, отримує власний набір навчальних даних (за допомогою бутстреппінга), після чого результат усереднюється, називається bagging (від bootstrap aggregating).

Концепція 2. При поділі вузлів обираються випадкові набори параметрів.

Друга базова концепція випадкового лісу полягає в використанні певної вибірки параметрів зразка для поділу кожного вузла в кожному окремому дереві. Зазвичай розмір вибірки дорівнює квадратному кореню із загального числа параметрів. Тобто, якщо кожен зразок набору даних містить 16 параметрів, то в кожному окремому вузлі буде використано 4. Хоча навчання випадкового лісу

можна провести і з повним набором параметрів, як це зазвичай робиться при регресії.

Рандомний ліс поєднує сотні або тисячі дерев прийняття рішень, навчаючи кожне на окремомій вибірці даних, розділяючи вузли в кожному дереві з використанням обмеженого набору параметрів. Підсумковий прогноз робиться шляхом усереднення прогнозів від всіх дерев.

Щоб краще зрозуміти перевагу випадкового лісу, необхідно уявити таку ситуацію: вам потрібно вирішити, чи підніметься ціна акцій певної компанії. У вас є доступ до дюжини аналітиків, з самого початку не знайомих зі справами цієї компанії. Кожен з аналітиків характеризується низьким ступенем похибки, так як не робить будь-яких припущень. Крім того, вони можуть отримувати нові дані з новинних джерел.

Труднощі завдання в тому, що новини, крім реальних сигналів, можуть містити шум. Оскільки передбачення аналітиків базуються виключно на даних - мають високу гнучкість – вони можуть бути спотворені інформацією, що не належить до справи. Аналітики можуть прийти до різних висновків, виходячи з одних і тих же даних. Крім того, кожен аналітик намагається робити прогнози, максимально корельовані з отриманими звітами (висока варіативність) і передбачення можуть значно відрізнятися при різних наборах новинних джерел.

Тому потрібно не спиратися на рішення якогось одного аналітика, а зібрати разом їх прогнози. Більш того, як і при використанні випадкового лісу, потрібно дозволити кожному аналітику доступ тільки до певних джерел новин, в надії на те, що ефекти шумів будуть нейтралізовані вибіркою. У реальному житті ми покладаємося на безліч джерел (ніколи не довіряйте єдиному огляду на Amazon). Інтуїтивно нам близька не лише ідея дерева рішень, але і комбінування їх в випадковий ліс.

2.2.3.1 Алгоритм класифікації Random Forest

Для вирішення задачі класифікації використовуємо алгоритм Рандомного Лісу (Random Forest, RF).

Цей алгоритм складений з безлічі дерев рішень, тому спочатку необхідно зрозуміти, як одне таке дерево вирішує проблему класифікації.

Дерево рішень – інтуїтивно зрозуміла базова одиниця алгоритму RF. Його можна розглядати як серію питань так / ні про вхідні дані. В кінцевому підсумку питання отримаємо передбачення певного класу (або величини в разі регресії). Це модель що інтерпретується, так як рішення приймаються так само, як і людиною: ми задаємо питання про доступні дані до тих пір, поки не дійдемо до певного рішення (в ідеальному світі).

Базова ідея дерева рішень полягає у формуванні запитів, з якими алгоритм звертається до даних. При використанні алгоритму CART (Classification and Regression Tree) питання (також звані поділи вузлів) визначаються таким чином, щоб відповіді вели до зменшення забруднення Джині (Gini Impurity). Це означає, що дерево рішень формує вузли, що містять велику кількість зразків (з набору вихідних даних), що належать до одного класу. Алгоритм намагається виявити параметри з подібними значеннями.

Подробиці забруднення Джині розглядаються нижче.

2.2.3.2 Принципи роботи алгоритму RF

Розглянемо просту бінарну класифікацію, зображеноу на діаграмі (рисунок 2.11).

Даний набір даних має всього два параметри (дві задані змінні), x_1 і x_2, а також 6 зразків, що несуть ці параметри. Зразки розділені мітками на два класи. Хоча це просте завдання, лінійні класи розділити неможливо. Це означає, що ми не можемо намалювати на запропонованій площині пряму лінію, яка відокремить один клас від іншого.

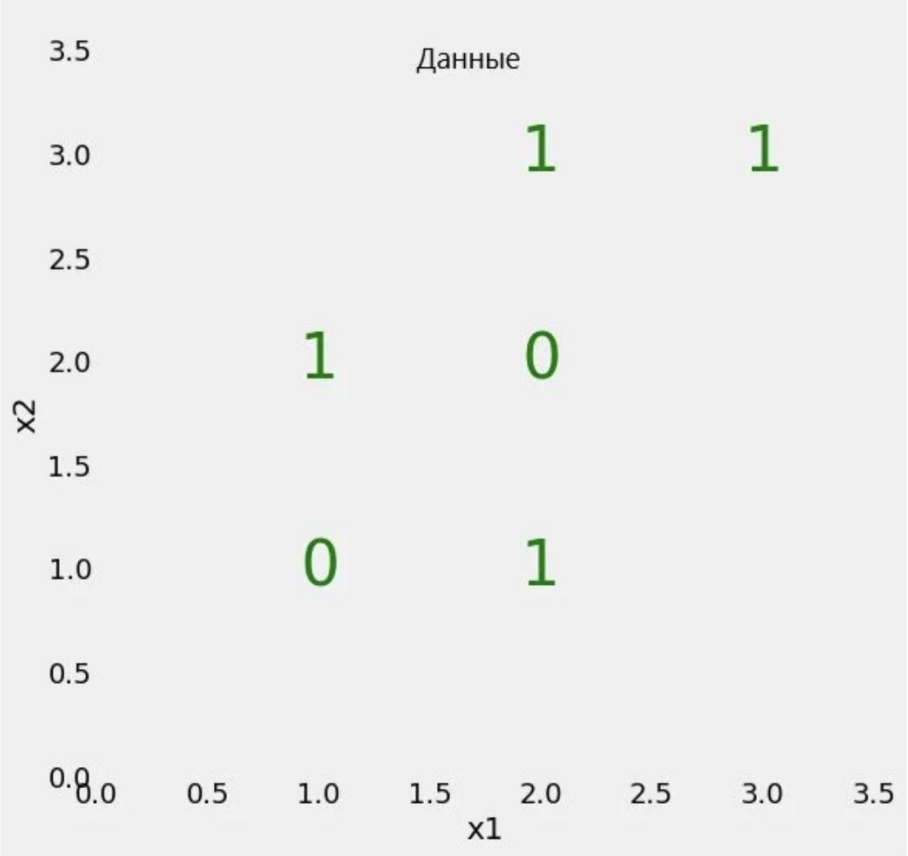

Рисунок 2.11 – Діаграма бінарної класифікації

У той же час ми можемо розбити площину на ділянки (вузли) декількома прямими лініями. Саме це робить дерево рішень в процесі тренування. По суті дерево рішень – нелінійна модель, створена за допомогою безлічі лінійних обмежувачів.

