Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Кваліфікаційна наукова
праця на правах рукопису

ШАПТАЛА РОМАН ВІТАЛІЙОВИЧ

УДК 004.89

ДИСЕРТАЦІЯ

КЛАСИФІКАЦІЯ ДОКУМЕНТІВ НА ОСНОВІ ВЕКТОРНИХ ПРЕДСТАВЛЕНЬ СЛОВНИКІВ ПРИ ОБРОБЦІ ПРИРОДНОЇ МОВИ У МАЛОРЕСУРСНОМУ СЕРЕДОВИЩІ

122 – Комп’ютерні науки

Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____Р.В. Шаптала

Науковий керівник: Кисельов Геннадій Дмитрович, кандидат технічних наук, доцент

Київ – 2023

АНОТАЦІЯ

Шаптала Р.В. Класифікація документів на основі векторних представлень словників при обробці природної мови у малоресурсному середовищі. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 122 «Комп'ютерні науки». – Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», 2023.

Метою дисертаційного дослідження є розробка та вдосконалення методів класифікації документів, написаних природною мовою, у малоресурсному середовищі за допомогою побудови векторних графових представлень словників природної мови.

Проблема обробки природної мови у малоресурсному середовищі є складною через брак даних, які можна використовувати для тренування сучасних алгоритмів машинного навчання. Малоресурсне середовище може бути викликано відсутністю чи дорожнечою ручної розмітки на початкових етапах побудови інформаційних систем, а також низькими популярністю та розвитком природної мови у інформаційному просторі.

Типові методи обробки природної мови вимагають наявність розмітки текстів відповідно до задачі, яка розв'язується. Така розмітка часто виконується експертами у прикладній галузі або людьми з високим рівнем лінгвістичної грамотності. Анотатори, які здатні виконувати розмітку, можуть не бути доступними для великого набору проєктів через відсутність фінансування (зазвичай це тимчасова робота, яка може не виправдати операційні кошти) чи мотивації (анотування – рутинна та одноманітна робота).

Зважаючи на те, що 63% контенту Інтернету написано англійською мовою, і більшість мов представлена менш ніж 1% веб-сторінок, величезна кількість мов є малоресурсними та, відповідно, менш дослідженими з точки зору підходів до обробки природних мов. Це призводить до того, що інформаційні системи, які вимушені

працювати на основі малопредставлених мов, часто потерпають від низької якості, порівняно з їх англомовними аналогами.

Тому, покращення вже існуючих та розробка нових методів обробки природної мови у малоресурсному середовищі є актуальною задачею.

У **першому розділі** дисертаційної роботи проведено аналітичний огляд методів та досліджень за темою дисертації. На основі огляду було сформовано класифікацію методів обробки природної мови у малоресурсному середовищі та виділено їх основні припущення, переваги і недоліки. У ході аналізу було з'ясовано, що існуючі методи для обробки природної мови у малоресурсному середовищі вимагають додаткових даних з суміжним до прикладної області змістом, що часто є не виправданим припущенням. Тому дослідження було спрямовано на розробку нових методів з використанням мінімальної кількості сторонніх даних, а саме – лінгвістичних словників, які існують у великій кількості мов як додатковий ресурс.

Використання словникової інформації не є чимось новим – їх не рідко використовують системи побудовані на правилах. Проблема таких підходів – вони явно роблять запити до словників чи онтологій (побудованих на їх основі) щодо зв'язків між сутностями у прикладних текстах. Таким чином, поєднання сучасних методів обробки природної мови та словникової інформації ускладнюється, адже перші оперують з векторними представленнями лінгвістичних сутностей, а другі – є дискретними елементами у графових структурах. Тому методи перетворення інформації, що міститься у словниках, у вектори, з якими можуть працювати новітні підходи машинного навчання у контексті обробки природної мови можуть дозволити більш ефективно розширити уявлення прикладних систем про мову, аніж явні правила пошуку у словниках. Через це було сформовано наступне наукове завдання: «Розробка методів обробки природної мови на основі векторних представлень словників у малоресурсному середовищі».

У **другому розділі** представлено загальну методику побудови векторних представлень словників та їх поєднання з методами обробки природної мови.

Проведено теоретичні дослідження щодо можливості отримання таких представлень, їх бажаних властивостей та шляхів застосування. Отримано класифікацію методів побудови векторних представлень графів, виділено їх ознаки та обмеження. До таких методів відносяться методи на основі факторизації, такі як HOPE, факторизація графу, Лапласівські проекції, GraRep, LLE; методи на основі випадкових блукань, такі як Node2Vec, DeepWalk та Walklets; методи на основі глибокого навчання, такі як SDNE, DNGR та GCN; та інші. Описано яким чином дані методи можна застосувати для моделювання словників та варіанти модифікацій алгоритмів для роботи зі словниковими даними.

Також було проаналізовано та упорядковано методи злиття кількох векторних представлень для отримання фінальних векторів, які можна використовувати для різноманітних задач обробки природної мови, наприклад класифікації документів. При цьому було виділено ті, що практично застосовувати у малоресурсному середовищі з обмеженим розміром розмічених даних, а саме метод конкатенації та зваженої суми векторних представлень. Для використання даних методів у контексті злиття векторних представлень документів на основі слів та словників була запропонована модифікація даних методів через додавання етапу пошуку відповідності слів.

У **третьому розділі** описано результати експериментальних досліджень. Для перевірки впливу різних методів побудови векторних представлень словників, а також злиття векторних представлень словників та методів обробки природної мови у малоресурсному середовищі на результати моделювання у практичному завданні, було обрано вирішення задачі класифікації документів. Експериментальні дослідження проведено у прикладній області містобудування та урбаністики, а саме класифікації петицій до Київської міської ради за напрямами, такими як транспорт, освіта, благоустрій тощо. В якості додаткової словникової інформації, на основі якої будуються векторні представлення для поєднання з типовими методами класифікації документів, було обрано словник синонімів української мови. Для розуміння методів

передобробки та формулювання практичних рекомендацій при роботі з подібними даними, у цьому розділі було детально описано та проаналізовано обидва джерела інформації. Малоресурсність середовища забезпечено через два аспекти вирішуваної задачі – петиції написані українською мовою, яка входить до третього десятка найпоширеніших мов світу та має невелику кількість якісних наборів даних для покращення якості роботи моделей, а також малим розміром набору даних при високій змістовній варіативності петицій.

Результати проведених досліджень показали, що векторні представлення словників на основі методів кодування вершин графів можна поєднувати з типовими векторними представленнями документів для покращення якості класифікації документів за допомогою підходів машинного навчання. Кожен крок запропонованого методу має набір параметрів та гіперпараметрів, від яких залежить результат та ефективність фінального рішення. Тому додатково наведено аналіз даних опцій, а також порівняння різних підходів до побудови представлень вершин графів у контексті словників. Для досягнення найкращих результатів пропонується використання методу на основі випадкових блукань - Node2Vec, який перетворює елементи словника у вектори за прийнятний час, не вимагає багато ресурсів та отримує вищі оцінки при подальшій класифікації документів. Для наступного кроку, а саме злиття векторних представлень документів та словникової інформації оптимальним виявився метод зваженої суми. Додатково наводяться практичні рекомендації по роботі з подібними даними, а саме особливості отримання, збереження та передобробки документів, побудови словників для кожного з методів класифікації документів, збереження та обробки словника синонімів, а також аналіз статистичної значущості результатів.

Наукова новизна одержаних результатів полягає у наступному:

1. Вперше **запропоновано метод** класифікації документів на основі векторних представлень словників при обробці природної мови у малоресурсному середовищі, який відрізняється від методів доповнення даних, що базуються на

словниках, тим що у ньому поєднуються векторні представлення документів з векторними представленнями елементів лінгвістичних словників, що дозволяє збільшити F1-міру якості класифікації документів у малоресурсному середовищі;

2. **Запропоновано векторну модель** слів зі словника синонімів, яка на відміну від інших будується на основі векторних представлень вузлів графу словника, що надає можливість її повторного використання в різних задачах обробки природної мови через трансферне навчання;
3. **Модифіковано методи** конкатенації та зваженої суми при злитті векторних представлень слів додаванням етапу пошуку відповідності слів з документу словам з словника синонімів, що дозволяє покрити відсутні у словнику словоформи без побудови моделей визначення частини мови та пошуку словоформ, що суттєво ускладнено у малоресурсних середовищах.

Практичне значення одержаних результатів полягає у тому, що:

1. Розроблений метод дозволяє значно **підвищити F1-міру якості** систем класифікації документів у малоресурсних середовищах. Таким чином розробники даних систем можуть **зменшити час та витрати на розробку**, адже вища якість системи досягатиметься з меншою кількістю розмітки, розширення якої може бути не доступним, або вимагати додаткових часових чи фінансових інвестицій;
2. Розроблено векторні представлення слів у словнику синонімів української мови, які можна **перевикористовувати** за допомогою трансферного навчання при створенні програмних систем у інших прикладних областях;
3. **Представлено набір даних** для класифікації тем петицій, націлений на тестування методів обробки природної мови у малоресурсному середовищі. Документи написані українською мовою та мають вузьку урбаністичну спеціалізацію, що робить набір даних відмінним від корпусів загального призначення;

4. Запропоновано **застосування розробленого методу до класифікації петицій** до Київської міської ради за темами, яка дозволяє автоматично пропонувати тему петиції при ручній розмітці, що може суттєво скоротити час на їх аналіз.

Ключові слова: математичне моделювання, інтелектуальний аналіз даних, машинне навчання, обробка природної мови, малоресурсне середовище, класифікація документів, нейронні мережі, векторні представлення документів, векторні представлення графів, злиття векторних представлень, Node2Vec.

ABSTRACT

Roman Shaptala. Dictionary embeddings for document classification in low-resource natural language processing. – Qualification scientific work as manuscript.

Doctor of Philosophy dissertation under 122 «Computer Science» specialty. – National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute," Kyiv, 2023.

The objective of this research is to develop and improve document classification methods in low-resource natural language processing through graph embeddings of linguistic dictionaries.

The field of low-resource language processing (NLP) is complicated because of the lack of data that can be used for modern machine learning methods training. Low-resource NLP settings can be caused by the absence or expensiveness of manual labeling during the first stages of information systems development, as well as low popularity and development of a natural language in the informational space.

Common NLP methods require labeled corpuses based on the task at hand. The labeling process is usually done by domain experts or people with high level of linguistic proficiency. However, annotators might not be available for a lot of projects because of costs (this work is temporary and might not cover the operational cost) or motivation (annotation is a routine and monotonous work).

Considering that 63% of the Internet is written in English, and most of natural languages are represented in less than 1% of all web pages, a lot of natural languages are considered low-resource, and are less researched in the field of natural language processing. This leads to information systems built to work with low-resource languages having lower quality than their English counterparts.

Consequently, improving existing low-resource natural language processing methods and the development of new ones is a relevant research problem.

In the **first chapter** of the dissertation, an analytical review of methods and research on the topic of the dissertation is carried out. Based on the review, a classification of low-

resource natural language processing methods was formed and their main assumptions, advantages and disadvantages were highlighted. During the analysis, it was found that existing methods for processing natural language in a low-resource environment require additional data with content adjacent to the application area, which is often an unjustified assumption. Therefore, the research was aimed at developing new methods using a minimum amount of extraneous data, namely linguistic dictionaries that exist in a large number of languages as an additional resource.

The use of dictionary information is not new - they are frequently used by rule-based systems. The problem with such approaches is that they explicitly query dictionaries or ontologies (built on their basis) regarding the relationships between entities in application texts. Thus, the combination of modern natural language processing methods and dictionary information is complicated, because the former operate with vector representations of linguistic entities, and the latter are discrete elements in graph structures. Therefore, methods of embedding information contained in dictionaries into vectors that can be used by the latest machine learning approaches in the context of natural language processing can allow to expand the understanding of language by the applied systems more effectively than explicit rules that query dictionaries. As a result, the following scientific task was formed: “Development of low-resource natural language processing methods based on dictionary vector representations”.

The **second chapter** presents the general methodology for building dictionary vector representations and their combination with natural language processing methods. Theoretical studies have been conducted on the possibility of obtaining such representations, their desired properties and ways of application. A classification of methods for constructing vector representations of graphs was obtained, their features and limitations were highlighted. These include methods based on factorization, such as HOPE, graph factorization, Laplacian projections, GraRep, LLE; methods based on random walks, such as Node2Vec, DeepWalk, and Walklets; deep learning-based methods such as SDNE,

DNGR, and GCN; and other. It is described how these methods can be applied to modeling dictionaries and options for modifying algorithms for working with dictionary data.

Methods for multiple vector representations fusion were also analyzed and organized. These allow to obtain final features that can be used for a variety of natural language processing tasks, such as document classification. At the same time, only some of them are practical to use in a low-resource environment with a limited size of labeled data, namely, the methods of concatenation and weighted sum of vector representations. To use these methods in the context of fusion of vector representations of documents based on words and dictionaries, a modification was proposed by adding a word-dictionary matching step.

The **third chapter** describes the results of experimental research. To test the influence of different methods of building vector representations of dictionaries, as well as the fusion of vector representations of dictionaries and methods of natural language processing in a low-resource environment in a practical task, document classification was chosen. Experimental studies were carried out in the domain of city planning and urbanism, namely, the classification of petitions to the Kyiv City Council in areas such as transport, education, landscaping, etc. As additional dictionary information, on the basis of which vector representations are built for combination with typical methods of document classification, a dictionary of synonyms of the Ukrainian language was chosen. In order to understand the methods of preprocessing and formulate practical recommendations when working with such data, this section describes exploratory data analysis of both sources of information. The lack of resources in the experiment environment is guaranteed by two aspects of the problem - the petitions are written in Ukrainian, which is only around thirtieth most widely spoken languages in the world and has a small number of high-quality data sets to improve the quality of the models, as well as the size of the data set which includes high content variability of the petitions.

The results of the research showed that vector representations of dictionaries based on graph node embedding methods can be combined with common vector representations of documents to improve the quality of document classification using machine learning

approaches. Each step of the proposed method has a set of parameters and hyperparameters, which the result and effectiveness of the final solution depend on. Therefore, an analysis of these options is additionally given, as well as a comparison of different approaches to the construction of graph node embeddings in the context of dictionaries. To achieve the best results, it is suggested to use random-walk based method - Node2Vec, which converts dictionary elements into vectors in an acceptable time, does not require a lot of resources, and receives higher F1-scores further down the pipeline – during document classification. For the next step, namely the fusion of vector representations of documents and dictionary information, the weighted sum method turned out to be better than concatination. In addition, practical recommendations for working with such data are provided, namely, the process of obtaining, saving and preprocessing documents for each of the proposed methods, saving and processing of a synonyms dictionary, as well as the analysis of statistical significance of the results.

Scientific novelty of the results includes:

1. For the first time, a method of document classification based on dictionary embeddings during low-resource natural language processing is proposed, which differs from dictionary-based methods of data augmentation in that it fuses vector representations of documents with vector representations of elements of linguistic dictionaries, which allows to increase F1-score of document classification in a low-resource environment;
2. A vector model of words from the dictionary of synonyms is proposed, which, unlike others, is built on the basis of vector representations of the nodes of the dictionary graph, which makes it possible to reuse it in various tasks of natural language processing through transfer learning;
3. The methods of concatenation and weighted sum during vector representations of words fusion have been modified by adding a stage of matching words from the document to words from the dictionary of synonyms, which allows for covering word forms missing from the dictionary without building models for part of speech tagging

and word form generation, which is significantly complicated in low-resource environments.

The practical significance of the results includes:

1. The proposed method makes it possible to significantly increase the F1-score of document classification systems in low-resource environments. This way, developers of these systems can reduce development time and costs, because higher system quality will be achieved with less labeling, the process which may not be available or require additional time or financial investment;
2. Vector representations of words in the dictionary of synonyms of the Ukrainian language were developed, which can be reused with the help of transfer learning when creating software systems in other applied areas;
3. A data set for the classification of petition topics is presented, aimed at testing low-resource natural language processing methods. The documents are written in Ukrainian and have a narrow urban specialization, which makes the data set different from general-purpose corpora;
4. It is proposed to apply the developed method to the topic classification of petitions to the Kyiv City Council, which allows for automatic suggestions of topic for the petition during manual labeling. This can significantly reduce the time for their analysis.

Keywords: mathematical modelling, data mining, machine learning, natural language processing, low-resource environment, document classification, neural networks, document embeddings, graph embeddings, embeddings fusion, Node2Vec.

Список публікацій здобувача

Публікації у виданнях, проіндексованих у Scopus

[1] R. Shaptala and G. Kyselov, “Enhancing document representations with synonyms graph node embeddings,” *J. Theor. Appl. Inf. Technol.*, vol. 100, no. 1, pp. 70–80, Jan. 2022.

Статті у наукових фахових виданнях України

[2] Р. Шаптала і Г. Кисельов, «Метод злиття багатомодальних векторних представлень слів у малоресурсному середовищі», *ВОТТП*, вип. 1, с. 174–179, Бер. 2023.

[3] Р. Шаптала і Г. Кисельов, “Класифікація текстових документів з використанням доповнення векторних представлень документів графовими представленнями елементів словника синонімів,” *Інформаційні технології та суспільство*, вип. 3 (5), с. 49–55, Січ. 2023.

[4] Р. Шаптала і Г. Кисельов, “Огляд методів злиття векторних представлень,” *Телекомунікаційні та інформаційні технології*, вип. 4 (77), с. 84–89, 2022.

[5] R. V. Shaptala and G. D. Kyselev, “Using graph embeddings for Wikipedia link prediction,” *Bull. Natl. Tech. Univ. “KhPI”. Ser. Syst. Anal. Control Inf. Technol.*, vol. 0, no. 1 SE-INFORMATION TECHNOLOGY, pp. 48–52, Jul. 2019.

[6] Shaptala R.V. and Kyselov G.D., “Vector space models of Kyiv city petitions,” *Sci. notes Taurida Natl. V.I. Vernadsky Univ. Ser. Tech. Sci.*, vol. 32, no. 1, pp. 169–177, 2021.

Наукові праці, які засвідчують апробацію матеріалів дисертації та інші публікації

[7] A. Samvelyan, R. Shaptala, and G. Kyselov, “Exploratory data analysis of Kyiv city petitions,” in *2020 IEEE 2nd International Conference on System Analysis Intelligent Computing (SAIC)*, 2020, pp. 1–4.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

NLP – Обробка природної мови (Natural language processing)

OCR – Оптичне розпізнавання символів (Optical character recognition)

BOW – Мішок слів (Bag-of-words)

CBOW – Неперервний Мішок Слів (Continuous Bag-of-Words)

TF-IDF – Term Frequency-Inverse Document Frequency

BFS – Пошук у ширину (Breadth-first search)

DFS – Пошук у глибину (Depth-first search)

EDA – Дослідницький аналіз даних (Exploratory data analysis)

Київрада – Київська міська рада

КМДА – Київська міська державна адміністрація

UMAP – Апроксимація та проекція рівномірного многовиду (Uniform manifold approximation and projection)

t-SNE – t-розподіленого представлення стохастичної близькості (t-distributed stochastic neighbor embedding)

PCA – Метод головних компонент (Principal component analysis)

ВСТУП

Обґрунтування вибору теми дослідження. Обробка природної мови є однією з найбільш прикладних областей застосування комп'ютерних наук та штучного інтелекту. Накопичення знань та методів для автоматичної обробки текстової інформації, згенерованої у процесі людської соціальної активності є надважливими завданнями, адже дозволяють людям взаємодіяти з комп'ютерами їхніми рідними мовами, а не спеціалізованими інтерфейсами на кшталт мов програмування. Основна ціль розвитку даної науки – надати здатність обчислювальній техніці «розуміти» зміст документів, враховуючи контекст та усі мовні нюанси використані у них. Саме концентрація на розумінні робить задачі обробки природної мови надзвичайно складними, адже лише тексту часто не вистачає для повного осмислення суті сказаного чи написаного – присутність знань про навколишній світ та можливі способи взаємодії з ним є невід'ємними складовими людської комунікації.

Вищезазначена проблема проявляє себе ще більше при обробці природної мови у малоресурсних середовищах. Малоресурсні середовища – це такі умови вирішення задач обробки природної мови, коли доступність даних, потрібних комп'ютерній системі щоб навчитись розуміти мову у прикладній області, обмежена. Виділяють дві основні причини існування малоресурсних середовищ [1]:

1. природна мова не має достатньої кількості джерел для отримання якісних даних через свою непопулярність чи малий рівень розвитку лінгвістичних наук;
2. певна спеціалізована прикладна область робить розмітку чи отримання нових якісних даних неможливими або надто дорогими, адже експертів у даній області – обмаль.

Найбільш вагомі та значні внески у розвиток області обробки природної мови у роботах закордонних авторів Кріса Меннінга [2], Дена Журафського [3], Себастіана Рудера [4], Річарда Зохера [5] та інших [6], [7]. Проте більшість з них досліджувало розвиток методів при наявності великої кількості даних, особливо в еру «великих даних», або при роботі з англійською мовою, для якої наявно багато розмічених та

нерозмічених даних у різних прикладних областях та контекстах. При цьому розвиток автоматичних підходів аналізу української мови забезпечувався українськими науковцями у сфері комп'ютерної лінгвістики, такими як Наталія Дарчук [8], Андрій Романюк [9], Євгенія Карпіловська [10], Наталія Бардіна [11], Валентина Перебийніс [12] і Тетяна Грязнухіна [13]. У цей же час, вплив малоресурсності середовища на результати роботи існуючих методів та способи подолання обмежень пов'язаних з відсутністю великої кількості даних вивчались значно менше та переважно іноземними вченими Барбарою Планк [14], Джейсоном Вей [15], Гьозде Гуль Сахіном [16], Майклом Хеддеріком [1] та іншими [17], [18], за останні 10 років. Себастіан Рудер назвав обробку природної мови у малоресурсних середовищах однією з чотирьох найбільших відкритих проблем у сучасній обробці природної мови [19]. Існування великої кількості малоресурсних мов та прикладних областей з спеціалізованими завданнями робить розвиток методів обробки природної мови у малоресурсних середовищах актуальною задачею.

Мета і завдання дослідження. Метою дисертаційного дослідження є розробка та вдосконалення методів класифікації документів, які написані природною мовою, у малоресурсному середовищі за допомогою побудови векторних графових представлень словників природної мови.

Для досягнення поставленої мети необхідно розв’язати наступні завдання:

- проаналізувати існуючі методи обробки природної мови у малоресурсних середовищах, виокремити їх переваги та недоліки, припущення на яких вони базуються та запропонувати варіанти їх покращення;
- проаналізувати існуючі методи побудови графових представлень, виділити ті, які можна застосувати для створення представлень словникових даних;
- вдосконалити методи обробки природної мови у малоресурсних середовищах на основі графових представлень словників;
- проаналізувати та обрати математичні методи злиття векторних представлень;

- побудувати програмні реалізації запропонованих методів та порівняти їх з існуючими.

Об'єкт дослідження. Об'єктом дисертаційного дослідження є процеси обробки природної мови.

Предмет дослідження. Предметом дисертаційного дослідження виступають математичні методи та моделі для обробки природної мови у малоресурсних середовищах.

Методи дослідження. Інтелектуальні методи обробки даних, математичні моделі та методи обробки природної мови, статистичний аналіз, структурно-функціональний аналіз, теорія штучних нейронних мереж та глибокого навчання, теорії злиття ознак та мультимодального машинного навчання. Дані методи були обрані зважаючи на мету та завдання дослідження, а також на практичність їх використання при вирішенні сучасних проблем обробки природної мови та штучного інтелекту.

Наукова новизна отриманих результатів. Наукова новизна одержаних результатів полягає у наступному:

1. Вперше **запропоновано метод** класифікації документів на основі векторних представлень словників при обробці природної мови у малоресурсному середовищі, який відрізняється від методів доповнення даних, що базуються на словниках, тим що у ньому поєднуються векторні представлення документів з векторними представленнями елементів лінгвістичних словників, що дозволяє збільшити F1-міру якості класифікації документів у малоресурсному середовищі;
2. **Запропоновано векторну модель** слів зі словника синонімів, яка на відміну від інших будується на основі векторних представлень вузлів графу словника, що надає можливість її повторного використання в різних задачах обробки природної мови через трансферне навчання;

3. **Модифіковано методи** конкатенації та зваженої суми при злитті векторних представлень слів додаванням етапу пошуку відповідності слів з документу словам з словника синонімів, що дозволяє покрити відсутні у словнику словоформи без побудови моделей визначення частини мови та пошуку словоформ, що суттєво ускладнено у малоресурсних середовищах.

