КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра теоретичної кібернетики

Роботу розглянуто й допущено до захисту на
засіданні кафедри теоретичної кібернетики
« » травня 2021 р.,
протокол №
Завідувач кафедри
Ю. В. Крак _____
(підпис)

Випускна кваліфікаційна робота

на здобуття ступеня бакалавра

за спеціальністю 122 Комп'ютерні науки
на тему:

Класифікація і кластеризація текстової інформації

Виконав студент 4 курсу
Костін Володимир Олександрович

_____
(підпис)

Науковий керівник:
професор, доктор фізико-математичних наук
Крак Юрій Васильович

_____
(підпис)

Засвідчую, що в цій дипломній роботі немає
запозичень з праць інших авторів без
відповідних посилань.

Студент _____
(підпис)

Київ – 2021

ЗМІСТ

СКОРОЧЕННЯ

ЕОМ- Електронна обчислювальна машина
NLP- Natural Language Processing
Ai- Artificial intelligence
JSON- JavaScript Object Notation
RNN- recurrent neural network
LSTM- long short-term memory
FNN- Feedback Neural Network
IDS- intrusion detection systems
NARX- нелінійна авторегресійна екзогенна модель

ВСТУП

Робота з інформацією є не від'ємною частиною нашого життя.

Чи то її пошук або навпаки запис. Більша частина інформації зберігається у текстовому вигляді. Однак це віднімає також багато часу.

Крім того не завжди є можливість вдало знайти потрібну інформацію.

Тим важливіше вдало систематизувати її. А класифікація і відповідно кластеризація є найкращими способом вирішити це питання.

Мета й завдання роботи.

Метою кваліфікаційної роботи є вивчення використання рекурентних методів глибокого навчання (Long short term memory, Recurrent neural network та варіації) для класифікації ланцюжків обмеженої довжини у текстах із підтримкою обчислень CUDA

Об’єкт, методи й засоби дослідження та розробки.

Об’єктом дослідження є ELSEVIER OA CC-BY CORPUS.

Мова розробки Python

середовище розробки PyCharm

Джерело текстової інформації ELSEVIER OA CC-BY CORPUS.

Можливі сфери застосування.

Розробка може сприяти покращення робот з текстовою інформацією.

1. СУЧАСНИЙ СТАН ЗАДАЧІ КЛАСИФІКАЦІЇ І КЛАСТЕРИЗАЦІЇ ТЕКСТОВОЇ ІНФОРМАЦІЇ

1.1 Поняття інформації

Інформація (від лат. Informātiō «роз'яснення, уявлення, поняття про що-небудь» ← informare «надавати вид, форму, навчати; мислити, уявляти» [1]) - відома незалежно від форми їх подання. Незважаючи на широку поширеність, поняття інформації залишається одним з найбільш дискусійних в науці, а термін може мати різні значення в різних галузях людської діяльності. Визначень інформації існує безліч, причому академік М. М. Моісеєв навіть вважав, що в силу широти цього поняття немає і не може бути суворого і досить універсального визначення інформації.

У міжнародних стандартах даються такі визначення:

• знання про предмети, факти, ідеї і т. д., якими можуть обмінюватися люди в рамках конкретного контексту; [2]

• знання щодо фактів, подій, речей, ідей і понять, які в певному контексті мають конкретний зміст; [3]

• відомості, що сприймаються людиною (або) спеціальними пристроями як віддзеркалення фактів матеріального або духовного світу в процесі комунікації.[4]

Хоча інформація повинна набути деякої форму подання (тобто перетворитися в дані), щоб нею можна було обмінюватися, інформація є в першу чергу інтерпретація (сенс) такого подання (ISO / IEC / IEEE 24765: 2010). Тому в строгому сенсі інформація відрізняється від даних, хоча в неформальному контексті ці два терміни дуже часто використовують як синоніми.

Спочатку «інформація» - відомості, що передаються людьми усним, письмовим або будь-яким іншим способом (за допомогою умовних сигналів, технічних засобів і т. Д.); з середини XX століття термін «інформація» перетворився в загальнонаукове поняття, що включає обмін відомостями між

людьми, людиною і автоматом, автоматом і автоматом; обмін сигналами в тваринному і рослинному світі; передачу ознак від клітини до клітини, від організму до організму (наприклад, генетична інформація); одне з основних понять кібернетики. [5]

1.1.1 Суть і кордони явища

Відповідно до сучасних уявлень, інформація вважається нематеріальної, а то, що міститься в структурі об'єктів, прийнято називати даними (representation form - ISO / IEC / IEEE 24765: 2010).

Для досліджень самоорганізації динамічних систем Генрі Кастлер запропонував таке визначення: «Інформація є запомненний вибір одного варіанта з декількох можливих і рівноправних».

1.1.2 Історія поняття

Слово «інформація» походить від лат. informatio, що в перекладі означає зведення, роз'яснення, ознайомлення. Поняття інформації розглядалося ще античними філософами.

Латинські слова «de saxis informibus» з Вульгати Ієроніма (342-419) переводяться як «з цілого каміння» (Втор. 27: 6), а слова «informem adhuc me», які переводяться як «Мого зародка» (Пс. 138: 16), можна перевести і як «аморфного ще мене», бо саме як «ще безформна» переводяться слова «adhuc informem» з Сповіді Августина (354-430).

Італійським словом «informa» в Комедії Данте (1265-1321) позначається вже не просто безформне, а процес формування, освіти, творіння (Ч. XVII 16-18, Ч. XXV 40-42, Р. VII 133-138).

У сучасному світі інформація - один з найважливіших ресурсів і в той же час одна з рушійних сил людського суспільства. Інформаційні процеси, що відбуваються в матеріальному світі, живій природі і людському суспільстві, вивчаються (або, принаймні, беруться до уваги) всіма науковими дисциплінами, від філософії до маркетингу. Історично в вивченні інформації

безпосередньо брали участь дві комплексні галузі науки - кібернетика та інформатика. Інформатика, сформувалася як наука в середині XX століття, відокремилася від кібернетики і займається дослідженнями в області методів отримання, зберігання, передачі та обробки семантичної інформації. Вивчення змісту інформації засноване на сукупності наукових теорій під загальною назвою семіотика.

В СРСР філософська проблематика поняття «інформація» розроблялася, починаючи з 1960-х років, коли вийшла стаття А. Д. Урсула «Природа інформації». З тих пір, явно чи неявно, розглядаються в основному дві концепції інформації: атрибутивна, по якій інформація властива всім фізичним системам і процесам (А. Д. Урсул, І. Б. Новік, Л. Б. Баженов, Л. А. Петрушенко та інші), і функціональна - інформація властива лише самоорганизующимся системам (П. В. Копнін, А. М. Коршунов, В. С. Тюхтін, Б. С. Українців і інші).

Але якщо, наприклад, провести навіть поверхневий аналіз змісту найбільш поширених сьогодні атрибутивних і функціональних понять інформації, стає ясно, що обидва ці поняття, в кінцевому рахунку, базуються на об'єктивному властивості матерії, встановленому в XIX столітті і порушені філософською категорією «рефлексія».

Однак обидві концепції не приділяють достатньої уваги вивченню очевидної реальності, що проявляється в тому, що інформація в тих формах, в яких вона існує сьогодні, є продуктом людської свідомості, яка сама є продуктом вищих форм (відомі форми) матерії.

Іншими словами, прихильники обох концепцій, ігноруючи людини, ігноруючи природу людської свідомості, негайно відносять інформацію (продукт свідомості) до властивостей матерії і відразу ж називають її «атрибутом матерії». В результаті цієї помилки обидві концепції не можуть дати нам суворого визначення інформації як концепції, тому що людські концепції наповнюються сенсом в результаті людського спілкування з

об'єктивною реальністю, а не в результаті дії, хоча і тонкого, що здаються переконливими, правдоподібні висновки, інші концепції.

Спроби представити інформацію у вигляді категорії також приречені на провал. Досить взяти до уваги, що людська практика за останні десятиліття так швидко змінила форму і зміст концепцій, а також їх ідеї та ставлення до того, що зараз називається «інформацією», що характер, сутність інформації і, звичайно ж,, значення цього поняття до сих пір вважається поняттям) істотно змінилися з плином часу.

1.2 Вимірювання інформації

Піонером в області інформаційної теорії був Ральф Хартлі. Він ввів поняття «інформації» (ентропії) як випадкової змінної і був першим, хто спробував визначити «міру інформації».

Найпростішою одиницею виміру інформації є біт - одиниця виміру кількості інформації, яка бере 2 логічних значення: так чи ні, істина або брехня, увімкнене; 1 або 0 в двійковій системі числення.

В сучасних обчислювальних системах одномоментно обробляються 8 біт інформації, звані байтом. Байт може приймати одне з 256 (2^8)різних значень (станів, кодів). Похідні від байта десяткові одиниці вимірювання, відповідно, називаються кілобайт (10^3 = 1000 байт),мегабайт (10^6 = 1 000 000 байт), гігабайт (10^9 = 1 000 000 000 байт)і т. Д.[6].

Похідні від байта виконавчі (бінарні) одиниці виміру іменуються кібібайт (2^{10} = 1024 байт), мебібайт (2^{20} = 1 048 576 байт), Гібібайт (2^{30} = 1 073 741 824 байт) і так далі[6].

Також, інформацію вимірюють такими одиницями як тритій, що Хартдіт (деціт) і Нат.

Кількість байтів				
Префікси СІ			Бінарні префікси	
Назва (Скорочення)	Префікс СІ	Альтернативне Використання	Назва (Скорочення)	Значенн
кілобайт (кБ)	10^3	2^{10}	кібібайт (КіБ)	2^{10}
мегабайт (МБ)	10^6	2^{20}	мебібайт (МіБ)	2^{20}
гігабайт (ГБ)	10^9	2^{30}	гібібайт (ГіБ)	2^{30}
терабайт (ТБ)	10^{12}	2^{40}	тебібайт (ТіБ)	2^{40}
петабайт (ПБ)	10^{15}	2^{50}	пебібайт (ПіБ)	2^{50}
ексабайт (ЕБ)	10^{18}	2^{60}	ексбібайт (ЕіБ)	2^{60}
зетабайт (ЗБ)	10^{21}	2^{70}	зебібайт (Зіб)	2^{70}
йотабайт (ЙБ)	10^{24}	2^{80}	йобібайт (ЙіБ)	2^{80}

Рисунок 1

1.3 Класифікація інформації

Інформацію можна розділити на види за різними критеріями:

За способом сприйняття:

• Візуальна - сприйняття органами зору.