Під час навчання ми використовуємо і параметри, і мітки, щоб модель навчилася сортувати дані на основі параметрів. Для таких завдань не використовується тестовий набір даних. Але при тестуванні моделі ми повідомляємо тільки параметри і порівнюємо результат сортування з тими мітками, які очікували отримати.

Далі ми перевіряємо точність передбачення нашої моделі. Зрозуміло, ми отримаємо точність 100%, так як повідомили моделі правильні відповіді (y) і не обмежували глибину дерева. Але слід пам'ятати, що подібна підгонка дерева рішень під тренувальні дані може спровокувати перенавчання моделі.

2.2.3.3 Візуалізація дерева рішень

Розглянемо візуалізацію моделі алгоритму на рисунку 2.12.

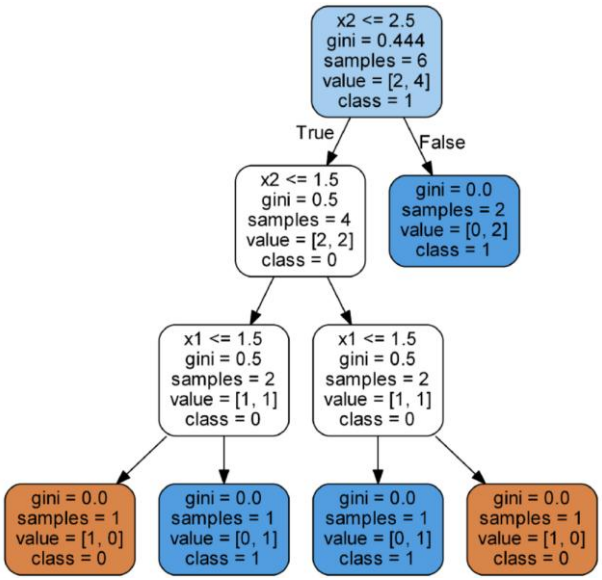

Рисунок 2.12 – Візуалізація моделі RF

У всіх вузлах, крім листя (кольорові вузли без вихідних зв'язків), міститься п’ять частин:

Частина 1. Питання про значення параметра зразка. Відповідь може приймати значення True або False. Це точка поділу вузла, в залежності від відповіді відомо, в якому напрямку вниз по дереву просунеться зразок даних.

Частина 2. Gini: середньозважене забруднення Джині має зменшуватися в міру того, як ми рухаємося вниз по дереву.

Частина 3. Samples: кількість зразків, що пройшли через цей вузол.

Частина 4. Value: відношення класів, які пройшли через цей вузол, виражене в абсолютних числах. Наприклад, верхній вузол виділив 2 зразка класу 0 і 4 зразка класу 1.

Частина 5. Class: клас більшості, що пройшов через вузол зразків. Для листя це прогнозоване значення всіх вузлів, що потрапляють в ці елементи.

Листя не містять питання, так як є фінальним прогнозованим значенням класифікації. Щоб обробити новий елемент набору даних, потрібно просто рухатися вниз по дереву, використовуючи параметри елемента для відповідей на

питання. У фіналі ви дійдете до одного з листів, значення Class якого і буде прогнозованим класом елемента.

Щоб поглянути на дерево рішень під іншим кутом, ми спроецуємо поділ моделі на вихідні дані (рисунок 2.13).

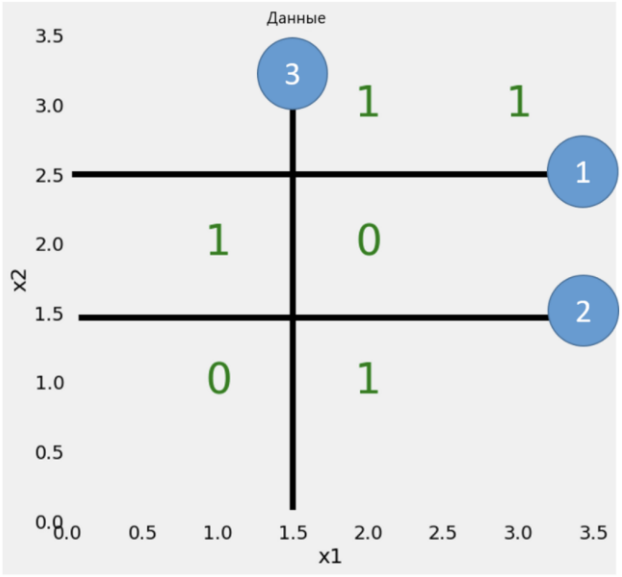

Рисунок 2.13 – Поділ моделі на вихідні дані

Кожний поділ відображається однією лінією, що розділяє зразки даних на вузли відповідно до значення параметрів. Оскільки максимальна глибина дерева не обмежена, поділ розміщує кожен елемент в вузол, що містить тільки елементи того ж класу. Пізніше ми розглянемо, як ідеальний поділ навчальних даних може привести до перенавчання.

2.2.3.4 Забруднення Джині

Забруднення Джині – ймовірність невірного маркування в вузлі випадково обраного зразка. Наприклад, у верхньому (корневому) вузлі ймовірність невірної класифікації зразка дорівнює 44.4%. Це можна обчислити за допомогою рівняння:

$$I_G(n) = 1 - \sum_{i=1}^{J} (p_i)^2 .$$

Забруднення Джині вузла n дорівнює 1 мінус сума відносин класу до загальної кількості зразків p_i, зведених в квадрат, для кожного з безлічі класів J (в нашому випадку це всього 2 класу). Звучить складно, тому покажемо, як обчислюється забруднення Джині для корневого вузла:

$$I_{root} = 1 - \left(\left(\frac{2}{6} \right)^2 + \left(\frac{4}{6} \right)^2 \right) = 1 - \frac{5}{9} = 0,444.$$

У кожному вузлі дерево рішень шукає таке значення певного параметра, яке призведе до максимального зменшення забруднення Джині. В якості альтернативи для поділу вузлів також можна використовувати концепцію накопичення інформації.