Практичне значення отриманих результатів. Практичне значення результатів, отриманих у ході дисертаційного дослідження, зводиться до наступного переліку:

1. Розроблений метод дозволяє значно підвищити якість систем класифікації документів у малоресурсних середовищах. Таким чином розробники даних систем можуть зменшити час та витрати на розробку, адже вища якість системи досягатиметься з меншою кількістю розмітки, розширення якої може бути не доступним, або вимагати додаткових часових чи фінансових інвестицій;
2. Розроблено векторні представлення слів у словнику синонімів української мови, які можна перевикористовувати за допомогою трансферного навчання при створенні програмних систем у інших прикладних областях;
3. Представлено набір даних для класифікації тем петицій, націлений на тестування методів обробки природної мови у малоресурсному середовищі. Документи написані українською мовою та мають вузьку урбаністичну спеціалізацію, що робить набір даних відмінним від корпусів загального призначення;
4. Сформовано рекомендації розробникам, які бажають покращити якість роботи існуючих моделей при обробці природної мови у малоресурсних середовищах через інкорпорацію словникової інформації, включаючи вибір методу злиття векторних представлень, вибір методу побудови графових представлень слів зі словника синонімів та гіперпараметрів даних методів;

5. Запропоновано застосування розробленого методу до класифікації петицій до Київської міської ради за темами, яка дозволяє автоматично пропонувати тему петиції при ручній розмітці, що може суттєво скоротити час на їх аналіз.

Особистий внесок здобувача. Усі основні результати дисертаційного дослідження, представлені до захисту, одержані автором особисто. У публікаціях у співавторстві, здобувачеві належать: розробка та експериментальне підтвердження ефективності методу доповнення векторних представлень документів за допомогою векторних представлень вузлів графу словника синонімів у [1]; представлення, аналіз та опис набору даних електронних петицій до Київської міської ради, опис процесу передобробки та класифікації мови петицій, побудова векторних моделей даних документів у [2]; аналіз методів побудови векторних представлень графів для прогнозу посилань у Вікіпедії, проведення та опис експериментів з порівняння даних методів у [3].

Апробація матеріалів дисертації. Основні положення та отримані наукові результати, що викладені в даній дисертаційній роботі, пройшли апробацію на міжнародній науково-технічній конференції “2020 IEEE 2nd International Conference on System Analysis Intelligent Computing (SAIC)”.

Публікації.

Публікації у виданнях, проіндексованих у Scopus

[1] R. Shaptala and G. Kyselov, “Enhancing document representations with synonyms graph node embeddings,” *J. Theor. Appl. Inf. Technol.*, vol. 100, no. 1, pp. 70–80, Jan. 2022.

Статті у наукових фахових виданнях України

[2] Р. Шаптала і Г. Кисельов, «Метод злиття багатомодальних векторних представлень слів у малоресурсному середовищі», *ВОТТП*, вип. 1, с. 174–179, Бер. 2023.

[3] Р. Шаптала і Г. Кисельов, “Класифікація текстових документів з використанням доповнення векторних представлень документів графовими

представленнями елементів словника синонімів,” *Інформаційні технології та суспільство*, вип. 3 (5), с. 49–55, Січ. 2023.

[4] Р. Шаптала і Г. Кисельов, “Огляд методів злиття векторних представлень,” *Телекомунікаційні та інформаційні технології*, вип. 4 (77), с. 84–89, 2022.

[5] R. V. Shaptala and G. D. Kyselev, “Using graph embeddings for Wikipedia link prediction,” *Bull. Natl. Tech. Univ. “KhPI”. Ser. Syst. Anal. Control Inf. Technol.*, vol. 0, no. 1 SE-INFORMATION TECHNOLOGY, pp. 48–52, Jul. 2019.

[6] Shaptala R.V. and Kyselov G.D., “Vector space models of Kyiv city petitions,” *Sci. notes Taurida Natl. V.I. Vernadsky Univ. Ser. Tech. Sci.*, vol. 32, no. 1, pp. 169–177, 2021.

Структура та обсяг дисертації. Дисертація складається із анотації, вступу, трьох розділів, висновків, списку використаних джерел та додатків. Загальний обсяг дисертації становить 151 сторінку, у тому числі: 110 сторінок основного тексту, 26 рисунків, 10 таблиць, список використаних джерел із 181 найменувань на 20 сторінках.

РОЗДІЛ 1. ОБРОБКА ПРИРОДНОЇ МОВИ У МАЛОРЕСУРСНОМУ СЕРЕДОВИЩІ

1.1 Обробка природної мови

Природними мовами називають будь-які мови, що виникли природним шляхом та були вживані людьми у процесі їх існування. Такі мови були створені без штучного втручання, планування чи узгодження окремими людьми чи організаціями. Саме це відрізняє їх від штучно створених і формальних мов, наприклад мов комп'ютерного програмування або регулярних мов, що використовуються у теорії автоматів. Попри наявність синтаксису в мов програмування чи діалектової варіації у деяких представників тваринного світу, такі приклади мов не вважаються природними у даному контексті. Природні мови можуть приймати різні форми, наприклад, мовленнєву чи письмову. Людська вербальна комунікація відбувається за допомогою голосу та слуху і є формою прояву природної мови. У цей же час текстова комунікація відбувається через написання та зчитування символьної інформації і також є формою прояву природної мови. Голос і текст – це те, яким чином людство спілкується між собою. Враховуючи важливість цих комунікаційних каналів для повсякденного життя кожного, методи для розуміння та обробки природної мови мають велику практичну цінність.

Обробка природної мови (англ. Natural language processing, NLP) - це область, яка перебуває на межі лінгвістики, інформатики та штучного інтелекту, та присвячена вивченню взаємодії між комп'ютерними системами та природною мовою. У широкому сенсі, обробка природної мови визначається як автоматичне маніпулювання природною мовою у її мовленнєвих та текстових формах за допомогою програмного забезпечення. Головною метою даної області є створення комп'ютерних систем, що здатні «розуміти» усі аспекти людської комунікації за допомогою природних мов. Це включає у собі інтерпретацію контексту, загальні знання про лінгвістичну структуру природних мов та здатність оновлювати їх з розвитком людства. Часті задачі в обробці природної мови включають розпізнавання

мовлення, розуміння природної мови та генерація природної мови. Технології, побудовані на основі методів обробки природної мови, здатні отримувати інформацію та ідеї, що містяться в текстових або аудіо документах, а також класифікувати й упорядковувати самі документи.

Задачі обробки природної мови розділяють за формами даних, з якими вони працюють, а також за рівнем одиниць мови, що фігурують в них. За формами даних виокремлюють обробку природного мовлення (задачі на основі голосової комунікації) та обробку текстів (задачі на основі символьних даних). Іноді до задач обробки природної мови також відносять оптичне розпізнавання символів (англ. Optical character recognition, OCR), у якій на зображенні шукають та видобувають текст, але дана задача більшою мірою лежить у області комп'ютерного зору, а обробка природних мов лише допомагає формувати повноцінний текст з послідовностей виокремлених символів. Прикладами задач обробки природного мовлення є розпізнавання мовлення (конвертація голосу в текст), сегментація мовлення (поділ аудіо людського мовлення на структурні частини такі як слова) та текст-у-мовлення (конвертація тексту в голосову репрезентацію, протилежно до розпізнавання мовлення). Високорівневими задачами обробки текстів є класифікація текстів, автоматичний підсумок текстів, генерація діалогу, виправлення помилок, машинний переклад та генерація відповідей на питання. Більшість інших задач обробки природної мови є допоміжними та часто є підзадачами високорівневих задач. Одиницями мови є звук, буква, слово, частина мови, словосполучення та речення. Кожна з цих одиниць має набір задач, які працюють на її рівні. Також природна мова складається з набору систем: фонетичної, лексичної, морфологічної та синтаксичної. Відповідно до даних систем виділяється інша класифікація задач обробки природної мови (наведена на рис. 1.1). Найнижчий морфологічний рівень досліджує словотворення, походження та зміну форм слів залежно від контексту. На цьому рівні опрацьовуються слова, які розглядаються як послідовності символів. На основі морфологічного рівня будується синтаксичний рівень. Він відповідає за грамотну

побудову та структуру речень. Часто даний рівень називають граматичним і це найбільш досліджений рівень у комп'ютерній лінгвістиці. Синтаксис працює з реченнями, які розглядаються як послідовності слів. Наступний рівень – семантичний. У ньому досліджуються проблеми розуміння сенсу речень. Семантика працює з реченнями, які розглядаються як послідовності слів з певними значеннями та ролями. Найвищий рівень – прагматичний, відповідає за розуміння загального змісту тексту, його тем та ідей. Він працює з текстами, які розглядаються як послідовності речень з певним змістом.

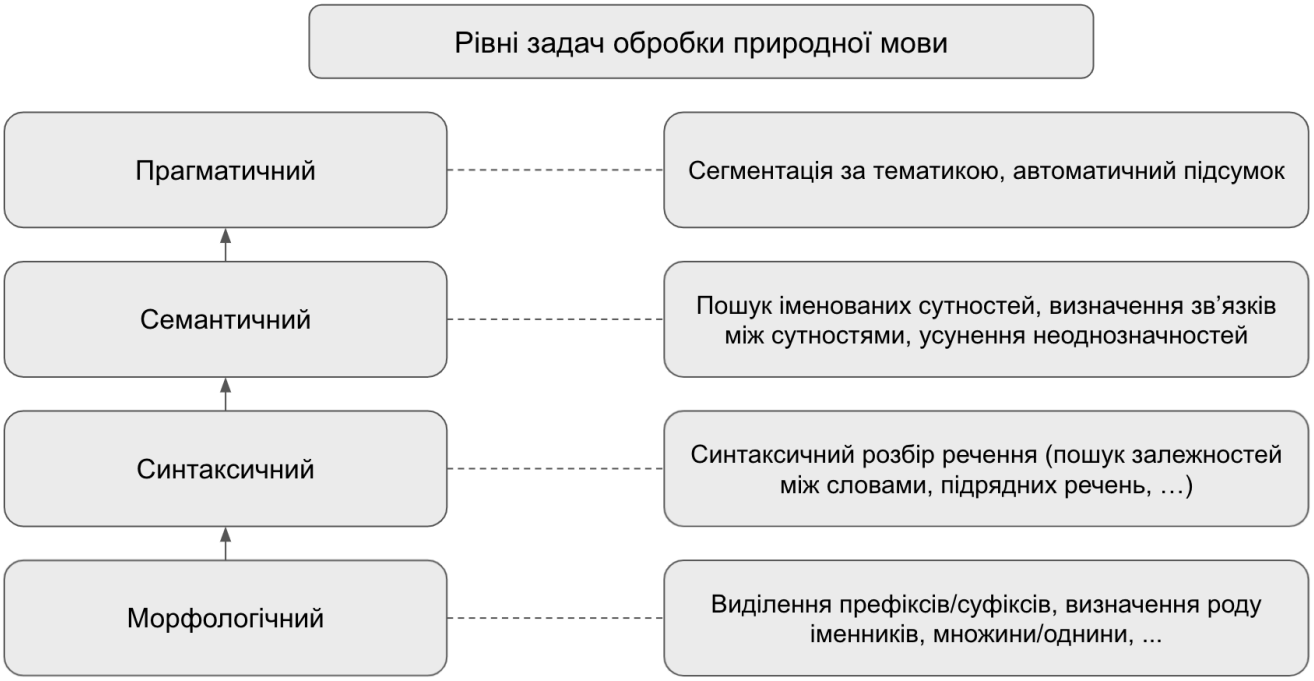

Рис. 1.1 – Рівні задач обробки природної мови

У даній дисертації досліджуються задачі класифікації та кластеризації текстів, семантичного зв'язку слів та текстів, а також допоміжна підзадача розбиття документів на слова (токенізація). Задачі обробки природного мовлення не розглядаються.

Вивчення обробки природної мови проводиться вже більше 70 років і розвинулось з області лінгвістики через появу обчислювальної техніки. За цей час область змінила набір парадигм та підходів і досягла неабияких результатів у задачах

машинного перекладу, класифікації документів, розпізнавання мовлення та інших. Класифікація основних методів обробки природної мови, а також деякі приклади підходів для окремих класів таких методів, наведені на рис. 1.2.

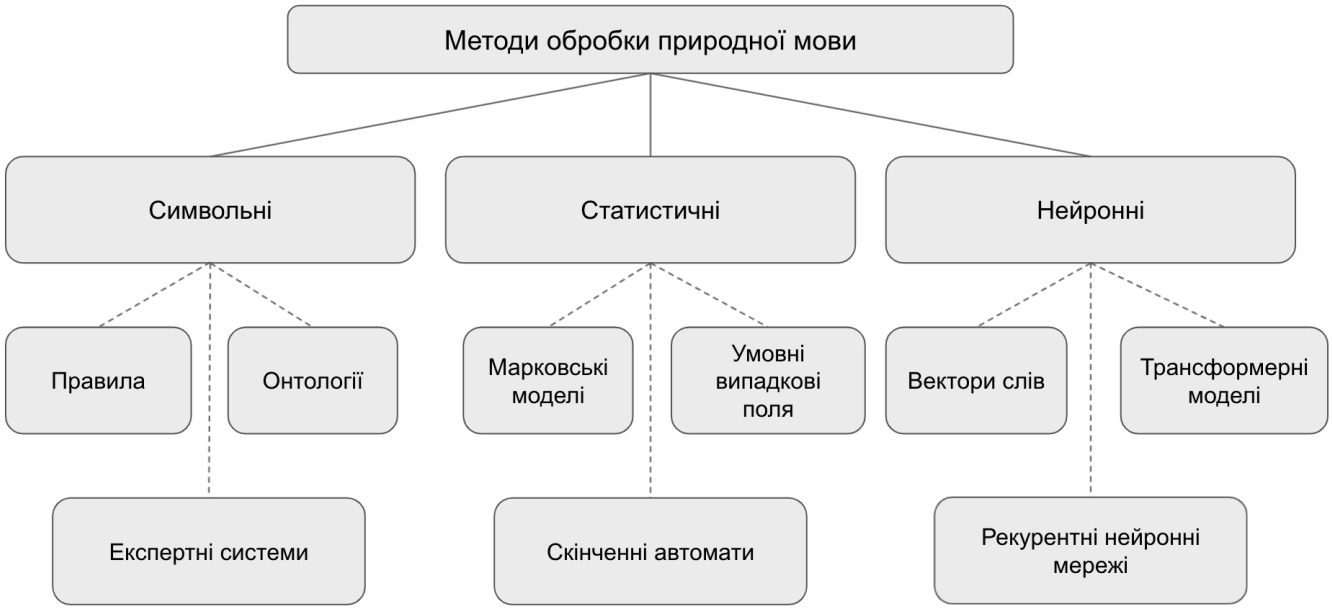

Рис. 1.2 – Методи обробки природної мови та їх приклади

Символьні методи обробки природної мови виникли першими та були популярними у часи 1950-1980их років. Вони спирались на ручне виокремлення закономірностей людської комунікації та створення машин, які їх відтворюють. Одним з перших серйозних досягнень обробки природної мови був Джорджтаунський експеримент [20], у результаті якого було перекладено 60 речень написаних російською мовою на англійську. Система складалась з шести граматичних правил та 250 лексем, виважено розроблених фахівцями з Джорджтаунського університету та компанії International Business Machines (IBM). Очевидно, що така система не мала ніякої здатності до узагальнення чи адаптації під усю варіативність природної мови, навіть у контексті двомовного машинного перекладу. Іншим відомим прикладом успішного застосування символьних методів обробки природних мов є діалогова система ELIZA [21] створена у Массачусетському технологічному інституті (MTI). Система працювала на основі пошуку ключових слів та їх вставки у шаблонні

речення-відповіді. Попри свою простоту та високу специфічність прикладної області (психологія) даного рішення, ELIZA змогла показати, що людське спілкування у багатьох випадках відбувається за певними сценаріями, а отже може бути ефективно імітоване за допомогою автоматичних систем. Основною проблемою розвитку методів обробки природної мови того часу була відсутність у таких систем знань про навколишнє середовище та зовнішній світ. Щоб боротись з цим обмеженням, багато дослідників [22], [23], [24] почали розробляти онтології – структурні представлення, ієрархії та визначення концепцій окремих прикладних областей чи усього світу. Це дозволило додавати правила, які залежать від контексту, а також частково розуміти зв'язки між окремими сутностями всередині текстових джерел даних. Попри переваги створення онтологій для доповнення моделей розумінням навколишнього світу, їх наповнення та підтримка вимагає великої кількості експертних рішень, а отже є дорогою та важко розширюваною. Цей недолік присутній у всіх символьних методів обробки природних мов – для того, щоб їх ефективно використовувати, потрібно побудувати набір правил, словників, онтологій, які б максимально широко покривали прикладну область. Враховуючи різноманіття лінгвістичних конструкцій та світових фактів, а також невизначеність та протиріччя багатьох тверджень, подальший розвиток символьних методів обробки природних мов надзвичайно складний, що привело до зменшення їх популярності після 1980их років. Варто зауважити, що дані методи все ще дуже активно використовуються у низькорівневих задачах обробки природних мов, таких як токенізація, стемінг, морфологічний аналіз, а також усіх інших задачах де влучність моделей має пріоритет над покриттям.

Статистичні методи обробки природної мови стали популярні у 1980их і були найбільш ефективними методами для вирішення багатьох практичних задач до 2010их. На відміну від символьних методів, вони пропонували знаходити закономірності у наборах даних автоматично, замість побудови ручних правил та онтологій. Це спонукало швидкий та жвавий розвиток даної вітки обробки природної мови, адже за цей час людство почало генерувати все більше і більше даних

текстового формату, які можна було використати у статистичних моделях. Більше того з розвитком статистичного інструментарію, більшість методів статистичної обробки природної мови дозволяють оцінити впевненість у своїх прогнозах, що надзвичайно важливо у такій багатозначній системі як природна мова. Наприклад, при побудові моделей мови у реченні «На небі з'явились темні хмари, мабуть буде …» замість символа «…» можна було б вставити одне з наступних слів: «дощ», «сніг» чи «гроза», і тоді коли символьний метод обрав те, що записано в правилі, статистичний метод міг би повідомити, що слово «дощ» найбільш ймовірне з певною впевненістю, яку можна врахувати у подальшому аналізі. До сьогодні дані методи залишаються релевантними та широко застосовуються, особливо у задачах, де важливе розуміння прийнятих рішень.

Починаючи з 2015го року статистичні методи обробки природної мови почали поступатись у популярності нейромережевим. Основним недоліком, який виправляють останні, є необхідність власноручно створювати ознаки для статистичних моделей, займатись так званою інженерією ознак. Нейронні методи обробки природної мови значно зменшили кількість підзадач, які потрібно розв'язати для вирішення високорівневих задач, адже почали це робити неявно усередині наскрізного підходу. Крім того, ітеративна природа навчання таких підходів, розвиток швидкості обробки даних та стрімкий ріст розмірів наборів даних для тренування дозволили нейромережевим підходам значно покращити якість роботи систем, що базуються на обробці природних мов. Основними недоліками даного класу методів є те, що вони вимагають багато даних та обчислювальних ресурсів для досягнення високих результатів, а також відсутність можливості інтерпретувати роботу моделей.

У даній дисертації досліджується набір нейронних методів обробки природної мови при вирішенні задачі класифікації документів у малоресурсному середовищі, пропонується покращення одного з таких методів, а також наводиться порівняння їх

роботи зі статистичними методами, які історично розвивались за обмеженої кількості даних.

1.2 Малоресурсні середовища

Попри популярність досліджень у області обробки природної мови у малоресурсних середовищах сам термін «малоресурсне середовище» історично мав змінне та нечітке визначення. Наприклад, Малоресурсна Мова для Виникаючих Інцидентів (англ. Low Resource Language for Emergent Incidents, LORELEI) – американський проект розвитку малоресурсних мов [25] – визначає малоресурсні середовища як ті, для яких не існує автоматизованих технологій обробки природної мови. Дане визначення фокусується лише на аспекті малоресурсності, який спричинений відсутністю якісної мовної анотації, ігноруючи можливу специфічність завдання, через яку технології можуть бути відсутні. Інші автори, наприклад Бермент [26], пропонують набір евристик для визначення низькоресурсності середовища, які включають наявність мінімальних наборів даних для різних підзадач обробки природної мови, інструментів та людських ресурсів. Такі евристики, хоч і створюють певну відправну точку для визначення малоресурсних середовищ, часто ігнорують додаткові джерела даних, які можуть бути наявні, наприклад пресу, за що їх неодноразово критикували [27]. У даній дисертації пропонується наступне визначення малоресурсного середовища:

«Середовище називається малоресурсним для певного завдання тоді коли не існує алгоритму, який би вирішував його з адекватною ефективністю на основі поточно наявних даних.»

Основою даного визначення є зв’язок малоресурсності з певним завданням обробки природної мови. Так, наприклад, завдання класифікації сентименту відгуків англійською мовою на фільми вирішене з 97.4% точності [28] через наявність розміченого набору даних з 50000 відгуків [29], тоді як таке ж завдання українською мовою не має достатньої кількості даних для вирішення. З іншого боку, навіть для англійської мови завдання пошуку сутностей у документах технічного спрямування

буде малоресурсним через обмежену кількість розмічених даних у даній області. Тому обробка природної мови у малоресурсних середовищах фокусується на розвитку методів та алгоритмів, які допоможуть розв'язувати подібні задачі незважаючи на присутні обмеження, незалежно викликані вони непопулярністю, малим рівнем розвитку технологій конкретної природної мови чи специфічністю конкретної прикладної області чи обома факторами.

У світі наявно 7139 живих природних мов [30]. Більшість з них використовуються в Азії, Африці та Південній Америці. Попри таку велику кількість та варіативність природних мов, майже усі дослідження у обробці природних мов присвячені англійській мові та малій кількості інших високоресурсних мов. Список мов, що відзначаються наявністю великої кількості ресурсів дуже обмежений; англійська, китайська (мандаринська), арабська та французька. Німецька, португальська, іспанська та фінська мови перебувають на межі між мало- та багаторесурсними мовами. Очікується, що дані мови стануть багаторесурсними в найближчий час через збільшення популярності їх досліджень. Перелічені мови мають великі колекції оцифрованого тексту та записаного мовлення, більша частина яких транскрибована, часто у вільному доступі. Крім цього для них доступні набори анотованих лінгвістичних ресурсів, такі як колекції граматичних дерев розбору текстів та набори для оцінок якості побудованих моделей для великої кількості завдань обробки природної мови. Також для багаторесурсних мов уже побудована велика кількість готових інструментів обробки природної мови, таких як морфологічні, синтаксичні аналізатори, детектори іменованих сутностей тощо. Станом на серпень 2019 року, карта Європейської асоціації мовних ресурсів (англ. European Language Resources Association, ELRA [31]) містила 961 ресурс для англійської мови (додатково 121 для американської англійської), 216 для німецької, 180 для французької, 130 для іспанської, 103 для китайської та 103 для японської мов. Єдиними іншими мовами, у яких налічувалось більше 50 ресурсів, є португальська, італійська, нідерландська, арабська та чеська. Решта близько 7000 мов світу не мають

і цієї кількості наборів даних [32]. Для української мови станом на вересень 2021го року карта ресурсів мов даної асоціації включає лиш один набір даних.

Більшість прикладних областей теж є малоресурсними середовищами. Біомедичний текст, юридичні документи, літературні твори – це все приклади специфічних доменів, для яких моделі, які працюють добре на текстах загального спрямування, показують гірші результати [33]. Більше того, зазвичай коли знаходиться нове застосування для методів обробки природної мови, дані для тренування та перевірки роботи створених рішень відсутні або дуже дорогі та повинні бути створені силами зацікавлених сторін. Цей факт - одна з найважливіших мотивацій для розробки нових методів обробки природної мови у малоресурсних середовищах.

Очевидно, що малоресурсні середовища можуть перетворюватись у багаторесурсні з розвитком інструментів розмітки, збільшенням інтересу до їх дослідження, появою нових джерел даних. Наприклад, до 2015 року жоден метод вирішення задачі генерації діалогів не досягав адекватних результатів через відсутність великих наборів даних для тренування. Але з появою нових джерел розмічених даних таких як OpenSubtitles (1689 субтитри діалогів з фільмів та серіалів) [34], Ubuntu Dialogue Corpus (1 мільйон діалогів з форума операційної системи Ubuntu) [35] та CMU Document Grounded Conversations Dataset (транскрипти 8939 типових людських діалогів) [36] генерація діалогів швидко розвинулась та показує гарні результати [37], які знайшли застосування у діалогових системах (чат-ботах).

Важливо також зазначити, що обробка природної мови у малоресурсних середовищах не рівна навчанню з нульовим, одним або кількома кадрами (англ. zero-, one- and few-shot learning). Хоча обидві області намагаються впоратися з нестачею навчальних даних та інформації усередині них, підходи до навчання з кількома кадрами досліджують екстрем – ситуації, коли наявно лише кілька прикладів певних класів або взагалі їх немає. Обробка природної мови з низькими ресурсами, з іншого боку, є більш загальним терміном, вона може передбачати навчання з кількома

кадрами, а може і не мати таких обмежень. Дана область описує типові проблеми обробки природної мови у середовищі, де поточно доступних даних недостатньо для високоякісних результатів.

