МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет кібербезпеки та програмної інженерії
Кафедра інженерії програмного забезпечення

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____
“_____”_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ
МАГІСТРА

Тема: “Методологічні засади розробки мультимедійних тезаурусів”

Виконавець: Луцик Олександр Романович

Керівник: к.т.н. доцент Шибицька Наталія Миколаївна

Нормоконтролер: Кравченко Ольга Сергіївна, асс

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки та програмної інженерії
Кафедра інженерії програмного забезпечення
Освітній ступінь магістр
Спеціальність 121 Інженерія програмного забезпечення
Освітньо-професійна програма «Інженерія програмного забезпечення»
Форма навчання денна

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
"_____"_____2023 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи студента
Луцика Олександра Романовича

1. Тема кваліфікаційної роботи: «Методологічні засади розробки мультимедійних тезаурусів» затверджена наказом ректора від 29.09.2023 р. № 1994/ст.
2. Термін виконання проекту: з 02.10.2022 р. по 31.12.2023 р.
3. Вихідні дані до роботи: сформувати методи автоматизованої побудови семантичної мережі тезауруса.
4. Зміст пояснювальної записки:
   1) Огляд існуючих методів побудови та наповнення тезаурусів
   2) Методи побудови семантичної мережі термінів тезауруса
   3) Комплекс програм, що реалізують описані методи та оціна їх ефективності
5. Перелік обов’язкових слайдів презентації:
   1) Загальна схема методу побудови семантичної мережі слів
   2) Загальна схема методу побудови синсетів
   3) Загальна схема методу побудови зв’язків

4) UML-діаграма методу побудови синсетів

5) UML-діаграма методу побудови зв’язків

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1.	Розробка та затвердження графіка роботи	02.10.2023 р. – 15.10.2023 р.	
2.	Підготовка та написання 1 розділу. Відсилка керівнику	15.10.2023 р. – 31.10.2023 р.	
3.	Підготовка та написання 2 розділу. Відсилка керівнику	31.10.2023 р. – 14.11.2023 р.	
4.	Підготовка та написання 3 розділу. Відсилка керівнику	14.11.2023 р. – 05.12.2023 р.	
5.	Редагування та друк пояснювальної записки, графічного матеріалу Відсилка ПЗ для перевірки на плагіат одним файлом.	05.12.2023 р. – 13.12.2023 р.	
6.	Проходження нормо-контролю, перепліт пояснювальної записки. Отримання відгуку керівника. Підготовка презентації та тексту доповіді.	13.12.2023 р. – 15.12.2023 р.	
7.	Передзахист	20.12.2023 р.	
8.	Отримання рецензії	21.12.2023 р.	
9.	Здати секретарю ДЕК: ПЗ, ГМ, CD-R з електронними версіями ПЗ, ГМ, презентацію, відгук керівника, рецензію, довідку про успішність, 2 папки, 2 конверта)	21.12.2023 р.	
10.	Захист дипломної роботи перед ЕК	26.12.2023 р.	

Дата видачі завдання: 02.10.2023 р.

Керівник дипломної роботи: к.т.н. доцент Наталія ШИБИЦЬКА

Завдання прийняв до виконання: Олександр ЛУЦИК

РЕФЕРАТ

Пояснювальна записка до дипломної роботи на тему «Методологічні засади розробки мультимедійних тезаурусів»: 100 с., 16 рис., 4 табл., 60 літературних джерел.

Ключові слова: ТЕЗАУРУС, СЕМАНТИЧНА МЕРЕЖА.

Об’єкт дослідження – автоматизація формування словникової бази тезауруса.

Предметом дослідження – методи та засоби побудови семантичної мережі тезауруса.

Метою даного дипломного проєкту є проведення комплексного дослідження методологічних засад розробки алгоритмів для автоматичної побудови семантичної мережі мультимедійного тизауруса, що дозволить, в цілому, покращити покращення доступності, релевантності та використання мультимедійної інформації в різних сферах, включаючи освіту, науку, культуру та бізнес.

Методи дослідження – проведення систематичного огляду наукової літератури щодо теми мультимедійних тезаурусів, для ознайомлення з існуючими теоріями, методами та підходами, вивчення методів розробки та використання тезаурусів у різних галузях та дослідження їхніх особливостей вивчення.

В ході виконання проєкту здійснено огляд теоретичних відомостей та літературних джерел по теорії проєктування та формування словникової бази тезаурусів, описано методику наповнення тезауруса новими словами, та встановлення семантичних зв’язків і відповідностей між ними та мультимедійними даними, що їх описують.

Результати дипломного проекту рекомендується використовувати їх як набір методів та рекомендацій при проєктуванні та для автоматичного наповнення мультимедійних словників з широким семантичним спектром.

ABSTRACT

Explanatory note for the thesis on «Methodological principles of multimedia thesauri development»: 100 pages, 16 figures, 4 tables, 60 literary sources.

Keywords: THESAURUS, SEMANTIC NETWORK.

The research object is the automation of building the vocabulary base of a thesaurus.

The research subject is the methods and tools for constructing the semantic network of a thesaurus.

The aim of this thesis project is to conduct a comprehensive study of the methodological principles for developing algorithms to automatically construct the semantic network of a multimedia thesaurus. This will, overall, enhance the accessibility, relevance, and utilization of multimedia information in various fields, including education, science, culture, and business.

Research methods include a systematic review of scientific literature on the topic of multimedia thesauri to become familiar with existing theories, methods, and approaches, studying the methods of developing and utilizing thesauri in different fields, and investigating their specificities.

Throughout the project, a review of theoretical information and literary sources on the theory of designing and forming the vocabulary base of thesauri has been conducted. The methodology for enriching the thesaurus with new words, establishing semantic connections and correspondences between them, and multimedia data describing them has been described.

The results of the thesis project are recommended for use as a set of methods and recommendations in designing and automatically populating multimedia dictionaries with a wide semantic spectrum.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

CLIL	–	Content and Language Integrated Learning
ECO	–	Extended Compositionality
ANTLR	–	Another Tool for Language Recognition
CPM	–	Critical Path Method

ВСТУП

В нашому сучасному світі інформаційної революції, зростаюча маса мультимедійної інформації стала невід'ємною частиною нашого повсякденного життя. З текстової інформації до зображень, аудіо та відео, мультимедійні дані зберігають і передають знання, емоції та ідеї. Однак, у великому сенсі, це завдання організації, індексації та навігації в цій величезній кількості даних вимагає надзвичайної уваги та методологічної підготовки.

Мультимедійні тезауруси представляють собою важливий інструмент для ефективного управління мультимедійною інформацією. Вони дозволяють організовувати та індексувати мультимедійні дані, роблячи їх доступними для користувачів на різних платформах та в галузях застосування. Однак, розробка та управління мультимедійними тезаурусами вимагають чітких методологічних підходів для досягнення високої якості та релевантності організованої інформації.

Дослідження методологічних засад розробки мультимедійних тезаурусів виконується з метою покращення організації та доступу до мультимедійних ресурсів в різних галузях діяльності, таких як освіта, наука, культура та бізнес. Воно спрямоване на розробку методологічних підходів, які сприятимуть створенню більш ефективних інструментів для роботи з мультимедійною інформацією та піднесуть якість обслуговування користувачів до нових висот. Результати цього дослідження сподіваються внести суттєвий внесок в розвиток інформаційних технологій та поліпшити доступність та релевантність мультимедійних даних у світі цифрової інформації.

Дана робота з теми зможе покращити процес створення та управління тезаурусами, що, в свою чергу, може позитивно позначитися на кількох

аспектах, включаючи якість надання української освіти. Тезауруси сприяють організації та доступності освітніх ресурсів, забезпеченню релевантності навчального матеріалу.

Мультимедійні тезауруси можуть бути використані для класифікації навчальних ресурсів, таких як відеоуроки, підручники, електронні курси тощо. Це полегшує навігацію студентів та вчителів в освітньому контенті. Тезауруси дозволяють встановлювати ключові теги для навчальних матеріалів, що полегшує пошук та забезпечує користувачам доступ до специфічної інформації. Можна буде використовувати алгоритми рекомендації для визначення найкращих навчальних ресурсів для кожного студента, враховуючи їхні індивідуальні потреби та інтереси. Вбудовані семантичні відносини у мультимедійних тезаурусах допомагають користувачам знаходити пов’язану інформацію та розуміти її в контексті, що покращує розуміння та навчання.

Покращення якості освіти в Україні дозволить залучати більше іноземців, в основному молодих студентів з-за кордону. Залучення іноземців до України та сприяння їхньому отриманню освіти та інтеграції в українське суспільство може бути одним із рішень для вирішення демографічних проблем країни. Співробітництво з іноземними студентами може мати численні переваги для України.

По-перше, прийом іноземних студентів допомагає збільшити кількість молодого населення в Україні, оскільки багато з них приїжджають для навчання і залишаються після закінчення навчання. Це сприяє збільшенню робочої сили та розвитку різних галузей економіки.

По-друге, іноземні студенти приносять додатковий економічний приріст для українських університетів та міст. Вони сплачують плату за навчання,

витрачають гроші на житло, харчування та інші послуги, що сприяє розвитку місцевої інфраструктури і підтримує бізнес в різних галузях.

По-третє, іноземні студенти приносять культурні та мовні різноманітності, що може збагатити українське суспільство. Вони стають частиною місцевої культурної спільноти і сприяють взаємному розумінню та міжнаціональному діалогу.

Основна перепона в процесі інтеграції іноземців полягає в мовному бар'єрі. Українська мова є державною мовою та мовою освіти в Україні. Іноземцям, які не володіють українською, може бути важко знайти роботу, отримати освіту або взаємодіяти з місцевими органами іншими сферами життя. Мовний бар'єр може стати перешкодою для їхньої адаптації в Україні. Володіння українською мовою може бути важливим для отримання доступу до освіти, медичних послуг та інших громадських послуг. Іноземці, які не володіють мовою, можуть зазнавати складнощів у використанні цих послуг. Тому, забезпечення можливості навчання мові, сприяння мовній інтеграції та підтримка іноземців у їхньому старанні вивчити українську мову може сприяти їхній адаптації та взаєморозумінню в українському суспільстві.

Хорошою загальносвітовою практикою «пом'якшення» інтеграції іноземних студентів в українське суспільство є методика предметно-мовного інтегрованого навчання CLIL. CLIL є інноваційним підходом до навчання, який поєднує в собі вивчення академічних предметів іноземною мовою з метою сприяти якісній інтеграції іноземних студентів в освітню систему та соціокультурне оточення. Даний підхід спрямований на вирішення проблеми мовного бар'єру та забезпечення ефективного навчання. У CLIL студенти вивчають предмети, такі як математика, науки, література, але роблять це іноземною мовою. Це сприяє одночасному розвитку академічних та мовних

навичок. Уроки зазвичай орієнтовані на практичні завдання та проекти, що дозволяє студентам застосовувати свої знання у практичних ситуаціях. Методика також сприяє розумінню інших культур та способів мислення через спілкування та вивчення предметів в іноземній мові.

Переваги CLIL включають покращення якості навчання та підвищення мотивації студентів, особливо іноземних. Він також сприяє більш успішній інтеграції іноземних студентів у академічне середовище, оскільки вони навчаються нарівні зі своїми колегами. Однак важливо забезпечити належну підготовку вчителів іноземної мови та предметів для впровадження CLIL, а також враховувати індивідуальні потреби студентів. Такий підхід може бути ефективним інструментом для покращення якості освіти та сприяння інтеграції іноземних студентів, якщо він використовується з розумінням та уважністю до особливостей кожної групи студентів.

Використання мультимедійного тезауруса української мови може виявитися надзвичайно корисним для успішного впровадження CLIL в українській освітній системі. Перш за все, важливо враховувати, що даний підхід передбачає вивчення академічних предметів іноземною мовою. Тому, для ефективного впровадження цього підходу в Україні, необхідно мати доступ до великої кількості відповідного академічного контенту та ресурсів, які б були доступні студентам та вчителям.

Мультимедійний тезаурус української мови може сприяти збагаченню академічного контенту та полегшити його доступність для студентів та вчителів, надавати інтерактивні та мультимедійні визначення слів, що допоможе студентам зрозуміти значення та контекст вживання термінів в контексті предметів. [61] За допомогою графіки, відеоматеріалів та ілюстрацій мультимедійний тезаурус може допомагати візуалізувати складні поняття і

теми, що допомагає студентам краще розуміти та запам’ятовувати матеріал. Також, мультимедійний тезаурус може включати в себе аудіо- та відеофайли, які допомагають студентам виробляти навички аудіювання та вимови.

Україна має значний досвід у вивченні мови та розвитку освітніх ресурсів. Наприклад, Мовний портал Міністерства освіти і науки України є одним із прикладів цього, де можна знайти багато корисних мовних ресурсів для навчання української мови. Інтеграція таких ресурсів у мультимедійний тезаурус може стати важливим кроком у створенні багатомовного навчального середовища для студентів, які вивчають академічні предмети іноземною мовою в Україні, що сприяє більш успішній реалізації підходу CLIL в українській освіті.

РОЗДІЛ 1

Дослідження методів розробки та наповнення тезаурусів

Дослідженню проблем тезаурусного моделювання різних областей присвячені роботи таких вітчизняних і зарубіжних науковців, як В. Демченко, М. Епштейн, О. Збанацька, В. Луков, С. Роу. А. Томас та ін. Проте слід зауважити, що комплексне, системне дослідження мультимедійного тезаурусу української мови досі відсутнє.

Мультимедійний тезаурус – це інформаційна система, яка об'єднує слова, терміни, поняття, або концепції, пов'язані між собою в мережу, за допомогою мультимедійних засобів (тексти, зображення, відео, аудіо тощо) та встановлює взаємозв'язки між ними. Вона призначена для полегшення пошуку, розуміння та інтерпретації інформації.

Термін «тезаурус» є науково актуальним та достатньо усталеним у педагогіці, він широко використовується в науково-педагогічній літературі, але, незважаючи на це, до теперішнього часу, ще немає однозначного і загальноприйнятого визначення цього поняття. Аналіз різних літературних джерел з проблематики дослідження, дозволив нам виділити три смислові групи визначень поняття «тезаурус»: перша група - «словникова», головні одиниці-ознаки: тезаурус – Міська Н.В. це система (спеціально організована система слів та виразів); тезаурус – це зв'язки (організація тезаурусу передбачає наявність семантичних зв'язків – прямих і зворотних між його смисловими елементами); тезаурус – це одномовний словник; друга група - «інформаційно-пошукова», головні одиниці-ознаки: тезаурус – це система; тезаурус – це

зв’язки між лексичними одиницями дискрепторної (інформаційнопошукової) мови; тезаурус – це система зв’язків між елементами інформаційно-пошукової та природної мови; тезаурус – компонент інформаційно-педагогічного середовища або програмного забезпечення взагалі; тезаурус – це одномовний словник; третя група – «знанієва», головні одиниці-ознаки: тезаурус – складна система понять, знань; тезаурус – зв’язки між поняттями якої-небудь області знань; тезаурус – зв’язки між різними областями знань; тезаурус – компонент уявлення людини; тезаурус – необов’язково одномовний словник. У першій смисловій групі тезаурус розглядається як тип побудови одномовного словника. У другій тезаурус представляє собою основу функціонування інформаційно-педагогічного середовища, інформаційно-пошукової мови. Третя група розглядає тезаурус як систему уявлень, знань людини про оточуючий світ або окремих його областей.

С. Сисоєва, І. Соколова у своїх дослідженнях виходять з того, що тезаурус характеризується системною цілісністю і процесуальною неперервністю в органічній єдності загального, особливого, індивідуального та специфічно-предметного. Як загальне – тезаурус розглядається як єдина картина світу, що відображена в поняттях та зв’язках між ними. Це так званий категоріальний рівень презентації тезауруса. Рівень особливого відображає тезауруси знань, що входять в предметні площини. По суті йдеться про синтаксично-детермінований відкритий інформаційний базис певної галузі науки, семантично структурований відповідно до специфічних для неї відносин, що є усталеними в науці або склалися на початок дослідження проблеми. На індивідуальному (особистісному) рівні тезаурус є, по суті, сукупністю системних знань одного окремого індивіда або групи суб’єктів, відображений певним чином в його або їх свідомості. Створення адекватного і відкритого для поповнення і розвитку

тезауруса, як правило, відбувається на основі моделювання предметної області. Застосовувані в цьому випадку моделі являють собою структурно-логічні схеми функціонально взаємозалежної системи об'єктів тієї чи іншої предметної області. Вважається, що тезаурус буде в цілому визначено, якщо будуть задані його морфологія, ієрархія і взаємозв'язки понятійних структур (ядер).

1.1. Моделювання тезауруса

Аналіз науково-педагогічної літератури з досліджуваної проблеми дозволив виділити три основні етапи моделювання наукового тезаурусу вітчизняної теорії управління освітою. Перший етап - моделювання предметного поля (предметної області) вітчизняної теорії управління освітою, другий етап – перетворення отриманої моделі предметного поля вітчизняної теорії управління освітою на модель тезаурусу вітчизняної теорії управління освітою, третій етап − наповнення моделі тезаурусу вітчизняної теорії управління освітою конкретним змістом у вигляді словника (довідника) термінів і понять.

Побудова (створення) тезаурусного поля будь-якого досліджуваного процесу, в тому числі і педагогічного, - це структурування і візуалізація наявної бази даних, представленої в систематизованих блоках (фасетах) [1]. Створення тезаурусного поля надає можливість дослідникові систематизувати вже наявну інформацію і структурувати спектр подальшого інформаційного, пізнавального або дослідницького пошуку. При складанні тезаурусних полів необхідно використовувати основні закономірності візуальної подачі текстового матеріалу: зчитування інформації відбувається «зліва – направо», «зверху – вниз»; грамотне розташування елементів один до одного за значимістю має співвідноситися як «ближче – далі», «вище – нижче», «минуле – теперішнє –

майбутнє», «центр – периферія», «фокус – перспектива»; кольори, графічні символи, коди і шифри, які відображають ієрархічні взаємозв’язки елементів що входять в тезаурус, здійснюють безпосередній вплив на ефективність сприйняття і запам’ятовування [1].

Предметна область мультимедійних тезаурів представляє собою важливий сегмент сучасної інформаційної науки та інформаційної технології. Мультимедійні тезауруси є інструментами, які допомагають в організації та візуалізації знань, полегшують пошук інформації та забезпечують ефективну комунікацію між користувачами та інформаційними ресурсами.

Семантика та синтаксис мультимедійних тезаурусів: мультимедійні тезауруси спроектовані таким чином, щоб відображати відносини між словами, термінами та поняттями, які можуть бути семантичними, синонімічними, антонімічними, частиною-цілою тощо. Це важливо для правильного розуміння контексту та забезпечення точності пошуку. Також тезаурус має враховувати синтаксичні відношення між словами та термінами, наприклад, граматичні відношення, структурні правила тощо.

Мовна многозначність: одне слово може мати кілька значень в різних контекстах. Мультимедійний тезаурус повинен враховувати цю мовну многозначність і надавати різні визначення для одного терміну з можливістю переходу між ними.

Візуалізація та інтерактивність: мультимедійні тезауруси можуть використовувати візуальні елементи, такі як діаграми, схеми, графіки та ілюстрації, для кращого розуміння понять та показу взаємозв’язків.

Інтерактивність також важлива, оскільки користувачі повинні мати можливість взаємодіяти з тезаурусом, додавати нову інформацію, коментувати та редагувати вміст.

Спільноти користувачів: багато мультимедійних тезаурусів надають можливість створення спільнот користувачів, де фахівці та ентузіасти можуть обговорювати та спільно вдосконалювати тезаурус.

Проблеми забезпечення доступності: для багатьох користувачів доступність мультимедійних тезаурусів є важливою проблемою. Забезпечення доступності для людей з обмеженими можливостями та різних груп користувачів є важливим завданням.

Застосування в різних галузях: мультимедійні тезауруси можуть бути використані у багатьох галузях, включаючи освіту, науку, музейну сферу, інформаційний пошук, мовознавство, медицину, археологію, інженерію та багато інших.

Спроби створення стандартів: у галузі мультимедійних тезаурусів було багато спроб створити стандарти, що визначають загальні правила їхньої розробки та використання, але це завдання залишається складним.

Лінгвістичні та інформаційні аспекти: мультимедійні тезауруси вимагають великої лінгвістичної та інформаційної експертизи для розробки, поповнення та підтримки.

Процес моделювання мультимедійного тезауруса грає важливу роль у створенні ефективного та корисного інструмента для інтеграції та розподілу мультимедійних ресурсів.

Моделювання мультимедійного тезауруса – це складний процес, спрямований на створення та розвиток мультимедійних тезаурсів. Цей процес включає в себе кілька ключових етапів та дій, спрямованих на створення структурованої бази даних та інтерфейсу, який дозволяє користувачам взаємодіяти з різними мультимедійними ресурсами, такими як тексти, зображення, відео, аудіо тощо.

Основні етапи процесу моделювання мультимедійного тезауруса включають наступне:

1) Збір та структурування інформації: Цей етап передбачає збір різноманітної інформації, яка буде включена до мультимедійного тезаурусу. Це можуть бути тексти, зображення, відео, аудіофайли та інші мультимедійні ресурси. Основна мета - систематизувати цю інформацію за певними категоріями чи темами.
2) Розробка бази даних: Для ефективного зберігання та організації інформації створюється база даних. Вона включає таблиці та структури даних, які дозволяють зв’язувати та інтегрувати тексти, мультимедійні файли та інші ресурси. Важливо, щоб база даних була добре спроектованою та оптимізованою для швидкого доступу до інформації.
3) Розробка інтерфейсу: Створення інтерфейсу, який дозволяє користувачам зручно взаємодіяти з мультимедійним тезаурусом. Цей інтерфейс повинен бути інтуїтивно зрозумілим та забезпечувати можливість пошуку, навігації та відображення різних типів мультимедійних ресурсів.
4) Створення зв’язків між термінами: Важливим аспектом мультимедійного тезаурусу є встановлення взаємозв’язків між термінами та поняттями. Це може включати в себе визначення семантичних, синтаксичних та асоціативних зв’язків між словами та термінами.

5) Індексування та пошук: Для ефективного пошуку інформації в мультимедійному тезаурусі створюються індекси та пошукові механізми. Це дозволяє користувачам швидко знаходити необхідну інформацію за ключовими словами, темами або категоріями.
6) Тестування та вдосконалення: Після створення мультимедійного тезаурусу, він піддається тестуванню користувачами, щоб виявити помилки та недоліки. На основі отриманих відгуків систему вдосконалюють та оновлюють.
7) Підтримка та розвиток: Мультимедійний тезаурус вимагає постійної підтримки та оновлення. Нова інформація, зміни в галузі або вимоги користувачів можуть впливати на розширення та розвиток системи.