Потім процес поділу повторюється з використанням «жадібної», рекурсивної процедури, поки дерево не досягне максимальної глибини або в кожному вузлі не залишаться тільки зразки одного класу. Загально-зважене забруднення Джині має зменшуватися з кожним рівнем. У нашому випадку на другому рівні воно складе 0,333:

$$I_{\text{second layer}} = \frac{n_{left}}{n_{parent}} \cdot I_{\text{left node}} + \frac{n_{right}}{n_{parent}} \cdot I_{\text{right node}} =$$

$$= \frac{4}{6} \cdot 0,5 + \frac{2}{6} \cdot 0,333.$$

Питома вага забруднення Джині для кожного вузла дорівнює відношенню кількості зразків, оброблених цим вузлом, до кількості оброблених батьківським вузлом. По такому принципу можна розрахувати забруднення Джині для

наступних рівнів дерева і окремих вузлів, використовуючи дані візуалізації. Таким чином, ефективна модель будується на базових математичних операціях.

В результаті загально-зважене забруднення Джині останнього шару зводиться до нуля. Це означає, що кожен кінцевий вузол містить тільки зразки одного класу, і випадково обраний зразок не може бути невірно класифікований. Звучить відмінно, але пам'ятайте, що це може бути сигналом того, що модель перенавчилася. Це відбувається, тому що вузли змодельовані тільки на навчальних даних.

2.2.3.5 Переваги алгоритму RF

Може скластися враження, що для вирішення завдання вистачило б і одного дерева рішень. Адже ця модель не робить помилок. Однак важливо пам'ятати, що алгоритм безпомилково відсортував тільки тренувальні дані. Цього і слід було очікувати, оскільки ми вказали вірні відповіді і не обмежили глибину дерева (кількість шарів). Але мета машинного навчання полягає в тому, щоб навчити алгоритм узагальнювати отриману інформацію і правильно обробляти нові дані, що раніше не зустрічалися.

Перенавчання відбувається, коли ми використовуємо дуже гнучку модель (з високою місткістю), яка просто запам'ятовує навчальний набір даних, підганяючи вузли під нього. Проблема в тому, що така модель виявляє не тільки закономірності в даних, але і будь-який присутній в ній шум. Таку гнучку модель часто називають високоваріативною, оскільки параметри, що формуються в процесі навчання (такі як структура дерева рішень) будуть значно варіюватися в залежності від навчального набору даних.

З іншого боку, у недостатньо гнучкої моделі буде високий рівень похибки, оскільки вона робить припущення щодо тренувальних даних (модель зміщується в бік упереджених припущень про дані). Наприклад, лінійний класифікатор передбачає, що дані розподілені лінійно. Через це він не володіє достатньою

гнучкістю для відповідності нелінійним структурам. Жорстка модель може виявитися недостатньо ємною навіть для відповідності тренувальним даними.

В обох випадках і при високій варіативності, і при високій похибці – модель не зможе ефективно обробляти нові дані.

Пошук балансу між зайвою і недостатньо гнучкої моделі є ключовою концепцією машинного навчання і називається компромісом між варіативністю і похибкою (bias-variance tradeoff).

Алгоритм дерева рішень перенавчається, якщо не обмежити його максимальну глибину. Він має необмежену гнучкістю і може зростати, поки не досягне стану ідеальної класифікації, в якій кожному зразку з набору даних буде відповідати один листок. Якщо повернутися назад до створення дерева і обмежити його глибину двома шарами (зробивши лише один поділ), класифікація більше не буде на 100% вірною. Ми зменшуємо варіативність за рахунок збільшення похибки.

В якості альтернативи обмеження глибини, яке веде до зменшення варіативності (добре) і збільшення похибки (погано), ми можемо зібрати безліч дерев в єдину модель. Це і буде класифікатор на основі комітету дерев прийняття рішень або просто «випадковий ліс».

2.2.4 TensorFlow

TensorFlow – це нейронна мережа, яка вчиться вирішувати завдання шляхом позитивного посилення і обробляє дані на різних рівнях (вузлах), що допомагає знаходити вірний результат.

Відкривши вихідний код бібліотеки машинного навчання TensorFlow, в Google спростили процес побудови і розгортання складних нейронних мереж. TensorFlow не надає кожному розробнику можливість скористатися плодами машинного навчання, але пропонує інтерфейси API для мов Python і C / C ++, що дозволяють підключатися до програми розробника.

Машинне навчання такого роду призначено винятково для дослідницьких цілей.

Бібліотеки TensorFlow помітно спрощують вбудовування в додатки самонавчального елементів і функцій штучного інтелекту, призначених для розпізнавання мови, організації комп'ютерного зору або обробки природної мови.

2.2.4.1 Принципи роботи TensorFlow

Принципи роботи с TensorFlow достатньо прості. Необхідно скласти граф операцій, потім передати в цей граф дані і дати команду виконати розрахунку. На рисунку 2.14 зображено приклад графу.

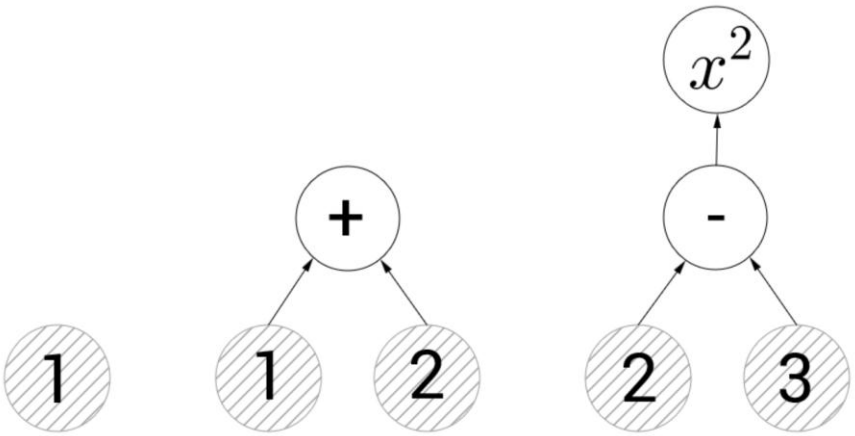

Рисунок 2.14 – Приклад графів для роботи с TensorFlow

Граф зліва містить тільки одну вершину, що представляє константу зі значенням 1. Тут і далі по тексту, в таких ілюстраціях, колами з сірої штрихуванням будуть позначатися вершини з константами, а без штрихування вершини з операціями. Центральний граф ілюструє операцію додавання. Якщо ми попросимо TensorFlow обчислити значення вершини, що представляє операцію складання, то він визначить значення спрямованих в нього ребер графа і підсумує їх (тобто, буде повернено 3). На правому ж графі у нас дві вершини з операціями - віднімання і зведення в квадрат. Якщо ми спробуємо вирахувати вершину, що представляє зведення в квадрат, то TensorFlow спершу виконає віднімання.