• Звукова - сприйнята органами слуху.

• Тактильна - сприйнята тактильними рецепторами.

• Нюхова - сприйнята нюховими рецепторами.

• Смакова - сприйнята смаковими рецепторами.

За формою подання:

• Текстова - що передається у вигляді символів, призначених позначати лексеми мови.

• Числова - у вигляді цифр і знаків (символів), що позначають математичні дії.

• Графічна - у вигляді зображень, предметів, графіків.

Звукова - усна або у вигляді запису і передачі лексем мови аудіальним шляхом.

• Відеоінформація - передана у вигляді відеозапису.

По призначенню:

• Масова - містить тривіальні відомості і оперує набором понять, зрозумілим більшій частині соціуму.

• Спеціальна - містить певний набір понять, при використанні якого відбувається передача інформації, яка може бути незрозумілою для більшості суспільства, але необхідною і зрозумілою в вузької соціальної групи, де ця інформація використовується.

• Секретна - передана вузькому колу осіб і за закритими (захищеним) каналам.

• Особиста (приватна) - набір відомостей про яку-небудь особистості, що визначає соціальний стан і типи соціальних взаємодій всередині популяції.

За значенням:

• Актуальна - інформація, цінна в даний момент часу.

• Достовірна - інформація, отримана без спотворень з надійних джерел.

• Зрозуміла - інформація, виражена мовою, зрозумілою того, кому вона призначена.

• Повна - інформація, достатня для прийняття правильного рішення або розуміння.

• Цінна - корисність інформації визначається суб'єктом, який отримав інформацію в залежності від обсягу можливостей її використання.

За істинності:

• Справжня.

• Хибна.

В основу класифікації покладено п'ять загальних ознак:

• Місце виникнення.

• Стадія обробки.

• Режим перегляду.

• Стабільність.

• Функція управління.

1.4 Інформація в різних областях діяльності

1.4.1 В інформатиці

Предметом вивчення інформатики є дані: способи їх створення, зберігання, обробки і передачі[7]. Дані - це інформація в формалізованому вигляді (в цифровому вигляді), що дозволяє автоматизувати її збір, зберігання і подальшу обробку на комп'ютері.

З цієї точки зору інформація є абстрактним поняттям, що розглядаються безвідносно до її семантичному аспекту, а під кількістю інформації зазвичай розуміється відповідний обсяг даних. Однак одні й ті ж дані можуть бути закодовані по-різному і мати при цьому різний обсяг, тому іноді розглядається також поняття «цінність інформації», яке пов'язане з поняттям інформаційної ентропії і є предметом вивчення теорії інформації.

1.4.2 В теорії інформації

Теорія інформації (математична теорія зв'язку) - вивчає процеси зберігання, перетворення і передачі інформації. Вона заснована на наукових методах вимірювання кількості інформації[8]. Теорія інформації розвинулася з потреб теорії зв'язку. Основоположними вважаються «Передача інформації[9]» Ральфа Хартлі опубліковані в 1928 році і «Роботи по теорії інформації і кібернетики[20]» Клода Шеннона, опубліковані в 1948 році. Теорія інформації вивчає межі можливостей систем передачі даних, а також основні принципи їх проектування і технічної реалізації[10].

З теорією інформації пов'язані радіотехніка (теорія обробки сигналів) і інформатика, що відносяться до вимірювання кількості інформації, що передається, її властивості і якими встановлено граничні співвідношення для систем. Основні розділи теорії інформації - кодування джерела (стискаючий кодування) і канальне (завадостіке) кодування. Інформація не входить в число предметів дослідження математики. Проте, слово «інформація» вживається в математичних термінах - власна інформація і взаємна інформація, що відносяться до абстрактної (математичної) частини теорії

інформації. Однак, в математичної теорії поняття «інформація» пов'язано з виключно абстрактними об'єктами - випадковими величинами, в той час як в сучасній теорії інформації це поняття розглядається значно ширше - як властивість матеріальних об'єктів. Зв'язок між цими двома однаковими термінами безсумнівна. Саме математичний апарат випадкових чисел використовував автор теорії інформації Клод Шеннон. Сам він має на увазі під терміном «інформація» щось фундаментальне (нередуціруемого).

В теорії Шеннона інтуїтивно покладається, що інформація має зміст. Інформація зменшує загальну невизначеність і інформаційну ентропію. Кількість інформації доступно виміру. Однак він застерігає дослідників від механічного перенесення понять з його теорії в інші області науки.

1.4.3 В теорії управління (кібернетики)

Засновник кібернетики Норберт Вінер дав таке визначення інформації: «Інформація - це позначення змісту, отримане нами з зовнішнього світу в процесі пристосування до нього нас і наших почуттів»[11].

Кібернетика розглядає машини і живі організми як системи, що сприймають, накопичують і передають інформацію, а також переробні її в сигнали, що визначають їх власну діяльність[12]. Матеріальна система в кібернетиці розглядається як безліч об'єктів, які самі по собі можуть перебувати в різних станах, але стан кожного з них визначається станами інших об'єктів системи. У природі безліч станів системи є інформацію, самі стану являють собою первинний код або код джерела. Таким чином, кожна матеріальна система є джерелом інформації.

Суб'єктивну (семантичну) інформацію кібернетика визначає як зміст або зміст повідомлення. Інформація - це характеристика об'єкта.

1.5 Кластеризація

Кластеризація (англ. Cluster analysis) - завдання угруповання безлічі об'єктів на підмножини (кластери) таким чином, щоб об'єкти з одного кластера були більш схожі один на одного, ніж на об'єкти з інших кластерів по якомусь критерію. Завдання кластеризації відноситься до класу задач навчання без учителя.

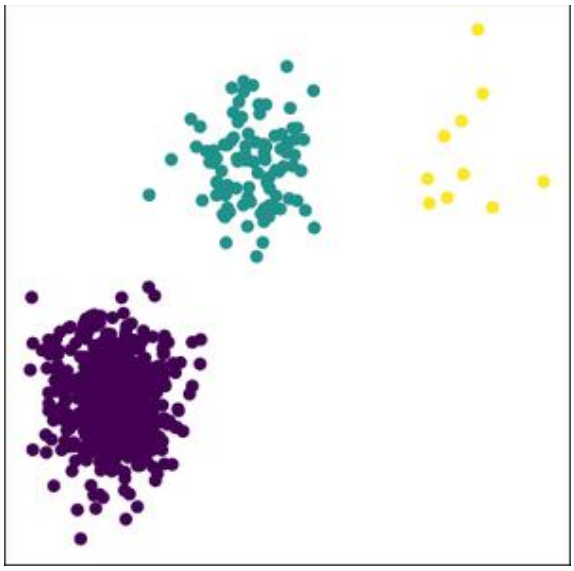

Рисунок 3

1.5.1 Постановка завдання кластеризації

Нехай X - безліч об'єктів, Y - безліч ідентифікаторів (міток) кластерів. На множині X задана функція відстані між об'єктами ρ (x, x '). Дана кінцева навчальна вибірка об'єктів Xm = {x1, ..., xm} ⊂X. Необхідно розбити вибірку на підмножини (кластери), тобто кожному об'єкту xi∈Xm зіставити мітку yi∈Y, таким чином щоб об'єкти всередині кожного кластера були близькі щодо метрики ρ, а об'єкти з різних кластерів істотно розрізнялися.

Визначення:

Алгоритм кластеризації - функція a: X → Y, яка будь-якого об'єкта x∈X ставить у відповідність ідентифікатор кластера y∈Y.

Безліч Y в деяких випадках відомо заздалегідь, однак частіше ставиться завдання визначити оптимальне число кластерів, з точки зору того

чи іншого критерію якості кластеризації. Кластеризація (навчання без вчителя) відрізняється від класифікації (навчання з учителем) тим, що мітки об'єктів з навчальної вибірки уі спочатку не задані, і навіть може бути невідомо саме безліч Y.

Рішення завдання кластеризації об'єктивно неоднозначно по ряду причин:

• Не існує однозначної критерію якості кластеризації. Відомий ряд алгоритмів, які здійснюють розумну кластеризації "з побудови", проте всі вони можуть давати різні результати. Отже, для визначення якості кластеризації і оцінки виділених кластерів потрібен експерт предметної області;

• Число кластерів, як правило, заздалегідь не відомо і вибирається за суб'єктивними критеріями. Навіть якщо алгоритм не вимагає початкового знання про число класів, конкретні реалізації часто вимагають вказати цей параметр[13];

• Результат кластеризації істотно залежить від метрики. Однак існує ряд рекомендацій по вибору метрик для певних класів задач.[14].

Число кластерів фактично є гіперпараметром для алгоритмів кластеризації.[15].

1.5.2 Типологія завдань кластеризації

Типи вхідних даних

• Признаковий опис об'єктів. Кожен об'єкт описується набором своїх характеристик, які називаються ознаками (англ. Features). Ознаки можуть бути як числовими, так і категоріальним;

• Матриця відстаней між об'єктами. Кожен об'єкт описується відстанню до всіх об'єктів з навчальної вибірки.

Обчислення матриці відстаней по признаковому опису об'єктів може бути виконано нескінченним числом способів в залежності від визначення

метрики між об'єктами. Вибір метрики залежить від навчальної вибірки і поставленого завдання.

Цілі кластеризації:

• Класифікація об'єктів. Спроба зрозуміти залежності між об'єктами шляхом виявлення їх кластерної структури. Розбиття вибірки на групи схожих об'єктів спрощує подальшу обробку даних і прийняття рішень, дозволяє застосувати до кожного кластеру свій метод аналізу (стратегія «розділяй і володарюй»). В даному випадку прагнуть зменшити число кластерів для виявлення найбільш загальних закономірностей;

• Стиснення даних. Можна скоротити розмір вихідної вибірки, взявши один або кілька найбільш типових представників кожного кластера. Тут важливо найбільш точно окреслити межі кожного кластера, їх кількість не є важливим критерієм;

• Виявлення новизни (виявлення шуму). Виділення об'єктів, які не підходять за критеріями ні в один кластер. Виявлені об'єкти в подальшому обробляють окремо.