1.3 Методи обробки природної мови у малоресурсному середовищі

Усі методи обробки природної мови у малоресурсному середовищі намагаються найбільш ефективно використати наявні ресурси або знайти додаткові неявні джерела інформації, які могли б допомогти у вирішенні тих чи інших задач. Часто такими джерелами виявляються дані зібрані для вирішення інших задач, словникова інформація, моделі, які навчались на сторонніх даних. Найбільш наївним підходом до вирішення задач обробки природної мови у малоресурсному середовищі є перетворення середовища у багаторесурсне, що відбувається у два кроки: (1) створення або отримання нових нерозмічених наборів даних; (2) розмітка отриманих даних. Такий підхід має набір очевидних недоліків, а саме: знаходження нових наборів даних може бути важкою або непосильною задачею через обмеженість та специфічність прикладної області, у якій генерується мало даних, або через слабку присутність природної мови у мережі Інтернет; розмітка даних може бути дорогою так як експертів, які можуть якісно розмітити дані, обмаль, вони дорогі та працюють повільно, а більш дешеві і швидкі працівники зменшують якість результуючого набору даних.

Методи обробки природної мови у малоресурсному середовищі поділяються на дві категорії: створення додаткових розмічених даних або трансферне навчання [1]. Також є методи, які за своїми ознаками не підпадають під дані класи, тому їх було виділено в додаткову загальну категорію «Інші». Класифікація методів обробки природної мови у малоресурсному показана на рис. 1.3.

1.3.1 Генерація додаткової розмітки

Для боротьби з відсутністю розмітки для конкретних завдань обробки природної мови, було розроблено різноманітні підходи для пошуку альтернативних

Рис. 1.3 – Класифікація методів обробки природної мови у малоресурсному середовищі

форм розмічених даних як «замінників» істинних анотацій, які містять сигнал для навчання моделей машинного навчання. Зазвичай генерація додаткової розмітки робиться за допомогою використання певної форми експертних спостережень у поєднанні з автоматизацією. Такі методи групують [1] у три основні категорії: доповнення даних, яке використовує наявні приклади з набору даних для створення нових, часто за допомогою евристик; міжмовні проекції, коли розмічені дані у багаторесурсній мові намагаються перевикористати у малоресурсній; віддалений нагляд, ідея якого полягає у комп'ютеризованій розмітці нерозмічених даних.

1.3.1.1 Доповнення даних

Нові приклади для доповнення набору даних можна отримати на основі існуючих, модифікуючи їх за допомогою перетворень, які не впливають на результат розмітки. Доповнення даних — це ідея, що прийшла з області комп'ютерного зору [38], де такі трансформації зображень, як наприклад, обрізання, обертання або масштабування, створюють додаткові приклади для навчання статистичних і нейромережевих підходів та підвищують ефективність вирішення багатьох завдань. У

спільноті комп’ютерного зору такий підхід дуже популярний, адже дані трансформації зображення є інваріантними до класифікації вмісту зображення.

При обробці природної мови, однак, трансформації менш очевидні і вимагають глибокого знання мови, яка фігурує у задачі (особливо її синтаксичних конструкцій). Так як текстові дані мають ієрархічну структуру, доповнення даних при обробці природної мови може відбуватись на різних рівнях. Найнижчим рівнем, на якому можуть відбуватись заміни є символьний. На ньому популярним методом доповнення даних є випадкова заміна символів у словах для імітації помилкового клавіатурного вводу [39], особливо у словах, які часто пишуться неправильно. На рівні окремих токенів чи лексем було запропоновано наступні трансформації: заміна слів їх змістовим еквівалентами, такими як синоніми [15], сутності одного типу [40], [41] або слова, які мають однакову морфологію [42]. Подібні заміни можуть керуватись простими правилами, а можуть і більш складними мовними моделями, які враховують контекст та пропонують більш відповідні замінники [43], [44]. На ще вищому рівні, доповнення даних також можна виконати на цілих частинах речення. Операції, які не впливають на значення розмітки на цьому рівні, включають у себе перестановку частин складносурядних речень [16], [45], спрощення речень шляхом видалення його частин [16] та зміну відношень між об’єктами усередині речення [46]. Ще одним цікавим підходом, який став доступним для використання з розвитком машинного перекладу, є перефразування речень через зворотний переклад. Для цього цільові речення з існуючого набору даних перекладаються іншою мовою (зазвичай багаторесурсною, наприклад англійською), а потім перекладаються назад, що через неоднозначність природніх мов продукує нові текстові форми того ж змісту [47]. Особливістю даного підходу є те, що помилки у початковому тексті часто не впливають на якість перекладу і виправляються у процесі оберненого перекладу, додаючи важливі доповнені дані до існуючого набору. Звичайно, цей метод припускає наявність системи перекладу між мовами, що розглядаються. При класифікації документів замість такої системи можна використати мовну модель. Такі моделі

навчаються на основі окремих класів, тобто на підмножині даних, які відповідають конкретному класу. Потім дана модель використовується для генерації додаткових речень, які відповідають даному класу. Робота [48] досліджує ефективність використання попередньо навчених мовних моделей для створення нових прикладів для навчання. Цей підхід цікавий, так як поєднує ідею доповнення даних та трансферного начання. Незважаючи на те, що цей метод є надзвичайно потужним через використання сучасних попередньо навчених моделей на основі трансформерів, основним його недоліком є припущення, що ці моделі існують і були навчені на величезному наборі текстових даних. Це робить його непридатним для використання при роботі з будь-якою малоресурсною природною мовою – середовищі, де немає великої кількості даних для навчання моделей з такими обсягами параметрів.

Інший підхід до доповнення даних пропонують автори [49] - замість написання ручних правил модифікації текстів навчати модель, яка штучно модифікує наявні дані не змінюючи загального змісту відповідно до певної оцінки. Цей підхід часто застосовується на рівні векторних представлень слів, речень чи документів.

Незалежно від способу генерації додаткової розмітки, цей процес може додавати неправильні або неприродні приклади в набір даних, а отже повинен використовуватись обережно. Майже усі перелічені способи також мають набір припущень, які варто приймати до уваги при розробці практичних застосувань на основі аналізу природної мови у малоресурсних середовищах.

1.3.1.2 Міжмовні проекції

Методи міжмовної проекції полягають в тому, що спочатку навчається модель розв’язання поставленого завдання у багаторесурсному мовному середовищі. Після цього, нерозмічені дані в оригінальному завданні вирівнюються з їх еквівалентами високоресурсною мовою за допомогою паралельних корпусів. Паралельні корпуси – це набори текстових даних, у яких одні і ті ж одиниці мови представлені одразу кількома мовами. Далі за допомогою вищесказаного класифікатора виконується класифікація нерозмічених даних у багаторесурсій мові. Результати класифікації (у

багаторесурсному середовищі) можна потім спроектувати назад до оригінальної мови на основі того ж вирівнювання між лексемами в паралельних текстах, що використовувався на другому етапі підходу [50]. Іноді міжмовні проекції відносять до методів віддаленого нагляду, але специфічність роботи з паралельними корпусами та іншими мовами дозволяє виділити його у окрему вітку методів обробки природних мов у малоресурсному середовищі. Міжмовні проекції були застосовані в умовах низьких ресурсів для завдань, таких як визначення частин мови [51]. Джерелами паралельних текстів часто бувають літературні твори та книги перекладені великою кількістю мов, наприклад Біблія [52]. Замість використання паралельних корпусів наявні розмічені набори даних написані багаторесурсними мовами також можна машинно перекласти на малоресурсну мову, знову ж, при існуванні відповідних інструментів перекладу.

Застосування методу міжмовних проекцій - дуже вимоглива до наявності допоміжних даних процедура, адже потрібно мати не лише розмічені набори даних на багаторесурсній мові, а і інструменти та засоби проектування між мало- та багаторесурсними мовами. Остання вимога рідко коли справджується, оскільки якщо у мові знайшлось достатньо ресурсів для створення системи машинного перекладу, то є велика ймовірність, що в ній достатньо ресурсів і для вирішення поточної задачі іншими методами. Також обмеженням побудові якісних систем машинного перекладу чи застосуванню паралельних корпусів є те, що їх тематика часто не збігається з наявною проблемою. Наприклад, багато паралельних корпусів існує на тему політики чи релігії, тоді як діалогових корпусів - обмаль.

1.3.1.3 Віддалений нагляд

Іншим способом генерування додаткових розмічених даних є віддалений нагляд [53]. Ідея методу полягає у розмітці нерозмічених даних на основі простих правил або сутностей у базах знань. Цей підхід зазвичай використовується для задач пошуку іменованих сутностей, а також задач на синтаксичних і морфологічних рівнях.

Основним припущенням методу є існування нерозмічених даних, що не завжди можливо в малоресурсних середовищах.

На відміну від методів доповнення даних, методи віддаленого або слабкого нагляду зберігають текст документів з наявного набору даних незмінним. Вони лиш намагаються розмітити нерозмічені дані. Дані методи отримують відповідну розмітку за допомогою автоматичного або напівавтоматичного процесу із зовнішнього джерела інформації. Наприклад, для задачі пошуку іменованих сутностей список назв географічних об'єктів можна отримати зі словника чи спеціалізованого інтернет ресурсу. Далі при знаходженні повного або часткового збігу слів у нерозмічених даних з сутністю в словнику, дане місце можна розмітити відповідним класом. Віддалений нагляд - популярний підхід для завдань пов'язаних з вилучення інформації, таких як пошук іменованих сутностей та пошук зв’язків між ними, де зовнішню інформацію можна отримати з баз знань, довідників, словників та інших форм структурованих джерел знань [54], [55]. Для інших задач обробки природної мови методи автоматичної та напівавтоматичної анотації варіюються від простого пошуку схожих рядків до складних процесів машинного навчання, включаючи класифікатори, системи на основі правил, статистичні підходи. Часто для побудови правил та евристик, за допомогою яких розмічають дані, використовують регулярні вирази [56].

У той час як методи віддаленого нагляду популярні для завдань із вилучення інформації, пошуку іменованих сутностей та пошуку зв’язків між ними, вони менш поширені в інших областях обробки природних мов. Тим не менш, віддалений нагляд також успішно використовується для інших завдань, розширюючи наявні ресурси автоматично проанотованими текстами. Так наприклад, Лі та ін. [57] використовують словник частин мови для класифікації розмітки частин мови у нових текстах. Для класифікації сентименту частин речень автори [58] створюють простий класифікатор на основі підходу «мішок слів», наповнюючи його зі словника, а потім тренують глибоку нейронну мережу на основі слабкого нагляду. Венг та ін. [59] використовують

повторення контексту для того, щоб розмітити настрій на рівні документа, а потім створити нові екземпляри з окремих речень.

Одним із варіантів модифікації віддаленого нагляду є [60], де пропонується генерувати слабо розмічені дані для боротьби з дефіцитом доступних ресурсів. Це робиться за допомогою контекстуалізованих представлень для прикладів, коли правила не спрацьовують. Хоча цей підхід призводить до значного покращення точності при класифікації запитань і спаму, метод покладається на наявність додаткових точок даних, які можна позначити за допомогою слабкої розмітки.

Методи дистанційного спостереження значною мірою покладаються на допоміжні нерозмічені дані. У малоресурсних середовищах будь-які допоміжні дані можуть бути недоступні. Це підкреслює необхідність оцінки таких методів у справжніх малоресурсних умовах, а не симуляції відсутності ресурсів через обмежене використання додаткових джерел у багаторесурсних мовах. Хоча методи віддаленого нагляду дозволяють отримувати розмічені дані швидше та дешевше, ніж власноручне анотування кожного нерозміченого екземпляру у наборі даних, вони все ще вимагають взаємодії людини для створення таких методів автоматичного анотування або побудови правил розмітки. Цей час і зусилля також можна було б витратити на анотацію більшої кількості даних, що могло б більш значно покращити результати роботи стандартних моделей. На жаль, у описі сучасних методів віддаленого нагляду не надається інформація про те, скільки часу зайняли дизайн та імплементація даних методів, що ускладнює порівняння цих підходів з іншими підобластями методів генерації додаткової розмітки.

1.3.2 Трансферне навчання

Методи трансферного навчання у свою чергу менше сфокусовані на створенні нових прикладів у наявних наборах даних, а працюють на основі перевикористання попередньо створених моделей, які вже мають певні знання про особливості мови і можуть повторно їх враховувати при зміні завдань чи середовищ. Цей тип підходів дуже успішний, тому що після навчання певної моделі її ваги легко переносяться між

завданнями і можуть покращити роботу інших методів у інших задачах обробки природної мови. Проте, для того, щоб дані методи працювали, моделі, які переносяться, потрібно спочатку натренувати на інших даних або завданнях. Цей процес вимагає існування окремого набору даних, зазвичай великого та іноді розміченого, а також значних витрат часу та обчислювальних ресурсів для процесу навчання. Крім того, для успішного покращення існуючих методів вихідна проблема (та, на основі якої тренується модель для переносу) повинна бути схожою на оригінальну, тобто повинен бути перетин між прикладними областями та завданнями. Якщо мова чи прикладна область вирішуваної задачі достатньо специфічна, методи трансферного навчання можуть не спрацювати. Трансферне навчання зменшує потребу в додаткових розмічених даних шляхом передачі навчених векторних представлень і моделей. Багато сучасних робіт з трансферного навчання в області обробки природних мов зосереджується на використанні попередньо навчених мовних представлень, які навчаються на великій кількості нерозмічених даних, таких як BERT [61].

1.3.2.1 Векторні представлення

Векторні представлення є основним вхідним компонентом багатьох моделей на основі нейронних мереж для завдань обробки природної мови. Це різного роду числові представлення слів або речень, оскільки нейронні архітектури не можуть обробляти текстові рядки або символи як такі. Автори [6] показали, що навчання нейронних моделей для задачі моделювання мови на великомасштабному корпусі призводить до отримання високоякісних векторних представлень слів, які також можна повторно використовувати і для інших NLP завдань. Проблемою таких представлень є те, що вони працюють лише з лексемами, які існували під час навчання нейронної моделі, і при переносі ваг, якщо нова область використовує інші форми слів, їх застосувати не можна. Для вирішення цієї проблеми було запропоновано будувати представлення на основі підслів, таких як n-грам (підхід FastText [62]), або діаграмного кодування [63]. Таким чином лексеми, які відсутні у словнику векторних

представлень, розбиваються на кілька підслів, які разом представляють оригінальне слово та існують у словнику. Автори [64] показали, що векторні представлення, що використовують підслівну інформацію, корисні для завдань моделювання послідовностей у обробці природної мови у малоресурсному середовищі та працюють краще ніж векторні представлення окремих слів для задачі розпізнавання іменованих сутностей. Для вирішення подібної проблеми у моделях, які не працюють на підслівному рівні, Юнгмайєр та ін. [65] запропонували згладжування Word2Vec моделей. Це дозволило виправити їх упередження щодо рідкісних слів, і покращило їх роботу при вирішенні задач обробки природної мови у малоресурсному середовищі.

Окремою значною перевагою векторних представлень є їх доступність. Так, попередньо підготовлені векторні представлення були опубліковані для більш ніж 270 мов для різних методів їх побудови. Це дозволило обробляти тексти багатьма мовами, у тому числі малоресурсними, хоч для цього вони повинні бути представлені у інтернет середовищі, наприклад у Вікіпедії. Найбільш потужними сучасними методами векторних представлень є попереднє навчання великих мовних моделей для створення контекстно-залежних векторів слів шляхом передбачення наступного слова або речення у тексті. Для цього використовуються попередньо навчені моделі трансформерів [66], такі як BERT [61] або RoBERTa [67]. Ці методи особливо корисні для мов з малою кількість розмічених даних для конкретних завдань обробки природної мови, але великою кількістю нерозмічених даних [68].

Хоча попередньо навчені мовні моделі досягають значно кращих результатів порівняно зі стандартними векторними представленнями слів, мало які з цих методів підходять для реальних малоресурсних середовищ. Наприклад, тренування усіх моделей трансформерів вимагає не тільки величезну кількість нерозмічених даних, а і потужні обчислювальні ресурси. Так як поки що зберігається тенденція, що більш продуктивні трансформери мають більший розмір (кількість ваг усередині нейронної мережі) [69], проблема наявності обчислювальних ресурсів для їх навчання все більш актуальна. Дослідники, компанії та зацікавлені сторони у вирішенні проблем обробки

природної мови у малоресурсному середовищі часто не мають доступу до подібного роду систем, а отже і до сучасних моделей трансформерів.

Для класифікації документів прості підходи на основі мішка слів працюють краще ніж складніші моделі, такі як трансформери, коли для кожного класу є лише кілька десятків або менше навчальних прикладів [70]. Для конкретних завдань та прикладних областей було продемонстровано [71], що моделі, створені лише для них, краще використовують великі обсяги нерозмічених, на відміну від загальних векторних представлень, які будуються за допомогою трансформерів. Більше того, якість даних для середовищ із низьким рівнем ресурсів, навіть нерозмічених, може бути значно нижчою порівняно з якістю даних у середовищах із високим рівнем ресурсів. Алабі та ін. [72] виявили, що векторні представлення слів, навчені на більшій кількості нерозмічених даних малоресурсною мовою, не є конкурентоспроможними порівняно з тими, що були навчені на менших, але уважно відібраних джерелах даних.

1.3.2.2 Багатомовні моделі мов

Подібно до методів доповнення даних за допомогою міжмовних проекцій, основною ідеєю методів багатомовних моделей мов є те, що мови з низьким рівнем ресурсів також можуть отримати вигоду від розмічених даних, які доступні багаторесурсним мовам. Відмінністю між цими двома підкласами методів є те, що при доповненні даних відбувається явний переклад з багаторесурсної мови на малоресурсну, тоді як у багатомовних моделях мов корисна інформація написана багаторесурсною мовою вбудовується у модель, яка може працювати і у малоресурсному середовищі. Зазвичай такі методи вимагають навчання багатомовних мовних представлень шляхом поєднання одномовних представлень [73] або навчання однієї моделі для багатьох мов, наприклад, багатомовного BERT [61] або XLMRoBERTa [74]. Ці моделі початково навчаються з використанням нерозмічених корпусів різними мовами. Оскільки модель бачить багато прикладів з різних мов під час навчання, вона може застосовуватись у міжмовних контекстах, адже

представлення слів, що позначають однакові сутності у різних мовах будуть близько у векторному просторі.

Багатомовні моделі мов також можуть працювати у середовищі, де специфічних до завдання даних у малоресурсній мові взагалі немає. Тоді розмічені дані з багаторесурсної мови використовуються при тренуванні багатомовної моделі, а потім дана модель застосовується до специфічних даних. Прикладами такої поведінки є успішні застосування багатомовних моделей для розпізнавання іменованих сутностей [75], розуміння прочитаного [76], а також аналізу відношень між сутностями у документах [77]. Попри те, що такі моделі можуть допомогти у вирішенні завдань перелічених вище, Ху та ін. [78] показали, що все ще існує великий розрив у їх продуктивності між мало- та багаторесурсними середовищами. Для зменшення даного розриву у практичних ситуаціях, автори [17] запропонували додати мінімальну кількість даних з цільового середовища (у діапазоні від 10 до 100 розмічених речень) при навчанні таких моделей, що призводить до значного підвищення продуктивності для класифікації документів при обробці природної мови у малоресурсному середовищі.

Сам процес неявної передачі інформації між двома мовами у багатомовних моделях мов можна покращити, створивши загальний багатомовний векторний простір для кількох мов. Це особливо корисно при побудові векторних просторів слів [79], а також великих мовних моделей. Загальний векторний простір зазвичай створюється через операцію вирівнювання. У ній спочатку обчислюється відображення між двома різними векторними просторами для різних мов. Таке відображення будується таким чином, щоб після об’єднання двох векторних просторів, однакові сутності в обох підпросторах мали подібні вектори [80]. Це дозволяє використовувати вектори слів з різних мов всередині однієї моделі та працює краще ніж коли два простори розділені [81]. Успішним застосуванням таких моделей є [18], де автори побудували двомовні векторні представлення слів за допомогою невеликого набору паралельних речень між багаторесурсною англійською та трьома

малоресурсними африканськими мовами, а саме суахілі, тагальською та сомалійською, щоб покращити ефективність пошуку документів для африканських мов.

Хоча багатомовні моделі мов є величезним кроком до уніфікації методів обробки різних природних мов, багато з таких універсальних моделей не працюють однаково добре з різними мовами. Наприклад, mBERT охоплює 104 мови, а XLM-R - 100 мов, що становить третину всіх мов, статті якими наявні у Вікіпедії. При цьому Ву та ін. [82] показали, що малоресурсні мови недостатньо представлені в mBERT та XLM-R. Зокрема, африканські мови погано представлені в даних трансформерних моделях, хоч мільйони людей ними розмовляють. Окрема проблема даних підходів була продемонстрована у [17], а саме те, що більш віддалені за лінгвістичним походженням мовні сім'ї отримують менше переваг від трансферного навчання.

1.3.2.3 Адаптація домену моделі мови

Мова спеціалізованого домену може значно відрізнятися від тієї, що вважається стандартною або розмовною, тому багато прикладних областей також відносяться до середовищ з малими ресурсами. Наприклад, наукові статті можуть містити формули та технічні терміни, які не зустрічаються в новинах. Більшість сучасних мовних моделей, що застосовуються для трансферного навчання попередньо навчені на даних загального домену, наприклад текстах з новин або веб-сторінок, що призводить до неоптимальної поведінки при застосуванні даних моделей до іншого домену. Одним із рішень для вирішення даної проблеми є адаптація натренованої моделі до цільового домену шляхом додаткового налаштування мовної моделі. Гуруранган та ін. [83] показали, що продовження навчання моделі на нерозмічених даних у специфічному домені або задачі призводить до підвищення продуктивності побудованих моделей як у середовищах з багатьма, так і з малою кількістю ресурсів англійською мовою. Загалом, це також відображається в кількості мовних моделей, які були опубліковані з адаптаціями до окремих доменів [84], [85], зокрема для домену біомедичних статей – BioBERT [86], а також для домену наукових текстів – SciBERT [87]. При цьому

попри те, що модель BERT, що тренувалась на текстах без адаптації, добре працює в галузі матеріалознавства, адаптована до наукового домену SciBERT працює краще [88]. Шу та ін. [89] показали, що якщо при попередьому навчанні моделі використовувати дані всередині та поза доменом, а потім ще і адаптувати її до середовища з малими ресурсами, то результат буде кращим, ніж без використання адаптації. Цікавим розвитком адаптованих моделей є те, що в попередньо навчених мовних моделях існують специфічні до певних доменів кластери, які можна використовувати для відбору даних під час навчання у малоресурсних середовищах [90].

Потужні моделі можна отримати шляхом поєднання представлень текстів із загального домену з представленнями текстів з цільового малоресурсного домену. Кіела та ін. [91] продемонстрували, що представлення з різних доменів можна об’єднувати за допомогою мета-представлень, які будуються на основі зваженої суми представлень усіх діючих доменів. Ланге та ін. [92] додатково покращили цей підхід за допомогою вирівнювання представлень навчених у різних доменах, використовуючи змагальний дискримінатор, який навчається розрізняти векторні простори для створення векторів інваріантних до домену.

1.3.3 Інші

Так як спільноти обробки природних мов та машинного навчання дуже пов'язані, багато сучасних підходів є комбінаціями або розширенням методів із даних дисциплін. Обмін ідеями між ними результував у розробці двох основних груп підходів, які не є доповненням даних та не є трансферним навчанням, а лиш модифікують дані методи через змагальні дискримінатори та метанавчання.

1.3.3.1 Змагальний дискримінатор

Відмінності в характеристиках між середовищами, у яких навчаються моделі для трансферного навчання, та цільовими малоресурсними середовищами можуть викликати проблеми при трансферному навчанні, особливо при використанні нейронних підходів, де важко відслідкувати, яку саме інформацію вивчила модель.

Змагальні дискримінатори [93] є формою регуляризації, яка може допомогти моделі не перенавчатись на ознаках, характерних для певного джерела даних. Даний підхід здатен покращити роботу більшості нейромережевих моделей, що було показано у роботах Гуі [94], Ліу [95], Касай [96], Грісхабера [97] та Чжоу [98], які будували незалежні від прикладної області векторні представлення за допомогою змагального навчання. Кім [99], Чен [100] та Ланге [101] досліджували подібні підходи для незалежних від мови векторних представлень при побудові міжмовних моделей.

1.3.3.2 Метанавчання

Іншим прикладом методів, що прийшли з області машинного навчання, є метанавчання [102]. Метанавчання у контексті малоресурсних середовищ базується на багатозадачному навчанні. Методи метанавчання будують модель, яка здатна вирішувати яким чином використовувати допоміжні ресурси для вирішення задачі з малим рівнем ресурсів на основі допоміжних задач у середовищах з великою кількістю ресурсів. У обробці природної мови такі підходи були запропоновані для таких завдань, як аналіз настроїв [103], класифікація намірів [104], розуміння природної мови [105], класифікація документів [106] та генерація діалогів [107]. Іншу цікаву ідею у даній області було запропоновано Рахімі та ін. [108] – вони запропонували створити ансамбль моделей обробки різних природних мов, а не задач, який потім залежно від цільової малоресурсної мови зважував попередньо натреновані моделі.