Тезаурусне моделювання - це методологічний підхід, що використовується для структурування та організації інформації, а також для визначення взаємозв'язків між словами, термінами або поняттями. Воно допомагає створити систему контролю та навігації для користувачів, щоб вони могли легко знаходити інформацію, пов'язану з конкретним терміном або темою. Тезаурусне моделювання використовується в різних сферах, таких як інформаційні системи, бібліотекознавство, мовознавство, інформаційний пошук, освіта та наука.

Асоціативне тезаурусне моделювання:

Сутність: Асоціативне тезаурусне моделювання базується на асоціативних зв'язках між словами або термінами. Це означає, що терміни групуються разом на основі їхнього спільного вживання в текстах або схожого контексту.

Приклад: Слова «сонце», «небо» і «світло» можуть бути асоційовані, оскільки вони часто зустрічаються в одному контексті, адже «сонце світить на небі».

Застосування: Асоціативні тезауруси полегшують користувачам знаходження схожих та пов’язаних термінів, що допомагає в розумінні текстів і підвищує ефективність пошуку.

Семантичне тезаурусне моделювання:

Сутність: Семантичні тезауруси базуються на семантичних зв’язках між словами або термінами. Вони враховують значення та синоніми, антоніми, гіпоніми (поняття, які є частиною більш загального поняття) і гіпероніми (поняття, які є більш загальними за інші) для створення взаємозв’язків.

Приклад: Слово «собака» є гіпонімом до «тварина» і їх можна пов’язати семантично, оскільки «собака» - це конкретний вид тварини.

Застосування: Семантичні тезауруси допомагають визначати відносини між словами та термінами на основі їхнього значення, що полегшує пошук та розуміння текстів.

Функціональне тезаурусне моделювання:

Сутність: Функціональні тезауруси визначають функціональні відносини між словами або термінами. Це може включати в себе відносини, такі як «частина-ціле,» «призначення,» «використання» і багато інших.

Приклад: Слово «колесо» може мати функціональні відносини «частина» до «автомобіль» і «призначення» та «рухати автомобіль».

Застосування: Функціональні тезауруси допомагають визначати функціональну роль і значення термінів або об’єктів в системі.

Алфавітний тезаурус:

Сутність: Алфавітні тезауруси організовані за алфавітним порядком, де кожен термін або термінологічна одиниця розміщується в алфавітному порядку, або буквених групах.

Приклад: Терміни в алфавітному тезаурусі розміщені від «А» до «Я», і користувачі можуть швидко знаходити необхідні терміни, використовуючи алфавітний пошук.

Застосування: Алфавітні тезауруси допомагають користувачам знаходити інформацію за допомогою алфавітного індексу, що спрощує пошук великої кількості термінів.

Терміни в мультимедійному тезаурусі можуть бути визначені автоматично (шляхом аналізу текстових документів та інших джерел інформації), витягнуті з інших баз даних (таких як інші тезауруси або онтології) або створені вручну експертами. У будь-якому випадку важливо дотримуватися певних правил та керуватися наступними принципами. Поняття визначається як унікальна комбінація характеристик, а визначення подається у вигляді описового виразу, що робить його відмінним від інших пов’язаних понять [6].

1) Термін повинен описуватись в однині. Виключення складають ті поняття, що самі є множинними.

Приклад: «Ідентифікаційний код»:

- добре визначення «Число, що ідентифікує людину»;
- погане визначення «Число для ідентифікації людей». Причина – у поганому визначенні використовується слово «люди», яке є неоднозначним, тому що це можна зрозуміти так, ніби один номер може посилатися на кілька людей.

2) Визначати, чим є наведене поняття, а не тільки чим воно не є:

Приклад: «Розмір вартості посилки»:

- добре визначення «Розмір витрат, які несе вантажовідправник для переміщення товарів з одного місця до іншого»;
- погане визначення «Розмір витрат, які не відносяться до витрат на пакування, оформлення, завантаження, розвантаження та страхування». У поганому прикладі не вказано, що входить до поняття елементу даних;

3) Визначення повинно складатися з описової фрази або речення, оскільки просто навести синоніми не дає достатньої точності. У визначенні необхідно включити важливі характеристики поняття, і це досягається за допомогою повних та граматично правильних речень.

Приклад – «Ім’я агента»:

- добре визначення «Назва сторони, яка уповноважена діяти від імені іншої сторони»;
- погане визначення «Представник» Причина – «Представник» є близьким синонімом імені елементу даних, який не може бути адекватним визначенням;

4) Використання лише широко відомих скорочень у визначеннях важливо, оскільки розуміння скорочень, таких як абревіатури та ініціали, зазвичай обмежено конкретним контекстом. У інших ситуаціях ті ж самі скорочення можуть призвести до неправильного розуміння або непорозуміння. Тому, для уникнення неоднозначності, визначення містять лише повні слова, а не скорочення. Винятком може бути використання широко вживаних скорочень, таких як «т.д.», або випадки, коли скорочення є більш зрозумілим і зручним для користувача, ніж повна форма складного терміна. У всіх випадках абревіатури мають бути розшифровані при їхньому першому зазначенні.

Приклад: «Висота над рівнем моря»:

- добре визначення: «Вертикальна відстань від середнього рівня моря до рівня, що описується»;
- погане визначення «Вертикальна відстань від до рівня, що описується». Погане визначення є незрозумілим, тому що абревіатуране є загальнозрозумілою і деякі користувачі змушені будуть звертатися до інших джерел для з'ясування її значення. Без зазначення повного слова пошук терміна у словнику може бути складним або взагалі неможливим.

5) Визначення не повинно включати в себе визначення інших даних або базових понять, які стосуються іншого елементу даних. Визначення термінів повинні бути включені в відповідний словник або глосарій. Якщо необхідно надати додаткове визначення для іншого терміну, це може бути зроблено як примітка, додана після первинного визначення, або як окремий запис у словнику. Зв'язані визначення можуть бути доступні за допомогою атрибутів посилання, тобто перехресних посилань.

Приклад: «Код типу зразка»:

- добре визначення «Код, який ідентифікує тип зразка»;
- погане визначення: «Код, який ідентифікує тип обраного зразка. Зразок - це мала частка, вилучена для проведення експериментів. Він може бути як єдиним зразком для тестування, так і сурогатним зразком для контролю якості. Зразок для контролю якості - це сурогатний зразок, обраний для перевірки результатів тестування єдиних зразків.» Краще визначення: «Код, що відзначає тип обраного зразка. Зразок - це мала частина, вилучена для проведення експериментів, і може бути як унікальним екземпляром для тестування, так і сурогатним зразком для контролю

якості. Сурогатний зразок для контролю якості використовується для перевірки результатів тестування унікальних зразків.» У покращеному визначенні видалено зайві визначення «зразка» та «зразка для контролю якості», що робить його більш коротким та зрозумілим.

6) У визначенні поняття необхідно відображати всі ключові характеристики цього поняття з врахуванням рівня деталізації, який вимагається в конкретному контексті. Важливо уникати зазначення неважливих деталей. Рівень докладності повинен відповідати потребам користувачів та конкретному середовищу, де використовується це визначення.

- Приклад: «Номер послідовності завантаження вантажу» (визначений контекст: будь-яка форма транспортування):
- добре визначення: Номер, який вказує послідовність, в якій здійснюється завантаження до транспортного засобу або елемента транспортного середовища;
- погане визначення: Номер, який відображає послідовність, в якій здійснюється завантаження до вантажівки. У даному контексті вантажі можуть транспортуватись різними транспортними засобами, приміром, вантажівками, судами, вантажними потягами. Транспортування не обмежене лише вантажівками.

7) Визначення повинно бути якомога більш точним та однозначним: іншими словами, воно повинно чітко передавати значення та інтерпретацію поняття, яке воно визначає. Визначення повинно бути настільки зрозумілим, щоб гарантувати її однозначне розуміння без будь-яких двозначностей.

Приклад: «Дата Отримання Вантажу»:

- добре визначення «Дата, на яку вантаж передається отримувачу»;

– погане визначення «Дата, на яку здійснюється доставка вантажу». У поганому визначенні не роз’яснюється, що таке «доставка». Під «доставкою» можна зрозуміти як момент розвантаження товару у певному місті, так і факт передачі товару кінцевому отримувачу. Не виключено, що кінцевий отримувач ніколи не отримає вантаж, або його передача може здійснитися через кілька днів після розвантаження;

8) Бути коротким: слід запобігати використання додаткових фраз описового характеру.

Приклад: «Ім’я набору символів»:

– добре визначення «Ім’я, що присвоюється набору фонетичних або ідеографічних символів, у які зашифровані дані»;
– погане визначення «Ім’я, що присвоюється набору фонетичних або ідеографічних символів, у яких зашифровані дані для забезпечення використання цього реєстру метаданих або, якщо говорити про загальний вжиток, спроможність системного обладнання і програмного забезпечення обробляти дані, зашифровані одним або декількома шифрами». У поганому визначенні усі фрази після виразу «у яких зашифровані дані» є зайвими описовими фразами;

9) Мати можливість використовуватися окремо: зміст поняття має бути наочним у визначенні. Для розуміння поняття не повинні бути потрібні додаткові роз’яснення.

Приклад: «Назва Міста Розміщення Школи»:

– добре визначення: Назва міста, де знаходиться школа;

– погане визначення: Див. сайт школи. Причина – погане визначення не є самостійним, необхідно звернутися до додаткового джерела (сайта школи) для розуміння поняття;

10) Бути поданим без використання пояснювальної інформації, функціонального використання або процедурної інформації: у разі потреби такі пояснення можуть бути розміщені у інших атрибутах метаданих;

11) Запобігати циклічних посилань: два визначення не повинні бути визначені одне через одне.

Приклад: два елементи даних з поганими визначеннями:

– «Ідентифікаційний Номер працівника – Номер, що призначається працівнику»;
– «Працівник – Людина, яка має відповідний ідентифікаційний номер працівника». Визначення посилаються одне на одне. При цьому в жодному визначенні не наведено зміст поняття.

12) Використовувати ту ж саму термінологію та логічну структуру для пов’язаних визначень: для близьких або пов’язаних визначень має використовуватись одна й та ж сама термінологія та синтаксис.

Приклад: Даний приклад ілюструє цю ідею. Обидва визначення відносяться до пов’язаних понять і тому мають однакову логічну структуру і близьку термінологію.

– «Дата відправлення товарів» – Дата, у яку товари були відправлені даній стороні;
– «Дата отримання товарів» – Дата, у яку товари були отримані даною стороною. Використання єдиної термінології та синтаксису значно

спрощує розуміння. Інакше користувачі можуть не зрозуміти, чому для пов’язаних визначень використовуються різні терміни.

1.2. Методи наповнення тезауруса

1) Ручне індексування:

Опис: Люди вручну присвоюють ключові слова, теги, категорії та іншу метаінформацію мультимедійним об’єктам.

Приклад алгоритму: Немає конкретного алгоритму, це ручний процес. Проте інструменти, такі як Adobe Lightroom для фотографій або інструменти каталогізації медіафайлів, можуть допомагати у встановленні метаінформації.

2) Автоматична індексація:

Опис: Використовуються алгоритми обробки природної мови (NLP) і комп’ютерного зору для автоматичного виділення ключових слів та тегів для мультимедійних об’єктів.

Приклад алгоритму: OpenCV для обробки зображень, бібліотеки для обробки мови, такі як NLTK або spaCy для текстового контенту.

3) Колаборативна індексація:

Опис: Залучення користувачів до внесення метаінформації, такої як теги, рейтинги та відгуки.

Приклад алгоритму: Реалізація системи коментарів і рейтингів на веб-сайті або платформі. Наприклад, Disqus для коментарів.

4) Машинне навчання:

Опис: Використовуються алгоритми машинного навчання для автоматичного аналізу мультимедійних об’єктів і встановлення зв’язків між ними.

Приклад алгоритму: Convolutional Neural Networks (CNN) для розпізнавання об’єктів на зображеннях, Word2Vec або Doc2Vec для автоматичного виділення ключових слів у тексті, рекомендаційні системи, такі як колаборативна фільтрація.

5) Аналіз контенту:

Опис: Аналіз самого мультимедійного контенту для виділення інформації.

Приклад алгоритму: Розпізнавання облич, таке як OpenFace або Dlib для фотографій. Google Cloud Vision або Amazon Rekognition для обробки зображень і відео. Розпізнавання мови, таке як Google Cloud Speech-to-Text або IBM Watson Speech to Text.

6) Обробка відгуків користувачів:

Опис: Аналіз відгуків, рейтингів та коментарів користувачів для поліпшення релевантності інформації.

Приклад алгоритму: Аналіз сентименту, такий як бібліотека *TextBlob* для текстових відгуків, або аналіз тональності у тексті з використанням нейронних мереж.

Кожен з цих методів може використовувати багато різних алгоритмів та бібліотек, залежно від конкретних вимог та області застосування. Реалізація кожного методу може бути унікальною і варіювати в залежності від проекту.

1.3. Класифікація мультимедійних даних для тезауруса

У мультимедійних тезаурусах терміни представлені не лише їх текстовим описом, а й мультимедійними даними, що візуально зображають зміст терміну. Для

досягнення цього мультимедійний контент попередньо розділяється на різні категорії і отримує ярлики, що є відповідними термінами в тезаурусі. Найефективніше розподілення об’єктів на групи досягається алгоритмами комп’ютерного зору.

Розпізнавання об’єктів – це завдання комп’ютерного зору, яке включає виявлення та локалізацію об’єктів на зображеннях чи відео. Це важлива частина багатьох застосувань, таких як нагляд, автомобілі самоїдущі, або робототехніка. Алгоритми виявлення об’єктів можна розділити на дві основні категорії: одноетапні виявники та двоетапні виявники.

Один з найраніших успішних спроб вирішити проблему виявлення об’єктів за допомогою глибокого навчання був модель R-CNN (Regions with CNN features), розроблений Россом Гіршиком і його командою в Microsoft Research у 2014 році. Ця модель використовувала поєднання алгоритмів пропозицій регіонів та згорткових нейронних мереж (CNN) для виявлення та локалізації об’єктів на зображеннях. Алгоритми виявлення об’єктів широко класифікуються на дві категорії відповідно до того, скільки разів одне й те ж вхідне зображення подається через мережу (рис. 1.1).

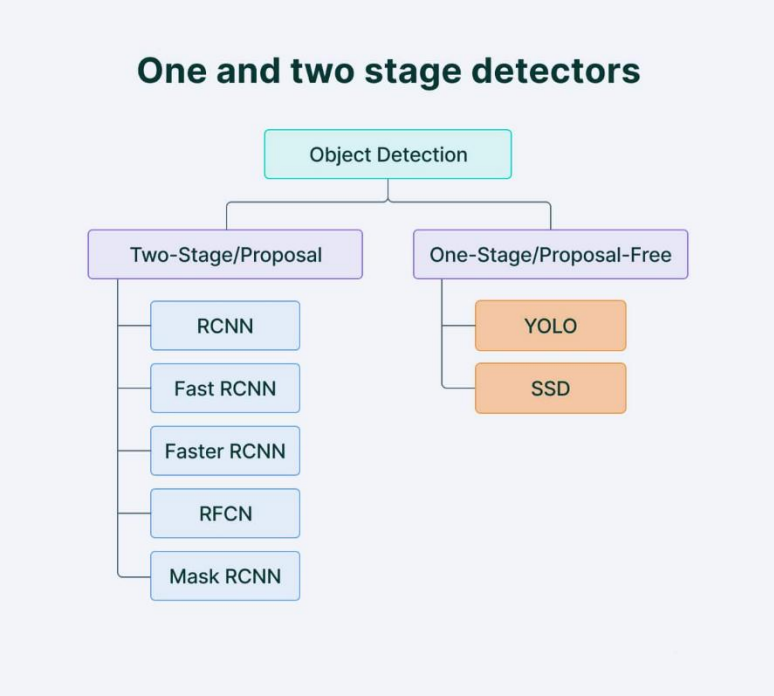

Рис. 1.1. Алгоритми для класифікації мультимедійних даних

Одноетапне виявлення об’єктів. Одноетапне виявлення об’єктів використовує один прохід вхідного зображення для прогнозування наявності та місцезнаходження об’єктів на зображенні. Воно обробляє всю картинку за один прохід, що робить його обчислювально ефективним. Однак одноетапне виявлення об’єктів, як правило, менш точне, ніж інші методи, і менш ефективне у виявленні малих об’єктів. Такі алгоритми можуть використовуватися для виявлення об’єктів в реальному часі в обмежених за ресурсами середовищах.

Двоетапне виявлення об’єктів. Двоетапне виявлення об’єктів використовує два проходи вхідного зображення для прогнозування наявності та місцезнаходження об’єктів. Перший прохід використовується для генерації набору пропозицій або потенційних місцезнаходжень об’єктів, а другий прохід використовується для уточнення цих пропозицій та зроблення остаточних прогнозів. Цей підхід є більш точним, ніж одноетапне виявлення об’єктів, але також є більш обчислювально витратним.

В цілому вибір між одноетапним і двоетапним виявленням об’єктів залежить від конкретних вимог та обмежень застосування. Загально кажучи, одноетапне виявлення об’єктів краще підходить для реального часу, тоді як двоетапне виявлення об’єктів краще підходить для застосувань, де важлива точність.

Метрики оцінки продуктивності моделей виявлення об’єктів:

Для визначення та порівняння прогностичної продуктивності різних моделей виявлення об’єктів нам потрібні стандартні кількісні метрики. Дві найпоширеніші метрики оцінки - це перетин над об’єднанням (англ. Intersection over Union) та середня точність (англ. Average precision).

Перетин над об’єднанням (IoU). Перетин над об’єднанням - це популярна метрика для вимірювання точності локалізації та розрахунку помилок локалізації в моделях виявлення об’єктів.

Для розрахунку IoU між передбаченими та фактичними межами об’єктів спочатку береться область перетину між відповідними межами об’єктів для одного й того ж об’єкта. Після цього розраховується загальна площа, покрита двома межами об’єктів - відома як «об’єднання», та площа перекриття між ними – відома як «перетин».

Частка перетину до об’єднання дає нам відношення перекриття до загальної площі, що надає гарну оцінку того, наскільки близько прогнозована рамка об’єкта знаходиться до оригінальної рамки об’єкта.

Середня точність (AP). Середня точність обчислюється як площа під кривою точності в залежності від повноти для набору прогнозів. Повнота розраховується як співвідношення загальних прогнозів, зроблених моделлю для класу, до загальної кількості існуючих міток для цього класу. Точність відноситься до співвідношення правильно визначених позитивів відносно загальних прогнозів, зроблених моделлю.

Повнота і точність пропонують компроміс, який графічно відображається у вигляді кривої при зміні порога класифікації. Площа під цією кривою точності відносно повноти дає нам Середню Точність для кожного класу моделі. Середнє цього значення, взяте по всіх класах, називається середньою середньою точністю (mAP).

Модель YOLO (англ. You Only Look Once) пропонує використання зв’язної нейронної мережі, яка в одну мить робить прогнози для рамок обмежень та ймовірностей класів. Це відрізняється від підходу, взятого попередніми алгоритмами виявлення об’єктів, які використовували класифікатори для виконання виявлення. Відповідно до фундаментально різного підходу до виявлення об’єктів, YOLO досяг

стану мистецтва, випередивши інші алгоритми виявлення об’єктів у реальному часі значним відступом. У той час як алгоритми, подібні до Faster RCNN, працюють, виявляючи можливі області інтересу за допомогою мережі пропозицій областей, а потім виконують визначення на цих областях окремо, YOLO виконує всі свої прогнози за допомогою одного повністю зв’язаного шару.

Методи, які використовують мережі пропозицій областей, виконують кілька ітерацій для одного й того ж зображення, тоді як YOLO обходиться однією ітерацією.

1.4. Семантичні мережі

У наукових дослідженнях та літературі з інженерії знань та штучного інтелекту слова «семантична мережа» та «онтологія» часто згадуються в схожих контекстах. Незважаючи на це, ці терміни позначають два різних підходи. Онтологія стосується опису предмету, у той час як семантична мережа відображає спосіб структурування знань у вигляді графа.

У попередніх роботах інженерії знань та обробки природної мови семантична мережа визначалась як орієнтований граф з визначеними вершинами (концепціями, подіями, характеристиками або значеннями), та зв’язками між ними [53]. Однією з перших робіт, що нагадує семантичну мережу, є дослідження про семантичну пам’ять від А. М. Коллінса та М. Р. Квилліана [14]. Люди сприймають оточуючий світ як ієрархію концепцій, пов’язаних загальними та частковими відносинами. Наприклад, якщо канарейка - це птах, то можна припустити, що у неї є крила. Інші визначення авторів також підтримують цю ідею [15, 54]. Існують шість різних типів семантичних мереж [55], але найбільш близькі до цієї роботи - це мережі визначень (англ. definitional networks), які висвітлюють класи та підкласи понять за допомогою відношення «is-a».

Семантичні мережі не накладають обмежень на структуру знань чи конкретну предметну область, якщо ці знання можна відобразити у вигляді орієнтованого графа [15, 54]. Вони є лише одним із способів представлення знань. Існують інші форми такого представлення, такі як продукційні правила, фрейми та формальні логічні моделі. Однак дослідження цих форм виходить за рамки даної роботи. Основна увага тут приділяється семантичним мережам визначень як способу представлення знань та тезаурусам як засобу зберігання знань.

У семантичних мережах відображені різноманітні семантичні відносини між концепціями. Ці відносини можуть бути симетричними та асиметричними. Синонімія та антонімія є прикладами симетричних семантичних відносин. Незважаючи на ключову роль синонімії в лінгвістиці, існують різні підходи до її визначення [60]. Оскільки це відносиння є рефлексивним та симетричним, але не обов'язково відповідає властивості транзитивності [31], в цій роботі синонімія розглядається як відношення толерантності у словнику чи сукупності лексичних значень слів. Відносиння частини-цілого, причини-наслідку, протилежності тощо є прикладами асиметричних семантичних відносин [24].

На сьогоднішній день найвідомішою семантичною мережею у сфері обробки природної мови є WordNet, яка базується на формалізації сприйняття людиною навколишнього середовища. На малюнку 1.1 наведено приклад семантичної мережі з верхнього рівня тезаурусу WordNet. У семантичній мережі WordNet поняття формуються на основі відношення синонімії, тому їх називають синсетами (від англійської «set of synonyms» – «множина синонімів», скорочено synset) [19].