Методи кластеризації:

• Графові алгоритми кластеризації. Найбільш примітивний клас алгоритмів. В даний час практично не застосовується на практиці;

Імовірнісні алгоритми кластеризації. Кожен об'єкт з навчальної вибірки відноситься до кожного з кластерів з певним ступенем імовірності:

• ЕМ-алгоритм;

• Ієрархічні алгоритми кластеризації. Упорядкування даних шляхом створення ієрархії вкладених кластерів;

Алгоритм k-середній (англ. K-means). Ітеративний алгоритм, заснований на мінімізації сумарного квадратичного відхилення точок кластерів від центрів цих кластерів;

• Поширення схожості (англ. Affinity propagation). Поширює повідомлення про схожості між парами об'єктів для вибору типових представників кожного кластера;

Зрушення середнього значення (англ. Mean shift). Вибирає центроїди кластерів в областях з найбільшою щільністю;

• Спектральна кластеризація (англ. Spectral clustering). Використовує власні значення матриці відстаней для зниження розмірності перед використанням інших методів кластеризації;

• Заснована на щільності просторова кластеризация для додатків з шумами (англ. Density-based spatial clustering of applications with noise, DBSCAN). Алгоритм групує в один кластер точки в області з високою щільністю. Самотньо розташовані точки позначає як шум.

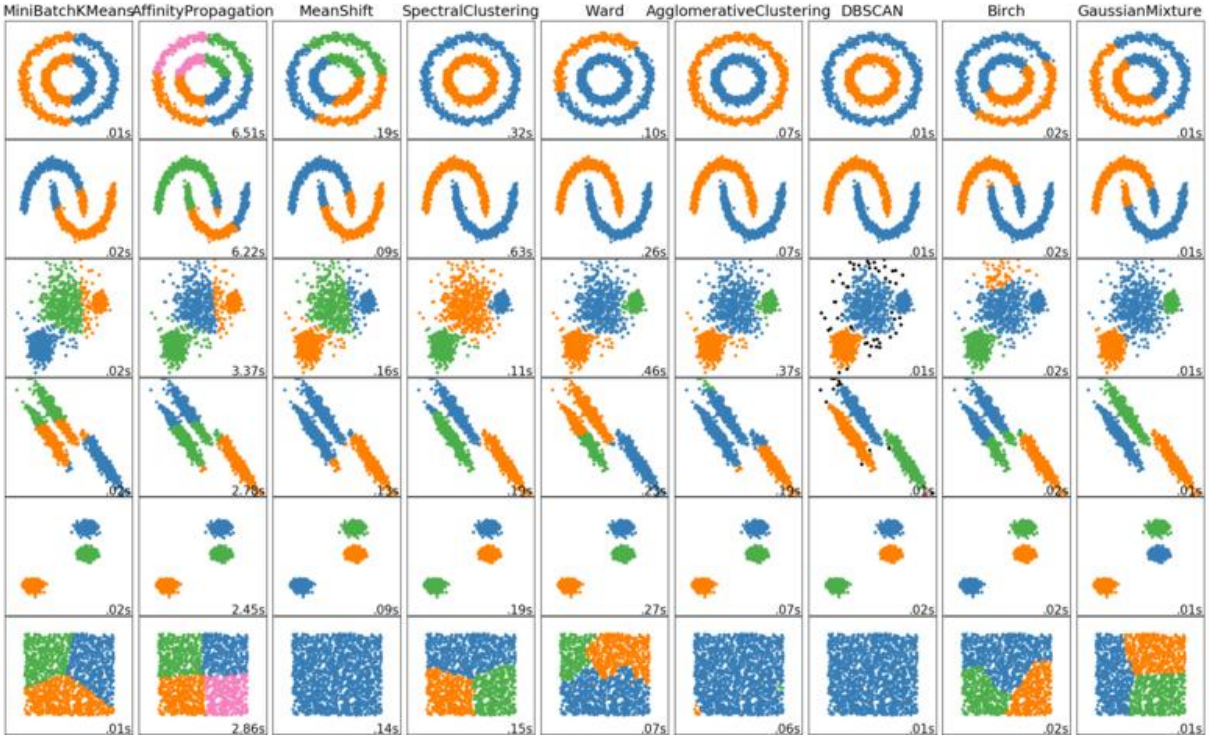

Рисунок 7 - Порівняння алгоритмів кластеризації з пакета scikit-learn

1.5.3 Застосування

.Інтернет

• Виділення груп людей на основі графа зв'язків в соціальних мережах;

• Підвищення релевантності відповідей на пошукові запити шляхом угруповання веб-сайтів по смисловим значенням пошукового запиту

Комп'ютерні науки

• Кластеризація використовується в сегментації зображень для визначення меж і розпізнавання об'єктів;

• Кластерний аналіз застосовується для визначення утворилися популяційних ніш в ході роботи еволюційних алгоритмів для поліпшення параметрів еволюції;

• Підбір рекомендацій для користувача на основі переваг інших користувачів в даному кластері;

• Визначення аномалій шляхом побудови кластерів та виявлення некласифікованих об'єктів.

1.6 Класифікація

Класифікація, (від лат. Classis «розряд» і facere «робити») - поняття в науці (в філософії, в формальній логіці і ін.), Що позначає різновид розподілу обсягу поняття за певним основи (ознакою, критерієм), при якому обсяг родового поняття (клас, безліч) ділиться на види (підкласи, підмножини), а види, в свою чергу діляться на підвиди і т.д.

Опис.

Класифікація широко застосовується як в науці (особливо, в природничих науках), так і в практичній діяльності, причому наукові класифікації відрізняються більш стійким характером, тому зберігаються довгий час[16][17]. Наприклад, класифікація хімічних елементів, створена Д. І. Менделєєвим продовжує доповнюватися донині. У класифікації важливе значення має вибір підстави (критерію, ознаки) поділу предмета. Підстава може бути істотним і неістотним. Класифікація виконана щодо істотного ознакою називається природною, класифікація виконана по несуттєвому ознакою - штучна (або, допоміжна) класифікація[18]. Одна з труднощів, яка виникає при класифікації є перехідна форма. Наприклад, при класифікації прав і свобод людини і громадянина свобода слова може бути віднесена як до природних (вродженим) прав, так і до політичних прав. Залежно від широти класифікації можуть бути енциклопедичними (універсальними) і спеціальними (галузевими), що включають класифікації вузького кола однорідних явищ[19].

Правила класифікації (ділення обсягу поняття)

Так як класифікація є різновидом поділу поняття, то їй притаманні всі правила, які використовуються при операції ділення обсягу понять.

1. Як і при розподілі, класифікацію необхідно проводити тільки по одному конкретному основи. Якщо дане правило буде порушено, то відбудеться перетин понять. Наприклад, в розподілі «Папір ділиться на білу, чорну, товсту, тонку» допущена помилка, так як розподіл вироблено не по одній підставі, а відразу за двома. Тобто, перша підстава - колір, другим підставою є товщина. Так, папір може бути білою і товстої, чорної і тонкої, або навпаки.

2. Необхідно дотримуватися співмірність поділу, тобто сума членів класифікації повинна дорівнювати обсягу родового поняття (класу, множини). Можливі помилки при недотриманні даного правила: * Неповна (вузька) класифікація. Тобто обсяг видових понять в результаті класифікації не вичерпують обсяг діленого поняття. Наприклад, в класифікації «Літературні жанри за змістом поділяються на трагедії, комедії, жахи» не вказано жанр - драма. * Класифікація із зайвими видовими поняттями. Прикладом даного виду помилок є поділ «Комп'ютери діляться на настільні, мобільні, переносні і персональні», в якому «персональні» комп'ютери є зайвим видовим поняттям.

3.Члени класифікації повинні взаємно виключати один одного.

4.Підрозділ на підкласи має бути безперервним.

1.6.1 Завдання класифікації

Завдання класифікації - завдання, в якій є безліч об'єктів (ситуацій), розділених деяким чином на класи. Визнач кінцеве безліч об'єктів, для яких відомо, до яких класів вони належать. Це безліч називається вибіркою. Класова приналежність інших об'єктів невідома. Потрібно побудувати алгоритм, здатний класифікувати (див. Нижче) довільний об'єкт з початкової множини.

Класифікувати об'єкт - значить, вказати номер (або найменування) класу, до якого належить даний об'єкт.

Класифікація об'єкта - номер або найменування класу, що видається алгоритмом класифікації в результаті його застосування до даного конкретного об'єкту.

У математичній статистиці завдання класифікації називаються також завданнями дискримінантного аналізу. У машинному навчанні завдання класифікації вирішується, зокрема, за допомогою методів штучних нейронних мереж при постановці експерименту у вигляді навчання з учителем.

Існують також інші способи постановки експерименту - навчання без учителя, але вони використовуються для вирішення іншого завдання - кластеризації або таксономії. У цих завданнях поділ об'єктів навчальної вибірки на класи не задається, і потрібно класифікувати об'єкти тільки на основі їх подібності між собою. У деяких прикладних областях, і навіть в самій математичній статистиці, через близькість завдань часто вже не розрізняють завдання кластеризації від завдань класифікації.

Деякі алгоритми для вирішення задач класифікації комбінують навчання з учителем з навчанням без учителя, наприклад, одна з версій нейронних мереж Кохонена - мережі векторного квантування, яких навчають з учителем.

1.6.2 Типологія завдань класифікації

Типи вхідних даних

• Признаковий опис - найбільш поширений випадок. Кожен об'єкт описується набором своїх характеристик, які називаються ознаками. Ознаки можуть бути числовими або нечислових.

• Матриця відстаней між об'єктами. Кожен об'єкт описується відстанями до всіх інших об'єктів навчальної вибірки. З цим типом вхідних

даних працюють деякі методи, зокрема, метод найближчих сусідів, метод парзеновского вікна, метод потенційних функцій.

• Часовий ряд або сигнал являє собою послідовність вимірювань в часі. Кожен вимір може представлятися числом, вектором, а в загальному випадку - признаковим описом досліджуваного об'єкта в даний момент часу.