1.4 Висновки до розділу

У рамках першого розділу даної дисертації було описано задачі обробки природної мови, особливості їх вирішення в умовах малоресурсного середовища, а також проаналізовано існуючі методи обробки природної мови у малоресурсному середовищі. Попри суттєві досягнення у розвитку даної області, кожне завдання в залежності від середовища має свої характерні ознаки та ресурси, методи використання яких можуть призводити до покращення роботи типових NLP алгоритмів. Майже кожен вищезазначений метод має набір недоліків. Зазвичай ними

є припущення про існування певного додаткового ресурсу інформації, який можна перевикористати для вирішення цільової задачі. Так методи віддаленого нагляду очікують наявність нерозмічених даних у тій же прикладній області, що і цільове завдання; методи трансферного навчання та метанавчання припускають існування моделей, які мають певне представлення про середовище, у якому знаходяться; методи міжмовних проекцій та багатомовних моделей мов спираються на те, що подібна до цільової задачі проблема вже вирішена у багаторесурсній мові та є інструменти, які дозволяють робити переклад чи вирівнювання між даними мовами; методи доповнення даних вимагають експертних знань про малоресурсну мову або прикладну область для побудови правил генерації додаткових даних.

Також, залежно від поточної ситуації, причин малоресурсності середовища та наявності доступу до сторонніх ресурсів повинні обиратись різні методи обробки природної мови. Наприклад, якщо задача обмежена специфічністю прикладної області та дорожнечею розмітки додаткових даних, але знаходиться у межах багаторесурсної природної мови, то простір доступних методів покращення ефективності її вирішення доволі різноманітний. З іншого боку, якщо задача перебуває у контексті малоресурсної мови для якої інструментів, моделей чи додаткових лінгвістичних даних немає, покращення роботи доступних підходів значно ускладнюється. Тому розробка та пошук нових підходів для вирішення задач обробки природної мови у малоресурсних середовищах – важка, але актуальна задача на перетині штучного інтелекту та комп’ютерної лінгвістики.

Враховуючи переваги, недоліки та припущення існуючих методів обробки природної мови у малоресурсному середовищі, метою дослідження проведеного в рамках цієї дисертаційної роботи є удосконалення методів обробки природної мови у малоресурсному середовищі. Таке удосконалення можна отримати через використання додаткової словникової інформації, яку в багатьох малоресурсних середовищах отримати легше ніж ту, що пропонується в істотній частині існуючих методів, а також через модифікацію способів представлення та поєднання даної

словникової інформації з поточними моделями обробки природної мови, таким чином щоб її можна було поширювати та застосовувати у різних прикладних областях та задачах.

РОЗДІЛ 2. ДОПОВНЕННЯ ПРЕДСТАВЛЕНЬ ДОКУМЕНТІВ СЛОВНИКОВИМИ ПРЕДСТАВЛЕННЯМИ

У даному розділі дисертації пропонується підхід до обробки природної мови у малоресурсному середовищі для вирішення задачі класифікації документів. Мета розробки даного підходу – покращення існуючих методів обробки природної мови у малоресурсному середовищі через використання словникової інформації у векторному вигляді.

Типовий метод класифікації документів на основі векторних представлень при обробці природної мови зображений на рис. 2.1. Кроки даного методу та їх мотивація:

1. Передобробка документів – виконується для того, щоб прибрати шум в даних, який заважатиме побудові векторних представлень слів та класифікації, наприклад видалення стоп-слів та приведення до консистентного регістру;

2. Побудова векторних представлень слів – виконується для того, щоб закодувати окремі лінгвістичні одиниці (слова) у щільні вектори їх ознак;

3. Побудова векторних представлень документів на основі векторних представлень слів – виконується для того, щоб закодувати документи у щільні вектори їх ознак;

4. Побудова класифікатора на основі векторних представлень документів після

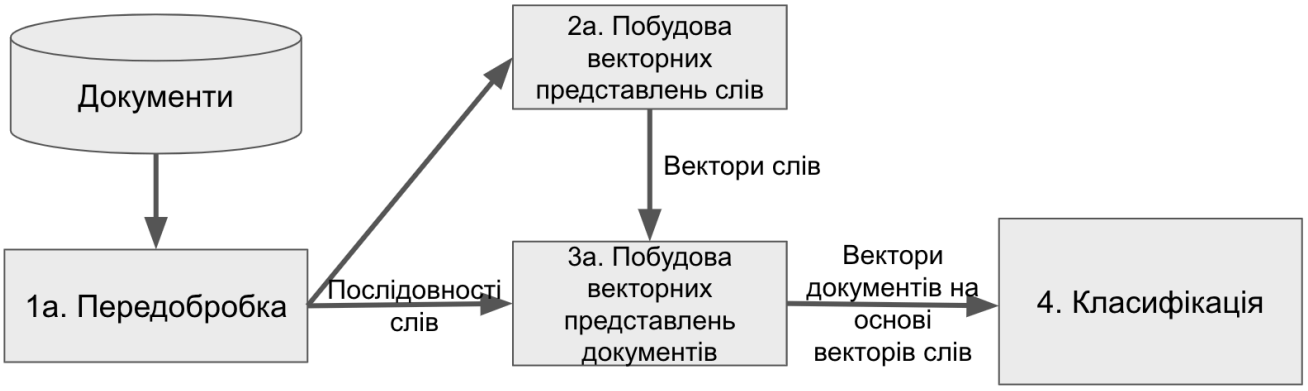

Рис. 2.1 – Діаграма типового методу класифікації документів на основі векторних представлень

злиття – сам процес класифікації документа моделлю-класифікатором на основі їх векторних представлень.

Ідея методу полягає в тому, щоб явно доповнити векторні представлення документів, які потім використовуються класифікатором, інформацією про синонімію слів природної мови взяту зі словника синонімів. Так як сам словник – дискретна структура даних, для взаємодії векторних представлень документів з його інформацією, сам словник потрібно закодувати – отримати його векторне представлення. Найбільш очевидна структура даних для зберігання словника – ненапрямлений граф, де вузли відображають слова, а ребра – синонімічний зв'язок. Так як ребра у графі одного виду та не мають напряму, закодувати інформацію, що міститься в ньому можна за допомогою методів побудови векторних представлень вузлів. Після отримання таких векторів, їх потрібно з'єднати з векторними представленнями документів за допомогою методів злиття векторних представлень. Результуючі доповнені представлення документів можна передавати до класифікатора, який навчатиметься на видозмінених ознаках та набуде здатності класифікувати нові приклади з малоресурсного середовища.

Отже, у даній дисертаційній роботі пропонується метод класифікації документів на основі векторних представлень словників при обробці природної мови у малоресурсному середовищі, який приймає на вхід документи для класифікації і словник, та полягає у наступних кроках:

1а. Передобробка документів;

2а. Побудова векторних представлень слів;

3а. Побудова векторних представлень документів на основі векторних представлень слів;

1б. Передобробка словника – виконується для того, щоб привести слова у словнику до вигляду максимально близького до слів у документах, це спростить подальший пошук відповідностей між ними;

2б. Побудова векторних представлень вузлів графа – виконується для того, щоб закодувати окремі слова у словнику синонімів у щільні вектори їх ознак;

3б. Побудова векторних представлень документів на основі векторних представлень вузлів – виконується для того, щоб закодувати документи альтернативним векторним представленням у щільні вектори їх ознак;

4. Злиття векторних представлень документів з кроків 3а та 3б – виконується для того, щоб об'єднати інформацію закодовану у двох векторних представленнях побудованих раніше;

5. Побудова класифікатора на основі векторних представлень документів після злиття.

При цьому кроки *1-3* для віток *а* та *б* можуть проводитись паралельно, адже не мають спільних залежностей. Схематично перелічені кроки з очікуваними входами та виходами зображені на рис. 2.2

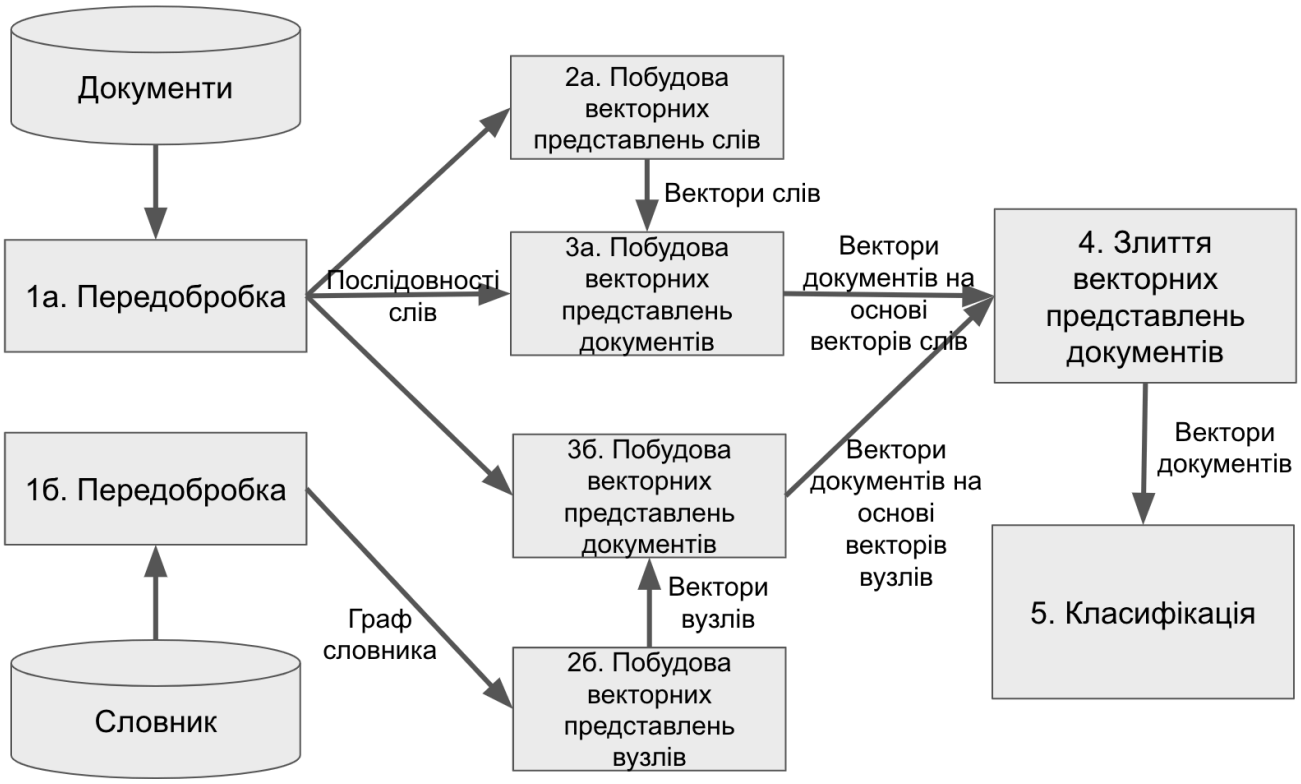

Рис. 2.2 – Діаграма запропонованого методу

Запропонований метод удосконалює метод доповнення даних на основі словникової інформації, який явно замінює слова синонімами на основі правил, роблячи його (1) легким для перенесення на інші застосування через векторні простори; (2) таким, що не створює штучні, з великою ймовірністю зашумлені приклади для навчання. З іншого боку, метод, що пропонується, розвиває ідею методів трансферного навчання, адже будує векторне представлення графа синонімів окремої природної мови, яка перевикористовується при злитті з векторним представленням документів для класифікації.

Наступні підрозділи детально описують варіації окремих компонентів методу, які можна використати при його імплементації.

Для вирішення задачі класифікації документів потрібно побудувати класифікаторну модель, яка приймає документ та віддає його клас. Так як текстові документи – це символьна інформація, яка самостійно не несе ніякого змісту для комп'ютерних систем, їх потрібно закодувати. Кодувати документи можна багатьма різними способами, але мета у них одна – перевести документ у вектор ознак, на основі яких класифікатор зможе оптимальним чином знайти клас, до якого даний документ відноситься. Таке кодування також можна розглядати як проекцію документів у певний багатовимірний векторний простір ознак, у якому класифікатор повинен навчитись розділяти області, що відповідають цільовим класам.

Одним з найбільш наївних способів побудувати ознаки документа є інженерія текстових ознак документа. До таких відносяться: довжина документа, кількість слів, пробілів, цифр, розділових знаків, абзаців, речень, статистики розподілів довжин слів у документі тощо. Основною проблемою даного методу є те, що він не враховує зміст документа, які саме слова у ньому вживаються, їх контекст. Для нього два документа з різним змістом, але однаковими статистиками виглядатимуть ідентично і отримають один і той же клас, хоч їх зміст може бути абсолютно протилежним. Даний спосіб представлення документів має свої застосування у простих текстових задачах, але вживається дуже рідко.

Натомість, для того, щоб метод міг враховувати зміст документів, потрібно щоб він розумів концепцію його складових – слів. Відповідно сучасні методи обробки природних мов для класифікації документів кодують не лише документи, а і слова, які в них зустрічаються. Це може відбуватись явно у таких підходах як мішок слів чи TF-IDF, а також неявно у нейромережевих підходах таких як рекурентні нейронні мережі чи трансформери, де представлення слів існують на нижніх рівнях мережі.

2.1 Представлення слів

У обробці природної мови представлення слів — це термін, який використовується для різних числових форм представлення слів при аналізі текстів. Дані представлення можуть бути векторами як цілих чисел, так і чисел з рухомою комою. Вектори з дійсними значеннями, часто будуються таким чином, щоб слова, які знаходяться ближче у векторному просторі були схожими за своїм словесним значенням [3]. Векторне представлення слів – це перетворення:

$$f: V_i \rightarrow H_i \in R^d \ \forall i \in [n], \text{ де}$$

V – словник, який складається з n слів;

d – розмірність векторного представлення H.

Представлення слів і фраз, якщо використовуються в якості основних ознак, підвищують ефективність вирішення задач обробки природної мови, таких як синтаксичний аналіз [109] та аналіз настроїв [110].

2.1.1 Унітарне кодування

Унітарне кодування – метод представлення слів, коли кожне слово, що є частиною текстових даних, кодується у вигляді бінарних векторів, що складаються лише з 1 і 0. Кожне слово записується або кодується як один унітарний код, причому кожен такий вектор є унікальним. Це дозволяє однозначно ідентифікувати слово за його унітарним кодом і навпаки [111].

Для створення унітарного коду, спочатку будується словник лексем, які зустрічаються у природній мові, чи прикладній області, що моделюється. Такий словник можна розглядати як відображення символьного представлення слова на

цілочисельний індекс. Такий словник є окремим методом кодування слів під назвою категорійне кодування, адже перетворює слово у чисельне представлення – індекс. У такого представлення є суттєвий недолік – моделі на його основі можуть сприймати більші числа як більш важливі, ніж менші, а отже в залежності від порядку слів у словнику результат класифікації може змінюватись. У контексті структури даних для словника зручно використовувати хеш-таблицю, де ключем виступає слово, а значенням – індекс. Для оберненої трансформації (отримання слова за індексом) достатньо мати список слів зі словника. Унітарне кодування вирішує цю проблему через створення окремих векторів для кожного слова.

Після створення словника, унітарним кодуванням слова буде вектор, всі значення якого рівні 0, окрім значення за індексом даного слова у словнику. Наприклад, нехай все, що ми знаємо про певне середовище – це речення «бути чи не бути». Тоді побудований словник буде складатись з 3 слів: {«бути»: 0, «чи»: 1, «не»: 2}. Вектори, що відповідають словам зі словника прийматимуть значення {«бути»: [1, 0, 0], «чи»: [0, 1, 0], «не»: [0, 0, 1]}. Подібні вектори також зручно передавати у матричній формі, наприклад:

бути:	1	0	0
чи:	0	1	0
не:	0	0	1

Існує дві основні проблеми з унітарним кодуванням для представлення слів. Перша проблема — це прокляття розмірності, яке виникає при роботі з просторами великих розмірностей [112]. Так як розмірність векторів слів при унітарному кодуванні рівна розміру словника, а сучасні словники можуть налічувати десятки чи сотні тисяч слів (навіть при використанні лише найпопулярніших слів), то для роботи з ними вимагається велика кількість пам’яті, а моделям машинного навчання, які сприймають ці вектори як ознаки, потрібно уміти не перенавчатись на окремих позиціях у векторах. Частково дана проблема вирішується за допомогою методів роботи з розрідженими матрицями, адже унітарне кодування майже повністю складається з нулів, але у контексті обробки природних мов та машинного навчання

таких підходів зовсім обмаль. Друга проблема полягає в тому, що з даних векторів важко витягнути значення. Кожне слово має унікальний ізольований унітарний код, і кожен такий код містить лише одну одиницю і багато нулів. Отриманий набір векторів не дозволяє зрозуміти відношення між словами. Усі вектори знаходяться на однаковій відстані один від одного, а отже за допомогою унітарного кодування не можливо зрозуміти схожість між словами, ієрархію чи побудувати кластери.

2.1.2 Word2Vec

Word2Vec [113] — це нейромережева модель для навчання векторних представлень слів із необробленого тексту, яка має високу ефективність у відношенні використання обчислювальних ресурсів. Основна ідея методу – побудувати модель, яка добре прогнозує поточні слова у реченні на основі слів, які зустрічаються поряд. Даний метод існує в двох варіантах: Неперервний Мішок Слів (англ. Continuous Bag-of-Words, CBOW) і Пропуск-Грами (англ. Skip-Gram). Алгоритмічно ці варіанти подібні, за винятком того, що CBOW намагається спрогнозувати цільове слово (наприклад у реченні з минулої секції – слово «чи») на основі слів контексту (слів «бути», «не», «бути»), в той час як skip-gram навпаки намагається спрогнозувати слова контексту («бути», «не», «бути») на основі цільового слова. Умовне зображення цих двох архітектур зображено на рис. 2.3 [114].

Для роботи методу потрібно вибрати певні гіперпараметри, одним з найважливіших з яких є розмір контекстного вікна. Він визначає скільки слів зліва чи справа від цільового слова у документі мережа повинна враховувати як ознаки для прогнозування самого ж цільового слова. Нейромережева архітектура складається з лише одного прихованого шару нейронів, а отже з двох наборів вагів W_{in} – вектори цільових слів, та W_{out} – вектори слів контексту. Розмірності цих матриць відповідно VxN та NxV, де V – розмір словника, а N – обрана розмірність векторних представлень. Дані ваги поширюються між різними словами контексту, тобто з розміром вікна контексту – 3 при CBOW архітектурі все ще буде одна, а не три, W_{in} матриця ваг. Функція активації на нейронах прихованого шару не використовується, тобто вона є

тотожним відображенням, але вихідні нейрони використовують softmax. Входами до нейронної мережі є пари слів закодованих унітарним кодуванням, які при матричному множенні з вищезазначеними вагами дають певний рядок або колонку. Таким чином в практичних застосуваннях унітарне кодування не використовують, а одразу роблять індексацію в ваги нейронної мережі.

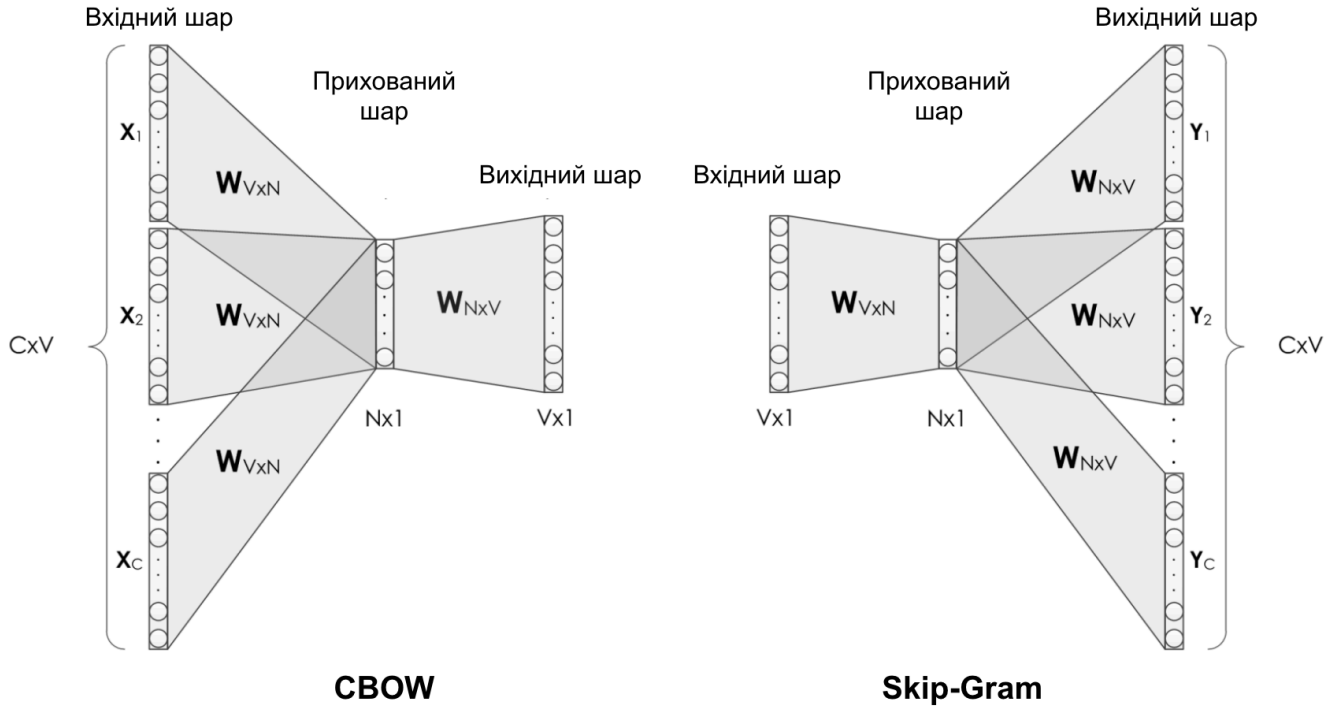

Рис. 2.3 – Архітектури Word2Vec

Наступною важливою опцією при побудові Word2Vec представлень є функція втрат та модифікація методу навчання. Для Skip-Gram архітектури функцією втрат виступає:

$$L(V_{t\pm C}|v_t; W_{in}, W_{out}) = \sum_{V_{t\pm C}} -\log\left(v_{t+c}^T \hat{v}_{t+c}\right), \text{ де}$$

$V_{t\pm C}$ – слова контексту;

v_{t+c} – унітарне кодування слова контексту на позиції $t+c$;

$\hat{v}_{t+c}$ – прогноз слова контексту за цільовим словом на позиції t.

Для CBOW, загальна ідея функції втрат залишається тою ж, але контекст та цільове слово міняються місцями:

$$L(v_t|V_{t\pm C}; W_{in}, W_{out}) = -\log(v_t^T \hat{v}_t), \text{ де}$$

v_t – унітарне кодування цільового слова на позиції *t*;

$\hat{v}_t$ – прогноз цільового слова за контекстом на позиції *t*.

Для оптимізації даних функцій втрат підходять ітераційні методи оптимізації, наприклад градієнтний спуск та його модифікації. На жаль, у такого визначення функції втрат є один важливий недолік – обчислення функції softmax при великому словнику доволі повільне, а отже навчання таким способом займає дуже великий час та не є практичним. Для обходу цього обмеження було застосовано метод негативного вибору (англ. negative sampling) [115], суть якого полягає в тому, що замість оптимізації оригінальної функції втрат та вирішення задачі прогнозу цільового слова за контекстом, можна навчитись відрізняти слова, які знаходяться в одному контексті від слів, яких в ньому немає (негативні приклади). Таким чином задача перетворюється у бінарну класифікацію без softmax і мережа навчається значно швидше, хоч і стає більш схильною до шуму в даних. Функція втрат у такому випадку перетворюється на:

$$L(v_t|V_{t\pm C}, V_K; W_{in}, W_{out}) = \sum_{V_{t\pm C}} \sigma(v_t^T v_{t+c}) + \sum_{i=1}^{K} \log\ \sigma(-v_t^T v_i), \text{ де}$$

v_t – векторне представлення цільового слова на позиції *t*;

v_{t+c} – векторне представлення контекстного слова на позиції *t*+*c*;

v_i – векторне представлення негативного прикладу *i* з набору *K* негативних прикладів V_K;

σ – логістична функція.

2.1.3 FastText

Word2Vec при навчанні представлень слів ігнорує їх морфологію, призначаючи кожному слову окремий вектор. Це є суттєвим недоліком, особливо для мов із великим словником і великою кількістю рідкісних слів. Для вирішення даної проблеми автори FastText [62] запропонували новий підхід, який базується на пропуск-грам моделі, де кожне слово розглядається як мішок символьних n-грамів.

Тоді кожен символьний n-грам має своє векторне представлення, а векторні представлення слів будуються як сума представлень символьних n-грам у слові. Навчання FastText векторів відбувається за допомогою негативного вибору, що зводиться до бінарної класифікації чи зустрічаються ці два слова у контексті.