Порожній граф можна побудувати за допомогою функції tf.Graph(), крім того граф за замовчуванням створюється при підключенні бібліотеки і якщо ви не будете явно вказувати граф, то буде використовуватися саме він. У прикладі нижче показано, як можна створити дві константи в двох різних графах.

```
import tensorflow as tf # в подальшому цей рядок буде опускатися

# зберігаємо граф по-замовчуванню в змінну
default_graph = tf.get_default_graph()
# оголошуємо константу в графі по-замовчуванню
c1 = tf.constant(1.0)

# створюємо пустий граф
second_graph = tf.Graph()
with second_graph.as_default():
  # в цьому блоці ми працюємо в другому графі
    c2 = tf.constant(101.0)

print(c2.graph is second_graph, c1.graph is second_graph) # True, False
print(c2.graph is default_graph, c1.graph is default_graph) # False, True
```

Передача даних і виконання операцій відбуваються в сесіях. Запуск сесії здійснюється викликом tf.Session, а її закриття викликом методу close на об'єкті сесії. Можна використовувати конструкцію with, яка автоматичний закриває сесію:

```
default_graph = tf.get_default_graph()
c1 = tf.constant(1.0)
second_graph = tf.Graph()
with second_graph.as_default():
    c2 = tf.constant(101.0)

session = tf.Session() # відкриваємо сесію на графі за замовчуванням
print(c1.eval(session=session))
# print(c2.eval(session=session)) # так не можна, не той граф
session.close()

# теж саме:
with tf.Session() as session:
    print(c1.eval()) # не потрібно передавати сесію в метод eval

# використовуємо інший граф:
with tf.Session(graph=second_graph) as session:
    print(c2.eval())  # не потрібно передавати сесію в метод eval

#Вивід:
```

```
# 1.0
# 1.0
# 101.0
```

Далі розглянемо побудову графів. У попередніх прикладах в граф додавалися константи, нижче пояснюється чим вони відрізняються від placeholder'ів і змінних.

Отже, placeholder – це вузол, через який в модель будуть передаватися нові дані, а змінна (Variable) – це вузол, який може змінюватися по ходу виконання графа. Я сподіваюся, що вищеописаний матеріал всім зрозумілий, тому що його якраз досить для того, щоб приступити до навчання першої моделі. У попередньому фрагменті коду ми склали граф лінійної функції $a \cdot x + b$, тепер же давайте підемо трохи далі і апроксимуємо функцію $a \cdot x + b$ по набору точок.

Щоб TensorFlow міг навчати модель нам потрібно додати ще дві речі: функцію втрат і сам алгоритм оптимізації.

Функція втрат – це функція, яка приймає значення функції передбачене моделлю і фактичне значення, а повертає відстань між ними (будемо називати це значення помилкою). Наприклад, якщо ми передбачаємо дійсне значення, то в якості опції втрат можна взяти квадрат різниці аргументів або модуль їх різниці. Якщо у нас завдання класифікації, то функція втрат може повертати 0 при правильній відповіді і 1 при помилках. Грубо кажучи, функція втрат повинна повернути невід'ємне дійсне число і воно повинно бути тим більше, чим сильніше модель помиляється і тоді завдання навчання моделі зведеться до мінімізації. І хоча остання пропозиція не зовсім коректно, зате в повній мірі відображає ідею машинного навчання.

З методів оптимізації ми розглядаємо тільки класичний градієнтний спуск, який пояснюється за допомогою візуалізацій. Нижче представлені два варіанти одного і того ж графіка

$$-\sin\left(\frac{1}{2}x^2 - \frac{1}{4}y^2\right) + \cos(2x+1).$$

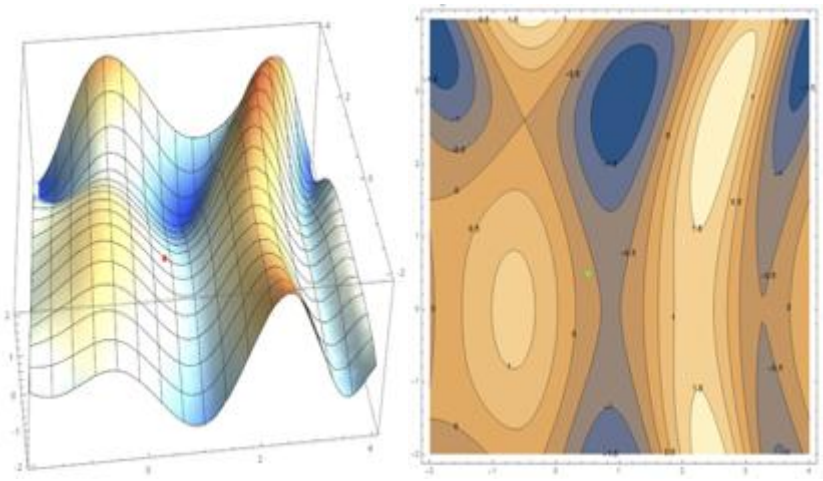

Рисунок 2.15 – Зображення точки локального мінімуму

Завдання методу – знайти локальний мінімум, тобто з точки (взятої навмання, на графіку $\left(\frac{1}{2};\frac{1}{2}\right)$) потрапити в поглиблення (темна зона на графіках).

Суть методу полягає в тому, щоб йти в напрямку протилежному градієнту функції в поточній точці.

Градієнт – це вектор, який вказує в напрямку найбільшого зростання функції. Математично це вектор з похідних по всіх аргументів

$$-grad(f) = \nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right).$$

Окремо варто сказати про швидкість з якою потрібно рухатися до мінімуму (стосовно до задачі машинного навчання, це буде називатися швидкістю навчання). Для отримання перших результатів, нам досить буде підібрати фіксовану швидкість. Однак часто буває хорошою ідеєю знижувати її по ходу виконання алгоритму, тобто рухатися все меншими і меншими кроками.