Зображення або відеоряд.

• Зустрічаються і більш складні випадки, коли вхідні дані подаються у вигляді графів, текстів, результатів запитів до бази даних, і т. Д. Як правило, вони наводяться до першого або другого випадку шляхом попередньої обробки даних і вилучення ознак.

Класифікацію сигналів і зображень називають також розпізнаванням образів.

Типи класів:

• Двокласова класифікація. Найбільш простий в технічному відношенні випадок, який служить основою для вирішення більш складних завдань.

• Багатокласова класифікація. Коли число класів досягає багатьох тисяч (наприклад, при розпізнаванні ієрогліфів або злитого мовлення), завдання класифікації стає істотно більш важкою.

• Непересічні класи.

• Пересічні класи. Об'єкт може належати одночасно до кількох класів.

• Нечіткі класи. Потрібно визначати ступінь належності об'єкта кожному з класів, зазвичай це дійсне число від 0 до 1.

2. РОЗРОБКА

2.1 Постановка завдань та цілі

Завдання - створити програми для виконання завдань класифікації та кластеризації текстової інформації. В якості джерела текстової інформації буде використовуватися онлайн-база даних ELSEVIER OA CC-BY CORPUS.

2.2 Обговорення вибору платформ та мовного програмування

В якості джерела текстової інформації було обрано онлайн базу даних ELSEVIER OA CC-BY CORPUS.

В якості програмного середовища використано PyCharm.

Для розробки програмного забезпечення було обрано мову Python.

2.2.1 ELSEVIER OA CC-BY CORPUS

2.2.1.1 Вступ

CORPUS Elsevier OA CC-BY для досліджень NLP та AI. Цей корпус містить 40 тис. (40, 091) статей про відкритий доступ (OA) CC-BY з журналів Elsevier, що представляють широкомасштабний, міждисциплінарний набір даних досліджень для підтримки досліджень NLP та МЛ. Дослідження застосування NLP та машинного навчання до наукового змісту привернули значну увагу в останні роки. Однак прогрес стримувався через обмежену доступність великих міждисциплінарних наборів даних. Випуск цього набору даних, мав миту допомогти дослідницькій спільноті в їх роботі щодо розширення розуміння спільності та відмінності між обробкою наукового тексту та тексту іншого характеру (наприклад, текст новин). Більше того, цей набір даних дозволяє досліджувати проблеми при обробці наукового тексту, яких не існує для інших типів даних.

2.2.1.2 Пов’язана робота

Набори даних з відкритим кодом академічних статей не є новими, примітними корпусами є корпус Pub-Med OA1 , CiteSeerX[20] , ACL[21] та arXiv[22]. Згадані вище корпуси представляють мільйони академічних статей, вільно доступних в Інтернеті в будь-якому PDF, LATEX або вільний текст. Однак для цих статей метадані зазвичай не вирішені, що означає встановлення зв'язку між документами за допомогою цитування є складним завданням. Щоб подолати це, такі корпуси як випущено графік літератури в семантичній науці[23], який нещодавно був замінений S2ORC[23]. Ці

корпуси містять документи з безлічі наборів даних (ACL, arXiv, PubMed · · ·), які потім були вирішені витягуючи зі статей посилання, місця та інші метадані. Однак оригінальні набори даних є упередженими до певних навчальних дисциплін та областей, таких як обчислювальна техніка. Отримані графіки літератури дозволяють картографування з високою роздільною здатністю зв'язків між записами, знайденими в академічних статтях, і, отже, для картографування стосунки між авторами, академічні концепції та місця публікацій, щоб назвати декілька. Подібними до S2ORC є набори даних, які були складені у відповідь на світові події, такі як COVID-19 [25]. Набір даних COVID-19 був отриманий із ряду вже існуючих наборів даних та доповнений додатковими опублікованими статей. Оскільки цей набір даних розроблено для певної причини, він упереджений до біологічного та медіального полів. Результати досліджень набору даних COVID19 є великими, включаючи розробка документів[26], переробку ліків[27] та виявлення фактора ризику[28]. Для всебічного огляду для підсумовування цього буде використана оглядова стаття внесок. Це, по суті, показує корисність конкретних доменних корпусів. Випуск цих типів наборів даних відкрив двері для розробки наборів даних для конкретних завдань, орієнтованих на кураторство дані для одного або декількох конкретних завдань. Наприклад, [29] оприлюднені марковані дані для розробки моделей для намірів цитування класифікація. Тоді[30] як випускаються анотовані статті для розробки систем вилучення інформації на рівні документів. Конкретні набори даних для розв'язування спільних посилань та вилучення сутності зазвичай походять із спільних завдань із семінарів такі як SemEval, TREC та BioNLP. Для цих завдань потрібні високоякісні анотації, які можуть бути складними (мається на увазі трудомістке і дороге) виготовлення. Відповідні набори даних, як правило, невеликі, високо куровані і часто зосереджуються на одному конкретному явищі, тематичному домені чи жанрі тексту. Дуже хороші огляди доступних наборів даних для типових завдань представлені в [31] (для розпізнавання іменованої сутності) та в

[32](для розпізнавання спільних посилань). Ці огляди показують, що набори даних, що містять академічну писемність, здебільшого обмежені біомедичною сферою. Більше того, обмежений обсяг цих наборів даних є серйозним обмеженням. Навіть коли моделі розроблені на основі цих даних показують багатообіцяючі результати, є обмежені можливості перевірки результатів у різних доменах або за допомогою більших корпорацій.

2.2.1.3 Вибірка даних

Корпус складається з наукових статей, опублікованих Elsevier з початку 2014 року, які є відкритим доступом (OA) та охоплюється ліцензією CC-BY 4.0.2 Для того, щоб створити збалансований набір даних з різних навчальних дисциплін, метод стратифікованої вибірки був використаний для досягнення рівного представництва з усіх дисциплін Elsevier, як це було представлено за кодами ASJC (All Science Journal Classification). Кодекси ASJC представляють наукову дисципліну журналів в Росії яку опублікувала стаття. Для спрощення цього 334 коди ASJC були згруповані в їх 27 предметів верхнього рівня класифікації.3 З кожного з 27 класів ASJC вищого рівня було відібрано 2000 документів. Кожна стаття може мати кілька ASJC коди, якщо статтю було обрано для одного класу, то її було видалено із залишку, що залишився. Отриманий зразок документи збалансовані між дисциплінами (див. таблицю 3 для розбивки документів на дисципліни) Базовим набором даних, з якого брали зразки статей, був очищений корпус. Це означає, що статті мали мати мінімум 20 речень, і вирок повинен бути "чистим", що означає, що якщо у реченні надмірна кількість XML або іншою мовою розмітки, тоді її було видалено зі статті. Можна вилучити максимум 20% вироку від артистичного, щоб бути включеним до базового корпусу.4

2.2.1.4 Data Structure

Кожен документ у корпусі міститься у власному файлі JSON. Назва файлу - це ідентифікатор статті. Дані для кожної статті структуровані, як описано у схемі JSON та описах полів нижче.

```
{
        " docId ": <str >,
              " metadata" : {
              " title ": <str >,
                    " authors" : [
                          {
                                " first ": <str >,
                                      " initial" : <str >,
                                      " last " : <str >,
                                       " email " : < str >
                                       },
                                  ...
                                      ] ,
                     " issn ": <str >,
                                       " volume " : <str >,
                                       " firstpage" : <str >,
                                       " lastpage" : <str >,
                                       " pub_year" : <int >,
                                       " doi " : <str >,
                                       " pmid " : <str >,
                                       " openaccess" : " Full ",
                                       " subjareas" : [<str >] ,
                                       " keywords" : [<str >] ,
                                       " asjc " : [<int >] ,
                                       },
                " abstract": [
                      {
                             " sentence": <str >,
                                   " startOffset" : <int >,
                                   " endOffset" : < int >
                                   },
                             ...
                                   ] ,
                                         " bib_entries":{
                                         < str > :{
                                                " title ":<str >,
                                                      " authors" : [
                                                            {
                                                                  " last ":<str >,
                                                                        " initial" : <str
>,
                                                                        " first " : < str >
                                                                        },
                                                                  ...
                                                                     ] ,
                                                      " issn ": <str >,
                                                                        " volume " : <str
>,
                                                                        " firstpage" : <str
>,
                                                                        " lastpage" : <str
>,
                                                                        " pub_year" : <int
>,
```

```
                                                    " doi " : <str >,
                                                    " pmid " : < str >
                                                    },
                                    ...
                                                    },
                              " body_text": [
                                    {
                                          " sentence": <str >,
                                                " secId " : <str >,
                                                " startOffset" : <int >,
                                                " endOffset" : <int >,
                                                " title " : <str >,
                                                " refoffsets" : {
                                                < str > :{
                                                      " endOffset":<int >,
                                                            " startOffset" : <
int >
                                                            }
                                                },
                                                 " parents": [
                                                      {
                                                       " id ": <str >,
                                                             " title " :
< str >
                                                       },
                                                      ...
                                                           ]
                                                },
                                    ...
                                                       ]
                                                      }
```

docId DocID - це ідентифікатор документа. Це унікально для документа і може бути перетворено на URL-адресу для документа шляхом додавання https // www.sciencedirect.com / science / pii / <docId>

abstract - це автор надав реферат до документа

body_text - Повний текст документа. Текст розділений на межі речень, що полегшує його використання в дослідницьких проектах. Кожне речення має заголовок (та ідентифікатор) розділу, з якого воно походить, разом із заголовками (та Ідентифікатори) батьківського розділу. Розділ найвищого рівня приймає індекс 0 у батьківському масиві. Якщо масив порожній, то заголовок розділу речення - це заголовок розділу найвищого рівня. Це дозволить реконструювати статтю структура. З речень витягнуто посилання. Ідентифікатори витягнутого посилання та їх відповідні зміщення в реченні можна знайти в полі “refoffsets”. Повний список літератури наведено в поле

"bib_entry" разом із відповідними метаданими посилань. Деякі з них будуть відсутні, оскільки ми підтримуємо лише "чистоту" речення,

bib_entities - це Всі посилання в документі можна знайти в цьому розділі. Якщо метадані для посилання доступне, воно додано до клавіші посилання. Де можливо, така інформація, як включені назви документів, автори та відповідні ідентифікатори (DOI та PMID). Ключі для кожного посилання можуть бути знайдено у реченні, де посилання використовується зі зміщенням початку та кінця де у реченні, що посилання було використано.