Модель FastText генерує векторне представлення не лише слова, а і його символьних n-грам. При цьому обирається *n* – гіперпараметр, який обмежує довжину n-грами. Наприклад, при *n=3* слово «червоний» розкладеться на 8 n-грам: «_че», «чер», «ерв», «рво», «вон», «они», «ний», «ий_», при цьому такі символьні послідовності на початку та вкінці слова враховують розділові знаки, якими відокремлюються від інших слів. Також до векторних представлень додається саме слово «червоний». Таким чином, якщо при застосуванні методу приходить слово, яке є в словнику – береться його представлення, але якщо приходить слово, якого немає у словнику – представлення будується як сума представлень його n-грам, які в словнику є, тобто векторне представлення у даному вигляді обраховується так:

$$s(w) = \sum_{g \in G_w} z_g, \text{ де}$$

w – нове слово;

z_g – векторне представлення n-грама *g* з набору n-грам для слова *w* - G_w.

Довжину n-грам обмежують не лише знизу, а і зверху, відтак автори методу рекомендують будувати всі n-грами довжинами між 3 та 6.

2.2 Представлення документів на основі представлень слів

Після отримання векторних представлень слів у документі, потрібно обрати процедуру їх агрегації у векторні представлення самого документа. Для цього використовується великий набір підходів, від найпростіших – суми векторів, до складних багатошарових нейронних мереж. У даному підрозділі описуються частина таких підходів, які можуть працювати у малоресурсному середовищі, а отже не вимагають великої кількості даних для навчання. Більше того, описані методи легкі у використанні та часто використовуються у практичних застосуваннях як базові моделі, які можна розширяти з ростом даних, ресурсів чи проєкту.

2.2.1 Мішок слів

Мішок слів (англ. Bag-of-words, BOW) [116] – статистичний підхід до побудови векторних представлень документів. У ньому документ представлений кількостями окремих слів, які до нього входять. Для цього потрібно спочатку побудувати словник слів, що зустрічаються у текстах, а потім порахувати входження даних слів у документі. Таким чином, документ з прикладу з минулих секцій «бути чи не бути» може бути закодований словником {«бути»: 0, «чи»: 1, «не»: 2} як {0: 2, 1: 1, 2: 1}, де ключ – індекс слова у словнику, а значення – кількість входжень даного слова у документ.

Підхід мішка слів тісно пов'язаний з унітарним кодуванням слів. Так як у кожному документі зустрічаються різні слова, і при цьому їх кількість може змінюватись, а розмір векторного представлення документа повинен бути стабільним, справжні вектори виглядають як сума унітарних кодів слів, які зустрічаються в документі. Наприклад, вищезазначений приклад матиме наступне векторне представлення документа:

$$\text{бути}\,[1,0,0] + \text{чи}\,[0,1,0] + \text{не}\,[0,0,1] + \text{бути}[1,0,0] = [2,1,1]$$

Представлення на основі підходу мішка слів також можуть бути бінарними. У цьому випадку кількість повторень одного і того ж слова у документів не рахується, а вектор позначає лише наявність даного слова у документі. Так, для прикладу вище бінарне представлення мішка слів виглядатиме [1, 1, 1]. Це робиться у середовищах та прикладних застосуваннях де кількість повторів слів не допомагає у класифікації документа, або модель перенавчається на їх значеннях. Так, для популярних слів небінарні векторні представлення можуть набувати значно вищих значень, ніж інші слова, що може негативно відобразитись на ітеративному навчанні деяких моделей.

У даного підходу є ряд недоліків. Першим з них є те, що він не враховує, що документ є послідовністю, а не просто набором слів. Таким чином документи «спочатку - ліворуч, а потім – праворуч» та «спочатку - праворуч, а потім – ліворуч» будуть мати абсолютно ідентичні представлення. Другим недоліком є те, що занадто

часті слова мають вищі значення у векторі представлення, що не корелює з кількістю інформації, яку вони додають. Так, наприклад, слово «і», яке вживається майже у всіх документах не несе великого змісту, тоді як слово, яке вживається лише в декількох документах може бути вирішальним для задачі класифікації документів.

2.2.2 TF-IDF

Для вирішення другої з вищезазначених проблем підходу мішка слів було запропоновано зважувати чисельні представлення слів відповідно до їх частот у документі, а також у середовищі. Такий підхід має назву TF-IDF [117] (англ. Term Frequency – Inverse Document Frequency, частота слова – обернена частота документа) та широко застосовується у прикладних сферах обробки природної мови. Для обчислення TF-IDF векторів для документа припускають, що значення у векторі, що відповідає значимості певного слова, прямо пропорційне кількості вживань цього слова у документі, і обернено пропорційне частоті вживання даного слова у інших документах у наборі даних.

TF-IDF є добутком двох елементів: TF та IDF. У свою чергу TF обчислюється за формулою:

$$TF(i,d) = \frac{n_i}{\sum_k n_k}, \text{ де}$$

n_i – частота слова i у документі d;

k – кількість унікальних слів у документі d.

IDF обчислюється як:

$$IDF(i) = \log\left(\frac{N_d}{N_{di}}\right), \text{ де}$$

N_d – загальна кількість документів у наборі даних;

N_{di} – кількість документів, що містять слово i.

Дане визначення TF-IDF найбільш поширене, але є велика кількість варіацій цих формул. Наприклад, унарний IDF повертає завжди 1, що зводить таке представлення до підходу мішка слів. Більше того, якщо при цьому TF робити бінарним, то результуючим підходом буде бінарний мішок слів. Саме тому BOW

можна розглядати як частковий випадок TF-IDF. Прикладами інших модифікацій TF-IDF є використання абсолютних кількостей замість частот у формулі, використання згладжування у знаменнику IDF. Зі сторони теорії інформації TF-IDF можна розглядати як кількість біт, яка потрібна для кодування пари (слово, документ) [118].

2.2.3 Злиття щільних представлень слів

Успіх використання нейронних мереж для обчислення векторних представлень слів призвів до створення методів генерації семантичних представлень довших фрагментів тексту, таких як речення та документи. Більшість досліджень у даній області намагалися обчислити векторні представлення, які кодують семантику послідовностей слів (фраз, речень і абзаців), за допомогою різноманітних методів, починаючи від простої аддитивної композиції векторів слів [115] до складних архітектур, таких як згорткові нейронні мережі [119] та рекурентні нейронні мережі [120], [110]. При цьому Вітінг та інші [121] показали, що складні методи працюють гірше, особливо в малоресурсних середовищах при трансферному навчанні, ніж простіші методи, які передбачають прості операції над векторами слів і базову лінійну регресію. Тому у даній роботі будуть розглядатись саме такі методи, ті що поєднують готові вектори слів без додаткового тренування, так як малоресурсне середовище не дає змоги це робити.

Перший метод, який можна застосовувати у малоресурсних середовищах для побудови векторів документів на основі векторів слів – це їх усереднення. Загальне формулювання даного методу це:

$$W_{jr} = \sum_i^k \frac{v_i S_{ir}}{k}, \text{ де}$$

W_{jr} – значення r-виміру у векторному представленні j-го документу;

k – кількість слів у документі j;

S_{ir} – значення r-виміру у векторному представленні слова i, що зустрілось у документі j;

v_i – вага слова i.

В залежності від значення, які надаються вагам слів *v*, метод усереднення векторних представлень буває незваженим (*v* рівний 1), та зваженим (*v* не рівний 1). Вага слів може задаватись відповідно до прикладної області і задачі на основі правил, або статистичних ознак слова. Так частота слів (TF) та частота слова – обернена частота документа (TF-IDF) можуть використовуватись для зважування векторів слів при побудові векторів документів через зважене усереднення, таким чином покращуючи результати окремо побудованих моделей на основі TF-IDF та незважених векторів слів [122].

Розширенням методу зваженого усереднення векторів слів є метод згладженої оберненої частоти (англ. smooth inverse frequency, SIF) [123]. Він працює на основі обчислення зваженого середнього векторів слів у документі з видаленням проекції середніх векторів на їх перший власний вектор («видалення загального компонента»). Вага слова у даному методі має форму:

$$v_i = \frac{\alpha}{(\alpha + p(i))}, \text{ де}$$

α – параметр згладжування;

p(i) – оцінка частоти слова *i*.

Цей метод досягає значно кращих результатів, ніж незважене усереднення, для різноманітних завдань обробки природних мов, і для більшості з цих завдань навіть перевершує деякі складні моделі, такі як рекурентні нейронні мережі. Метод також добре підходить при трансферному навчанні, адже вектори слів, навчені на текстах з різних прикладних областей, можуть перевикористовуватись у інших прикладних областях з вагами слів пристосованими під них. Більше того навіть використання ваг з іншої прикладної області часто працює достатньо надійно і не шкодить результатам, а параметр методу α має широкий діапазон значень з якими здатен досягти наближених до найкращих результатів.

2.3 Графові представлення

Для того, щоб покращити роботу методів представлення та класифікації документів при обробці природної мови у малоресурсному середовищі пропонується використати додаткову інформацію зі словника синонімів для такої мови. Для того, щоб поєднати його зміст зі змістом документів, потрібно отримати ознаки тих слів, які зустрічаються у словнику. Так як словник – це граф, де вузли містять слова, а ребра позначають синонімію між двома словами, ознаки можна отримати за допомогою методів векторного представлення графів. Вони дозволяють створити додатковий рівень абстракції для безлічі завдань, пов'язаних з графами, оскільки векторні представлення вузлів і ребер можна переносити між завданнями, замість того, щоб будувати окреме специфічне рішення для конкретного завдання. Загалом, граф як структура з різними видами елементів може мати набір різних векторних представлень: представлення вузлів, представлення ребер та представлення графа в цілому. У контексті даної дисертації цікавими з них є лише представлення вузлів, які можна перенести за допомогою трансферного навчання на різні задачі та прикладні області.

Методи побудови векторних представлень графів за головною їх ідеєю класифікують у декілька груп: методи на основі факторизації графа, методи на основі випадкових блукань та методи на основі глибокого навчання. Деякі з існуючих методів, однак, не можна віднести лише до однієї групи, тому їх виділяють у окрему категорію – інші [124]. Дана класифікація зображена на рис. 2.4.

Основні поняття, які фігурують у подальшому описі методів побудови векторного представлення графів: граф – G, множина вузлів графа G – V, множина ребер графа G – E, розмірність векторного представлення – d, векторне представлення графа – $Y \in \mathbb{R}^{|V| \times d}$, векторне представлення вузла i – $Y_i \in \mathbb{R}^{1 \times d}$, векторне представлення вхідних вузлів – Y_s, векторне представлення вихідних вузлів – Y_t, матриця суміжності графа G – $W \in \mathbb{R}^{|V| \times |V|}$, діагональна матриця степенів кожного

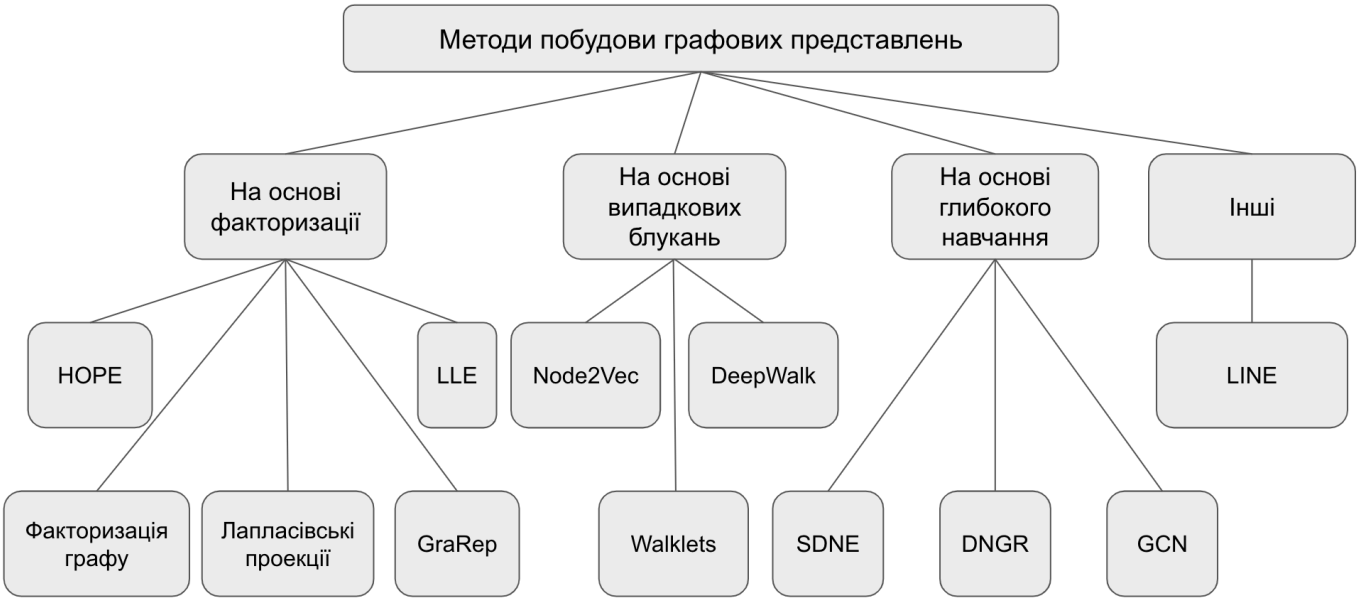

Рис. 2.4 – Класифікація методів побудови графових представлень

вузла – $D \in \mathbb{R}^{|V|\times|V|}$, Лапласіан графа – $L=D-W$, $L \in \mathbb{R}^{|V|\times|V|}$, матриця подібності графа $G - S \in \mathbb{R}^{|V|\times|V|}$.

Схематично, модель словника синонімів зображена на рис. 2.5. Очікується, що вектори слів у одному кластері графа матимуть більш близькі за косинусною відстанню представлення, на відміну від слів у інших кластерах. При цьому якщо кластери не пов'язані (не мають спільних слів, наприклад, на схемі слово - «Гарний»), то ця відстань буде значно вищою.

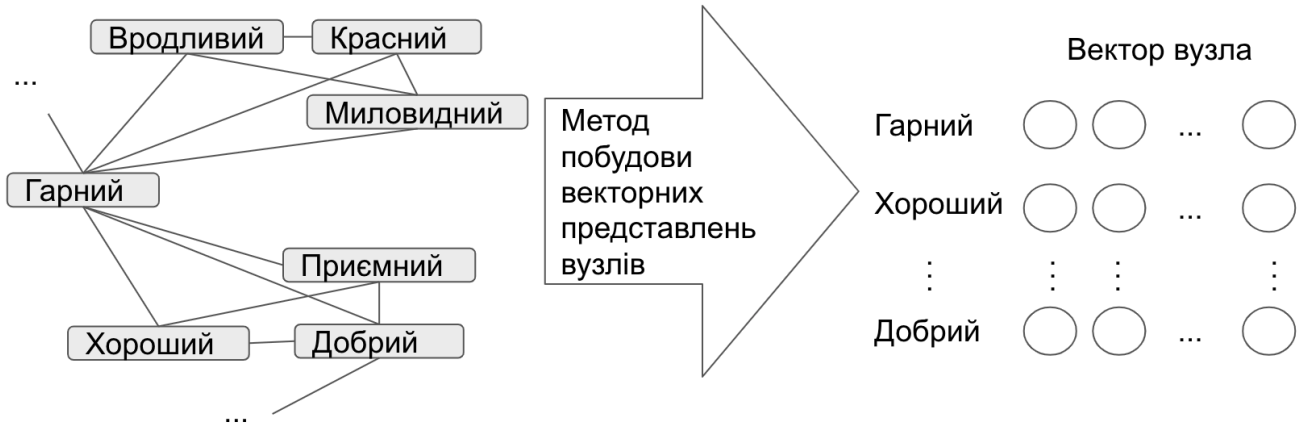

Рис. 2.5 – Схема моделі словника синонімів

Графові представлення на основі факторизації

Алгоритми на основі факторизації представляють зв'язки між вузлами у вигляді матриці, а потім факторизують цю матрицю для отримання векторних представлень. Один і той же граф можна представити різними матрицями, але зазвичай для алгоритмів на основі факторизації використовують матрицю суміжності вузлів, матрицю Лапласа, матрицю ймовірностей переходу з вузла у вузол та матрицю подібності Каца. Відповідно до обраного матричного представлення та його властивостей відрізняються і підходи до факторизації графу. Якщо отримана матриця додатно визначена (наприклад матриця Лапласа) можна використовувати власний розклад матриці. Для неструктурованих матриць можна використовувати ітеративні методи, такі як метод градієнтного спуску, щоб отримати графові представлення вузлів за лінійний час.

2.3.1.1 Факторизація графу

Факторізація графу була першим методом отримання векторного представлення графу, який працював за час пропорційний кількості ребер [125]. Щоб отримати векторні представлення, факторизація графу розкладає матрицю суміжності графу на множники, мінімізуючи наступну функцію втрат:

$$\mathrm{F}(Y,\lambda) = \frac{1}{2}\sum_{(i,j)\epsilon E}\left(W_{ij} - < Y_i, Y_j >\right)^2 + \frac{\lambda}{2}\sum_i \|Y_i\|^2,$$

де λ є коефіцієнтом регуляризації. Сума відбувається за наявними у графі ребрами, а не за усіма можливими ребрами, що дозволяє алгоритму працювати швидко за рахунок внесення шуму у рішення. Оскільки матриця суміжності часто не є додатно визначеною, мінімум функції втрат завжди буде більшим нуля, навіть якщо розмірність векторного представлення буде дорівнювати $|V|$. Часова складність алгоритму: $O(|E|d)$.

2.3.1.2 HOPE

Векторні представлення зі збереженням близькості вищого порядку (англ. High-Order Proximity preserved Embedding, HOPE) [126] зберігають близькість вищого порядку шляхом мінімізації

$$\|S - Y_s T_t^T\|_F^2,$$

де S — матриця схожості. Автори експериментували з різними оцінками схожості, включаючи індекс Каца, Rooted Page Rank, кількість спільних сусідів та оцінку Адаміка-Адара. Вони представили кожну міру подібності як $S = M_g^{-1} M_l$, де і M_g, і M_l є розрідженими. Це дозволяє методу HOPE використовувати сингулярний розклад матриці (SVD) [127] для ефективного отримання векторних представлень. Часова складність алгоритму: $O(|E|d^2)$. Автори методу продемонстрували його ефективність у задачах рекомендації вузла графу, прогнозування ребер графу та реконструкції графу.

2.3.1.3 LLE

Локально лінійні представлення (англ. Locally Linear Embedding, LLE) [128] – метод нелінійного зменшення розмірності, який передбачає, що кожен вузол є лінійною комбінацією своїх сусідів у векторному просторі. Якщо припустити, що елемент матриці суміжності W_{ij} графа G представляє вагу вузла j у представленні вузла i, то LLE визначає:

$$Y_i \approx \sum_j W_{ij} Y_j \qquad \forall i \in V.$$

При такому припущенні векторні представлення $Y^{N \times d}$ можна отримати за допомогою мінімізації:

$$L(Y) = \sum_i \left| Y_i - \sum_j W_{ij} Y_j \right|^2.$$

Для уникнення вироджених рішень, дисперсія векторних представлень обмежена: $\frac{1}{N} Y^T Y = I$. Додатково накладається умова, що векторне представлення повинна бути зосереджена навколо нуля: $\sum_i Y_i = 0$. Вищенаведена проблема оптимізації з обмеженнями може бути зведена до отримання власного розкладу, який

полягає в тому, щоб взяти нижні d + 1 власних векторів розрідженої матриці $(I - W)^T(I - W)$ і відкинути власні вектори, що відповідають найменшим власним значенням. Часова складність алгоритму: $O(|E|d^2)$.

2.3.1.4 GraRep

GraRep [129] – це метод, у якому ймовірність переходу з вузла у вузол визначається як $T = D^{-1}W$, а збереження близькості k-порядку відбувається шляхом мінімізації $\left\|X^k - Y_s^k Y_t^{kT}\right\|_F^2$, де X^k виводиться з T^k. Потім даний метод конкатенує Y_k для всіх k у нове представлення Y_s. GraRep доволі схожий на HOPE (2.3.1.2), який мінімізує $\|S - Y_s T_t^T\|_F^2$, де S — матриця подібності. Недоліком GraRep є масштабованість, оскільки T^k може мати O(|V|^2) ненульових записів. Часова складність алгоритму: $O(|V|^3)$. Автори методу продемонстрували його ефективність у задачах кластеризації, класифікації та візуалізації графів різного спрямування, таких як соціальна мережа, мережа документів новин та мережа цитування авторів наукових робіт.

2.3.1.5 Лапласівські проекції

Лапласівські проекції [130] мають на меті зберегти близькість між векторними представленнями двох вузлів, коли вага W_{ij} є високою. Зокрема, цей підхід намагається мінімізувати наступну цільову функцію:

$$\varphi(Y) = \frac{1}{2}\sum_{i,j}\left|Y_i - Y_j\right|^2 W_{ij} = tr(Y^T L Y), \text{ де}$$

L — Лапласіан графа G. На цільову функцію накладається обмеження $Y^T D Y = I$, щоб усунути тривіальне рішення. Розв'язання даної оптимізаційної задачі можна отримати, знайшовши власні вектори, що відповідають d найменшим власним значенням нормалізованого Лапласіана $L_{norm} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$. Часова складність алгоритму: $O(|E|d^2)$.

2.3.2 Графові представлення на основі випадкових блукань

Методи побудови векторних представлень графів на основі випадкових блукань генерують послідовності переходів з вузла у вузол, а потім навчаються на цих

послідовностях як вхідних даних. Такі методи особливо корисні, коли весь граф занадто великий і не поміщається в пам’ять, або усі вузли не є одразу доступні. Також їх використовують коли інші методи, які мають квадратичні відносно кількості вузлів вимоги до пам’яті, не працюють, або працюють надто повільно. Випадкові блукання були використані для апроксимації багатьох властивостей графа, включаючи центральність вузла [131] та подібність вузлів [132].

Для генерації послідовностей переходів з вузла в вузол класично використовули дві стратегії. Стратегії пошуку в ширину (Breadth-first search, BFS) і пошуку в глибину (Depth-first search, DFS) відіграють ключову роль у створенні представлень графу ітеративно – через онлайн навчання. Зокрема, сусіди вузла, відібрані за допомогою BFS, надають представленню характеристики, що апроксимують структурну еквівалентність між кластерами графу. Для встановлення структурної еквівалентності часто достатньо точно охарактеризувати місцеві околиці певних вузлів. Наприклад, структурну еквівалентність у комп’ютерних мережах (між комутаторами та маршрутизаторами) можна виявити просто спостерігаючи за безпосередніми околицями кожного вузла. Обмежуючи пошук найближчими вузлами, BFS надає представленню цієї характеристики через мікроскопічний погляд на околиці кожного вузла. Крім того, вузли у процесі створення вибірки можуть повторюватись декілька раз. Це теж важливо, адже зменшує дисперсію в характеристиці розподілу вузлів у найближчій околиці відносно вихідного вузла. Недоліком цього, є те, що лише невелика частина графу досліджується у одній ітерації.

На противагу BFS, DFS може досліджувати більші частини графу, оскільки при обході алгоритм може віддалятися від вихідного вузла вглиб графу. У DFS представлення вибіркових вузлів більш точно відображають макроскопічний погляд на граф та його ознаки, що є важливим для створення висновків про кластера та підграфи. Недолік DFS полягає в тому, що для багатьох задач важливо не лише зробити висновок, які залежності від вузла до вузла існують у мережі, а охарактеризувати точну природу цих залежностей. Також, більша глибина обходу

графу при навчанні призводить до моделювання складних залежностей, адже початковий вузол може бути дуже віддаленим від кінця згенерованого шляху, а отже представлення може бути менш репрезентативним. Методи на основі випадкових блукань мають кілька переваг над класичними підходами пошуку в ширину та глибину. Випадкові блукання є обчислювально ефективними з точки зору вимог як до простору, так і часу.

Найбільш популярними та перевіреними на практиці методами побудови векторних представлень вузлів, які працюють на основі ідеї випадкових блукань, є DeepWalk , Node2Vec та Walklets.

2.3.2.1 DeepWalk

DeepWalk [133] – метод, який зберігає близькість вищого порядку між вузлами, максимізуючи ймовірність спостереження попередніх і наступних k вузлів у випадковому блуканні з центром у вузлі v_i, тобто максимізуючи $log\ Pr\ (v_{i-k}, \dots, v_{i-1}, v_{i+1}, \dots, v_{i+k}|Y_i)$, де $2k + 1$ – довжина випадкового блукання. Для цього модель генерує велику кількість випадкових блукань і виконує ітеративну оптимізацію. Далі для відновлення ребер у даному методі використовується декодер добутку векторних представлень вузлів даного ребра. Часова складність алгоритму: $O(|V|d)$. Автори методу продемонстрували його ефективність у задачах багатокласової класифікації та кластеризації графу. Так як алгоритм навчання DeepWalk відноситься до класу онлайн алгоритмів, цей підхід є масштабованим. Він здатен будувати значущі векторні представлення для великих графів без ризику, що обчислювальних ресурсів може не вистачити. Також, цей підхід можна розподілити, що дозволяє окремим вузлам у кластерах оновлювати різні частини моделі одночасно.

DeepWalk не лише ефективний і масштабований, але й є привабливим узагальненням моделювання мови, адже документи та речення можна представити у формі графу, а метод Word2Vec (популярний у спільноті обробки природної мови) є частиною DeepWalk.