Завдання можна вирішити стохастичним методом, який має на увазі під собою обчислення градієнта тільки на невеликому фрагменті даних (пакеті), а не на всіх точках відразу. Таким чином стохастичний зпуск вимагає набагато менше обчислень, але не гарантує зменшення помилки на кожній ітерації. Як не

дивно цей «шум» в величині збіжності за часом може виявитися навіть корисний, тому що дозволяє «виходити» з локальних мінімумів.

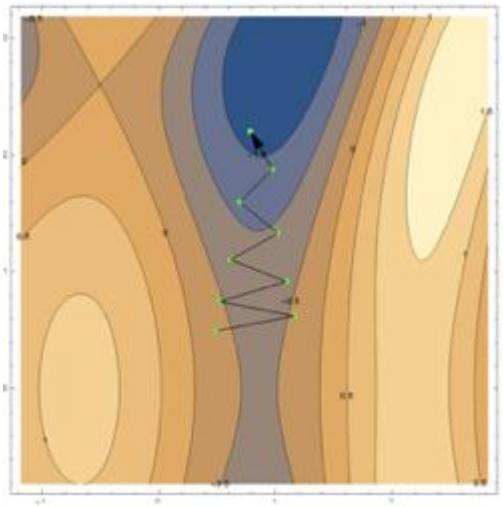

Рисунок 2.16 – Зображення градієнту

2.2.4.2 Побудова графів за допомогою TensorBoard

Також слід наділити увагу побудові графів у TensorBoard. TensorBoard використовується, щоб згенерувати граф у вигляді діаграми. Він читає поле імені, яке зберігається в кожній операції. Можна використовувати ці імена TensorFlow і перейти на більш звичні імена змінних Python. Використання tf.mul еквівалентно простому множенню з *, але тут можна встановити ім'я для операції.

```
>>> x = tf.constant(1.0, name='input')
>>> w = tf.Variable(0.8, name='weight')
>>> y = tf.mul(w, x, name='output')
```

TensorBoard дивиться в директорію виведення, створену з сесій TensorFlow. Ми можемо писати в цей висновок за допомогою SummaryWriter, і, якщо не робити нічого крім одного графа, то буде записаний тільки один граф.

Перший аргумент при створенні SummaryWriter – це назва директорії для виведення, яка буде створена при необхідності.

```
>>> summary_writer = tf.train.SummaryWriter('log_simple_graph',
sess.graph)
```

Тепер можна запустити TensorBoard в командному рядку.

```
$ tensorboard --logdir=log_simple_graph
```

TensorBoard запускається як локальний веб-додаток на порті 6006. («6006» це «goog» до гори ногами). Якщо зайти в браузері на localhost:6006/#graphs, то можна побачити діаграму графа, створеного в TensorFlow. Приклад такої діаграми представлений на рисунку 2.17.

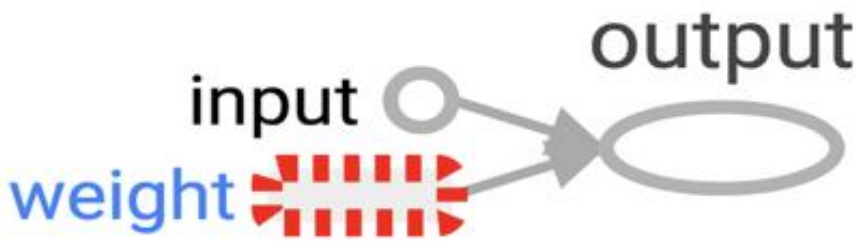

Рисунок 2.17 – Приклад діаграми графа побудованою за допомогою TensorBoard

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Python як мова високого рівня

Практичну частину даної дипломної роботи написано на Python – мові високого рівня.

Для мови Python існує ряд бібліотек, що дозволяють забезпечувати легковагові і ефективні API. Це дозволяє використовувати всього одну мову для створення серверного процесу, що забезпечує прикладний інтерфейс програмування для навчання моделей глибокого навчання, що важливо при написанні модулів аналізу. Таким чином, Python стає основою для модулів глибокого навчання.

З огляду на наявність добре розвинених і опрацьованих бібліотек на мові Python для доступу до соціальних мереж, виглядає раціональним використання її для розробки модуля аналізу даних. Також чимало хороших бібліотек для доступу до соціальних мереж мають і інші мови, наприклад, Ruby. Однак, при відсутності об'єктивних причин по використанню окремого мови програмування, вибір залишається на користь Python.

Python була обрана з наступних причин.

Стислість. Код на динамічно типизованій мові, якою є Python, як правило, виявляється коротше, ніж на інших популярних мовах, що забезпечує простоту розуміння. Легкість читання Python іноді називають «виконуваним псевдокодом». Це, звичайно, перебільшення, але сенс його в тому, що досвідчений програміст може прочитати написаний на Python код і зрозуміти, що він повинен робити.

Простота розширення. У стандартний дистрибутив Python вже входить багато бібліотек, в тому числі для обчислення математичних функцій, розбору XML-документів і завантаження веб-сторінок.

Інтерактивність. Програми на Python можна запускати прямо з командного рядка, а крім того, у нього є інтерактивний режим, в якому дозволяє здійснювати виклики функції, створювати об'єкти і тестувати пакети.

Мультипарадігмальність. Python підтримує об'єктно-орієнтована, процедурний і функціональний стилі програмування. Алгоритми машинного навчання досить різноманітні: для реалізації одного зручніше одна парадигма, а для реалізації іншого – інша. Іноді корисно передавати функції як параметри, іноді – зберігати стан в об'єкті. Python дозволяє і те й інше.

Багатоплатформність і безкоштовність. Існує одна еталонна реалізація мови Python для всіх основних платформ, і вона абсолютно безкоштовна.

3.2 Попередня обробка тексту

Крок 1. Видалення HTML розмітки: BeautifulSoup.

На даному етапі спочатку було видалено HTML теги. Для цього була використана бібліотека BeautifulSoup. Виклик .get_text() дає вам текст огляду, без тегів та розмітки. Якщо подивитися документацію BeautifulSoup, то можна переконатися, що це дуже потужна бібліотека – потужніша, ніж нам потрібно для цього набору даних. Тим не менш, не вважається надійної практикою видаляти розмітку за допомогою регулярних виразів, тому краще використовувати такий пакет, як BeautifulSoup.

Крок 2. Робота з пунктуацією, числами і стоп-словами: NLTK і регулярні вирази.

