МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Кваліфікаційна наукова
праця на правах рукопису

М’ЯКЕНЬКИЙ АРСЕНІЙ ВЯЧЕСЛАВОВИЧ

УДК 004.8:004.912

ДИСЕРТАЦІЯ
МЕТОД ІДЕНТИФІКАЦІЇ ПЕРИФРАЗ НА ОСНОВІ ГРАФОВИХ РЕПРЕЗЕНТАЦІЙ ДЛЯ МОДЕЛЮВАННЯ КОГНІТИВНОГО ПРОЦЕСУ РОЗУМІННЯ

Спеціальність: 122 – Комп’ютерні науки
Галузь знань: 12 – Інформаційні технології

Подається на здобуття ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело _____ А.В. М’якенький

Науковий керівник: Алексєєв Михайло Олександрович,
доктор технічних наук, професор

Дніпро – 2026

АНОТАЦІЯ

М'якенький А.В. Метод ідентифікації перифраз на основі графових репрезентацій для моделювання когнітивного процесу розуміння – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття ступеня доктора філософії за спеціальністю 122 «Комп'ютерні науки» (12 – Інформаційні технології). – Національний технічний університет «Дніпровська політехніка» Міністерства освіти і науки України, Дніпро, 2026.

В дисертаційній роботі розв'язана актуальна науково-прикладна задача моделювання когнітивного процесу розуміння природної мови шляхом розробки інтелектуальної системи ідентифікації перифразів з використанням семантичних графових репрезентацій, машинного навчання та трансформерних архітектур.

Мета дослідження полягає в підвищенні точності ідентифікації перифраз у текстах української мови в контексті моделювання когнітивного процесу розуміння шляхом розробки інтелектуальної системи, яка використовує семантичні графові репрезентації для виявлення семантичної еквівалентності між висловлюваннями на концептуальному рівні.

Об'єкт дослідження – процеси обробки та інтерпретації природної мови для визначення семантичної еквівалентності в інтелектуальних системах.

Предмет дослідження – методи та моделі створення інтелектуальних систем ідентифікації перифраз, що забезпечують відтворення й аналіз семантичної еквівалентності між висловлюваннями.

Наукова новизна отриманих результатів:

– Запропоновано когнітивно орієнтовану методику до ідентифікації перифраз у текстах українською мовою, що ґрунтується на використанні семантичних графових репрезентацій та дозволяє здійснювати аналіз семантичної еквівалентності висловлювань на концептуальному рівні в межах моделювання когнітивного процесу розуміння.

– Вперше розроблено когнітивно обґрунтовану модель перифраза як концептуально-еквівалентного смислового варіанту в межах семантичного простору абстрактних репрезентацій змісту, що дозволяє уніфікувати підхід до ідентифікації перифраз незалежно від їхньої лексичної та синтаксичної форми.

– Вперше розроблено інформаційну технологію формування AMR графів для українськомовних текстів на основі їхнього попереднього машинного перекладу. Така технологія гарантує збереження семантичної повноти текстів українською мовою при перекладі на рівні 94% за відсутності повноцінних українськомовних AMR корпусів.

– Удосконалено метод репрезентації текстів природної мови за допомогою AMR шляхом використання графових нейронних мереж з механізмом уваги, які дозволяють формувати векторні представлення, що відображають як локальні, так і глобальні властивості смислової структури висловлювання, що забезпечує підвищення точності у задачі оцінювання семантичної подібності текстів на 4% порівняно з наявними AMR орієнтованими підходами.

– Удосконалено метод ідентифікації перифраз шляхом застосування класифікації на основі векторних представлень AMR графів, що забезпечує прийняття рішень про семантичну еквівалентність висловлювань на рівні концептуально-смислових структур та дозволяє досягати показників точності та F1-міри на 11% та на 4,2% відповідно вищими за наявні підходи, що використовують AMR графи для ідентифікації перифраз.

Практичне значення отриманих результатів дослідження полягає у застосуванні розробленої інформаційної технології для автоматизованої ідентифікації перифраз у текстах українською мовою. Розроблені графові репрезентації та методи дозволяють підвищувати точність визначення семантичної еквівалентності між висловлюваннями.

Практична реалізація цієї технології може бути використана у системах штучного інтелекту для автоматичного розпізнавання перефразувань,

інформаційного пошуку, реферування текстів, навчальних і наукових застосунках, а також у розвитку когнітивно орієнтованих лінгвістичних систем.

Дисертація складається зі вступу, чотирьох розділів, висновків, списку використаних джерел та додатків. Повний обсяг дисертації – 196 сторінки; список використаних джерел зі 107 найменувань, 3 додатків. Робота проілюстрована 28 рисунками та містить 21 таблицю.

У вступі зазначено актуальність теми досліджень, сформульовано мету, об’єкт, предмет і основні завдання досліджень, обґрунтовано методи досліджень, викладено наукову новизну й визначено практичне значення одержаних результатів, зазначено особистий внесок здобувача, представлено загальну характеристику та структуру дисертації, а також наведено відомості щодо публікацій і результатів апробації й упровадження дисертаційної роботи.

У першому розділі дисертації проведено інформаційно-аналітичний огляд теоретичних засад когнітивного пізнання та сучасного стану досліджень у галузі моделювання когнітивних процесів у штучних інтелектуальних системах. Розглянуто когнітивний процес розуміння як ключовий механізм обробки природної мови та обґрунтовано його зв’язок із задачею ідентифікації перифраз. Проаналізовано традиційні методи ідентифікації перифраз на основі корпусів і знань, а також підходи, що базуються на глибокому навчанні та дистрибутивній семантиці, з визначенням їхніх переваг і обмежень. Окрему увагу приділено підходам до побудови семантичних репрезентацій, зокрема графовим моделям типу Abstract Meaning Representation. На основі аналізу обґрунтовано напрям подальших досліджень, спрямованих на розроблення інтерпретованої інтелектуальної системи ідентифікації перифраз, адаптованої до особливостей української мови.

У другому розділі сформовано теоретико-методологічну основу побудови інтелектуальної системи ідентифікації перифраз на основі графових семантичних репрезентацій. Процес розуміння розглянуто як механізм зіставлення смислових структур, що дало змогу формалізувати задачу ідентифікації перифраз як задачу визначення семантичної еквівалентності між

висловленнями. Обґрунтовано використання графових семантичних репрезентацій, зокрема формалізму Abstract Meaning Representation, для відокремлення концептуального змісту від поверхневої мовної форми. Розроблено метод багатомовної генерації абстрактних семантичних репрезентацій текстів природної мови з використанням нейронного машинного перекладу, орієнтований на обробку українськомовних текстів. Також розроблено метод визначення семантичної еквівалентності AMR графів на основі графових нейронних мереж із механізмами уваги та метод класифікації перифраз із використанням багатошарового перцептрона. Отримані результати формують методологічну основу інформаційної технології ідентифікації перифраз і створюють підґрунтя для розроблення архітектури інтелектуальної системи та її експериментальної перевірки.

У третьому розділі розроблено інформаційну технологію ідентифікації перифраз на основі графових репрезентацій. Побудовано архітектуру інтелектуальної системи у вигляді багаторівневого конвеєра, що інтегрує модулі підготовки текстових даних, побудови абстрактних семантичних репрезентацій у форматі AMR за допомогою нейронної моделі AMRBART, графового кодування семантичних структур із використанням графових нейронних мереж та модуль класифікації перифраз. Визначено структуру, функції та логіку взаємодії компонентів системи, що забезпечує декомпозицію когнітивного процесу розуміння на формалізовані підпроцеси. Розроблено алгоритм функціонування інтелектуальної системи та описано особливості її програмної реалізації, включно з вибором інструментальних засобів і методів оптимізації навчання нейронних моделей. Отримані результати формують завершену інформаційну технологію автоматичної ідентифікації перифраз на основі глибинних семантичних репрезентацій і створюють основу для подальшої експериментальної перевірки.

Четвертий розділ дисертації присвячено експериментальному дослідженню ефективності розробленої інформаційної технології ідентифікації перифраз на основі графових репрезентацій. Описано методику проведення

експериментів та сформовано набори даних для оцінювання якості підготовки текстів, побудови AMR графів, отримання їх векторизованих представлень та класифікації перифраз. Проведено експериментальну перевірку здатності нейронних моделей машинного перекладу зберігати семантичну повноту українськомовних текстів при підготовці до AMR парсингу, а також оцінено якість побудови абстрактних семантичних репрезентацій із використанням моделі AMRBART. Досліджено вплив механізму уваги у графових нейронних мережах на формування векторних представлень AMR графів та встановлено його переваги порівняно з базовими архітектурами. Експериментально підтверджено ефективність класифікації перифраз на основі векторних репрезентацій AMR графів і продемонстровано конкурентоспроможність запропонованої технології відносно сучасних трансформерних моделей за умов суттєво меншої обчислювальної складності.

Ключові слова: інтелектуальна система, обробка природної мови, ідентифікація перифраз, семантична еквівалентність, абстрактне представлення змісту, машинне навчання, багатошаровий перцептрон, графова нейронна мережа, трансформери, графові репрезентації, векторні представлення.

ABSTRACT

Miakenkyi A.V. Paraphrase identification method based on graph representations for modeling the cognitive process of comprehension – Qualification scientific work on manuscript rights.

Dissertation for the degree of Doctor of Philosophy in specialty 122 "Computer Science" (12 – Information Technologies). – National Technical University "Dnipro Polytechnic" of the Ministry of Education and Science of Ukraine, Dnipro, 2026.

The dissertation addresses a relevant scientific and applied problem of modeling the cognitive process of natural language comprehension through the development of an intelligent paraphrase identification system based on semantic graph representations, machine learning methods, and transformer architectures.

The aim of the research is to improve the accuracy of paraphrase identification in natural language texts within the framework of modeling the cognitive process of comprehension by developing an intelligent system that employs semantic graph representations to detect semantic equivalence between utterances at the conceptual level.

The object of the research is the processes of natural language processing and interpretation for determining semantic equivalence in intelligent systems.

The subject of the research comprises methods and models for constructing intelligent paraphrase identification systems that ensure the reconstruction and analysis of semantic equivalence between sentences.

Scientific novelty of the obtained results:

– For the first time, a cognitively oriented methodology for paraphrase identification in natural language texts has been proposed. The methodology is based on the use of semantic graph representations and enables the analysis of semantic equivalence between utterances at the conceptual level within the framework of modeling the cognitive process of comprehension.

– For the first time, a cognitively grounded model of paraphrase as a conceptually equivalent semantic variant within the semantic space of abstract meaning representations has been developed. This model unifies the approach to paraphrase identification independently of surface linguistic form.

– For the first time, an information technology for generating Abstract Meaning Representation (AMR) graphs for Ukrainian-language texts has been developed based on their preliminary machine translation. In the absence of comprehensive Ukrainian-language AMR corpora, this technology ensures the preservation of the semantic integrity of Ukrainian texts at a level of 94% during the translation process.

– The method for natural language text representation via AMR has been improved by utilizing Graph Neural Networks (GNN) with an attention mechanism. This approach enables the generation of embeddings that capture both local and global properties of the utterance's semantic structure. This enhancement yields a 4% increase in accuracy for semantic similarity tasks compared to existing AMR oriented approaches and achieves performance levels comparable to modern Transformer models while utilizing nearly 5 times fewer parameters.

– The paraphrase identification method has been improved through the application of classification based on vector representations of AMR graphs. This ensures decision-making regarding the semantic equivalence of utterances at the level of conceptual-semantic structures, and allows achieving accuracy and F1-score metrics that are 11% and 4.2% higher, respectively, compared to existing approaches that utilize AMR graphs for paraphrase identification.

The practical significance of the obtained research results lies in the application of the developed information technology for the automated identification of paraphrases in Ukrainian-language texts. The developed graph representations and methods allow for improving the accuracy of determining semantic equivalence between statements.

The practical implementation of this technology can be used in artificial intelligence systems for automatic paraphrase recognition, information retrieval, text summarization, educational and scientific applications, as well as in the development of cognitively oriented linguistic systems.

The dissertation consists of an introduction, four chapters, conclusions, a list of references, and appendices. The total length of the dissertation is 196 pages; the list of references includes 107 items, and there are 3 appendices. The work is illustrated with 28 figures and contains 21 tables.

The introduction substantiates the relevance of the research topic; formulates the aim, object, subject, and main research tasks; justifies the research methods; presents the scientific novelty and practical significance of the results; outlines the author's personal contribution; provides the general characteristics and structure of the dissertation; and includes information on publications, as well as the approbation and implementation of the research results.

The first chapter provides an analytical review of the theoretical foundations of cognitive cognition and the current state of research in modeling cognitive processes within artificial intelligent systems. The cognitive process of comprehension is considered as a key mechanism of natural language processing, and its connection with the task of paraphrase identification is substantiated. Traditional corpus-based and knowledge-based methods for paraphrase identification are analyzed, along with approaches based on deep learning and distributional semantics, identifying their advantages and limitations. Particular attention is paid to approaches for constructing semantic representations, especially graph-based models such as Abstract Meaning Representation (AMR). Based on the analysis, the direction for further research is substantiated, aimed at developing an interpretable intelligent paraphrase identification system adapted to the specific features of the Ukrainian language.

The second chapter develops the theoretical and methodological foundations for constructing an intelligent paraphrase identification system based on graph semantic representations. The process of comprehension is modeled as a mechanism for matching meaning structures, which makes it possible to formalize paraphrase

identification as the task of determining semantic equivalence between utterances. The use of graph semantic representations, particularly the Abstract Meaning Representation formalism, is substantiated as a means of separating conceptual content from surface linguistic form. A method for multilingual generation of abstract semantic representations of natural language texts using neural machine translation, oriented toward Ukrainian-language texts, is developed. In addition, a method for determining semantic equivalence of AMR graphs based on graph neural networks with attention mechanisms and a paraphrase classification method using a multilayer perceptron are proposed. The obtained results form the methodological basis of the information technology for paraphrase identification and provide the foundation for designing the system architecture and its experimental validation.

The third chapter develops an information technology for paraphrase identification based on graph representations. The architecture of the intelligent system is designed as a multi-level pipeline integrating modules for multilingual text data preparation, construction of abstract semantic representations in the AMR format using the AMRBART neural model, graph encoding of semantic structures using graph neural networks, and a paraphrase classification module. The structure, functions, and interaction logic of system components are defined, enabling the decomposition of the cognitive process of comprehension into formalized subprocesses. An operational algorithm of the intelligent system is developed, and features of its software implementation are described, including the selection of tools and methods for optimizing neural model training. The obtained results constitute a complete information technology for automatic paraphrase identification based on deep semantic representations and establish the basis for further experimental validation.

Chapter four focuses on an experimental evaluation of the performance of the proposed graph-based information technology for paraphrase identification. The experimental methodology is described, and datasets are constructed to evaluate the quality of multilingual text preparation, AMR graph construction, generation of their vector representations, and paraphrase classification. An experimental verification of the ability of neural machine translation models to preserve the semantic completeness

of Ukrainian-language texts during preparation for AMR parsing is conducted, and the quality of abstract semantic representation construction using a transformer-based model is evaluated. The impact of the attention mechanism in graph neural networks on the formation of vector representations of AMR graphs is investigated, demonstrating its advantages over baseline architectures. The effectiveness of paraphrase classification based on vector representations of AMR graphs is experimentally confirmed, and the competitiveness of the proposed technology relative to modern transformer-based models is demonstrated under conditions of significantly lower computational complexity.

Keywords: intelligent system, natural language processing, paraphrase identification, semantic equivalence, abstract meaning representation, machine learning, multilayer perceptron, graph neural network, transformers, graph representations, vector representations.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковано основні наукові результати дисертації:

Статті у наукових виданнях, включених до Переліку фахових видань, затвердженого МОН України:

1. М'якенький А. Аналіз методів розв'язання задачі ізоморфізму при моделюванні когнітивного процесу розуміння. *Information Technology: Computer Science, Software Engineering and Cyber Security*. 2024. № 1. С. 73–79. URL: https://doi.org/10.32782/it/2024-1-9

2. М'якенький А. В., Алексєєв М. О. Порівняльний аналіз методів семантичної репрезентації текстів природної мови у задачі ідентифікації перифраз. *Applied Questions of Mathematical Modeling*. 2025. Т. 8, № 2. С. 210–223. URL: https://doi.org/10.32782/mathematical-modelling/2025-8-2-22 (особистий внесок автора – порівняльний аналіз методів ідентифікації перифраз, обґрунтування використання AMR графів)

3. М'якенький А. В. Метод побудови AMR репрезентацій україномовних текстів із використанням моделей нейронного перекладу. *Таврійський науковий вісник. Серія: Технічні науки*. 2025. № 6. С. 114–122. URL: https://doi.org/10.32782/tnv-tech.2025.6.12

4. М'якенький А., Алексєєв О. Інформаційна технологія побудови векторних репрезентацій AMR графів з використанням графових нейронних мереж. *Information Technology: Computer Science, Software Engineering and Cyber Security*. 2026. №1. С. 183-191. URL: https://doi.org/10.32782/IT/2026-1-21 (особистий внесок автора – розробка архітектури на основі GAT, програмна реалізація та проведення експериментів)

Статті у міжнародних виданнях та виданнях України, включених до наукометричних баз даних:

5. Miakenkyi A. V., Aleksieiev M. O., Matsiuk S. M. Research on the effectiveness of using LSTM architecture in modeling the cognitive process of recognition. *Naukovyi Visnyk Natsionalnoho Hirnychoho Universytetu*. 2025. No. 1.

P. 90–95. URL: https://doi.org/10.33271/nvngu/2025-1/090 (особистий внесок автора – архітектура нейронної мережі Bi-LSTM, формування навчального датасету, аналіз результатів)

Наукові праці, які засвідчують апробацію матеріалів дисертації:

6. Алексєєв М. О., М'якенький А. В. Вибір методики навчання на підставі аналізу когнітивних моделей у системі дистанційної освіти. *Електротехнічні та інформаційні системи*. 2023. № 104. С. 3–8. URL: https://doi.org/10.32782/EIS/2023-104-1. (особистий внесок автора – розробка моделі оцінки когнітивних навичок, обґрунтування методу адаптивного добору навчальних методик)

7. М'якенький А. В. Дослідження методів моделювання когнітивних процесів. *Проблеми використання інформаційних технологій в освіті, науці та промисловості* : XVIII Міжнар. Конф., м. Дніпро, 24 листоп. 2023 р. Дніпро, 2025. С. 173–178.

8. М'якенький А. В., Алексєєв О. М. Двонапрямна LSTM модель усунення неоднозначності слів у тексті при моделюванні когнітивного процесу розуміння. *Проблеми використання інформаційних технологій в освіті, науці та промисловості* : XIX Міжнар. Конф., м. Дніпро, 27 листоп. 2024 р. Дніпро, 2025. С. 132–135.

9. М'якенький А. В. Універсальна модель репрезентації значення для мультилінгвістичного парсингу природної мови в задачі ідентифікації перифраз. *Integration of Education, Science and Business in Modern Environment: Summer Debates* : Proceedings of the 7th International Scientific and Practical Internet Conference Summer Debates. 2025. С. 171–173.

10. М'якенький А. В. Ідентифікація перифраз на основі вкладень AMR-графів за допомогою нейромережевої моделі. *International experience in scientific research* : Proceedings of IX International Scientific and Practical Conference, Chicago, 16–18 April 2026. Chicago, USA, 2026. P. 154–157.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

СУМ – словник української мови

AMR – Abstract Meaning Representation

AUC – Area Under Curve

API – Application Programming Interface

BART – Bidirectional and Auto-Regressive Transformers

CNN – Convolutional Neural Network

CUDA – Compute Unified Device Architecture

GAT – Graph Attention Network

GNN – Graph Neural Network

GPU – Graphics Processing Unit

IC – Information Content

LSC – Least Common Subsumer

LRMB – Layered Reference Model of the Brain

LTM – Long-Term Memory

MLP – Multilayer Perceptron

MRPC – Microsoft Research Paraphrase Corpus

NLP – Natural Language Processing

NMT – Neural Machine Translation

OAR – Object-Attribute-Relation

REST – Representational State Transfer

RNN – Recurrent Neural Network

ROC – Receiver Operating Characteristic

STS – Semantic Textual Similarity

SVM – Support Vector Machine

WSD – Word Sense Disambiguation

ВСТУП

Актуальність теми дослідження. Сучасний розвиток інформаційних технологій створює нові передумови для дослідження та моделювання складних когнітивних процесів, що лежать в основі людської розумової діяльності. Пізнання є фундаментальним процесом, який забезпечує здатність людини отримувати, інтерпретувати, зберігати та обробляти інформацію. Воно виступає основним механізмом відображення об’єктивної дійсності у свідомості, сприяючи формуванню знань і прийняттю рішень.

Моделювання когнітивних процесів є пріоритетним завданням міждисциплінарних досліджень, що об’єднують досягнення психології, когнітивістики, штучного інтелекту та лінгвістики. Особливу увагу привертає моделювання когнітивного процесу розуміння, який є ключовим компонентом пізнання і лежить в основі інтерпретації інформації, отриманої людиною з навколишнього середовища. Такі моделі повинні відображати не лише формальну обробку інформації, але й глибоке семантичне розуміння змісту, аналогічно до того, як це відбувається в людському мисленні.

Одним із перспективних підходів до формалізації процесу розуміння в комп’ютерних системах є дослідження перифразування – здатності передавати одну й ту ж інформацію різними мовними засобами. Розв’язання задачі ідентифікації перифраз дозволяє виявляти семантичну еквівалентність між різними формулюваннями однієї й тієї ж думки, що безпосередньо відповідає цілям когнітивного моделювання процесу розуміння. У цьому контексті ідентифікація перифраз розглядається як прикладна реалізація когнітивної моделі розуміння, яка формує концептуальну основу для побудови семантично орієнтованих інтелектуальних систем.

Попри активний розвиток задачі ідентифікації перифраз у галузі обробки природної мови (NLP), більшість сучасних підходів розглядають її переважно як класифікаційну проблему, що базується на поверхневих векторних репрезентаціях без глибинного семантичного аналізу. Попри помітний прогрес

для англійської та інших високоресурсних мов, дослідження ідентифікації перифраз для української мови обмежене нестачею відповідних мовних ресурсів, зокрема спеціалізованих наборів даних перифраз. Наявні підходи переважно орієнтовані на статистичні або евристичні методи, які не здатні повною мірою відобразити глибинні семантичні зв'язки між висловленнями.

Таким чином, розробка нових і вдосконалення наявних методів, технологій і моделей підвищення точності ідентифікації перифраз у текстах українською мовою є актуальною науково-прикладною задачею. Вона передбачає інтеграцію когнітивно орієнтованого підходу до моделювання процесу розуміння, застосування глибинного семантичного аналізу та використання семантичних графових репрезентацій, що дозволяє забезпечити більш точне виявлення концептуальної еквівалентності між висловленнями незалежно від їхньої лексичної та синтаксичної форми. Такий підхід сприяє розвитку когнітивно орієнтованих лінгвістичних систем і створює основу для практичного впровадження в задачах автоматичного розпізнавання перифраз, інформаційного пошуку та обробки природномовних текстів.

Зв'язок роботи з науковими програмами, планами, темами. Обраний напрямок досліджень пов'язаний із виконанням автором науково-дослідних робіт кафедри програмного забезпечення комп'ютерних систем НТУ «Дніпровська політехніка» «Методи, моделі та технології обробки даних в комп'ютерних системах загального та спеціального призначення» (державний реєстраційний номер 0124U004583) та «Високопродуктивні багатопроцесорні системи: особливості конструювання. Дослідження оцінок ефективності застосування до розв'язування прикладних задач» (державний реєстраційний номер 0121U113718).

Мета і завдання дослідження. Метою дисертаційної роботи є підвищення точності ідентифікації перифраз у текстах українською мовою в контексті моделювання когнітивного процесу розуміння шляхом розробки інтелектуальної системи, яка використовує семантичні графові репрезентації для виявлення семантичної еквівалентності між висловлюваннями на концептуальному рівні.

Для досягнення поставленої мети дисертаційного дослідження необхідно розв’язати наступні задачі:

– проаналізувати та формалізувати когнітивні передумови процесу розуміння природномовних висловлень, що мають значення для встановлення семантичної еквівалентності між ними, з метою визначення вимог до семантичних репрезентацій і механізмів їх порівняння;

– виконати аналіз наявних підходів до генерації абстрактних семантичних репрезентацій текстів природної мови;

– розробити метод генерації абстрактних репрезентацій змісту (AMR) для текстів природної мови, орієнтований на багатомовну обробку та збереження семантичної повноти текстів;

– розробити метод визначення семантичної еквівалентності між графовими семантичними репрезентаціями текстів природної мови;

– створити архітектуру інтелектуальної системи моделювання когнітивного процесу розуміння шляхом розв’язання задачі ідентифікації перифраз;

– визначити структуру, функціональні модулі та алгоритм роботи інтелектуальної системи ідентифікації перифраз, включно з модулями підготовки текстових даних, побудови AMR графів та класифікації;

– розробити інформаційну технологію підготовки багатомовних текстів та побудови AMR графів;

– провести експериментальну перевірку ефективності розробленої технології.

Об’єкт дослідження – процеси обробки та інтерпретації природної мови для визначення семантичної еквівалентності в інтелектуальних системах.

Предмет дослідження – методи та моделі створення інтелектуальних систем ідентифікації перифраз, що забезпечують відтворення й аналіз семантичної еквівалентності між висловлюваннями.

Методи дослідження. У роботі застосовано комплекс методів, що забезпечують побудову та верифікацію інтелектуальної системи для ідентифікації перифраз у контексті моделювання когнітивного процесу розуміння текстів природної мови. До них відносяться когнітивне моделювання для формалізації процесу розуміння та побудови архітектури системи, а також методи аналізу і формалізації семантичних зв'язків тексту з використанням графових репрезентацій AMR. Для підготовки вхідних даних застосовано методи нормалізації текстів, трансформації українськомовних висловлювань з використанням нейронного машинного перекладу для сумісності з англомовними моделями AMR. Семантичний аналіз і векторизацію репрезентацій забезпечують графові нейронні мережі та трансформерні моделі, а визначення семантичної еквівалентності висловлень реалізовано методами машинного навчання, зокрема бінарної класифікації. Для перевірки ефективності системи проведено комп'ютерні експерименти, тренування та валідацію моделей, оцінку точності результатів із застосуванням методів математичної статистики, а також порівняльний аналіз з наявними підходами, що дозволяє підтвердити ефективність розробленої інформаційної технології та підвищення точності ідентифікації перифраз.

Наукова новизна отриманих результатів. У дисертаційній роботі започатковано новий комплексний формалізований підхід до ідентифікації перифраз на основі семантичних графових репрезентацій та трансформерних архітектур, що дозволило підвищити точність визначення семантичної еквівалентності між висловленнями в українській мові.

При цьому отримано такі основні наукові результати:

– Запропоновано когнітивно орієнтовану методику до ідентифікації перифраз у текстах українською мовою, що ґрунтується на використанні семантичних графових репрезентацій та дозволяє здійснювати аналіз семантичної еквівалентності висловлювань на концептуальному рівні в межах моделювання когнітивного процесу розуміння.

– Вперше розроблено когнітивно обґрунтовану модель перифраза як концептуально-еквівалентного смислового варіанту в межах семантичного простору абстрактних репрезентацій змісту, що дозволяє уніфікувати підхід до ідентифікації перифраз незалежно від їхньої лексичної та синтаксичної форми.

– Вперше розроблено інформаційну технологію формування AMR графів для українськомовних текстів на основі їхнього попереднього машинного перекладу. Така технологія гарантує збереження семантичної повноти текстів українською мовою при перекладі на рівні 94% за відсутності повноцінних українськомовних AMR корпусів.

– Удосконалено метод репрезентації текстів природної мови за допомогою AMR шляхом використання графових нейронних мереж з механізмом уваги, які дозволяють формувати векторні представлення, що відображають як локальні, так і глобальні властивості смислової структури висловлювання, що забезпечує підвищення точності у задачі оцінювання семантичної подібності текстів на 4% порівняно з наявними AMR орієнтованими підходами.

– Удосконалено метод ідентифікації перифраз шляхом застосування класифікації на основі векторних представлень AMR графів, що забезпечує прийняття рішень про семантичну еквівалентність висловлювань на рівні концептуально-смислових структур та дозволяє досягати показників точності та F1-міри на 11% та на 4,2% відповідно вищими за наявні підходи, що використовують AMR графи для ідентифікації перифраз.

Практичне значення отриманих результатів дослідження полягає у застосуванні розробленої інформаційної технології для автоматизованої ідентифікації перифраз у текстах українською мовою. Розроблені графові репрезентації та методи обробки забезпечують ефективне моделювання когнітивного процесу розуміння на концептуальному рівні, що дозволяє підвищувати точність визначення семантичної еквівалентності між висловлюваннями.

Практична реалізація цієї технології може бути використана у системах штучного інтелекту для автоматичного розпізнавання перифраз, інформаційного пошуку, реферування текстів, навчальних і наукових застосунках, а також у розвитку когнітивно орієнтованих лінгвістичних систем.

Особистий внесок здобувача. Дисертаційна робота є самостійно виконаною науковою працею, у якій усі теоретичні та прикладні результати отримані автором особисто. Особистий внесок здобувача в отриманні наукових результатів підтверджується самостійним дослідженням теоретичних і методологічних аспектів проблеми, яка розглядається. У дисертації не використано ідей співавторів публікацій.

У наукових публікаціях, створених у співавторстві, здобувачу належить: порівняльний аналіз методів ідентифікації перифраз, обґрунтування доцільності застосування гібридних підходів на основі AMR графів для моделювання когнітивних процесів розуміння [2]; розробка інформаційної технології побудови векторних репрезентацій AMR графів, проєктування архітектури сіамської нейронної мережі на основі Graph Attention Network (GAT), реалізація процедури навчання та проведення експериментальної перевірки моделі [4]; розробка архітектури нейронної мережі Bi-LSTM для розв’язання задачі усунення багатозначності слів (WSD), формування спеціалізованого навчального набору даних на основі академічних словників, програмна реалізація та аналіз ефективності моделей [5, 8]; розробка моделі оцінки когнітивних навичок, обґрунтування методу адаптивного добору навчальних методик [6].

Апробація результатів дисертації. Основні положення та результати дисертаційної роботи було представлено та обговорено на таких міжнародних та всеукраїнських науково-технічних конференціях: XVIII Міжнародна науково-практична конференція «Проблеми використання інформаційних технологій в освіті, науці та промисловості» (м. Дніпро, НТУ «ДП», 2023 р.); XIX Міжнародна науково-практична конференція «Проблеми використання інформаційних технологій в освіті, науці та промисловості» (м. Дніпро, НТУ «ДП», 2024 р.); 7th

International Scientific and Practical Internet Conference «Integration of Education, Science and Business in Modern Environment: Summer Debates» (м. Дніпро, 2025); IX Міжнародна науково-практична конференція «INTERNATIONAL EXPERIENCE IN SCIENTIFIC RESEARCH» (м. Чікаго, 2026).

Публікації. Матеріали дисертаційної роботи повною мірою викладені у 10 публікаціях, з них – 5 статей у фахових періодичних виданнях України з технічних наук, з яких 1 – категорії А (індексується в Scopus), 4 – категорії Б; 5 наукових праць, що засвідчують апробацію матеріалів дисертації, зокрема 1 стаття в українському науковому періодичному виданні та 4 публікації у матеріалах міжнародних наукових конференцій.

Структура та обсяг роботи. Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел та додатків. Загальний обсяг дисертації становить 196 сторінок, містить 142 сторінки основної частини, включає 28 рисунків, 21 таблицю, 107 літературних джерел. Додатки на 25 сторінках включають список наукових публікацій здобувача за темою дисертації, документи, що підтверджують упровадження результатів дисертації, а також програмні компоненти розробленої комп’ютерної технології ідентифікації перифраз.

РОЗДІЛ 1
ІНФОРМАЦІЙНО-АНАЛІТИЧНИЙ ОГЛЯД СТАНУ ПРОБЛЕМИ МОДЕЛЮВАННЯ КОГНІТИВНИХ ПРОЦЕСІВ

1.1. Теоретико-методологічні засади когнітивного пізнання та моделювання когнітивних процесів у штучних системах

Сучасний розвиток інформаційних технологій та експоненційне зростання потужностей обчислювальних машин надає можливості для дослідження та вивчення складних процесів розумової діяльності людини з метою її моделювання. Сукупність цих процесів, метою яких є отримання, обробка та розуміння інформації, називають пізнанням або когніцією – воно відіграє ключову роль у відображенні об’єктивної дійсності у свідомості людини та становить основу її розумової діяльності.

Історично проблематика пізнання була предметом філософського аналізу, але з появою інформаційних технологій у другій половині XX століття з’явилися перші спроби формалізованого моделювання когнітивних процесів із використанням комп’ютерних систем, зокрема штучного інтелекту. Основною складністю таких моделей була потреба у створенні уніфікованого формалізму та встановленні універсальних обмежень для представлення процесів мислення у вигляді моделей. При цьому особливу увагу приділяли культурно-мовному контексту, який визначає способи інтерпретації інформації та прийняття рішень людиною.

З метою подолання зазначених проблем пізнання стає предметом міждисциплінарних досліджень у межах когнітивної науки, яка інтегрує результати філософії, психології, лінгвістики, нейробіології та комп’ютерних наук. Когнітивна наука розглядає пізнання як процес, що виходить за межі простих операцій з абстрактними символами, і наголошує на цілісності концептів, які несуть культурно-історичне навантаження. Відповідно до цього підходу, людина пізнає зовнішній світ, будуючи моделі та здійснюючи

прогнозування, поєднуючи сенсорну інформацію з апріорними очікуваннями та попереднім досвідом. Такий підхід створює основу для формалізації когнітивних процесів у штучних системах і визначає принципи побудови моделей, які намагаються відтворити людське розуміння та прийняття рішень.

Для дослідження та відтворення моделей пізнання важливе розуміння роботи та взаємодії когнітивних процесів розумової діяльності, які складають сутність пізнання людини. Когнітивні процеси виникають у розумі під час отримання стимулу з навколишнього середовища та до моменту явної поведінкової реакції на цей стимул, і вони необхідні для отримання знань, маніпулювання інформацією та міркування при вирішенні будь-яких завдань.

Одним з перших та вагомих досліджень з ідентифікації та класифікації когнітивних процесів є робота американського психолога Бенджаміна Блума, який, намагаючись звести до єдиної системи набір розрізнених цілей і завдань у навчанні, створив теорію класифікації когнітивних процесів. Ця класифікація отримала назву таксономії Блума [1].

Блум виділив шість категорій розумових процесів, починаючи від процесів нижчого порядку, які вимагають меншої когнітивної обробки, до процесів вищого порядку, які потребують глибшого навчання та більшого ступеня когнітивної обробки. Таксономія Блума уособлює у собі ієрархію процесів знання (knowledge), осмислення (comprehension), застосування (application), аналізу (analysis), оцінки (evaluation) та синтезу (synthesis).

З розвитком когнітивної науки робота Блума була переглянута у роботі Лорена Андерсона та Девіда Кратвола. У переглянутій версії три категорії були перейменовані. Знання було змінено на запам’ятовування (remembering), осмислення стало розумінням (understanding), а синтез було перейменовано на створення (creation) (рис. 1.1.). Крім того, етап створення став найвищим рівнем у системі класифікації, перед яким йде етап оцінки. Також перегляд таксономії додав новий вимір у всі шість когнітивних процесів. Він визначає чотири типи знань, які можуть бути розглянуті під час навчальної діяльності: фактичні; концептуальні; процедурні й метакогнітивні. Окрім перегляду основних рівнів

розумових процесів у таксономії Блума, Андерсон та Кратвол навели класифікацію відповідних когнітивних навичок, які використовуються людиною на кожному рівні таксономії [2].

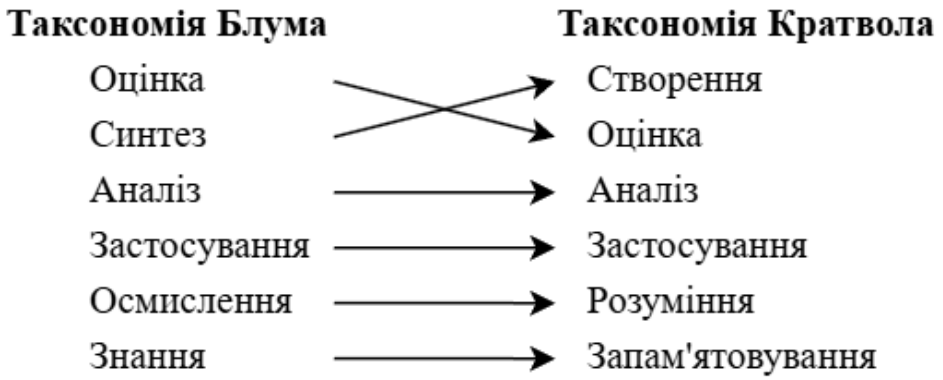

Рис. 1.1. Таксономія когнітивних навичок Блума та Кратвола

Однак, таксономія Блума та її переглянута версія розглядають когнітивні процеси переважно у контексті навчальної діяльності та не охоплюють повний спектр когнітивних функцій та їх взаємозв’язків у загальному пізнанні. З цією метою була розглянута багатошарова еталонна модель мозку (LRMB), яка формально описує життєві функції мозку та когнітивні процеси розуму. На відміну від таксономії Блума, модель LRMB виділяє 37 когнітивних процесів, об’єднаних у шість ієрархічних шарів: відчуття, пам’ять, сприйняття, дія, метакогнітивний шар та шар вищих когнітивних функцій (рис. 1.2) [3].

Рис. 1.2. Багатошарова еталонна модель мозку людини

Модель ґрунтується на розділенні життєвих функцій мозку на два рівні – свідомий і підсвідомий. Підсвідомий рівень охоплює шари відчуття, пам’яті, сприйняття, дії, а свідомий останні два шари – метакогнітивний та рівень вищих когнітивних функцій. Підсвідомі шари мозку фіксовані, коли людина народжується. На відміну від підсвідомих, свідомі шари мозку набуті, гнучкі, програмовані та використовуються людиною на основі готовності, цілей і мотивації [3].

Відповідна класифікація когнітивних процесів є фундаментальною основою когнітивного моделювання – основного інструменту когнітивної науки для побудови формальних описів і перевірки теорій пізнання. Когнітивне моделювання вивчає сутність пізнання та взаємодію когнітивних функцій людини шляхом створення обчислювальних моделей представлень, механізмів і процесів, що відтворюють різні аспекти мислення.

У межах когнітивного моделювання виділяють три типи моделей – обчислювальні, математичні та вербально-концептуальні [4]:

1. Обчислювальні моделі описують когнітивні процеси у вигляді алгоритмів або процедур, що дозволяють відтворити або симулювати їх роботу.

2. Математичні моделі подають зв’язки між змінними у вигляді формальних рівнянь.

3. Вербально-концептуальні моделі описують сутності, відношення й процеси засобами природної мови, без точних формальних обмежень.

Серед зазначених підходів найбільш перспективним у когнітивній науці є обчислювальне моделювання, оскільки воно надає не лише інструментарій для перевірки когнітивних теорій, а й основу для створення практичних інтелектуальних систем. При цьому математичні моделі розглядаються як підмножина обчислювальних, адже більшість із них можуть бути реалізовані у вигляді обчислювальних алгоритмів [5].

Теоретичні моделі когнітивних процесів можуть бути класифіковані за їх спрямованістю на продукт або процес. Теорії продукту описують зовнішні прояви когнітивної діяльності без уточнення внутрішніх механізмів, тобто виступають як «чорні скрині», що дають змогу оцінювати пізнання через досліджувану поведінку [6]. Теорії процесу, навпаки, зосереджуються на поясненні внутрішніх механізмів, через які виникає розумова діяльність людини, і намагаються відтворити сам перебіг когнітивних операцій.

Залежно від рівня деталізації процесів, обчислювальні моделі можуть варіюватися від чисто функціональних до детальних процесуальних, які враховують часову динаміку, послідовність і взаємодію ментальних станів. Відповідність таких моделей реальним когнітивним явищам може бути як якісною, так і кількісною, що залежить від ступеня формалізації та емпіричної перевірки [7]. Попри різноманітність підходів, головною метою когнітивного моделювання залишається пояснення пізнання через формалізацію його процесів.

Історично побудова моделей когнітивних процесів відбувається за трьома основними підходами: символічний, емерджентний та гібридний (рис. 1.3.).

Рис. 1.3. Еволюція підходів до побудови моделей когнітивних процесів

Першим підходом до когнітивного моделювання є символічний. У цьому традиційному підході людське пізнання можна розглядати приблизно так само, як процес символічної маніпуляції [8]. Символічні когнітивні архітектури – це класична теорія репрезентації, яка розглядає розумові операції над знаннями, що проходять через символічні репрезентації, а обробка інформації виконується послідовним та ієрархічним способом. Однак вони зазвичай не були чітко зіставлені з реальними даними. Тому було важко встановити когнітивну відповідність багатьох із цих моделей.

Розвиток моделей нейронних мереж у 1980-х роках призвів до появи іншого типу моделей у цій галузі – емерджентних моделей. На відміну від символічних моделей, які покладаються на різноманітні складні структури даних, що зберігають високоструктуровані фрагменти знань, в емерджентному підході для вивчення пізнання використовуються нейронні мережі, які

використовують прості, однакові та часто масово паралельні обчислення. Згідно з емерджентним підходом, пізнання є процесом адаптації до навколишнього середовища через збереження системою власної автономії. Основною концепцією когнітивного навчання в емерджентних моделях є пріоритетна конструкція навичок, а не отримання знань [9].

Гібридні моделі, які поєднують сильні сторони нейронних мереж і символічних моделей, з'явилися на початку 1990-х років. Моделі, що побудовані за даним підходом, поєднують у собі переваги символічних архітектур для представлення світу та реалізації міркування та аргументації на основі заданих правил та емерджентні архітектури нижчого рівня для реалізації механізмів отримання та обробки знань навколишнього середовища. Завдяки цьому гібридні моделі дають змогу створювати мультизадачні системи, які здатні навчатися та правильно інтерпретувати невідомі об'єкти самостійно без апріорної специфікації чи програмування [10].

Таким чином, еволюція підходів до когнітивного моделювання демонструє поступовий перехід від описових і вербальних теорій до формалізованих, обчислювальних моделей. Подальше дослідження у цій роботі зосереджено на моделюванні одного з центральних когнітивних процесів – процесу розуміння (comprehension), який лежить в основі сприйняття, інтерпретації та генерації природної мови.

1.2. Аналітичний огляд методів розв'язання задачі ідентифікації перифраз у контексті моделювання когнітивного процесу розуміння

Однією з ключових особливостей когнітивного процесу розуміння є здатність людини розпізнавати та співвідносити різні мовні форми, що виражають одне й те саме значення. У природному мовленні це виявляється, зокрема, у формі перефразування – коли одна і та ж ідея може бути передана за допомогою різних лексичних чи синтаксичних конструкцій. Для когнітивної моделі розуміння надзвичайно важливим є врахування цього явища, оскільки

воно демонструє здатність до абстрагування, узагальнення та зіставлення нової інформації з уже відомою.

Визначенням слова «перифраза» згідно з Оксфордським мовним словником є висловлювання, що передає те, що хтось написав або сказав, використовуючи інші слова [11]. Це визначення охоплює речення або фрази, що передають однакове семантичне значення, використовуючи різні формулювання. Оскільки визначення не специфікує конкретні властивості, що визначають семантичну еквівалентність між двома текстами, основною проблемою визначення перифраз є формалізація даних властивостей.