Metadata - це Метадані включають додаткову інформацію про статтю, таку як список авторів, відповідні посвідчення особи (DOI та PMID). Поряд із низкою класифікаційних схем, таких як ASJC та Предметна класифікація.

Author_highlights-це Основні моменти автора були включені до корпусу, де автор (автори) їх надав. охоплення становить 61% усіх статей. Автор виділяє основні моменти, що складаються з 4 до 6 речень, надані автором мета узагальнення основних висновків та результатів у статті.

2.2.1.5 Використання набору даних

Набір даних можна завантажити з Mendeley Data.5 Набір даних містить відформатовану версію JSON-файлу raw XML, до якого можна отримати доступ через API Elsevier.6 Оригінальні файли XML можна обробити за допомогою AnnotationQuery7 фреймворк, випущений Elsevier Labs. Необхідно використовувати наступне посилання при використанні набору даних:

@dataset{ https://10.17632/zm33cdndxs.3,
doi = {10.17632 / zm33cdndxs.2},
url =
{https://data.mendeley.com/datasets/zm33cdndxs/3},https://www.overleaf.com/project/5ef1aeeb7ff458000177cb45
author = "Daniel Kershaw and Rob Koeling",

keywords = {Science, Natural Language Processing, Machine Learning, Open Dataset},

title = {Elsevier OA CC - BY Corpus},

publisher = {Mendeley},

year = {2020},

month = aug

}

Таблиця 1: Покриття поля JSON	
Поля	Кількість статей
Abstract	99.25
Body_text	100.00
Author_highlight	61.31
Metadata	100.00
Metadata - issn	100.00
Metadata - firstpage	85.50
Metadata - lastpage	85.34
Metadata - pub_year	100.00
Metadata - doi	100.00
Metadata - openaccess	100.00
Metadata - subjectareas	100.00
Metadata - keywords	100.00
Metadata - asjc	100.00
Bib_entries	97.60

Таблиця 2: Розподіл років публікацій	
Рік публікації	Кількість статей
2014	3018
2015	4438
2016	5913
2017	6419
2018	8016
2019	10135
2020	2159

Таблиця 3: Розподіл статей за кодексом ASJC середнього рівня. Кожна стаття може належати до кількох кодів ASJC.	
Дисципліна	Кількість
Загальні	3847
Сільськогосподарські та біологічні науки	4840
Мистецтво та гуманітарні науки	982
Біохімія, генетика та молекулярна біологія	8356
Бізнес, управління та бухгалтерський облік	937
Хімічна інженерія	1878
Хімія	2490
Комп'ютерна наука	2039
Рішення наук	406
Науки про Землю і планети	2393
Економіка, економетрика та фінанси	976
Енергія	2730
Техніка	4778

Таблиця 4: Розподіл статей за кодексом ASJC середнього рівня. Кожна стаття може належати до кількох кодів ASJC.	
Environmental Science	6049
Імунологія та мікробіологія	3211
Матеріалознавство	3477
Математика	538
Медицина	7273
Неврологія	3669
Медсестринство	308
Фармакологія, токсикологія та фармацевтика	2405
Фізика та астрономія	2404
Психологія	1760
Соціальні науки	3540
Ветеринарні	991
Стоматологія	40
Медичні професії	821

2.2.1.6 Висновок по вибору CORPUS Elsevier OA CC-BY

Як видно з наведеної вище інформації CORPUS Elsevier OA CC-BY є онлайн базою з понад 40000 статей, яка супроводжується метаданими в XML та яку відповідно можна скачати.Основною його метою було допомогти дослідницькій спільноті в їх роботі.Крім того цей набір даних

дозволяє досліджувати проблеми при обробці наукового тексту, яких не існує для інших типів даних. І що не менш важливо він знаходиться у вільному доступі і будь хто може його скачати. Дивлячись на всі ці переваги було вирішено зупинити вибір саме на цій базі даних.

2.2.2 PyCharm

PyCharm - це інтегроване середовище розробки мови програмування Python. Насамперед надає засоби аналізу коду, графічний налагоджувач, інструмент для запуску юніт-тестів та підтримує веб-розробку Django.

PyCharm розроблений JetBrains [33] на базі IntelliJ IDEA.

PyCharm - це міжплатформене середовище розробки, сумісне з Windows, macOS, Linux. PyCharm Community Edition (безкоштовна версія) ліцензована за ліцензією Apache, а PyCharm Professional Edition (платна версія) - запатентоване програмне забезпечення(невільного програмного забезпечення)[34].

2.2.2.1 Історія

PyCharm був запущений на ринок інтегрованих середовищ розробки для Python, щоб конкурувати з PyDev (проте PyCharm в даний час використовує PyDev для налагодження коду) і більш поширеною середовищем розробки Komodo IDE. Бета-версія була випущена в липні 2010 року, версія 1.0 вийшла через три місяці.

Версія 2.0 вийшла 13 грудня 2011 року.

Версія 3.0 була випущена 24 вересня 2013 року.

PyCharm Community Edition, безкоштовна версія з відкритим вихідним кодом, була випущена 22 жовтня 2013 року.

У березні 2016 року JetBrains перейшла на модель ліцензування за передплатою, і з цим змінилася нумерація версій. Номер версії тепер виглядає як YYYY.R, де YYYY - рік випуску, а R –випуск на протягом цього року[35].

2.2.2.2 Ліцензування

PyCharm Professional Edition має кілька варіантів ліцензій, які розрізняються по функціональності, вартості та умов використання, і безкоштовні для освітніх установ і проектів з відкритим вихідним кодом.

Існує також безкоштовна версія Community Edition з урізаним набором функцій [36]. Поширюється за ліцензією Apache 2.

2.2.2.3 Модулі

Користувачі можуть писати свої власні плагіни, тим самим розширюючи можливості PyCharm. Деякі плагіни з інших IDE JetBrains можуть працювати з PyCharm. Існує більше тисячі плагінів, сумісних з PyCharm.

2.2.2.4 Можливості

•автодоповнення коду

•рефакторинг коду

•Підтримка Git, SVN, Mercurial та інших систем контролю версіями;

•Налагодження коду за допомогою PyDev;

2.2.2.5 Висновок по вибору PyCharm

PyCharm являє собою інтегроване середовище розробки мови програмування Python. І відповідно оскільки в якості мови програмування було обрано Python він був розглянутий одним із перших. І зважаючи на заручний інтерфейс а також великий асортимент модулів і враховуючи, що

PyCharm має бескоштовну ліцензіовану версію тому я важаю що краще зупинити вибір саме на цьому середовищі розробки

2.2.3 Python

Python - це високорівнева мова програмування загального призначення з динамічної строгою типізацією і автоматичним управлінням пам'яттю [37] [38], орієнтований на підвищення продуктивності розробника, читання і якості коду, а також на забезпечення переносимості програм, написаних на ньому.

Мова повністю об'єктно-орієнтований - всі об'єкти [37]. Незвичайною особливістю мови є виділення блоків коду з пробілами. Синтаксис ядра мови мінімалістичний, через що на практиці рідко доводиться звертатися до документації. Сам мова відома як інтерпретується і використовується, в тому числі для написання скриптів. Недоліками мови часто є більш низька швидкість і більше споживання пам'яті написаними на ньому програмами в порівнянні з аналогічним кодом, написаним на мовах програмування, таких як C або C ++ [39] [37].

2.2.3.1 Історія

Ідея реалізації мови з'явилася в кінці 1980-х років, а розробку її реалізації почав в 1989 році співробітник голландського інституту CWI Гвідо ван Россум [40]. Для розподіленої операційної системи Amoeba потрібний розширювана мова сценаріїв, і Гвідо почав розробку Python на дозвіллі, запозичивши деякий досвід для мови ABC (Гвідо брав участь в розробці цієї мови програмування, орієнтованого на програмування). У лютому 1991 року Гвідо опублікував джерело в групі новин alt.sources. З самого початку Python був розроблений як об'єктно-орієнтована мова.

3 грудня 2008 року [41], після тривалого тестування була випущена перша версія Python 3000 (або Python 3.0, також скорочено Py3k). Python 3000 усуває багато недоліків архітектури з максимально можливою (але не повною) сумісністю зі старими версіями Python.

Термін дії підтримки Python 2.7 був спочатку встановлений на 2015 рік, а потім перенесений на 2020 рік з побоювань, що велика частина існуючого

коду не може бути легко перенесена на Python 3. Підтримка Python 2 була націлена лише на існуючі проекти, в нових проектах слід використовувати Python 3 [42] [43]. Більше ніяких виправлень безпеки або інших поліпшень для Python 2.7 випускатися не буде [44] [45]. Коли закінчується термін дії Python 2.x, підтримуються тільки Python 3.6.x і новіше.

2.2.3.2 Модулі та пакети

Програмне забезпечення (додаток або бібліотека) на Python розроблено у вигляді модулів, які, в свою чергу, можуть бути зібрані в пакети. Модулі можуть перебувати як в каталогах, так і в ZIP-архівах. Модулі можуть бути двох типів за походженням: модулі, написані на «чистому» Python, і модулі розширення, написані на інших мовах програмування. Наприклад, в стандартній бібліотеці є «чистий» модуль pickle і його аналог на C: cPickle. Модуль оформлений як окремий файл, а пакет - як окремий каталог. Модуль підключається до програми оператором імпорту. Після імпорту модуль представлений окремим об'єктом, який дає доступ до простору імен модуля. Під час виконання програми модуль може бути перезапущений функцією reload ().

2.2.3.3 Можливості

Інтроспекція

Обробка винятків

Ітератори

Генератори Управління контекстом виконання

декоратори

Регулярні вирази

2.2.3.4 Висновок по вибору Python

Як видно з наведеної вище інформації Python являє собою дуже зручну мову для написання програм. Хоч він і містить деякі недоліки основним з яких є більш низька швидкість і більше споживання пам’яті у програмах написаних на ньому порівняно з аналогами написаних C або C ++. Він має і багато переваг основним для мене є широкий вибір модулів та пакетів.