1.5. Критерії якості семантичних мереж

Семантичні мережі створюються групами фахівців з лексикографії або автоматично за допомогою методів машинного навчання. Це значно ускладнює процес оцінки якості цих ресурсів, оскільки потрібно, щоб семантичні ресурси адекватно відображали навколишній світ чи конкретну предметну область. На сьогодні оцінка якості семантичних мереж є актуальною науковою проблемою. Існують три загальноприйняті підходи до оцінки якості семантичних мереж:

- Експертна оцінка;
- Порівняння з «золотим стандартом»;
- Проведення порівняльних тестів.

Експертна оцінка передбачає формування оцінки якості лексико-семантичного ресурсу запрошеним експертом або групою фахівців за попередньо розробленою методологією. В цьому випадку кожен елемент семантичної мережі оцінюється за певною шкалою вимірювань, що дозволяє провести якісну або кількісну оцінку. Важливо зазначити, що для експертної оцінки може використовуватися колективне судження великої кількості людей, отримане за допомогою краудсорсингу [36]. Використання краудсорсингу може виникати проблеми з надійністю результатів [45], що пов’язано з неоднозначністю суджень різноманітних учасників, чиї оцінки використовуються для оцінки якості даних. Незважаючи на це, краудсорсинг залишається популярним методом створення та оцінки якості мовних ресурсів [10].

Порівняння з «золотим стандартом» - найбільш поширений підхід до оцінки якості в галузі обробки природної мови та інформаційного пошуку. Цей підхід передбачає порівняння вмісту побудованої семантичної мережі з матеріалами певного заздалегідь підготовленого «золотого стандарту» - набору даних, який має відому високу якість. Перевагою цього підходу є можливість оцінки віддаленості нового ресурсу від вже існуючого на основі певного заданого кількісного показника якості.

Основна складність полягає в труднощах однозначного порівняння концепцій та зв’язків між тестовою онтологією та «золотим стандартом».

Висновки

У першому розділі проведено огляд сучасного стану досліджень та розробок у сфері побудови тезаурусів, створення їх семантичних мереж, показано можливості широкого застосування мультимедійних тезаурусів в освіті. А також розглянуто алгоритми для класифікації зображень по групах термінів в мультимедійних тезаурусах.

На сьогодні існує велика кількість актуальних завдань машинного розуміння тексту на природній мові, що використовують системи, побудовані на основі знань. Серед таких завдань варто зазначити розв’язання лексичної багатозначності, створення систем питань та відповідей, об’єктний пошук. Однак для використання таких методів потрібні семантичні ресурси, такі як семантичні мережі.

Існуючі методи автоматичної та автоматизованої побудови семантичних мереж передбачають або наявність доступних готових ресурсів високої якості для інтеграції (ECO [26]), або утворюють лише поняття (виведення значень слів [9, 17], MaxMax [32], CPM [45]), або формують лише семантичні відносини (лексико-синтаксичні шаблони [30], підбір матриці лінійного перетворення [23]). Крім того, сучасні методи побудови зв’язків між поняттями базуються на побудові семантичної ієрархії та її вирівнювання щодо готової ієрархії [49], включаючи побудовану для іншої мови [43]. Це зумовлено неможливістю формування такої структури автоматичним шляхом без зовнішніх джерел через необхідність пов’язати знання про об’єкти навколишнього світу.

Найбільш близьким методом серед запропонованих є «Метод видобутку, кластеризації, онтологізації» (ECO), показаний на рисунку 1.2. Основною перевагою

методу ECO є його здатність створювати поняття та встановлювати зв’язки між ними на основі слабоструктурованих словників. Однак виявлено дві ключові проблеми у методі ECO:

– Недостатнє підтвердження ефективності використання методу нечіткої кластеризації графа.
– Залежність від наявних високоякісних семантичних ресурсів під час процедури онтологізації, яка формує зв’язки між синсетами [49].

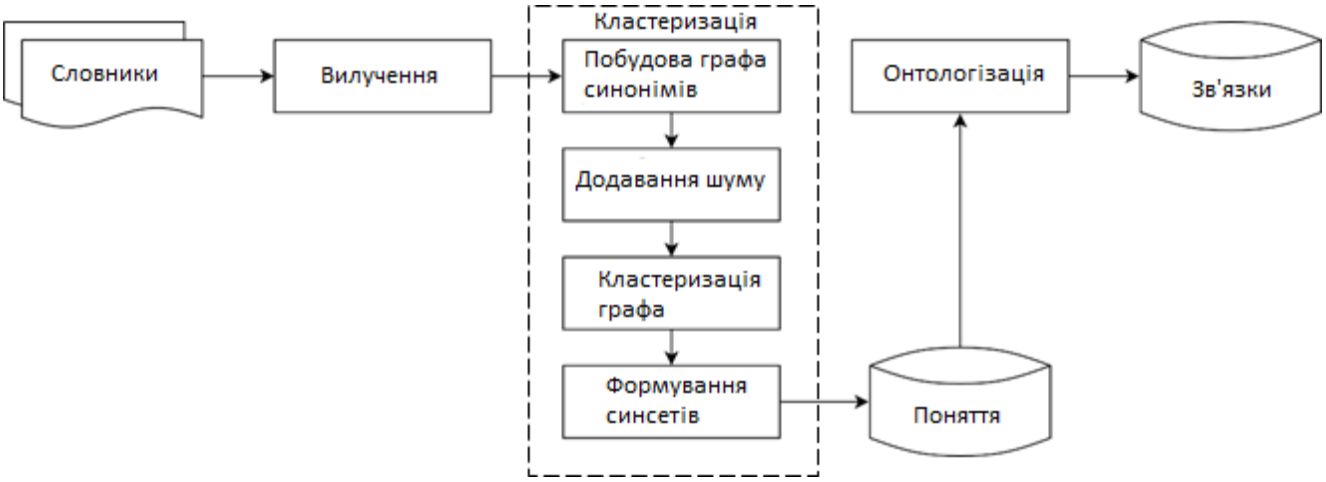

Рис. 1.2. Загальна схема методу «Вилучення, кластеризація, онтологізація»

Розв’язання цих проблем ставить завданням створення нових моделей, методів та алгоритмів для побудови семантичної мережі на основі матеріалів слабоструктурованих словників для обробки природної мови.

Розділ 2
Побудова семантичної мережі термінів

У традиційних семантичних мережах зв’язки формуються між окремими синсетами [20, 39, 62]. Джерелами даних для цих зв’язків є словники, створені

людьми, та словники, створені шляхом видобування пар слів із текстових корпусів [30]. Але недостатня доступність та повнота цих даних вимагає розширення лексичного покриття ресурсів. Це ускладнює використання процедур онтологізації [49] та методу ECO [27] для побудови зв'язків між синсетами через відсутність якісного тезаурусу для зіставлення цих зв'язків. Такі якісні семантичні ресурси не є загальнодоступними для всіх мов і потребують значного зусилля лексикографів.

Для вирішення проблеми обмеженої доступності даних та їх повноти пропонується не побудова зв'язків між усіма синсетами, а створення таких зв'язків між конкретними лексичними значеннями слів. Це відповідає сприйняттю людиною навколишнього світу, оскільки такі лексичні значення відповідають різним «концепціям слів» [44]. Таким чином, знання буде представлене у вигляді спеціальної моделі - семантичної мережі слів. У цій мережі семантичні зв'язки виникають між окремими лексичними значеннями слів. Це дає змогу як повторно використовувати, так і розширювати доступні лексико-семантичні ресурси для автоматичної побудови семантичної мережі слів без прямого втручання людини.

У цьому розділі описано модель представлення знань у вигляді семантичної мережі слів, а також запропоновано методи та алгоритми для її побудови. Запропонований метод побудови семантичної мережі слів включає метод створення синсетів на основі графа синонімів та метод формування зв'язків між значеннями слів. Ці моделі, методи та алгоритми призначені для вирішення проблем лексичної багатозначності та неповноти даних у мовних ресурсах. Мультимедійний контент пропонується пов'язувати до термінів тезауруса. Тобто, мультимедійний тезаурус представляє собою семантичну мережу термінів, їх значень та мультимедійних даних, що відображають сенс термінів.

2.1. Семантична мережа

Для створення семантичної мережі слів використовується змінений метод ECO [30], загальна схема якого наведена на рисунку 2.1. У процесі створення цієї семантичної мережі слів використовуються такі вхідні дані:

– словники зі слабкою структурою, що включають пари однозначних або багатозначних слів, які мають певне бінарне відношення в словнику. Ці словники необхідні для формування графа синонімів та його подальшої групування, а також для створення ієрархічних контекстів та з'єднання значень слів [56].
– нерозмічений корпус текстів для створення векторних представлень слів. Ці представлення використовуються для вагової оцінки графа синонімів на етапі групування та для розширення ієрархічних контекстів на етапі зв'язування.

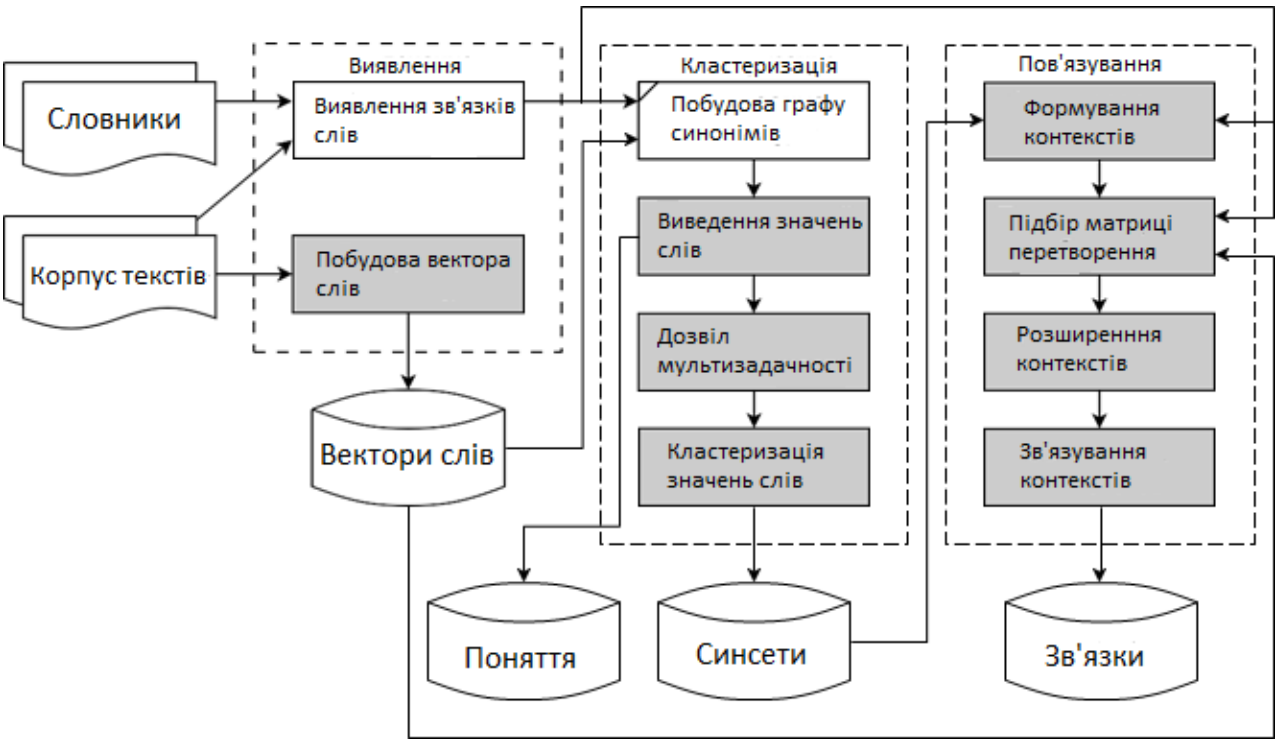

Рис. 2.1. Загальна схема методу побудови семантичної мережі слів: змінені блоки методу ECO помічені кутом зверху зліва; нові блоки виділені сірим кольором

Метод побудови семантичної мережі слів включає такі кроки:

– під час вилучення зв’язків, отримуємо зв’язки між словами з семантичних словників та корпусу текстів за стандартними методами [30]. Слова представляються у вигляді векторних моделей у просторі низької розмірності, використовуючи загальноприйняті підходи [55].
– на етапі кластеризації формуємо поняття семантичної мережі слів через побудову відповідностей з графом синонімів та уточнення значень слів, щоб вирішити їх багатозначність у контексті. Синсети створюються шляхом кластеризації графа значень слів.
– на фазі зв’язування реалізується зв’язування понять семантичної мережі слів через формування, розширення та зв’язування ієрархічних контекстів.

Цей метод приводить до створення мережі слів, де зв’язки між значеннями слова сформовані за допомогою семантики. Він відрізняється від ECO через використання слабоструктурованих словників та нерозмічених текстів. Ці словники включають пари слів з різними значеннями, що можуть мати симетричні (синонімія) або асиметричні (наприклад, «рід–вид», «частина–ціле») відношення. Корпус текстів служить для створення векторів слів за допомогою методів, таких як Skip-gram. На етапі вилучення рекомендується використовувати векторні представлення слів у низькорозмірному просторі, що створені на основі текстового корпусу, для вагомого оцінювання графа синонімів під час кластеризації та розширення ієрархічних контекстів під час зв’язування [42].

На етапі кластеризації пропонується використовувати новий спосіб створення груп слів на основі графа синонімів. Цей метод полягає у формуванні додаткового графа, де значення слів виводяться [9, 17, 46] з використанням процедури роз’яснення їх багатозначності в контексті [18]. Вершинами графа значень слів є самі ці значення,

а зв’язки утворюються внаслідок синонімічних відносин між лексичними значеннями. Створення груп слів відбувається за допомогою кластеризації цього додаткового графа за допомогою певного методу жорсткої кластеризації графа.

Замість процедури онтологізації пропонується використовувати спеціалізований метод зв’язування значень слів. За основу беруться наявні менш структуровані словники, з яких формуються ієрархічні контексти, що визначають вищі слова для слів у групах слів. Оскільки доступ до таких словників обмежений, розширення цих ієрархічних контекстів здійснюється за допомогою модифікованого методу вибору матриці лінійного перетворення [23]. Побудова семантичної мережі слів відбувається шляхом вибору значень слів у цих ієрархічних контекстах.

2.2. Метод побудови синсетів

На етапі кластеризації (рис. 2.1) відбувається формування груп слів за допомогою словника синонімів, що входить до початкових даних методу побудови семантичної мережі слів. Основне виклик у створенні цих груп полягає у врахуванні різноманітності значень слів: кожна група стає цільною для багатьох лексичних одиниць, які до неї належать [4].

Нехай словник синонімів $D \subseteq V \times V$ описує взаємозв’язок синонімії між словами. На основі цього відношення можна створити граф синонімів W, де зв’язані групи вершин вказують на однакове поняття [28, 35]. Для пошуку таких груп вершин у графі синонімів використовують методи клік [11] або визначення спільнот [57], але ці підходи не враховують різноманіття значень слів. Оскільки словник V містить як однозначні, так і багатозначні слова, у цьому розділі пропонується новий метод створення груп слів, який явно враховує полісемію на основі графа синонімів.

У наукових дослідженнях термін «синсет» зазвичай визначається як група слів, які мають однакове значення [19, 44]. Для спрощення позначень у цій роботі «елементами синсету» ми розумітимемо не самі слова зі словника V, а скоріше елементи набору лексичних значень слів V. Це уточнення визначення синсету звучить наступним чином.

Синсет $S \in \mathcal{S}$ - це набір $S \subseteq V$, в якому всі пари елементів S є синонімами один одного.

Якщо словник V складається тільки з однозначних слів, то можна використати алгоритм жорсткої кластеризації графа синонімів W для отримання необхідного набору синсетів $\mathcal{S}$. Наприклад, алгоритм «зіпсованого телефону» [9] чи кластеризація Маркова [35]. Проте в даній роботі таке припущення не використовується, тому кожному слову $u \in V$ відповідає набір значень $\text{senses}(u) \in V$, $|\text{senses}(u)| \geqslant 1$. З цим пов'язано пропозицію побудувати вспоміжний граф значень слів $W = (V, E)$. Оскільки вершинами у цьому графі є не слова, а значення слів, то застосування методу жорсткої кластеризації до графа W допоможе отримати набір синсетів $\mathcal{S}$.

Граф значень слів $W = (V, E)$ - це неорієнтований зважений граф, де множина вершин складається з лексичних значень слів, а множина ребер утворюється відповідно до синонімічних зв'язків у наборі лексичних значень слів.

Отже, постановка задачі створення синсетів використовує наступний підхід: знайти всі синсети $\mathcal{S}$ у графі W такі, що в кожному синсеті $S \in \mathcal{S}$ будь-яка пара значень слів $a \in S$, $b \in S$ перебуває у відношенні синонімії. Загальна схема

запропонованого методу створення синсетів подана на рисунку 2.2. і включає чотири етапи:

- побудова графа синонімів;

- вивід лексичних значень кожного слова;

- створення вспоміжного графа значень слів;

- кластеризація графа значень слів.

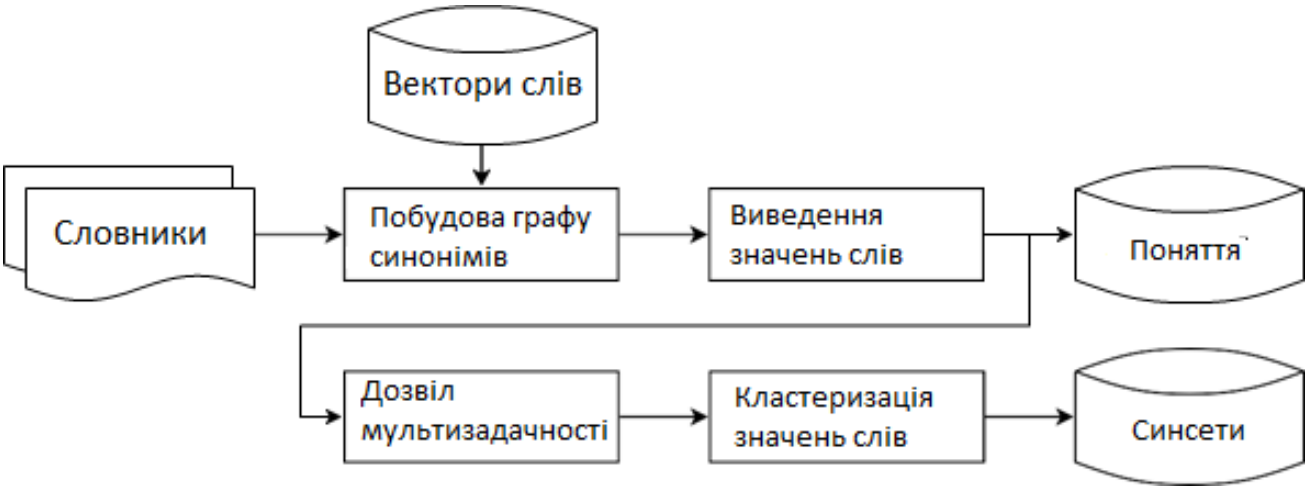

Рис. 2.2. Загальна схема методу побудови синсетів

2.2.1. Побудова графу синонімів

Граф синонімів $W = (V, E)$ формується на підставі словника синонімів $D \subseteq V \times V$, де множина ребер E графа включає двоелементні пари слів з словника синонімів:

$$E = \{\{u, v\} \in V \times V: (u, v) \in D, u \neq v\}. \quad (2.1)$$

Якщо задана певна міра семантичної близькості слів sim_{word}: $(u, v) \rightarrow 1R$, $u \in V$, $v \in V$, то вага ребер E обчислюється через визначення міри sim_{word} між словами, які відповідають вершинам, що інцидентні кожному ребру $\{u, v\} \in E$. Це може бути,

наприклад, косинус кута між векторними представленнями слів [53], або інша міра.

В результаті цих дій отримуємо зважений граф синонімів $W = (V, E)$, який служить основою для створення вспоміжного графа значень слів.

2.2.2. Виведення лексичних значень слів

В графі синонімів W є слова з одним і багатьма значеннями. Щоб визначити їх значення та створити набір таких значень V, запропоновано використовувати метод виведення значень слів на основі кластеризації оточень вершин, описаний у роботах [9, 17, 46].

Оточення вершини u у графі синонімів W - це неорієнтований граф $W_u = (V_u, E_u)$, який не містить саму вершину u:

$$V_u = \{v \in V : \{u, v\} \in E\}, \tag{2.2}$$

$$E_u = \{\{v, w\} \in E : v \in V_u, w \in V_u\}. \tag{2.3}$$

Для кожного слова $u \in V$ проводиться такий процес. Спочатку створюється оточення W_u з графа W. Потім W_u кластеризується за допомогою певного методу кластеризації. Кластери приймаються як значення слова u: $senses(u) = \{u^i : 1 \leqslant i \leqslant |C|\}$. Кожному значенню слова $u^i \in senses(u)$ приписується відповідний контекст $ctx(u^i) = C_i$. Контекст $ctx(u^i) \subset V$ - це набір синонімів слова $u \in V$ під номером $1 \leqslant i \leqslant |senses(u)|$.

Процедура виведення значень слів допомагає визначити їх значення та створити контексти, які представляють синоніми слів в відповідних лексичних значеннях. Також, ця процедура дозволяє створити набір понять V семантичної мережі слів, який є об’єднанням наборів лексичних значень слів:

$$V = \bigcup_{u \in V} senses(u). \tag{2.4}$$

На рис. 2.3 представлений приклад околиці слова «програма». При кластеризації цільове слово виключається. Це призводить до появи трьох кластерів, відповідно різним значенням цього слова:

{план, проєкт, графік}, {програмне забезпечення, застосунок, додаток}, {маніфест}.

Рис. 2.3. Приклад кластеризації округу слова «програма»

В таблиці 2.1. наведені результати виведення значення слова «програма» із прикладу на рисунку 2.3. : у колонці «Значення» представлені виявлені значення слова, в колонці «Контекст» перераховано контексти для кожного значення.

Таблиця 2.1.

Контексти слова «програма».

Значення	Контекст
програма[1]	план, проєкт, графік

програма[2]	програмне забезпечення, застосунок, додаток
програма[3]	маніфест

2.2.3. Побудова графа значень слів

Для побудови допоміжного графа значень слів $W = (V, E)$ необхідно зформувати множину його ребер E, породжену відношенням синонімії на множину лексичних значень слів V. Цього можливо досягти шляхом визначення значення слова в ребрах графа синонімів B = (V, E) з використанням контекстів значений слов.