Наявні дослідження пропонують різні підходи до визначення семантичної еквівалентності. Так, наприклад при розв’язанні задачі семантичного слідкування, семантична еквівалентність розглядається як зв’язок між двома текстами, де значення одного тексту може бути виведено з контексту іншого тексту, і навпаки [12]. Згідно з пропозиційною логікою, семантична еквівалентність визначається як такий ступінь симетрії між двома текстами, коли один текст є підмножиною іншого, але не обов’язково належить до нього [13]. Дане визначення дозволяє вважати перифразами такі речення, які не повністю передають зміст оригінального тексту. У іншій роботі наголошується що морфологічна структура речення не визначає його значення і що семантичне значення речення натомість визначається конкретними елементами, які його складають [14].

Таким чином, у різних дослідницьких контекстах семантична еквівалентність трактується по-різному – як симетричне логічне включення, як взаємозалежність контекстів або як функція розподілу лексичних одиниць у схожих ситуаціях. Водночас ці підходи демонструють, що встановлення повної семантичної відповідності між реченнями або текстами не завжди є можливим і необхідним для класифікації їх як перифраз.

Вимірювання ступеня семантичної подібності у текстах використовується для виконання більшості завдань обробки природної мови, таких як відповіді на питання, виявлення плагіату, оцінка систем машинного перекладу тощо.

Відповідно існує багато підходів та методів розв’язання задачі, які можна умовно поділити на декілька основних груп, а саме методи засновані на традиційних підходах та методи глибокого навчання.

1.2.1. Традиційні методи та підходи на основі корпусів і знань

Традиційні підходи до обробки природної мови зосереджувалися переважно на двох ключових рівнях лінгвістичного аналізу – морфології та синтаксисі. Ці рівні забезпечували основу для формального представлення тексту та виявлення структурних зв’язків між словами. У межах морфологічного аналізу увага приділялась формальним характеристикам слів: основам, афіксам, відмінкам, числу, часу, роду тощо. Синтаксичний аналіз, своєю чергою, дозволяє виявляти граматичні залежності між словами в реченні – наприклад, зв’язок між підметом і присудком або між дієсловом і його додатком. Загалом традиційні підходи до розв’язання задачі ідентифікації перифраз об’єднують у собі методи на основі знань, а також методи на основі корпусів.

Методи засновані на знаннях це підходи, які спираються на структуровані лінгвістичні ресурси, тобто на бази знань про мову, створені вручну або напівавтоматично. Вони відображають взаємозв’язки між поняттями, синонімами, паронімами, антонімами тощо. Більшість методів заснованих на знаннях передбачають використання WordNet, лексичної бази даних семантичних відношень, яка організовує слова в групи синонімів і визначає різні типи зв’язків між ними, такі як гіпонімія, гіперонімія, антонімія та інші [15].

Ступінь семантичної подібності у даних методах визначається за допомогою метрик, які поділяються на дві основні групи: метрики семантичної подібності та метрики семантичної спорідненості (рис. 1.4.). Семантична спорідненість в даному контексті використовується як загальне поняття, що не обов’язково прив’язане до форми концепту. Виходячи з цього поняття семантична подібність є різновидом семантичної спорідненості між двома текстовими одиницями та охоплює ширший діапазон зв’язків між концептами,

що включає додаткові відносини подібності, такі як «є різновидом», «є специфічним прикладом», «є частиною», «є протилежністю» [16].

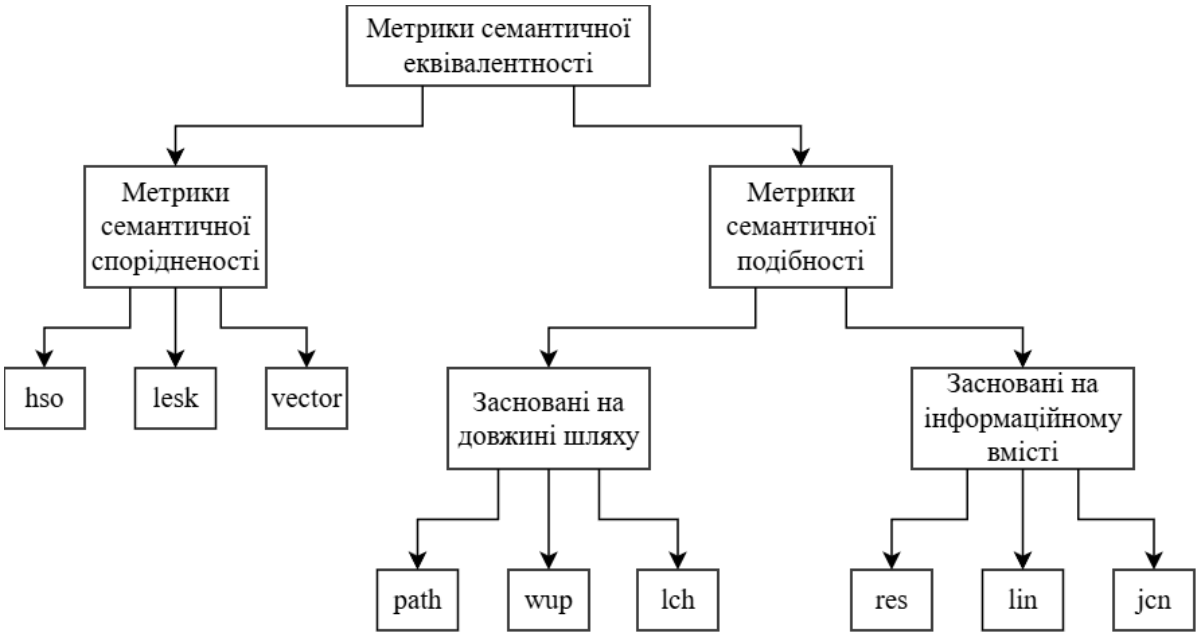

Рис. 1.4. Таксономія метрик семантичної еквівалентності у методах заснованих на знаннях

Перша група, що містить у собі метрики res, lin та jcn, базується на інформаційному наповненні. Інша група метрик базується на довжині шляху та містить у собі метрики lch, wup і path [17].

Метрика res визначає подібність між поняттями на основі інформаційного вмісту їхнього найближчого спільного предка:

$$sim_{res}(A, B) = IC\big(LCS(A, B)\big), \tag{1.1}$$

$$IC(c) = -\log P(c), \tag{1.2}$$

де IC(c) – інформаційне наповнення концепту c, P(c) – ймовірність зустрічі концепту c у корпусі текстів, LSC(A, B) – це найближчий спільний предок для концептів A та B у таксономічній ієрархії.

Метрики lin та jcn також спираються на інформаційний вміст понять, однак вони не обмежуються лише величиною спільної інформації, а враховують баланс між загальним та унікальним інформаційним внеском кожного поняття. У цих підходах інформаційний вміст найближчого спільного предка зіставляється із сумарним інформаційним вмістом двох концептів A та B.

У метриці lin семантична подібність визначається як відношення подвоєної величини інформаційного наповнення їхнього спільного предка до суми інформаційного наповнення обох понять:

$$sim_{lin}(A,B) = \frac{2 \times IC(LCS(A,B))}{IC(A)+IC(B)}, \quad (1.3)$$

де IC(A), IC(B) – інформаційне наповнення концептів A і B відповідно, LSC(A, B) – найближчий спільний предок для концептів A та B.

На відміну від цього, jcn пропонує трактувати різницю інформаційного наповнення як міру відстані між поняттями. Спочатку обчислюється інформаційна відстань:

$$dist_{jcn}(A,B) = IC(A) + IC(B) - 2 \times IC(LCS(A,B)), \quad (1.4)$$

$$sim_{jcn}(A,B) = \frac{1}{dist_{jcn}(A,B)}, \quad (1.5)$$

де IC(A), IC(B) – інформаційне наповнення концептів A і B відповідно, LSC(A, B) – найближчий спільний предок для концептів A та B.

Відповідно, для перетворення відстані у подібність використовують обернене значення, що отримане за формулою (1.5), або інші нормалізації. Таким чином, обидві метрики є більш чутливими до структурної та статистичної інформації, оскільки вони одночасно враховують кількість спільної інформації та кількість унікальної інформації кожного з концептів.

Інша група метрик оцінює подібність на основі топологічних характеристик ієрархії WordNet, зокрема довжини шляху та глибини вузлів. До

неї належать lch, wup та path. Основне припущення цієї групи полягає в тому, що семантично близькі поняття розташовані ближче одне до одного в ієрархічній структурі.

Найпростішою є метрика path, яка вимірює подібність як обернену величину довжини найкоротшого шляху між двома поняттями:

$$sim_{path}(A, B) = \frac{1}{length(A,B)}, \quad (1.6)$$

де length(A, B) – довжина найкоротшого шляху між концептами A та B.

Метрика lch додатково враховує максимальну глибину ієрархії:

$$sim_{lch}(A, B) = -\log\frac{length(A,B)}{2\times D}, \quad (1.7)$$

де length(A, B) – найкоротший шлях між поняттями A та B, D – максимальна глибина ієрархії в WordNet.

Метрика wup оцінює схожість через відношення глибини їх спільного предка до сумарної глибини самих понять:

$$sim_{wup}(A, B) = \frac{2\times depth(LCS(A,B))}{depth(A)+depth(B)}, \quad (1.8)$$

де depth(A), depth(B) – глибина понять A та B в ієрархії, LCS(A, B) – найближчий спільний предок понять A та B.

Окрім метрик семантичної подібності, існує також група метрик, що оцінюють семантичну спорідненість понять. До неї належать метрики hso, lesk та векторна міра vector.

Метрика hso побудована на структурних властивостях мережі WordNet. Вона враховує як довжину шляху між двома концептами, так і кількість змін типу семантичного зв’язку вздовж цього шляху. Логіка метрики полягає в тому,

що коротший шлях з мінімальною кількістю змішаних типів відношень свідчить про більшу семантичну близькість між поняттями.

Метрика lesk заснована на ідеї перетину словникових визначень. Для оцінки близькості двох слів порівнюють слова, що використовуються в їхніх визначеннях у лексичному ресурсі. Семантична спорідненість тим вища, чим більше спільних лексем або стійких словосполучень містять ці глоси.

Векторна метрика vector спирається на статистику спільної появи термінів у корпусі текстів. Кожному слову, що входить до визначень у WordNet, відповідає вектор контекстів, у яких воно зустрічається. Таким чином, кожне словникове визначення або концепт представляється як усереднений вектор своїх елементів, і порівняння виконується через співвідношення цих векторів.

Оскільки метрики семантичної спорідненості фіксують лише загальну тематичну чи асоціативну близькість, а не точну еквівалентність змісту, їх використання не знайшло поширення при розв’язанні задачі ідентифікації перифраз на відміну від метрик семантичної подібності. Так, наприклад ці метрики використовуються у роботі Р. Міхалчі та співавторів [18] для визначення семантичної подібності між словами у двох текстах, через перетворення подібності концептів до подібності слів за допомогою вибору слів у зв’язках WordNet.

Іншим важливим класом підходів до розв’язання задачі ідентифікації перифраз є методи, засновані на корпусах. Ці методи виходять із припущення, що семантичні зв’язки між словами або висловленнями можуть бути виявлені шляхом аналізу їхньої статистичної поведінки у великих текстових колекціях. Зокрема, спільна частота появи слів, характерні контексти їх уживання та отримані на цій основі багатовимірні векторні представлення дозволяють моделювати ступінь їхньої семантичної близькості.

У межах корпусного підходу сформувалося кілька напрямів. Один із них ґрунтується на побудові матриць спільної появи «слово – контекст» та їх подальшій матричній факторизації з метою отримання компактних семантичних представлень. Інший напрям передбачає використання класичних алгоритмів

машинного навчання, зокрема методів опорних векторів SVM, які застосовуються для класифікації або порівняння отриманих текстових репрезентацій (рис. 1.5.).

Рис. 1.5. Таксономія методів заснованих на корпусах

Матрична факторизація – це метод зменшення розмірності даних, який полягає у розкладанні великої матриці на добуток двох або більше менших матриць, що зберігають приховану семантичну інформацію. У контексті обробки природної мови така матриця зазвичай представляє частоту спільної появи слів і контекстів у текстовому корпусі. Факторизація дозволяє вивести латентні ознаки, які відображають семантичну близькість між словами або фразами. Одним із найпоширеніших методів є сингулярне розкладання матриці, що дозволяє зменшити розмірність простору і представити кожне слово як вектор у щільному латентному просторі. Такі вектори можна використовувати для обчислення подібності між словами й реченнями. Цей метод лежить в основі латентного семантичного аналізу LSA, уперше запропонованому в роботі Т. Ландауера та співавторів [19]. У LSA вхідними даними зазвичай є матриця термін-документ, де кожен рядок відповідає слову або терміну, а кожен стовпець – документу або контексту, а значення відображають частоту вживання слова в документі, з можливим зважуванням за схемою tf-idf, що поєднує дві метрики: частоту терміна tf та обернену частоту документів idf. Метод SVD зменшує розмірність цієї матриці, дозволяючи представити слова та документи у спільному низько вимірному векторному просторі, де семантично подібні

об’єкти розміщені близько один до одного. У цьому просторі можливо виконувати обчислення подібності, що дає змогу виявляти перифрази.

Підходи на основі SVM стали важливим етапом у розвитку автоматичної ідентифікації парафраз, особливо після публікації набору даних Microsoft Research Paraphrase Corpus у 2005 році [20]. У цьому дослідженні вперше була продемонстрована ефективність використання SVM для класифікації пар речень як перифраз або не перифраз. SVM добре підходять для обробки розріджених та високовимірних ознак, які часто виникають при векторизації текстів. Модель використовує гіперплощину для розмежування позитивних і негативних прикладів, максимізуючи відстань між класами. Завдяки стійкості до шуму в навчальних даних та можливості застосування нелінійних ядерних функцій, SVM показали хороші результати на обмежених або слабко анотованих даних. Одним з ефективних методів побудови класифікаторів для задачі ідентифікації парафраз є використання ядра на основі радіальної базисної функції в межах моделей опорних векторів [21]. На відміну від лінійного ядра, яке обчислює схожість на основі скалярного добутку вхідних векторів, RBF-ядро вимірює подібність між зразками на основі евклідової відстані, перетвореної через експоненційну функцію. У задачі виявлення парафраз цей підхід дозволяє моделі виявляти нелінійні залежності між ознаками, навіть коли сам простір ознак є досить вузьким. Це особливо корисно в умовах, коли кількість вхідних ознак обмежена, але наявна велика кількість навчальних даних, які дозволяють моделі навчитися складних семантичних співвідношень. Попри популярність SVM, їх використання у реальних сценаріях часто обмежене через високу обчислювальну складність і нездатність масштабуватися на великі обсяги текстових даних, особливо у випадках з високою розмірністю векторного простору. Внаслідок цього, SVM дедалі частіше поєднуються з іншими методами або замінюються сучасними нейронними підходами.

1.2.2. Моделі на основі глибокого навчання та дистрибутивної семантики

Поява SVM стала важливим перехідним етапом від класичних статистичних методів до складніших моделей штучного інтелекту, включаючи глибоке навчання. На відміну від простих евристичних або ймовірнісних моделей, SVM дозволяють моделювати нелінійні залежності у високовимірних просторах ознак, що стало критично важливим у задачах семантичного аналізу мови, зокрема в ідентифікації перифраз.

З розвитком глибинного навчання парадигма обробки природної мови поступово змістила фокус з традиційних підходів, що засновані на ручному експертному конструюванні ознак, до розподілених семантичних подань, що дозволяють моделювати значення слів, фраз і речень у вигляді векторів у безперервному просторі. Одним із ключових напрямів, що виник у цьому контексті, є композиційна дистрибутивна семантика, яка базується на гіпотезі, що значення складніших мовних одиниць, таких як речення чи абзац, можна отримати шляхом об’єднання семантичних представлень їх складових елементів. Методи, що направлені на обчислення семантичних репрезентацій є надзвичайно важливими для задачі ідентифікації перифраз, оскільки дозволяють порівнювати значення текстів не лише на поверхневому рівні, а й на глибинному семантичному рівні. Методи композиційної дистрибутивної семантики формують семантичні репрезентації на різних рівнях представлення та відповідно поділяються на наступні групи:

– методи на рівні слів – базуються на векторних представленнях слів, що формуються через контекст в корпусі;

– методи на рівні фраз – враховують синтаксичні та семантичні зв’язки між словами в межах коротких фразових конструкцій;

– методи на рівні речень – орієнтовані на моделювання цілісного значення речення за допомогою нейронних моделей.

Зазначені рівні репрезентації охоплюють широкий спектр методів, що поєднують різні підходи та архітектури до глибинного навчання, а саме:

1. Ранні архітектури – до них належать базові моделі, які формують семантичні представлення на основі простих вкладень слів або автокодувальників. Такі моделі зазвичай не враховували складну синтаксичну чи контекстуальну структуру, однак слугували основою для подальшого розвитку більш потужних архітектур.

2. Традиційні глибокі нейронні мережі – цей підхід включає використання згорткових нейронних мереж та рекурентних нейронних мереж, які здатні виявляти послідовні шаблони в тексті та моделювати залежності між елементами на рівні фраз і речень.

3. Трансформерні архітектури – сучасний стандарт у задачах обробки природної мови, який забезпечує гнучке моделювання складних залежностей у тексті. Серед них особливо виділяються моделі на кшталт BERT, GPT та їхні численні варіації, які демонструють високі результати в задачах семантичної подібності.

Повна таксономія архітектур на основі глибокого навчання представлена на рисунку 1.6.

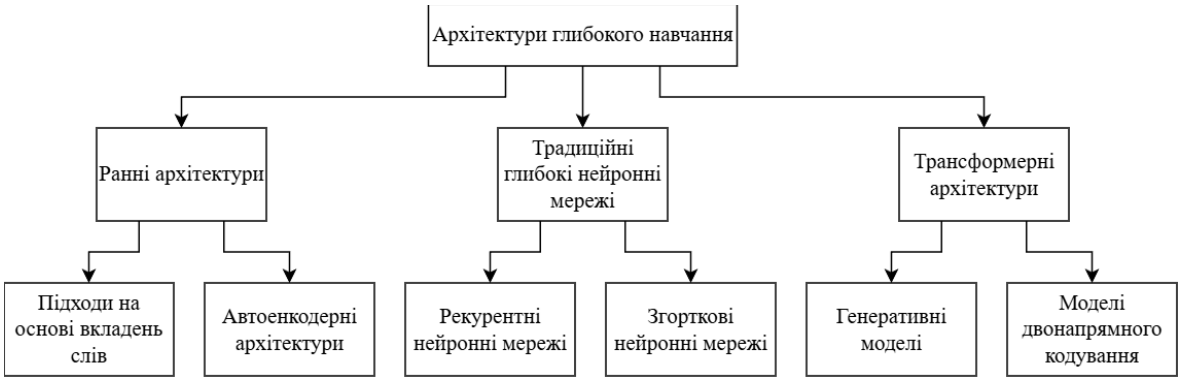

Рис. 1.6. Таксономія методів на основі глибокого навчання

Ці архітектури поєднуються в гібридних моделях, які дозволяють отримувати багаторівневі, контекстуально насичені представлення тексту для більш точної ідентифікації парафраз.

Основною проблемою ідентифікації перифраза є визначення семантичної подібності між двома реченнями-кандидатами. Лінгвістичні об’єкти – такі як слова, фрази чи речення – є дискретними одиницями, які не мають внутрішньої числової структури, що ускладнює їх використання у математичних моделях. Таким чином моделі ідентифікації перифраз мають не лише фіксувати лексичну відповідність, а й виявляти семантичну еквівалентність між різними лінгвістичними формулюваннями. Щоб вирішити це, необхідно перейти від дискретного представлення слів до векторних представлень, які моделюють семантичний зміст. З цією метою були запропоновані методи вкладання слів – числові вектори фіксованої довжини, які кодують семантичну подібність між словами. Завдяки репрезентації, що надають методи, слова можуть зберігати свою семантичну інформацію на основі контексту у якому використано слово [22].

Ідея використання векторних представлень з’явилася при розв’язанні проблеми розмірності у великомасштабних мовних моделях – тобто експоненційного зростання параметрів при зростанні словника [23]. Через те що задачею цих моделей було прогнозування появи слів в послідовному тексті, моделі, що обчислювали вкладення слів, отримали назву моделей на основі передбачення. Особливість запропонованих моделей полягає в тому, що слова спочатку проєктуються на шар вкладень, а вже потім передаються на наступні рівні нейронної мережі. Завдяки цьому шар вкладень «навчається» під час оптимізації функції втрат на основі завдання передбачення. У класичних моделях, таких як нейронна імовірнісна мовна модель, цей підхід дозволив формувати векторні простори, де семантично близькі слова мають подібні вектори [24]. Надалі ця ідея була вдосконалена у таких моделях, як Word2Vec, де архітектури CBOW та Skip-Gram навчали вектори за аналогічною логікою – передбачення слова за контекстом або навпаки [25]. У контексті ідентифікації

перифраз такі моделі стали основою для оцінки семантичної подібності між словами, яка згодом узагальнюється до вищих рівнів – фраз і речень.

Іншим важливим класом моделей вкладання слів є моделі на основі підрахунку, які формують семантичні представлення не шляхом прогнозування, а через аналіз частоти спільної появи слів та контекстів у великому корпусі текстів. Основна ідея цих підходів полягає у побудові матриці спільної появи, де кожен елемент відображає, наскільки часто певне слово з'являється в контексті іншого слова. Векторні представлення отримуються через зменшення розмірності такої матриці з використанням методів та технік матричної факторизації. Популярною моделлю, що реалізує даний підхід є Glove [26]. Її ключова ідея полягає в тому, що семантичні відношення між словами ефективно відображаються через статистичні співвідношення їхніх спільних появ у всьому корпусі. Завдяки використанню глобальної статистичної інформації про весь корпус, а не лише локального контексту, Glove часто демонструє вищу якість вкладень слів порівняно з передбачальними моделями.

Загалом усі моделі, що використовують вкладення слів, об'єднуються у групу простих моделей на основі вкладень слів. Основна ідея таких моделей полягає у використанні попередньо навчених або спільно навчених векторних представлень слів, які відображають семантичну схожість між словами в багатовимірному просторі. У підходах на основі вкладень слів кожне речення перетворюється на послідовність векторів, що відповідають словам у цьому реченні. Далі ці вектори агрегуються у єдине фіксоване представлення за допомогою простих операцій, таких як:

– середнє – обчислення середнього значення по кожній координаті всіх векторів;

– максимум – вибір максимального значення по кожній координаті;

– конкатенація середнього та максимуму – поєднання двох типів агрегації для збереження більшої кількості інформації;

– сумування – обчислення сумарного вектора всіх слів.

Агрегований вектор для кожного речення подається як вхід у класифікатор, який вивчає відповідність між парами речень, класифікуючи їх як перифраз або не перифраз. Попри простоту та ефективність моделей на основі векторних представлень слів, вони мають низку структурних обмежень, серед яких основним є нечутливість до порядку слів у реченнях. Обмеження полягає в тому, що у варіантах із середнім або максимальним згортанням, повністю ігнорується порядок слів у реченні через що, наприклад, речення «собака вкусив чоловіка» та «чоловік вкусив собаку» матимуть ідентичне або дуже схоже представлення, хоча їхній зміст кардинально різний. Спробою подолання цієї проблеми став метод, запропонований Д. Шеном та співавторами у роботі [27]. У цьому дослідженні було впроваджено комбіновані механізми агрегації – максимальне та ієрархічне об’єднання, що враховує структуру дерева семантичного розбору. На відміну від традиційного підходу, який використовує лише кількісні ознаки, автори дослідження запропонували навчати фіксовані представлення для n-грамів, що з’являються у корпусі, тим самим зберігаючи локальну просторову інформацію текстової послідовності.

Підходи композиційної дистрибутивної семантики на рівні фраз є важливим складником моделювання локальної семантики. На відміну від моделей, які формують значення тексту шляхом простого агрегованого об’єднання векторів слів, фразові методи орієнтовані на врахування синтаксичної структури та взаємозв’язків між компонентами фрази, що дозволяє краще захоплювати складні семантичні зв’язки.

Однією з перших архітектур, що дозволила ефективно моделювати композиційні семантичні залежності на рівні фраз та речень, стали рекурсивні автокодувальники. Ці моделі дозволяють відображати ієрархічну структуру природної мови, оперуючи не лише послідовностями, як у рекурентних нейронних мережах, а й деревоподібними структурами, що дозволяють закодувати синтаксичні зв’язки.

Ключова ідея рекурсивних автокодувальників полягає в тому, щоб рекурсивно об’єднувати векторні представлення пар слів або фраз, відповідно до

структури синтаксичного дерева речення, яке відображає будову речення відповідно до граматичних правил мови. На кожному кроці алгоритму модель формує вектор для батьківського вузла, об’єднуючи вектори двох дочірніх елементів – зазвичай через лінійне перетворення та нелінійну функцію активації. Обчислений вектор використовується як вхід для наступного рівня рекурсії. Паралельно цьому процесу, декодер намагається реконструювати початкові вектори дочірніх вузлів, що дозволяє мінімізувати функцію втрат та навчати модель методом навчання без вчителя.

Цю модель вперше було застосовано до задачі ідентифікації перифраз у роботі [28], де Р. Сохер зі співавторами запропонували розширену архітектуру рекурсивного автокодувальника, яка обчислювала векторні представлення для кожної вершини синтаксичного дерева. Модель було доповнено динамічним шаром об’єднання для розв’язання проблеми змінної довжини вхідних даних при класифікації перифраз, що дозволяє перетворювати представлення у фіксовані вектори.

Альтернативним підходом до формування фразових представлень є використання рекурентних нейронних мереж, відомих як RNN – класу нейронних мереж, які були розроблені для обробки послідовних даних, таких як текст, часова інформація, аудіо чи відео. На відміну від традиційних нейронних мереж, у яких кожен новий вхід обробляється незалежно від попередніх, рекурентні нейронні мережі мають зворотні зв’язки, що дозволяє їм зберігати інформацію про попередні стани. Використання рекурентних моделей, зокрема тих, що базуються на двонапрямній архітектурі або додатково використовують механізм уваги, демонструють покращення у вивченні контекстуально залежних представлень фраз. Однією з реалізацій такого підходу є кодувально-декодувальні RNN архітектури, які застосовуються для обробки пар фраз і речень з наступним обчисленням подібності між ними. У такій архітектурі перше речення кодується у вигляді вектора, після чого декодер, отримуючи на вхід друге речення, використовує механізм уваги, щоб зіставити частини обох речень

у процесі класифікації. Це дозволяє моделі враховувати структурні відповідності та семантичні зсуви між кандидатами на перифраз.

Такий підхід, наприклад, використовується у роботі [29], у якій К. Чо з співавторами використали кодувально-декодувальну архітектуру на основі рекурентних нейронних мереж, яка була інтегрована в класичну фразову систему статистичного машинного перекладу для обчислення оцінки пар фраз з метою покращення якості репрезентації фразових ознак.

Запропонований метод передбачав попереднє обчислення подібності між парами фраз у вихідному та цільовому реченнях на основі прихованих станів RNN кодувальника. Кожна пара фраз у системі оцінювалась за допомогою спеціального механізму оцінок, який дозволяв автоматично встановлювати, наскільки семантично сумісними є ці фрази. Таке вирівнювання з використанням RNN моделей забезпечило гнучке представлення фраз, здатне враховувати не лише локальний контекст, але й семантичну сумісність у межах повного речення.

Основною проблемою композиційної дистрибутивної семантики на рівні речень є забезпечення адекватної репрезентації як внутрішньої структури речення, так і взаємозв’язків між реченнями у межах певної семантичної задачі. Традиційні підходи глибинного навчання часто опрацьовують речення ізольовано, що призводить до втрати інформації про їхню взаємодію, особливо коли йдеться про складні чи латентні семантичні відповідності.

Одним із перших підходів до розв’язання цієї проблеми була модель Skip-Thought Vectors, яка використовує ідею методу Skip-gram моделі Word2vec, масштабуючи його на рівень речень у тексті [30]. Архітектура складається з кодувальника та декодера і використовує керовані рекурентні одиниці, причому кодувальник формує векторне представлення речення, а декодер намагається реконструювати сусідні речення у тексті. Таким чином, кожне речення кодується в єдиний вектор фіксованої довжини, що відображає не лише його внутрішню структуру, але й взаємозв’язки з навколишніми реченнями. Цей підхід не вимагає ручного конструювання ознак, і показав конкурентні результати у задачах ідентифікації перифраз, однак має обмеження у здатності репрезентувати

складні семантичні зв’язки, оскільки покладається на загальний контекст речення, без глибокого аналізу внутрішньої структури.

Альтернативні підходи відмовляються від використання рекурентних структур на користь згорткових нейронних мереж, які дозволяють моделювати локальні шаблони та фіксувати семантично релевантні фрагменти. На відміну від RNN, які працюють з вхідними даними послідовно, CNN дозволяють обробляти весь вхід паралельно, використовуючи фільтри згортки, що сканують текст та виявляють релевантні фрагменти. Одним з основних переваг CNN у задачі ідентифікації перифраз є здатність виявляти ключові семантичні фрагменти незалежно від їхньої позиції у реченні, що особливо корисно при пошуку перифраз, які можуть мати різну синтаксичну структуру, але однакове значення.

Наприклад, у моделі ARC-I, CNN використовуються для отримання векторного представлення кожного з речень, після чого отримані представлення передаються в багатошаровий персептрон для класифікації. Обмеженням моделі є те, що взаємодія між реченнями відбувається лише на фінальному етапі, коли репрезентації вже сформовані. Це було вирішено в моделі ARC-II, яка впроваджує двовимірну згортку безпосередньо над парою речень. Такий підхід дозволяє моделі виявляти взаємозалежності між словами й фразами з обох речень ще до їх остаточної агрегації, що значно покращує точність виявлення перифраз [31].

Розвитком використання нейромережевих архітектур при розв’язанні задачі ідентифікації перифраз стало впровадження механізмів уваги. У класичних нейронних мережах, таких як RNN або CNN, вся вхідна інформація обробляється однаково, незалежно від її значущості в конкретному контексті. Механізм уваги, навпаки, дозволяє моделі динамічно зважувати внесок кожного елемента послідовності залежно від задачі. Це означає, що моделі можуть приділяти більше уваги тим словам або фразам, які є семантично важливими для встановлення схожості між двома реченнями. Наприклад, в одному з досліджень механізм уваги був реалізований за допомогою формування матриць уваги як додаткових карт ознак, які додаються до вхідного шару згортки [32]. Перша така

матриця формується на основі порівняння репрезентацій двох речень і додається до вхідних ознак для фокусування на відповідних фрагментах. Друга матриця уваги формується після згортки для покращення результатів агрегації. Цей підхід дозволив враховувати взаємний вплив речень при побудові згорткових репрезентацій, орієнтуючи моделі на «парні» ознаки, що є ключовими для розпізнавання перифраз.

Попри суттєвий розвиток методів композиційної дистрибутивної семантики на рівні речень, більшість ранніх моделей на рівні речень все ще мали обмеження у точному вирівнюванні елементів між реченнями, що є критично важливим для задач ідентифікації перифраз. Це зумовило розвиток подальших досліджень з переходом до трансформерних моделей, які завдяки своїй архітектурі та використанню механізмів багатоголової уваги дозволяють моделювати взаємодії між усіма елементами речень.

Аналіз методів ідентифікації перифраз показує поступовий перехід від простих евристичних підходів та поверхневих векторних моделей до складних нейронних архітектур, здатних опрацьовувати синтаксичну, фразову та семантичну структуру речень. Водночас більшість розглянутих підходів мають суттєві обмеження щодо повноти представлення значення речення, зокрема:

1. Моделі на основі вкладень слів демонструють ефективність у базових випадках, однак ігнорують синтаксичні зв’язки й відповідно не можуть захопити контекст вищого рівня та не розпізнають структуру речення.
2. Моделі на рівні фраз на основі рекурсивних автокодувальників чи RNN-кодувальників покращують врахування синтаксичної структури, однак залежать від якісного парсингу та мають труднощі зі складними конструкціями.
3. Моделі на рівні речень вже фокусуються на глобальних семантичних зв’язках, однак вирівнювання елементів між реченнями часто реалізується після отримання повних репрезентацій, що призводить до втрати важливої локальної семантики. Механізми уваги зменшують цю проблему, однак обмежені в ідентифікації точних семантичних ролей елементів.

У зв’язку з вищезгаданим, обґрунтованим виглядає використання гібридного підходу до задачі ідентифікації перифраз, який поєднує сильні сторони статистичних моделей та нейронних мереж. Такий підхід дозволяє не лише враховувати лексичні та контекстні подібності, але й відображати структурні та реляційні залежності між елементами висловлення. Особливої актуальності набувають моделі, здатні формувати репрезентацію семантичних зв’язків між словами та фразами у вигляді графових структур, де вузли відповідають концептам або лексемам, а ребра – семантичним чи синтаксичним відношенням між ними.

Сучасні трансформерні архітектури, зокрема моделі, орієнтовані на формування контекстних вкладень, створюють передумови для автоматичного отримання таких структур, що робить можливим перехід від поверхневих текстових представлень до більш глибинних семантичних моделей. Саме інтеграція графових репрезентацій із механізмами глибинного навчання відкриває перспективи більш точного встановлення еквівалентності висловлень, особливо у випадках, коли перефразування супроводжується суттєвими синтаксичними трансформаціями.

1.3. Аналітичний огляд підходів до побудови семантичних репрезентацій у задачі ідентифікації перифраз

У межах когнітивної інформатики моделювання процесу розуміння тексту природної мови потребує репрезентації, що відображає як структурну, так і семантичну складність висловлювань. У сучасній когнітивно орієнтованій лінгвістиці та штучному інтелекті графові структури виступають потужним інструментом для репрезентації семантичної інформації, закладеної у текстах природної мови. Графова модель дозволяє явно виражати взаємозв’язки між поняттями, подіями та їх учасниками, що наближає обчислювальні підходи до способу обробки інформації людським мисленням. У контексті моделювання когнітивного процесу розуміння, графова семантика виконує роль

формалізованої структури знань, що забезпечує більш точне моделювання смислових залежностей.

Семантичний аналіз тексту за допомогою графів передбачає побудову орієнтованого або неорієнтованого графа, вузли якого відповідають лексемам, концептам або семантичним одиницям, а ребра – смисловим відношенням між ними [33]. До найбільш поширених типів графів, що застосовуються у завданнях NLP, належать:

1. Графи знань – це структуровані представлення енциклопедичної інформації, які моделюють об’єкти реального світу та їхні взаємозв’язки. Класичними прикладами є DBpedia, Wikidata, ConceptNet, Freebase. Вони використовуються для підкріплення текстового розуміння зовнішніми джерелами знань, забезпечуючи контекстуалізацію значень слів та уточнення смислових зв’язків у висловлюваннях [34]. У графах знань вершини відповідають сутностям, а ребра – відношенням між ними;

2. Графи залежностей, що формуються на основі синтаксичного аналізу речення та моделюють відношення залежностей між словами. Хоча першочергово вони відображають синтаксис, у багатьох випадках вони мають також семантичну релевантність. Зокрема, предикатно-аргументна структура речення, яка є основою розуміння смислу, відображається у вигляді спрямованих дуг між лексемами та залежними словами. Графи залежностей часто використовуються як основа для побудови складних семантичних графів.

3. Графи концептів – це семантичні структури, що моделюють ментальні репрезентації подій, об’єктів і ситуацій. Запропоновані Джоном Совою, ці графові структури ґрунтуються на когнітивних формалізмах, зокрема теорії фреймів, відповідно до якої кожна подія або дія описується у вигляді фрейму, вузли якого відображають учасників, ролі та контекстуальні параметри [35];

4. Графи сенсів є формалізованими структурами для представлення лексико-семантичних зв’язків між мовними одиницями. Вони моделюють лексичну багатозначність слів і синонімічні, антонімічні, гіпонімічні та інші

відношення, що формують семантичну організацію лексикону. Такі графові структури дозволяють ефективно виконувати пошук релевантних значень у контексті, а також побудову когнітивних моделей значення. У контексті задачі ідентифікації перифраз графи сенсів використовуються для виявлення спільних або еквівалентних значень між лексичними одиницями різних висловлювань, що критично важливо для оцінки семантичної подібності.

У ранніх дослідженнях ідентифікації перифраз синтаксичні структури речень будувалися на основі великих анотованих корпусів, зокрема Penn Treebank [36]. Цей корпус містить дві ключові форми лінгвістичної розмітки: розмітка частин мови та повний синтаксичний розбір речення із представленням його структури у вигляді вкладених дужкових конструкцій. Важливою рисою Penn Treebank є детальна розмітка фразової структури, що передбачає чітке визначення синтаксичних ролей та типів фраз, що забезпечує багаторівневу структурну репрезентацію. Приклад анотації речення з використанням фразових категорій та розміткою частин мови згідно зі Penn Treebank наведено на рисунку 1.7.

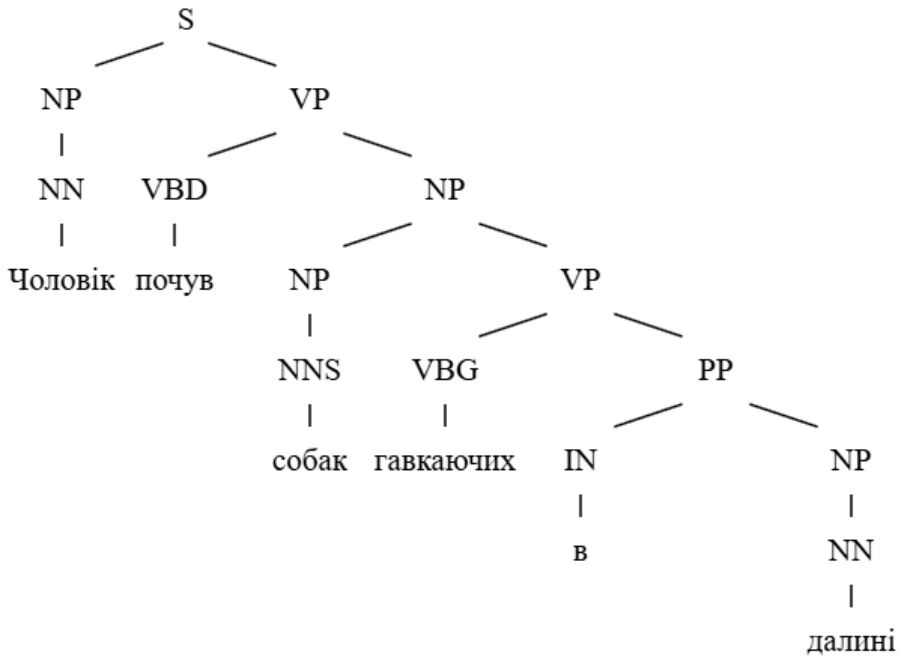

Рис. 1.7. Ієрархічна фразова репрезентація синтаксичної структури в Penn Treebank

Надалі розвиток синтаксично орієнтованих підходів до ідентифікації перифраз привів до поширення дерев залежностей як альтернативної моделі представлення структури речення. На відміну від фразових дерев, які описують ієрархію синтаксичних груп, дерева залежностей фокусуються на прямому встановленні граматичних відношень між окремими словами, що дозволяє явно відображати їхню синтаксичну та частково семантичну роль у висловленні. Важливим кроком у формалізації цього підходу стало створення стандарту анотування Universal Dependencies, який забезпечує уніфіковану систему позначення типів залежностей та єдиний набір морфосинтаксичних тегів для великої кількості мов. Приклад дерева залежностей речення наведено на рисунку 1.8.

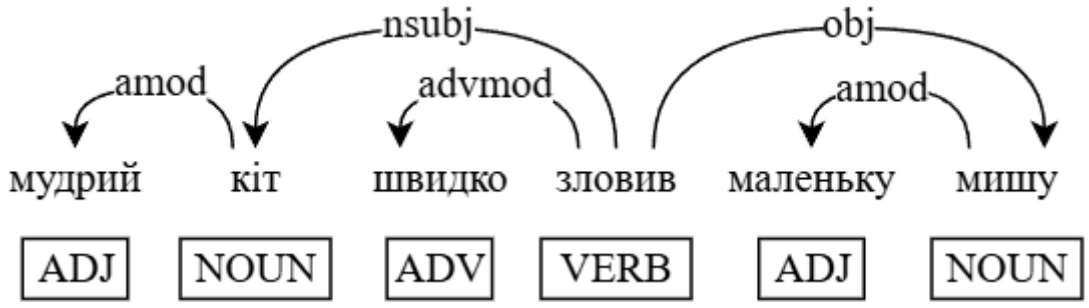

Рис. 1.8. Дерево синтаксичних залежностей речення у форматі Universal Dependencies

Як показано у роботі Р. Бунеску та Р. Муні [37], використання дерев залежностей дає змогу краще виявляти перифрази, що виникають внаслідок синтаксичної трансформації. Оскільки дерево залежностей прямо відображає рольову організацію висловлення, моделі, що його використовують, здатні фіксувати відповідності між учасниками ситуації незалежно від поверхневої форми речення. Це особливо важливо у випадках, коли семантична еквівалентність не супроводжується лексичною подібністю.

Одним з сучасних підходів до побудови представлення змісту речення є використання формалізму AMR [38]. AMR моделює глибинну семантику висловлення у вигляді графа, структура якого не залежить від поверхневої

синтаксичної форми, зокрема порядку слів або типу граматичної конструкції (рис. 1.9.). На відміну від традиційних дерев синтаксичного аналізу, де основним об’єктом опису виступають граматичні відношення між складниками речення, формалізм AMR зосереджений на семантичній абстракції, представляючи висловлювання у вигляді зв’язного орієнтованого графа, у якому вузли відповідають концептам, а ребра – відношенням між ними, включно з предикатно-аргументними зв’язками. Центральною особливістю AMR є використання рольової семантики PropBank, що дозволяє моделювати учасників ситуації не через поверхневі словоформи, а через функціональні ролі в події, що значною мірою підвищує стабільність репрезентації за умов синтаксичної варіативності.

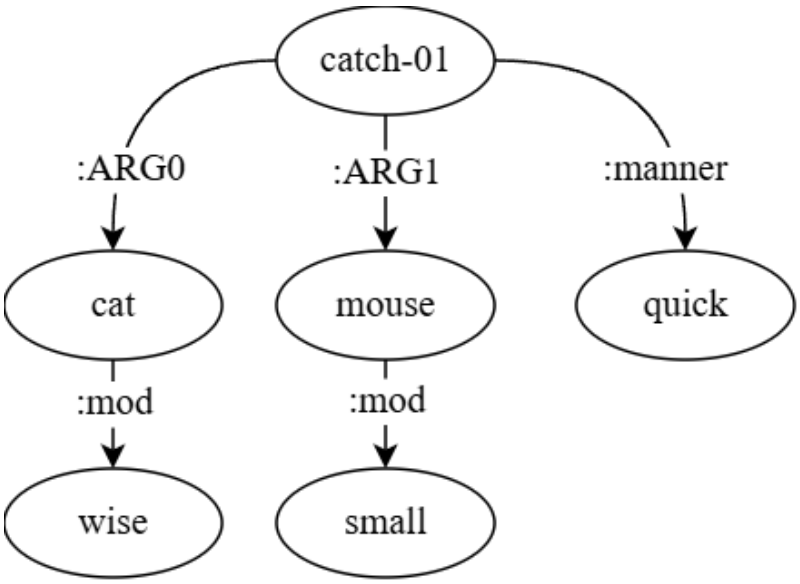

Рис. 1.9. Семантична структура речення у вигляді графа AMR

Перевага AMR проявляється насамперед у здатності зіставляти речення з істотно різними синтаксичними структурами, але зі спільним змістовим ядром. Таким чином, AMR уможливлює виявлення перифраз, які не можуть бути ідентифіковані за допомогою лексичних чи синтаксичних моделей, орієнтованих на поверхневу форму тексту. Сучасні підходи до генерації таких представлень ґрунтуються переважно на нейромережевих моделях типу Seq2seq та

Transformer, які кодують речення та генерують його графове представлення. Серед найвідоміших рішень варто відзначити SPRING і AMRBART, що демонструють високу точність генерації AMR графів для англійської мови.