При розгляді питання про те, як очистити текст, слід подумати про проблему даних, яку ми намагаємося вирішити. Для багатьох проблем можна буде прибрати пунктуацію. З іншого боку, в цьому випадку необхідно вирішити проблему аналізу настроїв, і цілком можливо, що «!!!» або «:-(» може нести почуття, і їх слід розглядати як слова. У даній роботі повністю прибрані знаки пунктуації.

Далі були видалені всі числа, але є й інші способи роботи з ними, які мають такий же зміст. Наприклад, їх можна розглядати як слова або замінити рядками-заповнювачами, такими як «NUM».

Щоб видалити знаки пунктуації та цифри, був використаний пакет для роботи з регулярними виразами, який називається re, який постачається з Python.

Повний огляд регулярних виразів виходить за рамки даного керівництва, але на даний момент досить знати, що [] вказує на наявність в групі і ^ означає «ні». Іншими словами, в наведеному вище затвердження re.sub() говориться: «Знайдіть все, що НЕ є рядкової буквою (a-z) або великою літерою (A-Z), і замініть її пропуском».

Також були перетворені відгуки в нижній регістр і розділені на окремі слова («токенізація» в мові NLP).

Нарешті, потрібно вирішити, як поводитися з словами, що часто зустрічаються, які не мають великого значення. Такі слова називаються «стоп-словами»; в англійській мові вони включають такі слова, як «a», «and», «is» і «the». Для навчання Word2Vec краще не видаляти стоп-слова оскільки алгоритм використовує більш широкий контекст пропозиції для отримання високоякісних векторів слів. З цієї причини було зроблено видалення стоп-слів необов'язковим. Python включає в себе вбудований список стоп-слів.

3.3 Представлення розподілених слів у векторному вигляді

В Python була використана відмінна реалізацію Word2Vec з gensim пакету.

Хоча Word2Vec не вимагає графічних процесорів (GPU), як багато алгоритмів глибокого навчання, він вимагає значних обчислювальних ресурсів. Як версія Google, так і версія Python покладаються на багатопоточність (паралельне виконання декількох процесів на комп'ютері для економії часу). Для того, щоб

навчити вашу модель в розумні терміни, був встановлений Cython. Word2Vec працює без встановленого Cython, але він займе кілька днів замість хвилин.

3.4 Підготовка до навчання моделі

Далі необхідний конкретний формат вводу. Word2Vec очікує окремі пропозиції, кожне з яких представляє собою список слів. Іншими словами, формат вводу являє собою список списків.

Нелегко розділити абзац на речення. Природною мовою є всі види помилок. Англійські речення можуть закінчуватися символами «?», «!», «...» або «.», Крім того, пробіли і головні букви також не є надійними посібниками. З цієї причини було використано токенайзер Punkt в NLTK для поділу речень, щоб використовувати це. Було встановлено NLTK і викликана функція nltk.download(), щоб завантажити відповідний навчальний файл для punkt.

3.5 Навчання та збереження моделі

Зі списком добре проаналізованих пропозицій можна навчати модель. Існує ряд варіантів вибору параметрів, які впливають на час виконання та якість одержуваної остаточної моделі.

Архітектура: варіанти архітектури: скіп-грама (за замовчуванням) або безперервний пакет слів. Було виявлено, що скіп-грама трохи повільніше, але дає кращі результати.

Алгоритм навчання: ієрархічний softmax (за замовчуванням) або негативна вибірка. Дефолт працював добре.

Зниження частих слів: в документації Google рекомендуються значення від 0,000 до 0,001. Для нас значення ближче 0,001, здавалося, поліпшили точність остаточної моделі.

Розмірність слів у векторі: великі функції призводять до збільшення часу виконання і часто, але не завжди, до отримання кращих моделей. Розумні значення можуть бути від десятків до сотень; було використано 300.

Розмір контексту / вікна. Скільки слів контексту слід враховувати алгоритму навчання? 10 добре працює для ієрархічного softmax (чим більше, тим краще).

Робочі потоки: кількість паралельних процесів для запуску. Це залежить від комп'ютера, але від 4 до 6 повинно працювати на більшості систем.

Мінімальна кількість слів: це допомагає обмежити розмір словникового запасу значущими словами. Будь-яке слово, яке зустрічається у всіх документах багато разів, ігнорується. Розумні значення можуть бути між 10 і 100. Загальний словниковий запас становив близько 15 000 слів. Більш високі значення також допомагають обмежити час виконання.

В цьому експерименті для класифікації документів було використано алгоритм Random Forest, алгоритм тренується на навчальній вибірці та зберігає необхідні дані.

4 РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНОГО ЕКСПЕРИМЕНТУ

Переглянемо модель, яка була створена з 75 000 відгуків навчання. Функція «doesnt_match» спробує визначити, яке слово в наборі найбільше відрізняється від інших:

```
>>> model.doesnt_match ("man woman child kitchen" .split ()) 'kitchen'
```

Наша модель здатна розпізнавати відмінності в значенні! Вона знає, що чоловіки, жінки і діти більш схожі один на одного, ніж на кухню. Додаткові дослідження показують, що модель чутлива до більш тонких відмінностей в значенні, таких як відмінності між країнами і містами:

```
>>> model.doesnt_match ("france england germany berlin" .split ())
'berlin'
```

Хоча з відносно невеликим навчальним набором, який ми використовували, це, звичайно, не ідеально:

```
>>> model.doesnt_match ("paris berlin london austria". split ()) 'paris'
```

Також можемо використовувати функцію «most_s Similar», щоб отримати уявлення про кластери слів моделі:

```
>>> model.most_simil ("man")

[(u'woman', 0.6056041121482849), (u'guy', 0.4935004413127899),
(u'boy', 0.48933547735214233), (u'men', 0.4632953703403473), (u'person',
0.45742249488830566), (u'lady', 0.4487500488758087), (u'himself',
0.4288588762283325), (u'girl', 0.4166809320449829), (u'his',
0.3853422999382019), (u'he', 0.38293731212615967)]
```

```
>>> model.most_simil ("queen")

[(u'princess', 0.519856333732605), (u'latifah',
0.47644317150115967), (u'prince', 0.45914226770401), (u'king',
0.4466976821422577), (u'elizabeth', 0.4134873151779175), (u'antoinette',
0.41033703088760376), (u'marie', 0.4061327874660492), (u'stepmother',
0.4040161967277527), (u'belle', 0.38827288150787354), (u'lovely',
0.38668593764305115)]
```

Розглянемо приклад, більш актуальний для аналізу настроїв:

```
>>> model.most_similar("awful")

[(u'terrible', 0.6812670230865479), (u'horrible',
0.62867271900177), (u'dreadful', 0.5879652500152588), (u'laughable',
0.5469599962234497), (u'horrendous', 0.5167273283004761), (u'atrocious',
0.5115568041801453), (u'ridiculous', 0.5104714632034302), (u'abysmal',
0.5015234351158142), (u'pathetic', 0.4880446791648865), (u'embarrassing',
0.48272213339805603)]
```

Таким чином, у нас є досить гарна модель для семантичного значення. Отримані результати класифікації представлені в таблиці нижче 4.1.