Нехай задано деяку міру близькості контекстів sim_{ctx}: $(ctx(a), ctx(b)) \rightarrow R, \forall a \in V, b \in V$. Оскільки елементи контекстів – це слова без вказаних значень, то виробляється дозвіл багатозначності контексту кожного значення слова $c \in V$. Кожному елементу $u \in ctx(s)$ ставиться у відповідність значення $\hat{u} \in B$ з найбільш близьким контекстом:

$$\hat{u} \in \arg max_{u' \in senses(u)} sim\big(ctx(s), ctx(u')\big) \quad (2.5)$$

де senses(u) – множина всех значень слова $u \in V$. Далі, кожному елементу з $s \in V$ ставится в відповідність контекст з розшиореною багатозначністю $ctx(s) \subset V$:

$$ctx(s) = \{\hat{u} : u \in ctx(s)\}. \quad (2.6)$$

На основі отриманих контекстів зі отриманою багатозначністю формується множина ребер E графа значень слів $W = (V, E)$:

$$E = \{\{\hat{u}, \hat{v}\} \in V \times V : \hat{v} \in ctx(\hat{u})\}. \quad (2.7)$$

При побудові множини ребер E графа значень слова кожному ребру вказується вага, що дорівнює вазі ребра графа синонімів, інцидентного вершинам, іншим словам, значення яких визначились в даній процедурі.

На рисунку 2.4. представлено приклад графа значень слів, отриманого шляхом виведення значень слів (рис. 2.3) і дозволи неоднозначності в контекстах. В цьому прикладі слово «програма» бере участь в утворенні трьох синсетів, зв'язку с різними значеннями цього слова: {програма[1], план[1], проєкт[2], графік[3]}, {програма[2], програмне забезпечення[1], застосунок[2], додаток[3]} і {програма[3], маніфест[1]}.

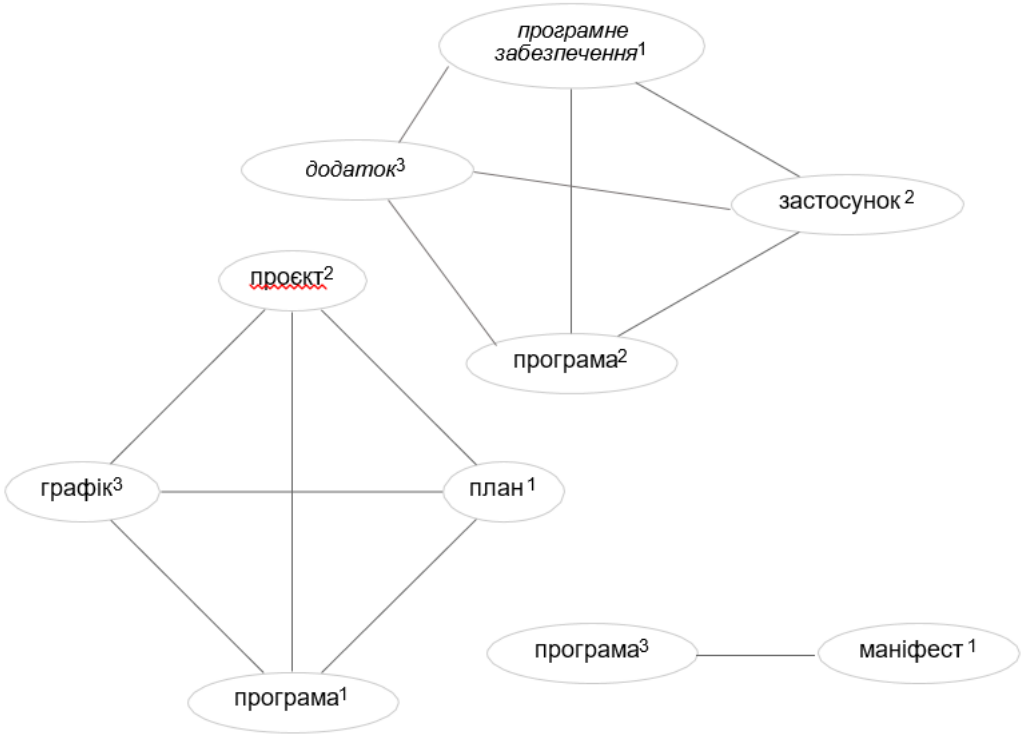

Рис. 2.4. Приклад графу значень слів

2.2.4. Кластеризація графу значень слів

Оскільки, ребра графу значень слів $W = (V, E)$ формуються за допомогою відношення синонімії між лексичними значеннями слів, можна зробити висновок, що

групи пов’язаних вершин у цьому графі також позначають одне й те саме поняття [47, 58]. В якості завершального кроку виконується кластеризація графа значень слів за допомогою якого-небудь алгоритму жорсткої кластеризації графа. Отримана в результаті кластеризації цього додаткового графа множина кластерів є шуканим набором синсетів S.

2.2.5. Алгоритм побудови синсетів

В якості вхідних даних алгоритм отримує на вхід словник синонімів $D \subseteq V \times V$. Результатом його роботи є набір лексичних значень слів V та набір синсетів S, які представляють собою кластери окремих лексичних значень слів. У алгоритму чотири гіперпараметри:

- $\text{Cluster}_{\text{Local}}$ - алгоритм жорсткої кластеризації, що використовується для кластеризації сусідніх вершин у графі синонімів при виведенні лексичних значень слів;
- $\text{Cluster}_{\text{Global}}$ - алгоритм жорсткої кластеризації, що використовується для пошуку синсетів у графі значень слів.
- sim_{word}: $(u, v) \rightarrow R$ – величина смислової близькості слів $u \in V$ u $v \in V$;
- sim_{ctx}: $(ctx(a), ctx(b)) \rightarrow R$ – величина близькості контекстів значень слів $a \in V$ u $b \in V$.

Алгоритм складається з кількох кроків, які допомагають побудувати граф синонімів, визначити значення певного слова і вирішити можливі неоднозначності в значенні слів. Використання жорстких алгоритмів кластеризації у цьому випадку виправдане припущенням, що кожна вершина графа може бути лише в одному кластері. Сам алгоритм не обмежує вибір конкретних методів жорсткої кластеризації графа, що дає

можливість використовувати широкий спектр алгоритмів, які відомі в області обробки мови.

Кроки виконання алгоритму:

Головная процедура.

Крок 1. **Побудувати граф синонімів** W;

Крок 2. Для всіх слів u $\in$ V виконати цикл

Крок 2.1. **Вивести значення слова** u;

Крок 3. Кінець циклу;

Крок 4. Побудувати множину значень всіх слів;

Крок 5. Для всіх значень слів s $\in V$ виконати цикл.

Крок 5.1. **Дозволити багатозначність контексту** s;

Крок 6. Кінець циклу;

Крок 7. Побудувати множину ребер: $E \leftarrow \{\{\hat{u}, \hat{v}\} \in V \times V : \hat{v} \in ctx(\hat{u})\}$;

Крок 8. Виконати кластеризацію графу $W = (V, E)$: $S \leftarrow Cluster_{Global}(W)$;

Крок 9. Кінець процедури.

Процедура побудови графу синонімів.

Дана процедура призначена для побудови графу синонімів. Вхідні дані для процедури – це множина довідників синонімів *D*. Результатом виконання процедури є граф синонімів W = (V, E), зважений за допомогою міри семантичної близькості слів sim_{word}.

Процедура виглядає наступним чином:

Крок 1.1. $V \leftarrow \bigcup_{(u,v)} \in D\{u, v\}$;

Крок 1.2. $E \leftarrow \{\{u, v\} \in V \times V: (u, v) \in D, u \neq v\}$;

Крок 1.3. Для всіх ребер $\{u, v\} \in E$ виконати цикл

Крок 1.3.1. $weight(u, v) \leftarrow sim_{word}(u, v)$;

Крок 1.4. Кінець циклу;

Крок 1.5. Кінець процедури.

Процедура виведення значень слова. Дана процедура призначена для визначення значень слова $u \in V$. Вхідними даними для процедури є граф синонімів $W = (V, E)$ та задане слово u. Результатом виконання процедури є множина senses(u), що містить усі виявлені значення слова u, причому для кожного виявленого значення складено контекст, що показує синоніми слова в даному значенні (див. приклад в таблиці 2.1).

Процедура выглядає наступним чином:

Крок 2.1.1. $senses(u) \leftarrow \emptyset$;

Крок 2.1.2. Дістати вершини околиці вершини u:

$V_u \leftarrow \{v \in V: \{u, v\} \in E\}$;

Крок 2.1.3. Дістати ребра околиці вершины u:

$E_u \leftarrow \{\{v, w\} \in E: v \in V_u, w \in V_u\}$;

Крок 2.1.4. Виконати кластеризацію графу $W_u = (V_u, E_u)$:

$C \leftarrow Cluster_{Local}(W_u)$;

Крок 2.1.5. $i \leftarrow 1$;

Крок 2.1.6. $ctx(u^i) \leftarrow C_i$;

Крок 2.1.7. $senses(u) \leftarrow senses(u) \cup \{u^i\}$;

Крок 2.1.8. Якщо $i < |C|$, то $i \leftarrow i + 1$ та перейти на крок 2.1.6;

Крок 2.1.9. Кінець процедури.

Процедура дозволу мультизадачності контексту.

Дана процедура призначена для побудови контексту з розширеною мультизадачністю для елемента s $\in V$. Вхідними даними для процедури є задане значення слова s та контексти значень всіх слів, що входять в контекст *ctx(s)*. Результат виконання процедури – контекст з розширеною мультизадачністю *ctx(s).*

Крок 6.1.1. $ctx(s) \leftarrow \emptyset;$

Крок 6.1.2. Для кожного слова в контексті u $\in$ ctx(s) виконати цикл;

Крок 6.1.2.1. $\hat{u} \leftarrow arg\ max_{u'} \in senses(u)\ sim_{ctx}(ctx(s), ctx(u));$

Крок 6.1.2.2. $ctx(s) \leftarrow ctx(s) \cup \{\hat{u}\};$

Крок 6.1.3. Кінець циклу;

Крок 6.1.4. Кінець процедури.

2.3. Метод побудови зв'язків

На етапі зв'язування відбувається формування чітких однобічних зв'язків між словами, які є частиною вихідних даних для створення семантичної мережі слів. Основна складність полягає у тому, що слова можуть мати декілька значень: у недостатньо структурованих словниках відсутня інформація про значення слів у парах, які належать асиметричним відношенням.

Позначимо *R* як асиметричне відношення, створене на основі словника. Якщо (w, h) ∈ R, то це впорядкована пара, де w ∈ V є словом, що перебуває на більш низькому рівні порівняно з h ∈ V. Інформація для створення відношення R представлена в матеріалах менш структурованих словників і не вказує на значення пов'язаних слів. Оскільки словник містить як однозначні, так і багатозначні слова, побудова множини зв'язків R у семантичній мережі лише на підставі цих елементів є неможливою.

У цьому розділі пропонується новий спосіб створення та розширення асиметричних семантичних зв'язків на основі ієрархічних контекстів, що

представляють найтиповіші вищі слова груп слів. У свою чергу, побудова семантичної ієрархії між такими групами є складною задачею, яка вирішується шляхом вирівнювання порівняно з іншими високоякісними семантичними мережами. Тому наявність зв'язків між окремими лексичними значеннями слів може допомогти створити множину зв'язків *R* у семантичній мережі слів N = (V, R).

Отже, постановка завдання формування зв'язків використовує наступну схему: для кожного синсету $S \in \mathbf{S}$ знайти набір вищестоячих значень слів $\widehat{hctx}(S) \subset V$ такий, що кожен елемент $\hat{h} \in \widehat{hctx}(S)$ є вищестоячим значенням відносно кожного елемента $s \in S$. Загальна схема запропонованого методу формування зв'язків представлена на рисунку 2.5 і включає чотири етапи:

- формування ієрархічних контекстів синсетів;

- вибір сімейства матриць лінійного перетворення;

- розширення ієрархічних контекстів;

- зв'язування значень слів за допомогою ієрархічних контекстів.

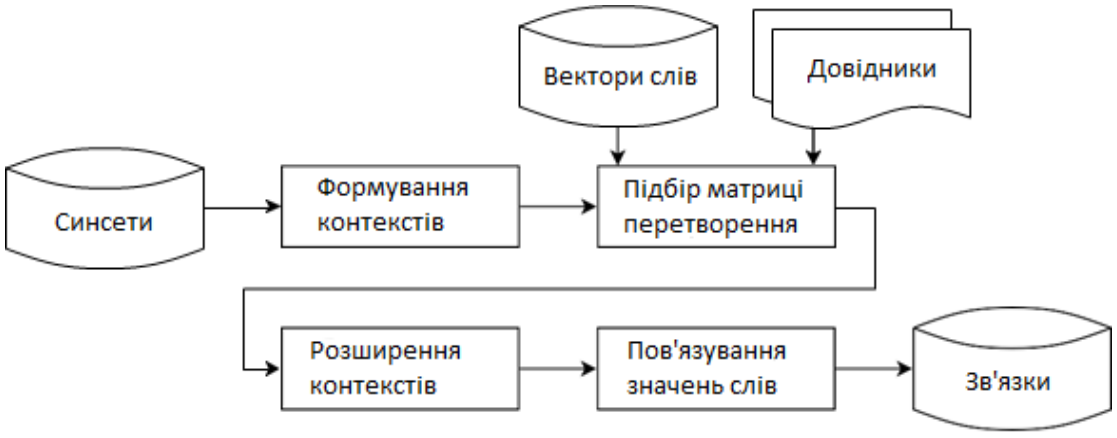

Рис. 2.5. Загальна схема методу побудови зв'язків

2.3.1. Побудова ієрархічних контекстів

Метод формування зв'язків ґрунтується на припущенні, що набори слів з вищих рівнів ієрархії для кожного елемента певного синсету, принаймні частково, співпадають. Це припущення виправдане, оскільки в поширених семантичних мережах елементами ієрархії є набори синонімів [19, 44]. Для цього вводиться поняття ієрархічного контексту синсету.

Ієрархічний контекст $hctx(S) \subset V$ синсету $S \in \mathbb{S}$ – це об'єднання множин слів з вищих рівнів ієрархії для кожного слова синсету S.

Нехай, $words(S) \subseteq V$ – множина слів значення який включені до синсету S. Тоді кожному такому синсету $S \in \mathbb{S}$ ставиться у відповідність ієрархічний контекст:

$$hctx(S) = \{h \in V : (w, h) \in R, w \in words(S), h \notin words(S)\}. \quad (2.8)$$

Оскільки значущість слів у ієрархічних контекстах відрізняється, пропонується використовувати міру tf-idf для вагового коефіцієнта елементів контексту. Ця міра широко застосовується в інформаційному пошуку [11]. Таким чином, у ієрархічному контексті hctx(S) сінсету $S \in \mathbb{S}$ вага кожного слова h ∈ hctx(S) обчислюється за формулою:

$$\text{tf–idf}(h, S, \mathbb{S}) = \text{tf}(h, S) \times \text{idf}(h, \mathbb{S}) \quad (2.9)$$

де tf(h, S) – частота слова h в синсеті S, idf(h, $\mathbb{S}$) – обернена частота слова h у множині синсетів $\mathbb{S}$. При цьому значення частоти слова h в синсеті S визначається як відношення кількості поява цього слова в ієрархічному контексті серед суми кількості поява інших слів у ньому:

$$tf(h,S) = \frac{|h' \in hctx(S) : h = h'|}{|hctx(S)|}; \quad (2.10)$$

В свою чергу, значення оберненої частоти документа виражається як відношення кількості усіх синсетів $|S|$ до кількості синсетів, ієрархічні контексти яких включають h:

$$idf(h,S) = log \frac{|S|}{|S' \in S : h \in \text{hctx}(S')|}; \quad (2.11)$$

У таблиці 2.2. наведені приклади ієрархічних контекстів для двох синсетів із словом «програма». Видно, що синсети містять інформацію про конкретні значення слів. Ієрархічні контексти, натомість, не містять такої інформації і містять багатозначні слова: слово «знак» може вживатися у значенні «дорожній знак», або у значенні «грошовий знак», і так далі.

Таблиця 2.2.

Приклад ієрархічних контекстів синсетів зі словом «програма».

Синсет	Ієрархічний контекст
{программа[1], план[1], …}	{документ, перелік, . . . }
{программа[2], застосунок[2], …}	{запис, інформація, . . . }
{программа[3], манифест[1], …}	{документ, заявка, . . . }

2.3.2. Розширення ієрархічних контекстів

Розширення ієрархічних контекстів призначене для додавання до ієрархічних контекстів слів, які мають схожий зміст з контекстом в цілому, але відсутні у взаємозв'язку *R*. Нехай $h \in hctx(S)$ – певне вищестояче слово ієрархічного контексту синсету S $\in S$. Нехай $\vec{h}$ - векторне представлення цього слова в просторі низької вимірності [55]. Нехай, $NN_n(\vec{h}) \in V$ – операція пошуку $n \in Z^+$ слів, векторні представлення яких відповідають найближчим сусідам векторного представлення $\vec{h}$

слова *h*. Оскільки в таких моделях, як Word2Vec, найближчими сусідами слів є синоніми, когіпоніми, партоніми та морфологічні варіанти лексичної одиниці [25, 55], необхідно перевіряти змістовність зв'язку слова-кандидата зі словами синсету. Тому для кожного синсету $S \in \mathcal{S}$ розширення ієрархічного контексту *hctx(S)* виконується у два етапи: формування кандидатів та їх перевірка.

Спочатку формується множина кандидатів $M_S \subset V$ шляхом об'єднання множини найближчих сусідів кожного елемента ієрархічного контексту *hctx(S)* без врахування слів, які вже є складовими контексту:

$$M_S = \bigcup_{h \in hctx(S)} NN_n(\vec{h}) \setminus hctx(S) \tag{2.12}$$

Нехай Φ^* - матриця така, що $\Phi^* \vec{w} = \vec{h}$, $\forall (w, h) \in R$. Перевірка ґрунтується на припущенні, що якщо слово $h \in M_S$ дійсно є вище стоящим у відношенні до будь-якого слова $w \in words(S)$, то вектор, отриманий множенням матриці Φ^* на векторне представлення нижчестоячого слова $\vec{w}$, знаходиться на євклідовій відстані від вектора вищестоячого слова $\vec{h}$, яка не перевищує певний заданий поріг $\delta \in R$ [23]. Таким чином, кандидат на вищестояче слово $h \in M_S$ додається до ієрархічного контексту hctx(S) лише тоді, коли виконується ця умова:

$$\exists w \in words(S) : \|\Phi^* \vec{w} - \vec{h}\| < \delta. \tag{2.13}$$

Розглянемо приклад на рисунку 2.6. Слово «організація» є відомим гіперонімом для слова «банк». Серед найближчих сусідів слова «банк» виділені слова «супермаркет», «корпорація», «установа» і так далі. Суть перевірки полягає у визначенні евклідової відстані між вектором, отриманим шляхом множення матриці Φ^* на векторне представлення слова «банк», та векторним представленням кожного слова-кандидата. У цьому випадку вектори лише двох слів не знаходяться далі від гіпероніма,

ніж на відстані δ: «корпорація» і «установа». Ці слова будуть додані до ієрархічного контексту, тоді як слово «супермаркет» не буде додано.

Рис. 2.6. Вибір слів в δ-радиусі слова «організація», що є вище стоящими по відношенню до слова «банк»: слово «супермаркет» не пройшло перевірку

2.3.3. Підбір матриці лінійного перетворення

Для поліпшення точності перетворення векторних представлень слів у векторні представлення вищестоячих слів пропонується модифікувати метод підбору матриці лінійного перетворення [23]. Відомо, що вектори семантично близьких слів розташовані достатньо близько один до одного, а вектори як вищестоячих, так і семантично не пов'язаних слів знаходяться достатньо далеко один від одного [42]. Навіть якщо знання про семантичну близькість між словами не дозволяє надійно передбачити тип семантичного зв'язку між словами, це спостереження дозволяє внести до даної моделі додаткову інформацію про взаємозв'язки слів. З властивості асиметрії відношення R випливає, що якщо у кожній парі слів $(w, h) \in R$ слово h є вищестоячим відносно слова w, то слово w не може бути вище стоящим відносно слова h. Для введення такої інформації в модель (1.18) вводиться член стабілізації H, вплив якого визначається значенням коефіцієнта λ:

$$\Phi_i^* \in \arg\min_{\Phi_i} \left(\frac{1}{|R|} \sum_{(\vec{w},\vec{h})\in R} \left\| \Phi_i \vec{w} - \vec{h} \right\|^2 + \lambda H \right) \quad (2.14)$$

Член стабілізації H збільшує значення функції, яку необхідно мінімізувати, з урахуванням припущення, що послідовне застосування цього лінійного перетворення до вектора $\Phi_i^* \vec{w}$ не повинно створювати вектор $\Phi^2\vec{w}$, близький до вихідного вектора $\vec{w}$:

$$H = \sum_{(\vec{w},\vec{h})\in R} ((\Phi^2\vec{w})^T\vec{w})^2. \quad (2.15)$$

Запропонований стабілізатор H не залежить від зовнішніх мовних ресурсів: для підбору матриці перетворення потрібно лише навчальний набір даних, представлений у вигляді відношення $R \subset V \times V$. Приклад цього спостереження показано на рис. 2.7.: відстань між словами «кіт» та «кішка», які близькі за значенням, є досить мала порівняно з відстанню від цих слів до гіперонімів «ссавець» та «тварина».

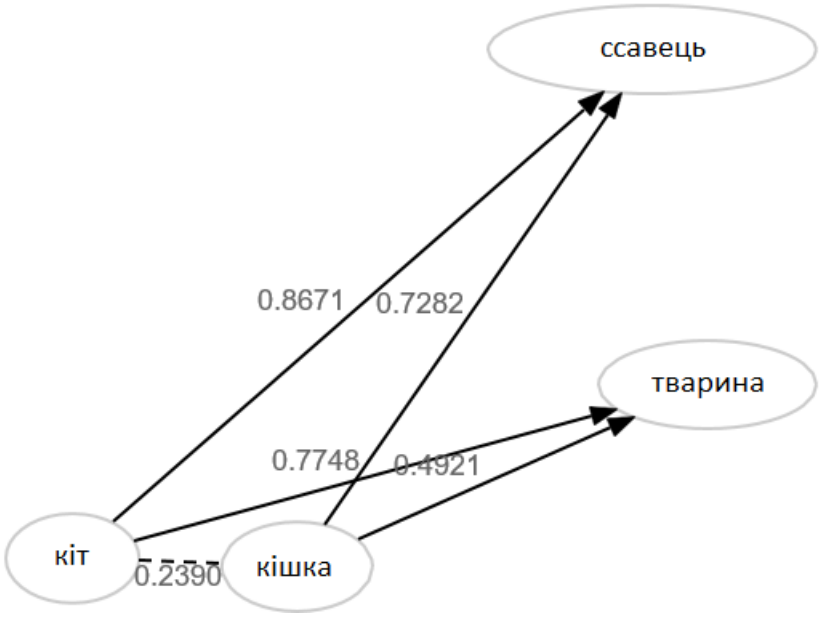

Рис. 2.7. Семантична близькість близьких и вище стоящих слів: в якості відстані використано косинусну відстань між відповідними векторами

2.3.4. Пов’язування ієрархічних контекстів

Для побудови семантичної мережі слів *N = (V, R)* потрібно сформувати множину стрілок *R*, що виникають з асиметричного відношення на множині лексичних значень слів *V*. Це можливо зробити шляхом вибору значень слів у ієрархічних контекстах синсетів.