Проведений огляд демонструє, що лексичні та синтаксичні моделі, хоч і забезпечують важливу основу для виявлення подібності між висловлюваннями, залишаються чутливими до поверхневих трансформацій речення: зміни порядку слів, варіативності морфологічних форм, різних типів синтаксичних конструкцій або стилістичних перефразувань. Моделі залежностей та фразових структур дозволяють частково подолати ці обмеження шляхом врахування граматичної організації висловлення, однак вони здебільшого продовжують відображати структурні аспекти речення, а не його концептуальний зміст.

На відміну від них, AMR репрезентації спрямовані на моделювання глибинної семантики. Представлення речень у вигляді графів концептів та ролей дозволяє виділити центральні ситуаційні відношення та учасників події, зменшуючи залежність від формально-граматичних варіацій. Це робить AMR особливо ефективним у завданнях, де перифразування реалізується через логіко-семантичні перетворення, а не через просту заміну слів.

Крім того, сучасні нейромережеві AMR парсери та моделі багатомовної генерації графових репрезентацій забезпечують достатній рівень автоматизації побудови AMR, що створює можливість їх практичного застосування в масштабних NLP-завданнях, включно з ідентифікацією перифраз. Додатково, вагомою перевагою AMR є можливість багатомовного парсингу, що забезпечується наявністю узагальненого концептуального простору, спільного для різних мов. Це означає, що незалежно від поверхневих морфологічних та синтаксичних розбіжностей між мовами, AMR графи відображають універсальну семантичну структуру висловлювання. Для задачі ідентифікації перифраз українських текстів це є особливо важливим, оскільки українська мова характеризується вільним порядком слів та значною кількістю синтаксичних варіантів вираження однакового змісту. Традиційні лексико-синтаксичні методи

у таких умовах часто демонструють нестабільність результатів через залежність від поверхневих форм.

1.4. Обґрунтування напрямку дослідження на основі аналізу актуальних проблем та постановка завдань дослідження

Проведений аналіз методів моделювання когнітивних процесів у підрозділі 1.1, підходів до ідентифікації перифраз у підрозділі 1.2, а також графових моделей семантичного представлення висловлень у підрозділі 1.3 засвідчує, що попри значний прогрес у галузі обробки природної мови, проблема формального встановлення семантичної еквівалентності між висловленнями залишається нерозв’язаною. Особливо складною вона постає для мов із високим ступенем морфологічної варіативності та багатим набором синтаксичних конструкцій, якою є українська.

Традиційні корпусні та підходи на основі знань здебільшого спираються на лексичні та словотвірні відповідності, що дозволяє розпізнавати поверхневі перифрази, основані на заміні синонімів чи варіаціях словоформ. Проте вони залишаються малоефективними у випадках глибших трансформацій. У таких випадках поверхневі лексичні ознаки більше не відображають подібності значення.

Сучасні нейромережеві моделі дистрибутивної семантики та контекстних вкладень, навпаки, здатні узагальнювати знання про смислові зв’язки на основі статистичних закономірностей уживання в корпусах. Однак їх функціонування є переважно непрозорим: вони не надають пояснення того, чому два висловлення вважаються семантично тотожними. Це ускладнює застосування таких моделей у контексті когнітивного моделювання, де важливо не лише визначити факт еквівалентності, але й відтворити або інтерпретувати механізм встановлення смислової відповідності, подібно до того, як це відбувається у свідомості людини.

Застосування графових семантичних репрезентацій, зокрема AMR, частково долає зазначені недоліки, оскільки дозволяє відокремити концептуальну структуру висловлення від його конкретної мовної реалізації. AMR формалізм формує єдину модель опису події: він визначає головну дію, її учасників, їхні ролі, особливості протікання дії, причинно-наслідкові зв’язки та інші складові змісту речення. Це створює можливість порівнювати висловлення на рівні глибинної семантичної організації, а не поверхневих мовних форм.

Проте наявні засоби побудови AMR графів, як показує аналіз досліджень, мають низку суттєвих обмежень, що перешкоджають їх застосуванню в задачі ідентифікації перифраз для української мови:

1. Багатомовний AMR парсинг залишається недостатньо розвиненим. Більшість доступних AMR парсерів оптимізовані під англійську й використовують її синтаксичні закономірності як базові.

2. Брак корпусів українських AMR розміток. Наявні ресурси є фрагментарними або експериментальними, що ускладнює навчання моделей без перекладу в англійську.

3. Недостатня формальна визначеність критеріїв порівняння AMR графів. Наявні метрики, зокрема Smatch та її модифікації, оцінюють переважно структурну подібність графів, проте не враховують когнітивну значущість виявлених розбіжностей. Зокрема, вони не дозволяють розрізняти критичні семантичні зміни та другорядні трансформації у змісті висловлення.

Таким чином, актуальною науковою проблемою є розроблення підходу, який:

– забезпечує отримання якісних семантичних графових репрезентацій для української мови в умовах багатомовного моделювання;

– дає можливість порівнювати AMR графи для встановлення ступеня семантичної еквівалентності;

– відтворює механізми людської здатності розпізнавати перифрази.

З урахуванням аналізу літературних джерел та виявлених обмежень наявних підходів, у межах даного дослідження необхідно розв’язати наступні взаємопов’язані науково-практичні задачі, що спрямовані на побудову когнітивно обґрунтованої інтелектуальної системи ідентифікації перифраз:

1. Проаналізувати та формалізувати когнітивні передумови процесу розуміння природномовних висловлень, що мають значення для встановлення семантичної еквівалентності між ними, з метою визначення вимог до семантичних репрезентацій і механізмів їх порівняння.
2. Проаналізувати наявні підходи до генерації абстрактних семантичних репрезентацій текстів природної мови.
3. Розробити метод генерації абстрактних семантичних репрезентацій для текстів природної мови, орієнтований на багатомовну обробку та збереження семантичної повноти текстів.
4. Розробити метод визначення семантичної еквівалентності між графовими семантичними репрезентаціями текстів природної мови.
5. Розробити архітектуру інтелектуальної системи моделювання когнітивного процесу розуміння шляхом розв’язання задачі ідентифікації перифраз у текстах природної мови.
6. Визначити структуру, функціональні модулі та алгоритм роботи інтелектуальної системи ідентифікації перифраз, включно з модулями підготовки текстових даних, побудови AMR графів та класифікації.
7. Розробити інформаційну технологію багатомовної підготовки текстів та побудови AMR графів.
8. Провести експериментальну перевірку ефективності розробленої технології.

1.5. Висновки за розділом 1

Узагальнюючи викладений у розділі матеріал, можна зробити такі висновки:

1. Когніцією або пізнанням називається сукупність процесів отримання, обробки та інтеграції інформації, що забезпечують відображення об’єктивної дійсності у свідомості людини та лежать в основі її розумової діяльності. Проблема моделювання когнітивних процесів полягає у формалізації цих складних розумових функцій для створення обчислювальних моделей, здатних відтворювати людське пізнання. Це ускладнюється потребою врахувати різні рівні когнітивних процесів, типи знань, культурно-мовний контекст та механізми взаємодії нижчих і вищих розумових функцій.

2. Дослідження, що стосуються моделювання когнітивних процесів, розглядають когнітивне пізнання як багаторівневий процес перетворення, інтерпретації та інтеграції інформації, який у сучасних інтелектуальних системах набуває форми формалізованих обчислювальних моделей. Виконаний теоретико-методологічний аналіз засвідчив еволюцію підходів від описових психологічних концепцій до формально визначених моделей, орієнтованих на машинну обробку знань та автоматизовану інтерпретацію смислу.

3. Когнітивний процес розуміння визначено як центральний компонент як людського мислення, так і штучних систем обробки природної мови. Показано, що здатність до встановлення смислової еквівалентності між різними мовними формами є ключовою характеристикою цього процесу, що безпосередньо зумовлює актуальність задачі ідентифікації перифраз у контексті когнітивного моделювання.

4. Аналітичний огляд методів ідентифікації перифраз дозволив встановити принципові відмінності між традиційними підходами на основі корпусів і знань та методами, що базуються на глибокому навчанні й дистрибутивній семантиці. З’ясовано, що традиційні методи характеризуються високим рівнем інтерпретованості, проте обмежені у здатності до узагальнення, тоді як нейромережеві моделі демонструють високу емпіричну ефективність, однак не забезпечують прозорості механізмів прийняття рішень щодо семантичної подібності.

5. Розгляд підходів до побудови семантичних репрезентацій показав, що графові моделі, зокрема формалізм AMR, мають суттєві переваги для задач ідентифікації перифраз, оскільки дозволяють відокремити поверхневу мовну форму від глибинної концептуальної структури висловлення та здійснювати порівняння на рівні смислових відношень і ролей.

6. Виконаний аналіз актуальних проблем засвідчив, що безпосереднє застосування AMR підходу для української мови ускладнене низкою факторів, зокрема: обмеженою доступністю багатомовних AMR парсерів, відсутністю масштабних корпусів розмічених даних, а також недосконалістю наявних метрик порівняння семантичних графів. Це зумовлює необхідність адаптації наявних методів та розроблення спеціалізованих інструментів, здатних адекватно враховувати морфологічні та синтаксичні особливості української мови.

7. Таким чином, невирішеною науковою проблемою є створення інтерпретованої інтелектуальної системи ідентифікації перифраз, яка б поєднувала когнітивно обґрунтовані семантичні графові репрезентації із перевагами сучасних методів машинного навчання для аналізу структурної та концептуальної подібності висловлень.

8. З урахуванням викладеного, надалі необхідно розв’язати наступні задачі дослідження:

– проаналізувати та формалізувати когнітивні передумови процесу розуміння природномовних висловлень, що мають значення для встановлення семантичної еквівалентності між висловленнями, з метою визначення вимог до семантичних репрезентацій і механізмів їх порівняння;

– проаналізувати наявні підходи до генерації абстрактних семантичних репрезентацій текстів природної мови;

– розробити метод генерації абстрактних семантичних репрезентацій для текстів природної мови, орієнтований на багатомовну обробку та збереження семантичної повноти текстів;

– розробити метод визначення семантичної еквівалентності між графовими семантичними репрезентаціями текстів природної мови;

– розробити архітектуру інтелектуальної системи моделювання когнітивного процесу розуміння шляхом розв’язання задачі ідентифікації перифраз у текстах природної мови;

– визначити структуру, функціональні модулі та алгоритм роботи інтелектуальної системи ідентифікації перифраз, включно з модулями підготовки текстових даних, побудови AMR графів та класифікації;

– провести експериментальну перевірку ефективності розробленої технології;

РОЗДІЛ 2
МЕТОДИ ІДЕНТИФІКАЦІЇ ПЕРИФРАЗ НА ОСНОВІ ГРАФОВИХ СЕМАНТИЧНИХ РЕПРЕЗЕНТАЦІЙ

2.1. Когнітивна модель процесу розуміння як механізму зіставлення смислових репрезентацій

У когнітивній інформатиці розум людини розглядається як сутність у якій когнітивні процеси взаємодіють між собою на різних рівнях пам’яті, а саме сенсорна буферна пам’ять, короткострокова пам’ять. довгострокова пам’ять та буферна пам’ять дій [39]. Загальну схему взаємодії цих рівнів пам’яті та когнітивних процесів обробки інформації наведено на рис. 2.1.

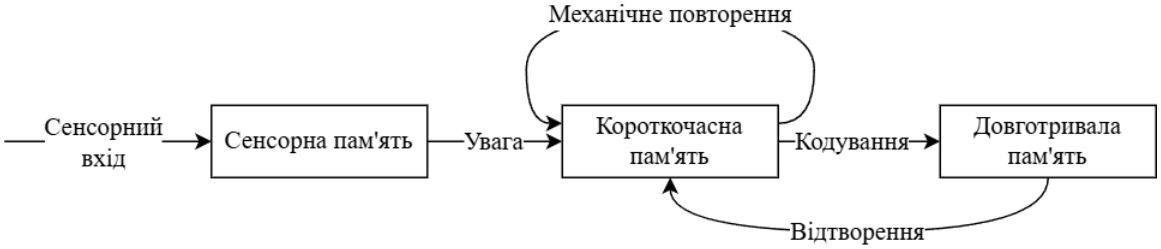

Рис. 2.1. Схема взаємодії рівнів пам’яті та когнітивних процесів обробки інформації

Задача когнітивного процесу розуміння полягає в пошуку зв’язків між отриманим об’єктом або атрибутом та відповідними об’єктами або атрибутами, які відомі та зберігаються на рівні довгострокової пам’яті (LTM) [3]. Важливо підкреслити, що процес розуміння не зводиться до розпізнавання мовних форм. Він передбачає встановлення відповідності між внутрішньою смисловою структурою отриманого висловлення та вже наявними структурами знань у довготривалій пам’яті. Таким чином, смисл висловлення не є простою послідовністю слів, а являє собою модель ситуації, представлену у вигляді взаємопов’язаних концептів.

Процес розуміння передбачає побудову внутрішнього уявлення, використовуючи попередні знання та вимагає попередньої обробки отриманої інформації іншими когнітивними процесами нижчого рівня згідно з ієрархією LRMB: пошук, запам’ятовування та репрезентація знань. Результатом обробки вищезгаданих процесів є основна одиниця процесу розуміння в когнітивній інформатиці, відома як поняття або концепт [40].

Концепт – це абстрактна структура що несе в собі певне значення необхідне для ідентифікації конкретної сутності реального світу та абстрактного суб’єкта сприйманого світу. Визначення концепту відбувається за допомогою змісту концепту та розширення концепту. Змістом концепту називають сукупність ознак, характеристик або властивостей, які визначають його сутність і відрізняють від інших концептів. Це внутрішня структура концепту, що описує, які риси повинні бути притаманні об’єктам, що належать до нього. Розширенням концепту своєю чергою називають множину всіх об’єктів або явищ, які підпадають під даний концепт і відповідають його змісту. Це зовнішній обсяг концепту, тобто все, що він охоплює в реальному світі [41].

Таким чином, при моделюванні процесу розуміння, знання можуть бути представлені у вигляді концептів, а сам процес може бути відтворений шляхом моделювання їх взаємодії один з одним, їх ієрархічної структури та механізмів обробки інформації як у висхідному, так і в низхідному напрямках. Такий підхід дозволяє розбити комплексний процес розуміння до одиниць мислення, які можна аналізувати та використовувати при моделюванні інтелектуальних інформаційних систем.

Для формалізації цих структур у когнітивній інформатиці широко використовується модель Object-Attribute-Relation (OAR), яка описує знання у вигляді трійок «об’єкт – його характеристики – зв’язки з іншими об’єктами» (рис. 2.2.). Дана модель дозволяє представити зміст висловлення у вигляді структурованої мережі концептів, що робить можливим порівняння, узгодження та інтеграцію нових смислів із наявними знаннями довготривалої пам’яті. Таким

чином процес розуміння може бути концептуально змодельований наступними кроками:

– пошук та встановлення відповідності між сутністю реального світу та віртуальною сутністю та наявними об’єктами та атрибутами;

– побудова часткової або повної моделі OAR для оброблюваної сутності;

– класифікація та кластеризація отриманого представлення відповідно до повної моделі OAR на рівні LTM пам’яті;

– оновлення нової OAR моделі та її зв’язків у LTM.

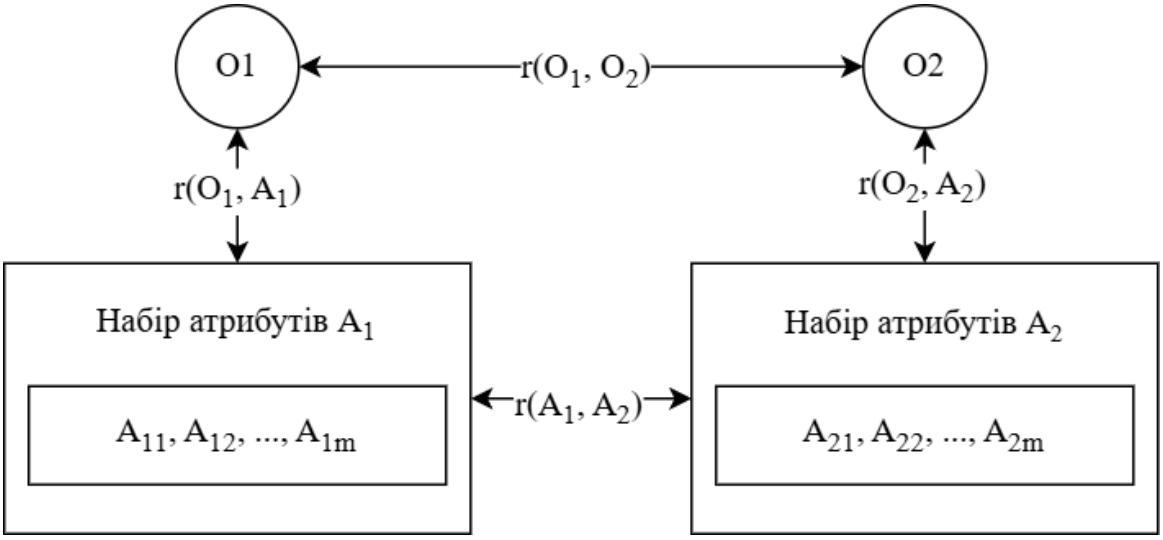

Рис. 2.2. Концептуальна схема OAR моделі представлення знань у когнітивній інформатиці

Першим кроком процесу розуміння є ідентифікація вхідного об’єкта, представленим у вигляді віртуальної сутності або концепту. Наприклад для визначення слова чи терміну, що має декілька значень, ініціюється процес розуміння значення цього поняття. Вхідними об’єктами у цьому випадку будуть слово або термін, що потребує ідентифікації.

Після ідентифікації об’єкта, відбувається пошук можливих зв’язків між наявними об’єктами з моделі OAR у пам’яті та вхідним об’єктом. Спочатку він аналізує інформацію, використовуючи знання, що зберігаються у пам’яті.

Наступним етапом є виявлення атрибутів і зв’язків, за допомогою яких формується модель OAR для об’єкта. Чим більше було знайдено пов’язаних об’єктів у пам’яті LTM, тим легше відбувається розуміння. На фінальному етапі перевіряється повнота та адекватність отриманих результатів для створення часткової моделі sOAR. Якщо отримано достатньо даних, формується повноцінна модель OAR після чого вона інтегрується у відповідний кластер знань у загальній структурі. Лише після цього процес розуміння завершується.

У випадку отримання неповної інформації про об’єкт, формується часткова модель sOAR, і виникає необхідність пошуку додаткової інформації з зовнішніх джерел. Цей крок повторюється доки не буде знайдено повної інформації щодо атрибутів об’єкта та їх зв’язків. У випадку вдалого пошуку повторюються попередні кроки, після чого в OAR модель інтегрується сформоване представлення про об’єкт. У випадку, коли додатковий пошук не дав результатів – в пам’яті зберігається неповна сформована OAR модель об’єкта, що може використовуватися як ідентифікатор невідомого концепта при подальшому розумінні, а результатом процесу вважається нерозуміння концепту. Це відповідає поведінці, коли невідоме слово запам’ятовується без визначення його сенсу. При цьому зберігається додаткова інформація про слово така як кількість літер, з якої букви воно починається або які інші слова воно може нагадувати.

Завершальним етапом процесу розуміння є формування оновлених моделей OAR у LTM, що означає остаточне завершення процесу. Ідеальним результатом процесу розуміння є формування повних та адекватних співвідношень об’єкта, що аналізується, з наявними моделями OAR, що означає повне розуміння концепту, або ситуацію коли це розуміння майже досягнуто. Іншим можливим результатом є часткове розуміння, дуже низький рівень розуміння або фактична відсутність розуміння, коли будується часткова модель OAR. У перших двох випадках часткова модель будується як субмодель, де не знайдено достатніх зв’язків. В останньому випадку для абсолютно нового концепту результатом пошуку є нерозуміння, коли створюється лише

ідентифікатор, що відповідає концепту, без будь-яких відомих зв’язків із наявними знаннями. Таким чином в когнітивній інформатиці розуміння не є задачею бінарної класифікації. Воно може варіюватися від 0%, що відповідає повній відсутності розуміння, до 100%, що відповідає повному розумінню.

Таким чином, когнітивний процес розуміння можна розглядати як механізм зіставлення нової смислової структури з уже наявними структурами знань у довготривалій пам’яті. Важливо зазначити, що у цьому підході смисл не визначається послідовністю слів або поверхневою граматичною формою. Він описується як структура ситуації, тобто взаємопов’язана мережа концептів та їх відношень. Відповідно, два висловлення можуть мати різну синтаксичну організацію, використовувати різні лексичні засоби, але при цьому описувати одну й ту саму ситуацію, зберігаючи спільну семантичну основу. Саме ця властивість лежить в основі перифраз.

Отже, ідентифікація перифраз є окремим випадком когнітивного процесу розуміння, оскільки вона передбачає визначення того, чи відображають два мовні вирази одну й ту саму смислову структуру, попри можливі відмінності поверхневої форми. Це дозволяє перейти до формалізації задачі в термінах семантичної еквівалентності.

2.2. Формалізація задачі ідентифікації перифраз як задачі класифікації семантичної еквівалентності

Аналіз лінгвістичних і обчислювальних підходів до вивчення перифраз показує, що не існує єдиного, усталеного визначення поняття «перифраза». Різні дослідження надають перевагу різним критеріям: від повної семантичної еквівалентності до приблизної смислової подібності або навіть лише функціональної взаємозамінності в межах конкретного контексту. Така невизначеність зумовлює складність формального опису перифраз, особливо в автоматизованих системах обробки природної мови.

У зв’язку з цим перифрази слід розглядати не як дискретне явище, що має чіткі межі, а як сукупність мовних трансформацій або шаблонів, що відображають типові способи вираження однакових або близьких за значенням смислів. Наприклад, Р. Бхагат та Е. Хові у своїй роботі [42] описують понад два десятки операцій перефразування – від лексичних замін до перебудови синтаксичних структур – що дозволяють формувати квазі перифрази, які передають схожі значення з використанням різних мовних засобів.

Серед найбільш згадуваних шаблонів перифраз можна виділити наступні:

1. Синоніми: заміна слів тексту на синоніми.
2. Стан: зміна стану речення з активного на пасивний та навпаки.
3. Зміна форми слова з однієї частини мови в іншу. Наприклад, зміна іменника на дієслово, прислівник або прикметник.
4. Розбиття складних речень на декілька простих.
5. Використання різних структур речень для вираження того самого значення.

У контексті моделювання когнітивних процесів, ідентифікація перифраз постає як фундаментальне завдання. Вона дозволяє простежити, яким чином нове текстове висловлення співвідноситься із вже наявними ментальними репрезентаціями, навіть коли лексичні та синтаксичні форми значно відрізняються.

Таким чином, задача аналізу та розпізнавання перифраз може розглядатися як емпіричний інструмент перевірки когнітивних моделей розуміння: якщо модель здатна встановлювати еквівалентність висловлень різної форми, вона відтворює ключові операції узагальнення, властиві людському розуму.

Формально процес ідентифікації перифраз розглядається як встановлення семантичної еквівалентності між двома мовними виразами. З цією метою висловлювання перетворюються у єдиний формальний простір для забезпечення можливості їх обробки комп’ютерними моделями Відповідно, процес перетворення формалізується як відображення виду:

$$\Phi : S \to M, \tag{2.1}$$

де S – множина висловлень природної мови, а M – простір семантичних представлень.

Основною одиницею смислу виступає поняття смислового ядра висловлення, що визначається як множина концептів та відношень між ними:

$$core(s) = \langle C, R \rangle, \tag{2.2}$$

де s – речення природної мови, C – множина концептів речення, R – множина семантичних відношень між концептами.

За умови ідеальної семантичної тотожності, два висловлення s_1 та s_2 вважаються семантично еквівалентними, якщо їх смислові ядра структурно узгоджені:

$$s_1 \equiv s_2 \Leftrightarrow core(s_1) \cong core(s_2), \tag{2.3}$$

де s_1 та s_2 – пара речень природної мови, core(s_1) та core(s_2) – смислові ядра відповідних речень визначені за формулою (2.2).

У прикладних задачах, де повний ізоморфізм може бути недосяжним через варіативність природної мови, задача ідентифікації перифраз формалізується як класифікація пар висловлень за критерієм семантичної подібності:

$$P(s_1, s_2) = [sim(core(s_1), core(s_2)) \geq \tau], \tag{2.4}$$

де τ – поріг прийняття рішення про еквівалентність.

Зазначена формалізація ідентифікації перифраз підкреслює необхідність точного та структурованого подання смислу висловлень. Для того, щоб автоматизовані системи могли аналізувати та порівнювати тексти на рівні

смислів, потрібна репрезентація знань, яка відображає не лише поверхневі лексичні та синтаксичні форми, а й концептуальні структури.

У цьому контексті репрезентація знань є ключовим етапом моделювання будь-якої інформаційної системи, оскільки вона визначає, як знання про об’єкти та їх відносини буде структуровано та збережено для подальшого аналізу та обробки. У когнітивній інформатиці при моделюванні когнітивного процесу розуміння для представлення знань використовується семантика концептів. У даному контексті семантика розглядається як сукупність значень, функцій і поведінкових моделей концептів, а також відношень між ними, які можуть бути отримані на основі наперед визначеного набору сутностей. Вона виходить за межі простої синтаксичної структури даних, щоб охопити глибинні знання та зв’язки, які є важливими для того, як людський розум сприймає та оброблює інформацію.

Отже, репрезентація знань у вигляді концептів і відношень між ними створює основу для формалізованого подання смислу текстів. Для автоматизованого відтворення когнітивних процесів розуміння необхідні методи, які дозволяють переводити природномовні висловлення у структуровані семантичні репрезентації. Одним із таких підходів є AMR, який забезпечує графове представлення змісту речення через концепти та їхні відношення.

2.3. Системний аналіз задачі ідентифікації перифраз

Задача ідентифікації перифраз полягає у встановлені семантичної еквівалентності між двома або більше мовними одиницями – словами, фразами, реченнями тощо. У цьому контексті, ідентифікація перифраз може використовуватися для аналізу здатності інтелектуальної системи до когнітивного узагальнення та семантичного аналізу. На відміну від простих лексичних і синтаксичних збігів, задача ідентифікації перифраз передбачає встановлення відповідності між висловлюваннями на рівні глибинної семантики, навіть за наявності суттєвих формальних відмінностей між ними.

Перифрази – це мовні конструкції, що передають однаковий концептуальний зміст, попри розбіжності в синтаксичній побудові чи стилістичному оформленні. При моделюванні когнітивного процесу розуміння їх можна розглядати як різні реалізації однієї й тієї самої ментальної репрезентації у свідомості. Саме здатність системи ідентифікувати такі репрезентації є індикатором її відповідності принципам когнітивного моделювання.

Під час обробки текстів природної мови, два схожі за формою висловлювання можуть мати відмінну семантику внаслідок полісемії чи культурних конотацій, а формально різні – виявлятися концептуально тотожними. Це створює виклик для побудови моделей, які здатні виходити за межі поверхневих структур і працювати на рівні концептів. У цьому контексті ефективна ідентифікація перифраз є не лише завданням зіставлення формулювань, а й інструментом моделювання ключових когнітивних процесів – таких як семантичне узагальнення, трансформація репрезентацій та реконструкція змісту.

На сьогодні для розв’язання задачі ідентифікації перифраз застосовуються чотири основні підходи: лексико-семантичний, статистичний, семантично-графовий та нейромережевий [43]. Лексико-семантичні методи базуються на встановленні синонімічних і семантично споріднених зв’язків, однак залежать від повноти ресурсів, що доступні для мови, таких як великі тезауруси, синонімічні словники, бази знань тощо. Статистичні підходи використовують спільне вживання мовних одиниць у великих текстових корпусах для виявлення зв’язків між ними та оцінки їх схожості.

Графові методи, зокрема на основі AMR, дозволяють формалізувати структуру значення незалежно від конкретної мовної реалізації. Такі графи моделюють взаємозв’язки між концептами, діями та учасниками ситуацій, що описуються у тексті, що дозволяє системі зіставляти висловлювання за їх концептуальним змістом. Завдяки структурованості графів, семантично-графові

методи забезпечують можливість моделювання когнітивних механізмів узагальнення, абстрагування та реконструкції значення.

Перевага нейромережевих моделей – у їхній здатності до узагальнення семантичних шаблонів на основі векторних представлень, отриманих шляхом навчання на великих корпусах тексту. Останнім часом, особливої уваги набуває інтеграція нейромережевих підходів із графовими семантичними моделями, зокрема трансформерних архітектур типу Graph2Seq або графових нейронних мереж (GNN) для генерації AMR графів. Таке поєднання дозволяє забезпечити високу точність, інтерпретованість та відповідність когнітивним уявленням про розуміння мови.

Таким чином, системне розв’язання задачі ідентифікації перифразів можливе на основі поєднання когнітивного підходу до моделювання розуміння, графових репрезентацій змісту та сучасних методів глибинного навчання. Це відкриває нові можливості для створення інтелектуальних систем, здатних до семантичного аналізу, когнітивного узагальнення та міжмовної інтерпретації.

З огляду на викладене, постає задача створення інтелектуальної системи, здатної виявляти перифрази українською мовою шляхом аналізу їхніх AMR репрезентацій. Основна мета цієї задачі полягає у побудові моделі, яка на основі AMR графів двох речень визначає, чи виражають вони одне й те саме значення, навіть якщо їх синтаксична або лексична форма відрізняється.

У рамках цієї задачі передбачається:

– створення або адаптація засобів генерації AMR графів для української мови;

– визначення формальних критеріїв семантичної подібності між AMR графами;

– реалізація методу класифікації пар речень на основі їх графових представлень;

– дослідження ефективності такого підходу як когнітивної моделі семантичного узагальнення.

Оскільки для української мови AMR парсери майже не представлені, розв’язання задачі передбачає подолання низки лінгвістичних та технічних викликів, зокрема пов’язаних з побудовою корпусів, адаптацією моделей та формалізацією порівняльних механізмів.

2.4. Підходи генерації абстрактних семантичних репрезентацій для текстів природної мови

Серед сучасних семантичних формалізмів AMR вирізняється здатністю узагальнено відображати зміст цілого речення у вигляді єдиної графової структури. На відміну від формально-логічних чи розподільчих моделей, AMR орієнтований на семантичну абстракцію, що дозволяє уніфіковувати синтаксичну варіативність [44].

AMR репрезентації моделюють рольові семантичні відношення між концептами, що відображають зміст речення незалежно від його синтаксичної форми. У цьому вигляді кожне речення представлено як спрямований, ациклічний граф, у якому вершини відповідають концептам, а ребра репрезентують відношення між ними.

Для анотації основних елементів AMR використовують формалізм PropBank, що представлений у вигляді дерев синтаксичного розбору з розміткою частин мови та фразової структури [45]. PropBank анотує речення структурами предикат-аргумент, вказуючи, які учасники речення виконують певні ролі стосовно предиката. Ці ролі базуються на ідеї тематичних ролей та поділяються на наступні категорії:

- агент – той, хто виконує дію;
- пацієнт – об’єкт дії;
- локація – місце, де відбувається дія;
- інструмент, ціль тощо.

Використання анотацій дозволяє узагальнити різноманітні синтаксичні реалізації одного й того самого змісту та охопити широкий спектр семантичних та синтаксичних ознак, зокрема:

1. Базові семантичні відношення – AMR включає набір уніфікованих відношень, що описують часові, просторові, причинні, модальні та інші логіко-семантичні зв’язки між концептами, подіями й атрибутами.

2. Кореференція – повторне використання змінних дозволяє моделювати кореферентні зв’язки, зберігаючи ідентичність сутностей у межах графа.

3. Зворотні відношення – спеціальні форми семантичних зв’язків, які дозволяють побудувати граф з фокусом не на дії, а на її учасниках або об’єктах, змінюючи напрямок семантичного зв’язку.

4. Частини речення, зокрема дієслівні та іменникові конструкції, прикметники та прийменники.

5. Зв’язкові конструкції – зв’язки типу «є чимось» моделюються через окремі відношення, що вказують на сутність, якій приписується певна властивість або категорія.

Таким чином, завдяки підходу до репрезентації семантики текстових конструкцій, AMR графи відповідають ментальним репрезентаціям, які виникають під час інтерпретації тексту: вони структурують значення у вигляді мережі концептів і зв’язків, що дозволяє реалізувати обробку тексту як процес логічного виведення над репрезентаціями. Такий підхід узгоджується з принципами розуміння, відповідно до яких зміст є не простою сумою елементів, а результатом їх інтеграції в цілісну семантичну структуру.

Задача автоматичної генерації AMR репрезентацій є центральною проблемою семантичного аналізу природної мови. Згенерований граф має відповідати як семантичному змісту речення, так і структурним правилам AMR формалізму, тобто під час генерації важливо чітко представити вирівнювання між лексемами слів у вхідному реченні та відповідними поняттями та

відношеннями в його графі AMR. Це вирівнювання забезпечує пряму відповідність між текстом та його семантичним представленням.

Перші підходи до побудови AMR репрезентацій базувалися на вирівнюванні та використовували перехідні системи для побудови графів. Але з розвитком нейронних мереж, сучасні підходи активно використовують методи глибокого навчання для отримання репрезентацій. Загалом сучасні методи генерації AMR можна умовно розділити на наступні підходи: методи, які використовують моделі типу послідовність-послідовність та методи, що використовують графові нейронні мережі для прогнозування графів.

Підхід послідовність-послідовність є архітектурним підходом в машинному навчанні, який здебільшого використовується в моделях обробки мови [46]. Даний підхід реалізується за допомогою рекурентних нейронних мереж RNN та використанням кодувальника та декодера, що дозволяє за допомогою кодувальника перетворювати вхідну послідовність на стислий контекстуальний вектор, після чого декодер покроково генерує вихідну послідовність (рис. 2.3). Завдяки застосуванню рекурентних нейронних мереж (RNN) кожен елемент вхідної послідовності обробляється з урахуванням попереднього контексту, що забезпечує збереження інформації про попередні стани. Це дає змогу моделі враховувати як структурні особливості тексту, так і релевантність окремих його складників у процесі генерації або обробки послідовності.

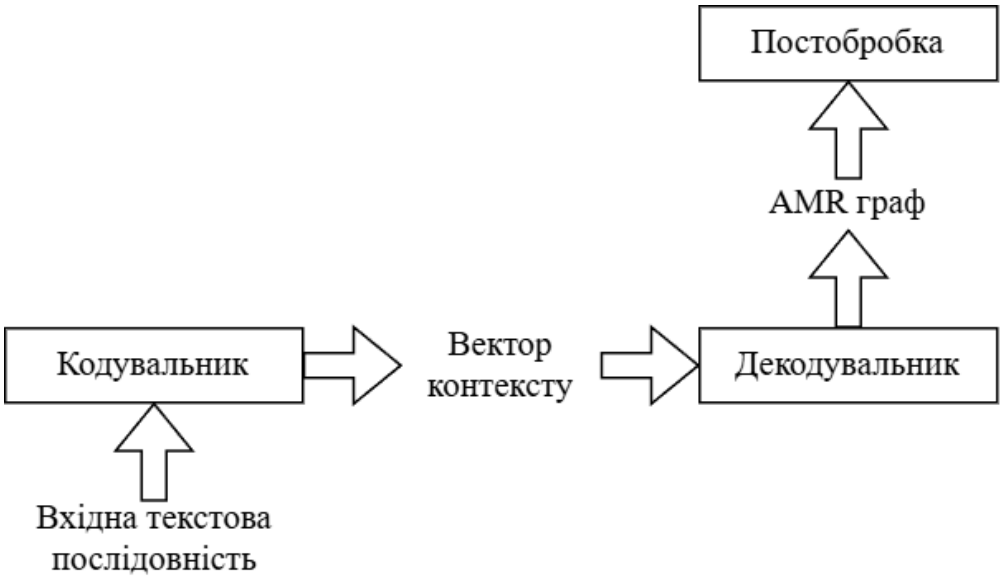

Рис. 2.3. Архітектура з використанням кодувальника та декодера для перетворення вхідної послідовності тексту в AMR граф

Одними з перших моделі типу послідовність-послідовність у контексті задачі генерації AMR були застосовані у роботі С. Пенга та співавторів [47], де використовувалася архітектура двонапрямних рекурентних нейронних мереж LSTM з використанням механізмів уваги. Для обробки вхідної послідовності кодувальник використовує двонапрямну RNN для захоплення інформації зі зворотного порядку вхідної послідовності, після чого декодер на кожному кроці за допомогою механізму уваги обчислює зважену суму прихованих шарів, зосереджуючись на важливих частинах послідовності для генерації лінеаризованого AMR графа.

Запровадження технології трансформерів призвело до значного покращення результатів генерації AMR графів. Вперше запропоновані у роботі А. Васвані та співавторів [48], трансформери відкрили нову парадигму в обробці природної мови, дозволяючи ефективно моделювати довготривалі залежності між елементами послідовності. Завдяки архітектурі повністю заснованій на механізмах уваги, трансформери дозволили уникнути обмежень рекурентних моделей, забезпечивши паралелізацію обчислень і значне підвищення якості в задачах перекладу, узагальнення, генерації та аналізу тексту.

У задачах генерації AMR для побудови кодувальника та декодера популярною є модель BART [49]. Наприклад, вона використовується у парсері SPRING, котрий вважається одним із популярних сучасних методів до генерації AMR графів [50]. SPRING є моделлю перетворення послідовності на послідовність, що має на меті створити лінеаризований граф AMR на основі заданого речення. Вхідні дані представляються як послідовність токенів $s = \langle BOS, w_1, w_2, ..., w_n, EOS \rangle$, у якій кожен токен w_i є окремим словом з лексикону V, а BOS та EOS є спеціальними токенами для позначення початку та кінця строки. Для прогнозування вихідної послідовності токенів SPRING використовує параметризовану функцію, яка приймає у себе вхідну послідовність та поточний стан частково згенерованої вихідної послідовності й на кожному кроці видає розподіл ймовірностей наступного токена вихідної послідовності. Повторюючи цю операцію послідовно, модель може обчислити ймовірність повної послідовності шляхом перемноження умовних ймовірностей для кожного токена зліва направо. Навчання моделі здійснюється шляхом мінімізації функції втрат негативного логарифма умовної ймовірності на множині пар речення-граф. Після завершення навчання, модель використовується для генерації вихідної послідовності з найвищою умовною ймовірністю. Вихідною послідовністю при розв’язанні задачі є лінеаризований AMR граф виду $g = \langle BOS, g_1, g_2, ..., g_m, EOS \rangle$, де g_i – графовий токен, що відповідає токену слова з вхідної послідовності.

Аналогічно BART був застосований в парсері AMRBART, який був тонко налаштований як для завдань перетворення AMR на текст, так і для перетворення тексту на AMR [51]. Основна відмінність цієї моделі від SPRING полягає в тому, що вона поєднує попереднє навчання на графах AMR з тонким налаштуванням, що дає змогу моделі краще засвоювати взаємозв’язки між графовими та текстовими репрезентаціями мовних одиниць. В основі моделі лежить попередньо натренований трансформер, який навчається за методом усунення шуму – тобто шляхом відновлення оригінальних даних на основі їх навмисно спотворених версій. Такий підхід дозволяє моделі навчитися розуміти структуру

вхідної інформації, узагальнювати її та бути стійкою до шуму або неповноти, що особливо корисно в задачах генерації, реконструкції чи перекладу. Для навчання використовуються дві основні стратегії усунення шуму:

1. На рівні вузлів та ребер – модель навчається реконструювати локальні порушення структури графа.

2. На рівні підграфів – модель відновлює більші семантичні блоки, що були частково або повністю замасковані.

З метою спільного попереднього навчання для графів і текстів, модель також імплементує чотири задачі усунення шуму:

1. Усунення шуму у тексті з графовим доповненням. Для розв’язання задачі модель отримує маскований текст як основний вхід та повний AMR граф як додатковий контекст. Метою є відновити оригінальний текст на основі пошкодженого тексту та повного графа.

2. Усунення шуму у графі з текстовим доповненням В цій задачі модель отримує маскований AMR граф як основний вхід та повний текст у вигляді контексту. Завдання полягає у відновленні повного графа на основі його маскування та відповідного тексту.

3. Усунення шуму у тексті з маскованим графовим доповненням. У цьому випадку і текст, і граф подаються до моделі у маскованій формі. Модель має відновити початковий текст, використовуючи обидва спотворені входи.

4. Усунення шуму у графі з маскованим текстовим доповненням. Аналогічно до попереднього, модель повинна відновити початковий AMR граф, використовуючи масковані версії як тексту, так і графа.

За допомогою цих задач модель реалізує двостороннє семантичне узгодження між текстом і графом, сприяючи кращому перенесенню знань між модальностями. Таким чином, модель AMRBART формує узгоджене подання як графових, так і текстових послідовностей, що дозволяє ефективно розв’язувати задачі AMR парсингу та генерації в обох напрямах. Після завершення навчання декодування виконується шляхом знаходження вихідної послідовності з найвищою ймовірністю.

Іншим підходом до генерації AMR є представлення цієї задачі як задачі передбачення графової структури на основі вхідного тексту (рис. 2.4.). На відміну від моделей типу послідовність-послідовність, у даному підході безпосередньо моделюється графова структура AMR у вигляді об’єкта з вузлами й ребрами без декодування лінійних послідовностей. Моделі спочатку прогнозують вузли графа, які відповідають ключовим смисловим одиницям, таким як дії, об’єкти та атрибути, а після цього встановлює між ними зв’язки, що відображають семантичну структуру висловлення.

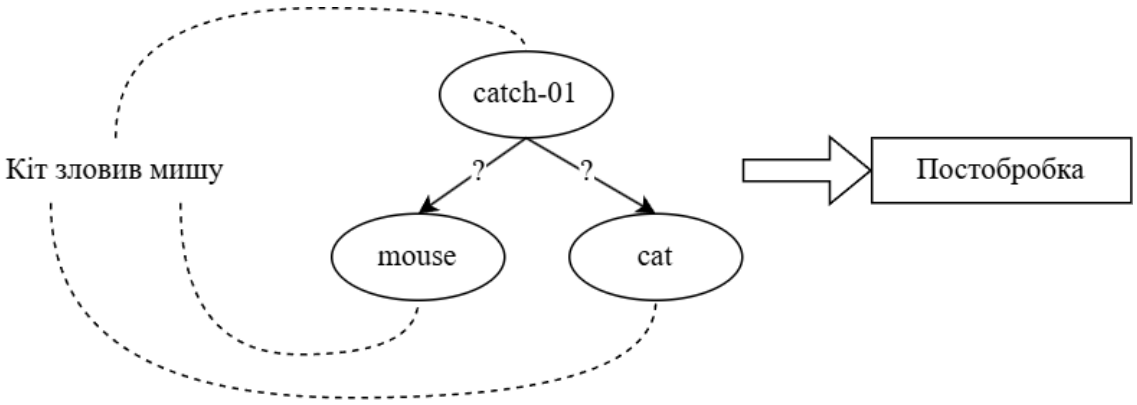

Рис. 2.4. Схема прогнозування структурних компонентів AMR графа

Одним із підходів у цьому напрямі є AMR Graph Prediction [52], який реалізує спільне моделювання вузлів і ребер графа, одночасно обробляючи вирівнювання між вхідними словами та вузлами. На відміну від традиційних методів, які розглядають вирівнювання та синтаксичний розбір як окремі кроки, ця модель одночасно передбачає всю структуру AMR.