.

2.3.2.2 Node2Vec

Node2Vec [134] – це ітеративний метод побудови векторних представлень графів на основі випадкових блукань та Word2Vec моделей. Node2Vec зберігає близькості між вузлами високих степенів за допомогою ймовірностей переходів між вузлами під час випадкових блукань. Процес в основному складається з двох кроків: генерація упереджених випадкових блукань і оптимізація ваг вузлів. Перший крок створює послідовності вузлів графа різної довжини, щоб модель могла зрозуміти як локальні, так і глобальні особливості графа. Саме упередженість у генерації випадкових блукань відрізняє Node2Vec від подібних методів, адже це забезпечує компроміс між пошуком в графі в ширину (англ. breadth-first search, BFS) і в глибину (англ. depth-first search, DFS). Цим керують два основні гіперпараметри моделі p - гіперпараметр повернення, та q - гіперпараметр виходу. Вибір правильного відношення цих гіперпраментрів дозволяє Node2Vec зберегти локальну структуру графу, а також близькості вищих степенів. Даний метод є узагальненням методу DeepWalk, адже при p=1 та q=1, дані підходи – еквівалентні. На другому кроці виконується ітеративна процедура, яка оновлює випадково ініціалізовані векторні представлення вузлів відповідно до векторних представлень їх контекстних вузлів. Контекстні вузли — це вузли ліворуч і праворуч від певного вузла в послідовності, створеній випадковим блуканням. Часова складність алгоритму: $O(|V|d)$. Автори методу продемонстрували його ефективність у задачах багатокласової класифікації та прогнозу ребер графу у прикладних областях прогнозування взаємодії протеїнів, класифікації зв’язків у Wikipedia та соціальних мережах для блогінгу.

2.3.2.3 Walklets

Node2Vec (2.3.2.1) і DeepWalk (2.3.2.2) неявно зберігають близькість вищого порядку між вузлами, генеруючи велику кількість випадкових блукань, які з'єднують вузли на різних відстанях. У той же час, підходи, засновані на факторизації, такі як факторизація графу (2.3.1.1) і HOPE (2.3.1.2), зберігають її явно, моделюючи відстані

між вузлами у своїй цільовій функції. Walklets [135] – метод, що поєднує ідею явного моделювання відстаней між вузлами з ідеєю випадкових блукань. Модель змінює стратегію генерації випадкового блукання, що використовується в інших методах, пропускаючи деякі вузли у графі. Таким чином випадкові послідовності створюються на різних рівнях графу, для різних значень кількості пропусків. Інтуїтивно подібною процедурою є факторизація матриці A^k в методі GraRep (2.3.1.4). Отриманий набір випадкових блукань використовується для навчання моделі методом Word2Vec, подібно до інших методів на основі випадкових блукань. Часова складність алгоритму: $O(|V|d)$. Автори методу продемонстрували його ефективність у задачах багатокласової класифікації на наборах даних пов'язаних з соціальними мережами, такими як Flickr та YouTube. Як і два попередні методи, він навчається за допомогою онлайн навчання, а отже – масштабований, може працювати з величезними графами, такими як граф словника синонімів.

2.3.3 Графові представлення на основі глибокого навчання

Зростання популярності досліджень глибокого навчання у областях комп'ютерного зору та обробки природних мов призвело до створення методів на основі глибоких нейронних мереж для моделювання графів. Основною ідеєю таких підходів є використання глибоких автокодерів, які успішно застосовуються як методи зменшення розмірності через здатність моделювати нелінійні структури у даних [136]. У випадку графів глибокі автокодери здатні кодувати, а потім відтворювати різні структурні аспекти графів у залежності від їх представлення, яке визначає тип методу на основі глибокого навчання.

2.3.3.1 SDNE

Венг та ін. [137] запропонували метод Структурних Глибоких Мережевих Представлень (англ. Structural Deep Network Embedding, SDNE), ідея якого - використовувати глибокі автокодери для збереження близькості між вузлами першого та другого порядку. Вони досягають цього шляхом спільної оптимізації обох близькостей. Підхід використовує набір нелінійних функцій на різній глибині

нейронної мережі для отримання векторних представлень вузлів. Модель складається з двох частин: навчання без розмітки та навчання з розміткою. Перша частина містить у собі автокодер, спрямований на пошук такого векторного представлення вузла, яка б могла реконструювати сусідні вузли. Друга частина базується на Лапласівських проекціях (2.3.1.5), які штрафують модель, коли близькі вершини графу відображаються далеко одна від одної у побудованому векторному просторі. Більш детально: щоб охопити дуже нелінійну структуру графів, автори SDNE пропонують використати глибоку архітектуру, яка складається з кількох нелінійних функцій відображення для трансформації вхідних даних у дуже нелінійний латентний простір. Це допоможе охарактеризувати структуру мережі. Крім того, для вирішення проблеми збереження структури та розрідженості, було запропоновано напівконтрольовану модель для використання близькостей як першого так і другого порядків. На першому етапі для кожної вершини отримуються виділяються її сусіди. Відповідно, напівконтрольована модель проектується так, щоб зберегти близькість другого порядку шляхом реконструкції структури сусідства кожної вершини. Тим часом для малої частини пар вузлів, можна отримати попарну подібність, тобто близькості першого порядку. Контрольована модель, використовує це як інформацію для навчання представлень у латентному просторі.

Шляхом спільної оптимізації обох моделей метод SDNE може добре зберігати високонелінійну локально-глобальну структуру графу та надійно працює навіть з розрідженими графами. Часова складність алгоритму: $O(|E||V|)$.

2.3.3.2 DNGR

Глибокі Нейронні Мережі для Вивчення Векторних Представлень Графу (англ. Deep Neural Networks for Learning Graph Representations, DNGR) [138] поєднує випадкове блукання з глибоким автокодером. Випадкове блукання відбувається подібно до DeepWalk. Це забезпечує перетворення незваженої структурної інформації графа в лінійні послідовності, які виражають зв’язок між вершинами графа. Таке

випадкове блукання є універсальним методом побудови вибірки, який підходить для незважених графів.

Модель складається з 3-х етапів: випадкове блукання, обчислення позитивної поточкової взаємної інформації (англ. Positive Pointwise Mutual Information, PPMI) та створення шумопоглинаючих автокодерів. Модель випадкового блукання використовується на вхідному графі для створення імовірнісної матриці спільного виникнення, аналогічної матриці подібності в методі на основі факторизації HOPE (2.3.1.2). Потім дана матриця перетворюється в PPMI матрицю і подається на вхід набору шумопоглинаючих автокодерів для отримання векторних представлень. Використання матриці PPMI на вході гарантує, що модель автокодера може захопити близькість між вузлами вищого порядку. У свою чергу, використання шумопоглинаючих автокодерів сприяє надійності моделі за наявності шуму у графі, а також захопленні базової структури, необхідної для таких завдань, як прогноз ребер та класифікація вузлів. Часова складність алгоритму: $O(|V|^2)$.

2.3.3.3 GCN

Методи на основі глибоких нейронних мереж, які були описані у попередніх підрозділах, а саме SDNE та DNGR, приймають на вхід дані про глобальних сусідів кожного вузла (рядок матриці PPMI для DNGR і матриці суміжності для SDNE). Знаходження цих вхідних даних може бути обчислювально дорогим і неоптимальним кроком для великих розріджених графів. Графові згорткові мережі (англ. Graph Convolutional Networks, GCN) [139] вирішують цю проблему, визначаючи оператор згортки на графі. Модель ітеративно агрегує векторні представлення сусідів для вузла та застосовує нелінійну функцію до отриманого представлення та представлення на попередній ітерації для отримання нового представлення. Агрегування векторних представлень лише локальних сусідів певного вузла робить цей метод масштабованим, а численні ітерації та глибина мережі дозволяють навченим векторним представленням вузла характеризувати і глобальну структуру графа.

Підходи до побудови графових згорткових мереж розрізняються у виборі згорткових фільтрів, які можна широко розділити на просторові та спектральні. Просторові фільтри працюють безпосередньо на вхідному графі та його матриці суміжності, тоді як спектральні фільтри діють на спектрі Лапласіана графа. Часова складність алгоритму: $O(|E|d^2)$.

2.3.4 Інші

Методи, які не можна однозначно віднести до однієї з вищезазначених категорій зазвичай спираються на поєднання одразу кількох ідей з суміжних парадигм. Так [140] будує векторне представлення графа за його відстанями до графів-прототипів. [141] спочатку створює векторне представлення кількох вузлів орієнтирів, використовуючи їх попарні найкоротші шляхи. Потім інші вектори інших вузлів модифікуються так, щоб їх відстані до підмножини орієнтирів були максимально близькими до реальних найкоротших шляхів. [142] пропонує оптимізувати функцію втрат на основі ребер (максимізуючи ймовірність спостереження сусідів вузла замість більш віддалених вузлів) разом з функцією втрат на основі атрибутів вузла. Підхід KR-EAR [143] виділяє зв'язки в графі знань на основі атрибутів та на основі відношень. Він створює відносний потрійний кодер (TransE, TransR) для того, щоб побудувати представлення кореляцій між сутностями та відношеннями, а також атрибутний потрійний кодер для того, щоб побудувати представлення кореляцій між сутностями та атрибутами. Struct2vec [144] будує структурне представлення вузлів за допомогою ієрархічної метрики для векторного представлення вузлів. Більшість з таких методів вимагають наявність додаткових атрибутів у вузлів графа, що є суттєвим обмеженням при роботі у контексті обробки природної мови у малоресурсних середовищах. Тому для досліджень даної дисертації більш детально розглядається лише підхід LINE.

2.3.4.1 LINE

Векторні представлення високомасштабної інформаційної мережі (англ. Large-scale information network embedding, LINE) [145] – метод, який явно визначає дві функції, по одній для близькостей першого та другого порядку, і мінімізує їх

комбінацію. Функція близькості першого порядку подібна до функції, що використовується у підході факторизації графу (2.3.1.1) у тому, що вони обидві спрямовані на те, щоб матриця суміжності та добуток векторних представлень були близькими. Різниця полягає в тому, що метод факторизації графу робить це безпосередньо шляхом мінімізації різниці між ними. Замість цього LINE визначає два спільних розподіли ймовірностей для кожної пари вершин, один з яких використовує матрицю суміжності, а інший — поточні векторні представлення. Тоді LINE мінімізує розбіжність Кульбака-Лейблера цих двох розподілів. Перший з розподілів виглядає наступним чином:

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-Y_i Y_j^T)}.$$

Другий розподіл визначається так:

$$\hat{p}_1(v_i, v_j) = \frac{W_{ij}}{\sum_{(i,j)\in E} W_{ij}}.$$

Тоді функція для мінімізації, що використовується підходом LINE:

$$L = KL(p_1, \hat{p}_1) = -\sum_{(i,j)\in E} W_{ij} \log\, p_1(v_i, v_j)$$

Аналогічно автори визначають розподіли ймовірностей та цільову функцію для близькості другого порядку. Часова складність алгоритму: *O(|E|d)*.

2.4 Злиття векторних представлень

Злиття векторних представлень (англ. fusion) є ключовою темою дослідження в області мультимодального машинного навчання, яке присвячене побудові моделей здатних об’єднувати інформацію з різних унімодальних джерел даних у одне зручне для використання представлення. Злиття векторних представлень можна визначити як набір трансформацій:

$$f : s_{it} \in S_i^{(doc)} \rightarrow \mathbf{w_{it}^{(doc)}} \in R^m, \forall i \in \{1, \ldots, n\}$$

$$g : s_{it} \in S_i^{(doc)} \rightarrow \mathbf{w_{it}^{(dict)}} \in R^d, \forall i \in \{1, \ldots, n\}$$

$$\mathbf{w_i^{(doc)}} = \frac{1}{\|S_i^{(doc)}\|} \sum_t^{\|S_i^{(doc)}\|} \mathbf{w_{it}^{(doc)}}$$

$$\mathbf{w_i^{(dict)}} = \frac{1}{\|S_i^{(doc)}\|} \sum_t^{\|S_i^{(doc)}\|} \mathbf{w_{it}^{(dict)}}$$

$$h : \mathbf{w_i^{(doc)}} \in R^m, \mathbf{w_i^{(dict)}} \in R^d \rightarrow \mathbf{w_i} \in R^k$$

, де $\mathbf{w_{it}^{(doc)}}$ — векторне представлення розміру *m* слова на місці *t* у документі *i* на основі тексту, $\mathbf{w_{it}^{(dict)}}$ — векторне представлення розміру *d* слова на місці *t* у документі *i* на основі словника, $S_i^{(doc)}$ — множина слів у документі *i,* $\mathbf{w_i^{(doc)}}$ — векторне представлення документа *i* на основі тексту, $\mathbf{w_i^{(dict)}}$ — векторне представлення документа *i* на основі словника, а *k* — результуючий розмір представлення.

Існує чіткий зв'язок між злиттям і мультимодальними векторними представленнями. Так, підхід класифікується як підхід злиття, якщо він зосереджується на архітектурах для інтеграції унімодальних представлень для певного завдання. Кожна об’ємна прикладна область, у якій застосовують сучасні методи глибокого навчання має набір методів, які можуть генерувати унімодальні векторні представлення. Наприклад, для області комп’ютерного зору такими представленнями (іноді називаються «візуальними») є виходи фінальних шарів згорткових нейронних мереж, які кодують інформацію на зображенні [146]. Для області обробки природних мов існують мовні представлення – для документів це виходи рекурентних нейронних мереж, трансформерів, а для слів – векторні представлення слів [147]. За допомогою методів мультимодального навчання такі представлення можна поєднувати для вирішення задач, які вимагають розуміння мультимодальних даних, наприклад генерації опису картинок, де мовна модель поєднується з візуальною.

Загалом класифікація методів злиття за механізмами роботи у мультимодальному навчанні налічує три типи: прості методи на основі операцій, методи на основі уваги та методи на основі білінійного об’єднання [148]. Перший тип

припускає, що векторні представлення з різних джерел інформації можна інтегрувати за допомогою простих операцій, таких як конкатенація або зважена сума, які не мають пов’язаних параметрів, а при навчанні вищі шари глибоких моделей можуть пристосуватись до виходів таких операцій. До другого типу відносяться модифікації механізму уваги - зваженої суми набору векторів зі скалярними вагами, які динамічно генеруються невеликою моделлю «уваги» на кожному кроці. При цьому часто використовується кілька таких моделей, які можуть зберегти додаткову інформацію шляхом об’єднання результатів, отриманих від кожного шару. Методи на основі уваги використовуються з моделями, що вимагають великої кількості даних, таких як згорткові чи рекурентні нейронні мережі, так як додають великий набір параметрів, які потрібно підбирати у ході навчання. Саме тому у контексті обробки природної мови у малоресурсному середовищі вони не розглядаються. Останній тип - методи на основі білінійного об’єднання - часто використовується для злиття векторів візуальних ознак з векторами текстових ознак для створення спільного векторного простору шляхом обчислення їх зовнішнього добутку, що гарантує мультиплікаційну взаємодію між усіма елементами в обох векторах. Цей метод також називають злиттям другого порядку. На відміну від простих операцій комбінування векторів (припускаючи, що кожен вектор має n елементів), таких як зважена сума або конкатенація, які призводять до n- або $2n$-вимірних представлень, білінійне об'єднання створює n^2-вимірне представлення шляхом лінеаризації матриці, створеної зовнішнім добутком, у вектор. Це робить даний метод більш виразним, але через високу розмірності фінальних представлень (як правило, від сотень тисяч до кількох мільйонів), білінійне об’єднання часто вимагає додаткових декомпозицій та велику кількість даних, щоб пов’язана модель могла навчатися належним чином та ефективно. Подібно до попереднього типу, дані методи непрактично застосовувати у малоресурсних середовищах.

Методи злиття можна також розділити за стадією, на якій відбувається злиття під час відповідних процедур їх застосування. Злиття на ранніх шарах нейронної

моделі може зменшувати сигнал від внутрішньомодальної взаємодії, тоді як занадто пізнє злиття – зовнішньомодальної взаємодії. Тому останнім часом зазвичай використовуються методи злиття на проміжних шарах фінальної моделі [149].

Так як слова у лінгвістичних словниках зазвичай надаються у стандартному написанні (наприклад, українською мовою для дієслів - неозначена форма, для іменників - називний відмінок), а у прикладних текстах зустрічаються зі словозміною, потрібно це враховувати для пошуку відповідних вузлів графа при злитті. Саме тому пропонується модифікувати прості методи на основі операцій для використання зі словниковою інформацією. За наявності експертних знань мови та великої кількості розмічених наборів даних для пошуку відповідного вузла варто побудувати дві моделі: моделі визначення частини мови та моделі пошуку словозмін. Таким чином за відсутності слова з документу в словнику визначається його частина мови, а потім на її основі будуються або усі можливі словозміни або та словозміна, на основі якої побудований словник. Результуючі форми слів знаходяться у векторних представленнях графа словника.

У малоресурсному середовищі вищезазначена процедура не практична, адже експертних знань та розмічених наборів даних для побудови проміжних моделей може не існувати. Тому пропонується використати один з методів визначення відстаней між рядками (наприклад, відстань Левенштейна [150]) як критерій подібності, за яким можна знайти форму слова у графі словника:

$$g(s_t) = W^{(dict)}_{argmin_j(distance(s_t,s_j))}, \forall s_j \in S^{(dict)}$$

, де $distance(s_t, s_j)$ – критерій відстаней між словами s_t та s_j.

Їй буде відповідати слово з мінімальною дистанцією до слова з документа. Таким чином можна знайти правильну словоформу без побудови складної моделі. Недоліком цього підходу є низька швидкодія при великому розмірі словника, адже часова складність обчислення відстані Левенштейна для двох слів довжин n та m є $O(n*m)$, а таких порівнянь для одного документа з k слів та словника з t слів у

найгіршому випадку (усі слова є словоформами) потрібно зробити *k*t*. Також у мовах, де словоформи не є синтаксично подібними між собою, а також у виняткових випадках (наприклад, слово «бути» та його форма «є») даний метод не здатен працювати та вимагає додаткових правил.

2.4.1 Злиття на основі конкатенації

Злиття на основі конкатенації може використовуватися для поєднання або низькорівневих вхідних ознак, або високорівневих векторних представлень, отриманих попередньо навченими моделями. Прикладом злиття низькорівневих ознак є робота [151], де для задачі оцінки переконливості людей у мультимедійних даних конкатенують вектори текстових транскриптів на основі мішка слів або TF-IDF, візуальні дескриптори відео на основі аналізу емоцій, орієнтації та рухів лиця, а також набір акустичних дескрипторів таких як якості голосу, просодії та Мелчастотні кепстральні коефіцієнти. У свою чергу, автори [152] показують, що конкатенація високорівневих ознак навіть таких простих як виходи моделей різних модальностей може покращувати результати вирішення різноманітних задач комп'ютерного зору. Іншим прикладом об'єднання високорівневих векторних представлень є дослідження [153], де показано, що для вирішення задачі відповідей на візуальні питання можна використати злиття векторних представлень картинок побудованих за допомогою згорткової нейронної мережі GoogLeNet та векторних представлень слів. Класифікаторами, на вхід яких приходять сконкатеновані векторні представлення, часто виступають нейронні мережі, адже вони здатні неявно зважувати ознаки та виокремлювати ще більш абстрактні ознаки у проміжних шарах.

Злиття за допомогою конкатенації визначається як:

$$\mathbf{w_i^{concat}} = \mathbf{w_i^{(doc)}} \frown \mathbf{w_i^{(dict)}}, k = m + d$$, де

$\mathbf{w_i^{concat}}$ — векторне представлення документа *i* після злиття через конкатенацію.

Схематичне зображення запропонованої модифікації методу злиття векторних представлень слів та векторних представлень вузлів граф на основі конкатенації для побудови векторного представлення документа з *N* слів зображено на рис. 2.6.

Очевидно, що для методу злиття на основі класифікації розмір фінального

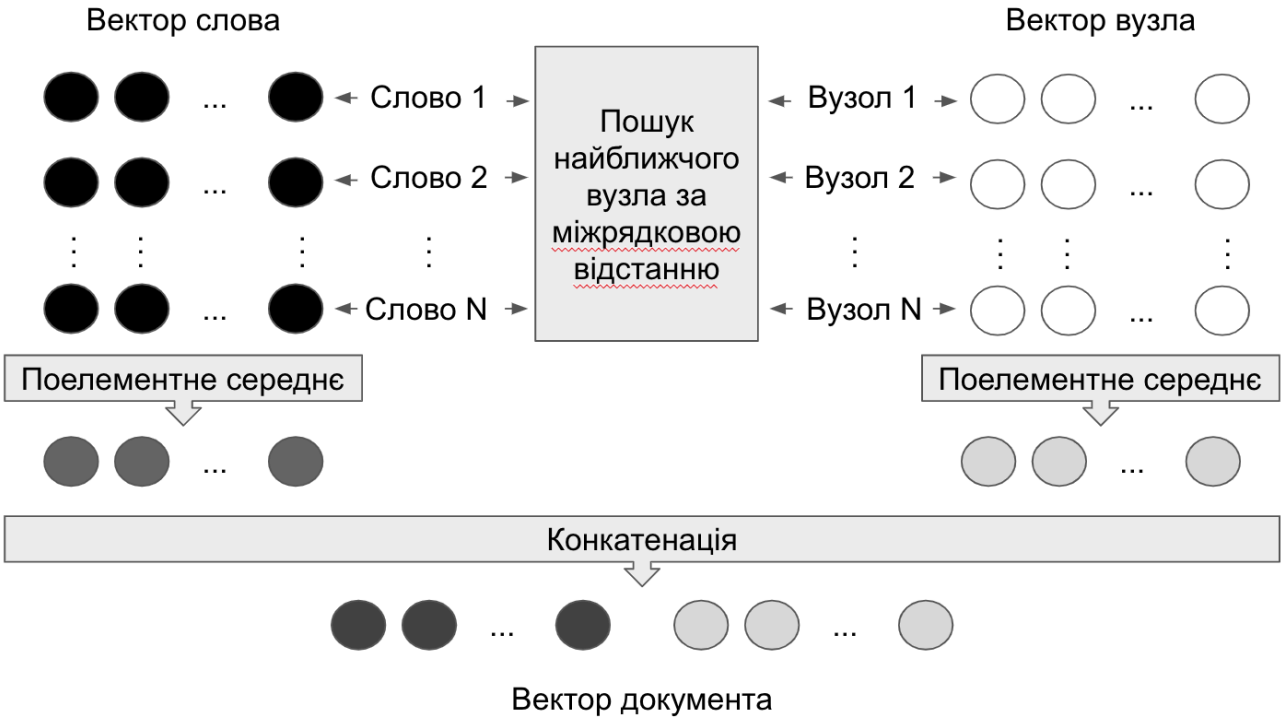

Рис. 2.6 – Злиття на основі конкатенації

векторного представлення рівний сумі розмірів представлень учасників операції: $k = m + d$.

2.1.1 Злиття на основі зваженої суми

Злиття на основі зваженої суми визначається як:

$$\mathbf{w_i^{weighted\_sum}} = \alpha \mathbf{w_i^{(doc)}} + \beta \mathbf{w_i^{(dict)}}, k = m = d$$

, де

$\mathbf{w_i^{weighted\_sum}}$ — векторне представлення документа i після злиття на основі зваженої суми [154]. У контексті дисертаційних досліджень β зважує векторні представлення словника, а α – документа. Схематично дана модифікація методу злиття на основі зваженої суми показана на рис. 2.7.