Нехай задана певна міра близькості ієрархічного контексту та слів синсету sim_{hctx}: *(hctx(A), words(B)) → R,* $\forall A \in S, B \in S$. Оскільки елементами ієрархічних контекстів є слова без вказання значень, розв’язується багатозначність ієрархічного контексту кожного синсету S ∈ S. Кожному елементу h ∈ hctx(S) відповідає значення $\hat{h}$ ∈ V, включене в найближчий синсет:

$$\hat{h} \in \arg\max_{h' \in \text{senses}(h): S' \in \mathcal{S}, h' \in S', S \neq S'} \text{sim}_{\text{hctx}}(\text{hctx}(S), \text{words}(S')) \tag{2.16}$$

де words(S) – множина слів, значення яких включені в синсет S. Далі, кожному синсету S ∈ S ставится у відповідність ієрархічний контекст с дозволеною мультизадачністю $\widehat{hctx}(S) \in V$:

$$\widehat{hctx}(S) = \{\hat{h} : h \in hctx(S)\} \tag{2.17}$$

На основі синсетів і контекстів з вирішеною багатозначністю формується набір ребер *R* для семантичної мережі слів *N = (V, R)*, де вузли представляють лексичні значення слів, а набір ребер породжується асиметричним відношенням між лексичними значеннями слів:

$$R = \bigcup_{S \in \boldsymbol{S}} S \times \widehat{hctx}(S) \tag{2.18}$$

На рисунку 2.8. показано приклад семантичної мережі слів для слова «програма» з

кількома значеннями. Слова з різними значеннями, але співпадаючими лексемами, не мають загальних зв’язків.

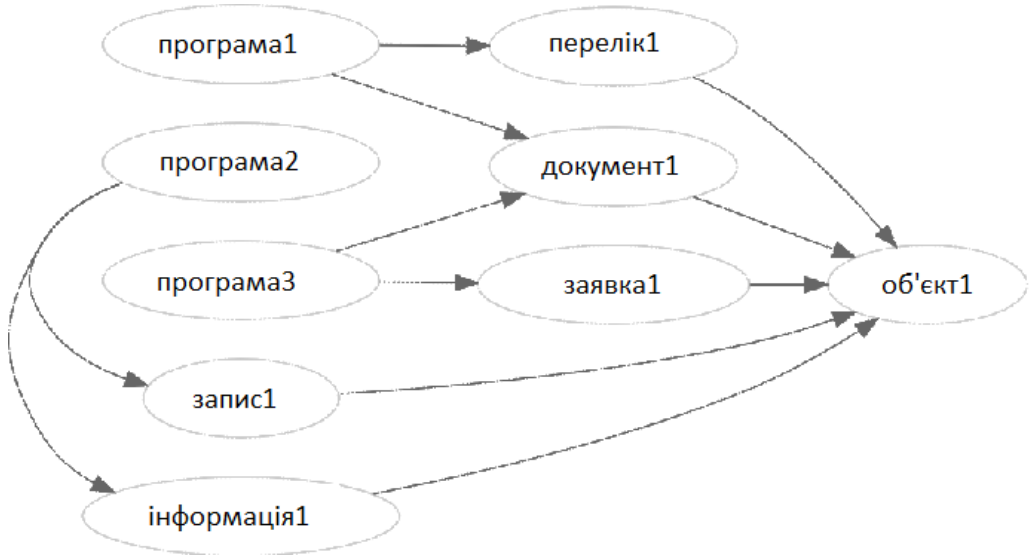

Рис. 2.8. Приклад фрагменту семантичної мережі слів: три різних значения слова «програма» не мають спільних зв’язків

2.3.5. Алгоритм побудови зв’язків

Вхідними даними для алгоритму є множина синсетів S, асиметричне відношення $R \subset V \times V$, та однозначне векторне представлення u кожного слова $u \in V$ у просторі $1R^d$, де $|V| \gg d$. Результатом работы алгоритма буде семантична мережа слів N. Алгоритм має пять параметрів:

- $n \in Z^+$ – кількість найближчих сусідів, отримуваних при розширенні контекстів;
- $k \in N$ – кількість вимірів при підборі матриці лінійного перетворення;
- $\lambda \in R$ – вплив стабілізації на функцію втрат при підборі матриці лінійного перетворення;
- $\delta \in R^+$ – максимальна відстань до найближчого сусіда, включеного при

розширенні ієрархічного контексту;

– sim_{hctx}: $(hctx(S), words(S')) \rightarrow R$ – міра близькості ієрархічного контексту $hctx(S) \subseteq V$ і слів синсету $S' \in S$: $words(S') \subseteq V$.

Алгоритм складається з основної процедури та трьох додаткових процедур підбору матриці лінійного перетворення, створення ієрархічного контексту синсету та вирішення багатозначності ієрархічного контексту.

Головна процедура.

В загальному вигляді, головна процедура виглядає наступним чином:

Крок 1. **Підібрати матрицю лінійного перетворення**;

Крок 2. Для всіх синсетів $S \in \mathbf{S}$ виконати цикл;

Крок 2.1. **Побудувати ієрархічний контекст синсету** *S*;

Крок 3. Кінець циклу;

Крок 4. Для всіх синсетів $S \in \mathbf{S}$ виконати цикл;

Крок 4.1. $tf\text{–}idf(h, S, \mathbf{S}) \leftarrow tf(h, S) \times idf(h, \mathbf{S})$;

Крок 5. Кінець циклу;

Крок 6. Для всіх синсетів $S \in \mathbf{S}$ виконати цикл;

Крок 6.1. Дозволити мультизадачність ієрархічного контексту *hctx(S)*;

Крок 7. Кінець циклу;

Крок 8. Побудувати за'язки між значеннями слів: $R \leftarrow \bigcup_{S \in \mathbf{S}} S \times \widehat{hctx}(S)$;

Крок 9. Побудувати семантичну мережу слів $N \leftarrow (V, R)$;

Крок 10. Кінець процедури.

Процедура вибору матриці лінійного перетворення.

Ця процедура призначена для вибору *k* матриць лінійного перетворення на основі методу [38] зі стабілізацією (2.14). Вхідними даними для процедури є $k \in N$ – кількість кластерів, R – асиметричне відношення, $\lambda \in R$ – важливість члена стабілізації H.

Результатом виконання процедури є *k* матриць $\Phi_i^*: 1 \leqslant i \leqslant k$. Використовується метод *k*-середніх для розбиття вихідного лінійного простору на *k* підпросторів [32], щоб врахувати його неоднорідності з використанням зсуву $(\vec{h} - \vec{w}), \forall(\mathrm{w}, \mathrm{h}) \in \mathrm{R}$ [23]. Процедура має вигляд наступним чином:

Крок 1.1. Для кожної пари слів $(w, h) \in R$ виконати цикл

Крок 1.1.1. $offsets(w, h) \leftarrow (\vec{h} - \vec{w})$;

Крок 1.2. Кінець циклу.

Крок 1.3. $C \leftarrow k\text{-}means(offsets, k)$;

Крок 1.4. $i \leftarrow 1$;

Крок 1.5. $\Phi_i^* \leftarrow \arg min_{\Phi_i} \frac{1}{|R_i|} \sum_{(\vec{w},\vec{h})\in R_i} (||\Phi_i \vec{w} - \vec{h}||^2 + \lambda((\Phi_i^* \vec{w})^T \vec{w})^2)$;

Крок 1.6. Якщо $i < k$, то $i \leftarrow i + 1$ та перейти на крок 1.4.

Крок 1.7. Кінець процедури.

Процедура побудови ієрархічного контексту. Ця процедура виконує побудову ієрархічного контексту синсету. Вхідними даними для процедури є синсет $S \in \mathbb{S}$ та k матриць лінійного перетворення, якщо запитано розширення контексту. Результатом виконання процедури є ієрархічний контекст hctx(S). Процедура має наступний вигляд:

Крок 2.1.1. $hctx(S) \leftarrow \{h \in V : (w, h) \in R, w \in words(S), h \notin words(S)\}$;

Крок 2.1.2. Сформувати множину слів-кандидатів у ієрархічний контекст:

$M_S \leftarrow \bigcup_{h\in hctx(S)} NN_n(\vec{h}) \setminus hctx(S)$;

Крок 2.1.3. Для кожної пари слів $(w, h) \in words(S) \times M_S$ виконати цикл;

Крок 2.1.3.1. Вибрати одну з *k* матриць лінійного перетворення Φ^* для пари слів (w, h) на основі зміщення $(\vec{h} - \vec{w})$;

Крок 2.1.3.2. Якщо $||\vec{\mathrm{w}}\Phi^* - \vec{\mathrm{h}}|| < \delta$, то $hctx(S) \leftarrow hctx(S) \cup \{h\}$;

Крок 2.1.4. Кінець циклу;

Крок 2.1.5. Кінець процедури.

Процедура вирішення багатозначності ієрархічного контексту. Ця процедура призначена для вирішення багатозначності ієрархічного контексту. Вхідними даними для процедури є синсет S ∈ S та його ієрархічний контекст hctx(S). Результатом виконання процедури є ієрархічний контекст синсету S із вирішеною багатозначністю hctx d(S).

Процедура має наступний вигляд:

Крок 6.1.1. $\widehat{hctx}(S) \leftarrow \emptyset$;

Крок 6.1.2. Для кожного слова $h \in hctx(S)$ виконати цикл;

Крок 6.1.2.1. $\hat{h} \leftarrow \arg max_{h' \in senses(h): S' \in \mathbf{S}, h' \in S', S \neq S'} sim_{hctx}(hctx(S), words(S'))$;

Крок 6.1.2.2. $\widehat{hctx}(S) \leftarrow \widehat{hctx}(S) \cup \{\hat{h}\}$;

Крок 6.1.3. Кінець циклу;

Крок 6.1.4. Кінець процедури.

Висновки

У розділі 2 розглянуто спосіб створення мережі слів з точки зору моделі представлення знань. Там розповідається про створення цієї мережі, метод формування груп слів за схожістю значень у синонімічному графі та спосіб встановлення зв'язків між різними значеннями слів. Описані моделі, методи та алгоритми мають на меті вирішення проблем з багатозначністю слів і нестачею даних у семантичних словниках.

Метод, що пропонується для створення груп слів, допомагає автоматично визначити різні значення слів та об'єднати їх у спільні групи, утворюючи вузли в мережі слів – концепції. Цей підхід відрізняється від інших тим, що створює додатковий граф значень слів через визначення та усунення багатозначності слів. Це

дозволяє використовувати методи жорсткої класифікації графів. На основі цього методу було розроблено алгоритм, який формує групи слів на основі матеріалів зі слабо структурованих словників, де вказується синонімічне співвідношення.

Метод утворення зв'язків, описаний у цьому розділі, дозволяє автоматично визначити найбільш відповідні вищестоячі слова для слів у групах. Це сприяє формуванню зв'язків у мережі слів. Цей метод відрізняється від інших тим, що розширює доступні лексико-семантичні ресурси та вирішує проблеми багатозначності слів за допомогою ієрархічних контекстів. На основі методу утворення зв'язків розроблений алгоритм, який створює та розширює семантичні зв'язки між значеннями слів на основі інформації зі слабо структурованих словників, де вказано асиметричні відношення.

Розділ 3
Комплекс програм побудови семантичної мережі слів

На основі описаних в другому розділі моделей, методів та алгоритмів розроблено комплекс програм для автоматичної побудови семантичної мережі слів. Комплекс програм представляє собою сукупність застосунків з консольним інтерфейсом, що дозволяє сформувати граф синонімів, визначити лексичні значення слів, побудувати граф значень слов, провести його кластеризацію, отримати синсети, розширити ієрархічні контексти, й записати результат работи у вигляді семантичної мережі слів.

В даному розділі описується архітектура та особливості реалізації моделей, методів і алгоритмів побудови семантичної мережі слів у вигляді комплексу програм.

3.1. Архітектура комплексу програм

Архітектура розробленого комплексу програм для автоматичної побудови

семантичної мережі слів представлена у вигляді UML-діаграми пакетів на рисунку 3.1. Для забезпечення тестовості програм та можливості запуску різних комбінацій параметрів запропонованих алгоритмів, комплекс програм має модульну структуру та включає чотири основні модулі:

- Модуль побудови синсетів, який реалізує алгоритм, описаний у розділі 2.2;
- Модуль зв’язування, який реалізує алгоритм, описаний у розділі 2.3;
- Модуль підбору матриці лінійного перетворення, який реалізує стабілізований метод, описаний у розділі 2.3.3;
- Модуль експорту даних, який реалізує перетворення семантичної мережі слів у стандартний формат подання семантичних мереж RDF [7].

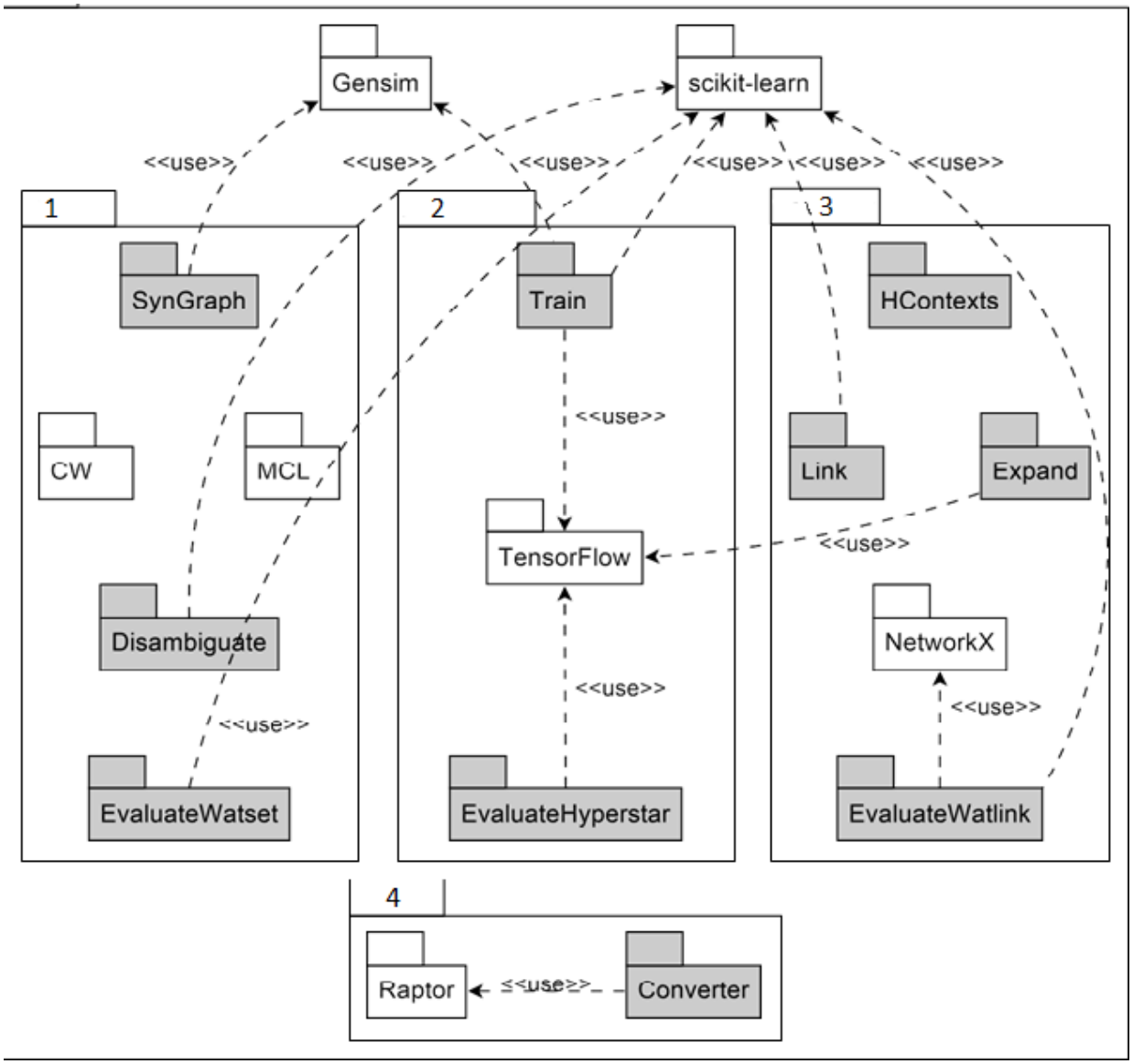

Рис. 3.1. UML-діаграма пакетів

До складу комплексу програм входять:

– програма побудови графа синонімів SynGraph;

– програма розв’язання багатозначності в контекстах Disambiguate;

– програма оцінки якості побудови синсетів EvaluateWatset;

– програма підбору матриці лінійного перетворення Train;

– програма оцінки якості підбору матриці лінійного перетворення

EvaluateHyperstar;

- програма побудови ієрархічних контекстів HContexts;
- програма розширення ієрархічних контекстів Expand;
- програма зв’язування значень слів Link;
- програма оцінки якості зв’язування значень слів EvaluateWatlink;
- програма перетворення семантичної мережі слів в формат Семантичної павутини Converter.

3.1.1. Модуль побудови синсетів

Модуль реалізує метод побудови синсетів на основі графа синонімів, описаний у розділі 2.2. На рисунку 3.2. представлена UML-діаграма алгоритму побудови синсетів, яка складається з трьох етапів:

- підготовка даних (SynGraph);
- побудова синсетів (CW, MCL та Disambiguate);
- тестування (EvaluateWatset).

Спочатку завантажуються матеріали слабоструктурованих словників і виділяється множина пар синонімів. За потреби, обчислюється значення семантичної близькості між парами синонімів на основі косинусної міри близькості між векторами слів. У випадку відсутності векторного представлення певного слова, використовується середнє значення близькості, обчислене для всіх пар слів. Ця інформація використовується під час побудови графа синонімів. У випадку відсутності інформації про семантичну близкість слів передбачено два альтернативні варіанти: використання одиничних ваг для кожного зв’язку в графі синонімів або підрахунок кількості згадок пари синонімів у вихідних словниках.

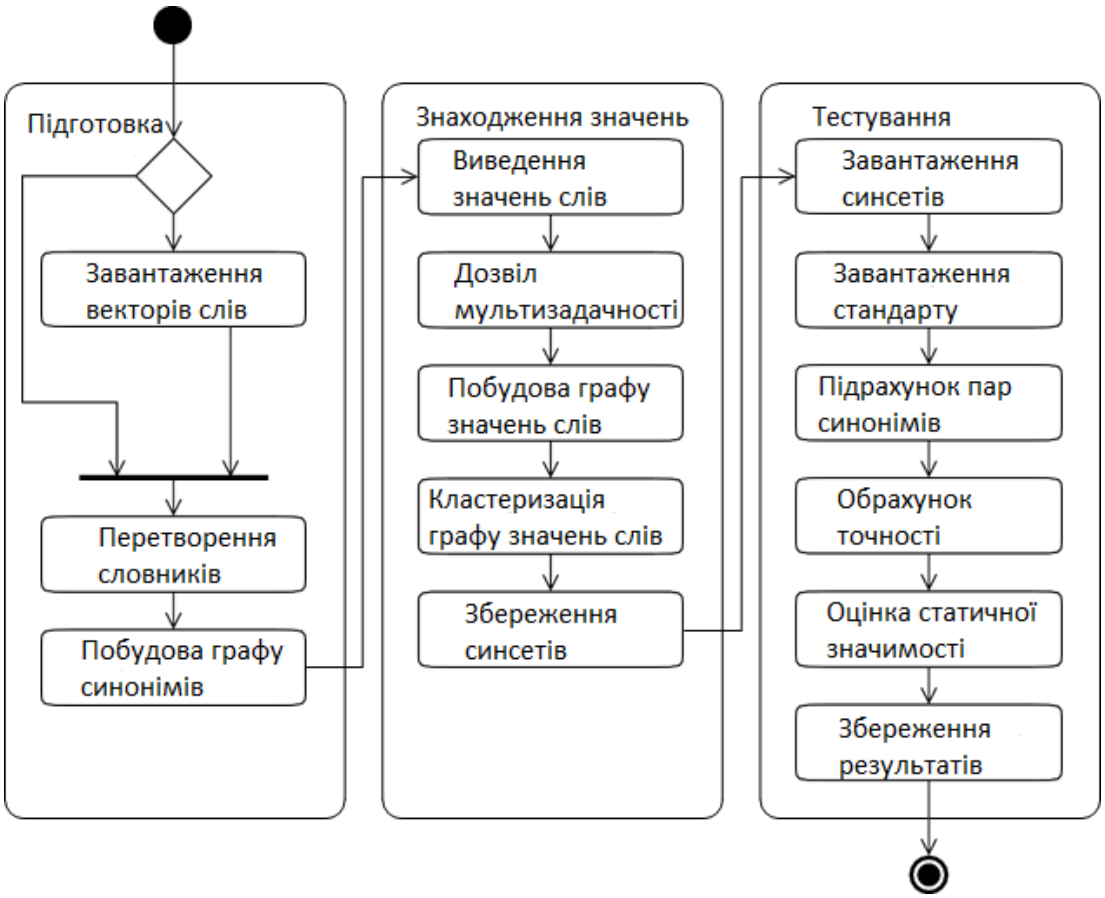

Рис. 3.2. UML-діаграма алгоритму побудови синсетів

На етапі виведення значень слів дозволяється використовувати два різних алгоритми жорсткої кластеризації графа: Chinese Whispers [9] або MCL [34]. Під час вирішення багатозначності проводиться роз'єднання багатозначності в контекстах, де для підвищення продуктивності використовується традиційний підхід паралельності за даними: кожне слово обробляється незалежно у окремому процесі. Визначення номера значення слова в контексті здійснюється шляхом максимізації косинусної міри близькості (2.5). Якщо лексичне значення слова в контексті визначити не вдасться, то це слово виключається з контексту. В результаті роз'єднання багатозначності формується граф значень слів, кластеризація якого для отримання синсетів виконується методом Chinese Whispers або MCL. Синсети отримують унікальні номери

і записуються у текстовий файл. Це необхідно як для використання даних у інших завданнях, так і для оцінки якості. При оцінці якості завантажуються побудовані синсети та синсети золотого стандарту. Потім кожен синсет з n значень слів перетворюється на множину з $\frac{n(n-1)}{2}$ пар слів, після чого обчислюється кількість збігів пар синонімів у отриманому ресурсі та золотому стандарті. Обчислюються значення парних інформаційно-пошукових критеріїв точності, повноти і F1-міри [33, 34], а також оцінюється статистична значимість кожного критерію (див. розділ 1.3). Після виконання всіх вказаних процедур результати оцінки записуються у текстовий файл.