Запропонована модель складається з трьох основних компонентів: модель ідентифікації концептів, модель ідентифікації відношень та модель вирівнювання. Формально, генеративна модель AMR включає перші дві компоненти разом з рівномірним апріорним розподілом над вирівнюваннями між словами та концептами. Модель вирівнювання, своєю чергою, використовується під час навчання для отримання варіаційного наближення до справжнього апостеріорного розподілу між концептами, відношеннями та

словами вхідного речення, що дозволяє ефективно навчати модель без необхідності в зовнішньому вирівнюванні. Така структура дозволяє моделі більш узгоджено формувати структуру графа, зменшуючи кількість помилок на етапах синтезу та вирівнювання. Передбачення ребер здійснюється з використанням біафінного класифікатора, який на основі пар вузлів визначає типи відношень. У такий спосіб AMR GP здатна генерувати повноцінні графи без необхідності у лінеаризації або використанні попередньо натренованих вирівнювачів.

У іншому підході використовується модель Sequence-to-Graph Transduction (STOG), яка розглядає AMR парсинг як послідовний процес із двох фаз [53]. На першому етапі модель прогнозує список концептів за допомогою розширеної мережі Pointer-Generator [54], що поєднує можливість генерувати нові вузли та копіювати їх із вхідного тексту, враховуючи контекст. На другому етапі здійснюється побудова відношень між вузлами за допомогою біафінного класифікатора. STOG використовує єдину функцію втрат, що мінімізує похибки як у прогнозуванні концептів, так і у визначенні їх зв’язків, тим самим підтримуючи структурну цілісність графа. Однією з переваг цієї моделі, як і у моделі AMR GP, є відмова від використання зовнішнього вирівнювача – STOG безпосередньо вчиться встановлювати відповідність між словами тексту та вузлами графа, що підвищує гнучкість моделі.

2.5 Метод генерації абстрактних семантичних репрезентацій для текстів природної мови

Аналіз наявних підходів до генерації AMR графів показав, що методи прямого прогнозування графової структури забезпечують високий рівень точності для мов із великими розміченими корпусами. Ці підходи моделюють вузли та ребра графа безпосередньо, одночасно обробляючи вирівнювання між словами та концептами. Водночас їх застосування для української мови обмежене через кілька ключових факторів:

1. Обмежена кількість розмічених даних – для української мови наявні корпуси AMR є значно меншими, ніж для англійської, що ускладнює навчання моделей, які потребують великої кількості прикладів для прогнозування графових структур.

2. Складність налаштування моделей вирівнювання – пряме прогнозування графів потребує високоточного вирівнювання слів із концептами та ребрами графа. В умовах обмежених даних це призводить до підвищеного рівня помилок і нестійкої генерації.

3. Складність адаптації під лінгвістичні особливості української мови – особливості синтаксису та морфології української мови ускладнюють пряме прогнозування ребер і концептів без додаткової лінеаризації.

Таким чином, застосування цих моделей вимагає значних обчислювальних ресурсів та складного налаштування, що робить їх адаптацію до мов з обмеженими лінгвістичними ресурсами, до яких наразі належить українська в контексті AMR анотацій, неоптимальною.

З іншого боку, замість прямого прогнозування графа G = (V, E), що є складною структурною задачею, метод послідовність-послідовність зводить її до задачі генерації лінеаризованої послідовності. Вхідна послідовність слів відображається у вихідну послідовність токенів, яка репрезентує графову структуру (рис. 2.5.).

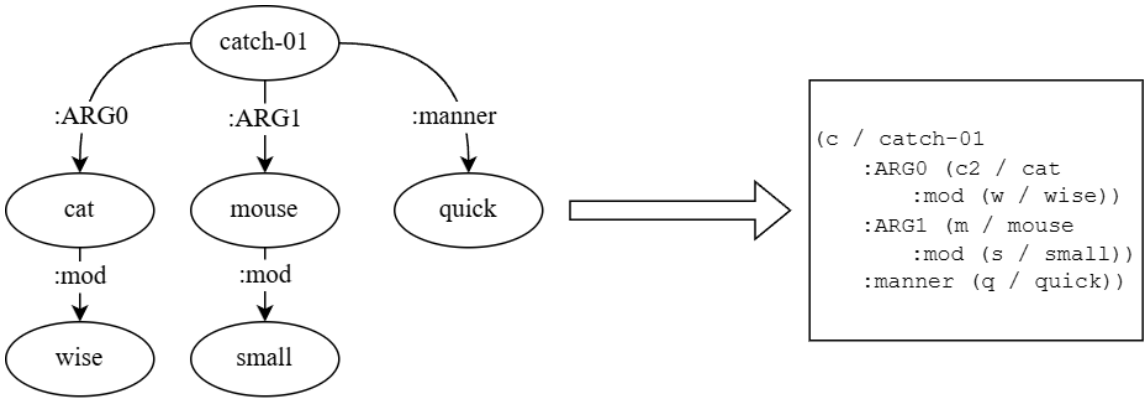

Рис. 2.5. Репрезентація AMR графа у вигляді лінеаризованої послідовності в нотації Penman

Основні переваги цього підходу для мов з обмеженими ресурсами полягають у наступному:

1. Адаптація до лімітованих корпусів – для розв’язання задачі можна використовувати попередньо навчені трансформерні моделі, які були натреновані на великих багатомовних або англомовних корпусах, і лише тонко налаштовувати їх для українських даних.

2. Лінеаризація графа як спрощення завдання – перетворення AMR графа у послідовність токенів спрощує задачу генерації, дозволяючи моделі оперувати форматом тексту, що значно зменшує потребу у великій кількості розмічених прикладів для навчання.

3. Використання трансформерних архітектур – застосування сучасних трансформерних моделей на кшталт BART, T5 або GPT, які здатні моделювати довготривалі залежності, забезпечувати контекстуальне розуміння і відновлення структурованого виводу на основі неповного або зашумленого вхідного тексту.

4. Гнучкість у відновленні семантичних структур – через механізми уваги та стратегії декодування моделі типу послідовність-послідовність можуть ефективно зосереджуватися на ключових концептах тексту та формувати зв’язки між ними у вихідній AMR послідовності, забезпечуючи узгодженість структур без прямого прогнозування ребер.

З формального погляду задачу генерації AMR графів можна представити як перетворення вхідної послідовності токенів природної мови $X=(x_1,\ldots,x_n)$, у якій x_i належать до словника природної мови V_{text}, у лінеаризовану послідовність елементів семантичного графа $Y=(y_1,\ldots,y_n)$, де y_i належать до словника AMR елементів V_{AMR}, що включає концепти, семантичні ролі та структурні маркери.

Задача генерації графової репрезентації текстів природної мови полягає у побудові умовної ймовірності вихідної послідовності за умови вхідної:

$$P(Y|X;\theta) = \prod_{t=1}^{m} P(y_t|y_{<t}, X;\theta), \qquad (2.5)$$

де X – вхідна послідовність токенів речення природною мовою, θ – параметри моделі, m – загальна довжина згенерованої послідовності елементів графа, y_t – токен AMR графа, що генерується на кроці t, $y_{<t}$ – контекст, сформований на попередніх кроках.

Метою навчання моделі є пошук оптимальних параметрів θ шляхом максимізації логарифмічної правдоподібності за всією навчальною вибіркою згідно з формулою (2.5):

$$\mathcal{L}(\theta) = \sum_{(X,Y)} \log P(Y|X;\theta), \tag{2.6}$$

де X – послідовність токенів вхідного речення українською мовою, Y – лінеаризована послідовність AMR графа, що описує семантичну структуру відповідного речення, θ – параметри моделі, які оптимізуються під час навчання.

Таким чином етап лінеаризації дозволяє перевести задачу генерації семантичних представлень з галузі графового прогнозування до класу задач перетворення послідовностей. Це дозволяє застосовувати попередньо навчені на великих текстових корпусах, трансформерні моделі, які демонструють високі результати у захопленні довготривалих залежностей та генерації послідовностей. Для мов з обмеженими ресурсами, таких як українська, це надає можливість використати наперед навчені моделі, замість навчання складної графової моделі з нуля.

Серед трансформерних моделей типу послідовність-послідовність особливу увагу привертає модель BART. На відміну від класичних архітектур з кодувальником та декодером, які навчаються на прямому перекладі, BART базується на концепції автокодувальника з усуненням шуму [49].

Під час попереднього навчання BART трансформує вхідний текст із метою відновлення його початкової форми. Такий механізм робить модель особливо придатною для завдань генерації AMR, оскільки семантичний парсинг вимагає від моделі відновлення повного контексту речення та на його основі створення

нової, синтаксично складної структури, що є завданням авторегресійного декодера. Попереднє навчання з усуненням шуму дозволяє моделі опановувати не лише поверхневі, але й глибинні синтаксичні та семантичні зв’язки тексту, що критично важливо для коректної побудови семантичного графа.

Базова архітектура BART не враховує специфіку семантичних графів, через що пряме тонке налаштування стандартної моделі на парах «текст – AMR граф» є необхідним для адаптації до завдання генерації AMR. Для подолання обмежень стандартного підходу та підвищення точності моделювання структурованих семантичних репрезентацій було обрано спеціалізовану модифікацію BART – AMRBART, розроблену з урахуванням потреб задачі AMR парсингу [51].

На відміну від стандартного BART, який відновлює лише текст, AMRBART оперує комплексом завдань усунення шуму, що охоплюють як відновлення тексту за наявності графа, так і відновлення графа за наявності тексту. AMRBART формує єдиний узгоджений семантичний простір, у якому текстові токени та графові концепти набувають близьких векторних представлень. Така властивість робить модель оптимальною для застосування у методі генерації AMR для української мови, оскільки вона вже володіє базовим розумінням граматики AMR і потребує значно менше специфічних даних для тонкого налаштування на завданні парсингу українського тексту порівняно зі стандартним BART.

Таким чином, враховуючи проаналізовані обмеження підходів прямого прогнозування графа та специфіку задачі в умовах обмежених лінгвістичних ресурсів, запропонований у дисертаційній роботі метод генерації AMR для української мови базується на парадигмі моделей типу послідовність-послідовність. Цей метод концептуально реалізується через застосування трансформерної архітектури AMRBART, що модифікована для спільного семантичного навчання на текстових та графових даних.

2.5.1. Метод багатомовної генерації абстрактних семантичних репрезентацій із використанням нейронного машинного перекладу

На сьогодні головною проблемою є генерація AMR репрезентацій для мов із низькою ресурсною базою, до яких належить українська мова. Попри те що концепція оригінальної моделі генерації AMR абстрагується від мовних особливостей, таких як порядок слів, морфологія чи службові слова, ця модель здебільшого спирається на використання ресурсів, призначених для генерації текстів англійською мовою. Відповідно, більшість сучасних корпусів даних для генерації AMR орієнтовані на англійську мову, що суттєво ускладнює навчання якісних моделей для інших мов.

Один з напрямів розв'язання описаної проблеми має назву міжмовний AMR парсинг [55] і полягає у перетворенні речення цільовою мовою у відповідний AMR граф англійською мовою, де вузли представлені англомовними словами, наборами фреймів PropBank або спеціальними ключовими словами AMR (рис. 2.6). Ранні дослідження описують методи, в яких здійснювалося порівняння вручну згенерованих AMR репрезентацій для різних мов із їх англомовними аналогами з метою виявлення спільних семантичних структур. Загалом можна виділити два підходи до створення корпусів AMR для інших мов. Перший підхід передбачає адаптацію правил анотації англійською мовою до цільової мови та анотування даних цільової мови експертами. Другий підхід полягає в перекладі речень з цільової мови англійською, після чого перекладені речення використовуються як вхідні дані для AMR парсерів англійської мови. Останній підхід дозволяє застосовувати корпуси англійської мови для генерації AMR для інших мов, що сприяє прискоренню розробки міжмовних моделей.

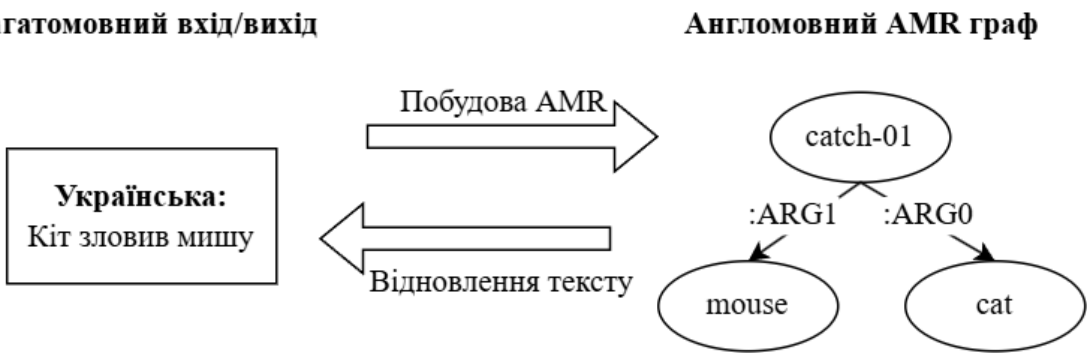

Рис. 2.6. Концептуальна схема міжмовного AMR парсингу

У галузі багатомовного AMR парсингу було запропоновано підхід XL-AMR, у якому використовуються методи трансферного навчання та багатомовні моделі [56]. Основна ідея XL-AMR полягає в тому, щоб уникнути необхідності створювати окремі AMR парсери для кожної мови, натомість здійснити перенесення знань з англомовного великого AMR корпусу на інші мови. Для цього XL-AMR використовує попередньо натреновані багатомовні трансформери, такі як mBERT або XLM-RoBERTa, які формують спільний латентний простір представлень для декількох мов [57, 58]. Це дозволяє здійснювати безпосередній парсинг речень з інших мов у формальну структуру AMR, навіть якщо для цих мов бракує розмічених даних.

Проте, попри свій науковий внесок, цей підхід має низку критичних обмежень при застосуванні до мов із низькими ресурсами, зокрема української. По-перше, якість згенерованих AMR значною мірою залежить від обсягу паралельних корпусів між цільовою мовою та англійською, а для української такі корпуси мають недостатнє покриття та нерівномірну тематику. По-друге, морфологічна складність та вільний порядок слів української мови знижують точність прямого міжмовного парсингу, оскільки модель не отримує явних семантичних підказок, доступних у текстах мов із строгим порядком слів. По-третє, відсутність українських AMR анотацій унеможливлює тонке настроювання, що є необхідним для високої якості парсингу.

З іншого боку, дослідження в галузі прямого парсингу з цільової мови показали, що розвиток систем нейронного машинного перекладу (NMT)

дозволяє досягати, а в багатьох випадках перевершувати результати прямих міжмовних AMR парсерів. Так, у роботі С. Уріга та співавторів [59] було продемонстровано ефективність підходу translate+parse (T+P), який передбачає двоетапну обробку: спочатку речення з цільової мови перекладається на англійську за допомогою сучасної NMT-системи, а потім отриманий англомовний текст аналізується вже перевіреним англомовним AMR парсером. Експериментальні результати демонструють, що підхід з використанням моделей машинного перекладу суттєво перевершує прямий міжмовний парсинг XL-AMR: середнє значення Smatch для T+P складає від 59.1 до 72.3 для німецької, італійської, іспанської та китайської мов, що перевищує відповідні показники XL-AMR більш ніж на 20 балів, підтверджуючи ефективність використання нейронного перекладу для збереження ключових семантичних відношень в AMR графах. Таким чином, нейронний переклад може розглядатися не як допоміжний етап, а як ефективна методика адаптації AMR парсингу до нових мов, що особливо важливо для мов з обмеженими ресурсами.

2.6. Метод визначення семантичної еквівалентності графових репрезентацій текстів природної мови

Визначення семантичної еквівалентності висловлень природної мови є ключовою задачею при побудові інтелектуальних систем розуміння тексту, оскільки воно визначає можливість ототожнення змісту незалежно від лексичних, синтаксичних та стилістичних варіацій формулювання. Традиційні методи порівняння текстів спиралися здебільшого на поверхневі лінгвістичні ознаки, однак такі підходи не здатні адекватно відображати глибинні семантичні залежності між концептами, що формують значення висловлення.

У попередніх підрозділах було показано, що використання семантичних графових представлень AMR дає можливість відобразити структуру висловлення у формі абстрактної моделі семантики, яка фіксує події, учасників, ролі та логіко-семантичні відношення між ними. AMR графи забезпечують

інваріантність щодо синтаксичної форми та лексичного наповнення, що робить їх придатними для аналізу перифраз та для формального встановлення семантичної еквівалентності.

Однак, пряме порівняння AMR графів традиційними графовими метриками має низку обмежень. Такі метрики є загальними для довільних графових структур і не враховують специфіку семантичних ролей, а також варіативність репрезентації одних і тих самих концептів у різних контекстах. Це призводить до втрати релевантної інформації про семантичну близькість, особливо для випадків часткової еквівалентності смислових структур.

Враховуючи зазначені обмеження, виникає необхідність у створенні методу, що визначає семантичну еквівалентність AMR графів не лише через структурну ізоморфність, але й через узгодженість семантичних ролей, концептуальних вузлів та функціональних залежностей, що відображають когнітивні процеси інтерпретації значення висловлення.

У запропонованому підході AMR граф розглядається як орієнтований анотаційний граф:

$$G = (V, E, L_V, L_E), \tag{2.7}$$

де V – множина вузлів графа, що відповідають концептам або подіям, E – множина орієнтованих ребер між вузлами, L_V – функція маркування вузлів на множину концептів, L_E – функція маркування ребер на множину семантичних ролей.

Така модель відображає пропозиційну структуру висловлення як систему взаємопов’язаних смислових одиниць, що утворюють інваріантне ядро значення. Семантична близькість між двома висловленнями у цьому формалізмі трактується як ступінь відповідності їхніх графів G_1 та G_2. Однак, безпосереднє порівняння графів в просторі є проблематичним з кількох причин:

1. Семантична варіативність вузлів. Один і той самий концепт може бути виражений різними лексемами або різними лексико-граматичними конструкціями, що не повинно вважатися відмінністю змісту.

2. Контекстна функціональність ролей. Семантичні ролі можуть набувати специфічних інтерпретацій залежно від ситуації.

3. Структурна неоднорідність. Два еквівалентні за змістом висловлення можуть мати різну поверхневу організацію графа, навіть при збереженні однакового змісту.

Таким чином перевірка ізоморфізму дискретних структур, виявляється недостатньо гнучкою для завдань ідентифікації перифраз через природну варіативність мови та можливі помилки автоматичного парсингу. Подолання наведених обмеженнь потребує переходу від дискретних структур до обчислювальних методів, що реалізується через відображення графів у неперервний латентний векторний простір. У такому представленні семантична близькість перестає бути бінарною і трансформується у метричну характеристику відстані між векторами.

Для формалізації цього переходу вводиться операція векторного представлення графів:

$$\Phi : G \longrightarrow \mathbb{R}^d, \tag{2.8}$$

де G – множина AMR графів, а $\mathbb{R}^d$ – простір векторних представлень розмірності d.

Це відображення задовольняє вимогу збереження інваріантності до поверхневих трансформацій, при одночасному збереженні відмінностей у пропозиційній структурі. Тобто, графи, що описують однакові або близькі за значенням висловлення, повинні бути відображені в близькі точки латентного простору, тоді як графи з різним змістом – у віддалені.

Векторні представлення у латентному просторі мають назву вкладення і слугують компактним числовим описом складної структури AMR графа. Вони перетворюють дискретну графову репрезентацію, що містить вузли й ребра, у щільний багатовимірний вектор, зберігаючи при цьому критично важливу інформацію про семантику та топологію графа.

Основна мета вкладень полягає у забезпеченні можливості обчислювати семантичну подібність між висловленнями на рівні векторів за допомогою стандартних метрик, таких як косинусна, евклідова або манхеттенська відстані. Вкладення дозволяють зменшити обчислювальні витрати порівняно з класичними методами порівняння графів, водночас забезпечуючи диференційовану репрезентацію, що придатна для навчання нейронних моделей, зокрема для класифікації перифраз та оцінки семантичної близькості речень.

Серед наявних підходів до побудови вкладень AMR графів виділяють кілька основних напрямків. Перший напрямок включає методи вкладання графів, серед яких найбільш відомими є Node2vec та Graph2Vec [60, 61].

Метод Node2vec генерує векторні представлення окремих вузлів графа. Основна ідея полягає у використанні модифікованих випадкових обходів графа для створення контекстів вузлів. Node2vec дозволяє навчити вектори вузлів таким чином, щоб вузли з подібним топологічним оточенням мали близькі вектори у латентному просторі. Метод надає гнучкість у балансуванні між локальними та глобальними структурами графа через параметри, які контролюють схему випадкових обходів. Graph2Vec своєю чергою поширює ідею Node2vec на рівень цілих графів. Він представляє граф як множину шаблонів підграфів, після чого застосовує механізм агрегації для отримання єдиного векторного представлення графа.

Обидва підходи успадковують методи вкладень слів: близькі у векторному просторі вузли або графи мають схоже семантичне значення [62]. Водночас такі методи мають обмеження у контексті AMR графів, а саме:

1. Обмежена здатність моделювати складні семантичні взаємозв’язки: Методи на основі випадкових обходів не завжди ефективно враховують

ієрархічну та контекстуальну природу відношень між концептами, що критично для AMR, де смислові ролі та кореференції суттєво впливають на значення речення.

2. Відсутність інтеграції глобального контексту: Node2vec та Graph2Vec переважно фокусуються на локальних топологіях вузлів або шаблонах підграфів, ігноруючи глобальні структурні залежності, які визначають семантичну структуру речення.

У зв’язку з цим, для отримання більш інформативних та значущих вкладень AMR графів доцільно застосувати графові нейронні мережі (GNN). GNN – це клас нейронних мереж, спеціально розроблених для обробки даних із графовою структурою, де елементи представлені вузлами, а зв’язки між ними – ребрами. Основна ідея GNN полягає у послідовній агрегації інформації від сусідніх вузлів і ребер, що дозволяє кожному вузлу навчатися з урахуванням локального контексту та глобальної топології графа. GNN дозволяють інтегрувати як локальну інформацію про вузли та ребра, так і глобальний контекст графа через багатошарову агрегацію повідомлень. Такий підхід забезпечує ефективне навчання семантичних репрезентацій графів, здатних зберігати структурні особливості AMR, що є критично важливим для подальшої класифікації перифраз та визначення семантичної подібності.

У межах метода векторизації AMR графів також застосовуються згорткові графові нейронні мережі. Вибір згорткової архітектури для аналізу AMR графів базується на аналогії з обробкою зображень у згорткових нейронних мережах (CNN) [63]. У CNN згортка дозволяє моделі виявляти локальні шаблони зображення та комбінувати їх у складніші семантичні ознаки. Аналогічно, у графових даних згортка виконує роль агрегації локального контексту: кожен вузол оновлює свій стан на основі ознак сусідніх вузлів і типів ребер. Таким чином, модель може поступово узагальнювати інформацію від локальних до глобальних залежностей, що є важливим для AMR графів, які репрезентують глибинну семантичну структуру речень. Загальна архітектура згорткових графових мереж представлена на рисунку 2.7.

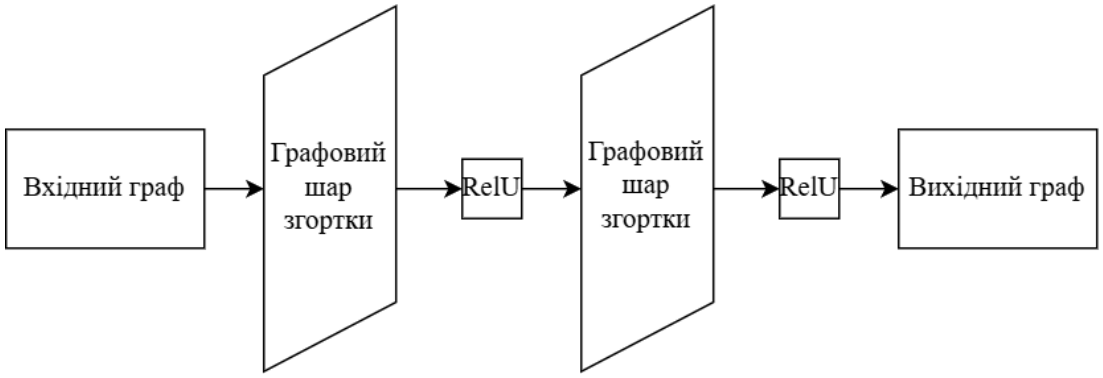

Рис. 2.7. Узагальнена архітектура графових згорткових мереж

Крім того, графові згортки мають властивість параметричного повторного використання, тобто однакові ваги застосовуються до всіх сусідніх вузлів, що забезпечує ефективне узагальнення та скорочення кількості параметрів у порівнянні з рекурентними або трансформерними підходами при обробці фіксованих графових структур. Зважаючи на це згорткові графові архітектури є оптимальним компромісом між стабільністю навчання та обчислювальною ефективністю.

Отримані векторні представлення графів використовуються як ознаки для класифікації при побудові моделі ідентифікації перифраз між двома реченнями. На цьому етапі реалізується функція класифікації парних представлень, яка відображає два вектори розмірності d у бінарний результат. Вихідне значення такої функції інтерпретується як ймовірність наявності семантичної еквівалентності між реченнями.

Серед підходів реалізації класифікаційних моделей, які дозволяють ефективно працювати з високорозмірними векторними просторами та моделювати складні залежності між ознаками, можна виділити Метод k-ближчих сусідів, SVM та багатошарові перцептрони (MLP).

Найпростішим методом класифікації вважається Метод k-ближчих сусідів (k-NN), який класифікує нову пару речень шляхом пошуку найближчих прикладів у векторному просторі за обраною метрикою подібності [64]. Його перевага полягає у простоті реалізації та здатності працювати з нелінійними

розподілами даних. Водночас одним із головних недоліків методу є його висока чутливість до розмірності простору ознак, особливо в контексті роботи з векторними репрезентаціями, отриманими з AMR графів. У просторах з великою кількістю вимірів відстані між точками стають все менш інформативними, і всі приклади виявляються приблизно однаково віддаленими один від одного. Це суттєво ускладнює коректну класифікацію.

Іншим методом класифікації є використання SVM з ядровими функціями, які дозволяють знаходити нелінійні межі поділу між класами у просторі векторів [21]. Ядра, що враховують семантичну подібність, можуть ефективно відображати складну взаємодію між ознаками. Основним недоліком використання SVM для класифікації векторних репрезентацій, є його обмежена масштабованість, а також чутливість до вибору ядра. Невдале налаштування може призвести до перенавчання або, навпаки, до втрати здатності моделі захопити складні семантичні зв’язки.

Багатошарові перцептрони частково вирішують обмеження, характерні для SVM, завдяки своїй гнучкій архітектурі та здатності до навчання складних нелінійних функцій [65]. На відміну від SVM, які не масштабуються при зростанні розмірності чи кількості даних, MLP автоматично навчаються отримувати релевантні ознаки з великих обсягів вхідних даних. Крім того, MLP легко комбінують різні типи вхідних ознак – як векторні подання, так і евристичні метрики подібності – в одному узагальненому представленні. Завдяки цьому MLP краще справляються з неоднорідністю перифраз та мають потенціал до навчання на великих корпусах без потреби в спеціально підібраному ядрі або трансформації простору.

У межах запропонованого підходу доцільним є застосування багатошарового перцептрона як класифікаційного механізму. Така архітектура володіє рядом суттєвих переваг: вона забезпечує здатність моделювати складні нелінійні залежності між ознаками, підтримує ефективне навчання на великих корпусах даних та дозволяє гнучко комбінувати різнорідні ознаки подібності. Модель MLP може містити один або кілька прихованих шарів з нелінійними

функціями активації, наприклад ReLU, що дозволяє захоплювати складні семантичні шаблони у векторних репрезентаціях AMR графів. Для підвищення стабільності навчання та запобігання перенавчанню можливе використання інструментів регуляризації, таких як Dropout або Batch Normalization. Вихідний шар MLP може бути реалізований із сигмоїдальною або softmax-активацією залежно від формату задачі – бінарної класифікації чи багатокласової, що забезпечує коректне прогнозування ймовірності належності пари речень до класу перифраз.

2.7. Висновки за розділом 2

1. У розділі сформовано теоретичне підґрунтя для побудови інтелектуальної системи ідентифікації перифраз на основі графових семантичних репрезентацій. Аналіз концептуальних положень дозволив визначити фундаментальні принципи моделювання смислової еквівалентності природномовних висловлень та обґрунтувати вибір формальних моделей і методів, відповідно поставленій науковій задачі.

2. Процес розуміння визначено як механізм зіставлення нових смислових структур із наявними структурами знань, представленими у вигляді взаємопов’язаних концептів та їх відношень. Такий підхід дозволяє розбити комплексний процес розуміння на одиниці мислення, які можуть бути формалізовані, проаналізовані та використані при моделюванні інтелектуальних інформаційних систем. У межах дослідження на цій основі було обрано та формалізовано задачу ідентифікації перифраз як окремий випадок встановлення семантичної еквівалентності між висловленнями, що відрізняються поверхневою мовною формою, але мають спільний концептуальний зміст.

3. Моделювання смислових структур процесу розуміння вимагає формалізмів, здатних забезпечити відокремлення концептуального змісту від поверхневої мовної форми та створити умови для формалізованого зіставлення семантичних структур. У ході дослідження обґрунтовано доцільність

використання графових семантичних репрезентацій, зокрема формалізму AMR, як оптимальної моделі подання смислу висловлень для задачі ідентифікації перифраз. Показано, що AMR забезпечує уніфікацію структури висловленнь та дозволяє здійснювати їх порівняння на семантичному рівні.

4. На основі обраного формалізму розроблено метод генерації абстрактних семантичних репрезентацій текстів природної мови, орієнтований на багатомовне середовище. Запропонований підхід враховує відсутність повноцінного україномовного AMR корпусу та ґрунтується на використанні проміжного англомовного представлення, що забезпечує узгодженість концептуальних структур і стабільність формування AMR графів для українських текстів.

5. Розроблено метод визначення семантичної еквівалентності між AMR графами, який базується на використанні графових нейронних мереж для моделювання структурних і реляційних властивостей семантичних репрезентацій. Показано, що застосування графових моделей дозволяє адекватно відображати неоднорідність семантичних відношень і формувати векторні представлення, що враховують як локальні, так і глобальні характеристики смислової структури висловлення.

6. Розроблено метод класифікації перифраз на основі оцінювання семантичної еквівалентності з використанням багатошарового перцептрона. Запропонований метод використовує вкладення графових семантичних репрезентацій та забезпечує моделювання нелінійних залежностей між їх ознаками, що дає змогу приймати рішення щодо ступеня семантичної еквівалентності між висловленнями.

Таким чином, у межах розділу сформовано теоретичне підґрунтя інформаційної технології ідентифікації перифраз, яке охоплює етапи формування багатомовних абстрактних семантичних репрезентацій та їх глибинного семантичного зіставлення. Отримані результати створюють підґрунтя для розробки інтелектуальної системи та її подальшої експериментальної перевірки.

РОЗДІЛ 3
ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ІДЕНТИФІКАЦІЇ ПЕРИФРАЗ НА ОСНОВІ ГРАФОВИХ РЕПРЕЗЕНТАЦІЙ

У попередньому розділі було сформовано концептуальну основу підходу до ідентифікації перифраз, яка ґрунтується на представленні змісту висловлень у вигляді абстрактних семантичних графів. На основі аналізу когнітивної моделі процесу розуміння та формальних критеріїв семантичної еквівалентності було запропоновано метод генерації абстрактної семантичної репрезентації висловлень на основі формалізму AMR, а також метод визначення семантичної подібності цих репрезентацій шляхом побудови вкладень графів та їх подальшої класифікації.

Метою даного розділу є перехід від теоретичного обґрунтування до практичної реалізації інформаційної технології, яка реалізує запропоновані методи. Розділ описує архітектуру системи ідентифікації перифраз, послідовність обробки даних, функціональні модулі, моделі та програмні компоненти, що забезпечують роботу технології.

3.1. Архітектура інтелектуальної системи ідентифікації перифраз на основі графових репрезентацій

Розроблена інтелектуальна система спрямована на моделювання процесу семантичного сприйняття та інтерпретації висловлювань шляхом автоматизованого виявлення перифраз. На відміну від традиційних підходів до класифікації текстових подібностей, які базуються на порівнянні поверхневих характеристик, запропонована система оперує на рівні глибинних семантичних структур, відтворюючи семантичні ролі, предикатно-аргументні відношення, логічну структуру висловлення та приховані когнітивні зв’язки між його компонентами.

Архітектура системи побудована у форматі багаторівневого конвеєра, який поєднує методи комп'ютерної лінгвістики, семантичного аналізу графових структур та машинного навчання. На вхід система приймає два речення природною мовою та здійснює їх послідовну трансформацію: лінгвістичну нормалізацію, побудову семантичного графа у вигляді AMR, векторизацію графової структури та подальшу класифікацію семантичної еквівалентності. Архітектура системи складається з чотирьох взаємопов'язаних модулів. Загальну архітектуру системи ілюструє діаграма, наведена на рисунку 3.1.

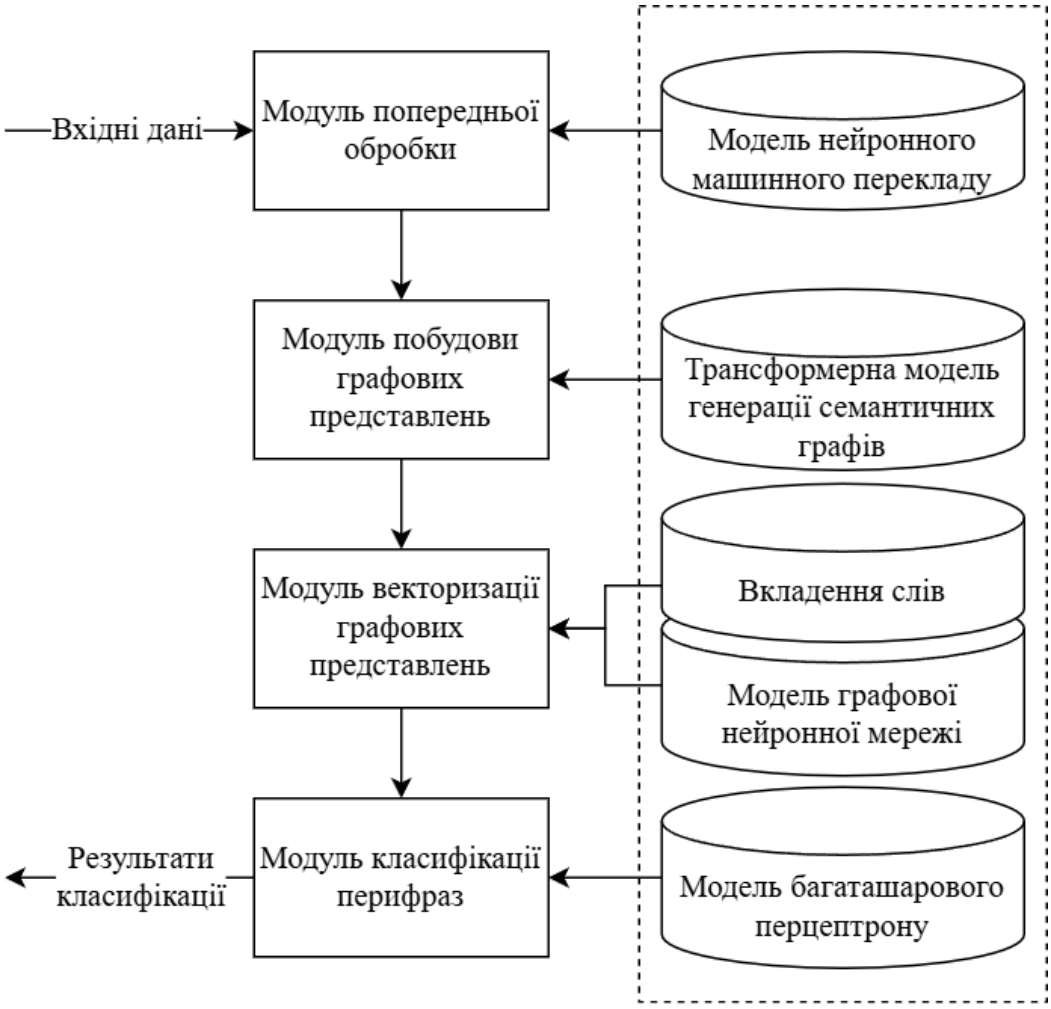

Рис. 3.1. Загальна архітектура інформаційної технології ідентифікації перифраз на основі AMR графів

Модуль попередньої лінгвістичної обробки та перекладу тексту моделює базові операції розуміння мовного сигналу – виділення одиниць аналізу та їх

лінгвістичну інтерпретацію. Також даний модуль містить етап перекладу вхідних текстів української мови за допомогою системи нейронного перекладу з метою подальшої генерації AMR графів з використанням англомовних парсерів.

Модуль генерації AMR графів виконує перетворення речень у формальні семантичні графи на основі предикатно-аргументної структури. У модулі графової векторизації побудовані AMR графи трансформуються у векторні репрезентації, що зберігають топологічні властивості графа, інформацію про семантичні ролі та контекстну взаємодію концептів. Модуль класифікації семантичної еквівалентності виконує оцінку ступеня подібності між двома вкладеннями AMR та визначає, чи є вони перифразами.

Представлена архітектура не лише розв’язує задачу автоматичної ідентифікації перифраз, але й відображає модель когнітивного процесу розуміння природної мови. У когнітивній лінгвістиці розуміння розглядається як багаторівневий процес, що включає аналіз поверхневої форми висловлювання, інтерпретацію смислових структур та порівняння значень на основі концептуальних зв’язків. Запропонована архітектура відтворює ці етапи у вигляді послідовного обчислювального конвеєра.

Перший етап – лінгвістична нормалізація та структурування тексту – є аналогом початкової когнітивної обробки, під час якої із вхідного мовного сигналу виділяються ключові смислові компоненти. У системі цю функцію реалізує поєднання модуля машинного перекладу та AMR парсингу. AMR графи абстрагують речення до інваріантної семантичної репрезентації, що відображає логічні предикати, аргументи та смислові відношення.

Другий етап архітектури – відображення у векторний простір – моделює процес формування у когнітивній системі абстрактних ментальних репрезентацій, шляхом перетворення графові структури у вкладення, які зберігають приховані відношення між концептами. Використання графових мереж або трансформерних кодувальників з увагою до локальних і глобальних залежностей дозволяє імітувати когнітивний механізм інтеграції знань та семантичної узгодженості.

Третій етап – оцінка семантичної еквівалентності – відображає механізм, відомий у когнітивній психології як процес порівняння концептуальних структур. Еквівалентність двох висловлювань визначається за наявністю тотожних смислових ролей і семантичних зв’язків. У реалізованій архітектурі цей етап відповідає модулю класифікації, який обчислює ступінь семантичної відповідності між двома графами. Таким чином, класифікація перифраз реалізується як аналіз відповідності смислових структур, що наближає систему до природних когнітивних механізмів. У сукупності ці три етапи реалізують обчислювальну модель розуміння, яка узгоджується з сучасними уявленнями когнітивної науки про процес семантичної обробки.

3.2. Структура та функції інтелектуальної системи ідентифікації перифраз на основі графових репрезентацій

Інтелектуальна система ідентифікації перифраз реалізована як комплексна система, де структура та функції інтегровані для забезпечення багаторівневої обробки інформації. Основною структурною рисою системи є модульність із чітким поділом завдань між окремими блоками. Взаємодія модулів відбувається за послідовним принципом: вихід одного блоку слугує входом для наступного.

Загалом структура системи представлена чотирма взаємопов’язаними модулями:

1. Модуль попередньої обробки та перекладу тексту реалізує підготовку вхідних даних для подальшої семантичної обробки. На вхід модуль приймає текст українською мовою, який проходить базову нормалізацію. На виході модуль повертає текст у форматі, сумісному з AMRBART, готовий для генерації AMR графа.

2. Модуль парсингу графових представлень з використанням AMR відповідає за формалізацію змісту висловлень. На вхід він приймає англомовний текст від попереднього модуля. Парсер AMRBART з вхідного тексту генерує лінеаризоване представлення графа у стандартній нотації PENMAN. На виході

модуль повертає структуровані графові репрезентації, що слугують вхідними даними для модуля графового кодування.

3. Модуль векторизації графових представлень реалізує перетворення AMR графів у векторні репрезентації або вкладення. На вхід подається AMR граф з попереднього кроку. Далі граф обробляється графовою нейронною мережею, що зберігає топологічну структуру, семантичні ролі та контекстні взаємозв’язки між концептами. На виході модуль формує щільний вектор високої розмірності, який інкапсулює семантику всього висловлення та готує дані для оцінки семантичної еквівалентності.

4. Модуль класифікації перифраз здійснює фінальну оцінку. На вхід він отримує пару вкладень графа. Внутрішній алгоритм обчислює міру семантичної відстані між ними та передає її на класифікатор, навчений розрізняти пари перифраз. На виході модуль повертає клас результату – «перифраз» або «не перифраз», а також оцінку ступеня подібності, яка використовується для подальшого прийняття рішення.

Взаємодія між цими модулями забезпечується контролером системи, який реалізує загальну логіку. Контролер приймає початковий запит, послідовно викликає кожен модуль, передаючи результати роботи попереднього модуля на вхід наступному, та повертає кінцевий результат. Така структура системи забезпечує інформаційні зв’язки між окремими програмними компонентами як у горизонтальному напрямку – під час передавання даних між послідовними етапами обробки, так і у вертикальному – під час взаємодії між API та обчислювальними модулями. Загальна структура представлена на рисунку 3.2.

Рис. 3.2. Структурно-функціональна схема взаємодії модулів інформаційної технології

Функціональна модель системи базується на принципах горизонтальної та вертикальної узгодженості. Горизонтальна взаємодія передбачає обмін даними між модулями одного рівня обробки, тоді як вертикальна визначає ієрархію перетворення інформації – від попередньої обробки тексту до формування вкладень графа та видачі результатів класифікації. Взаємодія між описаними модулями системи, їх вхідна та вихідна інформація наведена в таблиці 3.1.

Таблиця 3.1

Взаємодія модулів інформаційної технології ідентифікації перифраз

Модуль	Джерело	Вхідні дані	Вихідні дані	Адресат
Попередня обробка	Ввід користувача	Речення природною мовою	Нормалізовані речення	Модуль парсингу
Парсинг AMR графів	Модуль попередньої обробки	Нормалізовані речення	Графові AMR репрезентації	Модуль векторизації
Векторизація графів	Модуль парсингу	AMR репрезентації	Векторні репрезентації	Модуль класифікації
Класифікація перифраз	Модуль векторизації	Векторні репрезентації	Оцінка перифрази	Вихідний інтерфейс

Відповідно до функціональної моделі, кожен модуль інтелектуальної системи інкапсулює окрему задачу, що забезпечує локальність обчислень, модульність архітектури та можливість незалежного удосконалення компонентів без порушення цілісності системи. Типи задач, що виконуються кожним модулем, наведено у таблиці 3.2.