Для злиття на основі зваженої суми, де вагами виступають скаляри, важливою вимогою до вхідних векторів є те, що попередньонавчені векторні представлення різних модальностей повинні мати одну розмірність, а отже мати змогу поелементно додаватись, тобто щоб m дорівнювало d, а отриманий розмір k дорівнював m. Якщо

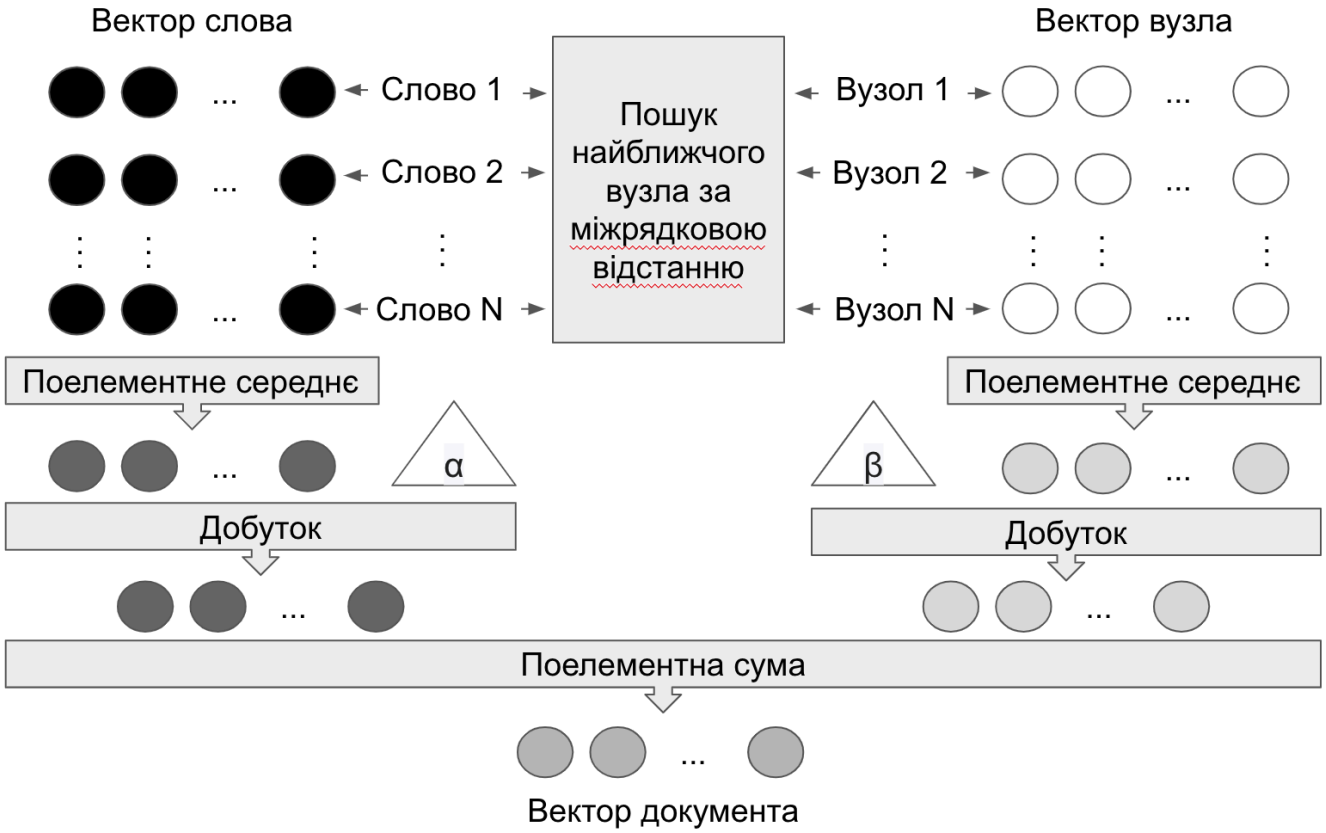

Рис. 2.7 – Злиття на основі зваженої суми

розмірності вхідних векторів не співпадають, то можна використати методи зменшення розмірності такі як метод головних компонент або - при використанні нейронних мереж - спроектувати невідповідні за розміром простори у відповідні за допомогою додаткового прихованого шару [155].

У схемі злиття на основі зваженої суми є важлива деталь – ваги, на які домножуються векторні представлення різних модальностей. Обидва гіперпараметри α і β можна інтерпретувати як рівень внеску кожного з типів представлень у кінцевий вектор. Зважаючи на збитковість інформації, яка може виникати у кількох джерелах даних, а також властивість глибоких моделей перенавчатись, іноді ці ваги обмежують

відношенням $\beta = 1 - \alpha$, але для запропонованого методу відповідних обмежень не накладається. Це робиться, тому що векторні представлення словників збагачують векторні представлення документів новою інформацією і не повинні зменшувати їх сигнал.

2.2 Висновки до розділу

У другому розділі дисертації було представлено метод доповнення векторних представлень документів словниковими представленнями для покращення роботи методів класифікації документів при обробці природної мови у малоресурсному середовищі. Цей метод полягає в тому, що словник можна розглядати як граф, де вузли – слова, а зв'язки між словами (відповідно до типу словника) – ребра, а отже інформацію, яка у ньому знаходиться можна закодувати методами векторного представлення графів. Далі ці векторні представлення можна поєднати з векторними представленнями документів методами злиття векторних представлень. Таким чином утворюється вектор ознак, що кодує статистичні властивості наявних даних та додаткові словникові уявлення про середовище, який можна передати до моделі класифікації.

Описаний метод складається з набору компонент, кожен з яких має різні методи реалізації. У даному розділі було проаналізовано методи, які можна використовувати для кожного компонента у малоресурсному середовищі, а також яким чином поєднувати результати їх роботи. Обмеженість доступності даних для тренування у малоресурсних середовищах робить частину сучасних методів обробки природних мов, таких як трансформери, непрактичною. Так, для побудови векторних представлень документів можна використовувати як статистичні підходи, наприклад мішок слів чи TF-IDF, так і нейромережеві підходи, що базуються на векторних представленнях слів. У свою чергу, векторні представлення слів можна будувати методами Word2Vec або FastText, який є модифікацією першого для роботи з невідомими словами.

Методи побудови векторних представлень графів теж різняться за принципами своєї дії, а отже вибирати їх потрібно у залежності від прикладного середовища, наявних даних та особливостей словника, який кодується. Зазвичай лінгвістичні словники доволі великі, так як сучасні природні мови можуть налічувати десятки та сотні тисяч слів, тому методи побудови векторних представлень графа на основі факторизації графу, які розкладають матричні представлення графа на складові можуть працювати надто довго або вимагати велику кількість пам'яті. У даному випадку методи на основі випадкових блукань підходять до прикладної області краще, адже можуть ітеративно оновлювати векторні представлення оптимізуючи певну функцію втрат. Іншим варіантом побудови векторних представлень графа є методи на основі глибокого навчання, такі як глибокі графові згорткові мережі та структурні глибокі мережеві представлення, які здатні навчатись ітеративно і досягають високих результатів при моделюванні графів.

Ще одним важливим елементом запропонованого методу є процес злиття векторних представлень документів та векторних представлень словника. Хоч методи злиття векторних представлень часто вбудовуються у метод класифікації документів, що додає велику кількість параметрів до моделі, у малоресурсному середовищі сигналу для навчання додаткових параметрів може не вистачити. Тому у ході дослідження, було розглянуто лише методи злиття векторних представлень на основі простих операцій, а саме конкатенації та зваженої суми. Обидва підходи можуть бути використані у малоресурсному середовищі для отримання доповнених векторних представлень документів, які можна передавати алгоритмам класифікації. Так як розроблений метод вимагає знаходження відповідності між словами у документі та словнику синонімів, дані підходи були модифіковані за допомогою етапу пошуку відповідника за одним з критеріїв міжрядкової відстані, таких як відстань Левенштейна. Дана модифікація дозволяє знаходити відповідності між словами без побудови окремих моделей визначення частини мови та пошуку словоформ, що

суттєво ускладнено у малоресурсних середовищах, де розмітки для даних підзадач може не існувати.

У наступному розділі описуються експерименти з запропонованим методом, обираються та мотивуються найкращі комбінації компонентів, архітектури та практичні особливості застосування даного методу при обробці природної мови у малоресурсному середовищі на прикладі класифікації петицій до Київської міської ради за темами за наявності словника синонімів української мови.

РОЗДІЛ 3. ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ

3.1 Опис експериментальних даних

Дослідницький аналіз даних (англ. Exploratory data analysis, EDA) має вирішальне значення для розуміння основних концепцій, присутніх у даних, і є першим кроком кожного проєкту, пов’язаного з даними. Саме на основі проведення такого аналізу, можна чітко усвідомити ознаки, особливості даних, їх загальні статистики, а також виокремити потенційні проблеми при моделюванні та упередженості, які можуть бути присутні у даних та передатись моделям, що навчаються. EDA в першу чергу використовується для того, щоб побачити, що може знаходитись усередині даних за межами завдання формального моделювання або перевірки гіпотез, і забезпечує краще розуміння залежних та незалежних змінних присутніх у наборі даних і взаємозв’язків між ними. Також даний процес може допомогти визначити, чи підходять статистичні методи, які розглядаються, для аналізу даних. Основна мета EDA – переглянути дані, перш ніж робити якісь припущення. Це може допомогти виявити очевидні помилки, а також краще зрозуміти закономірності в даних, виявити викиди або аномальні події, знайти цікаві співвідношення між змінними [156].

Дослідники даних можуть використовувати дослідницький аналіз даних, щоб переконатися, що результати, які вони отримують, є дійсними та можуть застосовуватись до певних бажаних завдань і цілей бізнесу. EDA також допомагає усім зацікавленим сторонам пересвідчитись та підтвердити те, що поставлені запитання є правильними. Частими результатами EDA є статистичні ознаки наборів даних (розподіли змінних, стандартні відхилення та інші), класифікація змінних за типами (чисельні чи категоріальні), візуалізації окремих змінних чи закономірностей між змінними.

У даному дослідженні було зібрано два набори даних, EDA яких наводиться у даному розділі. Перший набір даних – петиції до Київської міської ради має відповідну розмітку за темами. Для того, щоб зробити таку розмітку автоматичною,

визначається завдання класифікації документів, яке ми використовуємо для порівняння методів, запропонованих у дисертації. Класичні методи обробки природної мови використовують лише інформацію, наявну в самих петиціях для створення якісних моделей класифікації, тому попередній аналіз даної інформації надзвичайно важливий. Другий набір даних – словник синонімів української мови, який використовується для покращення існуючих методів класифікації документів у запропонованому методі. Так як найприродніша структура для такого типу даних це граф, а не таблиця, замість звичайних методів EDA, які аналізують змінні, пропонується аналіз властивостей побудованого графу.

Прямим результатом EDA є формулювання набору кроків для передобробки даних до моделювання. Таким чином, на вхід запропонованим підходам будуть надаватись чисті дані, що сприятиме покращенню роботи як стандартних підходів, так і запропонованих у даному дослідженні.

3.1.1 Набір даних для класифікації документів

Системи онлайн-петицій відіграють ключову роль у функціонуванні та вдосконаленні кожної країни. Такі системи дозволяють громадянам брати участь у розвитку країни, надсилаючи владі обґрунтовані ідеї та висловлюючи стурбованість щодо певних рішень або ситуацій. Це означає, що кожен громадянин може подати власну пропозицію адміністрації країни. Зрозуміло, що не кожна петиція може бути корисною, адже деякі петиції часто висловлюють лише суб’єктивні погляди різних осіб. Тому, в першу чергу, було б корисно дослідити петиції та з’ясувати, які вони мають закономірності. Результати такого дослідницького аналізу петицій також можуть бути корисними для подальшої автоматизації розгляду звернень. У цій роботі далі наводяться статистичні характеристики та закономірності онлайн-петицій українською мовою, поданих до Київської міської ради, отримані шляхом дослідницького аналізу даних.

Тема аналізу онлайн-петицій не нова. Автори [157] використовували прихований розподіл Діріхле (англ. latent Dirichlet allocation, LDA) з метою

автоматизації виділення тем у наборі петицій. У [158] було проаналізовано Twitter як платформу для обговорення і дебатів на рахунок електронних петицій, а також проведено аналіз тональності їх текстів. Задача оцінки популярності петиції на основі семантики текстового набору даних була досліджена у [159]. Більшість літератури за цією темою обговорює петиції, написані англійською мовою з величезними текстовими наборами даних. З іншого боку, у даній роботі використовується невеликий набір петицій українською мовою, а отже, дослідницький аналіз даних повинен враховувати у собі обмеженість кількості даних та мовне середовище, у якому проводиться.

Київ — столиця та найбільш густонаселене місто України. Його населення у липні 2015 року становило 2 887 974 [160] (хоча за неофіційними даними ця цифра перевищує 3 мільйони, оскільки місто постійно зростає), що робить Київ 7-м за чисельністю населення містом у Європі [161].

У жовтні 2015 року Київська міська рада запровадила систему онлайн-петицій та запустила тестовий сайт для подачі петицій громадянами – petition.kievcity.gov.ua. За даними сайту, електронна петиція – це сервіс, який дає можливість громадянам подавати свої ініціативи до Київради. У разі отримання 10 000 підписів протягом 90 днів ініціативи будуть розглянуті відповідальними людьми Київради, реалізовані (за можливості), а також буде оприлюднено офіційну відповідь. Петиція, яка не отримала необхідної кількості електронних підписів, може бути повернута автору для обговорення [162].

З метою отримання текстів петицій з веб-сайту було виконано процедуру індексування веб-сторінок за допомогою пошукового робота. Пошуковий робот (кроулер, павук) — це частина програмного забезпечення, яка сканує Інтернет сторінки за заздалегідь визначеними правилами в пошуках цінної інформації [163]. Одним з найпопулярніших і добре розроблених інструментів для створення таких пошукових роботів є Scrapy - швидкий високорівневий фреймворк мови програмування Python для сканування веб-сайтів і веб-скрейпінгу, який

використовується для пошуку веб-сайтів і вилучення структурованих даних з їхніх сторінок. Його можна використовувати для широкого спектру цілей, від інтелектуальної обробки даних до моніторингу та автоматизованого тестування [164]. Простий програмний інтерфейс у вигляді класу Python, який має фреймворк, можливість знаходити елементи сторінки за допомогою мови запитів XML Path Language (Xpath) [165] та можливість відфільтровувати повторювані сторінки зробили його дуже ефективним інструментом для вирішення будь-яких пошукових проблем.

Збір петицій зайняв 20 хвилин на процесорі Intel Core i5 з тактовою частотою 2,8 ГГц при підключенні до Інтернету 100 Мбіт/с і закінчився 6560 елементами, де кожен елемент є петицією зі своїм ідентифікатором, текстовим вмістом, назвою, автором, темою і датою публікації. Так як веб-сайт Київради для правильної роботи пагінації переліку петицій сильно опирається на HTTP-Кукі, при імплементації скрипта автоматичного парсингу даних сторінок було використано мета-інформацію у вигляді Cookiejar – автоматичного менеджера кукі, що передавався від запиту до запиту. Усі правила «чемного кроулінгу» [166] було витримано за допомогою автоматичного тротлінгу (зменшення частоти запитів з метою уникнення атаки на відмову в обслуговуванні (англ. Denial-of-service attach, DOS attack)) та обмеження кількості відкритих зв’язків до одного (без розподілених запитів чи обробки їх результатів). Для збереження результатів пошукового робота було обрано формат jsonlines (jsonl), так як кожна лінія у даному форматі – самостійна сутність з описом усіх полів, а тому будь-який зріз збереженого у цьому форматі набору даних буде готовий до аналізу. Більше того у даному форматі немає проблеми з кодуванням та вичиткою багаторядкових текстів, тоді ж як у більш стандартних табличних форматах, як comma-separated values (CSV), збереження таких записів ускладнюється потребою замикання як ком, так і знаків переносу рядків.

Як згадувалося раніше, якщо представити набір даних таблично, він складається з 6560 рядків та 6 стовпців:

1. «id» - унікальний ідентифікатор петиції, який автоматично призначається веб-сайтом. Не має особливого значення для моделювання чи аналізу, але корисний для перевірки коректності отримання даних;
2. «name» – текстове поле різної довжини із заголовком петиції. Зазвичай містить короткий зміст петиції або заклик до дії;
3. “content” – текстове поле, що складається з повного тексту певної петиції;
4. «author» – текстове, яке зазвичай містить ім’я, прізвище та по батькові автора. Через відсутність підтвердження на оригінальному веб-сайті може також складатися з об’єднаних кількох імен осіб, а також електронні адреси та назви організацій;
5. «datetime» – дата розміщення петиції у форматі «РРРР-ММ-ДД»;
6. «topic» – тема петиції;

Історію подачі петицій можна побачити на рис. 3.1. З нього видно, що найбільшу кількість петицій було подано протягом перших кількох місяців існування сайту у 2015 році.

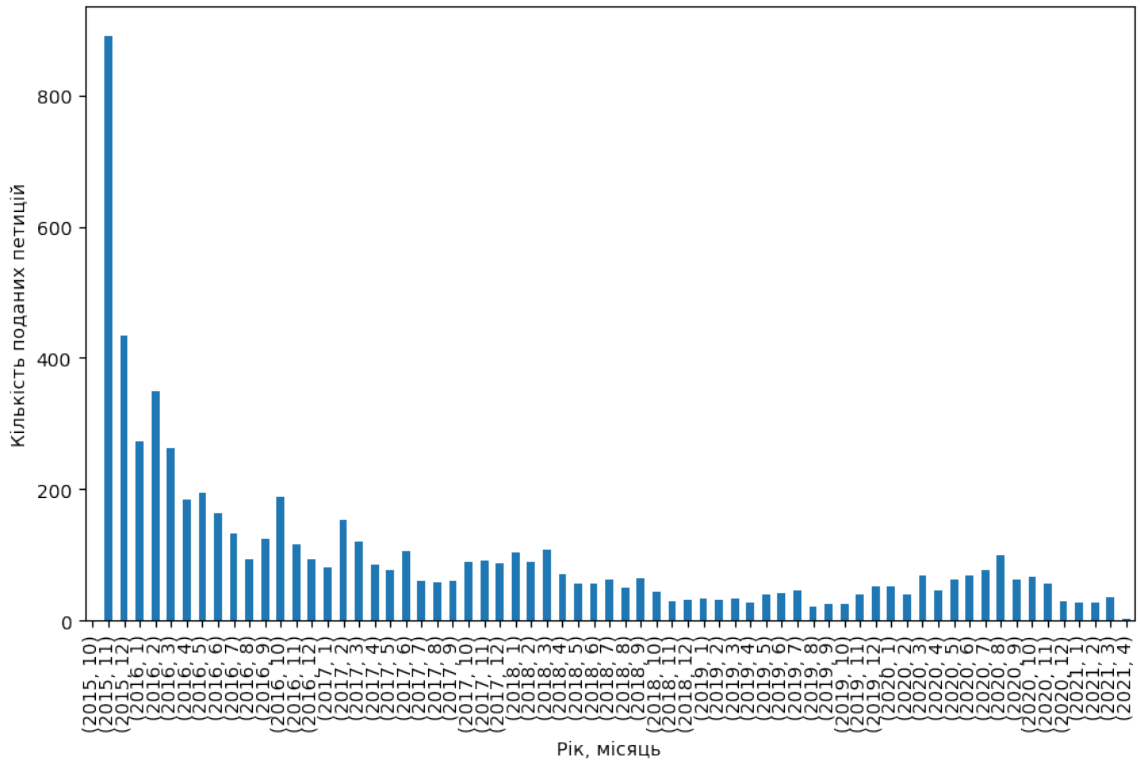

Рис. 3.1 – Кількість петицій за роком та місяцем їх подання

Далі наводяться візуалізації розподілів стовпців набору даних та деякі їх описові статистики. Так як розподіли проаналізованих характеристик даних не є нормальними, додатково наведені медіани та моди розподілів, оскільки вони є більш значущими для аналізу.

Перше, що можна зробити з текстовими стовпцями, це зрозуміти розподіл їх довжини. Однак довжина символів не має такого значення, як підрахунок кількості неунікальних слів у петиції. Ефективною візуалізацією такої змінної є гістограма.

Розподіл кількості слів у назвах петиції можна побачити на рис. 3.2. Медіанна кількість слів у назві петиції становить 10 слів, а мода — 7 слів. Ці назви зазвичай закликають до дії такими словами як: «Змінити», «Створити», «Зберегти» та «Допомогти». Більше того, оскільки у період дослідження назви вулиць міста часто змінювались, особливо у зв’язку з набуттям чинності «закону про декомунізацію», майже 4% назв петицій містять слово «Перейменувати».

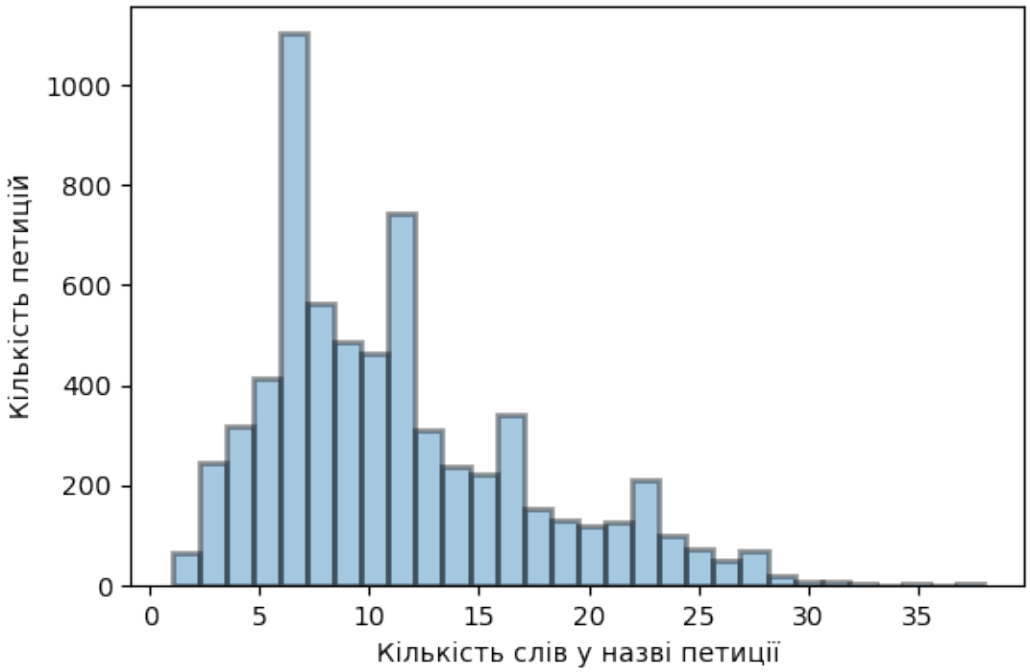

Рис. 3.2 – Розподіл кількості слів у назвах петицій

Розподіл кількості слів у змісті петиції можна побачити на рис. 3.3. Медіанна кількість слів у назві петиції становить 72 слова, а мода — 47 слів. Даний розподіл має довгий хвіст, тому що деякі петиції надзвичайно багатослівні та детальні.

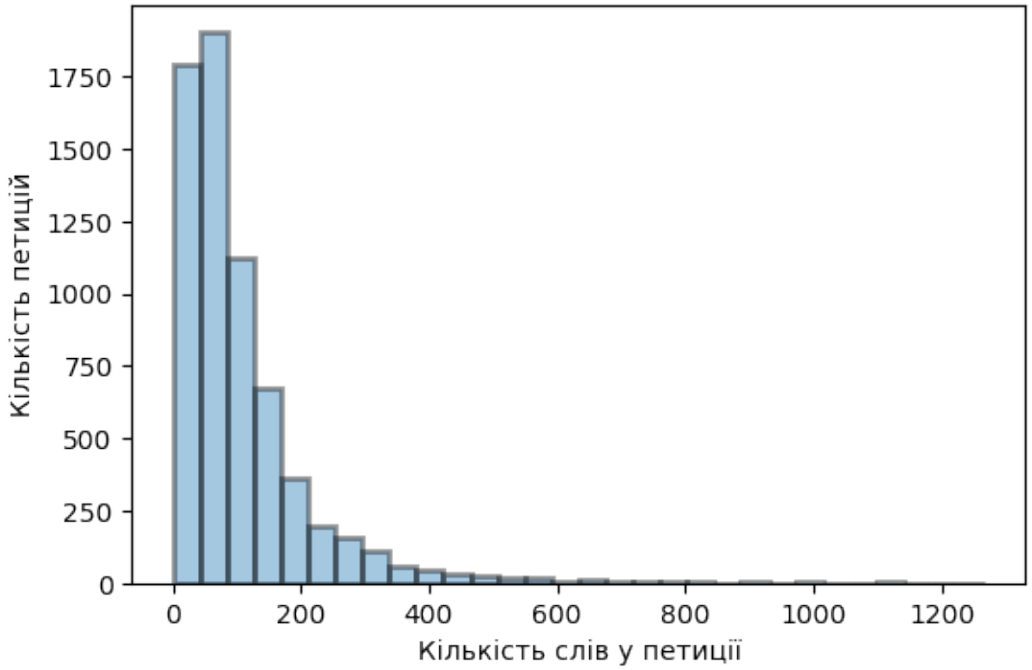

Рис. 3.3 – Розподіл кількості слів у назвах петицій

Після переводу усіх петицій в нижній регістр та видалення стоп-слів (слова, що зустрічаються дуже часто та рідко несуть велике семантичне навантаження, приклади: «і», «та», «є») розподіл 20 найпоширеніших слів серед петицій до Київради наведено в таблиці 3.1.

Таблиця 3.1 – Найпоширеніші слова серед петицій до Київради

Слово	Кількість	Відсоток, %
міста	2228	0.45
києва	2005	0.4
вул	1724	0.34
україни	1619	0.32
метро	1214	0.24
вулиці	1179	0.23
людей	1025	0.2

києві	1005	0.2
транспорту	970	0.19
руху	878	0.17
має	878	0.17
пропоную	769	0.15
прошу	730	0.14
кількість	707	0.14
столиці	680	0.13
біля	636	0.12
міської	633	0.12
території	628	0.12
київ	593	0.11
киян	577	0.11

Видно, що найчастіше зустрічаються слова, які або ідентифікують об’єкти містобудування («місто», «вулиця», «столиця»), або вживаються перед пропозицією дій («прошу», «пропоную»), або мають відношення до транспорту («метро», «транспорт», «руху»). Слово «вулиця» є дуже поширеним у змісті петиції не лише через згадану вище причину перейменування, а й тому, що це посилання на предмет дій, які пропонуються петицією.