Таблиця 4.1 – Результати класифікації

Method	precision	recall	F-measure	accuracy
Word2Vec	90,3%	87,2%	88,7%	89,8%

5 АНАЛІЗ МОЖЛИВИХ ЗАСТОСУВАНЬ

Моніторинг настроїв є важливим і актуальним інструментом управління, активно застосовуваним соціально-політичними, фінансово-економічними та громадськими структурами. Такий моніторинг дає можливість проаналізувати ставлення чи очікування громадськості з приводу якоїсь думки, що дозволяє більш грамотне майбутнє планування та вистроєння стратегій у різних структурах.

Також принципи моніторингу настроїв можна використовувати у галузі навчання. Академічні галузі зазвичай збирають відгуки студентів про основні аспекти курсу, таких як підготовка, зміст, методи викладання, пунктуальність, навички, оцінка та досвід навчання. Відгуки збираються з точки зору як якісних, так і кількісних показників. Недавні підходи до аналізу зворотного зв'язку використовують ручні методи і фокусуються в основному на кількісних коментарях. Таким чином, оцінка не може бути зроблена за допомогою більш глибокого аналізу. Тому є змога розробити систему інтелектуального аналізу зворотного зв'язку зі студентами, яка застосовуватиме підхід аналізу тексту та аналізу настроїв, щоб надати викладачам кількісний і глибший аналіз якісного зворотного зв'язку зі студентами, який поліпшить навчальний досвід студентів.

Ще однією з важливих галузей, де може використовуватися даний моніторинг – це медицина. Обробка природної мови була недавно використана для отримання клінічної інформації з вільного тексту в медичній карті. У клінічній мові існує проблема, яка полягає в тому, що значення клінічних об'єктів сильно залежить від модифікаторів тверджень, таких як заперечення, невизначеність, гіпотетичність, досвід тощо. Неправильне призначення тверджень може призвести до неточної діагностики стану пацієнтів або негативно вплинути на подальше дослідження, як моделювання хвороби. Таким чином, високоефективні клінічні системи, які можуть автоматично виявляти заперечення і інший статус підтвердження даних цільових медичних результатів (наприклад, хвороби, симптому) в клінічному контексті, дуже затребувані. Тому можливо створити систе-

му глибокого навчання, засновану на встановленні слів і заснованих на увазі двонаправлених мереж короткочасної пам'яті, для виявлення тверджень в клінічних замітках. На відміну від попередніх сучасних методів, які вимагали введення знань, така система може являти собою систему машинного навчання з поганими знаннями і може бути легко розширена або перенесена в інші області.

Програмний комплекс призначений для надання послуг з аналізу даних різним користувачам, потреби яких швидше за все будуть абсолютно різними. Кожен окремо взятий аналітик повинен мати свої власні настройки аналізу, результати і звіти. Ця вимога формує потребу в забезпеченні аутентифікації різних користувачів, для кожного з яких необхідно зберігати потрібні дані і надавати доступ до них. У той же час дані одного користувача повинні бути недоступні для іншого.

Найбільш простим і популярним методом забезпечення перевірки користувача в веб-проект є реєстрація нового користувача, що включає в себе секретні дані (пароль), а також процедура входу в систему («логін»), що вимагає введення секретного пароля, відомого тільки особі, спочатку реєструється в системі.

Аутентіфіцированний користувач отримує доступ до результатів моніторингу та аналітики. Однак, ці результати ще повинні бути попередньо сформовані на основі потреб окремо взятого аналітика. Розглядаючи аналітика як кінцевого споживача результатів аналізу, від якого не можна вимагати наявності будь-яких знань процедури аспектного аналізу тональності, слід чітко обмежити інформацію, яку він може надати як вхідні дані для моніторингу.

Новозареєстрований користувач системи має можливість подати заявку на проведення аналізу. Заявка містить назву предметної області та перелік аспектів, необхідних для аналізу. В даному розділі розглядається ситуація, коли заявка користувача являє собою унікальну формулювання предметної області та переліку аспектів, що виключає наявність заздалегідь підготовлених даних і готових до моніторингу агентів або ключових слів.

З огляду на той факт, що кінцевий користувач може не знати особливостей процедури моніторингу та аналізу, заявка може мати неточний характер, тому з боку сервісу заявка повинна бути переглянута людиною, який, в разі знаходження неточностей або інших проблемних місць в заявці, може зв'язатися з аналітиком і уточнити необхідні параметри. Ця процедура називається верифікацією заявки, а особа, відповідальна за процедуру верифікації (з боку сервісу), іменується менеджером по роботі з клієнтами.

Після верифікації заявки менеджер по роботі з клієнтами передає її оператору – особі, що відповідає за роботу з інтерфейсом системи, що дозволяє визначати ключові слова і проводити відбір агентів, схильних до моніторингу, а також встановлювати деталі цього моніторингу (періодичність, перелік аспектів і ін.). Оператор, відповідно до поданих заявок, фіксує перераховані параметри відповідно до заявленої предметною областю. На основі певної конфігурації система вже може отримати і зберегти в базі повідомлень наявні публікації, які, згодом, сформують (повністю або частково) навчальну вибірку для налаштування параметрів моделі глибокого навчання.

По завершенню обробки заявки оператором, в системі запускається моніторинг заданих ключових слів (агентів) і збір інформації. Однак, як вже було згадано, для настройки моделі глибокого навчання під задану тематику і аспекти, необхідно визначити навчальну вибірку. Вихідними даними для цієї вибірки є витягнуті повідомлення користувачів соціальних мереж. Необхідно участь людини для аспектно-орієнтованої розмітки повідомлень.