Таблиця 3.2

Модулі інформаційної системи та задачі, що ними розв’язуються

Модуль	Тип задачі
Попередня обробка	Нормалізація та переклад вхідних речень
Генерація AMR графів	Формування семантичної структури та абстрагування змісту від синтаксису

Продовження таблиці 3.2

Модуль	Тип задачі
Векторизація графів	Побудова векторних представлень та моделювання семантичних залежностей
Класифікація перифраз	Порівняння концептуальних структур та визначення перифраз

Виходячи зі структури системи, представленої на рисунку 3.1, та змісту задач, інкапсульованих у модулях, інтелектуальна система виконує такі основні функції:

– приймання, попередня обробка, нормалізація та зберігання текстових даних, включно з перетворенням вхідних висловлень на уніфіковане лінгвістичне подання, необхідне для подальшої семантичної інтерпретації;

– формування абстрактних семантичних репрезентацій висловлень на основі AMR графів, що забезпечує відокремлення змісту від поверхневої форми, збереження предикатно-аргументної структури та концептуальних залежностей;

– векторизація семантичних структур шляхом перетворення AMR графів у щільні вкладення, що відображають топологію графа, семантичні ролі та контекстуальні зв’язки;

– оцінювання ступеня семантичної еквівалентності між висловленнями, яке включає порівняння векторних представлень та формування класифікаційного висновку щодо наявності перифраз;

– організація внутрішніх інформаційних потоків і керування взаємодією модулів, включно з обробкою проміжних результатів, синхронізацією задач та контролем цілісності даних.

– архівування, документування та представлення результатів аналізу, зокрема збереження семантичних графів, векторних репрезентацій та висновків

моделі у форматі, придатному для подальшої інтерпретації, візуалізації та інтеграції із зовнішніми системами.

3.3. Модуль підготовки текстових даних для генерації AMR графів

Ефективність побудови абстрактних семантичних репрезентацій значною мірою залежить від якості попередньої обробки текстових даних, що передують етапу AMR парсингу. Оскільки сучасні AMR моделі переважно орієнтовані на англійську мову, виникає необхідність забезпечити коректний переклад українськомовного вхідного речення до англомовної форми, яка зберігатиме семантичну структуру та предикатно-аргументні відношення [66]. Для розв’язання цієї задачі у складі інформаційної технології передбачено модуль багатомовної попередньої обробки, що реалізує алгоритми нормалізації тексту та нейронного машинного перекладу.

Структурно цей модуль становить перший етап обчислювального конвеєра і виконує декілька взаємопов’язаних функцій. На початковій стадії модуль забезпечує стандартизацію вхідних текстів, що дозволяє усунути надлишкову поверхневу варіативність, після чого текст подається на переклад.

У структурі модуля особливе місце займає нейронна модель машинного перекладу, на якій базується перехід від українськомовного висловлення до англомовної форми, придатної для подальшої генерації AMR графів. У межах інформаційної технології застосовуються сучасні трансформерні моделі перекладу, здатні не лише відтворювати зміст на рівні поверхневої лексики, а й зберігати глибинну предикатно-аргументну організацію висловлення. На відміну від традиційних статистичних методів, ці моделі формують контекстно чутливі векторні представлення речення, що дозволяє передавати семантичні ролі й логічні залежності між концептами з високим ступенем точності.

У рамках інформаційної технології розглядаються три багатомовні моделі, що поєднують високу якість відтворення смислу та ефективність перекладу:

1. OPUS-MT – модель на базі трансформерної архітектури, оптимізована для українсько-англійських пар речень на основі корпусів OPUS. Модель характеризується високою швидкістю роботи та збереженням синтаксичних зв’язків у коротких і середніх реченнях [67].

2. M2M-100 – багатомовна модель прямого перекладу. Завдяки великому обсягу тренувальних даних M2M-100 добре передає семантичні ролі та залежності, що є суттєвим для подальшого AMR парсингу [68].

3. NLLB-200 – оптимізована версія моделі No Language Left Behind, що підтримує переклад між 200 мовами. Попри зменшений розмір, модель демонструє високі показники збереження змісту за оцінкою BERTScore[69].

Схематичну структуру модуля нейронного машинного перекладу наведено на рисунку 3.3. Модуль складається з двох функціональних етапів: нормалізації вхідної текстової послідовності та нейронного машинного перекладу, який здійснює проєкцію україномовного висловлення в англомовний простір. Після проходження крізь модуль багатомовної підготовки нормалізований англомовний текст подається на вхід AMR парсера, що забезпечує перехід від поверхневої мовної форми до абстрактної змістової структури у вигляді AMR графа.

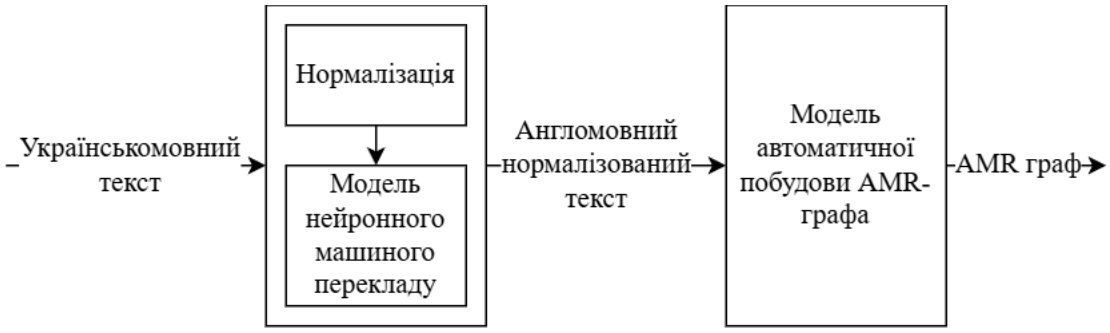

Рис. 3.3. Архітектура модуля нейронного машинного перекладу та нормалізації вхідних речень

Функціональна організація модуля багатомовної підготовки даних відповідає вимогам AMR парсингу та побудована з метою мінімізації впливу

вхідних варіацій на подальший семантичний аналіз. На першому етапі висловлення проходить процедуру нормалізації, яка охоплює технічні операції вирівнювання пробілів, уніфікації символів, корекції службових знаків та усунення артефактів форматування. Після виконання нормалізації відкоригований текст подається до нейронної системи машинного перекладу, яка формує його англомовний еквівалент. Узгоджений та нормалізований перекладений текст передається до модуля генерації AMR графів.

Таким чином, інформаційна технологія багатомовної підготовки виконує роль проміжної ланки у семантичному конвеєрі інтелектуальної системи. Забезпечуючи точне відтворення смислової організації українськомовних висловлень англійською мовою, вона створює передумови для коректної побудови AMR графів та аналізу семантичної еквівалентності висловлень. Експериментальна оцінка якості моделей та дослідження впливу перекладу на збереження семантичної повноти тексту наведені у розділі 4.

3.4. Модуль генерації AMR графів на основі нейронної моделі AMRBART

Після етапу підготовки текстових даних система переходить до побудови абстрактних семантичних репрезентацій висловлень у форматі AMR. Цей етап є центральним у структурі інформаційної технології, оскільки саме AMR граф містить узагальнену предикатно-аргументну структуру висловлення та формує основу для подальшої векторизації й порівняння змісту речень.

Сучасні підходи до генерації AMR графів демонструють, що найвищої точності в задачах парсингу досягають нейронні моделі, що базуються на трансформерній архітектурі. У межах даної технології реалізовано модуль генерації AMR графів, який ґрунтується на нейронній моделі AMRBART [51]. Ця модель є модифікацією трансформерної архітектури BART, адаптованої для задачі AMR парсингу шляхом тонкого настроювання на корпусах AMR 2.0 [70] та AMR 3.0 [71]. Архітектура AMRBART складається з наступних компонентів:

– кодувальник – виконує багаторівневе кодування вхідного англомовного речення, формує контекстуальні представлення кожного токена, захоплюючи синтаксичні та семантичні залежності;

– декодер – генерує лінійне представлення AMR графа у форматі PENMAN та використовує механізм перехресної уваги до прихованих станів кодувальника, а також механізм самоуважності для формування послідовності графових токенів [72];

– семантичні примітиви – модель використовує словник концептів, ролей та структурних елементів AMR, сформований під час навчання на корпусах AMR.

Завдяки спеціалізованим механізмам обробки іменованих сутностей, моделі ролей та концептів, а також підтримці структурних шаблонів AMR, модель здатна відтворювати складні предикативні конструкції та структуру аргументів. Це дозволяє мінімізувати помилки втрати смислових компонентів та забезпечити високий рівень відповідності графа оригінальному змісту висловлення. Для стабілізації процесу генерації використовується багатоваріантне декодування, а обробка результату включає перевірку коректності вкладених структур, збалансованості дужок та формалізованих семантичних залежностей.

Модуль генерації AMR графів є ключовим елементом інформаційної технології, оскільки забезпечує перехід від текстової форми висловлення до формалізованої предикатно-аргументної структури. У межах запропонованої архітектури система розглядає AMR граф як обчислювальну репрезентацію смислу висловлень, на основі якої здійснюється їхнє подальше зіставлення.

У структурному аспекті модуль організований як самостійна функціональна компонента, що очікує у вигляді вхідних даних англомовні речення. Формат вхідних даних реалізовано у вигляді текстової послідовності без додаткових анотацій, що забезпечує сумісність модуля з різними моделями машинного перекладу та спрощує його інтеграцію. Після отримання вхідних

даних нейронна модель AMRBART реалізує двоступеневий механізм обробки. У цьому процесі формується узгоджена семантична репрезентація, що зберігає структуру предикатів, ролей, кореферентних зв’язків та логічних відношень між концептами. Архітектура модуля наведена на рисунку 3.4.

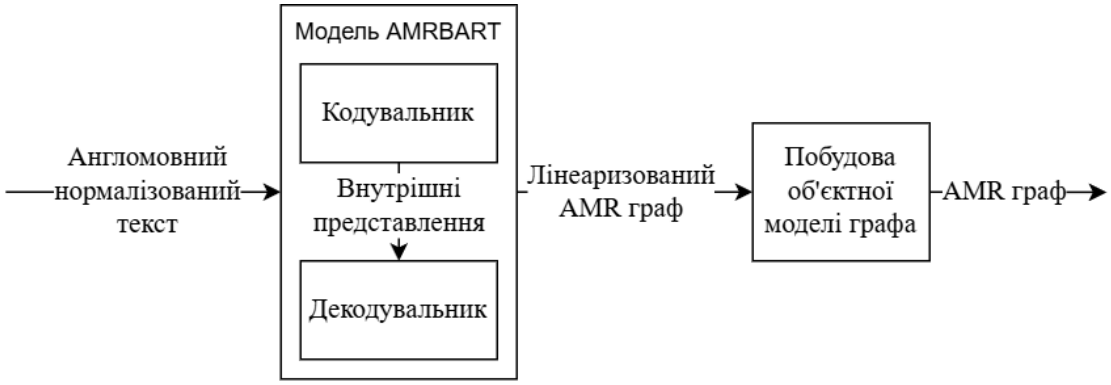

Рис. 3.4. Архітектура модуля трансформації англомовного тексту в AMR граф

Оскільки результат роботи AMRBART має форму текстової послідовності, модуль включає етап обробки, що забезпечує перетворення PENMAN рядку у графове подання. На цьому етапі здійснюється побудова об’єктної моделі графа, де кожний концепт представлено як вершину, а відношення між концептами – як ребро з роллю. Такий формат є необхідним для узгодженої взаємодії з модулем графової векторизації, який очікує на вхід структурний граф, а не його лінійне представлення.

Функціональна організація модуля передбачає також виконання низки процедур виправлення та нормалізації графів. Під час генерації AMR графів можливе виникнення технічних артефактів, пов’язаних із синтаксичною складністю або неоднозначністю вхідного тексту. Тому в модулі реалізовано механізми перевірки коректності структури, відновлення кореневого вузла у випадку його втрати, усунення незамкнених або фрагментарних підграфів та приведення ролей до канонічної нотації AMR 3.0. Ці процедури не змінюють

зміст висловлення, але забезпечують цілісність та сумісність отриманих графів із модулем побудови графових вкладень.

Важливим етапом є обробка виняткових ситуацій. У разі, якщо AMRBART не може сформувати коректний граф, модуль фіксує помилку, здійснює повторну спробу генерації зі зміненими параметрами. Такий підхід забезпечує стійкість обчислювального конвеєра до семантичного шуму та варіативності вхідних даних, особливо тих, що походять з автоматичного перекладу.

Завершальним етапом роботи модуля є передавання сформованого графа на вхід модуля побудови вкладень, який здійснює векторизацію семантичної структури. Таким чином, модуль генерації AMR графів виконує роль зв’язувальної ланки між мовною формою висловлення та його обчислювальним векторним поданням, забезпечуючи формування представлення для подальшої класифікації перифраз.

3.5. Модуль формування графових вкладень на основі AMR графів

Модуль кодування AMR графів у векторне представлення є третім елементом інформаційної технології, що забезпечує формування векторних репрезентацій семантичних структур. На цьому етапі здійснюється трансформація графів, отриманих у результаті роботи модуля AMR парсингу, у багатовимірні вектори, які відображають предикатно-аргументну організацію висловлень і дозволяють системі виконувати подальше порівняння змісту незалежно від синтаксису.

Структурно модуль реалізований за архітектурою сіамської нейронної мережі [73]. Основна ідея цього підходу полягає у тому, що два вхідних графи незалежно проходять через ідентичні нейронні підмережі з однаковими параметрами, після чого отримані вкладення порівнюються за допомогою певної міри подібності. Така архітектура дозволяє навчати модель таким чином, щоб семантично близькі графи мали близькі вектори, а різні за змістом – віддалені.

Архітектура сіамської мережі є загальновизнаним рішенням у задачах пошуку схожості між об’єктами, зокрема у задачах порівняння речень, пошуку перифраз або дублювання змісту [74]. Перевагою цього підходу є те, що модель навчається безпосередньо на відстані між парами прикладів у спільному латентному просторі, а не на класових мітках. Це забезпечує вищу узагальнювальну здатність при роботі з графовими структурами, які не мають строгої ізоморфної відповідності, але можуть бути семантично еквівалентними. Архітектура модуля векторизації AMR графів наведена на рисунку 3.5.

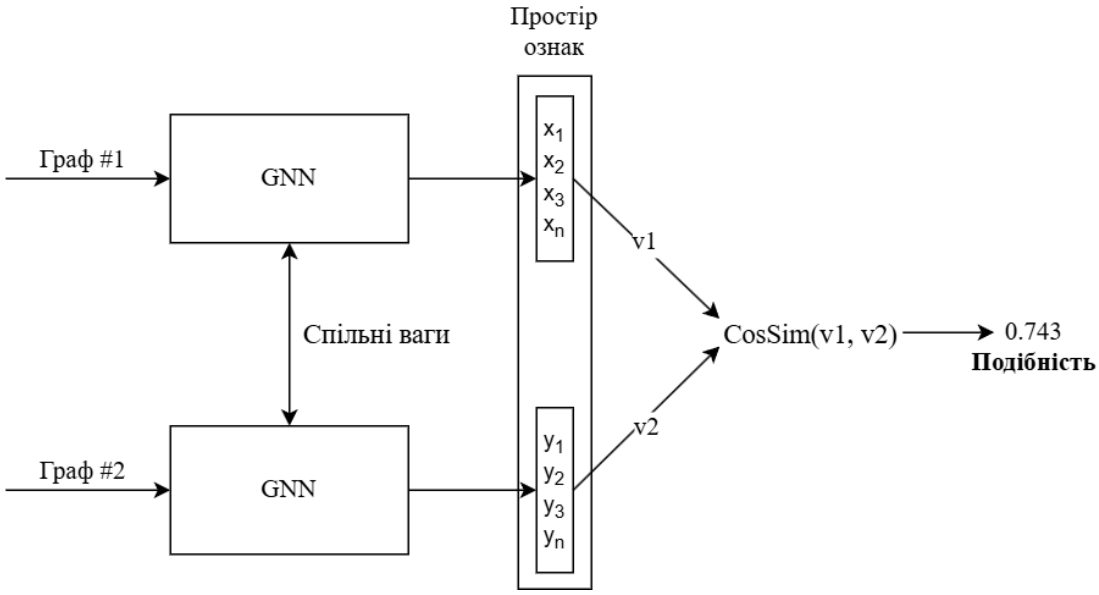

Рис. 3.5. Архітектура сіамської нейронної мережі для побудови векторних репрезентацій двох графів

Процес побудови векторних репрезентацій AMR графів у межах розробленого модуля передбачає кілька послідовних етапів, які забезпечують перехід від символьного графового подання до числового простору вкладень, придатного для порівняння за мірою семантичної подібності.

На початковому етапі вхідні дані у форматі об’єктної моделі графа проходять процес декодування. У результаті здійснюється реконструкція топології графа, тобто відновлення множини вузлів, що репрезентують концепти, та ребер, які відображають структуру смислових зв’язків між

концепами. Далі кожен вузол графа асоціюється з набором числових ознак, які відображають різні аспекти його семантики та структури. До складу вектора ознак можуть входити:

– вкладення концепту, що відображає лексичне значення концепту в багатовимірному просторі;

– вкладення типу семантичного зв’язку, яке враховує роль вузла у структурі відношень;

– позиційні та топологічні ознаки, що описують місце вузла у графі.

Таке попереднє представлення забезпечує збереження не лише локальної семантики вузлів, а й глобальної структури графа, що є критичним для точного моделювання відношень між концептами.

Отримана репрезентація подається на вхід графовій нейронній мережі типу GATv2Conv, яка формує контекстуальні вкладення вузлів за допомогою механізму адаптивної уваги (рис. 3.6). На відміну від згорткових моделей зі статичною агрегацією, GATv2Conv обчислює вагові коефіцієнти для кожного сусіда окремо, враховуючи не лише тип ребра, але й конкретний зміст вузлів, між якими встановлено зв’язок [75]. Це дозволяє моделі динамічно визначати, які елементи AMR графа мають найбільше семантичне значення для конкретного контексту, що особливо важливо у графах з нерівномірним розподілом інформації, складними вкладеними структурами та численними ролями.

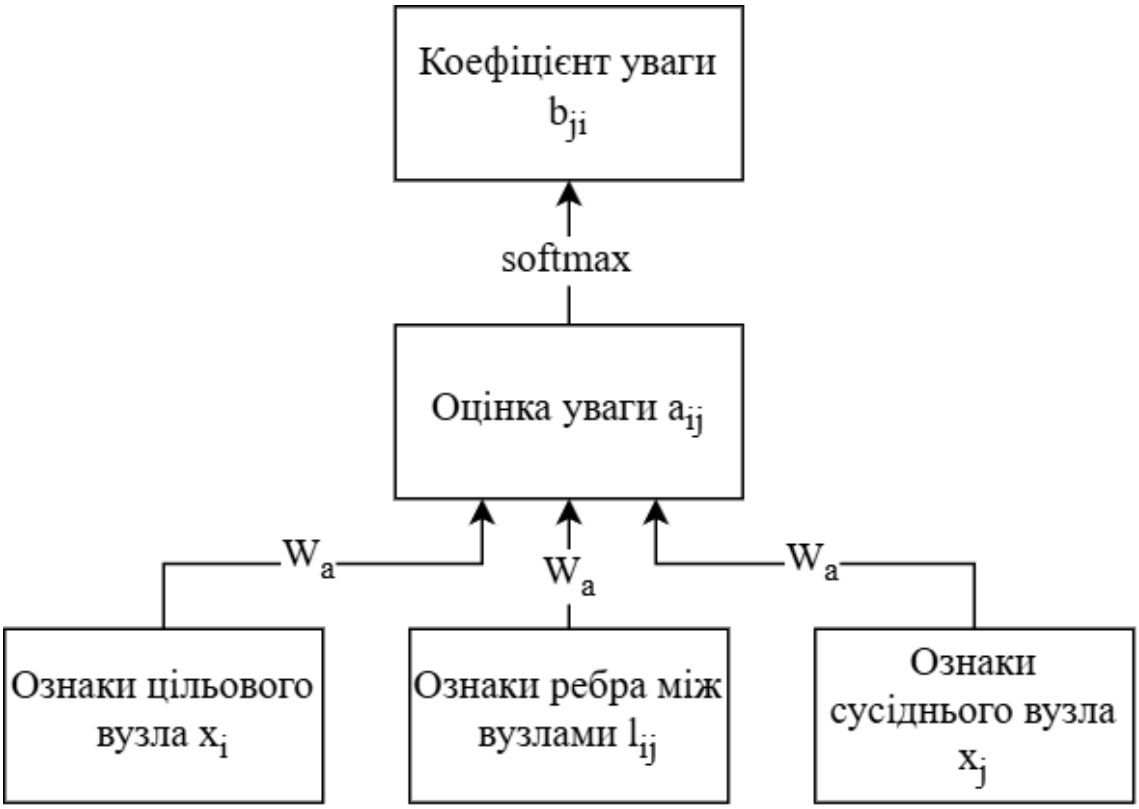

Рис. 3.6. Процес обчислення вагових коефіцієнтів ребер у графовій мережі з механізмом уваги

Функціонально для кожного вузла обчислюється вектор, що інтегрує інформацію як про сам вузол, так і про його сусідів.

$$h_i' = \sum_{j \in N(i)} \alpha_{ij} W h_j, \tag{3.1}$$

де α_{ij} – коефіцієнти уваги, W – матриця лінійного перетворення, h_j – вектор ознак сусіднього вузла.

Завдяки механізму уваги мережа динамічно визначає вагу кожного зв’язку, віддаючи перевагу більш релевантним у поточному контексті. Після проходження кількох шарів GATv2Conv вектори вузлів агрегуються у єдину графову репрезентацію. Для цього застосовуються методи агрегування, які дозволяють інтегрувати інформацію з усіх вузлів у компактне, але інформативне векторне представлення графа [76].

У межах даної інформаційної технології використано механізм агрегування на основі механізму уваги, оскільки він забезпечує найбільш гнучку та семантично обґрунтовану агрегацію інформації у випадку AMR графів [77]. На відміну від традиційних методів, таких як агрегування за середнім значенням або агрегування за максимумом, де вклад кожного вузла у фінальне подання визначається рівномірно або за простими евристиками, агрегування на основі механізму уваги динамічно обчислює вагу кожного вузла залежно від його інформативності для глобальної смислової структури.

За аналогією з механізмом уваги при формуванні репрезентацій вузла у графовій нейронній мережі, агрегування формує вагові коефіцієнти, які відображають, наскільки сильно кожен вузол впливає на формування підсумкового вкладення графа. Для вузла і коефіцієнт уваги визначається за формулою:

$$\beta_i = softmax(q^T \tanh(W_p h_i)), \tag{3.2}$$

де q – навчальний вектор запиту, що визначає семантичну важливість ознак, W_p – навчальна матриця вагових коефіцієнтів для проєкції ознак вузла у латентний простір; h_i – вкладення і вузла, отримане на попередніх шарах графової нейронної мережі.

Це дозволяє посилити вплив ключових елементів AMR графа і водночас обмежити внесок другорядних модифікаторів.

Навчання сіамської нейронної мережі здійснюється шляхом оптимізації відстані між векторними репрезентаціями двох AMR графів. У межах даної інформаційної технології використовується косинусна подібність [78] як основна метрика, що відображає семантичну близькість у високорозмірному просторовому представленні вкладень та обчислюється за формулою:

$$CosSim(g_1, g_2) = \frac{g_1 \cdot g_2}{\|g_1\| \|g_2\|}, \tag{3.3}$$

де g_1 та g_2 – векторні репрезентації графів.

Процес навчання полягає у мінімізації косинусної подібності для пар графів, що позначені як перифрази, та її максимізації для графів, що не є перифразами. Таким чином, модель формує простір, у якому близькі за смислом висловлення розташовуються близько один до одного, а висловлення з різним змістом – віддаляються одне від одного. Цей підхід забезпечує інваріантність до стилістичних відмінностей та локальних трансформацій графа, що дає змогу моделі навчатися саме на рівні глибинних семантичних відповідностей.

Результатом роботи модуля графового кодування є векторна репрезентація AMR графа. На відміну від поверхневих мовних представлень, отриманий вектор зберігає предикатно-аргументну організацію, рольові відношення, логічні зв’язки та структурну ієрархію графа. Завдяки використанню багатошарової GNN-архітектури отримане вкладення включає як локальні характеристики окремих вузлів, так і глобальні шаблони графової структури.

Отримана векторна репрезентація є універсальною, оскільки перетворює різнорідні графові структури у єдиний порівнюваний простір. Така форма подання даних забезпечує можливість подальшого застосування математичних методів оцінки відстані між графами, процедур оптимізації, а також її використання у модулі класифікації, який завершує конвеєр системи.

У загальній архітектурі інтелектуальної технології модуль графового кодування виконує роль центральної ланки, що забезпечує перехід від символічної семантики AMR до числового представлення, придатного для машинного навчання. Застосування цього модуля забезпечує перехід від структурної графової репрезентації до операційного рівня обчислень, що формує основу для фінального етапу – ухвалення рішення щодо семантичної еквівалентності висловлювань.

3.6. Модуль класифікації перифраз на основі семантичних векторних представлень

Модуль класифікації перифраз забезпечує перехід від векторних семантичних репрезентацій до формального рішення про семантичну еквівалентність двох текстових висловлень. Його функціональне призначення полягає у визначенні ступеня подібності між двома векторними поданнями графа та у віднесенні пари висловлень до класу «перифраза» або «не перифраза». У контексті когнітивної моделі, цей модуль виконує функцію інтерпретаційного механізму, який порівнює дві смислові структури та ухвалює остаточний висновок щодо їх відповідності.

Структура класифікатора реалізована у вигляді багатошарового перцептрона із вентильним механізмом фільтрації ознак (рис. 3.7) [79]. Така структура поєднує можливості глибинної нелінійної трансформації та адаптивного керування потоком інформації між шарами, що дозволяє моделі зберігати релевантні ознаки та усувати шум. Загалом модуль складається з трьох основних блоків:

1. Блок первинної обробки ознак, у якому комбінований вектор проходить лінійне перетворення, нормалізацію та нелінійну активацію. На цьому етапі відбувається первинне узагальнення ознак і формування прихованого представлення.

2. Вентильний механізм забезпечує фільтрацію ознак на основі показника косинусної подібності. Висока подібність векторів зумовлює високі значення активації вентиля, завдяки чому семантично значущі ознаки передаються на наступні рівні обробки без змін. Натомість за умови низької подібності механізм зменшує вагу відповідних ознак, що мінімізує ймовірність хибнопозитивних класифікацій. Таким чином вентильний механізм зберігає лише ті компоненти ознакового простору, що характеризуються високим ступенем семантичної відповідності.

3. Блок вторинного узагальнення, призначений для виділення високорівневих класифікаційних характеристик після фільтрації. Цей блок завершує формування узагальненої семантичної репрезентації, необхідної для прийняття класифікаційного рішення.

Фінальний шар моделі виконує лінійне відображення отриманого представлення в логіт, який після перетворення сигмоїдною функцією формує оцінку ймовірності перифрази.

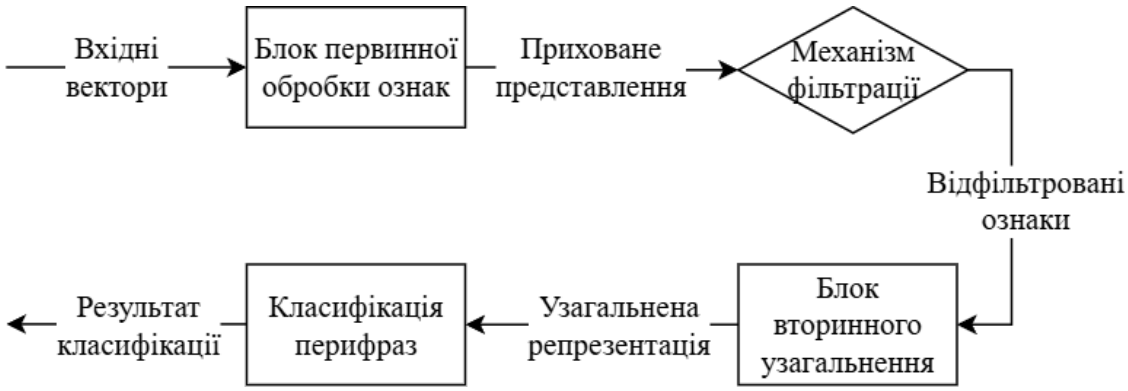

Рис. 3.7. Архітектура класифікатора перифраз із механізмом керованої селекції ознак

Функціонально модуль класифікації перифраз виконує перехід від абстрактних графових репрезентацій, поданих у вигляді векторних вкладень, до фінального рішення про семантичну відповідність двох висловлень. Його основне призначення полягає у трансформації отриманих ознак у числову оцінку еквівалентності, що реалізується шляхом аналізу багатовимірної конфігурації двох векторів та виявлення специфічних шаблонів взаємодії між ними.

На вхід модуль отримує два вектори однакової розмірності, сформовані графовим кодувальником на попередньому етапі. Модуль здійснює інтеграцію кількох типів ознак для формування розширеного вектора x_0:

$$x_0 = [u, v, |u - v|, u \odot v], \tag{3.4}$$

де u, v – вхідні векторні представлення AMR графів; |u - v| – вектор абсолютної різниці елементів, що моделює локальну семантичну дивергенцію; u Θ v – елементний добуток, що відображає локальну кореляцію ознак.

Такий підхід дозволяє системі поєднати локальні й глобальні характеристики семантичного збігу: локальні відповідають відображенню подібності структурних фрагментів AMR графів, тоді як глобальні узагальнюють відстань між висловленнями у всьому семантичному просторі. Незалежно від ознакового простору додатково обчислюється косинусна подібність між векторами. Вона використовується на етапі фільтрації ознак та виступає критерієм їх інформативності.

Модуль функціонує як багатошарова нейронна мережа, що включає двоступеневу обробку ознак. Перший етап відповідає за формування проміжного латентного простору:

$$h_1 = Dropout(GELU(BN(W_1 x_0 + b_1))), \quad (3.5)$$

де W_1, b_1 – навчальні параметри лінійного шару, BN – шар пакетної нормалізації; GELU – функція активації, Dropout – механізм регуляризації для запобігання перенавчанню.

На цьому етапі структуровані ознаки, отримані з вихідних векторів, проходять через блок лінійних перетворень, нормалізацію та нелінійну активацію.

На другому етапі на основі значення косинусної подібності за допомогою сигмоїдального перетворення обчислюється коефіцієнт:

$$g = \sigma(CosSim(u, v)), \quad (3.6)$$

де σ – сигмоїдна функція активації; CosSim(u, v) – значення косинусної подібності між вхідними векторами.

Даний коефіцієнт використовується у вентильному механізмі для фільтрації ознак. Стан простору ознак при цьому оновлюється за формулою:

$$\widetilde{h_1} = g \cdot h_1, \tag{3.7}$$

де g – вентильний коефіцієнт.

Висока семантична близькість між висловленнями збільшує пропускну здатність вентиля, тоді як низька – зменшує вагу відповідних ознак. Такий механізм забезпечує адаптивність моделі до різних типів внутрішньої семантичної організації AMR графів.

Після вентильної фільтрації модуль формує цілісний вектор ознак, який надходить до кінцевого шару класифікації. На цьому завершальному етапі виконується лінійне перетворення до одновимірного логіта за формулою:

$$z = W_o\widetilde{h_1} + b_o, \tag{3.8}$$

де, W_o, b_o – параметри вихідного шару.

Отримане числове значення з формули (3.8) визначає ступінь належності двох висловлень до класу перифраз. Для отримання ймовірності наявності перифрази, до логіта застосовується сигмоїдне перетворення:

$$p = \sigma(z), \tag{3.9}$$

У межах класифікаційного модуля ймовірність p не використовується як безпосередній клас, а передається на етап порогового прийняття рішення. Оскільки в задачах семантичної еквівалентності межа між класами не є чіткою оптимальний поріг τ підбирається шляхом максимізації міри F1:

$$\tau^* = arg \max_{\tau \in [0,1]} F_1(\tau), \tag{3.10}$$

де τ – поріг прийняття рішення, що варіюється в діапазоні [0, 1], $F_1(\tau)$ – значення F_1 міри, обчислене для конкретного значення порогу τ.

Після обчислення оптимального порога τ класифікатор інтерпретує значення p як:

– перифраз, якщо ймовірність семантичної відповідності перевищує значення порогу,

– не перифраз, якщо p нижче оптимального порогу.

У загальній структурі інформаційної технології цей модуль виконує ключову функцію прийняття рішення щодо семантичної еквівалентності. Він завершує обчислювальний конвеєр, інтегруючи як інформацію про локальні семантичні відповідності, так і глобальний контекст, сформований графовими нейронними мережами. У когнітивній моделі це відповідає фінальному етапу процесу розуміння – зіставленню концептуальних структур і прийняттю рішення на основі їхньої відповідності. Таким чином, модуль класифікатора забезпечує аналіз глибинної семантичної узгодженості висловлень, що необхідно для точного виявлення перифраз.

3.7. Інструменти практичної реалізації та алгоритм роботи інтелектуальної системи ідентифікації перифраз

Програмна реалізація розробленої інформаційної технології була здійснена як модульна система, у якій кожен компонент архітектури відповідає окремому етапу обробки даних – перекладу, генерації AMR графів, побудови графових вкладень та класифікації семантичної еквівалентності. Реалізація виконана з використанням сучасних бібліотек глибинного навчання та інструментів роботи з графовими структурами, що забезпечує масштабованість, відтворюваність експериментів та високу ефективність обчислень. Для розробки інтелектуальної системи ідентифікації перифраз застосовано мову

програмування Python, що на сьогодні є стандартом для побудови систем штучного інтелекту та обробки природної мови. Python забезпечує широкий спектр інструментів для реалізації глибоких нейронних мереж, графових обчислень, обробки текстових даних і розробки модульних інформаційних технологій, що обумовило його вибір як основної платформи програмної імплементації.

У рамках системи використано фреймворк PyTorch, який забезпечує автоматичне диференціювання, гнучке керування обчислювальними графами та можливість ефективного використання пристроїв CUDA [80]. Для реалізації графових нейронних мереж застосовано бібліотеку PyTorch Geometric, яка містить оптимізовані реалізації GNN-шарів, операцій агрегації та алгоритмів пакетної обробки нерегулярних графових структур [81].

Модуль багатомовного перекладу реалізовано через API HuggingFace Transformers, що дозволяє використовувати попередньо навчені моделі та оптимізувати їх обчислення у середовищі PyTorch [82]. Модуль побудови AMR графів інтегрує модель AMRBART, яка використовується в режимі автогенерації PENMAN-представлень. Обробка AMR графів здійснюється через бібліотеки Penman та amrlib, які забезпечують перетворення графів у формати, сумісні з PyTorch Geometric.

Для класифікації перифраз побудовано MLP-класифікатор із використанням фреймворку глибинного навчання PyTorch. Цей вибір зумовлений його гнучкістю під час створення архітектур нейронних мереж та високою швидкістю обчислення тензорних операцій на GPU. Усі компоненти реалізації організовано таким чином, щоб модуль класифікації був ізольований від графового кодувальника, що дозволяє експериментувати з різними моделями векторизації без зміни класифікаційної частини.

Алгоритм роботи інтелектуальної системи зображений на рис. 3.8. На початковому етапі системний контролер приймає на вхід пару текстових висловлень українською мовою та формує запит до модуля багатомовної підготовки даних. У цьому модулі кожне висловлення незалежно передається до

нейронної моделі машинного перекладу, яка формує англомовні інтерпретації. Після отримання перекладу здійснюється нормалізація тексту, що усуває технічні артефакти форматування. Нормалізовані англомовні речення повертаються контролеру як проміжні представлення, придатні для подальшого AMR парсингу.

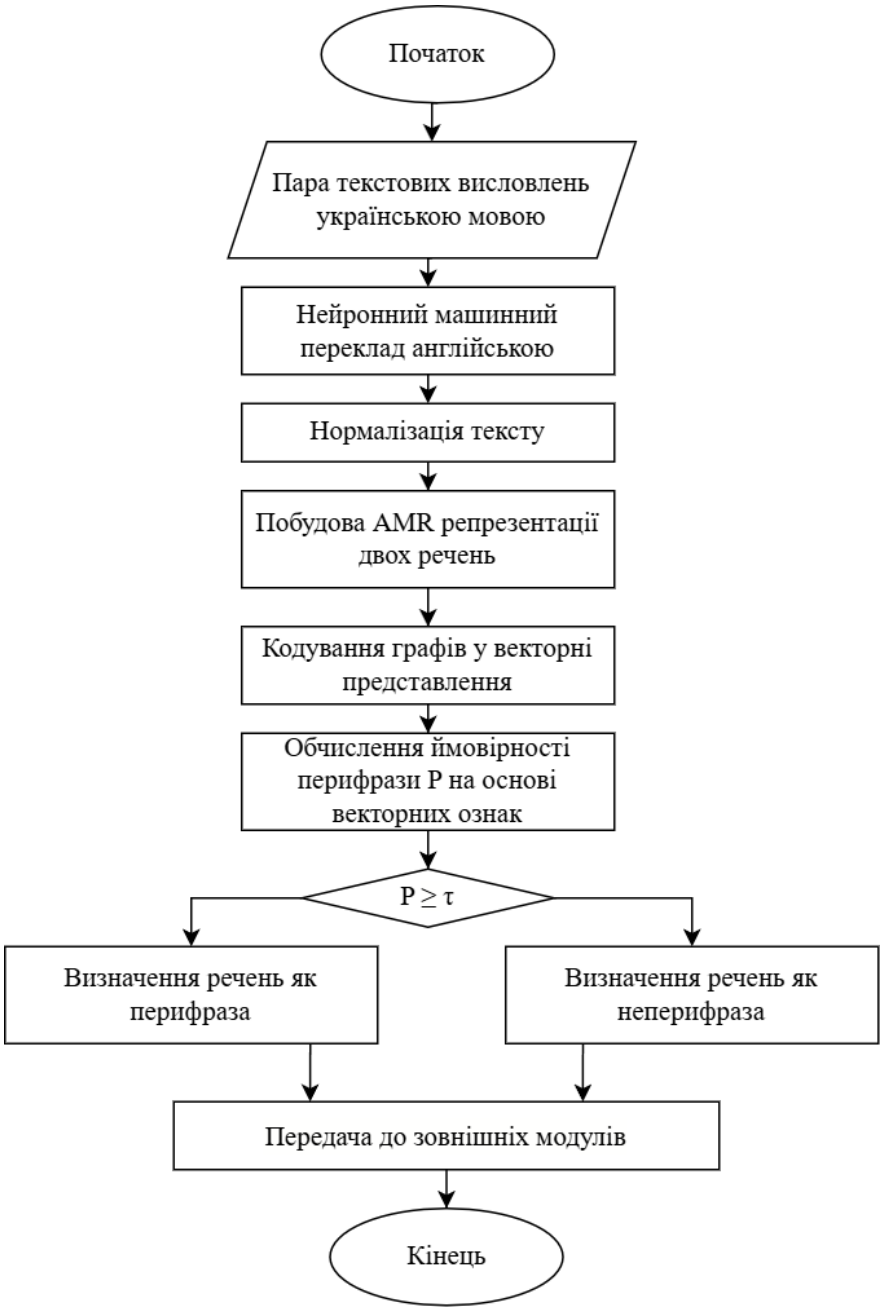

Рис. 3.8. Блок-схема алгоритму функціонування інтелектуальної системи ідентифікації перифраз

На наступному етапі системний контролер послідовно передає обидва речення до модуля генерації AMR графів. Нейронна модель AMRBART кодує кожне речення та генерує відповідне лінеаризоване представлення AMR графа, яке потім перетворюється у графовий формат. У результаті формується пара абстрактних семантичних структур, що репрезентують глибинний зміст висловлень у вигляді предикатно-аргументних графів.

Після побудови AMR графів системний контролер ініціює етап графового кодування. Обидві графові структури передаються до модуля векторизації, де послідовність шарів графової нейронної мережі GATv2Conv формує контекстуальні вкладення вузлів. Далі застосовується агрегування з використанням механізму уваги, яке формує вкладення графа фіксованої розмірності. На виході цього етапу система отримує пару векторних представлень, що відображають семантичний зміст кожного з висловлень у спільному просторі.

Наступним кроком є класифікація семантичної еквівалентності. Системний контролер передає пару вкладень до модуля класифікації перифраз, де формується розширений вектор ознак. Багатошарова нейронна мережа класифікатора виконує послідовні лінійні перетворення з нелінійною активацією, у результаті чого формується вихідний логіт, який визначає ступінь перифрази. Логіт перетворюється на ймовірнісну оцінку за допомогою сигмоїдної функції, після чого застосовується адаптивний поріг, визначений на етапі валідації за критерієм максимізації міри F1. Таким чином, модуль класифікації повертає не лише числове значення ймовірності, а й бінарне рішення про наявність або відсутність перифрази.

Фінальним етапом алгоритму є інтерпретація та маршрутизація результатів системним контролером. На основі отриманої ймовірності та бінарного рішення контролер формує узагальнений вихідний запис, який може включати: пари вхідних висловлень, їхні AMR подання, вкладення графів, значення ймовірності семантичної еквівалентності та остаточний класифікаційний висновок. Цей результат передається зовнішньому інтерфейсу

або наступним аналітичним модулям, а за потреби – може використовуватися для накопичення статистики, аналізу похибок і подальшого тонкого настроювання моделі.

З метою забезпечення зручної взаємодії з розробленими алгоритмами та для проведення подальших експериментальних досліджень, програмну реалізацію системи було розширено до архітектури повноцінного клієнт-серверного веб-застосунку (рис. 3.9). Серверна частина інкапсулює всі обчислювальні процеси та надає доступ до них через спеціалізований REST API. Такий архітектурний підхід дозволяє абстрагувати складність математичних обчислень і вимоги до апаратного забезпечення від кінцевого користувача. Клієнтська частина реалізована у вигляді сучасного односторінкового веб-застосунку і виконує роль інтерактивного дослідницького середовища.

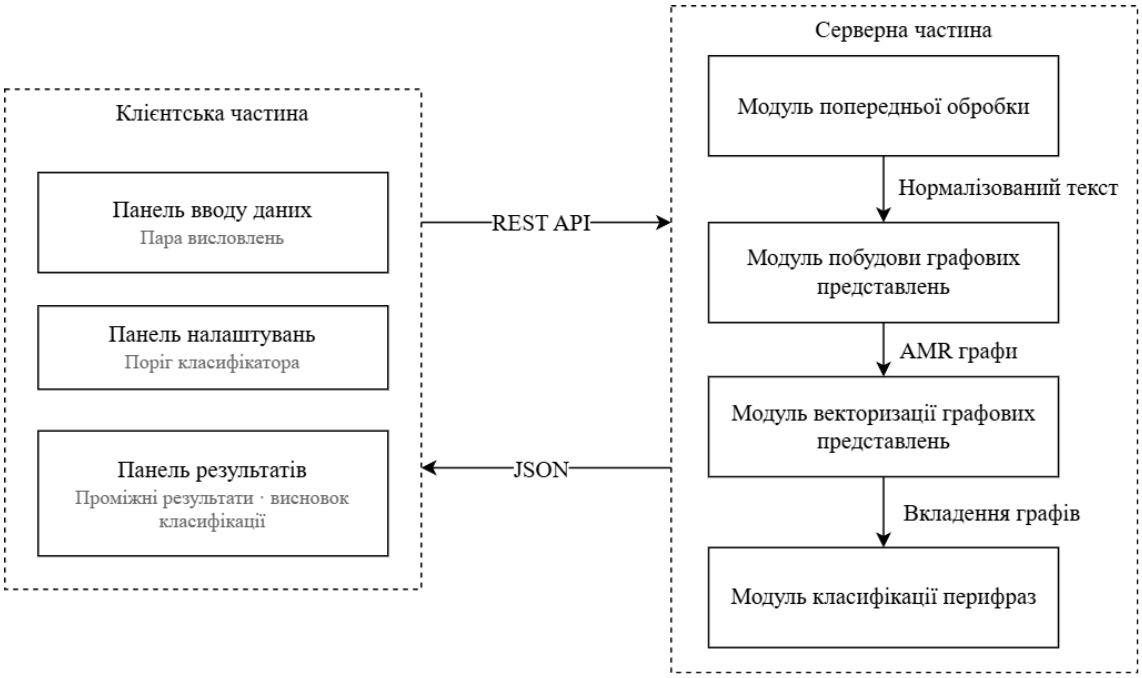

Рис. 3.9. Архітектура клієнт-серверного веб-застосунку

Головним користувачем системи є дослідник, який ініціює процеси автоматизованого аналізу текстових пар. Усі запити користувача проходять через API-шлюз, що виконує маршрутизацію даних до серверних обчислювальних модулів. Ключовою вимогою до логіки системи є забезпечення

розширеного аналізу результатів: алгоритм побудовано таким чином, щоб надавати досліднику доступ як до фінального висновку, так і до проміжних метрик на кожному етапі обчислювального конвеєра.