3.1.2. Модуль подбору матриці лінійного перетворення

Модуль виконує підбір матриці лінійного перетворення векторних представлень нижчестоящих слів у векторні представлення вищестоящих слів на основі модифікованого підходу, спочатку запропонованого в [23]. На рис. 3.3. зображена UML-діаграма активності підбору матриці лінійного перетворення, що складається з трьох умовних кроків:

- підготовка даних (Train, режим підготовки);
- навчання моделі (Train, режим підбору параметрів);
- тестування (EvaluateHyperstar).

Вихідними даними для підбору матриці є вектори слів та впорядковані пари слів, утворені за асиметричним відношенням, отримані зі словників. У процесі використовуються лише ті пари слів, для яких є вектори. Це зумовлено тим, що вектори слів формуються на основі великого корпусу текстів із різними методами попередньої обробки, такими як фільтрація низькочастотних слів [3]. Отримані пари векторів слів розподіляються на три вибірки у співвідношенні: 60 % даних складають

навчальну вибірку для підбору параметрів, 20 % - перевірочну вибірку для підбору параметрів, і решта 20 % - тестову вибірку для оцінки якості моделі.

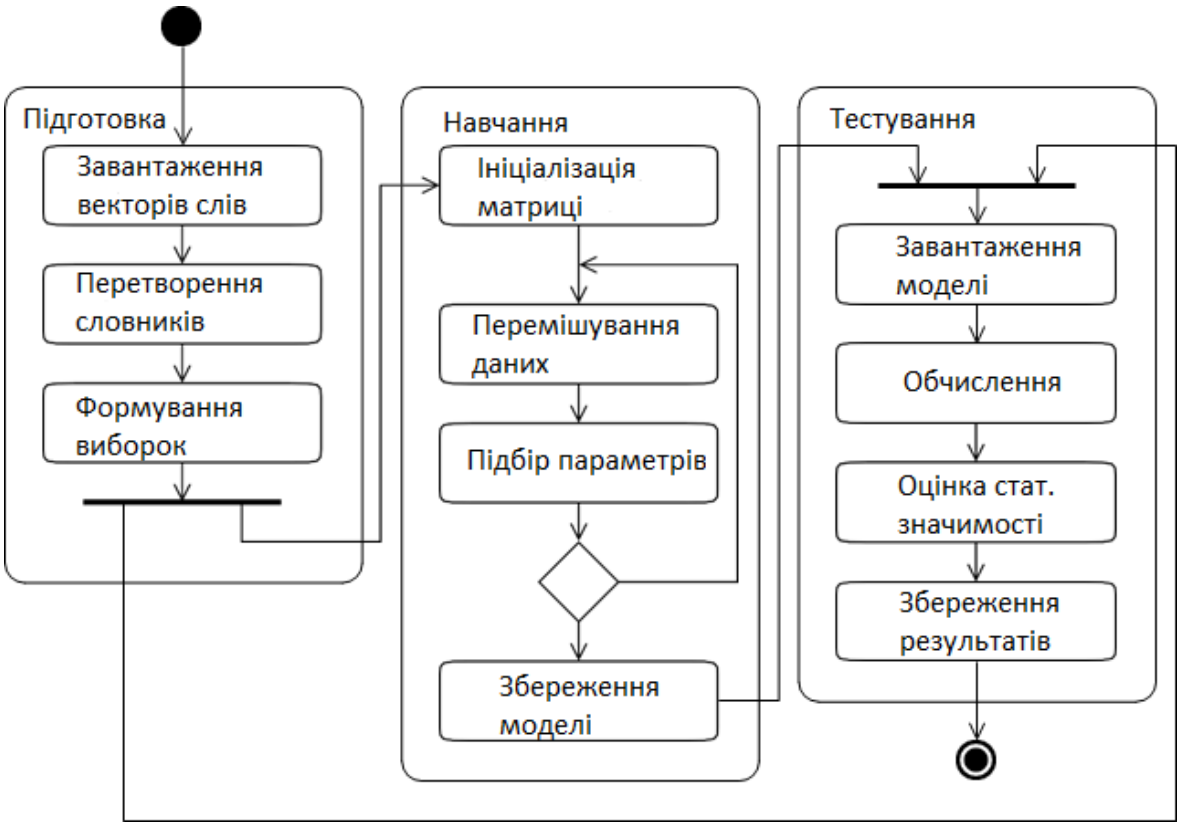

Рис. 3.3. UML-діаграма алготирму підбору матриці лінійного перетворення

На початку навчання всі елементи матриці генеруються як незалежні один від одного випадкові величини, що мають стандартний нормальний розподіл з параметрами $\mu = 0$ і $\sigma = 0{,}1$; жодні припущення про властивості матриці не використовуються [23]. На кожному кроці навчання дані перемішуються, і виконується підбір значень елементів матриці з метою мінімізації функції втрат (2.14). Процес навчання завершується при досягненні вказаної кількості кроків, передбаченої при запуску; отримане двійкове представлення матриці записується у файл.

Оцінка якості передбачає завантаження отриманих матриць і обчислення значення критерію hit@k на перевірочній вибірці для підбору параметрів або на

тестовій вибірці для оцінки якості роботи методу. Оцінюється статистична значимість значення цього критерію (див. розділ 1.3). Після виконання всіх зазначених процедур результати оцінки записуються у текстовий файл.

3.1.3. Модуль побудови зв’язків

Даний модуль втілює метод побудови зв’язків, описаний у розділі 2.3. Крім того, цей модуль створює семантичну мережу слів на основі раніше отриманих синсетів і доступних упорядкованих пар слів, породжених асиметричним відношенням. На рисунку 3.4. представлена UML-діаграма активності побудови зв’язків, що складається з чотирьох умовних кроків:

- підготовка даних (HContexts);
- розширення (Expand; необов’язковий крок);
- зв’язування (Link);
- тестування (EvaluateWatlink).

Вихідними даними для побудови зв’язків є синсети і матеріали слабоструктурованих словників, що містять перераховані у текстовому вигляді упорядковані пари слів, породжені асиметричним відношенням. На основі цих даних формуються ієрархічні контексти кожного синсета. За потреби завантажуються вектори слів, матриця лінійного перетворення, та розширюється ієрархічний контекст з використанням раніше отриманої матриці лінійного перетворення.

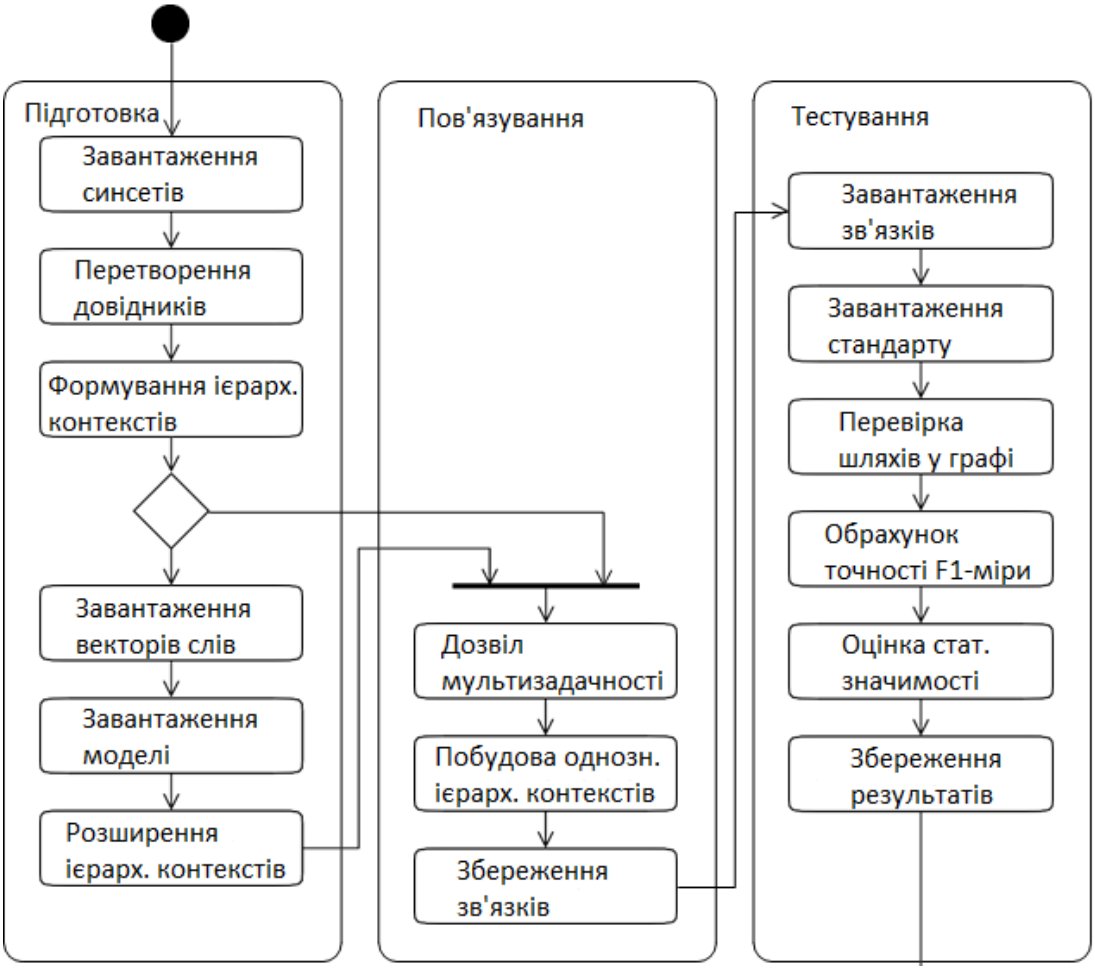

Рис. 3.4. UML-діаграма алгоритму побудови зв’язків

Дозвіл мультизадачності в ієрархічних контекстах реалізовано за допомогою трьох різних методів оцінки кожного вищестоячого слова в ієрархічному контексті: tf, idf та tf–idf [11], де «термін» означає слово, а «документ» – ієрархічний контекст синсету. Для збільшення продуктивності використовується традиційний метод паралельності по даним: кожен синсет обробляється незалежно у власному процесі. У ієрархічному контексті hctx(S) синсету S номер значення кожного слова $h \in hctx(S)$ визначається за допомогою максимізації косинусної міри близькості (2.16). Якщо лексичне значення слова в ієрархічному контексті визначити не вдається, то це слово виключається з ієрархічного контексту. Під час побудови ієрархічних контекстів зі знятою багатозначністю використовуються лише кілька елементів ієрархічного

контексту, які отримали максимальну вагу після розробки етапу розрішення багатозначності.

Семантична мережа слів зберігається у текстовий файл. При оцінці якості завантажуються побудовані зв’язки та зв’язки між словами золотого стандарту у вигляді орієнтованих графів. Потім для кожної пари слів перевіряється наявність шляху від нижчестоячого слова до вищестоячого в графі значень золотого стандарту. Обчислюються значення критеріїв інформаційно-пошукової точності, повноти та F1-мери [11], а також оцінюється статистична значимість кожного критерію (див. розділ 1.3). Після виконання всіх зазначених процедур результати оцінки записуються у текстовий файл.

3.2. Реализація комплексу програм

При реалізації комплексу програм були використані мови програмування Python, та Java. Завдяки наявності високоякісних бібліотек для вирішення завдань аналізу даних та вбудованій підтримці багатобайтових кодувань, основною мовою програмування було обрано Python версії 3. Зв’язування програм на різних мовах програмування здійснюється за допомогою сценаріїв командного інтерпретатора Bash. Операційною системою призначення є Linux.

Для підвищення швидкості розробки використовуються наступні зовнішні бібліотеки та залежності:

- бібліотека алгоритмів машинного навчання, підготовки та обробки даних scikit-learn [56] для мови програмування Python;
- реалізація алгоритму кластеризації Chinese Whispers [9] (CW) на мові програмування Java;

– бібліотека тематичного моделювання та роботи з векторами слів Gensim [52] для мови програмування Python;
– бібліотека методів оптимізації TensorFlow для мови програмування Python;
– бібліотека роботи з графами NetworkX [28] для мови програмування Python.

На рисунку 3.5. показана UML-діаграма варіантів використання комплексу програм для побудови семантичної мережі слів.

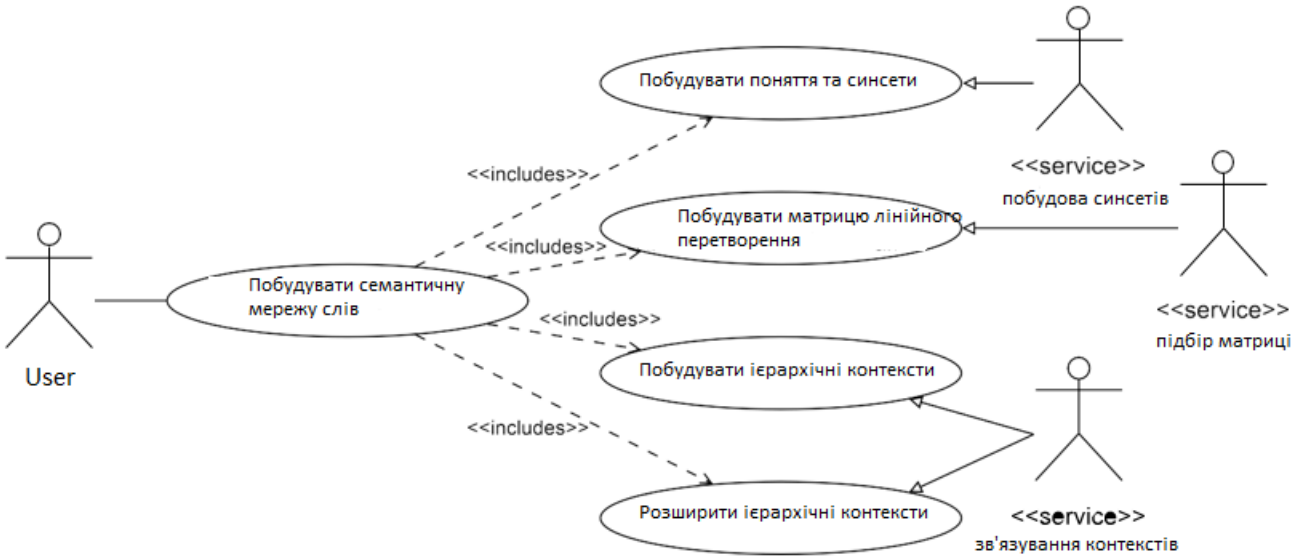

Рис. 3.5. *UML*-діаграма варіантів використання комплексу програм побудови семантичної мережі слів

Під час виконання операцій максимізації або мінімізації (формули 2.5, 2.14, 2.16) використовується аргумент максимізації або мінімізації з мінімальним ідентифікатором за інших рівних умов. У всіх випадках використовується косинусна міра близькості як міра схожості. Для представлення векторного уявлення контекстів, ієрархічних контекстів та слів у синсетах використовується загальноприйнята модель «мішок слів» [13].

Запис синсетів та семантичної мережі слів відбувається у текстових файлах у кодуванні UTF-8. Поля розділяються комами. Для представлення лексичних значень слів у текстових файлах використовується розширення нотації, яка використовується в BabelNet [55]. Запис словоі вказує на і-те значення слова, що належить частині мови t. Наприклад, запис лук2 позначає, що слово «лук» є іменником (англ. noun) і використовується в другому значенні («лук як стрілкова зброя»).

З одного боку, таке представлення в текстових файлах вимагає вказаних роздільників для трьох полів: лексеми, частини мови та номера значення. З іншого боку, таке представлення не дозволяє вказати словникові помітки та частоту зустрічання даного значення, хоча така інформація, особливо частотна, є дуже важливою для розв’язання практичних завдань [10].

Для вирішення цієї проблеми записів значень слів у текстові файли була використана контекстно-вільна граматика, побудована у вигляді розширеної форми Бекуса-Наура за допомогою утиліти ANTLR [46]. Ця граматика дозволяє виражати слова без значень (кіт), слова зі значеннями з вказанням та без вказання частини мови (кіт^NOUN#1 і кіт#1), а також словникові помітки (котик#1_пестлива_форма) та частоту (котик:10.5). Багатослівні вирази та множинні помітки розділяються підкресленням (_). Пробіли, табуляція та символи нового рядка не допускаються.

3.3. Представлення знань

Для забезпечення взаємодії та поєднання комплексу програм з зовнішніми інформаційними системами, семантична мережа слів описується за допомогою структури RDF (Resource Description Framework - Фреймворку Опису Ресурсів). У цій нотації знання виражаються у вигляді трійок «суб’єкт-предикат-об’єкт» [21]. Всі об’єкти, що отримані у результаті роботи розроблених методів, конвертуються до формату RDF-трійок, використовуючи моделі SKOS [19] та Lemon [41]. На діаграмі класів отриманої семантичної мережі слів, зглядно формалізму VOWL [47],

представлено на рисунку 3.6. У якості форматів зберігання використовуються текстові формати Turtle та N-Triples; для ідентифікації трійок застосовується префікс urn:swn.

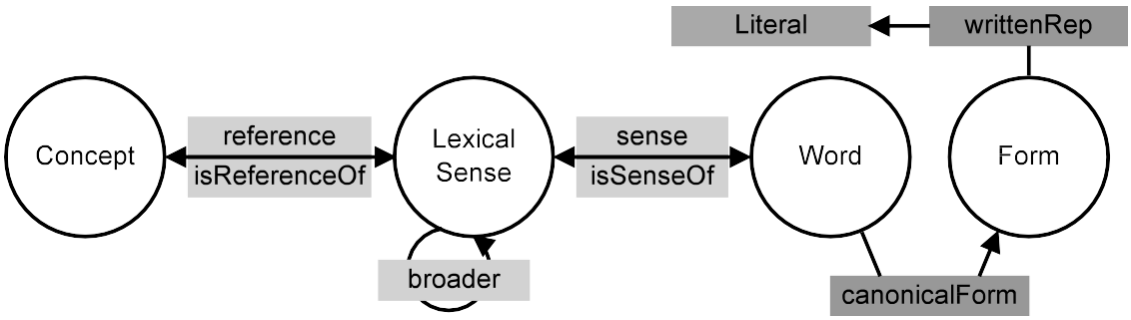

Рис. 3.6. VOWL-діаграма семантичної мережі слів

Перетворення даних полягає у записі кожного компонента даних у вигляді екземплярів класів, що відповідають моделям SKOS і Lemon (таблиця 3.1.). Спочатку кожне слово розглядається як екземпляр класу lemon:Word. Оскільки цей клас дозволяє слову мати різні форми, канонічна форма слова (лема) записується окремо як екземпляр класу lemon:Form. Між словом та його лемою утворюється властивість канонічної форми lemon:canonicalForm. Рядкове представлення слова записане як літерал, пов’язаний з екземпляром леми за допомогою властивості lemon:writtenRep. Лексичні значення слів записуються як екземпляри класу lemon:LexicalSense. Оскільки в семантичній мережі слів семантичні зв’язки формуються між окремими значеннями слів, нижче знаходящіся значення пов’язуються з вище розташованими за допомогою властивості lemon:broader та оберненої властивості lemon:narrower. Оскільки значення слів об’єднуються у синсети, кожне лексичне значення прив’язується до екземпляру класу skos:Concept, що відповідає синсету, за допомогою властивості lemon:reference та оберненої властивості lemon:isReferenceOf.

Таблиця 3.1.

Семантична мережа слів у вигляді зв’язаних даних.

Сутність	Назва	Клас
Слово	Лексична одиниця	lemon:Word
	Канонічна форма (лема)	lemon:Form
	Зв’язок одиниці та леми	lemon:canonicalForm
	Задане значення слова	lemon:sense
	Письмове представлення слова	lemon:writtenRep
Значення слова	Лексическое значение слова	lemon:LexicalSense
	Задана лексична одиниця	lemon:isSenseOf
	Задане поняття	lemon:reference
Гіперонім	Вищестояще значення слова	lemon:broader
Гіпонім	Нижчестояще значення слова	lemon:narrower
Поняття	Множина синонімів	skos:Concept
	Задане лексичне значення	lemon:isReferenceOf

3.4. Оцінка ефективності розроблених методів

У цьому підрозділі пропонується процес практичної перевірки ефективності описаних методів. Експерименти базуються на порівнянні результатів застосування цих методів із вже відомими та перевіреними даними – так званим золотим стандартом. Під час оцінки за золотим стандартом використовуються кількісні показники, які демонструють, наскільки добре досліджувані дані відповідають золотому стандарту (детальніші відомості про методику оцінки якості семантичних мереж можна знайти у розділі 1.3).

Поняття та зв’язки – це різні об’єкти, і немає загальноприйнятого методу для оцінки їх універсально. У цьому розділі експериментальна оцінка понять та зв’язків проводиться окремо:

– Оцінка понять передбачає, що слова, що утворюють групи слів у досліджуваному наборі даних, мають відповідність у золотому стандарті.
– Оцінка зв’язків вимагає, щоб зв’язки між словами у досліджуваному наборі даних відповідали зв’язкам у золотому стандарті.

Оцінка за золотим стандартом вважається вірною, коли пара слів у досліджуваному наборі даних або присутня також у золотому стандарті, або, навпаки, відсутня і там, і тут. Якщо в золотому стандарті відсутня певна пара слів, яка є у досліджуваному наборі, це може вказувати на некоректність визначення цієї пари слів. Такий підхід сприяє вищим оцінкам методів, які мають більше вірних збігів та менше помилок.

Метод формування синонімів базується на методі нечіткої кластеризації графу синонімів. Його ефективність оцінюється порівнянням з іншими методами кластеризації графів, застосовуючи три показники якості: парну точність, повноту та F1-міру. Для цього перетворюють як оцінювані дані, так і золотий стандарт на набори синонімів та порівнюють їх.

Метод побудови зв’язків перевіряється наявністю шляху між словами в графі золотого стандарту, згідно з описом у розділі 1.3. Цей підхід дозволяє розрахувати точність, повноту та F1-міру. Золотий стандарт тут перетворюється на семантичну мережу слів, яку порівнюють із наявністю шляху між вершинами у вхідних даних.