та загалом він зручніший і простіший у використані ніж C ++ або Java. Саме тому мовою написання краще обрати Python.

3. Результати роботи

Було досліджено класифікації і кластеризаці текстової інформації.

Для дослідження класифікації і кластеризації роботи було обрано використання рекурентних методів глибокого навчання (Long short term memory, Recurrent neural network та варіації) для класифікації ланцюжків обмеженої довжини у текстах із підтримкою обчислень CUDA.

Рекурентні методи глибокого навчання (RNN) - це клас штучних нейронних мереж, в яких зв'язки між вузлами утворюють спрямований граф у часовій послідовності. Це дозволяє йому демонструвати динамічне поведінку в часі. Похідні з нейронних мереж прямого взаємодії, RNN можуть використовувати свій внутрішній стан (пам'ять) для обробки послідовностей змінної довжини введення. Це робить їх застосовними для таких завдань, як несегментоване, пов'язане розпізнавання рукописного введення або розпізнавання мови. Термін «рекурентна нейронна мережа» використовується без розбору для позначення двох широких класів мереж зі схожою загальною структурою, де один - це кінцевий імпульс, а інший - нескінченний імпульс. Обидва класи мереж демонструють тимчасове динамічну поведінку. Рекурентна мережу з кінцевими імпульсами є спрямований ациклічний граф, який можна розгорнути і замінити строго прямий нейронною мережею, в той час як мережа з нескінченним

повторенням імпульсів - це спрямований циклічний граф, який неможливо розгорнути.

Як кінцеві імпульсні, так і нескінченні імпульсні рекурентні мережі можуть мати додаткові збережені стани, і зберігання може безпосередньо контролюватися нейронною мережею. Сховища також можна замінити іншою мережею або графікою, якщо він включає затримки часу або має цикли зворотного зв'язку. Такі відстежувані стани називаються закритим станом або закритою пам'яттю, і вони є частиною довгих мереж короткочасної пам'яті (LSTM) та керованих повторюваних блоків. Її також називають нейронною мережею зворотного зв'язку (FNN).

Архітектури

У рівняннях нижче малі регістри представляють вектори. Матриці

Wq і Uq містять, відповідно, ваги вхідного та періодичного з'єднань, де індексом q може бути або вхідний шлюз i, вихідний шлюз o, ворота забуття c, залежно від обчислюваної активації. Таким чином, у цьому розділі ми використовуємо "векторні позначення". Так, наприклад, $c_t \epsilon R^h$ це не просто одна комірка одного блоку LSTM, але містить h клітинки блоку LSTM.

LSTM з воротами забуття

Компактними формами рівнянь для прямого проходу блоку LSTM із затвором забуття є:

$f_{t=Q_g}(W_f x_t + U_f h_{t-1} + b_f)$

$i_{t=Q_g}(W_i x_t + U_i h_{t-1} + b_i)$

$o_{t=Q_g}(W_0 x_t + U_0 h_{t-1} + b_0)$

$c_t = Q_c(W_c x_t + U_c h_{t-1} + b_c)$

$c_t = f_t \text{ o } c_{t-1} + i_t \; o \; c_t$

$h_t = o_t \; o \; Q_h(c_t)$

де початкові значення$c_0 = 0 \; i \; h_0$ а оператор o позначає продукт Адамара (елементний продукт). Індекс t індексує крок часу.

Змінні

$x^t \epsilon\ R^d$: вектор введення в блок LSTM

$f_t \epsilon\ R^h$: вектор активації забути ворота

$i_t \epsilon\ R^h$: вектор активації входу / оновлення ворота

$o_t \epsilon\ R^h$: вектор активації вихідних воріт

$h_t \epsilon\ R^h$: прихований вектор стану, також відомий як вихідний вектор блоку LSTM

$c_t \epsilon\ R^h$: вектор активації клітинного входу

Функції активації

Q_g: сигмовидна функція.

Q_c: гіперболічна дотична функція.

Q_h: гіперболічна дотична функція або, як пропонується глазок LSTM, $Q_h(x) = x$

Вічко LSTM

Малюнок знизу є графічним зображенням блоку LSTM з підключеннями до очей (тобто оглядовим отвором LSTM). Підключення до отворів дозволяють воротам отримувати доступ до каруселі з постійною помилкою (CEC), активацією якої є стан комірки.

h_{t-1}не використовується,c_{t-1}використовується в більшості місць.

$f_{t=Q_g}(W_f x_t + U_f c_{t-1} + b_f)$
$i_{t=Q_g}(W_i x_t + U_i c_{t-1} + b_i)$
$o_{t=Q_g}(W_o x_t + U_o c_{t-1} + b_o)$
$c_{t=f_t} o\ c_{t-1} + i_t o Q_c(W_c x_t + b_c)$
$h_t = o_t o Q_h(c_t)$

Вічко згортковий LSTM

Кінцевий згортковий LSTM. * Позначає оператор згортки.

$f_{t=Q_g}(W_f * x_t + U_f * h_{t-1} + V_f o c_{t-1} + b_f)$
$i_{t=Q_g}(W_i * x_t + U_i * h_{t-1} + V_i o c_{t-1} + b_i)$

$c_{t=} f_t o c_{t-1} + i_i o Q_c(W_c * x_t + U_c * h_{t-1} + b_t)$
$o_{t=} Q_g(W_o * x_t + U_o * h_{t-1} + V_0 o c_t + b_o)$
$h_t = o_t o Q_h(c_t)$

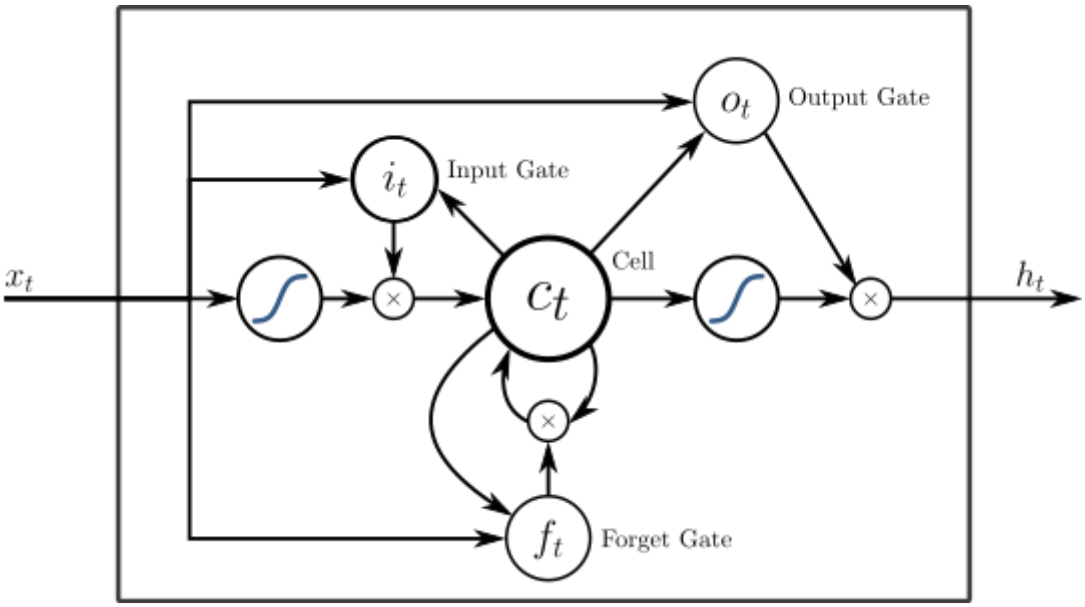

Рисунок 8.

Довга короткочасна пам'ять (LSTM)-це - це штучна архітектура рекурентних нейронних мереж (RNN), що використовується в галузі глибокого навчання. На відміну від стандартних нейронних мереж прямого зв'язку, LSTM має з'єднання зворотного зв'язку. Він може не тільки обробляти окремі точки даних (наприклад, зображення), але й цілі послідовності даних (наприклад, мовлення або відео). Наприклад, LSTM застосовується до таких завдань, як несегментоване, підключене розпізнавання рукописного вводу, розпізнавання мови та виявлення аномалій мережевого трафіку, або IDS (системи виявлення вторгнень).

Загальний блок LSTM складається з комірки, вхідного отвору, вихідного отвору та заслінки забуття. Клітина запам’ятовує значення через довільні проміжки часу, і три ворота регулюють потік інформації в клітину та з неї.

Мережі LSTM добре підходять для класифікації, обробки та прогнозування на основі даних часових рядів, оскільки між важливими

подіями в часових рядах можуть бути затримки невідомої тривалості. LSTM були розроблені для вирішення проблеми зникаючого градієнта, з якою можна зіткнутися при навчанні традиційних RNN. Відносна нечутливість до довжини зазору є перевагою LSTM над RNN, прихованими моделями Маркова та іншими методами навчання послідовності у багатьох додатках.

Пов’язані поля та моделі

RNN можуть поводитися нестабільно. У таких випадках для аналізу може бути використана теорія динамічних систем. Вони насправді є рекурсивними нейронними мережами зі специфічною структурою: структурою лінійної схеми. Тоді як рекурсивні нейронні мережі працюють на будь-якій ієрархічній структурі, поєднуючи дочірні погляди з батьківськими, рекурсивні нейронні мережі працюють на лінійній прогресії часу, поєднуючи попередній крок часу та прихований вигляд у поданні для поточного кроку часу. Зокрема, RNN можуть виступати як нелінійні версії фільтрів з кінцевою імпульсною характеристикою та нескінченною імпульсною характеристикою, а також як нелінійна авторегресійна екзогенна модель (NARX).

Бібліотеки

Apache Singa

Caffe: Створено Центром бачення та навчання Берклі (BVLC). Він підтримує як процесор, так і графічний процесор. Розроблений на C ++ і має обгортки Python та MATLAB.

Chainer: Перша стабільна бібліотека глибокого навчання, яка підтримує динамічні, певні на виконання нейронні мережі. Повністю на Python, виробнича підтримка процесора, графічного процесора, розподіленої навчання.