Для практичної реалізації зазначених вимог, передбачено такий базовий покроковий сценарій взаємодії дослідника з програмним середовищем:

1. Користувач здійснює доступ до клієнтського застосунку та вводить пару україномовних висловлень у відповідні текстові поля головної панелі інтерфейсу.

2. За необхідності дослідник налаштовує параметри системи. Зокрема, задається поріг чутливості для фінального класифікатора, що дозволяє регулювати строгість критеріїв оцінювання семантичної еквівалентності.

3. Користувач ініціює процес ідентифікації. Клієнтський застосунок формує запит та через API-шлюз передає дані на сервер для послідовного виконання завдань нейронного машинного перекладу, AMR парсингу та класифікації перифраз.

4. Інформаційна панель у режимі реального часу відображає згенеровані системою англомовні відповідники, а також виводить проміжні кількісні метрики побудованих абстрактних семантичних репрезентацій.

5. Після завершення графового кодування та класифікації досліднику надається остаточний бінарний висновок щодо наявності перифраз між вхідними текстами, який супроводжується загальним відсотком впевненості моделі.

Загальний вигляд та графічний інтерфейс розробленого застосунку, що забезпечує виконання описаного сценарію взаємодії, показано на рис. 3.10.

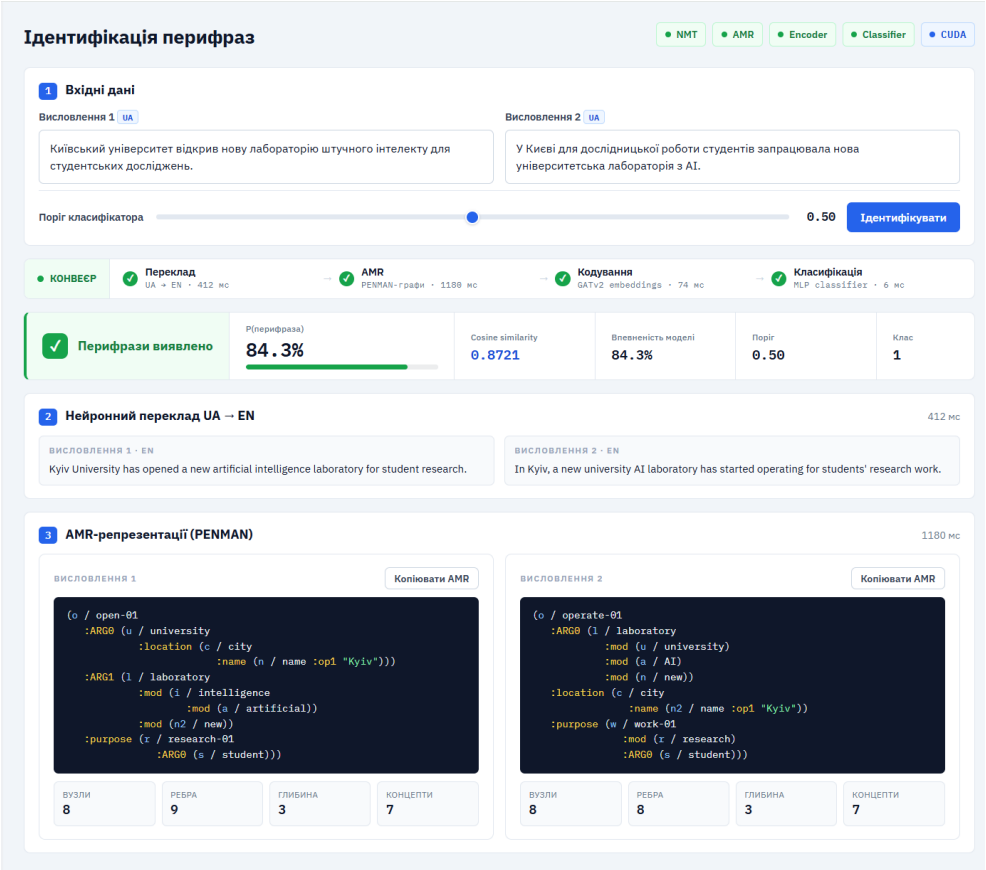

Рис. 3.10. Графічний інтерфейс клієнтської частини веб-застосунку для ідентифікації перифраз

Таким чином, структура та алгоритм роботи інтелектуальної системи забезпечують узгоджений перехід від природномовного вхідного сигналу до формалізованого рішення про перифраз, в основу якого покладено порівняння абстрактних семантичних репрезентацій висловлень у векторному просторі. Це дозволяє реалізувати інформаційну технологію, яка не лише виконує автоматичну ідентифікацію перифраз, але й узгоджується з теоретичною моделлю когнітивного процесу розуміння.

3.8. Висновки за розділом 3

1. У третьому розділі розроблено інформаційну технологію ідентифікації перифраз у текстах природної мови. Запропонована технологія відтворює ключові механізми семантичної обробки мовлення – від первинного

сприйняття висловлення до прийняття висновку про його смислову еквівалентність – у вигляді багаторівневого семантичного конвеєра, що інтегрує нейронні моделі машинного перекладу, побудову AMR графів, графові нейронні мережі та адаптивні класифікаційні методи.

2. Розроблено архітектуру інтелектуальної системи, яка містить послідовно пов’язані модулі: підготовки текстових даних, побудови AMR графів за допомогою нейронної моделі AMRBART, кодування семантичних структур за допомогою GNN-архітектур, та класифікації перифраз. Визначено структуру інформаційної технології, логіку взаємодії між модулями та функціональне призначення кожного компонента у системі. Модульна організація системи створює можливість незалежного налаштування її складників.

3. Розроблено алгоритмічне та програмне забезпечення модуля підготовки текстових даних, що є першим етапом обчислювального конвеєра системи. Модуль виконує нормалізацію та автоматизований переклад українськомовних висловлень англійською мовою із застосуванням сучасних трансформерних моделей нейронного машинного перекладу. Такий підхід дозволяє зберегти глибинну предикатно-аргументну організацію висловлень та створює необхідні передумови для подальшої коректної генерації абстрактних семантичних структур.

4. Розроблено алгоритмічне та програмне забезпечення модуля генерації абстрактних семантичних структур. Модуль забезпечує перехід від текстової форми висловлення до формалізованої предикатно-аргументної структури у вигляді семантичного AMR графа. У межах цього компонента інтегровано нейронну модель AMRBART, яка здатна ефективно відтворювати складні предикативні конструкції та структуру аргументів речень завдяки підтримці структурних шаблонів AMR.

5. Розроблено алгоритмічне та програмне забезпечення модуля отримання векторних представлень AMR графів із застосуванням графових нейронних мереж та механізму уваги. Модуль побудовано за архітектурою сіамської нейронної мережі, що передбачає незалежне обчислення двох вхідних

графів підмережами зі спільними параметрами. Така структурна організація дозволяє отримувати репрезентації, що відображають як локальні, так і глобальні властивості смислової структури висловлювань, гарантуючи зближення векторів для семантично еквівалентних графів та їх віддалення для різних за змістом.

6. Розроблено алгоритмічне та програмне забезпечення модуля класифікації перифраз, який виконує роль інтерпретаційного механізму для переходу від векторних семантичних репрезентацій до формального рішення про еквівалентність текстів. Структура класифікатора базується на архітектурі багатошарового перцептрона, доповненого вентильним механізмом керованої селекції ознак. Запропонований підхід забезпечує моделювання нелінійних залежностей між ознаками графових семантичних репрезентацій та прийняття рішення щодо ступеня семантичної еквівалентності між висловленнями.

7. Розроблено алгоритм функціонування інтелектуальної системи та ключові особливості її програмної реалізації, включно з вибором мовних і програмних засобів, структур даних та методів оптимізації навчання нейронних моделей. Для взаємодії з системою реалізовано програмний комплекс у вигляді клієнт-серверного застосунку, що виконує роль інтерактивного дослідницького середовища. Запропонований архітектурний підхід інкапсулює розроблені обчислювальні модулі на серверній стороні та надає доступ до них через спеціалізований REST API. Це забезпечує зручну взаємодію з системою та створює програмну платформу для проведення подальших експериментальних досліджень.

Таким чином, у третьому розділі розроблено архітектуру, визначено структуру та здійснено програмну реалізацію інтелектуальної системи, яка здатна виконувати автоматичну ідентифікацію перифраз на основі глибинних семантичних репрезентацій. Отримані результати створюють основу для подальших експериментальних досліджень, наведених у четвертому розділі дисертації.

РОЗДІЛ 4
ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ІДЕНТИФІКАЦІЇ ПЕРИФРАЗ НА ОСНОВІ ГРАФОВИХ РЕПРЕЗЕНТАЦІЙ

4.1. Методика експериментального дослідження інформаційної технології ідентифікації перифраз на основі графових репрезентацій

Експериментальний етап дослідження спрямований на верифікацію ефективності розробленої інформаційної технології у розділі 3. Методика експерименту охоплює наступні взаємопов’язані рівні аналізу: оцінювання коректності проміжних етапів обробки та фінальне оцінювання класифікаційного модуля.

У межах експериментального дослідження розроблено набір процедур, що забезпечують верифікацію кожного компоненту обчислювального конвеєра. Для цього було визначено такі цілі:

– оцінити здатність багатомовних моделей перекладу зберігати семантичну цілісність висловлення як передумову успішної AMR інтерпретації;

– визначити точність та стійкість моделі AMRBART при формуванні предикатно-аргументної структури у вигляді AMR графа;

– дослідити властивості векторних репрезентацій, сформованих графовими нейронними мережами;

– оцінити здатність класифікаційного модуля розпізнавати семантичну еквівалентність висловлень.

Методика експерименту ґрунтується на принципах відтворюваності, модульності та наскрізного контролю якості. Кожен етап перевіряється зіставленням із відповідними еталонними даними або із загальноприйнятими метриками. Так, якість перекладу оцінюється на основі BLEU та BERTScore; точність AMR парсингу – за допомогою метрики Smatch; ефективність графових

векторних репрезентацій – через кореляційні метрики Пірсона та Спірмена; якість класифікації – за показниками точності та міри F1.

Дослідження реалізовано у середовищі Python з використанням бібліотек PyTorch, PyTorch Geometric, HuggingFace Transformers, SentencePiece, scikit-learn та інструментів візуалізації Matplotlib і Seaborn [83, 84]. Експериментальні обчислення виконувались на графічному прискорювачі NVIDIA серій RTX, що забезпечило можливість опрацювання великих корпусів та навчання моделей.

Для проведення експериментальних досліджень було відібрано набори даних, які репрезентують різні рівні семантичної обробки в межах розробленої інформаційної технології – від етапу багатомовного перекладу до задачі класифікації перифраз. Навчання та валідація моделей, описаних у розділі 3, здійснювалися на корпусах, релевантних до відповідних функціональних модулів системи та сформованих для розв’язання задач нейронного перекладу, побудови абстрактних семантичних репрезентацій, формування графових векторних представлень і оцінювання семантичної еквівалентності висловлень. Для кожного етапу обробки використовувалися незалежні тренувальні та тестові вибірки, що забезпечувало об’єктивність оцінювання між фазами навчання й валідації.

Такий підхід до формування експериментальної бази дозволяє забезпечити узгодженість між функціональною організацією інформаційної технології та схемою експериментальної перевірки, а також гарантує коректність інтерпретації отриманих результатів для кожного з її структурних компонентів.

4.2. Експериментальна перевірка ефективності нейронних моделей машинного перекладу зберігати семантичну повноту текстових даних при підготовці до AMR парсингу

Багатомовний переклад є першим етапом у розробленій інформаційній технології. Якість та семантична повнота перекладеного тексту безпосередньо визначають коректність подальшої генерації AMR графів. Оскільки формалізм

AMR орієнтований переважно на англомовні тексти, українськомовні вхідні висловлення попередньо перекладаються англійською мовою за допомогою моделей нейронного машинного перекладу.

У межах експериментального дослідження було здійснено порівняльний аналіз відомих багатомовних моделей нейронного машинного перекладу, а саме OPUS-MT, M2M-100 та NLLB-200. Обрані моделі репрезентують різні підходи до побудови універсальних нейронних перекладачів і відрізняються як обсягом покриття мов, так і глибиною контекстного моделювання.

OPUS-MT реалізує підхід багатомовного перекладу на основі паралельних корпусів OPUS та характеризується високою ефективністю для мовних пар із наявними навчальними даними. M2M-100 є універсальною трансформерною моделлю, що навчена без використання англійської як проміжної мови, що дозволяє забезпечити пряму міжмовну семантичну трансляцію. NLLB-200 орієнтована на покриття низькоресурсних мов і використовує уніфікований багатомовний простір представлень, що забезпечує підвищену стійкість до мовної варіативності. Усі зазначені моделі застосовувалися у режимі попередньо навчених нейронних трансформерних архітектур без додаткового тонкого налаштування на спеціалізованих AMR корпусах.

Для експериментальної перевірки якості перекладу було використано українсько-англійський корпус Flores-200, який містить тематично збалансовані тексти різних стилістичних регістрів [85]. Даний корпус характеризується наявністю еталонних перекладів, синтаксичною та семантичною різнорідністю висловлень, а також достатньою кількістю прикладів для статистично обґрунтованого оцінювання. Узагальнені характеристики використаного набору даних наведено в таблиці 4.1, а схему його поділу на навчальну, валідаційну та тестову вибірки – у таблиці 4.2. Зазначений поділ забезпечує коректне порівняльне оцінювання альтернативних моделей перекладу в однакових умовах.

Таблиця 4.1

Загальні характеристики набору даних Flores-200

Характеристика	Значення
Тип корпусу	Паралельний
Мовна пара	Українська-англійська
Кількість пар речень	200000
Стиль текстів	Публіцистика, офіційні тексти

Таблиця 4.2

Структура набору даних Flores-200 за типами вибірок та обсягом даних

Вибірка	Кількість	Частка
Навчальна	180000	90%
Валідаційна	10000	5%
Тестова	10000	5%

Оцінювання якості нейронного перекладу здійснювалося з використанням двох метрик – BLEU та BERTScore [86, 87]. Метрика BLEU ґрунтується на зіставленні n-грам перекладеного тексту з еталоном і формально визначається як середнє геометричне значення точності n-грам з урахуванням штрафу за надмірну стислість:

$$BLEU = BP \cdot exp(\sum_{n=1}^{N} w_n \log p_n), \tag{4.1}$$

$$BP = \begin{cases} 1, & \text{якщо } c > r \\ \exp\left(1 - \frac{r}{c}\right), & \text{якщо } c \leq r \end{cases}, \tag{4.2}$$

де p_n – точність n-грам, w_n – вагові коефіцієнти для n-грам, N – максимальна довжина n-грам, BP – штраф за стислість, що обчислюється за формулою (4.2), c – довжина перекладеного тексту, r – довжина еталонного

тексту. Дана метрика відображає ступінь лексико-синтаксичної відповідності, однак не завжди коректно враховує глибинну семантичну еквівалентність.

BERTScore, на відміну від BLEU, базується на контекстуальних представленнях слів, отриманих із трансформерної мовної моделі, та обчислює узгодженість між еталонним і згенерованим перекладами у семантичному просторі:

$$BERTScore = \frac{1}{|x|}\sum_{x_i \in x} \max_{y_j \in y} cosine(v_{x_i}, v_{y_i}), \quad (4.3)$$

де x та y – множини токенів еталонного та згенерованого речень відповідно, v_{x_i} та v_{y_i} – контекстуальні вектори токенів.

В межах проведеного експерименту окрему увагу було приділено особливостям перекладу з української мови. Оскільки українська мова належить до мов із розвиненою морфологією, вільним порядком слів та високим рівнем синтаксичної варіативності, це зумовлює труднощі при відтворенні нейронними моделями перекладу предикатно-аргументної структури висловлення. До таких проблем, наприклад, можна віднести втрату дійової особи, спрощення відмінкових зв’язків або неправильне відображення завершеності дії. З огляду на це, пріоритет при виборі моделі надавався показнику BERTScore. Застосування BERTScore є важливим у контексті підготовки даних до AMR парсингу, оскільки саме ця метрика дозволяє кількісно оцінити збереження смислу незалежно від поверхневої варіативності перекладу.

Результати досліджень ефективності нейронних моделей перекладу українсько-англійської мови представлені у таблиці 4.3.

Таблиця 4.3

Порівняльна характеристика ефективності нейронних моделей перекладу для українсько-англійських мовних пар

Модель	Кількість параметрів, млн.	BLEU	BERTScore
OPUS-MT	390	26.5	0.93
M2M-100	615	26.6	0.94
NLLB-200	600	31.2	0.96

Аналіз отриманих результатів показує, що OPUS-MT забезпечує оптимальне поєднання високої якості перекладу для української мови та швидкості обчислень, що робить її найбільш ефективною для інтеграції у систему AMR парсингу. Показники BLEU та BERTScore демонструють, що модель добре зберігає семантичну структуру речень при перекладі, що критично для подальшого побудування точних AMR графів. Хоча M2M-100 та NLLB-200 демонструють вищі оцінки якості, їх великі розміри та менша продуктивність ускладнюють практичне використання у багатокрокових конвеєрах семантичного аналізу.

Таким чином, результати експериментальних досліджень підтвердили доцільність використання моделей нейронного машинного перекладу в складі інформаційної технології підготовки текстових даних до AMR парсингу, оскільки вони забезпечують ефективний міжмовний переклад зі збереженням ключових семантичних характеристик висловлень. Застосування нейронного перекладу як проміжного етапу дозволяє підвищити коректність подальшої генерації абстрактних семантичних репрезентацій та забезпечує практичну придатність розробленої технології для обробки текстів українською мовою.

4.3. Експериментальна перевірка якості побудови абстрактних семантичних репрезентацій у форматі AMR

На етапі генерації абстрактних семантичних репрезентацій здійснюється перехід від поверхневої мовної форми до формалізованої предикатно-аргументної структури висловлення. Якість AMR графів безпосередньо визначає ефективність подальших етапів графового кодування та класифікації перифраз. У зв’язку з цим особливе значення має експериментальна перевірка точності та стійкості автоматичного AMR парсингу.

Для генерації абстрактних семантичних репрезентацій було використано спеціалізовану наперед треновану модель AMRBART. Зазначена модель була тонко налаштована на корпусі AMR 3.0, який є одним з еталонних наборів даних для задач абстрактного семантичного аналізу. Корпус містить речення англійською мовою з ручною розміткою у форматі AMR, що охоплює як прості, так і синтаксично та семантично складні конструкції. Характеристики використаного набору даних наведено в таблиці 4.4

Таблиця 4.4

Загальні характеристики набору даних AMR 3.0

Характеристика	Значення
Мова текстів	Англійська
Тип даних	Пари речень та анотованих AMR графів
Тип анотації	Експертна ручна
Загальна кількість речень	59 255
Формат зберігання	PENMAN
Стиль текстів	Новини, форуми, публіцистика

Структурно кожен приклад у корпусі представлений реченням та відповідним йому еталонним AMR графом, сформований з залученням експертів-лінгвістів. Графи містять вузли, що відповідають семантичним концептам, та ребра, які кодують предикатно-аргументні, модальні, темпоральні та інші відношення. Такий формат даних дозволяє здійснювати як навчання нейронних моделей генерації графів, так і їх об'єктивне порівняльне оцінювання, шляхом порівняння з еталонними значеннями. З метою експериментальних досліджень корпус поділено на навчальну, валідаційну та тестову вибірки, як це показано у таблиці 4.5.

Таблиця 4.5

Структура набору даних AMR 3.0 за типами вибірок та обсягом даних

Вибірка	Кількість	Частка
Навчальна	55635	93%
Валідаційна	1722	3,5%
Тестова	1898	3,5%

Додатково обрана модель була перевірена на незалежному корпусі AMR 2.0. Корпус AMR 2.0 є попередником AMR 3.0 і містить його концептуальну та методологічну основу. Він також містить англомовні речення, для яких вручну побудовано відповідні AMR графи експертами-лінгвістами. Використання AMR 2.0 як незалежної тестової вибірки для моделі, навченої на AMR 3.0, дозволяє перевірити стійкість сформованих семантичних репрезентацій до зміни джерел даних. Характеристики набору даних та схему його поділу на навчальну, валідаційну та тестову вибірки наведено відповідно у таблицях 4.6 та 4.7.

Таблиця 4.6

Загальні характеристики набору даних AMR 2.0

Характеристика	Значення
Мова текстів	Англійська
Тип даних	Пари речень та анотованих AMR графів
Тип анотації	Експертна ручна
Загальна кількість речень	39260
Формат зберігання	PENMAN
Стиль текстів	Новини, форуми, публіцистика

Таблиця 4.7

Структура набору даних AMR 2.0 за типами вибірок та обсягом даних

Вибірка	Кількість	Частка
Навчальна	36521	93%
Валідаційна	1368	3,5%
Тестова	1371	3,5%

Основною метрикою оцінювання якості AMR парсингу у дослідженні було обрано Smatch – загальноприйнятий стандарт для порівняння AMR графів [88]. Smatch здійснює зіставлення двох графів шляхом пошуку оптимального вирівнювання між їх вузлами та обчислення міри F1 на основі кількості спільних триплетів, що складаються з двох вузлів та зв’язку між ними. Значення метрики інтерпретується як міра структурно-семантичної подібності між автоматично згенерованим і еталонним графом:

$$Smatch = \frac{2 \cdot |M|}{|G_{sys}| + |G_{gold}|}, \tag{4.4}$$

де |M| – кількість спільних триплетів, $|G_{sys}|$ та $|G_{gold}|$ – кількість триплетів у згенерованому та еталонному графах відповідно.

Значення Smatch варіюється від 0 до 100, де 100 відповідає повному збігу. Перевагою Smatch є його інваріантність до ідентифікаторів вузлів та здатність коректно відображати як точність відтворення окремих семантичних відношень, так і цілісної структури графа. Це робить його релевантним інструментом для оцінювання якості AMR у контексті подальшого графового кодування.

Експериментальне дослідження передбачало порівняння якості AMR графів, згенерованих моделлю AMRBART, з результатами, отриманими за допомогою альтернативних підходів, серед яких використовувалась модель SPRING – нейронна модель AMR парсингу трансформерного типу. Результати експериментів наведені у таблиці 4.8.

Таблиця 4.8

Порівняльна оцінка якості генерації семантичних графів

Модель	Набір даних	Smatch
AMRBART	AMR 2.0	85.4
	AMR 3.0	84.2
SPRING	AMR 2.0	82.8
	AMR 3.0	83

Результати, викладені у таблиці 4.8, показують, що AMRBART перевищує результати базових моделей попередніх поколінь. У порівнянні зі SPRING, модель AMRBART забезпечує більш точне відтворення вкладених предикатно-аргументних структур та зберігає семантичну узгодженість при аналізі довгих речень зі складною синтаксичною організацією.

Таким чином, експериментальні дані підтверджують доцільність використання моделі AMRBART як базового модуля генерації абстрактних семантичних репрезентацій у розробленій інформаційній технології, оскільки

вона поєднує високу точність AMR парсингу та стійкість до зміни корпусів навчання й оцінювання.

4.4. Експериментальне дослідження впливу механізму уваги у графових нейронних мережах при побудові векторних репрезентацій AMR графів

Даний підрозділ присвячено експериментальному дослідженню розробленого модуля побудови вкладень AMR графів, який реалізує перехід від структурованих абстрактних семантичних репрезентацій до їх векторних репрезентацій, придатних для подальшого машинного зіставлення та класифікації. Особливу увагу приділено впливу механізму уваги у складі графових нейронних мереж на якість формування векторних репрезентацій абстрактних семантичних структур.

Алгоритм навчання модуля кодування AMR графів у межах даного дослідження побудований за двоетапною схемою послідовної оптимізації, що поєднує структурне самонавчання та наступне тонке настроювання векторного простору. Така організація навчального процесу зумовлена необхідністю, з одного боку, сформувати у графової нейронної мережі чутливість до внутрішньої топології AMR структур, а з іншого – забезпечити узгодженість векторних представлень із експертними уявленнями про семантичну подібність.

На першому етапі навчання здійснюється попереднє самонавчання графового кодувальника на задачі реконструкції ребер AMR графа. Для кожного графа випадковим чином маскується частина семантичних зв’язків, після чого нейронна мережа навчається відновлювати їх на основі локальних та контекстуальних представлень вузлів. На другому етапі виконується тонке налаштування моделі на задачі семантичної текстової подібності. Вкладення, сформовані на основі AMR графів двох висловлень, відображаються у спільний векторний простір, де оптимізація здійснюється за різницею між прогнозованою косинусною подібністю та еталонними значеннями семантичної близькості.

Обидва етапи реалізуються в межах сіамської архітектури з розділеними вагами кодувальника для кожного графа, що гарантує інваріантність моделі до порядку подачі вхідних графів і забезпечує відображення всіх AMR структур у єдиний узгоджений простір ознак. Оптимізація параметрів виконується градієнтними методами з використанням адаптивних оптимізаторів, що забезпечує стійкість процесу навчання та запобігає перенавчанню.

Для реалізації двоетапного навчання модуля графового кодування AMR графів було використано два набори даних, що відповідають різним рівням обробки: корпус AMR 3.0 для етапу структурного самонавчання та корпус STS Benchmark для етапу тонкого настроювання моделі.

На етапі реконструкції ребер для попереднього навчання графового кодувальника використовувався корпус AMR 3.0. Як було зазначено вище, цей корпус містить анотовані AMR графи для англомовних речень, що робить його репрезентативним джерелом для навчання структурних закономірностей предикатно-аргументних семантичних репрезентацій.

На другому етапі здійснювалося тонке настроювання шляхом розв’язання задачі семантичної текстової подібності з використанням корпусу STS Benchmark, запропонованого у роботі Д. Сера та співавторів [89]. Задача семантичної текстової подібності (STS) – це задача кількісного визначення ступеня семантичної подібності між двома текстовими одиницями. STS Benchmark є еталонним набором даних для задачі та широко використовується для валідації моделей семантичної близькості. Характеристики набору даних наведено у таблиці 4.9, а схема поділу на вибірки – у таблиці 4.10.

Таблиця 4.9

Загальні характеристики набору даних STS Benchmark

Характеристика	Значення
Мова текстів	Англійська
Тип даних	Пари речень та їх оцінки семантичної подібності
Тип анотації	Експертна ручна
Загальна кількість речень	8628
Основні джерела текстів	Новини, форуми, описи, енциклопедичні тексти

Таблиця 4.10

Структура набору даних STS Benchmark за типами вибірок та обсягом даних

Вибірка	Кількість	Частка
Навчальна	5749	66,6%
Валідаційна	1500	17,4%
Тестова	1379	16,0%

Набір даних містить 8 628 пар речень, кожна з яких має експертну оцінку подібності у шкалі від 0 до 5, де:

– 0 – речення не мають спільного змісту;

– 1-2 – мають незначні спільні фрагменти або тематику;

– 3-4 – частково збігаються за змістом, але відрізняються деталями;

– 5 – повністю еквівалентні за значенням.

Розподіл оцінок у корпусі є рівномірним, що забезпечує можливість навчання моделі розрізняти як повні перифрази, так і часткову семантичну схожість.

З метою дослідження впливу механізму уваги на якість формування векторних репрезентацій AMR графів експериментальні дослідження були

спроєктовані як серія контрольованих порівняльних експериментів. Усі обчислювальні експерименти проводилися для двох архітектур графових нейронних мереж – R-GCN [90] та GATv2Conv [75] – у поєднанні з двома різними схемами агрегації вузлових представлень: агрегація по середньому та агрегація з використанням механізму уваги. Така експериментальна конфігурація дозволила здійснити кількісну оцінку внеску механізму уваги як на рівні оновлення вузлових ознак у процесі графових згорток, так і на етапі глобальної агрегації інформації у фіксований вектор графа.

Для експериментального дослідження було використано нейронну мережу, побудовану у розділі 3 з використанням мови програмування Python та спеціалізованих бібліотек для глибинного навчання й обробки графових структур. Реалізація включала модулі формування ознак вузлів, графові згорткові шари, механізми агрегації та оптимізації параметрів у межах сіамської архітектури. Навчання нейронної мережі здійснювалося з використанням адаптивних методів оптимізації, що забезпечувало стійкість збіжності та запобігало перенавчанню.

Основні характеристики побудованої нейронної мережі наведено у таблиці 4.11, яка відображає конфігурацію графового кодувальника, параметри навчання, типи агрегації та налаштування оптимізації.

Таблиця 4.11

Конфігурація архітектури та параметри навчання модуля графового кодування

Параметр	Значення	Опис
Кількість шарів мережі	2	Глибина графового кодувальника
Типи шарів	R-GCN, GATv2Conv	Порівняння згорткової реляційної архітектури та архітектури з механізмом уваги

Продовження таблиці 4.11

Параметр	Значення	Опис
Розмірність початкових ознак вузлів	1024	Розмір вектора початкових вкладень вузлів вхідного графа
Розмірність прихованого шару	512	Розмір вектора прихованого стану ознак нейронної мережі
Розмірність вихідного вкладення графа	256	Розмір вихідного вектора ознак AMR графа
Типи агрегації	Global Mean Pooling, Attention Pooling	Алгоритм формування глобальної репрезентації графа
Регуляризація	Вилучення нейронів	Алгоритм запобігання перенавчанню
Оптимізація	Adam	Адаптивний метод оптимізації навчання
Швидкість навчання	$1 \cdot 10^{-4}$	Крок оптимізації
Розмір пакета	256	Підмножина набору даних, що використовується під час епохи навчання

Перший етап навчання полягає у мінімізації похибки BCEWithLogitsLoss. Метою цього етапу є навчити модель точно відновлювати структуру графа, тобто передбачати наявність або відсутність ребра між кожною парою вузлів. Вибір функції BCEWithLogitsLoss обґрунтовано необхідністю стабілізації градієнтів за рахунок логарифмування правдоподібності [91]. Для кожної потенційної зв’язки модель прогнозує ймовірність існування ребра між вузлами z_{ij} та мінімізує

значення функції втрат між нею та цільовим значенням y_{ij}, яке показує чи існує ребро в графі. Функція втрат визначається як:

$$\mathcal{L}_{BCE} = -\frac{1}{N}\sum_{i,j}\left[y_{ij}\log(\sigma(z_{ij})) + (1-y_{ij})\log(1-\sigma(z_{ij}))\right], \quad (4.5)$$

де N – кількість екземплярів графів; y_{ij} – істинна мітка наявності ребра між вузлами і та j, $\sigma(z_{ij})$ – прогнозована ймовірність існування ребра.

На другому етапі навчання над вихідними даними графових шарів проводиться операція агрегування, після якої кожен граф отримує векторне представлення фіксованої розмірності. Навчання на другому етапі фокусується на мінімізації функції втрат Cosine Sentence Loss, яка вимірює різницю між прогнозованою косинусною подібністю векторів для пари графів j, k та еталонними значеннями схожості та є стандартом для оцінки семантичної близькості в задачах STS [92]. Функція втрат визначається як:

$$\mathcal{L}_{COSSent} = \frac{1}{M}\sum_{j,k}\left(cosine(v_j, v_k) - s_{jk}\right)^2, \quad (4.6)$$

де M – загальна кількість пар графів; v_j, v_k – векторні представлення графів j та k, $cosine(v_j, v_k)$ – косинусна подібність між векторами, s_{jk} – еталонне значення семантичної схожості для пари графів j, k.

Крім того, на цьому етапі модель вчиться максимізувати коефіцієнти кореляції Пірсона та Спірмена, які оцінюють відповідність прогнозованих значень семантичної близькості фактичним. Коефіцієнт Пірсона оцінює лінійну залежність між прогнозованими та анотованими значеннями схожості та визначається за наступною формулою [93]:

$$r = \frac{\sum_{j,k}(\hat{s}_{jk}-\bar{\hat{s}})(s_{jk}-\bar{s})}{\sqrt{\sum_{j,k}(\hat{s}_{jk}-\bar{\hat{s}})^2 \sum_{j,k}(s_{jk}-\bar{s})^2}}, \quad (4.7)$$

де $\hat{s}_{jk}$ – прогнозована косинусна подібність для пари графів j та k, s_{jk} – еталонне значення семантичної схожості для пари графів j, k, $\bar{\hat{s}}$ та $\bar{s}$ – середні значення прогнозованих та еталонних оцінок по всій вибірці відповідно. Високе значення метрики свідчить, що модель правильно масштабує прогнозовані подібності.

Своєю чергою коефіцієнт Спірмена оцінює, наскільки зберігається ранг семантичної близькості між прогнозованими та анотованими значеннями та визначається наступним чином [94]:

$$\rho = 1 - \frac{6\sum_{j,k} d_{jk}^2}{M(M^2-1)}, \tag{4.8}$$

де d_{jk} – різниця між рангами прогнозованого та еталонного значень для пари графів j та k, M – загальна кількість пар графів. Високий показник метрики означає, що модель правильно відтворює порівняльну семантичну близькість між усіма парами.

Аналіз навчання моделі на першому етапі показав швидке засвоєння базових структурних закономірностей без вираженого перенавчання з досягненням стабільних показників ROC-AUC [95]. Динаміка навчання розроблених графових моделей на другому етапі відображена у вигляді порівняльних кривих зміни кореляційних коефіцієнтів Пірсона та Спірмена на валідаційній вибірці (рис. 4.1 та рис. 4.2). Наведені графіки дають змогу проаналізувати характер збіжності оптимізаційного процесу, вплив обраних архітектурних рішень та стратегій агрегації векторних репрезентацій на кінцеву якість моделі.

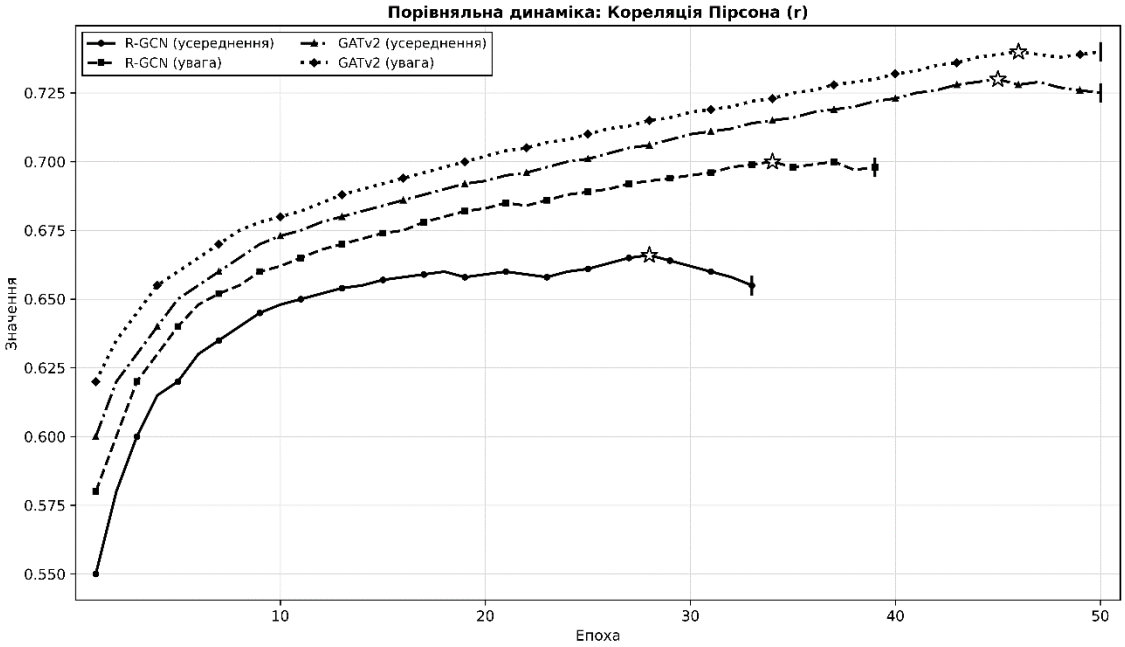

Рис. 4.1. Динаміка зміни коефіцієнта Пірсона на валідаційній вибірці під час навчання моделей

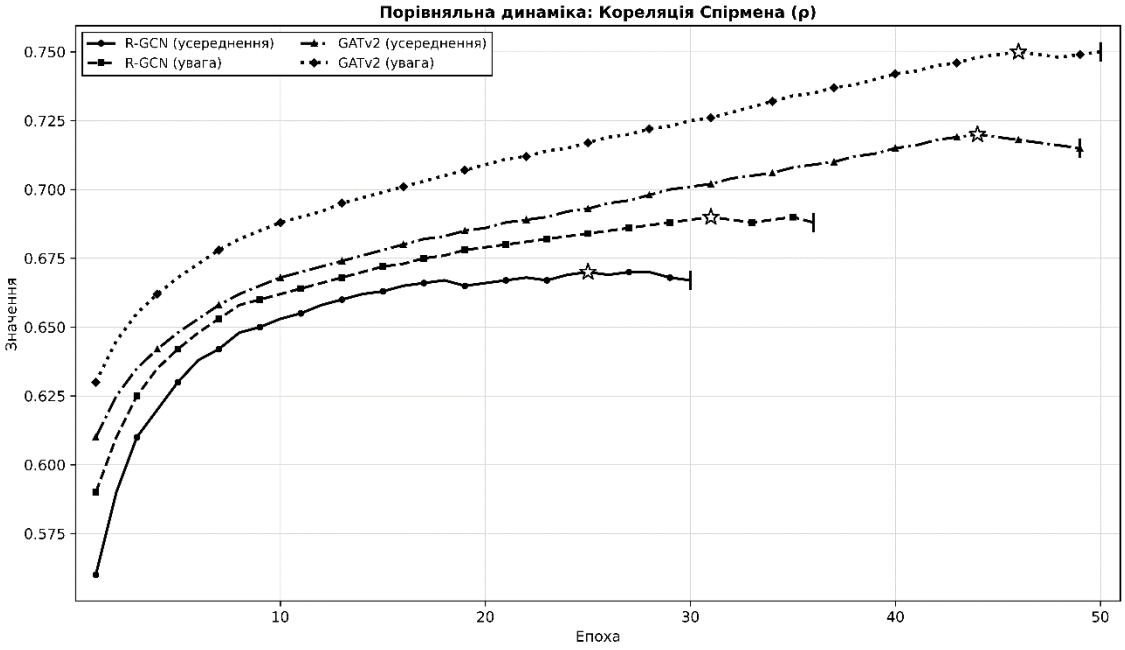

Рис. 4.2. Динаміка зміни коефіцієнта Спірмена на валідаційній вибірці під час навчання моделей

Візуалізація зміни коефіцієнтів кореляції Спірмена ρ та Пірсона r свідчить про поступове зростання узгодженості між прогнозованими та анотованими значеннями семантичної подібності. Архітектура GATv2Conv демонструє

значно вищі та стабільніші результати кореляції порівняно з R-GCN протягом усього процесу навчання. Крім того, використання механізму уваги на етапі агрегування ознак графа перевершує операцію глобального усереднення для обох архітектур. Хоча моделі R-GCN швидше досягають оптимального значення похибки, їхня узагальнювальна здатність залишається порівняно низькою. Натомість архітектури GATv2 потребують тривалішого навчання, проте забезпечують вищу якість векторних репрезентацій.

Подальше навчання не призводить до покращення метрик, що є підставою для застосування механізму ранньої зупинки та фіксації оптимального стану моделі. Фінальні результати навчання для різних комбінацій архітектур наведено в таблиці 4.12.

Таблиця 4.12.

Порівняльні результати тестування нейромережевих моделей генерації вкладень AMR графів

Архітектура	Тип агрегації	Spearman (ρ)	Pearson (r)
RGCN	Global mean pooling	0.67	0.66
	Attention pooling	0.69	0.70
GATv2Conv	Global mean pooling	0.72	0.73
	Attention pooling	0.75	0.74

Отримані результати експериментів показали, що серед протестованих конфігурацій найвищу ефективність продемонструвала архітектура GATv2Conv у поєднанні з агрегуванням з використанням механізму уваги. Це свідчить, що використання механізму уваги є важливим не тільки на етапі обчислення, але й на етапі агрегації у фіксований вектор. У порівнянні з простим агрегуванням за середнім, агрегування з використанням механізму уваги забезпечує вищу

узгодженість прогнозованих значень з анотованими, що підтверджується вищими коефіцієнтами кореляції Спірмена та Пірсона.

Отриману модель було порівняно з іншими підходами, що використовують AMR для задачі оцінки семантичної подібності речень. Зокрема, до порівняння включено AMRSim, яка демонструє підхід без використання вирівнювання до оцінки схожості графів [96]. Модель створює вкладення для AMR графа за допомогою трансформера з GNN-адаптерами, що дозволяє враховувати як лінійний порядок концептів, так і їхню структурну взаємодію. Крім того, розглядалися підходи, які інтегрують AMR у трансформерні моделі з метою покращення точності вкладання речень, зокрема моделі Xpara та LabSE [97]. У таких методах структурна інформація графа використовується як додатковий контекст для трансформера, що дозволяє моделі захоплювати семантичні взаємозв’язки між концептами речення, які важко виявити з однієї лише лінійної послідовності токенів. Також проведено порівняння з класичною метрикою Smatch, яка оцінює максимальний структурний збіг AMR графів [98]. Результати порівняння наведені у таблиці 4.13.

Таблиця 4.13.

Порівняльна характеристика ефективності розробленої моделі з наявними підходами у задачі STS

Модель	Архітектура	Spearman	Pearson	Кількість параметрів, млн.
Запропонована модель	GAT	0.75	0.74	16,3
AMRsim	GNN	-	0.70	100
Smatch	Алгоритм Smatch	0.53	0.55	0
Xpara	RoBERTa	0.84	-	120
LabSE	BERT	0.76	-	110

Порівняльний аналіз результатів показує, що запропонована модель на основі графових нейронних мереж демонструє покращення якості побудови векторних репрезентацій на 4% у порівнянні з іншими підходами, що використовують AMR графи, а також зіставні показники якості за Спірменом та Пірсоном за значно меншої кількості параметрів ніж трансформерні моделі типу Xpara та LaBSE. Трансформерні підходи забезпечують високі значення кореляції, однак досягають цього ціною вищих обчислювальних витрат, що обмежує їх застосування в ресурсно обмежених середовищах. Водночас класичні методи, зокрема Smatch, демонструють значно нижчу кореляцію через відсутність векторного семантичного узагальнення.

Таким чином, отримані результати підтверджують, що розроблена модель забезпечує оптимальний компроміс між точністю, швидкістю та ресурсною ефективністю, що робить її доцільним для практичних інтелектуальних систем ідентифікації перифраз.