Метод підбору матриці лінійного перетворення для розширення ієрархічних контекстів оцінюється за загальноприйнятою методологією в оцінці методів машинного навчання з учителем. Для цього використовуються навчальні, перевірочні та тестові набори. Оскільки метод може повертати кілька можливих відповідей для кожного слова, використовується показник hit@k [22]. Він визначає правильну відповідь тоді, коли хоча б одна з перших k відповідей методу співпадає з тестовим

набором для конкретного слова. В таблиці 3.2. представлено методи та міри якості, що пропонується використовувати при експериментальній перевірці запропонованих алгоритмів.

Спочатку проводиться оцінка методу формування синонімів. На основі найкращої конфігурації цього методу проводиться експериментальна оцінка методу побудови зв'язків без розширення ієрархічних контекстів, а також стабілізованого методу вибору матриці лінійного перетворення. Потім, на основі найкращої конфігурації методу вибору матриці лінійного перетворення проводиться експериментальна оцінка методу побудови зв'язків з розширенням ієрархічних контекстів. В усіх експериментах пропонується використовувати 500-вимірні векторні представлення слів.

Таблиця 3.2.

Методи та міри якості при експериментальній перевірці.

Метод	Міра якості
Побудова синсетів	Попарна точність, повнота і F1-міра
Побудова зв'язків	Точність, повнота і F1-міра на основі перевірки існування шляхів у графі
Розширення зв'язків	Приклади з хоча б однією коректною відповіддю в десяти перших відповідях (hit@10)

3.5. Оцінка методу побудови синсетів

Для експериментальної оцінки методу, описанного в розділі 2.2, пропонується виконати порівняння з аналогічними методами по матеріалах двох різних золотих стандартів.

Як міра якості використовується парна точність, повнота та F1-міра. Для цього в золотому стандарті та оцінюваному наборі даних кожний синсет, що містить n ∈ N слів, перетворюється у $\frac{n(n-1)}{2}$ пар синонімів для оцінки [50]. Оцінка якості проводиться на основі присутності або відсутності певних пар синонімів у золотому стандарті. У цьому експерименті вважаються найкращими методи, які мають високі значення повноти та F1-міри.

3.5.1. Опис експерименту

Під час даного експерименту було проведено порівняння шести різних алгоритмів кластеризації графів:

- Описаний у розділі 2.2, це метод нечіткої кластеризації графів. Використано реалізацію на мові програмування Python, яка детально описана у підрозділі 3.1.1. В ході експериментів використовувалися різні параметри, зокрема:

  – $\text{Cluster}_{\text{Local}} \in \{\text{CW}_{\text{top}}, \text{CW}_{\text{nolog}}, \text{CW}_{\text{log}}, \text{MCL}\}$;
  – $\text{Cluster}_{\text{Global}} \in \{\text{CW}_{\text{top}}, \text{CW}_{\text{nolog}}, \text{CW}_{\text{log}}, \text{MCL}\}$;
  – $\text{sim}_{\text{word}} \in \{\text{ones, count, sim}\}$;$\text{sim}_{\text{ctx}} = \text{cos}$.

- Алгоритм зіпсованого телефону (англ. Chinese Whispers, сокр. CW): це метод жорсткої кластеризації графів [26]. Використана оригінальна реалізація на Java від авторів алгоритму, яка містить три варіанти:

  – CW_{top}: Оригінальний варіант методу;
  – CW_{nolog}: Варіант методу з використанням степені вершини для призначення кластерів;
  – CW_{log}: Той самий метод, але з використанням натурального логарифма степені вершини.

– Кластеризація Маркова (MCL): це метод жорсткої кластеризації графів [31]. Використано оригінальну реалізацію на мові програмування *C* від автора методу.
– MaxMax: Це метод нечіткої кластеризації графів [29]. Оскільки його реалізація не загальнодоступна, використано власну реалізацію на Java, побудовану на основі псевдокоду з оригінальної статті.
– ECO: Це метод нечіткої кластеризації графів [27]. Оскільки його реалізація не є загальнодоступною, використано власну реалізацію на мові програмування Python, яка базується на короткому описі, наданому авторами оригінальної статті. У зв'язку з відсутністю докладної інформації, ймовірність того, що слова u та v потрапляють до одного кластера, порівнюється з визначеним пороговим значенням та відповідно оцінюється як:

$$p_{u,v} = \frac{\#(u,v)}{\#(u) + \#(v) - \#(u,v)'};$$

де #(u, v) – кількість випадків зустрічання слів u та v у тому ж самому кластері, #(u) та #(v) – загальна кількість зустрічей слів u та v відповідно.

– Метод перколяції клік (CPM), який є алгоритмом нечіткої кластеризації графів [43]. Використана реалізація цього алгоритму на мові програмування Python з бібліотеки NetworkX [12]. Алгоритм призначений для невагованих графів, тому під час експериментів ваги ребер ігнорувалися. Використовувалися наступні значення гіперпараметра, що вказує на розмір мінімальної кліки: $k \in \{2, 3, 4\}$.

Для визначення значень слів ($Cluster_{Local}$) використовувався алгоритм кластеризації Маркова, а для кластеризації графа значень слів ($Cluster_{Global}$) використовувалася оригінальна версія алгоритму «Chinese Whispers».

У експерименті пропонується використати доступні україномовні словники синонімів як вихідні дані.

На основі об’єднання цих вихідних словників буде побудовано об’єднаний граф синонімів – це неорієнтований граф. Щоб вивчити вплив ваг ребер графа синонімів $W = (V, E)$ на результат кластеризації, у експерименті пропонується розглянути три різних міри схожості слів sim_{word}:

– ones: надання однакової ваги кожному ребру $\{u, v\} \in E$:

$$\text{weight}(u, v) = 1;$$

– count: кожному ребру $\{u, v\} \in E$ призначається вага, рівна кількості появ відповідної пари слів в словниквх синонімів D:

$$\text{weight}(u, v) = \sum_{D \in D}]\_D((u, v)),$$

де $]\_D$ – індикаторна функція множини D;

– sim: кожному ребру $\{u, v\} \in E$ призначається вага, рівна значенню косинуса кута між векторними представленнями слів:

$$\text{weight}(u, v) = cos(\square u, \square v).$$

Оскільки, словник використовуваних словників відрізняється від словника золотого стандарту, під час обчислення інформаційно-пошукових оцінок слід використані лише ті пари синонімів, в яких обидва слова входять в перетин словника золотого стандарту та об’єднаного словника наборів даних, отриманих у результаті виконання методів кластеризації.

3.6. Оцінка методу побудови зв’язків

Для перевірки ефективності методу, описаного у розділі 2.3, проводиться порівняння якості на п’яти різних наборах даних, що включають пари слів, створені

асиметричним відношенням, до та після використання запропонованого методу на основі золотого стандарту. Використано точність, повноту та F1-міру, розраховані методом, що базується на перевірці наявності шляху в графі (розділ 1.3), як метрики якості. Оцінюваний набір даних та золотий стандарт перетворюються у семантичну мережу слів. Якість кожного зв'язку у наборі даних визначається наявністю шляху від меншого слова до більшого слова в графі золотого стандарту. Зв'язок вважається встановленим правильно, якщо існує шлях від меншого значення слова до більшого значення слова в золотому стандарті. У експерименті використовуються лише родо-видові зв'язки через їх доступність та поширеність. Оцінка якості базується на наявності або відсутності шляху від меншого слова до більшого слова в золотому стандарті. Зв'язок між парами слів в оцінюваному ресурсі вважається коректним, якщо в золотому стандарті існує шлях від одного значення слова до іншого (див. розділ 1.3). Найефективнішими в цьому експерименті вважаються методи, що отримали високі значення повноти та F1-міри.

3.6.1. Опис експерименту

Конфігурація методу використовує алгоритм «пошкодженого телефону» для виводу значень слів та марківський алгоритм кластеризації для кластеризації графа значень слів. Синсети, отримані під час попереднього експерименту, використовуються для побудови ієрархічних контекстів. В експерименті використовуються такі значення гіперпараметрів методу без застосування розширення ієрархічних контекстів:

- кількість найближчих сусідів при розширенні: $n = 0$;
- міра близькості ієрархічних контекстів: $sim_{hctx} = cos$.

Оскільки у цьому експерименті не відбувається розширення ієрархічних контекстів, значення гіперпараметрів k, λ і δ не впливають на результати. Для вивчення впливу ваги слів на побудову ієрархічних контекстів у експерименті розглядаються три різних підходи до вагового коефіцієнта слів у ієрархічних контекстах:

– tf–idf: використовується значення tf–idf згідно з формулою (2.9);
– tf: при розрахунку tf–idf значення idf приймається за одиницю, тобто

$$\text{tf–idf}(h, S, S) = \text{tf}(h, S) \times 1;$$

– idf: при розрахунку tf–idf значення tf приймається за одиницю, тобто

$$\text{tf–idf}(h, S, S) = 1 \times \text{idf}(h, S).$$

Використовуються чотири різних наборів даних з родовидовими зв'язками між словами:

– пари слів, отримані з електронної бібліотеки за допомогою лексико-синтаксичних шаблонів загального призначення [81] (далі – «шаблони»);
– пари слів із набору даних «шаблони», які зустрічаються не менше тридцяти разів у колекції документів (далі – «шаблони + Ч»);
– пари слів, отримані з тлумачень Малого академічного словника [15] за допомогою спеціалізованих лексико-синтаксичних шаблонів [5] (далі – «МАС»);
– об'єднання трьох ресурсів з видаленням дублікатів пар: «шаблони + Ч» і «МАС» в єдиний набір даних (далі – «Всі словники»).

Оскільки словник використаних словників відрізняється від словника золотого стандарту, то при розрахунку інформаційно-пошукових оцінок використовувалися лише ті пари слів, обидва слова яких входять в перетин словника золотого стандарту та об'єднаного словника наборів даних, отриманих під час виконання методів побудови зв'язків.

3.7. Оцінка методу підбору матриці лінійного перетворення

Для експериментальної оцінки методу розширення зв’язків проводиться порівняння базового методу вибору матриці лінійного перетворення [38] з його стабілізованою варіацією, що представлена у розділі 2.3.3. Як міра якості використовується метрика hit@k [37]. Для оцінки методу вибору матриці лінійного перетворення пропонується використовувати матеріали україномовного Вікісловника [8]. Отже, для кожного підпорядкованого слова метод генерує k відповідей, що відповідають вищестоящим словам. Якщо множина відповідей містить правильну відповідь, то таке підпорядковане слово вважається обробленим правильно. У цьому експерименті кращим вважатиметься метод, який досягне високих значень hit@k.

3.7.1. Опис експерименту

Відповідно до загальноприйнятої методології навчання з учителем у машинному навчанні, експерименти проводяться на трьох вибірках: навчальній, перевірочній та тестовій. Щоб уникнути ефекту лексичного перенавчання (англ. lexical overfitting), який змушує метрику якості зростати [41], поділ вихідного набору даних на основі україномовного Вікісловника на три вибірки був здійснений таким чином, що жодна з отриманих вибірок не містить підпорядкованого слова зі спільними вищестоящими словами. З метою розширення навчальної вибірки, не порушуючи цей принцип, були додані пари слів з набору даних «Шаблони + Ч» [40], що описані у розділі 3.6.

На перевірочній вибірці відбувається підбір гіперпараметрів, які використовуються для порівняння результатів роботи базового та стабілізованого методів на тестовій вибірці:

– кількість кластерів: $1 \leqslant k \leqslant 30$;

– вплив стабілізатора: $\lambda \in \{10l : l \in \{-1, 0, 1, 2\}\}$.

3.8. Оцінка методу побудови зв'язків з розширенням

Для експериментальної оцінки методу, який використовує розширення ієрархічних контекстів, застосовується та ж сама стратегія, що й у експерименті з побудовою зв'язків без їхнього розширення. Найкращими у цьому експерименті вважаються методи, які отримали високі значення повноти та F1-міри.

3.8.1. Опис експерименту

У цьому експерименті використовуються ієрархічні контексти, отримані за допомогою підходу з взважуванням tf–idf на основі синсетів, отриманих за допомогою методу побудови синсетів. Для розширення використовується сімейство матриць лінійного перетворення, отриманих у рамках експерименту у підрозділі 3.7. Крім того, досліджуються наступні значення гіперпараметрів методу побудови зв'язків:

– кількість найближчих сусідів для розширення: $n = 10$;
– кількість кластерів для вибору матриці лінійного перетворення: $k = 20$ (оптимальне значення, отримане у розділі 4.3);
– вплив стабілізації на функцію втрат під час вибору матриці лінійного перетворення: $\lambda = 1$ (оптимальне значення, отримане у розділі 4.3);
– максимальна відстань до найближчого сусіда: $\delta = \{ r : r \in N, r \leqslant 10 \}$;
– міра схожості ієрархічних контекстів: $sim_{hctx} = cos$.

Оскільки набір даних «шаблони» містить значну кількість некоректно визначених семантичних відносин, цей набір даних був виключений з експерименту. Натомість використовувався лише похідний набір даних із фільтрацією за частотою «шаблони + Ч».

Висновки

У розділі 3 наведено опис комплекс програм, що реалізує методи, моделі та алгоритми, запропоновані у розділі 2. А також, описано методологію проведення практичних обчислювальних експериментів, за підходом порівняння зі золотим стандартом. Комплекс програм здійснює побудову семантичної мережі слів та її запис у форматах Семантичної Павутини на основі інформаційної моделі (рис. 3.6). Описана архітектура комплексу програм, що включає в себе програми для побудови синсетів, формування та розширення семантичних зв'язків.

Програми для створення семантичної мережі слів написані з використанням паралельності за даними, що дозволяє використовувати обчислювальні вузли з великою кількістю доступних ядер центрального процесора для прискорення обчислень. Крім того, для реалізації методів використані високоефективні зовнішні бібліотеки, такі як scikit-learn [49] і TensorFlow [16]. Всі програми працюють у режимі командного рядка. Зв'язування програм, написаних на різних мовах програмування, здійснюється через перенаправлення потоків стандартного вводу та виводу у сценаріях оболонки командного процесора Bash та утиліти make.

Вхідні дані представлені у вигляді текстових файлів, поля яких розділені символом табуляції. Усі проміжні результати, крім матриці лінійного перетворення, також представлені у текстовому вигляді. Кінцевий результат записується у стандартному форматі подання семантичних мереж у вигляді N-Triples [20].

ВИСНОВКИ

У магістерській роботі розглянуто питання розробки та вивчення ефективних методів автоматичної побудови семантичної мережі для мультимедійного тезауруса. Досліджено сучасні підходи до автоматичної побудови семантичних ресурсів. Запропоновано модель семантичної мережі слів, яка з'єднує лексичні значення слів за допомогою семантичних зв'язків з дозволеною багатозначністю. На її основі розроблені методи та алгоритми автоматичної побудови понять і автоматичного розширення семантичних зв'язків. Наведено детальний опис для практичної експериментальної перевірки запропонованих алгоритмів побудови семантичної мережі. Розроблені моделі, методи та алгоритми реалізовані у вигляді комплексу програм, який працює на багатоядерних та багатопроцесорних обчислювальних системах для виконання ресурсозатратних операцій.

Основні результати, отримані під час виконання дослідження у дисертації, є новими та не охоплені раніше опублікованими науковими працями інших авторів, обзор яких було надано в розділі 1. Слід відзначити основні відмінності.

Існуючі методи побудови синсетів на основі нечіткої кластеризації графа, такі як MaxMax [55], CPM [58] і ECO [49], не виконують процедуру виведення значень слів у явному вигляді та спрямовані на кластеризацію графів спільної зустрічності слів. Методи виведення значень слів [26, 36, 59], з іншого боку, не вирішують багатозначність отриманих значень слів та не використовують ці значення слів для побудови понять. Існуючий метод вирішення багатозначності в контекстах [38] не передбачає побудову графа значень слів. Описаний у розділі 2.2 метод виявлення понять відрізняється тим, що він використовує існуючий метод виведення значень слів, після чого будує граф значень слів за допомогою значень слів з дозволеною

багатозначністю, а потім здійснює жорстку кластеризацію отриманого графа значень слів за допомогою добре відомих методів жорсткої кластеризації графа [26, 35].

Існуючі методи побудови зв’язків, такі як онтологізація [50], ECO [26] і BabelNet [59], передбачають побудову зв’язків між синсетами на основі попередньо підготовленої семантичної ієрархії високої якості. У обох випадках використовується тезаурус англійської мови WordNet [2]. Описаний у розділі 2.3 метод побудови зв’язків не потребує такого ресурсу для вирішення задачі. Методи видобутку зв’язків, насамперед, шаблони [33, 48] і їх варіації для тлумачних словників [5], не вказують конкретні значення слів, що призводить до виникнення лексичної багатозначності. Так само обмеженням володіє Вікісловник та інші загальнодоступні ресурси, побудовані за допомогою краудсорсингу. Запропонований у цій роботі метод побудови зв’язків призначений для вказання конкретних значень пов’язаних слів. Крім того, підхід до розширення ієрархічних контекстів у цьому методі дозволяє додати додаткові зв’язки, які відповідають за змістом.

Методи підбору матриці лінійного перетворення для пошуку вищестоящих слів на основі векторних представлень нижчестоящих слів почали розвиватися відносно недавно. Базовий метод підбору матриці лінійного перетворення [7] не враховує явно асиметрію семантичних зв’язків. Описаний у розділі 2.3.3 метод стабілізації функції втрат, що використовується для підбору елементів матриці лінійного перетворення, дозволяє включити в модель додаткову інформацію про зв’язки слів у вигляді негативних прикладів. Інші існуючі методи видобутку зв’язків на основі векторних представлень слів [52] потребують працезатратну операцію повного синтаксичного розбору тексту.

Можна виділити наступні напрямки подальших досліджень:

– використання векторних представлень окремих лексичних значень слів для побудови синсетів та зв’язків;
– застосування краудсорсингу для поповнення та розширення початкових даних;
– узагальнення запропонованих методів на інші класи семантичних зв’язків;
– формування зв’язків між окремими поняттями на основі семантичної мережі слів;
– розробка інтегрального показника якості семантичних мереж.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Abadi M. et al.* TensorFlow: A System for Large-Scale Machine Learning // 12th USENIX Symposium on Operating Systems Design and Implementation (OS- DI 16), November 2–4, 2016, Savannah, GA, USA. Berkeley, CA, USA: USENIX Association, 2016. P. 265–283.
2. *Allan K.* Concise Encyclopedia of Semantics. Oxford, UK: Elsevier Science, 2009. 1104 pp.
3. *van Assem M., Malaisé V., Miles A., Schreiber G.* A Method to Convert Thesauri to SKOS // 3rd European Semantic Web Conference, ESWC 2006, Budva, Montenegro, June 11–14, 2006 Proceedings. Berlin, Heidelberg, Germany: Springer-Verlag Berlin Heidelberg, 2006. P. 95–109.
4. *Bagga A., Baldwin B.* Algorithms for Scoring Coreference Chains // Proceedings of the Linguistic Coreference Workshop at The First International Conference on Language Resources and Evaluation (LREC), May 26, 1998, Granada, Spain. 1998. P. 563–566.

5. *Bartunov S., Kondrashkin D., Osokin A., Vetrov D. P.* Breaking Sticks and Ambiguities with Adaptive Skip-gram // *Journal of Machine Learning Research*. 2016. Vol. 51. P. 130–138.

6. Рогушина Ю.В., Гладун А.Я Використання онтології як засобу інтеграції знань про інформаційну систему // Відбір і обробка інформації. – 2006, Львів: ФМІ ім. Г.В. Карпенка. – № 5, Вип. 24 (100). – С. 43-49.

7. *Berners-Lee T., Hendler J., Lassila O.* The Semantic Web // *Scientific American*. 2001. Vol. 284, no. 5. P. 28–37.

8. *Biemann C.* Ontology Learning from Text: A Survey of Methods // *GLDV-Journal for Computational Linguistics and Language Technology*. 2005. Vol. 20, no. 2. P. 75–93.

9. *Biemann C.* Chinese Whispers: An Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems // Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing (TextGraphs-1), June 9, 2006, New York, NY, USA. Stroudsburg, PA, USA: As- sociation for Computational Linguistics, 2006. P. 73–80.

10. *Biemann C.* Creating a system for lexical substitutions from scratch using crowdsourcing // *Language Resources and Evaluation*. 2013. Vol. 47, no. 1. P. 97–122.

11. *Bomze I. M., Budinich M., Pardalos P. M., Pelillo M.* The maximum clique problem // Handbook of Combinatorial Optimization. Springer, 1999. P. 1–74.

12. *Bordea G., Lefever E., Buitelaar P.* SemEval-2016 Task 13: Taxonomy Extrac- tion Evaluation (TExEval-2) // Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), June 16–17, 2016, San Diego, CA, USA. Stroudsburg, PA, USA: Association for Computational Linguistics, 2016.

P. 1081–1091.

13. *Braslavski P., Ustalov D., Mukhin M., Kiselev Y.* YARN: Spinning-in-Progress // Proceedings of the 8th Global WordNet Conference (GWC2016), January 27–30, 2016, Bucharest, Romania. Global WordNet Association, 2016. P. 58–65.

14. *Collins A. M., Quillian M. R.* Retrieval time from semantic memory // *Journal of Verbal Learning and Verbal Behavior*. 1969. Vol. 8, no. 2. P. 240–247.

15. *Deliyanni A., Kowalski R. A.* Logic and Semantic Networks // *Communications of the ACM*. 1979. Vol. 22, no. 3. P. 184–192.

16. *Deng J., Dong W., Socher R. et al.* ImageNet: A Large-Scale Hierarchical Image Database // IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009), June 20–25, 2009, Miami, FL, USA. IEEE, 2009. P. 248–255.

17. *Dorow B., Widdows D.* Discovering Corpus-Specific Word Senses // 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2003), April 12–17, 2003, Budapest, Hungary. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003. P. 79–82.

18. *Faralli S., Panchenko A., Biemann C., Ponzetto S. P.* Linked Disambiguated Distributional Semantic Networks // The Semantic Web – ISWC 2016: 15th International Semantic Web Conference, Kobe, Japan, October 17–21, 2016, Proceedings, Part II. Cham, Germany: Springer International Publishing, 2016. P. 56–64.

19. *Fellbaum C.* WordNet: An Electronic Database. MIT Press, 1998. 449 pp.

20. *Fowlkes E. B., Mallows C. L.* A Method for Comparing Two Hierarchical Clusterings // *Journal of the American Statistical Association*. 1983. Vol. 78, no. 383. P. 553–569.

21. *Freeman L. C.* Centered graphs and the structure of ego networks // *Mathematical Social Sciences*. 1982. Vol. 3, no. 3. P. 291–304.