Deeplearning4j: Глибоке навчання на Java і Scala на Spark з підтримкою декількох графічних процесорів. Універсальна бібліотека глибокого навчання

для виробничого стека JVM, що працює на механізмі наукових обчислень C ++. Дозволяє створювати власні шари. Інтегрується з Hadoop і Kafka.

Flux: включає інтерфейси для RNN, включаючи GRU і LSTM користувача Julia.

Keras: Високорівнева, простий у використанні API, що забезпечує оболонку для багатьох інших бібліотек глибокого навчання.

Microsoft Cognitive Toolkit

MXNet : сучасна система глибокого навчання з відкритим кодом, використовується для навчання і розгортання глибоких нейронних мереж.

PyTorch: Тензори та динамічні нейронні мережі в Python із сильним прискоренням графічного процесора.

TensorFlow: Теаноподібна бібліотека з ліцензією Apache 2.0 з підтримкою процесора, графічного процесора та запатентованого Google TPU для мобільних пристроїв.

Theano: Довідкова бібліотека глибокого навчання для Python з API значною мірою сумісна з популярною бібліотекою NumPy. Дозволяє користувачеві писати символічні математичні вирази, а потім автоматично генерує їх похідні, усуваючи необхідність користувачеві кодувати градієнти або зворотно поширюватися. Ці символічні вирази автоматично компілюються в код CUDA для швидкої реалізації на графічному процесорі.

Torch : Наукова обчислювальна база з широкою підтримкою алгоритмів машинного навчання, написана мовою C та lua. Головним автором є Ронан Колоберт, і в даний час його використовують Facebook AI Research та Twitter.

CUDA

CUDA (англ. Compute Unified Device Architecture) - програмно-апаратна архітектура, яка дозволяє виконувати обчислення з використанням графічних процесорів NVIDIA, що підтримують GPGPU (універсальні обчислення на графічних процесорах) - обчислення загального призначення

на графічних процесорах. Це архітектура паралельних обчислень від NVIDIA, яка дозволяє значно підвищити продуктивність обчислень за рахунок використання графічних процесорів (GPU). Напрямок обчислень еволюціонує від «централізованої обробки даних» на CPU до «спільного обробітку» на CPU і GPU. Для реалізації нової обчислювальної парадигми NVIDIA винайшла архітектуру паралельних обчислень CUDA, яка в даний час представлена в графічних процесорах GeForce, ION, Quadro, Tesla і забезпечує необхідну основу для розробників програмного забезпечення.

Архітектура паралельних обчислень CUDA

CUDA ™ - це архітектура паралельних обчислень, розроблена NVIDIA. Для розробки додатків з використанням архітектури CUDA можна використовувати різні мови і API, включаючи C, C ++, Fortran, OpenCL і DirectCompute. Архітектура CUDA пропонує сотні ядер, здатних обробляти паралельні потоки, в той час як модель програмування CUDA дозволяє розробникам зосередитись на реалізації паралелізму в своїх алгоритмах, а не на структурі мови. Архітектура CUDA останнього покоління під кодовою назвою «Fermi» є найбільш передовою архітектурою обчислень на GPU в історії. Архітектура Fermi, налічує понад три мільярди транзисторів, робить спільну обробку GPU і CPU комплексної і забезпечує неймовірну продуктивність для ряду обчислювальних додатків. Підтримка C ++ спрощує паралельну обробку за допомогою графічних процесорів на базі Fermi і забезпечує більш високу продуктивність в ще більш широкому діапазоні додатків. Ось кілька прикладів додатків, які забезпечують значне підвищення продуктивності: трасування променів, аналіз методом кінцевих елементів, високоточні наукові обчислення, розріджені матриці в лінійній алгебрі, алгоритми сортування та пошуку.

Технологія NVIDIA® CUDA ™

Єдине середовище розробки C, яка дозволяє програмістам і розробникам писати програмне забезпечення для вирішення складних обчислювальних завдань за менший час завдяки многоядерной обчислювальної потужності графічних процесорів. Мільйони графічних процесорів з підтримкою CUDA вже встановлені по всьому світу, і тисячі програмістів безкоштовно використовують інструменти CUDA для прискорення обчислювальних додатків, від кодування відео і аудіо до пошуку нафти і газу, моделювання генома людини, створення тривимірних зображень і інших обчислювальних задач. . науково-прикладна сфера. CUDA SDK дозволяє програмістам реалізовувати на спеціальному спрощеному діалекті алгоритми програмування C, доступні на графічних процесорах NVIDIA, і включати спеціальні функції в текст програми на C. CUDA дає розробнику можливість організувати доступ до набору інструкцій. для графічного прискорювача і управляти його пам'яттю, організовувати на ньому складні паралельні обчислення.

У цій роботі досліджує ця три типи лексичних ланцюжків: точні, синонімічні і семантичні. Лексична ланцюжок пов'язує семантично пов'язані слова в документі. Ми досліджуємо їх потенціал, досліджуючи статистику корпусу на рівні документа (914 текстів), щоб оцінити їх загальну здатність розрізняти легкий і складний текст і завдання класифікації (11000 пропозицій), щоб визначити корисність функцій рівня пропозицій для спрощення. Для вивчення статистики корпусу ми протестували п'ять характеристик рівня документа для кожного типу ланцюжка: загальна кількість ланцюжків, середня довжина ланцюжка, середній інтервал ланцюжка, кількість пересічних ланцюжків і кількість ланцюжків, що перевищують половину довжини документа. Ми виявили істотні відмінності між простим і складним текстом по середній довжині ланцюжка і середній кількості поперечних ланцюжків. Щоб вивчити класифікацію пропозицій, ми

порівняли особливості лексичної ланцюжка зі стандартними характеристиками словникового мішка по ряду класифікаторів: логістична регресія, наївний байесовский алгоритм, дерева рішень, лінійне ядро SVM, RBF і випадковий ліс. Лексичні характеристики ланцюжка були виконані набагато краще, ніж базовий рівень «словникового мішка» у всіх класифікаторах, а кращий класифікатор досяг точності ~ 90% (в порівнянні з 78% для словникового мішка). В цілому, ми виявляємо, що деякі функції лексичної ланцюжка надають конкретну інформацію, корисну для ідентифікації складних речень тексту, на додаток до того, що є серед стандартних лексичних функцій.

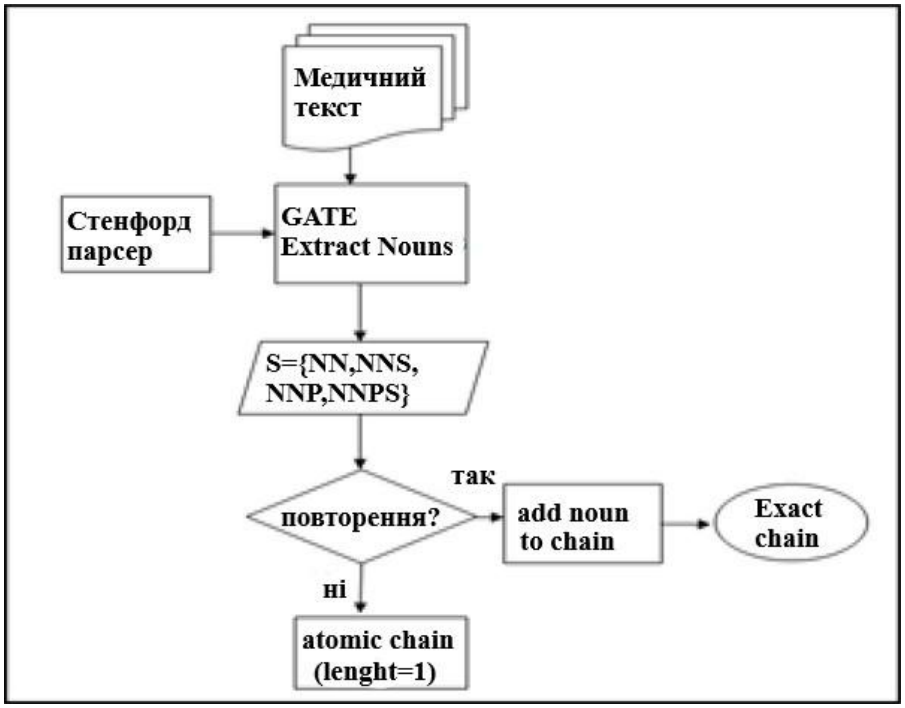

Рисунок 9.Обчислення точних ланцюгів.

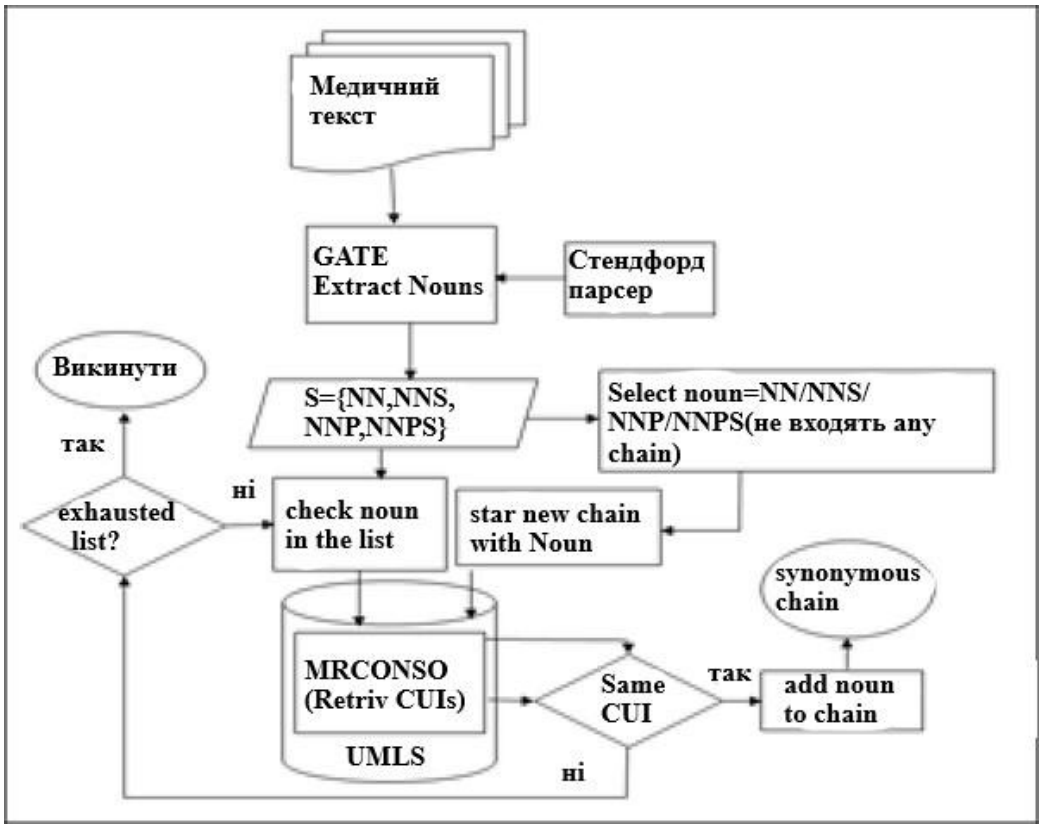

Рисунок 10. Обчислення синонімічних ланцюгів за допомогою бази даних.