4.5. Експериментальне дослідження класифікації перифраз на основі векторних репрезентацій AMR графів

Експериментальне дослідження модуля класифікації перифраз є фінальним етапом дослідження розробленої інформаційної технології ідентифікації перифраз на основі графових репрезентацій AMR. На попередніх стадіях експерименту було послідовно реалізовано процеси багатомовної підготовки текстів, генерації абстрактних семантичних графів нейронною моделлю, а також їх векторного кодування з використанням графових нейронних мереж. Отримані на цих етапах векторні репрезентації було використано як вхідні ознаки для навчання класифікаційного модуля. Така експериментальна схема забезпечила комплексну перевірку здатності всієї системи здійснювати автоматичне виявлення перифраз на основі порівняння їхніх абстрактних семантичних структур, а не поверхневих лексичних характеристик.

Класифікаційний модуль було реалізовано у вигляді багатошарового перцептрона, що забезпечує нелінійне перетворення простору ознак та формування узагальненого рішення щодо семантичної еквівалентності пари висловлень. Вибір архітектури MLP зумовлений її здатністю ефективно моделювати складні нелінійні залежності між компонентами векторних представлень, отриманих на основі AMR графів, а також відносною простотою налаштування та обчислювальною ефективністю. Вхідний шар класифікатора приймає інтегровані ознаки взаємодії двох векторів, після чого вони послідовно обробляються кількома прихованими шарами з нелінійними функціями активації та механізмами регуляризації, що забезпечує стійкість навчання і зниження ризику перенавчання. Вихідний шар формує скалярний логіт, який після сигмоїдного перетворення інтерпретується як імовірність наявності перифрази між двома висловленнями.

Експериментальне навчання та оцінювання класифікатора проводилося на корпусі Microsoft Research Paraphrase Corpus (MRPC), який є стандартним еталонним набором даних для задачі ідентифікації перифраз [20]. Корпус містить пари англомовних речень, анотованих за наявністю або відсутністю перифрази, що забезпечує можливість формалізованої перевірки якості класифікації. Характеристики набору даних наведені у таблиці 4.14. У ході експерименту використовувався поділ на навчальну та тестову вибірки, схема якого наведена у таблиці 4.15.

Таблиця 4.14.

Загальні характеристики набору даних MRPC

Характеристика	Значення
Мова корпусу	Англійська
Тип даних	Пари речень з бінарними мітками
Кількість пар речень	5801
Тематика даних	Новини, інформаційні повідомлення

Таблиця 4.15

Структура набору даних MRPC за типами вибірок та обсягом даних

Вибірка	Кількість	Частка
Навчальна	3668	63,3%
Валідаційна	408	7,0%
Тестова	1725	29,7%

З огляду на структурні характеристики обраного корпусу, у якому спостерігається статистичний дисбаланс у бік позитивних пар речень, для навчання класифікаційного модуля було застосовано функцію втрат Focal Loss [99]. Такий вибір зумовлений її здатністю компенсувати нерівномірність розподілу класів, знижувати вплив «легких» прикладів та підсилювати роль «важких» або неоднозначних випадків, що забезпечує стабільніше навчання та підвищує точність розпізнавання перифраз. Функція втрат визначається як:

$$\mathcal{L}_{focal}(p_t) = -\alpha(1 - p_t)^{\gamma} \log p_t, \tag{4.9}$$

де p_t – прогнозована ймовірність моделі, α – збалансований ваговий коефіцієнт, що компенсує чисельний дисбаланс між класами у навчальній вибірці, γ – параметр фокусування, що контролює вплив легких і важких прикладів на оновлення ваг. У проведених експериментах параметри Focal Loss було встановлено як α=0.5, γ=1.5, що забезпечує збалансоване зменшення ваги найбільш представлених прикладів та підвищує чутливість моделі до рідкісних або складних для класифікації спостережень.

Оцінювання якості роботи класифікаційного модуля здійснювалося на основі метрик бінарної класифікації, розрахованих за допомогою елементів матриці помилок. Базовими показниками оцінки є точність та повнота, що визначаються за формулами:

$$Precision = \frac{TP}{TP+FP}, \quad (4.10)$$

$$Recall = \frac{TP}{TP+FN}, \quad (4.11)$$

де TP – кількість істинно позитивних рішень, FP – кількість хибнопозитивних рішень, FN – кількість хибнонегативних рішень.

Для отримання збалансованої кількісної характеристики якості класифікації, особливо в умовах нерівномірного розподілу класів, використовується міра F1 [100]. Вона інтегрує властивості точності та повноти у вигляді їхнього гармонійного середнього і формально визначається як:

$$F_1 = 2 \cdot \frac{Presicion \cdot Recall}{Precision+Recall}, \quad (4.12)$$

Оскільки вихід класифікатора представлений у вигляді логітів, тобто ненормованих значень, що відображають міру впевненості моделі щодо належності прикладу до позитивного класу, для отримання бінарного рішення необхідно визначити поріг класифікації τ, за яким логіти перетворюються на ймовірнісні оцінки та подальші бінарні мітки

У стандартних задачах двокласової класифікації поріг τ=0.5 вважається типовим. Проте для задачі ідентифікації перифраз, де класи можуть бути суттєво нерівномірно представлені, а вимоги до балансу між точністю та повнотою є критичними, використання фіксованого порогу може призводити до не оптимальних рішень. Пошук оптимального порогу τ здійснювався за критерієм максимізації міри F1, що дозволяє врахувати одночасно як точність передбачення позитивних випадків, так і повноту їхнього охоплення. Такий підхід є обґрунтованим у задачах ідентифікації перифраз, оскільки він мінімізує ризик систематичних помилок одного типу та забезпечує збалансовану семантичну чутливість моделі.

З метою кількісної оцінки здатності моделі коректно розрізняти перифрази та не перифрази було також використано показник AUC-ROC, який характеризує

відповідність ранжування ймовірнісних оцінок, що генеруються класифікатором, та забезпечує інваріантність до вибору порога прийняття рішення [95].

ROC-крива відображає функціональну залежність між часткою істинно позитивних випадків та часткою хибнопозитивних випадків при зміні порога класифікації $\tau \in [0, 1]$. Зазначені показники обчислюються на основі елементів матриці помилок:

$$TPR(\tau) = \frac{TP(\tau)}{TP(\tau)+FN(\tau)}, \quad (4.13)$$

$$FPR(\tau) = \frac{FP(\tau)}{FP(\tau)+TN(\tau)}, \quad (4.14)$$

де TP – кількість істинно позитивних випадків; FN – кількість хибнонегативних випадків; FP – кількість хибнопозитивних випадків; TN – кількість істинно негативних випадків.

Площа під ROC-кривою AUC інтерпретується як інтегральний показник здатності класифікатора відрізняти позитивні приклади від негативних на всьому діапазоні порогів прийняття рішень і визначається через інтеграл:

$$AUC = \int_0^1 TPR d(FPR), \quad (4.15)$$

де TPR – значення частки істинно позитивних випадків, FPR – значення частки хибнопозитивних випадків.

Значення AUC відповідає ймовірності того, що випадково обрана перифраза отримає від класифікатора вищу оцінку близькості, ніж випадково обрана не перифраза.

Результати класифікації на тестовій вибірці наведено у таблиці 4.16. Через наявний дисбаланс класів з переважанням позитивних пар метрика точності не повною мірою відображає реальну якість роботи модуля. Натомість міра F1, що

інтегрує показники точності та повноти для позитивного класу, демонструє істотно вищий рівень, що свідчить про здатність класифікатора одночасно підтримувати високий рівень виявлення перифраз та обмежувати кількість хибнопозитивних рішень навіть в умовах нерівномірного розподілу класів.

Таблиця 4.16.

Результати ідентифікації перифраз для набору даних MRPC на основі класифікаційних метрик

Показник	Значення
Accuracy	0.795
Precision	0.832
Recall	0.902
F1	0.851
AUC-ROC	0.775

Отримані результати класифікації перифраз на обраному наборі даних підтверджують здатність розробленого модуля класифікації відокремлювати пари перифраз на основі абстрактних семантичних репрезентацій AMR. Показник AUC-ROC свідчить про достатню роздільну здатність моделі та коректне впорядкування пар за ступенем семантичної подібності.

Для додаткової перевірки класифікації перифраз з використанням української мови було використано матеріали словника української мови у 20 томах (СУМ-20) [101]. СУМ-20 охоплює загальновживану, наукову, публіцистичну та абстрактну лексики й містить приклади вживання у формі повних речень, насичених предикатно-аргументними та синтаксично ускладненими конструкціями. Це забезпечує його репрезентативність для задач семантичного аналізу та дозволяє використовувати отриману вибірку як незалежне джерело для перевірки коректності AMR інтерпретації текстів після нейронного перекладу з української мови на англійську.

З огляду на відсутність оприлюднених спеціалізованих корпусів перифраз українською мовою було сформовано експериментальний набір даних. Для кожного речення було застосовано генеративну мовну модель GPT-5.1 [102] із завданням сформувати семантично еквівалентне висловлення, що зберігає зміст оригіналу іншими лексичними й синтаксичними засобами. Таким чином було отримано позитивні приклади перифраз. Негативні приклади формувалися шляхом випадкового поєднання речень, що не мають семантичної спорідненості, взятих із тієї самої вибірки. Структуру набору даних було збалансовано за однаковим співвідношенням позитивних та негативних прикладів. Приклад запиту до моделі, а також результати генерації наведено у таблиці 4.17.

Таблиця 4.17

Зразки синтезованих українськомовних перифраз та відповідних запитів для їх генерації

Вхідне речення	Запит до моделі	Вихідне речення
П’ять-шість, а подекуди й більше сліпих лисенят приводить руда в березні-квітні	Перефразуй наведене речення українською мовою, зберігши його зміст, але змінивши формулювання. Уникай простого переставляння слів, намагайся використати інші мовні конструкції.	У березні-квітні руда лисиця народжує зазвичай п’ять-шість, а іноді й ще більше незрячих лисенят.

Для проведення незалежної валідації класифікаційного модуля було сформовано експериментальну вибірку обсягом 1200 пар речень. Зазначена вибірка використовувалася виключно для перевірки здатності класифікатора

узагальнювати знання на українськомовних даних та оцінки його стійкості поза межами стандартних англомовних корпусів.

Сформований набір українськомовних перифраз було подано на вхід розробленому обчислювальному конвеєру, що включає послідовні етапи перекладу на англійську мову, генерації AMR графів, отримання векторних представлень та фінальної класифікації. Такий підхід забезпечив відтворення повного циклу обробки для українських текстів і дозволив оцінити ефективність запропонованої інформаційної технології. Результати класифікації наведено у таблиці 4.18.

Таблиця 4.18.

Результати ідентифікації перифраз для набору даних українськомовних перифраз на основі класифікаційних метрик

Показник	Значення
Accuracy	0.80
Precision	0.83
Recall	0.85
F1	0.84
AUC-ROC	0.77

Отримані показники узгоджені з результатами, отриманими на корпусі MRPC, що свідчить про збереження здатності класифікатора коректно відокремлювати перифрази. Попри наявність проміжного етапу перекладу та AMR інтерпретації, рівень міри F1 та площі під ROC-кривою демонструють достатній рівень роздільної здатності та підтверджують ефективність запропонованої інформаційної технології в умовах роботи з текстами української мови.

Результати класифікатора було зіставлено як з AMR орієнтованими підходами, так і з сучасними трансформерними моделями, що розв’язують

задачу ідентифікації перифраз на корпусі MSRP. Зокрема, до порівняння включено метод, у якому AMR структури використовуються у поєднанні зі статистичними ознаками та лінійними класифікаторами [103]. Також наведено результати трансформерних моделей SBERT/SRoBERTa [104], T5 [105] та гібридних архітектур з використанням рекурентних нейронних мереж [106, 107]. Порівняльні результати наведено у таблиці 4.19.

Таблиця 4.19.

Порівняльна характеристика ефективності розробленої моделі та сучасних нейромережевих архітектур на наборі даних MRPC

Модель	Accuracy	F1
Запропонована модель	0.795	0.851
AMR for Paraphrase Detection	0.687	0.809
SBERT/SRoBERTa	0.829	-
T5	0.82	0.799
LSTM	0.79	0.854
Multi-cascade + augmentation	0.783	0.848

Наведені у таблиці результати свідчать, що розроблений класифікатор забезпечує вищу точність та міру F1, ніж AMR орієнтовані підходи. У порівнянні із трансформерними моделями класифікатор досягає зіставних оцінок, попри те, що трансформерні моделі навчаються на значно більших обсягах даних і мають суттєво більшу кількість параметрів.

Інтерпретація отриманих результатів експериментів дозволила визначити подальші напрями дослідження, а саме необхідність формування спеціалізованого корпусу україномовних перифраз із ручною семантичною анотацією та забезпечення прямого створення AMR графів українською мовою, що дозволить усунути проміжні етапи перекладу та генерації AMR графів, які

зумовлюють можливість накопичення похибок і, відповідно, впливають на ефективність класифікації.

Таким чином, отримані результати підтверджують ефективність запропонованого підходу до класифікації перифраз на основі векторних представлень AMR графів, демонструють його здатність забезпечувати якісну ідентифікацію перифраз, а також засвідчують перспективність подальшого розвитку запропонованої інформаційної технології у напрямі розширення мовної підтримки та підвищення точності семантичного аналізу.

4.6. Висновки за розділом 4

У межах розділу було здійснено експериментальне дослідження інформаційної технології ідентифікації перифраз на основі абстрактних семантичних репрезентацій. Отримані результати мають теоретичне та прикладне значення та дозволяють сформулювати такі висновки.

1. Доведено ефективність використання трансформерних моделей нейронного машинного перекладу як етапу підготовки текстових даних до AMR-парсингу. Експериментально встановлено, що застосування нейронних моделей перекладу при роботі з українськомовними текстами забезпечує збереження семантичної структури на рівні 94%, що свідчить про відсутність суттєвих втрат концептуальної інформації, необхідної для побудови абстрактно-смислових репрезентацій.

2. Експериментально підтверджено високу якість побудови абстрактних семантичних репрезентацій у форматі AMR із застосуванням нейронної моделі AMRBART. Отримані результати засвідчили, що згенеровані моделлю семантичні графи демонструють на 2,6% вищу узгодженість за відповідними метриками, ніж результати базових підходів, що підтверджує ефективність інтеграції AMRBART у розроблений обчислювальний конвеєр для задач глибинної семантичної інтерпретації текстів.

3. Верифіковано ефективність застосування механізму уваги у графових нейронних мережах при побудові векторних представлень AMR-графів. Експериментальне моделювання довело перевагу запропонованої архітектури GATv2Conv на 5% порівняно з графовими моделями без використання механізмів уваги. Порівняння із наявними AMR орієнтованими підходами показало приріст точності на 4% у задачі оцінювання семантичної подібності, що підтверджує здатність моделі точніше відображати абстрактну структуру змісту.

4. Проведено експериментальне дослідження модуля класифікації перифраз на англомовному корпусі MRPC, що дозволило верифікувати базову здатність моделі ідентифікувати семантичну еквівалентність на основі AMR-графів. Експериментально отримано значення точності та F1 міри результатів роботи класифікатора, що склали 79,1% та 85,1% відповідно. Для експериментальної перевірки роботи класифікатора на українських даних було сформовано набір даних на основі словника СУМ, шляхом автоматичної генерації перифраз. За результатами тестування на розширеному україномовному наборі даних експериментально отримано значення точності та F1 міри, що склали 80% та 84% відповідно. Отримані результати за показниками точності перевищують показники наявних підходів, що використовують AMR графи для ідентифікації перифраз на 11%., а за показниками F1 міри на 4,2%.

ВИСНОВКИ

В дисертаційній роботі вирішена актуальна науково-прикладна задача підвищення точності ідентифікації перифраз у текстах українською мовою в контексті моделювання когнітивного процесу розуміння шляхом розробки інтелектуальної системи, яка використовує семантичні графові репрезентації для виявлення семантичної еквівалентності між висловлюваннями на концептуальному рівні.

У ході досліджень отримані наступні основні та наукові теоретичні та практичні результати:

1. Проведено аналіз проблеми моделювання когнітивних процесів розуміння природномовних висловлень та встановлення семантичної еквівалентності між ними. Показано, що наявні підходи переважно орієнтовані на статистичні або евристичні методи, які не здатні повною мірою відобразити глибинні семантичні зв’язки між висловленнями. Відзначено актуальність досліджень, результатом яких має стати побудова нових та вдосконалення наявних методів і моделей автоматизованої ідентифікації перифраз на основі абстрактних семантичних графових репрезентацій з метою підвищення точності розпізнавання семантичної еквівалентності україномовних текстів.

2. Запропоновано когнітивно орієнтовану методику ідентифікації перифраз у текстах українською мовою на основі методів машинного навчання. Її застосування дозволяє здійснювати автоматизований аналіз семантичної еквівалентності висловлювань на концептуальному рівні, розглядаючи складний процес розуміння як формалізований алгоритм зіставлення смислових структур з використанням семантичних графових репрезентацій AMR.

3. Вперше запропоновано когнітивно обґрунтовану формалізовану модель перифраза, в якій він розглядається як концептуально-еквівалентний смисловий варіант у межах семантичного простору абстрактних репрезентацій змісту. На відміну від існуючих підходів, орієнтованих на лексичний або синтаксичний перетин векторів, це створило передумови для уніфікації підходу

до виявлення подібності між висловлюваннями незалежно від особливостей їхньої поверхневої лексичної та морфологічної будови.

4. Вперше запропоновано інформаційну технологію формування AMR-графів для українськомовних текстів на основі їхнього попереднього нейронного машинного перекладу. Експериментально доведено, що розроблена технологія гарантує збереження семантичної структури текстів на рівні 94%, а програмна інтеграція моделі AMRBART забезпечує генерацію абстрактних графів на основі отриманого перекладу із рівнем структурної узгодженості, який на 2,6 % перевищує результати наявних підходів.

5. Удосконалено метод репрезентації текстів природної мови за допомогою AMR шляхом використання графових нейронних мереж з механізмом уваги. Вони дозволяють формувати векторні представлення, що відображають як локальні, так і глобальні властивості смислової структури висловлювання. Експериментально підтверджено, що даний метод забезпечує підвищення точності у задачі оцінювання семантичної подібності текстів на 4% порівняно з наявними AMR орієнтованими підходами.

6. Удосконалено метод ідентифікації перифраз шляхом класифікації на основі векторних представлень AMR графів шляхом застосування багатошарового перцептрона із вентильним механізмом керованої селекції ознак. Він забезпечує прийняття рішень про семантичну еквівалентність висловлювань на рівні концептуально-смислових структур. Запропонований метод дозволяє досягати показників точності та F1-міри на 11% та на 4,2% відповідно вищими за наявні підходи, що використовують AMR графи для ідентифікації перифраз.

7. Реалізовано алгоритмічне та програмне забезпечення інформаційної технології у вигляді інтерактивного клієнт-серверного вебзастосунку. Для експериментальної перевірки ефективності роботи створеної системи з українськомовними даними було сформовано цільовий набір текстів на основі словника СУМ-20 шляхом автоматизованої генерації перифраз. За результатами тестування експериментально отримано значення точності та F1-міри, що склали

80% та 84% відповідно. Таким чином, підтверджено наукову гіпотезу дослідження щодо можливості ефективної автоматизованої ідентифікації перифраз українськомовних висловлень шляхом побудови, векторизації та машинного аналізу абстрактних семантичних репрезентацій.

Отримані результати підтверджують перспективність застосування розробленої інформаційної технології для автоматизованої ідентифікації перифраз у текстах українською мовою. Практична реалізація цієї технології може бути використана у системах штучного інтелекту для автоматичного розпізнавання перифраз, інформаційного пошуку, реферування текстів, навчальних і наукових застосунках, а також у розвитку когнітивно орієнтованих лінгвістичних систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Bloom B. S. Taxonomy of educational objectives: The classification of educational goals. [S.l.] : Longman, 1956.

2. A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives / ed. by A. L. W, K. D. R, B. B. S. 1913-. New York : Longman, 2001. 301 p.

3. A layered reference model of the brain (LRMB) / Yingxu Wang et al. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*. 2006. Vol. 36, no. 2. P. 124–133. URL: https://doi.org/10.1109/tsmcc.2006.871126.

4. Bechtel W., Abrahamsen A., Graham G. The Life of Cognitive Science. *A Companion to Cognitive Science*. Oxford, UK, 2017. P. 1–104. URL: https://doi.org/10.1002/9781405164535.part1.

5. Modeling human and organizational behavior: Application to military simulations / ed. by P. R. W, M. A. S, National Research Council (U.S.). Panel on Modeling Human Behavior and Command Decision Making: Representations for Military Simulations. Washington, D.C : National Academy Press, 1998. 418 p.

6. Vicente K. J., Wang J. H. An ecological theory of expertise effects in memory recall. *Psychological Review*. 1998. Vol. 105, no. 1. P. 33–57. URL: https://doi.org/10.1037/0033-295x.105.1.33.

7. Sun R., Ling C. X. Computational cognitive modeling, the source of power, and other related issues. *AI Magazine*. 1998. Vol. 19, no. 2. P. 113–120. URL: https://doi.org/10.1609/aimag.v19i2.1374.

8. Bell C. G. Computer structures: Readings and examples. New York : McGraw-Hill, 1971. 668 p.

9. Kolers P. A., Smythe W. E. Symbol manipulation: Alternatives to the computational view of mind. *Journal of Verbal Learning and Verbal Behavior*. 1984. Vol. 23, no. 3. P. 289–314. URL: https://doi.org/10.1016/s0022-5371(84)90182-8.

10. A world survey of artificial brain projects, Part II: Biologically inspired cognitive architectures / B. Goertzel et al. *Neurocomputing*. 2010. Vol. 74, no. 1-3. P. 30–49. URL: https://doi.org/10.1016/j.neucom.2010.08.012.

11. Stenson N., Hornby A. S. Oxford Advanced Learner's Dictionary of Current English. *The Modern Language Journal*. 1996. Vol. 80, no. 3. P. 412. URL: https://doi.org/10.2307/329464.

12. Dagan I., Glickman O., Magnini B. The PASCAL Recognising Textual Entailment Challenge. *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*. Berlin, Heidelberg, 2006. P. 177–190. URL: https://doi.org/10.1007/11736790_9.

13. Dras M. Tree adjoining grammar and the reluctant paraphrasing of text. Sydney, 1999.

14. Harris Z. S. Distributional Structure. *Papers in Structural and Transformational Linguistics*. Dordrecht, 1970. P. 775–794. URL: https://doi.org/10.1007/978-94-017-6059-1_36.

15. Miller G. A. WordNet. *Communications of the ACM*. 1995. Vol. 38, no. 11. P. 39–41. URL: https://doi.org/10.1145/219717.219748.

16. Patwardhan S., Banerjee S., Pedersen T. Using Measures of Semantic Relatedness for Word Sense Disambiguation. *Computational Linguistics and Intelligent Text Processing*. Berlin, Heidelberg, 2003. P. 241–257. URL: https://doi.org/10.1007/3-540-36456-0_24.

17. Corley C., Mihalcea R. Measuring the semantic similarity of texts. *the ACL Workshop*, Ann Arbor, Michigan, 30 June 2005. Morristown, NJ, USA, 2005. URL: https://doi.org/10.3115/1631862.1631865.

18. Mihalcea R., Corley C., Strapparava C. Corpus-based and knowledge-based measures of text semantic similarity. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2006. P. 775–780. URL: https://dl.acm.org/doi/10.5555/1597538.1597662.

19. Landauer T. K., Foltz P. W., Laham D. An introduction to latent semantic analysis. *Discourse Processes*. 1998. Vol. 25, no. 2-3. P. 259–284. URL: https://doi.org/10.1080/01638539809545028.

20. Brockett C., Dolan W. B. Support Vector Machines for Paraphrase Identification and Corpus Construction. *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*. 2005. URL: https://aclanthology.org/I05-5001/.

21. Chang C.-C., Lin C.-J. LIBSVM. *ACM Transactions on Intelligent Systems and Technology*. 2011. Vol. 2, no. 3. P. 1–27. URL: https://doi.org/10.1145/1961189.1961199.

22. Almeida F., Xexéo G. Word Embeddings: A Survey. *arXiv preprint*. 2023. URL: https://arxiv.org/abs/1901.09069.

23. Baroni M., Dinu G., Kruszewski G. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland. Stroudsburg, PA, USA, 2014. URL: https://doi.org/10.3115/v1/p14-1023.

24. A neural probabilistic language model / Y. Bengio et al. *J. Mach. Learn. Res.* 2003. Vol. 3. P. 1137–1155. URL: https://dl.acm.org/doi/10.5555/944919.944966.

25. Efficient estimation of word representations in vector space / T. Mikolov et al. *arXiv preprint*. 2013. URL: https://arxiv.org/abs/1301.3781.

26. Pennington J., Socher R., Manning C. Glove: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar. Stroudsburg, PA, USA, 2014. URL: https://doi.org/10.3115/v1/d14-1162.

27. Baseline Needs More Love: On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms / D. Shen et al. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*,

Melbourne, Australia. Stroudsburg, PA, USA, 2018. URL: https://doi.org/10.18653/v1/p18-1041.

28. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection / R. Socher et al. *Proceedings of the 25th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA, 2011. P. 801–809. URL: https://dl.acm.org/doi/10.5555/2986459.2986549.

29. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation / K. Cho et al. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar. Stroudsburg, PA, USA, 2014. URL: https://doi.org/10.3115/v1/d14-1179.

30. Skip-Thought Vectors / R. Kiros et al. *arXiv preprint*. 2015. URL: https://arxiv.org/abs/1506.06726.

31. Convolutional Neural Network Architectures for Matching Natural Language Sentences / B. Hu et al. *Neural Information Processing Systems*. 2014.

32. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs / W. Yin et al. *Transactions of the Association for Computational Linguistics*. 2016. Vol. 4. P. 259–272. URL: https://doi.org/10.1162/tacl_a_00097.

33. Nastase V., Mihalcea R., Radev D. R. A survey of graphs in natural language processing. *Natural Language Engineering*. 2015. Vol. 21, no. 5. P. 665–698. URL: https://doi.org/10.1017/s1351324915000340.

34. Ilievski F., Szekely P., Schwabe D. Commonsense Knowledge in Wikidata. *arXiv preprint*. 2020. URL: https://arxiv.org/abs/2008.08114.

35. Sowa J. F. Knowledge Representation: Logical, Philosophical, and Computational Foundations: Logical, Philosophical, and Computational Foundations. Course Technology, 1999. 608 p.

36. Taylor A., Marcus M., Santorini B. The Penn Treebank: An Overview. *Treebanks*. Dordrecht, 2003. P. 5–22. URL: https://doi.org/10.1007/978-94-010-0201-1_1.

37. Bunescu R. C., Mooney R. J. A shortest path dependency kernel for relation extraction. *the conference,* Vancouver, British Columbia, Canada, 6–8 October 2005. Morristown, NJ, USA, 2005. URL: https://doi.org/10.3115/1220575.1220666.

38. Abstract Meaning Representation for Sembanking / L. Banarescu et al. *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*. Sofia, Bulgaria, 2013. P. 178–186.

39. Wang Y. Cognitive Informatics: A New Transdisciplinary Research Field. *Brain and Mind*. 2003. Vol. 4, no. 2. P. 115–127. URL: https://doi.org/10.1023/a:1025419826662.

40. Wang Y. On Concept Algebra. *International Journal of Cognitive Informatics and Natural Intelligence*. 2008. Vol. 2, no. 2. P. 1–19. URL: https://doi.org/10.4018/jcini.2008040101.

41. Formal Concept Analysis / ed. by B. Ganter, G. Stumme, R. Wille. Berlin, Heidelberg : Springer Berlin Heidelberg, 2005. URL: https://doi.org/10.1007/978-3-540-31881-1.

42. Bhagat R., Hovy E. What Is a Paraphrase?. *Computational Linguistics*. 2013. Vol. 39, no. 3. P. 463–472. URL: https://doi.org/10.1162/coli_a_00166.

43. Androutsopoulos I., Malakasiotis P. A Survey of Paraphrasing and Textual Entailment Methods. *Journal of Artificial Intelligence Research*. 2010. Vol. 38. P. 135–187. URL: https://doi.org/10.1613/jair.2985.

44. Mansouri B. Survey of Abstract Meaning Representation: Then, Now, Future. *arXiv preprint*. 2025. URL: https://arxiv.org/abs/2505.03229.

45. Palmer M., Gildea D., Kingsbury P. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*. 2005. Vol. 31, no. 1. P. 71–106. URL: https://doi.org/10.1162/0891201053630264.

46. A systematic review on sequence-to-sequence learning with neural network and its models / H. Yousuf et al. *International Journal of Electrical and Computer Engineering (IJECE)*. 2021. Vol. 11, no. 3. P. 2315. URL: https://doi.org/10.11591/ijece.v11i3.pp2315-2326.

47. Addressing the Data Sparsity Issue in Neural AMR Parsing / X. Peng et al. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Valencia, Spain. Stroudsburg, PA, USA, 2017. URL: https://doi.org/10.18653/v1/e17-1035.

48. Attention Is All You Need / A. Vaswani et al. *arXiv preprint*. 2023. URL: https://arxiv.org/abs/1706.03762.

49. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension / M. Lewis et al. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Stroudsburg, PA, USA, 2020. URL: https://doi.org/10.18653/v1/2020.acl-main.703.

50. Bevilacqua M., Blloshmi R., Navigli R. One SPRING to Rule Them Both: Symmetric AMR Semantic Parsing and Generation without a Complex Pipeline. *Proceedings of the AAAI Conference on Artificial Intelligence*. 2021. Vol. 35, no. 14. P. 12564–12573. URL: https://doi.org/10.1609/aaai.v35i14.17489.

51. Bai X., Chen Y., Zhang Y. Graph Pre-training for AMR Parsing and Generation. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland. Stroudsburg, PA, USA, 2022. URL: https://doi.org/10.18653/v1/2022.acl-long.415.

52. Lyu C., Titov I. AMR Parsing as Graph Prediction with Latent Alignment. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Stroudsburg, PA, USA, 2018. URL: https://doi.org/10.18653/v1/p18-1037.

53. AMR Parsing as Sequence-to-Graph Transduction / S. Zhang et al. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/p19-1009.

54. Simple and Effective Curriculum Pointer-Generator Networks for Reading Comprehension over Long Narratives / Y. Tay et al. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/p19-1486.

55. Wein S., Schneider N. Classifying Divergences in Cross-lingual AMR Pairs. *Proceedings of The Joint 15th Linguistic Annotation Workshop (LAW) and 3rd Designing Meaning Representations (DMR) Workshop*, Punta Cana, Dominican Republic. Stroudsburg, PA, USA, 2021. URL: https://doi.org/10.18653/v1/2021.law-1.6.

56. Blloshmi R., Tripodi R., Navigli R. XL-AMR: Enabling Cross-Lingual AMR Parsing with Transfer Learning Techniques. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Stroudsburg, PA, USA, 2020. URL: https://doi.org/10.18653/v1/2020.emnlp-main.195.

57. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding / J. Devlin et al. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota, 2019. P. 4171–4186. URL: https://doi.org/10.18653/v1/N19-1423.

58. Unsupervised Cross-lingual Representation Learning at Scale / A. Conneau et al. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Stroudsburg, PA, USA, 2020. URL: https://doi.org/10.18653/v1/2020.acl-main.747.

59. Translate, then Parse! A Strong Baseline for Cross-Lingual AMR Parsing / S. Uhrig et al. *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, Online. Stroudsburg, PA, USA, 2021. URL: https://doi.org/10.18653/v1/2021.iwpt-1.6.

60. Grover A., Leskovec J. node2vec. *KDD '16: The 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA. New York, NY, USA, 2016. URL: https://doi.org/10.1145/2939672.2939754.

61. graph2vec: Learning Distributed Representations of Graphs / A. Narayanan et al. 2017. ArXiv:1707.05005. URL: https://doi.org/10.48550/arXiv.1707.05005.

62. Efficient Estimation of Word Representations in Vector Space / T. Mikolov et al. *Proceedings of the International Conference on Learning Representations*. 2013. URL: https://api.semanticscholar.org/CorpusID:5959482.

63. O'Shea K., Nash R. An Introduction to Convolutional Neural Networks. *arXiv preprint*. 2015. URL: https://arxiv.org/abs/1511.08458.

64. Fix E., Hodges J. L. Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. *International Statistical Review / Revue Internationale de Statistique*. 1989. Vol. 57, no. 3. P. 238. URL: https://doi.org/10.2307/1403797.

65. Multilayer perceptron and neural networks / M.-C. Popescu et al. *WSEAS Transactions on Circuits and Systems*. 2009. Vol. 8.

66. Damonte M., Cohen S. B. Cross-Lingual Abstract Meaning Representation Parsing. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana. Stroudsburg, PA, USA, 2018. URL: https://doi.org/10.18653/v1/n18-1104.

67. Tiedemann J., Thottingal S. OPUS-MT -- Building open translation services for the World. *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation* / ed. by A. Martins et al. Lisboa, Portugal, 2020. P. 479–480. URL: https://aclanthology.org/2020.eamt-1.61/.

68. Beyond English-Centric Multilingual Machine Translation / A. Fan et al. *Journal of Machine Learning Research*. 2021. Vol. 22, no. 1. P. 48.

69. No Language Left Behind: Scaling Human-Centered Machine Translation / N. Team et al. *arXiv preprint*. 2022. URL: https://arxiv.org/abs/2207.04672.

70. Knight K., al. e. Abstract Meaning Representation (AMR) Annotation Release 2.0 LDC2017T10. URL: https://doi.org/10.35111/s444-np87.

71. Knight K., others. Abstract Meaning Representation (AMR) Annotation Release 3.0. URL: https://doi.org/10.35111/44cy-bp51.

72. Goodman M. W. Penman: An Open-Source Library and Tool for AMR Graphs. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Online. Stroudsburg, PA, USA, 2020. URL: https://doi.org/10.18653/v1/2020.acl-demos.35.

73. Signature Verification Using a “Siamese” Time Delay Neural Network / J. Bromley et al. *International Journal of Pattern Recognition and Artificial Intelligence*. 1993. Vol. 07, no. 04. P. 669–688. URL: https://doi.org/10.1142/s0218001493000339.

74. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/d19-1410.

75. Brody S., Alon U., Yahav E. How Attentive are Graph Attention Networks?. *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=F72ximsx7C1.

76. A Comprehensive Survey on Graph Neural Networks / Z. Wu et al. *IEEE Transactions on Neural Networks and Learning Systems*. 2021. Vol. 32, no. 1. P. 4–24. URL: https://doi.org/10.1109/tnnls.2020.2978386.

77. Gated Graph Sequence Neural Networks / Y. Li et al. *arXiv preprint*. 2017. URL: https://arxiv.org/abs/1511.05493.

78. Singhal A. Modern Information Retrieval: A Brief Overview. *IEEE Data Eng. Bull.* 2001. Vol. 24. P. 35–43.

79. Language modeling with gated convolutional networks / Y. N. Dauphin et al. *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. Sydney, NSW, Australia, 2017. P. 933–941. URL: https://doi.org/10.1016/j.compbiomed.2022.106462.

80. PyTorch: An Imperative Style, High-Performance Deep Learning Library / A. Paszke et al. *arXiv preprint*. 2019. URL: https://arxiv.org/abs/1912.01703.

81. Fey M., Lenssen J. E. Fast Graph Representation Learning with PyTorch Geometric. *arXiv preprint*. 2019. URL: https://arxiv.org/abs/1903.02428.

82. Transformers: State-of-the-Art Natural Language Processing / T. Wolf et al. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online, 2020. P. 38–45. URL: https://doi.org/10.18653/v1/2020.emnlp-demos.6.

83. Kudo T., Richardson J. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Brussels, Belgium. Stroudsburg, PA, USA, 2018. URL: https://doi.org/10.18653/v1/d18-2012.

84. Scikit-learn: Machine Learning in Python / F. Pedregosa et al. *J. Mach. Learn. Res.* 2011. Vol. 12. P. 2825–2830. URL: http://scikit-learn.sourceforge.net.

85. FLORES-200 Dataset. URL: https://github.com/facebookresearch/flores/tree/main/flores200.

86. Bleu: A Method for Automatic Evaluation of Machine Translation / K. Papineni et al. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA, 2002. P. 311–318. URL: https://doi.org/10.3115/1073083.1073135.

87. BERTScore: Evaluating Text Generation with BERT / T. Zhang et al. *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=SkeHuCVFDr.

88. Cai S., Knight K. Smatch: an Evaluation Metric for Semantic Feature Structures. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Sofia, Bulgaria, 2013. P. 748–752. URL: https://aclanthology.org/P13-2131/.

89. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation / D. Cer et al. *Proceedings of the 11th International*

Workshop on Semantic Evaluation (SemEval-2017), Vancouver, Canada. Stroudsburg, PA, USA, 2017. URL: https://doi.org/10.18653/v1/s17-2001.

90. Modeling Relational Data with Graph Convolutional Networks / M. Schlichtkrull et al. *The Semantic Web*. Cham, 2018. P. 593–607. URL: https://doi.org/10.1007/978-3-319-93417-4_38.

91. Heaton J. Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning. *Genetic Programming and Evolvable Machines*. 2017. Vol. 19, no. 1-2. P. 305–307. URL: https://doi.org/10.1007/s10710-017-9314-z.

92. Rahutomo F., Kitasuka T., Aritsugi M. Semantic Cosine Similarity. *Proceedings of the Conference*. 2012.

93. Pearson K. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*. 1895. Vol. 58, no. 347-352. P. 240–242. URL: https://doi.org/10.1098/rspl.1895.0041.

94. Spearman C. The Proof and Measurement of Association between Two Things. *The American Journal of Psychology*. 1987. Vol. 100, no. 3/4. P. 441. URL: https://doi.org/10.2307/1422689.

95. Fawcett T. An introduction to ROC analysis. *Pattern Recognition Letters*. 2006. Vol. 27, no. 8. P. 861–874. URL: https://doi.org/10.1016/j.patrec.2005.10.010.

96. Shou Z., Lin F. Evaluate AMR Graph Similarity via Self-supervised Learning. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada. Stroudsburg, PA, USA, 2023. URL: https://doi.org/10.18653/v1/2023.acl-long.892.

97. Retrofitting Multilingual Sentence Embeddings with Abstract Meaning Representation / D. Cai et al. *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates. Stroudsburg, PA, USA, 2022. URL: https://doi.org/10.18653/v1/2022.emnlp-main.433.

98. Opitz J., Parcalabescu L., Frank A. AMR Similarity Metrics from Principles. *Transactions of the Association for Computational Linguistics*. 2020. Vol. 8. P. 522–538. URL: https://doi.org/10.1162/tacl_a_00329.

99. Focal Loss for Dense Object Detection / T.-Y. Lin et al. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2020. Vol. 42, no. 2. P. 318–327. URL: https://doi.org/10.1109/tpami.2018.2858826.

100. Van Rijsbergen C. Information Retrieval. Butterworths, 1975. URL: https://books.google.com.ua/books?id=EJ2PQgAACAAJ.

101. Словник української мови online. Томи 1-16 (А-РЯХТЛИВИЙ). URL: https://sum20ua.com/?wordid=0&page=0 (дата звернення: 01.10.2023).

102. Openai gpt-5 system card / A. Singh et al. *arXiv preprint*. 2025. URL: https://arxiv.org/abs/2601.03267.

103. Abstract Meaning Representation for Paraphrase Detection / F. Issa et al. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana. Stroudsburg, PA, USA, 2018. URL: https://doi.org/10.18653/v1/n18-1041.

104. Predicate-Argument Based Bi-Encoder for Paraphrase Identification / Q. Peng et al. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland. Stroudsburg, PA, USA, 2022. URL: https://doi.org/10.18653/v1/2022.acl-long.382.

105. Palivela H. Optimization of paraphrase generation and identification using language models in natural language processing. *International Journal of Information Management Data Insights*. 2021. Vol. 1, no. 2. P. 100025. URL: https://doi.org/10.1016/j.jjimei.2021.100025.

106. Shahmohammadi H., Dezfoulian M., Mansoorizadeh M. Paraphrase detection using LSTM networks and handcrafted features. *Multimedia Tools and Applications*. 2020. URL: https://doi.org/10.1007/s11042-020-09996-y.

107. Shakeel M. H., Karim A., Khan I. A multi-cascaded model with data augmentation for enhanced paraphrase detection in short texts. *Information Processing & Management*. 2020. Vol. 57, no. 3. P. 102204. URL: https://doi.org/10.1016/j.ipm.2020.102204.

ДОДАТОК А

Список публікацій здобувача за темою роботи

Наукові праці, в яких опубліковано основні наукові результати дисертації:

Статті у наукових виданнях, включених до Переліку фахових видань, затвердженого МОН України:

1. М’якенький А. Аналіз методів розв’язання задачі ізоморфізму при моделюванні когнітивного процесу розуміння. *Information Technology: Computer Science, Software Engineering and Cyber Security*. 2024. № 1. С. 73–79. URL: https://doi.org/10.32782/it/2024-1-9

2. М’якенький А. В., Алексєєв М. О. Порівняльний аналіз методів семантичної репрезентації текстів природної мови у задачі ідентифікації перифраз. *Applied Questions of Mathematical Modeling*. 2025. Т. 8, № 2. С. 210–223. URL: https://doi.org/10.32782/mathematical-modelling/2025-8-2-22 (особистий внесок автора – порівняльний аналіз методів ідентифікації перифраз, обґрунтування використання AMR графів)

3. М’якенький А. В. Метод побудови AMR репрезентацій українськомовних текстів із використанням моделей нейронного перекладу. *Таврійський науковий вісник. Серія: Технічні науки*. 2025. № 6. С. 114–122. URL: https://doi.org/10.32782/tnv-tech.2025.6.12

4. М’якенький А., Алексєєв О. Інформаційна технологія побудови векторних репрезентацій AMR графів з використанням графових нейронних мереж. *Information Technology: Computer Science, Software Engineering and Cyber Security*. 2026. №1. С. 183-191. URL: https://doi.org/10.32782/IT/2026-1-21 (особистий внесок автора – розробка архітектури на основі GAT, програмна реалізація та проведення експериментів)

Статті у міжнародних виданнях та виданнях України, включених до наукометричних баз даних:

5. Miakenkyi A. V., Aleksieiev M. O., Matsiuk S. M. Research on the effectiveness of using LSTM architecture in modeling the cognitive process of

recognition. *Naukovyi Visnyk Natsionalnoho Hirnychoho Universytetu*. 2025. No. 1. P. 90–95. URL: https://doi.org/10.33271/nvngu/2025-1/090 (особистий внесок автора – архітектура нейронної мережі Bi-LSTM, формування навчального датасету, аналіз результатів)

Наукові праці, які засвідчують апробацію матеріалів дисертації:

6. Алексєєв М. О., М'якенький А. В. Вибір методики навчання на підставі аналізу когнітивних моделей у системі дистанційної освіти. *Електротехнічні та інформаційні системи*. 2023. № 104. С. 3–8. URL: https://doi.org/10.32782/EIS/2023-104-1. (особистий внесок автора – розробка моделі оцінки когнітивних навичок, обґрунтування методу адаптивного добору навчальних методик)

7. М'якенький А. В. Дослідження методів моделювання когнітивних процесів. *Проблеми використання інформаційних технологій в освіті, науці та промисловості* : XVIII Міжнар. Конф., м. Дніпро, 24 листоп. 2023 р. Дніпро, 2025. С. 173–178.

8. М'якенький А. В., Алексєєв О. М. Двонапрямна LSTM модель усунення неоднозначності слів у тексті при моделюванні когнітивного процесу розуміння. *Проблеми використання інформаційних технологій в освіті, науці та промисловості* : XIX Міжнар. Конф., м. Дніпро, 27 листоп. 2024 р. Дніпро, 2025. С. 132–135.

9. М'якенький А. В. Універсальна модель репрезентації значення для мультилінгвістичного парсингу природної мови в задачі ідентифікації перифраз. *Integration of Education, Science and Business in Modern Environment: Summer Debates* : Proceedings of the 7th International Scientific and Practical Internet Conference Summer Debates. 2025. С. 171–173.