Особа, посадовим обов'язком якого є розмітка тональності повідомлень згідно заданим критеріям, іменується «розмітник». Кінцевою задачею розмітника є аспектна оцінка тональності текстового фрагмента за допомогою розробленого програмного комплексу.

Спираючись на подані операції, можна виділити список основних функціональних вимог до системи:

а) загальні функціональні вимоги:

1) реєстрація та аутентифікація користувачів;

2) подача, перегляд, редагування та затвердження заявки;

3) перегляд результатів моніторингу;

б) функціональні вимоги по роботі з соціальними мережами:

1) вибір соціальних мереж для моніторингу;

2) визначення набору ключових слів для моніторингу;

3) вибір агентів соціальних мереж (для кожної соціальної мережі);

4) збереження конфігурації – ключових слів і агентів соціальних мереж;

5) моніторинг соціальних мереж, збереження публікацій в базі;

в) функціональні вимоги по роботі з аналізатором тональності:

1) інтерфейс розмітки повідомлень згідно заданим в заявці аспектам;

2) збереження конфігурації навчальної вибірки;

3) навчання deep learning моделі, збереження параметрів;

4) використання збережених параметрів для аналізу тональності (настрою) довільного текстового фрагмента;

г) адміністративні функціональні вимоги:

1) створення і редагування службових користувачів;

2) можливість зміни параметрів конфігурації системи.

На основі зазначених функціональних вимог необхідно розробити архітектуру програмного комплексу, передбачивши відмовостійкість і гнучкість подальшого масштабування, вибрати найбільш ефективні інструменти (мови програмування, бібліотеки програмних компонентів, системи управління базами даних).

ВИСНОВКИ

Під час виконання атестаційної роботи було досліджено проблему інтелектуального аналізу повідомлень у соціальних мережах.

Був проведений аналіз існуючих шляхів вирішення проблеми інтелектуального аналізу повідомлень у соціальних мережах. Проведений системний аналіз проблеми інтелектуального аналізу повідомлень у соціальних мережах, надав можливість обрати оптимальний шлях її вирішення.

У даній роботі була побудована математична модель аналізу повідомлень та був розглянутий один з алгоритмів комп'ютерного бачення – навчання класифікатора. Було детально розглянуто алгоритм рішення, а також описані його переваги і недоліки.

Для розв’язання поставленої задачі був створений програмний продукт, який дозволив наглядно продемонструвати аналіз відгуків інформаційного порталу. Процентний показник, який ми отримали при навчанні моделі, вказує на високу точність результатів в розмірі 89,8%.

Аналіз настрою є дуже актуальною темою і можливий для використання у різних галузях і сферах, таких як медицина, навчальна, політична, фінансова та комерційна.

В ході експериментів по векторному поданню і тестуванню алгоритмів глибокого навчання зазначалося, що кожна з цих областей є великим науковим полем для досліджень і фізично не може бути досконально розглянуте в рамках однієї атестаційної роботи. Фокус даної роботи сприяє прийняттю методів deep learning в аналізі настроїв користувачів соціальних мереж, на основі залишених ними коментарів, тому більш глибокі дослідження і розробки окремих моделей і алгоритмів є напрямками подальшої роботи. В якості продовження даної роботи можна сформувати задачі для навчання класифікатору з більшою кількістю класів.

Програмні засоби готові до практичного використання для аналізу класифікації настроїв користувачів соціальних мереж.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Сегаран Т. Программируем коллективный разум. Санкт-Петербург : Символ-Плюс, 2008. 368 с.

2. Абрамов М. В., Вахромеева А. В. Программное обеспечение для мониторинга социальных медиа (на примере анализа ситуации в Украине) // IV международная социологическая конференция «Продолжая Грушина» (г. Москва, 27-28 февраля 2014 г.). Москва : ВЦИОМ, 2014. С. 138–140.

3. Kramer A. D., Guillory J. E., Hancock J. T. Experimental evidence of massivescale emotional contagion through social networks // Proceedings of the National Academy of Sciences. 2014. № 111 (24). P. 8788–8790.

4. Козырев Г. И. Общественное мнение в структуре власти и управления // Третья международная научно-практическая социологическая конференция «Продолжая Грушина» (г. Москва, 28 февраля – 1 марта 2013 г.). Москва : ВЦИОМ, 2013. С. 314–317.

5. Тавокин Е. П. Основы методики социологического исследования: учеб. пос. Москва : ИНФРА-М, 2009. 239 с.

6. Девятко И. Ф. Инструментарий онлайн-исследований: попытка каталогизации // Онлайн исследования в России 3.0. Москва : Издательский дом «Кодекс», 2012. С. 17–30.

7. Морозова Ю. И. Построение семантических векторных пространств различных предметных областей // Третья школа молодых ученых ИПИ РАН : сб. докладов. Москва : ИПИ РАН, 2012. С. 4–11.

8. Mikolov T. Efficient estimation of word representations in vector space // arXiv preprint arXiv. 2013. № 1301 (3781). P. 285–297.

9. Rong X. word2vec Parameter Learning Explained // arXiv preprint arXiv. 2014. № 1411 (2738). P. 2671–2679.

10. Mikolov T. Distributed representations of words and phrases and their compositionality // Advances in Neural Information Processing Systems. 2013. P. 3111–3119.

11. LeCun Y. Gradient-based learning applied to document recognition // Proceedings of the IEEE. 1998. № 86 (11). P. 2278–2324.

12. Kim Y. Convolutional neural networks for sentence classification // arXiv preprint arXiv. 2014. № 1408 (5882). P. 1746–1751.

13. Heigold G., Neumann G., van Genabith J. Neural morphological tagging from characters for morphologically rich languages // arXiv preprint arXiv. 2016. № 1606 (06640). P. 9–10.

14. Zhang X., Zhao J., LeCun Y. Character-level Convolutional Networks for Text Classification // arXiv preprint arXiv. 2017. № 1509 (01626). P. 9–10.

15. Very Deep Convolutional Networks for Text Classification / Conneau A., Schwenk H., Barrault L. [et al.] // arXiv preprint arXiv. 2016. № 1606 (01781). P. 10–11.

16. A Practitioners' Guide to Transfer Learning for Text Classification using Convolutional Neural Networks / Semwal T., Mathur G., Yenigalla P. [et al.] // arXiv preprint arXiv. 2018. № 1801 (06480). P. 9–10.

18. Bai S., Kolter J. Z., Koltun V. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling // arXiv preprint arXiv. № 1803 (01271). P. 14–15.