22. *Frome A., Corrado G. S., Shlens J. et al.* DeViSE: A Deep Visual-Semantic Embedding Model // Advances in Neural Information Processing Systems 26 (NIPS 2013), December 5–10, 2013, Harrah and Harveys, NV, USA. Curran As- sociates, Inc., 2013. P. 2121–2129.

23. *Fu R., Guo J., Qin B. et al.* Learning Semantic Hierarchies via Word Embeddings // Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014) (Volume 1: Long Papers), June 22–27, 2014, Baltimore, MD, USA. Stroudsburg, PA, USA: Association for Computational Linguistics, 2014. P. 1199–1209.

24. *Gábor K., Zargayouna H., Tellier I. et al.* Exploring Vector Spaces for Semantic Relations // Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017), September 9–11, 2017, Copenhagen, Den- mark. Stroudsburg, PA, USA: Association for Computational Linguistics, 2017. P. 1815–1824.

25. *Gfeller D., Chappelier J.-C., De Los Rios P.* Synonym Dictionary Improve- ment through Markov Clustering and Clustering Stability // Proceedings of the International Symposium on Applied Stochastic Models and Data Analysis (AS- MDA'05), May 17–20, 2005, Brest, France. 2005. P. 106–113.

26. *Gonçalo Oliveira H., Gomes P.* Onto.PT: Automatic Construction of a Lexical Ontology for Portuguese // Proceedings of the 2010 Conference on STAIRS 2010:

Proceedings of the Fifth Starting AI Researchers' Symposium, August 16–20, 2010, Lisbon, Portugal. Amsterdam, The Netherlands: IOS Press, 2010. P. 199– 211.

27. *Gonçalo Oliveira H., Gomes P.* ECO and Onto.PT: a flexible approach for creating a Portuguese wordnet automatically // *Language Resources and Evaluation*. 2014. Vol. 48, no. 2. P. 373–393.

28. *Hagberg A. A., Schult D. A., Swart P. J.* Exploring Network Structure, Dynamics, and Function using NetworkX // Proceedings of the 7th Python in Science Conference (SciPy2008), August 19–24, 2008, Pasadena, CA, USA. 2008. P. 11–15.

29. *Hartigan J. A., Wong M. A.* Algorithm AS 136: A K-Means Clustering Algo- rithm // *Journal of the Royal Statistical Society. Series C (Applied Statistics)*. 1979. Vol. 28, no. 1. P. 100–108.

30. *Hearst M. A.* Automatic Acquisition of Hyponyms from Large Text Corpora // Proceedings of the 14th Conference on Computational Linguistics (COLING '92) - Volume 2, August 23–28, 1992, Nantes, France. COLING '92. International Committee on Computational Linguistics, 1992. P. 539–545.

31. *Herrmann D. J.* An old problem for the new psychosemantics: Synonymity // *Psychological Bulletin*. 1978. Vol. 85, no. 3. P. 490–512.

32. *Hope D., Keller B.* MaxMax: A Graph-Based Soft Clustering Algorithm Applied to Word Sense Induction // Computational Linguistics and Intelligent Text Processing: 14th International Conference, CICLing 2013, Samos, Greece, March 24–30, 2013, Proceedings, Part I. Berlin, Heidelberg, Germany: Springer-Verlag Berlin Heidelberg, 2013. P. 368–381.

33. *Hutchins J.* ALPAC: The (In)Famous Report // *Readings in machine translation*. 2003. Vol. 14. P. 131–135.

34. *Jurgens D., Klapaftis I.* SemEval-2013 Task 13: Word Sense Induction for Graded and Non-Graded Senses // Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Work- shop on Semantic Evaluation (SemEval 2013), June 14–15, 2013, Atlanta, GA, USA. Stroudsburg, PA, USA: Association for Computational Linguistics, 2013. P. 290–299.

35. *Kamps J., Marx M., Mokken R. J., de Rijke M.* Using WordNet to Measure Semantic Orientations of Adjectives // Fourth International Conference on Language Resources and Evaluation (LREC 2004), May 26–28, 2004, Lisbon, Portugal. European Language Resources Association (ELRA), 2004. P. 1115–1118.

36. *Kittur A., Chi E. H., Suh B.* Crowdsourcing User Studies with Mechanical Turk // Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI ‘08), April 5–10, 2008, Florence, Italy. New York, NY, USA: ACM, 2008. P. 453–456.

37. *Lassila O., McGuinness D.* The Role of Frame-Based Representation on the Semantic Web // *Linköping Electronic Articles in Computer and Information Science*. 2001. Vol. 6, no. 005.

38. *Lenat D. B., Guha R. V.* Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project. 1st edition. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990. 391 pp.

39. *Lohmann S., Negru S., Haag F., Ertl T.* Visualizing Ontologies with VOWL // *Semantic Web*. 2016. Vol. 7, no. 4. P. 399–419.

40. *Manandhar S., Klapaftis I., Dligach D., Pradhan S.* SemEval-2010 Task 14: Word

Sense Induction & Disambiguation // Proceedings of the 5th International Workshop on Semantic Evaluation (SemEval-2010), July 15–16, 2010, Upp- sala, Sweden. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010. P. 63–68.

41. *McCrae J., Spohr D., Cimiano P.* Linking Lexical Resources and Ontologies on the Semantic Web with Lemon // The Semantic Web: Research and Applica- tions: 8th Extended Semantic Web Conference, ESWC 2011, Heraklion, Crete, Greece, May 29 – June 2, 2011, Proceedings, Part I. Berlin, Heidelberg, Germany: Springer Berlin Heidelberg, 2011. P. 245–259.

42. *Mikolov T., Sutskever I., Chen K. et al.* Distributed Representations of Words and Phrases and their Compositionality // Advances in Neural Information Processing Systems 26 (NIPS 2013), December 5–10, 2013, Harrah and Harveys, NV, USA. Curran Associates, Inc., 2013. P. 3111–3119.

43. *Navigli R., Ponzetto S. P.* BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network // *Artificial Intelligence*. 2012. Vol. 193. P. 217–250.

44. *Niles I., Pease A.* Towards a Standard Upper Ontology // Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS ‘01) - Volume 2001, October 17–19, 2001, Ogunquit, ME, USA. New York, NY, USA: ACM, 2001. P. 2–9.

45. *Panchenko A., Faralli S., Ruppert E. et al.* TAXI at SemEval-2016 Task 13: a Taxonomy Induction Method based on Lexico-Syntactic Patterns, Substrings and Focused Crawling // Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), June 16–17, 2016, San Diego, CA, USA. Strouds- burg, PA, USA: Association for Computational Linguistics, 2016. P. 1320–1327.

46. *Panchenko A., Simon J., Riedl M., Biemann C.* Noun Sense Induction and Disambiguation using Graph-Based Distributional Semantics // Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016), September 19–21, 2016, Bochum, Germany. Bochum, Germany: Bochumer Linguistische Arbeitsberichte, 2016. P. 192–202.

47. *Parr T.* The Definitive ANTLR 4 Reference. The Pragmatic Programmers, LLC, 2013. 328 pp.

48. *Pembeci İ.* Using Word Embeddings for Ontology Enrichment // *International Journal of Intelligent Systems and Applications in Engineering*. 2016. Vol. 4, no. 6. P. 49–56.

49. *Pennacchiotti M., Pantel P.* Ontologizing Semantic Relations // Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL 2006), July 17–21, 2006, Sydney, Australia. Stroudsburg, PA, USA: Association for Computational Linguistics, 2006. P. 793–800.

50. *Pu X., Pappas N., Popescu-Belis A.* Sense-Aware Statistical Machine Translation using Adaptive Context-Dependent Clustering // Proceedings of the Second Conference on Machine Translation (WMT 17), September 7–8, 2017, Copenhagen, Denmark. Stroudsburg, PA, USA: Association for Computational Linguistics, 2017. P. 1–10.

51. *Quillian M. R.* Word concepts: A theory and simulation of some basic semantic capabilities // *Behavioral Science*. 1967. Vol. 12, no. 5. P. 410–430.

52. *Řehuřek R., Sojka P.* Software Framework for Topic Modelling with Large Corpora

// New Challenges for NLP Frameworks Programme: A workshop at LREC 2010, May 22, 2010, Valetta, Malta. European Language Resources As- sociation (ELRA), 2010. P. 51–55.

53. *Roussopoulos N., Mylopoulos J.* Using Semantic Networks for Data Base Management // Proceedings of the 1st International Conference on Very Large Data Bases (VLDB ‘75), September 22–24, 1975, Framingham, MA, USA. New York, NY, USA: ACM, 1975. P. 144–172.

54. *Shapiro S. C.* Encyclopedia of Artificial Intelligence. 2nd edition. New York, NY, USA: John Wiley & Sons, Inc., 1992. 1724 pp.

55. *Shwartz V., Goldberg Y., Dagan I.* Improving Hypernymy Detection with an Integrated Path-based and Distributional Method // Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), August 7–12, 2016, Berlin, Germany. Stroudsburg, PA, USA: Association for Computational Linguistics, 2016. P. 2389–2398.

56. *Tarjan R.* Depth-First Search and Linear Graph Algorithms // *SIAM Journal on Computing*. 1972. Vol. 1, no. 2. P. 146–160.

57. *Wang M., Wang C., Yu J. X., Zhang J.* Community Detection in Social Networks: An In-depth Benchmarking Study with a Procedure-oriented Framework // *Proceedings of the VLDB Endowment*. 2015. Vol. 8, no. 10. P. 998–1009.

58. *Welch B. L.* The generalization of ‘Student’s’ problem when several different population variances are involved // *Biometrika*. 1947. Vol. 34, no. 1-2. P. 28–35.

59. *Wilcoxon F.* Individual Comparisons by Ranking Methods // *Biometrics Bulletin*. 1945. Vol. 1, no. 6. P. 80–83.

60. *Zeng X.-M.* Semantic Relationships between Contextual Synonyms // *US-China*

Education Review. 2007. Vol. 4, no. 9. P. 33–37.

61. Шибицька Н.М. Новітні мультимедійні технологій в освіті / Н.М.Шибицька // Збірник тез II науково-практичної конференції «Наукові аспекти геодезії, землеустрою та інформаційних технологій». - К.: Університет «Україна», 2013. – С. 166-171.
62. Шибицька Н.М. Логіко-семантична модель структурування змісту навчання //Modern International Relations: Current Problems Of Theory And Practic /Collective monograph.- Łodz: Naukowe Wyższej Szkoły Biznesu i Nauk o Zdrowiu, 2021. – p.353-360.

ДОДАТОК А. Текст програми

Формування кластерів:

```
#bash
#!/bin/bash

set -o pipefail
export LANG=en_US.UTF-8 LC_COLLATE=C

SCRIPT_DIRECTORY=$(dirname "$(readlink -f "$0")")

GOLD_FILE=$1
shift

if [ -z "${GOLD_FILE}" ]; then
  echo "Usage: $0 gold-synsets.tsv [resource-synsets.tsv ...]"
  exit 1
fi

LEXICON_TEMP=$(mktemp)
```

```
GOLD_DATA_TEMP=$(mktemp)
RESOURCE_DATA_TEMP=$(mktemp)

cleanup() {
  rm -f -- "$LEXICON_TEMP" "$GOLD_DATA_TEMP" "$RESOURCE_DATA_TEMP"
}

trap 'cleanup; exit 1' INT TERM HUP EXIT

create_lexicon_data() {
  local input_file=$1
  local output_file=$2
  $SCRIPT_DIRECTORY/../lexicon.awk -v TO_LOWER=1 "$input_file" > "$output_file"
}

create_lexicon_data "$GOLD_FILE" "$GOLD_DATA_TEMP"

for RESOURCE_FILE in "$@"; do
  create_lexicon_data "$RESOURCE_FILE" "$RESOURCE_DATA_TEMP"
done

sort --parallel=$(nproc) -S1G -uo "$RESOURCE_DATA_TEMP" "$RESOURCE_DATA_TEMP"

LEXICON_SIZE=$(comm -12 "$GOLD_DATA_TEMP" "$RESOURCE_DATA_TEMP" | tee "$LEXICON_TEMP" |
wc -l)

generate_cnl_data() {
  local input_lexicon=$1
  local input_gold=$2
  local output_data=$3
  $SCRIPT_DIRECTORY/cnl.py "$input_lexicon" < "$input_gold" > "$output_data"
}
```

```
generate_cnl_data "$LEXICON_TEMP" "$GOLD_FILE" "$GOLD_DATA_TEMP"


echo -e "path\twords\tsynsets\tlexicon\tgenconv_nmi\tovp_nmi"


for RESOURCE_FILE in "$@"; do
  generate_cnl_data "$LEXICON_TEMP" "$RESOURCE_FILE" "$RESOURCE_DATA_TEMP"


  WORDS=$($SCRIPT_DIRECTORY/../lexicon.awk "$RESOURCE_FILE" | wc -l)
  SYNSETS=$(wc -l < "$RESOURCE_FILE")


  if [ -z "${NONMI+x}" ]; then
    GENCONVNMI=$($SCRIPT_DIRECTORY/../../GenConvNMI/bin/Release/gecmi "$GOLD_DATA_TEMP"
"$RESOURCE_DATA_TEMP" || true)
    OVPNMI=$($SCRIPT_DIRECTORY/../../OvpNMI/bin/Release/onmi "$GOLD_DATA_TEMP"
"$RESOURCE_DATA_TEMP" || true)
  else
    GENCONVNMI="e"
    OVPNMI="e"
  fi


  echo -e "$RESOURCE_FILE\t$WORDS\t$SYNSETS\t$LEXICON_SIZE\t$GENCONVNMI\t$OVPNMI"
done


cleanup
```

Пошук подібних слів:

```
#! python3


import argparse
import csv
import sys
```

```
from signal import signal, SIGINT
from gensim.models import KeyedVectors


def exit_gracefully(signum, frame):
    sys.exit(1)


signal(SIGINT, exit_gracefully)


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('--sim', type=float, default=.3)
    parser.add_argument('w2v', type=argparse.FileType('rb'))
    args = parser.parse_args()


    w2v = KeyedVectors.load_word2vec_format(args.w2v, binary=True,
unicode_errors='ignore')
    w2v.init_sims(replace=True)
    print(f'Using {w2v.vector_size} word2vec dimensions from "{args.w2v.name}".',
file=sys.stderr)


    reader = csv.reader(sys.stdin, delimiter='\t', quoting=csv.QUOTE_NONE)


    for row in reader:
        word1, word2 = row[0], row[1]


        try:
            similarity = w2v.similarity(word1, word2)


            if similarity < 0:
                similarity = args.sim
        except KeyError:
            similarity = args.sim
```

```
        print(f'{word1}\t{word2}\t{similarity}')

if __name__ == "__main__":
    main()
```

Побудова зв’язків:

```
import argparse
import concurrent.futures
import itertools
import sys
from collections import defaultdict, Counter
from signal import signal, SIGINT

from sklearn.feature_extraction import DictVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.metrics.pairwise import cosine_similarity as calculate_similarity
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import Binarizer

signal(SIGINT, lambda signum, frame: sys.exit(1))

WEIGHT_OPTIONS = {
    'tf': TfidfTransformer(use_idf=False),
    'idf': Pipeline([('binary', Binarizer()), ('idf', TfidfTransformer())]),
    'tfidf': TfidfTransformer()
}

argument_parser = argparse.ArgumentParser()
argument_parser.add_argument('--synonyms', required=True, type=argparse.FileType('r'))
argument_parser.add_argument('--isas', required=True, type=argparse.FileType('r'))
```

```
argument_parser.add_argument('--weight', choices=WEIGHT_OPTIONS.keys(),
default='tfidf')

argument_parser.add_argument('-k', nargs='?', type=int, default=6)

args = argument_parser.parse_args()


synonyms, index, lexicon = {}, defaultdict(list), set()


for line in args.synonyms:

    line = line.rstrip()


    if not line:

        continue


    row = line.split('\t')


    synonyms[row[0]] = [w for w in row[2].split(', ') if w]


    for w in synonyms[row[0]]:

        index[w].append(row[0])


    lexicon.update(synonyms[row[0]])


index = {w: {id: i + 1 for i, id in enumerate(ids)} for w, ids in index.items()}


isas = defaultdict(set)


for line in args.isas:

    line = line.rstrip()


    if not line:

        continue
```

```
    row = line.split('\t')

    if len(row) > 1 and row[0] in lexicon and row[1] in lexicon:
        isas[row[0]].add(row[1])

hypernyms_context = {}

for identifier, words in synonyms.items():
    hypernyms_counter = Counter(itertools.chain(*(isas[w] for w in words if w in
isas)))

    if hypernyms_counter:
        hypernyms_context[identifier] = hypernyms_counter

vectorizer = Pipeline([('dict', DictVectorizer()), (args.weight,
WEIGHT_OPTIONS[args.weight])]).fit(hypernyms_context.values())

def process_entry(identifier):
    if identifier not in hypernyms_context:
        return (identifier, {})

    hypernyms_vector, candidates = vectorizer.transform(hypernyms_context[identifier]),
Counter()

    for hypernym in hypernyms_context[identifier]:
        hypernym_senses = Counter({hid:
calculate_similarity(vectorizer.transform(Counter(synonyms[hid])),
hypernyms_vector).item(0) for hid in index[hypernym]})

        for hid, cosine in hypernym_senses.most_common(1):
            if cosine > 0:
                candidates[(hypernym, hid)] = cosine

    matches = [(hypernym, hid, cosine) for (hypernym, hid), cosine in
```

```
            candidates.most_common(len(candidates) if args
```

Обхід графа (пошук оптимальних шляхів):

```
import argparse
import csv
import itertools
import warnings
from collections import defaultdict
from concurrent.futures import ProcessPoolExecutor
from scipy.stats import wilcoxon
from sklearn.exceptions import UndefinedMetricWarning
from sklearn.metrics import confusion_matrix, precision_recall_fscore_support
import networkx as nx

warnings.simplefilter('ignore', category=UndefinedMetricWarning)

parser = argparse.ArgumentParser()
parser.add_argument('--gold', required=True)
parser.add_argument('--significance', action='store_true')
parser.add_argument('--alpha', nargs='?', type=float, default=0.01)
parser.add_argument('file_paths', nargs='+')
args = parser.parse_args()

def sanitize_word(word):
    return word.lower().replace(' ', '_')

def load_graph(file_path):
    graph = nx.DiGraph()

    with open(file_path, newline='') as file:
        reader = csv.reader(file, delimiter='\t')
```

```
        for row in reader:
            if len(row) > 1 and row[0] and row[1]:
                graph.add_edge(sanitize_word(row[0]), sanitize_word(row[1]))

    graph.senses = defaultdict(list)

    for node in graph.nodes():
        graph.senses[node.rsplit('#', 1)[0]].append(node)

    return graph

with ProcessPoolExecutor() as executor:
    file_paths = args.file_paths + [args.gold]
    resources = {file_path: graph for file_path, graph in zip(file_paths,
executor.map(load_graph, file_paths))}

gold_graph = resources.pop(args.gold)

def has_path_between_nodes(graph, source, target):
    if source not in graph.senses or target not in graph.senses:
        return False

    for source_node, target_node in itertools.product(graph.senses[source],
graph.senses[target]):
        if nx.has_path(graph, source_node, target_node):
            return True

    return False

lexicon = gold_graph.senses.keys() & set.union(*(set(G.senses.keys()) for G in
resources.values()))

edges = [pair for pair in
```

```
         {(word1.rsplit('#', 1)[0], word2.rsplit('#', 1)[0]) for word1, word2 in
gold_graph.edges()} |

         set.union(*(set(G.edges()) for G in resources.values()))

         if pair[0] in lexicon and pair[1] in lexicon]


true_values = [int(has_path_between_nodes(gold_graph, *pair)) for pair in edges]


index = defaultdict(list)


for pair in edges:

    index[pair[0]].append(pair)


sorted_hyponyms = sorted(index)


def evaluate_node(graph, pairs):

    true_values_for_node = [int(has_path_between_nodes(gold_graph, *pair)) for pair in
pairs]

    pred_values_for_node = [int(has_path_between_nodes(graph, *pair)) for pair in
pairs]


    return (true_values_for_node, pred_values_for_node)


def calculate_scores_for_graph(graph):

    if not args.significance:

        return


    labels = [evaluate_node(graph, index[word]) for word in sorted_hyponyms]


    scores = {score: [None] * len(labels) for score in ('precision', 'recall', 'f1')}


    for i, (true_values_for_node, pred_values_for_node) in enumerate(labels):

        precision, recall, f1, _ =
precision_recall_fscore_support(true_values_for_node, pred_values_for_node,
average='binary')
```

```
        scores['precision'][i] = float(precision)
        scores['recall'][i] = float(recall)
        scores['f1'][i] = float(f1)

    return scores

def evaluate_resource(file_path):
    graph = resources[file_path]

    pred_values = [int(has_path_between_nodes(graph, *pair)) for pair in edges]

    tn, fp, fn, tp = confusion_matrix(true_values, pred_values).ravel()

    precision, recall, f1, _ = precision_recall_fscore_support(true_values,
pred_values, average='binary')

    return {
        'tn': tn.item(),
        'fp': fp.item(),
        'fn': fn.item(),
        'tp': tp.item(),
        'precision': precision.item(),
        'recall': recall.item(),
        'f1': f1.item(),
        'scores': calculate_scores_for_graph(graph)
    }

with ProcessPoolExecutor() as executor:
    results = {file_path: result for file_path, result in zip(resources,
executor.map(evaluate_resource, resources))}

def pairwise(iterable):
```

```
    a, b = itertools.tee(iterable)
    next(b, None)
    return zip(a, b)


def calculate_significance(metric):
    if not args.significance:
        return {}


    desc, rank = sorted(results.items(), key=lambda item: item[1][metric],
reverse=True), 1


    ranks = {}


    for (file_path1, results1), (file_path2, results2) in pairwise(desc):
        x, y = list(results1['scores'][metric]), list(results2['scores'][metric])


        ranks[file_path1] = rank


        rank += int(wilcoxon(x, y).pvalue < args.alpha)


        ranks[file_path2] = rank


    return metric, ranks


with ProcessPoolExecutor() as executor:
    ranks = {metric: result for metric, result in executor.map(calculate_significance,
('precision', 'recall', 'f1'))}
print('\t'.join(
    ('file_path', 'edges', 'tn', 'fp', 'fn', 'tp', 'precision', 'recall', 'f1',
'precision_rank', 'recall_rank', 'f1_rank')))


for file_path, values in results.items():
    print('\t'.join((
```

```
file_path,
str(resources[file_path].size()),
str(values['tn']),
str(values['fp']),
str(values['fn']),
str(values['tp']),
str(values['precision']),
str(values['recall']),
str(values['f1']),
str(ranks['precision'].get(file_path, 0)),
str(ranks['recall'].get(file_path, 0)),
str(ranks['f1'].get(file_path, 0)) ))
```