10. М'якенький А. В. Ідентифікація перифраз на основі вкладень AMR-графів за допомогою нейромережевої моделі. *International experience in scientific research* : Proceedings of IX International Scientific and Practical Conference, Chicago, 16–18 April 2026. Chicago, USA, 2026. P. 154–157.

ДОДАТОК Б

Акти впровадження та використання результатів дисертаційного дослідження

ЗАТВЕРДЖУЮ
Перший проректор
НТУ «Дніпровська політехніка»
_____ Артем ПАВЛИЧЕНКО
«15» ____04____ 2026 р.

АКТ
впровадження результатів дисертаційного дослідження М'якенького Арсенія Вячеславовича на тему: «Метод ідентифікації перифраз на основі графових репрезентацій для моделювання когнітивного процесу розуміння» в навчальний процес

Цим актом засвідчується факт того, що науково-прикладні результати досліджень дисертаційної роботи здобувача М'якенького А.В. на тему «Метод ідентифікації перифраз на основі графових репрезентацій для моделювання когнітивного процесу розуміння» упроваджено до навчального процесу Національного технічного університету «Дніпровська політехніка» в межах підготовки здобувачів вищої освіти за освітньо-науковим рівнем «Доктор філософії» та «Магістр» за спеціальністю 122 Комп'ютерні науки, а саме під час викладання фахових академічних дисциплін: «Сучасні методи і системи підтримки прийняття рішень», «Методологія наукових досліджень» та «Моделі та методи створення програмних систем паралельної та розподіленої обробки даних». У цьому контексті використано наступні теоретичні й практичні здобутки дисертаційної роботи: результати синтезу архітектури інтелектуальної системи ідентифікації перифраз, підходи та результати створення методу ідентифікації перифраз на основі семантичних графових репрезентацій, методи та засоби побудови програмно-алгоритмічного комплексу формування та обробки AMR графів.

В.о. декана ФІТ Ілля ОЛІШЕВСЬКИЙ

Заст. зав. каф. ПЗКС Андрій МАРТИНЕНКО

Науковий керівник Михайло АЛЕКСЄЄВ

ДОВІДКА

про використання результатів наукових досліджень аспіранта кафедри програмного забезпечення та комп'ютерних систем НТУ «Дніпровська політехніка» М'якенького Арсенія Вячеславовича

Результати дисертаційної роботи М'якенького А.В., поданої на здобуття наукового ступеня доктора філософії за спеціальністю 122 – Комп'ютерні науки, використані в компанії Товариство з обмеженою відповідальністю «КМК «Арт Ерія» (ЄДРПОУ: 42412162)у процесах обробки неструктурованої текстової інформації в межах виконання робіт з аналізу масивів даних та автоматизації лінгвістичного аналізу.

Запропонований автором метод на основі семантичних графових репрезентацій забезпечує автоматизовану ідентифікацію перифраз у текстах українською мовою шляхом виявлення еквівалентності між висловлюваннями незалежно від їхньої синтаксичної структури. Використання алгоритмів класифікації на основі векторних представлень графів підвищує показники точності розпізнавання дублікатів та смислових збігів у масивах неструктурованих даних. Застосування розробленої інформаційної технології впливає на процеси нормалізації текстів, зменшуючи частку семантично дубльованої інформації під час попередньої підготовки текстових даних. Впровадження результатів дослідження у програмні продукти компанії дозволяє оптимізувати алгоритми автоматичного пошуку, налаштувати фільтрацію текстових даних та знизити витрати обчислювальних ресурсів на аналіз природної мови.

Директор ТОВ «КМК «Арт Ерія» Черник Д. Р.

ДОДАТОК В

Лістинг програми

```
#model.py
import torch.nn as nn
import torch.nn.functional as F
from sts.encoder import GATEncoder, PretrainRGCNEncoder
from sts.decoder import EdgeDecoder

class SiameseAMRNetwork(nn.Module):
    def __init__(self, roberta_dim, hidden_dim, out_dim, num_relations, config):
        super().__init__()
        self.config = config
        self.encoder = GATEncoder(
            in_dim=roberta_dim,
            hidden_dim=hidden_dim,
            out_dim=out_dim,
            num_relations=num_relations,
            dropout_rate=config.get('dropout_rate', 0.2),
        )

    def forward(self, g1, g2):
        e1, e2 = self.encoder(g1), self.encoder(g2)
        if self.config.get("normalize_embeddings", True):
            e1 = F.normalize(e1, p=2, dim=1)
            e2 = F.normalize(e2, p=2, dim=1)
        return e1, e2

    def predict_similarity(self, g1, g2):
        o1, o2 = self.forward(g1, g2)
        return F.cosine_similarity(o1, o2, dim=1)

class AMRPretrainModel(nn.Module):
    def __init__(self, roberta_dim, hidden_dim, out_dim, num_relations, config):
        super().__init__()
        self.encoder = PretrainRGCNEncoder(
            in_dim=roberta_dim,
            hidden_dim=hidden_dim,
            out_dim=out_dim,
            num_relations=num_relations,
            dropout_rate=config.get('dropout_rate', 0.2),
        )
        self.decoder = EdgeDecoder()

    def forward(self, g1):
        node_embeddings = self.encoder(g1)
        return node_embeddings, g1

#decoder.py
import torch.nn as nn

class EdgeDecoder(nn.Module):
    def forward(self, node_embeddings, edges):
        source_nodes = node_embeddings[edges[0]]
        target_nodes = node_embeddings[edges[1]]
        logits = (source_nodes * target_nodes).sum(dim=1)
        return logits
```

```
#encoder.py
import torch.nn as nn
import torch.nn.functional as F
from torch_geometric.nn import RGCNConv, aggr, GATv2Conv

class PretrainGATEncoder(nn.Module):
    def __init__(self, in_dim, hidden_dim, out_dim, num_relations, dropout_rate=0.2):
        super().__init__()
        RELATION_EMBEDDING_DIM = 128
        HEADS = 8
        self.relation_embedding = nn.Embedding(num_relations, RELATION_EMBEDDING_DIM)
        self.conv1 = GATv2Conv(in_dim, hidden_dim, heads=HEADS,
edge_dim=RELATION_EMBEDDING_DIM)
        self.conv2 = GATv2Conv(hidden_dim * HEADS, out_dim, heads=1,
edge_dim=RELATION_EMBEDDING_DIM)
        self.norm1 = nn.LayerNorm(hidden_dim * HEADS)
        self.norm2 = nn.LayerNorm(out_dim)
        self.dropout = nn.Dropout(p=dropout_rate)
        self.projection1 = nn.Linear(in_dim, hidden_dim * HEADS) if in_dim != hidden_dim
* HEADS else nn.Identity()
        self.projection2 = nn.Linear(hidden_dim * HEADS, out_dim) if hidden_dim * HEADS !=
out_dim else nn.Identity()

    def forward(self, data):
        x, edge_index, edge_type = data.x, data.edge_index, data.edge_type
        edge_attr = self.relation_embedding(edge_type)

        x_res = self.projection1(x)
        x = self.conv1(x, edge_index, edge_attr=edge_attr)
        x = F.relu(self.norm1(x))
        x = self.dropout(x) + x_res

        x_res = self.projection2(x)
        x = self.conv2(x, edge_index, edge_attr=edge_attr)
        x = F.relu(self.norm2(x))
        x = x + x_res
        return x


class GATEncoder(nn.Module):
    def __init__(self, in_dim, hidden_dim, out_dim, num_relations, dropout_rate=0.2):
        super().__init__()
        RELATION_EMBEDDING_DIM = 128
        HEADS = 8
        self.relation_embedding = nn.Embedding(num_relations, RELATION_EMBEDDING_DIM)
        self.conv1 = GATv2Conv(in_dim, hidden_dim, heads=HEADS,
edge_dim=RELATION_EMBEDDING_DIM)
        self.conv2 = GATv2Conv(hidden_dim * HEADS, out_dim, heads=1,
edge_dim=RELATION_EMBEDDING_DIM)
        self.norm1 = nn.LayerNorm(hidden_dim * HEADS)
        self.norm2 = nn.LayerNorm(out_dim)
        self.dropout = nn.Dropout(p=dropout_rate)
        self.projection1 = nn.Linear(in_dim, hidden_dim * HEADS) if in_dim != hidden_dim
* HEADS else nn.Identity()
        self.projection2 = nn.Linear(hidden_dim * HEADS, out_dim) if hidden_dim * HEADS !=
out_dim else nn.Identity()
        gate_nn = nn.Sequential(
            nn.Linear(out_dim, out_dim // 2),
            nn.Tanh(),
            nn.Linear(out_dim // 2, 1),
```

```
        )
        self.pool = aggr.AttentionalAggregation(gate_nn=gate_nn)

    def forward(self, data):
        x, edge_index, edge_type, batch = data.x, data.edge_index, data.edge_type,
data.batch
        edge_attr = self.relation_embedding(edge_type)

        x_res = self.projection1(x)
        x = self.conv1(x, edge_index, edge_attr=edge_attr)
        x = F.relu(self.norm1(x))
        x = self.dropout(x) + x_res

        x_res = self.projection2(x)
        x = self.conv2(x, edge_index, edge_attr=edge_attr)
        x = F.tanh(self.norm2(x))
        x = x + x_res
        return self.pool(x, index=batch)

class RGCNEncoder(nn.Module):
    def __init__(self, in_dim, hidden_dim, out_dim, num_relations, dropout_rate=0.2):
        super().__init__()
        self.conv1 = RGCNConv(in_dim, hidden_dim, num_relations)
        self.conv2 = RGCNConv(hidden_dim, out_dim, num_relations)
        self.norm1 = nn.LayerNorm(hidden_dim)
        self.norm2 = nn.LayerNorm(out_dim)
        self.dropout = nn.Dropout(p=dropout_rate)
        self.projection1 = nn.Linear(in_dim, hidden_dim) if in_dim != hidden_dim else
nn.Identity()
        self.projection2 = nn.Linear(hidden_dim, out_dim) if hidden_dim != out_dim else
nn.Identity()
        gate_nn = nn.Sequential(
            nn.Linear(out_dim, out_dim // 2),
            nn.Tanh(),
            nn.Linear(out_dim // 2, 1),
        )
        self.pool = aggr.AttentionalAggregation(gate_nn=gate_nn)

    def forward(self, data):
        x, edge_index, edge_type, batch = data.x, data.edge_index, data.edge_type,
data.batch

        x_res = self.projection1(x)
        x = self.conv1(x, edge_index, edge_type)
        x = F.relu(self.norm1(x))
        x = self.dropout(x) + x_res

        x_res = self.projection2(x)
        x = self.conv2(x, edge_index, edge_type)
        x = F.tanh(self.norm2(x))
        x = x + x_res
        return self.pool(x, index=batch)

class PretrainRGCNEncoder(nn.Module):
    def __init__(self, in_dim, hidden_dim, out_dim, num_relations, dropout_rate=0.2):
        super().__init__()
        self.conv1 = RGCNConv(in_dim, hidden_dim, num_relations)
        self.conv2 = RGCNConv(hidden_dim, out_dim, num_relations)
        self.norm1 = nn.LayerNorm(hidden_dim)
        self.norm2 = nn.LayerNorm(out_dim)
```

```
        self.dropout = nn.Dropout(p=dropout_rate)
        self.projection1 = nn.Linear(in_dim, hidden_dim) if in_dim != hidden_dim else
nn.Identity()
        self.projection2 = nn.Linear(hidden_dim, out_dim) if hidden_dim != out_dim else
nn.Identity()

    def forward(self, data):
        x, edge_index, edge_type = data.x, data.edge_index, data.edge_type

        x_res = self.projection1(x)
        x = self.conv1(x, edge_index, edge_type)
        x = F.relu(self.norm1(x))
        x = self.dropout(x) + x_res

        x_res = self.projection2(x)
        x = self.conv2(x, edge_index, edge_type)
        x = F.relu(self.norm2(x))
        x = x + x_res
        return x

#classifier_model.py
import torch
import torch.nn as nn
import torch.nn.functional as F

class GateClassifier(nn.Module):
    def __init__(self, embedding_dim):
        super().__init__()
        input_dim = embedding_dim * 4
        self.block1 = nn.Sequential(
            nn.Linear(input_dim, 256),
            nn.BatchNorm1d(256),
            nn.GELU(),
            nn.Dropout(0.3),
        )
        self.block2 = nn.Sequential(
            nn.Linear(256, 256),
            nn.BatchNorm1d(256),
            nn.GELU(),
            nn.Dropout(0.3),
        )
        self.output_layer = nn.Linear(256, 1)

    def forward(self, u, v):
        cos_sim = F.cosine_similarity(u, v, dim=1).unsqueeze(1)
        diff = torch.abs(u - v)
        prod = u * v
        combined = torch.cat([u, v, diff, prod], dim=1)
        x = self.block1(combined)
        gate = torch.sigmoid(cos_sim)
        x = x * gate
        x = self.block2(x)
        return self.output_layer(x)

#feature_extractor.py
import re
import torch
import penman
from torch_geometric.data import Data
from transformers import RobertaTokenizer, RobertaModel
```

```
class AmrInitializer:
    def __init__(self, device='cpu', model_name='roberta-large'):
        self.device = device
        self.tokenizer = RobertaTokenizer.from_pretrained(model_name)
        self.roberta = RobertaModel.from_pretrained(model_name).to(self.device)
        self.roberta.eval()
        self.embedding_cache = {}

    @staticmethod
    def _handle_composite_concept(concept: str) -> str:
        clean_concept = re.sub(r'-\d+$', '', concept)
        return clean_concept.replace('-', ' ')

    def _get_roberta_embedding(self, text: str) -> torch.Tensor:
        if text in self.embedding_cache:
            return self.embedding_cache[text]
        with torch.no_grad():
            inputs    =    self.tokenizer(text,    return_tensors='pt',    truncation=True,
max_length=128).to(self.device)
            outputs = self.roberta(**inputs)
            embedding = outputs.last_hidden_state[:, 0, :].squeeze().cpu()
            self.embedding_cache[text] = embedding
            return embedding

    def graph_to_data(self, g: penman.Graph, relation_map: dict) -> Data:
        nodes = sorted(list(set(
            [v for v, _, _ in g.triples] +
            [v for _, _, v in g.triples if isinstance(v, str)]
        )))
        if not nodes:
            return None
        node_to_idx = {node: i for i, node in enumerate(nodes)}

        node_features = []
        for node_id in nodes:
            concept_word = "<UNK>"
            if node_id in g.variables():
                try:
                    instance_triple = next(t for t in g.instances() if t.source == node_id)
                    concept_word = self._handle_composite_concept(instance_triple.target)
                except StopIteration:
                    pass
            else:
                concept_word = str(node_id).strip('"')
            node_features.append(self._get_roberta_embedding(concept_word))

        if not node_features:
            return None
        x = torch.stack(node_features)

        edge_index, edge_type = [], []
        all_triples = g.instances() + g.edges()
        unk_rel_idx = relation_map.get('<UNK_REL>', 0)
        for t in all_triples:
            if t.source in node_to_idx and t.target in node_to_idx:
                edge_type.append(relation_map.get(t.role, unk_rel_idx))
                edge_index.append([node_to_idx[t.source], node_to_idx[t.target]])

        edge_index = (
```

```
            torch.tensor(edge_index, dtype=torch.long).t().contiguous()
            if edge_index else torch.empty((2, 0), dtype=torch.long)
        )
        edge_type = torch.tensor(edge_type, dtype=torch.long)
        return Data(x=x, edge_index=edge_index, edge_type=edge_type)

#app.py
from __future__ import annotations

from flask import Flask, jsonify
from flask_cors import CORS

from server.api.identify import bp as identify_bp
from server.api.meta import bp as meta_bp
from server.config import get_config
from server.logging_utils import configure_logging, get_logger
from server.services.pipeline import PipelineService

log = get_logger(__name__)

def create_app() -> Flask:
    configure_logging()
    cfg = get_config()

    app = Flask(__name__)
    CORS(app, origins=cfg.cors_origins)

    pipeline = PipelineService(cfg)
    app.extensions["pipeline"] = pipeline

    app.register_blueprint(meta_bp)
    app.register_blueprint(identify_bp)

    @app.errorhandler(Exception)
    def _on_error(e):
        log.exception("Unhandled error: %s", e)
        return jsonify({"error": "internal_error", "details": str(e)}), 500

    if cfg.eager_load:
        pipeline.warm_up()

    log.info("Flask app ready (device=%s, eager_load=%s)", cfg.device, cfg.eager_load)
    return app

app = create_app()

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=8000, debug=False)

#api.py
from __future__ import annotations
from flask import Blueprint, current_app, jsonify, request
from pydantic import ValidationError
from server.logging_utils import get_logger, new_request_id
from server.schemas import IdentifyRequest

bp = Blueprint("identify", __name__)
log = get_logger(__name__)
```

```
@bp.post("/api/identify")
def identify():
    new_request_id()
    payload = request.get_json(silent=True) or {}
    try:
        req = IdentifyRequest.model_validate(payload)
    except ValidationError as ve:
        return jsonify({"error": "validation_error", "details": ve.errors()}), 422

    pipeline = current_app.extensions["pipeline"]
    log.info(
        "Identify request: len1=%d len2=%d threshold=%s",
        len(req.text_uk_1),
        len(req.text_uk_2),
        req.threshold,
    )
    result = pipeline.run(req)
    return jsonify(result.model_dump()), 200

#schemas.py
from __future__ import annotations
from typing import Literal, Optional
from pydantic import BaseModel, Field

class IdentifyRequest(BaseModel):
    text_uk_1: str = Field(..., min_length=1, max_length=4000)
    text_uk_2: str = Field(..., min_length=1, max_length=4000)
    threshold: Optional[float] = Field(None, ge=0.0, le=1.0)

class TranslationBlock(BaseModel):
    en_1: str
    en_2: str
    latency_ms: float

class GraphMetrics(BaseModel):
    nodes: int
    edges: int
    depth: int
    unique_concepts: int

class AmrBlock(BaseModel):
    graph_1: str
    graph_2: str
    metrics_1: GraphMetrics
    metrics_2: GraphMetrics
    latency_ms: float

class EncodingBlock(BaseModel):
    cosine_similarity: float
    pearson_correlation: float
    spearman_correlation: float
    embedding_dim: int
    latency_ms: float

class ClassificationBlock(BaseModel):
    prob: float
    label: Literal[0, 1]
    confidence: float
    threshold: float
    mode: Literal["model", "threshold-fallback"]
```

```
    latency_ms: float

StageStatus = Literal["ok", "error", "skipped"]

class StageStatusBlock(BaseModel):
    nmt: StageStatus = "skipped"
    amr: StageStatus = "skipped"
    encoder: StageStatus = "skipped"
    classifier: StageStatus = "skipped"
    error: Optional[str] = None

class IdentifyResponse(BaseModel):
    request_id: str
    translation: Optional[TranslationBlock] = None
    amr: Optional[AmrBlock] = None
    encoding: Optional[EncodingBlock] = None
    classification: Optional[ClassificationBlock] = None
    stage_status: StageStatusBlock

#pipeline.py
from __future__ import annotations
from server.config import ServerConfig, get_config
from server.logging_utils import current_request_id, get_logger
from server.schemas import (
    AmrBlock,
    ClassificationBlock,
    EncodingBlock,
    IdentifyRequest,
    IdentifyResponse,
    StageStatusBlock,
    TranslationBlock,
)
from server.services.amr_service import AMRService
from server.services.classifier_service import ClassifierService
from server.services.encoder_service import EncoderService
from server.services.nmt_service import NMTService

log = get_logger(__name__)

class PipelineService:
    def __init__(self, config: ServerConfig | None = None):
        self.config = config or get_config()
        self.nmt = NMTService(self.config)
        self.amr = AMRService(self.config)
        self.encoder = EncoderService(self.config)
        self.classifier = ClassifierService(self.config)

    def warm_up(self):
        log.info("Warming up pipeline...")
        self.nmt._ensure_loaded()
        self.amr._ensure_loaded()
        self.encoder._ensure_loaded()
        self.classifier._ensure_loaded()
        log.info("Warm-up complete.")

    def models_loaded(self) -> dict:
        return {
            "nmt": self.nmt.is_loaded(),
            "amr": self.amr.is_loaded(),
            "encoder": self.encoder.is_loaded(),
```

```
            "classifier": self.classifier.has_model(),
        }

    def run(self, req: IdentifyRequest) -> IdentifyResponse:
        threshold = req.threshold if req.threshold is not None else
self.config.default_threshold
        status = StageStatusBlock()
        translation = amr_block = encoding = classification = None

        try:
            en_1, en_2, t_lat = self.nmt.translate_pair(req.text_uk_1, req.text_uk_2)
            translation = TranslationBlock(en_1=en_1, en_2=en_2, latency_ms=t_lat)
            status.nmt = "ok"
        except Exception as e:
            log.exception("NMT stage failed")
            status.nmt = "error"
            status.error = f"nmt: {e}"
            return IdentifyResponse(request_id=current_request_id(), stage_status=status)

        try:
            amr_1, amr_2, m1, m2, a_lat = self.amr.parse_pair(en_1, en_2)
            amr_block = AmrBlock(
                graph_1=amr_1,
                graph_2=amr_2,
                metrics_1=m1,
                metrics_2=m2,
                latency_ms=a_lat,
            )
            status.amr = "ok"
        except Exception as e:
            log.exception("AMR stage failed")
            status.amr = "error"
            status.error = f"amr: {e}"
            return IdentifyResponse(
                request_id=current_request_id(),
                translation=translation,
                stage_status=status,
            )

        try:
            e1, e2, cos, pearson, spearman, e_lat = self.encoder.encode_pair(amr_1, amr_2)
            encoding = EncodingBlock(
                cosine_similarity=cos,
                pearson_correlation=pearson,
                spearman_correlation=spearman,
                embedding_dim=self.config.output_dim,
                latency_ms=e_lat,
            )
            status.encoder = "ok"
        except Exception as e:
            log.exception("Encoder stage failed")
            status.encoder = "error"
            status.error = f"encoder: {e}"
            return IdentifyResponse(
                request_id=current_request_id(),
                translation=translation,
                amr=amr_block,
                stage_status=status,
            )
```

```
        try:
            res = self.classifier.classify(e1, e2, cos, threshold)
            classification = ClassificationBlock(**res)
            status.classifier = "ok"
        except Exception as e:
            log.exception("Classifier stage failed")
            status.classifier = "error"
            status.error = f"classifier: {e}"

        return IdentifyResponse(
            request_id=current_request_id(),
            translation=translation,
            amr=amr_block,
            encoding=encoding,
            classification=classification,
            stage_status=status,
        )
```

```
#nmt_service.py
from __future__ import annotations
import time
from typing import Tuple
from server.config import ServerConfig
from server.logging_utils import get_logger
log = get_logger(__name__)

class NMTService:
    def __init__(self, config: ServerConfig):
        self.config = config
        self._pipe = None

    def _ensure_loaded(self):
        if self._pipe is not None or self.config.use_mock_nmt:
            return
        from transformers import pipeline

        log.info("Loading NMT model: %s", self.config.nmt_model_name)
        self._pipe = pipeline(
            "translation",
            model=self.config.nmt_model_name,
            device=0 if self.config.device == "cuda" else -1,
        )

    def is_loaded(self) -> bool:
        return self.config.use_mock_nmt or self._pipe is not None

    def translate_pair(self, text_1: str, text_2: str) -> Tuple[str, str, float]:
        start = time.perf_counter()
        if self.config.use_mock_nmt:
            en_1 = f"[MOCK-EN] {text_1}"
            en_2 = f"[MOCK-EN] {text_2}"
        else:
            self._ensure_loaded()
            outputs = self._pipe([text_1, text_2], max_length=512)
            en_1 = outputs[0]["translation_text"]
            en_2 = outputs[1]["translation_text"]
        latency = (time.perf_counter() - start) * 1000.0
        return en_1, en_2, latency

#amr_service.py
```

```
from __future__ import annotations
import json
import os
import time
from pathlib import Path
from typing import Tuple
import penman
from server.config import ServerConfig
from server.logging_utils import get_logger
from server.schemas import GraphMetrics

log = get_logger(__name__)

def _graph_depth(g: penman.Graph) -> int:
    if not g.triples:
        return 0
    children: dict[str, list[str]] = {}
    for src, role, tgt in g.triples:
        if role == ":instance":
            continue
        if isinstance(tgt, str):
            children.setdefault(src, []).append(tgt)

    top = g.top
    if top is None:
        return 0

    best = 0
    stack: list[tuple[str, int, set[str]]] = [(top, 0, {top})]
    while stack:
        node, depth, seen = stack.pop()
        best = max(best, depth)
        for child in children.get(node, []):
            if child in seen:
                continue
            stack.append((child, depth + 1, seen | {child}))
    return best

def compute_metrics(g: penman.Graph) -> GraphMetrics:
    nodes = set(g.variables())
    edges = [t for t in g.triples if t[1] != ":instance"]
    concepts = {t.target for t in g.instances()}
    return GraphMetrics(
        nodes=len(nodes),
        edges=len(edges),
        depth=_graph_depth(g),
        unique_concepts=len(concepts),
    )

_MOCK_AMR_TEMPLATE = "(x / unknown :ARG0 (y / thing))"

class AMRService:
    def __init__(self, config: ServerConfig):
        self.config = config
        self._stog = None
        self._resolved_model_dir: str | None = None

    def _ensure_amrlib_compat_config(self, model_dir: Path) -> None:
        config_path = model_dir / "config.json"
        if not config_path.exists():
```

```
            return

        config_data = json.loads(config_path.read_text(encoding="utf-8"))
        task_specific_params = config_data.get("task_specific_params") or {}
        parse_amr = task_specific_params.get("parse_amr") or {}
        parse_amr.update(
            {
                "model_name_or_path":                 parse_amr.get("model_name_or_path",
self.config.amr_hf_repo),
                "tok_name_or_path": str(model_dir),
                "max_in_len":                            int(parse_amr.get("max_in_len",
config_data.get("max_position_embeddings", 1024))),
                "max_out_len":                          int(parse_amr.get("max_out_len",
config_data.get("max_position_embeddings", 1024))),
            }
        )
        task_specific_params["parse_amr"] = parse_amr
        config_data["task_specific_params"] = task_specific_params
        config_path.write_text(json.dumps(config_data,    ensure_ascii=True,    indent=2),
encoding="utf-8")

        tokenizer_config_path = model_dir / "tokenizer_config.json"
        if tokenizer_config_path.exists():
            tokenizer_config  =  json.loads(tokenizer_config_path.read_text(encoding="utf-
8"))
            if tokenizer_config.get("tokenizer_class") == "AMRBartTokenizer":
                tokenizer_config["tokenizer_class"] = "BartTokenizer"
                tokenizer_config_path.write_text(
                    json.dumps(tokenizer_config, ensure_ascii=True, indent=2),
                    encoding="utf-8",
                )

        meta_path = model_dir / "amrlib_meta.json"
        if not meta_path.exists():
            meta_path.write_text(
                json.dumps(
                    {
                        "model_type": "stog",
                        "version": "hf",
                        "date": "2026-05-05",
                        "inference_module": ".parse_xfm.inference",
                        "inference_class": "Inference",
                        "model_fn": "pytorch_model.bin",
                        "base_model": self.config.amr_hf_repo,
                        "kwargs": {},
                    },
                    ensure_ascii=True,
                ),
                encoding="utf-8",
            )

    def _resolve_model_dir(self, amrlib_module) -> str:
        if self._resolved_model_dir is not None:
            return self._resolved_model_dir

        explicit_dir = self.config.amr_model_dir.strip()
        if explicit_dir:
            if not os.path.isdir(explicit_dir):
                raise  FileNotFoundError(f"Configured  AMR_MODEL_DIR  does  not  exist:
{explicit_dir}")
```

```
            self._ensure_amrlib_compat_config(Path(explicit_dir))
            self._resolved_model_dir = explicit_dir
            return self._resolved_model_dir

        package_default = os.path.join(amrlib_module.defaults.data_dir, "model_stog")
        if os.path.isdir(package_default):
            self._ensure_amrlib_compat_config(Path(package_default))
            self._resolved_model_dir = package_default
            return self._resolved_model_dir

        project_cache_dir = self.config.project_root / "data" / "amrlib"
        known_candidates = [
            project_cache_dir / "model_stog",
            project_cache_dir / "model_parse_xfm_bart_large-v0_1_0",
            project_cache_dir / "model_stog_hf",
        ]
        for candidate in known_candidates:
            if candidate.is_dir() and (candidate / "amrlib_meta.json").exists():
                self._ensure_amrlib_compat_config(candidate)
                self._resolved_model_dir = str(candidate)
                return self._resolved_model_dir

        from huggingface_hub import snapshot_download

        hf_dir = project_cache_dir / "model_stog_hf"
        snapshot_download(
            repo_id=self.config.amr_hf_repo,
            local_dir=str(hf_dir),
            local_dir_use_symlinks=False,
            resume_download=True,
        )
        self._ensure_amrlib_compat_config(hf_dir)

        self._resolved_model_dir = str(hf_dir)
        return self._resolved_model_dir

    def _ensure_loaded(self):
        if self._stog is not None or self.config.use_mock_amr:
            return
        import amrlib

        model_dir = self._resolve_model_dir(amrlib)
        log.info("Loading amrlib STOG model (dir=%r)...", model_dir)
        self._stog = amrlib.load_stog_model(model_dir=model_dir)

    def is_loaded(self) -> bool:
        return self.config.use_mock_amr or self._stog is not None

    def parse_pair(self, en_1: str, en_2: str) -> Tuple[str, str, GraphMetrics,
GraphMetrics, float]:
        start = time.perf_counter()
        if self.config.use_mock_amr:
            amr_1 = amr_2 = _MOCK_AMR_TEMPLATE
        else:
            self._ensure_loaded()
            graphs = self._stog.parse_sents([en_1, en_2])
            amr_1 = graphs[0] or _MOCK_AMR_TEMPLATE
            amr_2 = graphs[1] or _MOCK_AMR_TEMPLATE

        g1 = penman.decode(amr_1)
```

```
        g2 = penman.decode(amr_2)
        m1 = compute_metrics(g1)
        m2 = compute_metrics(g2)
        latency = (time.perf_counter() - start) * 1000.0
        return amr_1, amr_2, m1, m2, latency

#encoder_service.py
from __future__ import annotations
import time
from typing import Tuple
import penman
import torch
import torch.nn.functional as F
from torch_geometric.data import Batch, Data
from server.amr_relations import build_relation_map
from server.config import ServerConfig
from server.logging_utils import get_logger
from sts.feature_extractor import AmrInitializer
from sts.model import SiameseAMRNetwork

log = get_logger(__name__)

class _MockInitializer:
    def __init__(self, embedding_dim: int):
        self.embedding_dim = embedding_dim

    def graph_to_data(self, g: penman.Graph, relation_map: dict) -> Data:
        nodes = sorted(list(set(
            [v for v, _, _ in g.triples] +
            [v for _, _, v in g.triples if isinstance(v, str)]
        )))
        if not nodes:
            return None
        node_to_idx = {n: i for i, n in enumerate(nodes)}
        x = torch.zeros((len(nodes), self.embedding_dim))

        edge_index, edge_type = [], []
        unk = relation_map.get("<UNK_REL>", 0)
        for t in g.instances() + g.edges():
            if t.source in node_to_idx and t.target in node_to_idx:
                edge_type.append(relation_map.get(t.role, unk))
                edge_index.append([node_to_idx[t.source], node_to_idx[t.target]])
        ei = (
            torch.tensor(edge_index, dtype=torch.long).t().contiguous()
            if edge_index else torch.empty((2, 0), dtype=torch.long)
        )
        et = torch.tensor(edge_type, dtype=torch.long)
        return Data(x=x, edge_index=ei, edge_type=et)

class EncoderService:
    def __init__(self, config: ServerConfig):
        self.config = config
        self.relation_map = build_relation_map(config.num_relations)
        self._initializer = None
        self._model: SiameseAMRNetwork | None = None

    def _ensure_loaded(self):
        if self._model is not None:
            return
        if self.config.use_mock_roberta:
```

```
            self._initializer = _MockInitializer(self.config.embedding_dim)
        else:
            self._initializer = AmrInitializer(
                device=self.config.device,
                model_name=self.config.roberta_model_name,
            )

        model = SiameseAMRNetwork(
            roberta_dim=self.config.embedding_dim,
            hidden_dim=self.config.hidden_dim,
            out_dim=self.config.output_dim,
            num_relations=self.config.num_relations,
            config={"dropout_rate": 0.0, "normalize_embeddings": True},
        ).to(self.config.device)

        state = torch.load(self.config.sts_checkpoint, map_location=self.config.device,
weights_only=False)
        state_dict = state["model_state"] if isinstance(state, dict) and "model_state" in
state else state
        model.load_state_dict(state_dict, strict=False)
        model.eval()
        self._model = model

    @staticmethod
    def _pearson_correlation(emb_1: torch.Tensor, emb_2: torch.Tensor) -> float:
        left = emb_1.reshape(-1).float()
        right = emb_2.reshape(-1).float()
        left = left - left.mean()
        right = right - right.mean()
        denom = torch.linalg.vector_norm(left) * torch.linalg.vector_norm(right)
        if torch.isclose(denom, torch.tensor(0.0, device=denom.device)):
            return 0.0
        return torch.dot(left, right).div(denom).item()

    @staticmethod
    def _rank_tensor(values: torch.Tensor) -> torch.Tensor:
        flat = values.reshape(-1)
        order = torch.argsort(flat)
        ranks = torch.empty_like(flat, dtype=torch.float32)
        ranks[order] = torch.arange(1, flat.numel() + 1, device=flat.device,
dtype=torch.float32)
        return ranks

    @classmethod
    def _spearman_correlation(cls, emb_1: torch.Tensor, emb_2: torch.Tensor) -> float:
        return cls._pearson_correlation(cls._rank_tensor(emb_1), cls._rank_tensor(emb_2))

    @torch.no_grad()
    def encode_pair(self, amr_1: str, amr_2: str) -> Tuple[torch.Tensor, torch.Tensor,
float, float, float, float]:
        self._ensure_loaded()
        start = time.perf_counter()

        g1 = penman.decode(amr_1)
        g2 = penman.decode(amr_2)
        d1 = self._initializer.graph_to_data(g1, self.relation_map)
        d2 = self._initializer.graph_to_data(g2, self.relation_map)

        b1 = Batch.from_data_list([d1]).to(self.config.device)
        b2 = Batch.from_data_list([d2]).to(self.config.device)
```

```
        e1, e2 = self._model(b1, b2)
        cos = F.cosine_similarity(e1, e2, dim=1).item()
        pearson = self._pearson_correlation(e1, e2)
        spearman = self._spearman_correlation(e1, e2)
        latency = (time.perf_counter() - start) * 1000.0
        return e1, e2, cos, pearson, spearman, latency

#classifier_service.py
from __future__ import annotations
import time
from typing import Optional
import torch
from server.config import ServerConfig
from server.logging_utils import get_logger
from sts.classifier_model import GateClassifier

log = get_logger(__name__)


class ClassifierService:
    def __init__(self, config: ServerConfig):
        self.config = config
        self._model: Optional[GateClassifier] = None
        self._tried_load = False

    def _ensure_loaded(self):
        if self._tried_load:
            return
        self._tried_load = True
        ckpt = self.config.classifier_checkpoint
        if not ckpt.exists():
            return
        model = GateClassifier(self.config.output_dim).to(self.config.device)
        state = torch.load(ckpt, map_location=self.config.device, weights_only=False)
        if isinstance(state, dict) and "model_state" in state:
            state = state["model_state"]
        model.load_state_dict(state, strict=False)
        model.eval()
        self._model = model

    def has_model(self) -> bool:
        self._ensure_loaded()
        return self._model is not None

    @torch.no_grad()
    def classify(self, emb_1: torch.Tensor, emb_2: torch.Tensor, cosine_similarity: float,
threshold: float) -> dict:
        start = time.perf_counter()
        self._ensure_loaded()

        if self._model is not None:
            logits = self._model(emb_1, emb_2).squeeze(-1)
            prob = torch.sigmoid(logits).item()
            mode = "model"
        else:
            prob = (cosine_similarity + 1.0) / 2.0
            mode = "threshold-fallback"

        label = 1 if prob >= threshold else 0
```

```
        confidence = prob if label == 1 else (1.0 - prob)
        latency = (time.perf_counter() - start) * 1000.0
        return {
            "prob": float(prob),
            "label": int(label),
            "confidence": float(confidence),
            "threshold": float(threshold),
            "mode": mode,
            "latency_ms": float(latency),
        }
```

```
#App.tsx
import { useEffect, useState } from "react";
import { fetchHealth, identify } from "./api";
import { type HealthResponse, type IdentifyResponse } from "./api";
import { ResultsPanel } from "./components/ResultsPanel";

type Stage = "nmt" | "amr" | "encoder" | "classifier";

const stages: Array<[string, Stage]> = [
  ["UA → EN", "nmt"],
  ["AMR", "amr"],
  ["Encoder", "encoder"],
  ["Decision", "classifier"],
];

function ready(h: HealthResponse | null) {
  return !!h && (h.models_loaded.nmt || h.mocks.nmt)
    && (h.models_loaded.amr || h.mocks.amr)
    && h.models_loaded.encoder;
}


function status(
  h: HealthResponse | null,
  r: IdentifyResponse | null,
): Record<Stage, boolean> {
  const s = r?.stage_status as any;
  if (s) return {
    nmt: s.nmt === "ok",
    amr: s.amr === "ok",
    encoder: s.encoder === "ok",
    classifier: s.classifier === "ok",
  };
  return {
    nmt: !!h && (h.models_loaded.nmt || h.mocks.nmt),
    amr: !!h && (h.models_loaded.amr || h.mocks.amr),
    encoder: !!h?.models_loaded.encoder,
    classifier: !!h,
  };
}


export function App() {
  const [text1, setText1] = useState("");
  const [text2, setText2] = useState("");
  const [threshold, setThreshold] = useState(0.5);
  const [busy, setBusy] = useState(false);
  const [health, setHealth] = useState<HealthResponse | null>(null);
  const [result, setResult] = useState<IdentifyResponse | null>(null);
```

```
const [error, setError] = useState<string | null>(null);
const [healthError, setHealthError] = useState<string | null>(null);

useEffect(() => {
  let active = true;
  async function refresh() {
    try {
      const next = await fetchHealth();
      if (active) {
        setHealth(next);
        setHealthError(null);
      }
    } catch (e: any) {
      if (active) {
        setHealthError(e?.message ?? String(e));
      }
    }
  }
  void refresh();
  const timer = window.setInterval(refresh, 5000);
  return () => {
    active = false;
    window.clearInterval(timer);
  };
}, []);


async function run() {
  if (!text1.trim() || !text2.trim() || busy) return;
  setBusy(true);
  setError(null);
  setResult(null);
  try {
    setResult(
      await identify({
        text_uk_1: text1,
        text_uk_2: text2,
        threshold,
      }),
    );
  } catch (e: any) {
    setError(e?.message ?? String(e));
  } finally {
    setBusy(false);
  }
}

const st = status(health, result);
const device = result?.device ?? (health as any)?.device ?? "device: n/a";


return (
  <main className="pp-shell">
    <section className="pp pp--i">
      <header className="pp-header">
        <h1 className="pp-title">Ідентифікація перифраз</h1>
        <div className="chip-row">
          {stages.map(([label, key]) => (
            <span
              key={key}
```

```
        className={`chip ${st[key] ? "ok" : "wait"}`}
      >
        {label}
      </span>
    ))}
    <span className="chip device">{device}</span>
  </div>
</header>

<section className="card input-card">
  <h2>
    <span>1</span> Вхідні дані
  </h2>
  <label>
    Висловлення 1 <span>UA</span>
  </label>
  <textarea
    value={text1}
    onChange={(e) => setText1(e.target.value)}
  />
  <label>
    Висловлення 2 <span>UA</span>
  </label>
  <textarea
    value={text2}
    onChange={(e) => setText2(e.target.value)}
  />
  <div className="action-row">
    <label>
      Поріг класифікації: {threshold.toFixed(2)}
      <input
        type="range"
        min={0.05}
        max={0.95}
        step={0.01}
        value={threshold}
        onChange={(e) => setThreshold(Number(e.target.value))}
      />
    </label>
    <button
      disabled={!ready(health) || busy}
      onClick={run}
    >
      {busy ? "Оброблення..." : "Ідентифікувати"}
    </button>
  </div>
  {healthError && (
    <p className="error">API: {healthError}</p>
  )}
</section>

<nav className="pipeline">
  {stages.map(([label, key], i) => (
    <span key={key} className={st[key] ? "ok" : "wait"}>
      {i + 1}. {label}
    </span>
  ))}
</nav>

{error && <p className="error">Помилка запиту: {error}</p>}
```

```
        {result
          ? <ResultsPanel result={result} threshold={threshold} />
          : (
            <section className="card">
              Результати з'являться після виконання конвеєра.
            </section>
          )}
      </section>
    </main>
  );
}


#ResultsPanel.tsx
import { type GraphMetrics, type IdentifyResponse } from "../api";

const f = (v?: number, d = 4) =>
  Number.isFinite(v) ? v!.toFixed(d)
    : "н/д";


function Metrics({ m }: { m: GraphMetrics }) {
  return (
    <div className="metrics">
      <span>Вузли: {m.nodes}</span>
      <span>Ребра: {m.edges}</span>
      <span>Глибина: {m.depth}</span>
      <span>Концепти: {m.unique_concepts}</span>
    </div>
  );
}


function Amr({
  title, graph, metrics, latency,
}: {
  title: string; graph: string; metrics: GraphMetrics; latency?: number;
}) {
  return (
    <article className="amr-card">
      <h3>{title} {latency !== undefined && <span>{latency.toFixed(0)} ms</span>}</h3>
      <pre>{graph}</pre>
      <Metrics m={metrics} />
    </article>
  );
}


export function ResultsPanel({
  result, threshold,
}: {
  result: IdentifyResponse; threshold: number;
}) {
  const positive = result.classification?.label === 1;

  return (
    <>
      {result.classification && (
        <section className={`verdict ${positive ? "positive" : "negative"}`}>
          <h2>{positive ? "Перифраз виявлено" : "Перифраз не виявлено"}</h2>
```

```
            <p>
              P(перифраза) = {(result.classification.prob * 100).toFixed(1)}%;
              впевненість = {(result.classification.confidence * 100).toFixed(1)}%;
              поріг = {threshold.toFixed(2)};
              cosine = {f(result.encoding?.cosine_similarity)}
            </p>
          </section>
        )}

        {result.translation && (
          <section className="card">
            <h2><span>2</span> Нейронний переклад UA → EN</h2>
            <p><b>Висловлення 1:</b> {result.translation.en_1}</p>
            <p><b>Висловлення 2:</b> {result.translation.en_2}</p>
            <p>{result.translation.latency_ms.toFixed(0)} ms</p>
          </section>
        )}

        {result.amr && (
          <section className="card">
            <h2><span>3</span> AMR-репрезентації (PENMAN)</h2>
            <Amr
              title="Висловлення 1"
              graph={result.amr.graph_1}
              metrics={result.amr.metrics_1}
              latency={result.amr.latency_ms}
            />
            <Amr
              title="Висловлення 2"
              graph={result.amr.graph_2}
              metrics={result.amr.metrics_2}
            />
          </section>
        )}

        {result.encoding && (
          <section className="card metrics">
            <span>Cosine: {f(result.encoding.cosine_similarity)}</span>
            <span>Pearson: {f(result.encoding.pearson_correlation)}</span>
            <span>Spearman: {f(result.encoding.spearman_correlation)}</span>
          </section>
        )}
      </>
    );
}
```