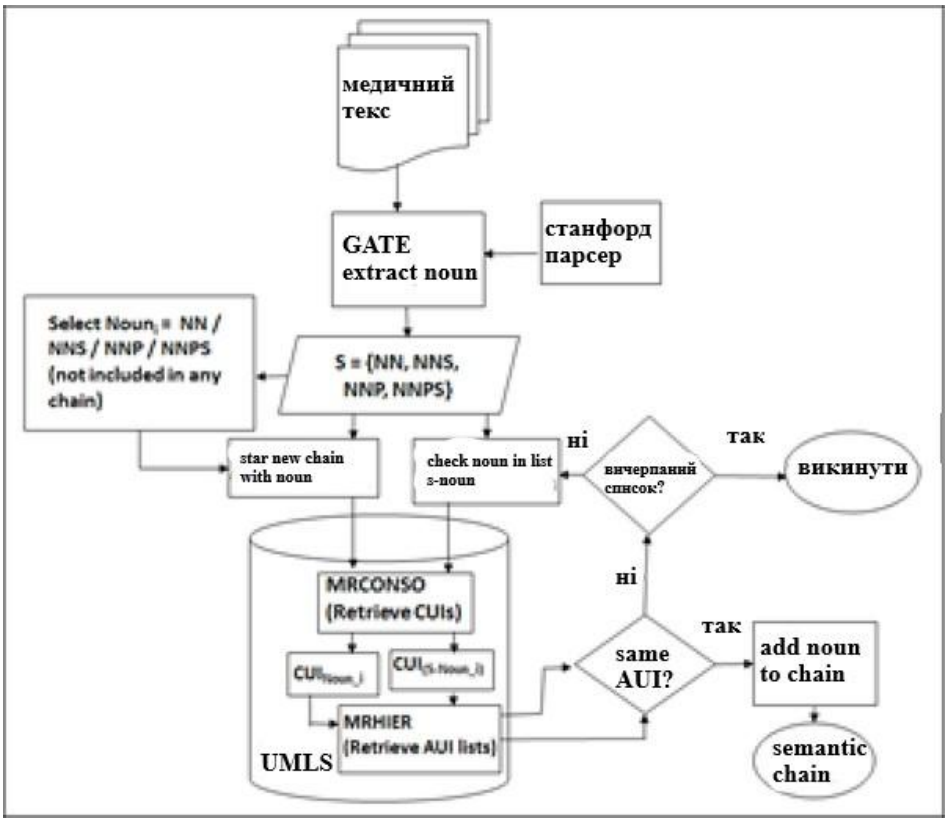

Рисунок 11.Обчислення семантичних ланцюгів за допомогою бази даних.

ВИСНОВКИ

В процесі написання цієї класифікаційної роботи було ретельно розібрано поняття інформації,кластеризації а також класифікації.

На прикладі використання рекурентних методів глибокого навчання (Long short term memory, Recurrent neural network та варіації) для класифікації ланцюжків обмеженої довжини у текстах із підтримкою обчислень CUDA.

Доцільно і далі продовжувати дослідження цієї теми адже робота з інформаціє є невід’ємною частиною повсякденності сучасної людини.

Крім того текстова інформація є найбільшим джерелом знань для сучасного суспільства. А отже є необхідність спрощувати і тим самим покращувати роботу з нею.

ПЕРЕЛІК ПОСИЛАНЬ

1. Інформація.Велика Радянська Енциклопедія.
2. SO / IEC 10746-2: 1996, Information technology - Open Distributed Processing - Reference Model: Foundations.3.2.5:
3. ISO / IEC 2382 до: 2015 Information technology - Vocabulary:
4. ГОСТ 7.0-99 Інформаційно-бібліотечна діяльність, бібліографія. терміни та визначення
5. Інформація // Великий Енциклопедичний словник. - 2000. // Великий енциклопедичний словник / 2-е вид., Перераб. і доп.
6. International Standard IEC 80000-13 Quantities and Units - Part 13: Information science and technology, International Electrotechnical Commission (2008).
7. Захаров В. П. Інформаційні системи (документальний пошук): Навчальний посібник / В. П. Захаров. - СПб .: СПб.гос університет, 2002. - 188 с.
8. Інформації теорія / Ю. В. Прохоров // Випромінювання плазми
9. C. E. Shannon "A Mathematical Theory of Communication" (Переклад в збірнику Шеннон К. "Роботи з теорії інформації і кібернетики". - М .: ІЛ, 1963. - 830 с., С. 243-322)
10. Теорія інформації // «Енциклопедія Кругосвет».
11. Вінер Н. Кібернетика, або управління і зв'язок в тварині і машині; або Кібернетика і суспільство / 2-е видання. - М .: Наука, Головна редакція видань для зарубіжних країн, 1983. - 344 с.
12. Інформація в математиці / Ю. В. Прохоров
13. scikit-learn - Clustering
14. Cornwell, B. (2015). Linkage Criteria for Agglomerative Hierarchical Clustering. Social Sequence Analysis, 270-274
15. Shalamov Viacheslav, Valeria Efimova, Sergey Muravyov, and Andrey Filchenkov. "Reinforcement-based Method for Simultaneous Clustering Algorithm Selection and its Hyperparameters Optimization". Procedia Computer Science 136 (2018): 144-153.
16. Асмус В. Ф., 1947, с. 62.
17. Гетманова А. Д., 1995, с. 52.
18. Гетманова А. Д., 1995, с. 50.
19. Класифікація // Казахстан. Національна енциклопедія.
20. C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. Citeseer: An automatic citation indexing system. In Proceedings of the Third ACM

Conference on Digital Libraries, DL ’98, page 89–98, New York, NY, USA, 1998. Association for Computing Machinery

21. Dragomir R. Radev, Pradeep Muthukrishnan, and Vahed Qazvinian. The acl anthology network corpus. In Proceedings of the 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries, NLPIR4DL ’09, page 54–61, USA, 2009. Association for Computational Linguistics.
22. Tarek Saier and Michael Färber. unarxive: a large scholarly data set with publications’ full-text, annotated in-text citations, and links to metadata. Scientometrics, pages 1–24, 2020.
23. Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, and et al. Construction of the literature graph in semantic scholar. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers), 2018.
24. Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. S2orc: The semantic scholar open research corpus. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020.
25. Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex Wade, Kuansan Wang, Nancy Xin Ru Wang, Chris Wilhelm, Boya Xie, Douglas Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. Cord-19: The covid-19 open research dataset, 2020
26. Dan Su, Yan Xu, Tiezheng Yu, Farhad Bin Siddique, Elham J. Barezi, and Pascale Fung. Caire-covid: A question answering and multi-document summarization system for covid-19 research, 2020.
27. Qingyun Wang, Manling Li, Xuan Wang, Nikolaus Parulian, Guangxing Han, Jiawei Ma, Jingxuan Tu, Ying Lin, Haoran Zhang, Weili Liu, Aabhas Chauhan, Yingjun Guan, Bangzheng Li, Ruisong Li, Xiangchen Song, Heng Ji, Jiawei Han, Shih-Fu Chang, James Pustejovsky, Jasmine Rah, David Liem, Ahmed Elsayed, Martha Palmer, Clare Voss, Cynthia Schneider, and Boyan Onyshkevych. Covid-19 literature knowledge graph construction and drug repurposing report generation, 2020.
28. Francis Wolinski. Automatic extraction of risk factors from covid-19 literature. 2020

29. Arman Cohan, Waleed Ammar, Madeleine van Zuylen, and Field Cady. Structural scaffolds for citation intent classification in scientific publications. Proceedings of the 2019 Conference of the North, 2019.
30. Sarthak Jain, Madeleine van Zuylen, Hannaneh Hajishirzi, and Iz Beltagy. Scirex: A challenge dataset for document-level information extraction. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020.
31. Archana Goyal, Vishal Gupta, and Manish Kumar. Recent named entity recognition and classification techniques: A systematic review. Computer Science Review, 29:21 – 43, 2018.
32. Rhea Sukthanker, Soujanya Poria, Erik Cambria, and Ramkumar Thirunavukarasu. Anaphora and coreference resolution: A review. Information Fusion, 59:139 – 162, 2020.
33. JetBrains Strikes Python Developers with PyCharm 1.0 IDE
34. PyCharm Community Edition and Professional Edition Explained: Licenses and More
35. JetBrains Toolbox - Release and Versioning Changes
36. PyCharm Editions Comparison
37. Yogesh Rana. Python: Simple though an Important Programming language
38. SkipMontanaro. Why is Python a dynamic language and also a strongly typed language - Python Wiki
39. Mark Lutz. A Python Q & A Session (англ.). Learning Python, 3rd Edition [Book]. O'Reilly Media, Inc. (2007)
40. Kalyani Adawadkar. Python Programming - Applications and Future (англ.) // International Journal of Advance Engineering and Research Development. - 2017. - April (iss. SIEICON-2017). - P. 1-4. - ISSN 2348-447
41. Python 3.0 Release
42. PEP 373 - Python 2.7 Release Schedule
43. PEP 373 - Python 2.7 Release Schedule
44. Sunsetting Python 2

.