Хвіст розподілу здебільшого складається з орфографічних помилок, унікальних сутностей (назви вулиць, на які посилаються у тексті петиції), іноземних слів («йорк») та сутностей (“тетріс” – маючи на увазі швидке та масове спорудження хмарочосів у Києві).

Під час аналізу колонки «автор» було підтверджено, що викидів за цією ознакою (людей, які створили аномальну кількість петицій) у зібраному наборі даних немає. Максимальна кількість петицій, яку подала особа, налічує 68 запитів, що становить близько 1% усіх звернень. Крім того, при детальнішому розгляді було виявлено, що всі клопотання цієї особи унікальні.

Незважаючи на те, що більшість документів у наборі даних петицій міста Києва мають відносно малу кількість слів, є також викиди від двох до 860 слів. Серед інших

статистичних ознак набору даних: загальна кількість слів – 368046, кількість унікальних слів – 65374, середня кількість слів на документ – 74, середня довжина документа – 780 символів, середня кількість символів на слово – 7.

Крім аналізу слів, що використовуються у петиціях, варто розглянути статистику структурного аспекту текстів петицій. Вона показана на рис. 3.4 і має наступні ознаки:

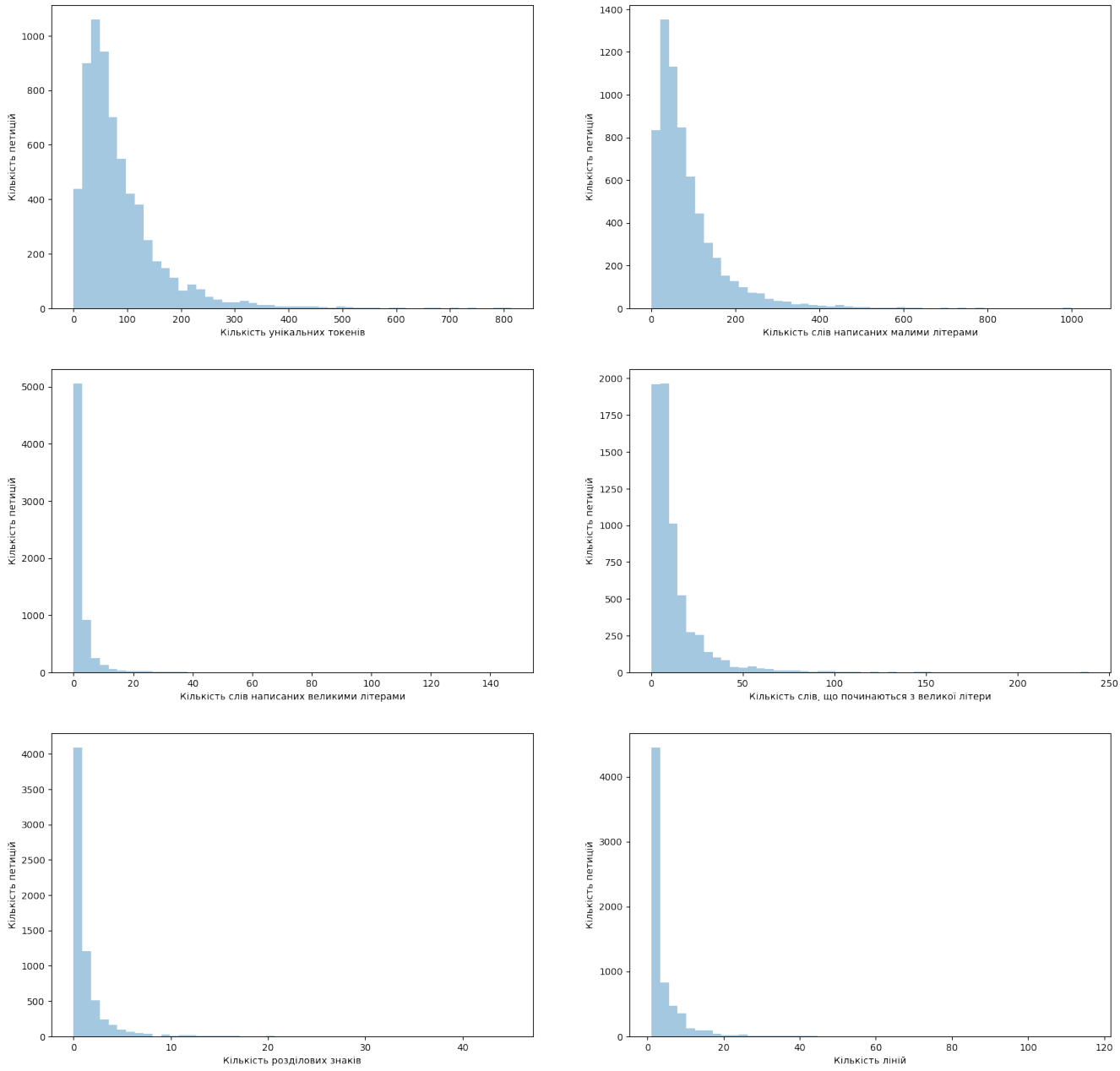

Рис. 3.4 – Розподіли структурних ознак петицій

• медіанна кількість унікальних токенів у вмісті петиції становить 64 з мінімумом 2 і максимумом у 814 унікальних токенів у петиції;

• медіанна кількість неунікальних токенів у нижньому регістрі в петиції становить 62 з мінімальним значенням 0 і максимумом у 1041 неунікальних токенів у нижньому регістрі в петиції;

• медіанна кількість неунікальних токенів у верхньому регістрі в петиції становить 1 з мінімальним значенням 0 і максимумом у 71 неунікальних токенів у верхньому регістрі в петиції;

• медіанна кількість неунікальних токенів, які починаються з великої букви, в петиції становить 7 з мінімальним значенням 0 і максимумом у 239 неунікальних, які починаються з великої букви, в петиції;

• медіанна кількість розділових знаків у змісті петиції дорівнює 0 з мінімумом 0 і максимумом у 45 знаків пунктуації в петиції;

• медіанна кількість рядків у змісті петиції становить 2, при цьому мінімум - 1 і максимум - 94 рядки в петиції.

Важливим результатом аналізу петицій до Київради є те, що у наборі даних присутні петиції написані не тільки українською мовою, а і російською. Тому одним з найважливіших етапів передобробки даних буде класифікація мови петиції та видалення неукраїномовних звернень.

Однією з найпоширеніших проблем у задачах класифікації документів у малоресурсних середовищах є дисбаланс класів. Набір петицій до Київради не є винятком, більше того цей дисбаланс лежить в самій основі прикладної області, оскільки різні аспекти життя в міському середовищі викликають різну кількість дискусій, протестів чи пропозицій громадян. Деякі класи в зібраному наборі даних займають менше 1% усіх зразків кожен, тому ще однією особливістю передобробки набору даних є вилучення їх з дослідження, поки не буде набрано достатньої кількості петицій для моделювання. Остаточний розподіл точок даних між класами показано на рис 3.5.

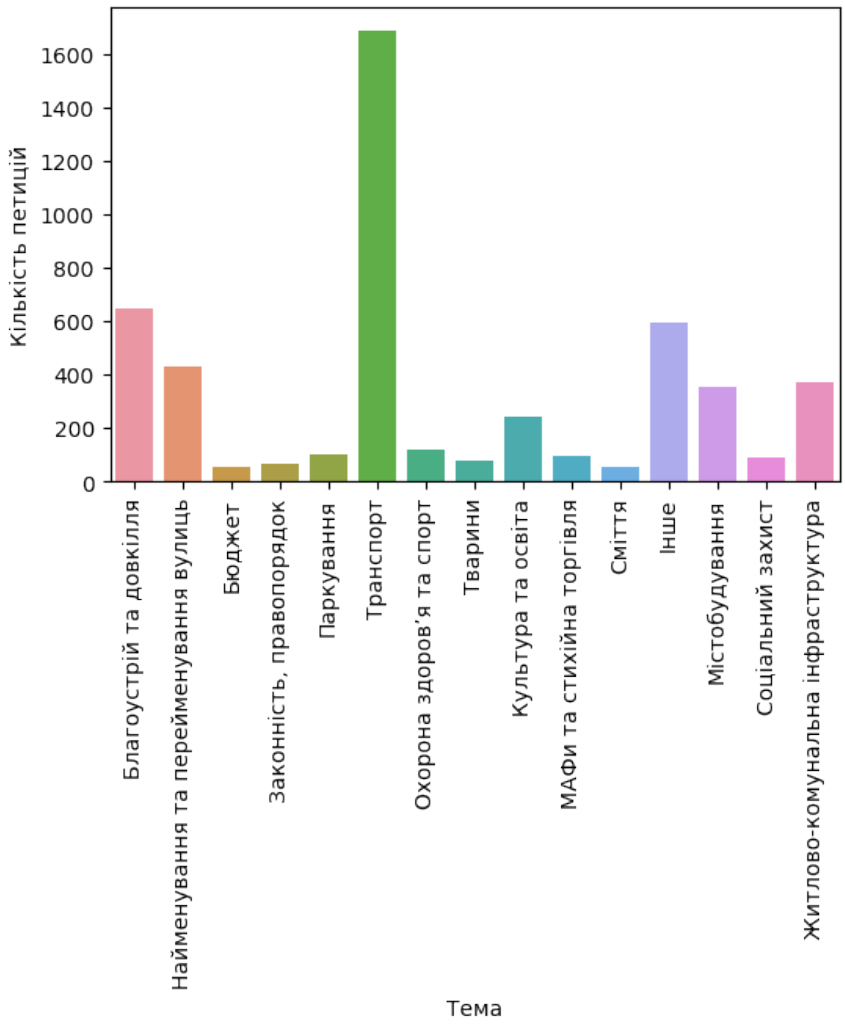

Рис. 3.5 – Розподіл тем у петиціях до Київради

3.1.2 Словник синонімів української мови

Для побудови графа синонімів української мови було використано словник синонімів з Офіційного сайту української мови [167]. Оскільки це онлайн-словник, який містить у собі додаткову інформацію, а також написаний у форматі не сприятливому для автоматичного зчитування, потрібен був певний елемент попередньої обробки, щоб отримати чисті пари синонімів. Ці етапи попередньої обробки складаються з видалення прикладів речень, контексту та скорочень, а також дублікатів пар слово-синонім. Після цього було створено порожній неорієнтований незважений граф, тобто заповнено його словами як вузлами, і якщо два слова є

синонімами, відповідні вузли було з’єднано ребром. Статистичні ознаки графа синонімів наведена в таблиці 3.2, а розподіл степенів його вузлів наведено на рис. 3.6.

Таблиця 3.2 – Статистичні ознаки графа синонімів

Ознака	Значення
Кількість вузлів	44664
Кількість ребер	391047
Середня степінь вершини	8.755
Кількість з’єднаних компонентів	545
Максимальна кліка	44
Середня кластеризація	0.797

З аналізу властивостей графіка можна зробити висновок, що синонімія в українській мові може бути дуже глибокою і існує багато альтернатив щодо того, як одна думка може бути вираженою, що ще більше посилює припущення, що доповнення базових моделей інформацією зі словника синонімів може бути корисним для покращення якості класифікації документів написаних українською.

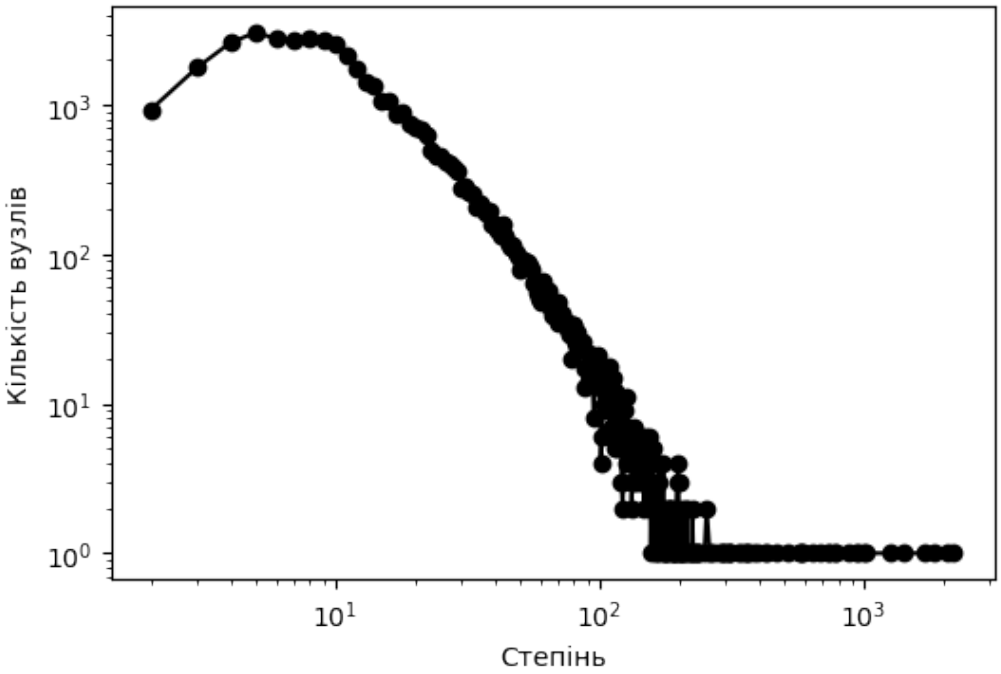

Рис. 3.6 – Розподіл степенів вузлів графу синонімів

3.2 Передобробка експериментальних даних

Отже, на основі проведеного дослідницького аналізу експериментальних даних було виділено набір кроків для їх передобробки. Для набору даних електронних петицій до Київради першим таким кроком є усунення не україномовних петицій.

За даними дослідження 2016 року, лише 69% українців вдома користуються українською мовою [168]. Серед інших мов, поширених в Україні, найбільшу роль відіграє російська. Це створює проблеми для аналізу будь-яких текстових даних, створених українцями. Набір електронних петицій до Київради не є винятком. Отже, для автоматичної роботи з цими даними необхідно вирішити задачу виявлення або класифікації мови, перш ніж переходити до подальшого моделювання. Для вирішення даної задачі та аналізу якості результатів, було вручну розмічено 5725 петицій мітками, що відповідають мові петиції. Далі було досліджено кілька найпопулярніших інструментів класифікації мови.

Щоб показати різницю між інструментами, ми повідомляємо їх матриці невідповіностей, влучність, повноту та F1-міру.

У статистиці та машинному навчанні матриця невідповідності (англ. Confusion matrix, CM) — це таблиця, яка дозволяє візуалізувати похибки моделі. Кожен рядок матриці показує істинні класи (кількість екземплярів, що належать до істинного класу), а кожен стовпець представляє результат передбачення/класифікації (кількість екземплярів у прогнозованому класі).

Влучність (позитивне прогнозне значення) — це частка правильно передбачених точок даних серед усіх передбачень для певного класу.

Повнота (чутливість) — це частка правильно передбачених точок даних серед усіх істинних точок даних певного класу.

F1-міра – це середнє гармонійне значення влучності та повноти. Його максимальне значення дорівнює 1, коли і влучність, і повнота дорівнюють 1, а мінімальне значення — 0, коли модель не присвоїла жодної точки даних правильному класу.

Усі ці три метрики активно використовуються в машинному навчанні та науці про дані, оскільки вони показують більш значущі результати, ніж точність, коли набір даних незбалансований (як це є у випадку набору петицій).

Першим інструментом, який ми оцінили для автоматичного розпізнавання української та російської мов у нашому наборі даних, є langdetect. langdetect є прямим портом бібліотеки Google для визначення мови з Java на Python. Оригінальна бібліотека визначення мови створює мовні профілі з xml анотацій Вікіпедії та виявляє мову тексту за допомогою наївного байєсового фільтра. Автори інструменту стверджують, що він досягає точності у 99% для 53 мов [169]. Матрицю невідповідностей для langdetect у наборі електронних петицій можна побачити на рис. 3.7. Умовні позначки назв класів на рисунку мають наступну відповідність: «uk» – українська мова, «ru» – російська мова, «mk» – македонська мова, «bg» – болгарська мова.

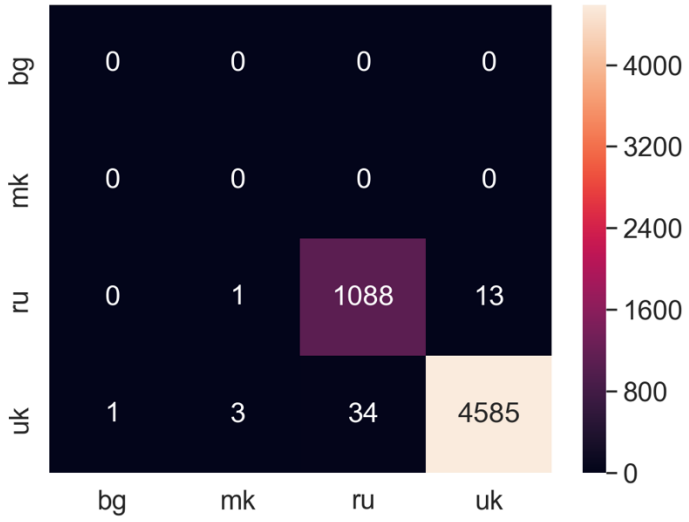

Рис. 3.7 – Матриця невідповідностей класифікації мови петицій за допомогою langdetect

Другим інструментом, який використовувався для класифікації мови електронних петицій міста Києва, є модель FastText для ідентифікації мови, яка здатна

розпізнавати 176 мов. Вона була натренована на основі даних з Wikipedia, Tatoeba та SETimes [62].

Модель є модифікацією популярного алгоритму CBOW з лінійним класифікатором поверх векторів слів, оптимізована для швидкості роботи за допомогою ієрархічного софтмаксу та n-грам з хешуванням [170]. Матриця невідповідностей цього підходу в наборі даних електронних петицій міста Києва показана на рис. 3.8.

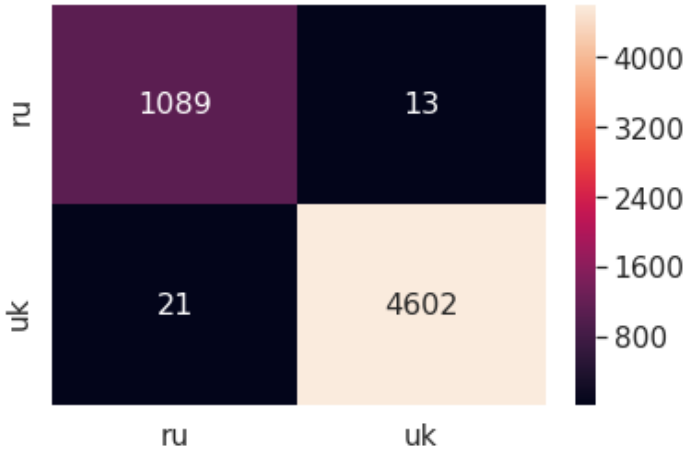

Рис. 3.8 – Матриця невідповідностей класифікації мови петицій за допомогою FastText

Результати оцінки та порівняння ефективності роботи даних моделей можна побачити в таблиці 3.3.

Таблиця 3.3 – Порівняння ефективності класифікації мови langdetect та FastText

Інструмент (мова)	Оцінка			
	Влучність	Повнота	F1-міра	Кількість істинних точок
langdetect (українська)	1.0	0.99	0.99	4623
fasttext (українська)	1.0	1.0	1.0	4623
langdetect (російська)	0.97	0.99	0.98	1102
fasttext (російська)	0.98	0.99	0.98	1102

На основі даного аналізу, можна зробити висновок, що для цього конкретного набору даних fasttext працює краще, ніж langdetect, і показує майже ідеальні результати обома мовами. Крім того, з матриць невідповідностей видно, що langdetect помиляється на деяких петиціях у сторону мов, які не присутні у даній прикладній області, таких як болгарська та македонська, тоді як FastText – ні.

Наслідком очищення петицій написаних не українською мовою стало зменшення набору даних до 5332 текстів. Тобто мову 1228 петицій (19% оригінального набору даних) було класифіковано як російську та, задля того, щоб на експерименти не впливали іншомовні особливості, видалено.

Після етапу видалення не україномовних петицій усі тексти було переведено у нижній регістр. Даний крок передобробки зумовлений тим, що багато слів у петиціях перебувають у різних регістрах, хоч мають один і той же зміст. Це, в свою чергу, виникає з декількох причин. Першою є, типова для всіх задач обробки природної мови, капіталізація перших слів у новому реченні, або після певних розділових знаків. Таким чином, без переводу у нижній регістр слово «транспорт» у середині речення та слово «Транспорт» на початку речення були двома окремими екземплярами у словнику і мали б різні представлення. Ця проблема виражається ще більше у малоресурсних середовищах, де частота слів з великої букви на початку речень зовсім низька, адже речень не так багато. Відповідно отримані представлення, які спираються на контекст не будуть узгоджені між версіями слова у нижньому регістрі та верхньому регістрі. Другою причиною є те, що при написанні петицій, автори часто ігнорують перевірку правильності вибору регістру або навмисне його змінюють (як знак протесту), особливо для власних назв, тому в наборі даних можна знайти одні і ті ж слова, незалежно від позиції у реченні, в різних регістрах. Наприклад, якщо не робити переведення до нижнього регістру, то слова «Київ» та «київ» будуть знову ж мати різні представлення, хоч відносяться до однієї сутності. Окремим завданням, яке можна вирішувати в майбутньому, це побудова моделі істинного регістру (англ.

Truecasing) української мови, яка дозволила б виправляти помилки та приводити текст до правильного регістру.

Наступним кроком передобробки текстів електронних петицій до Київради який пропонується у роботі є видалення стоп-слів. Стоп-слова – це часті, але переважно невідповідні для завдання класифікації лексеми, які зберігають у окремому словнику (іноді названий негативним) для подальшого видалення з джерел текстової інформації. Зазвичай це найпоширеніші слова у будь-якій мові (наприклад, артиклі, прийменники, займенники, сполучники тощо) і не додають багато інформації до текстів. Таким чином їх видалення дозволяє статистичним моделям та моделям, на основі машинного навчання, більше фокусуватись на важливих словах. Єдиного списку стоп-слів для певної мови не існує, адже цей список може відрізнятись від завдання до завдання та від прикладної області до прикладної області. Наприклад, слово «наказую» не є надзвичайно частим при аналізі текстів українською мовою різного спрямування і несе багато інформації, тоді як в контексті аналізу наказів Кабінету Міністрів України, де в кожному прикладі присутнє дане слово, воно може бути додане до негативного словника. Додатковими перевагами видалення стоп-слів є зменшення набору даних та розміру словника при роботі з великими даними, а отже і швидше навчання ітеративних алгоритмів машинного навчання. У даній роботі видалення стоп-слів було проведено з метою обмеження їх впливу під час усереднення та злиття векторних представлень слів, а також зменшення розміру словника, а отже оперативної пам’яті, потрібної для роботи з ними. Для цього було використано негативний словник наявний у бібліотеці stop-words [171] для мови програмування Python. Дана бібліотека налічує списки стоп-слів для 22 мов світу, включаючи українську, російську, англійську, німецьку, французьку та інші. Для української мови у бібліотека налічує 385 стоп-слів, типовими представниками яких є «твій», «нас», «ту», «там», «цього» та інші. Так як не всі петиції написані ідеальною українською мовою, більше того частина з них має русизми або окремі речення

написані російською, на основі аналізу частоти слів у петиціях до даного набору стоп-слів було додано «и», «а» та «с».

Останнім етапом передобробки петицій є видалення або заміна зайвих символів та нормалізація тексту. До таких символів відносяться усі невидимі символи, які можна замінити на стандартні відповідники – пробіл чи знак нової лінії, та символи не представлені у стандарті Unicode. Додатково, послідовні повторення пробілів чи знаків нової лінії були замінені на одноразові, так як це спрощує процес поділу петиції на окремі токени.

3.3 Аналіз отриманих векторних просторів

3.3.1 Аналіз векторних просторів слів

На основі методології, описаної у розділі 2, було натреновано два набори векторів слів: Word2Vec і FastText. Обидва методи отримували однакові вхідні дані, а саме набір передоброблених петицій. На даному етапі словник синонімів участь не приймає.

Додатково при побудові векторних просторів слів було видалено нечасті токени. Кожен токен, який зустрівся у наборі даних менше 20 разів, було проігноровано при побудові моделей. Це робиться для того, щоб прибрати нерелевантні токени, а також орфографічні помилки, які могли виникнути на етапі написання петицій або їх токенізації. Це зробило тренувальний процес більш стабільним.

Навчання обох векторних моделей слів зайняло близько 30 хвилин із 4 процесами на процесорі Intel Core i5 2,8 ГГц. Графічний процесор у даному випадку не потрібен, оскільки розмір набору даних малий. Для тренування було використано бібліотеку Gensim [172], яка дозволяє будувати векторні моделі слів з простим у використанні, але потужним програмним інтерфейсом мовою програмування Python.

3.3.1.1 Простір слів Word2Vec

У процесі експериментів було обрано оптимальні гіперпараметри на структуру моделі Word2Vec для побудови векторів слів петицій. Ними виявились:

- Розмірність векторів – 100;
- Кількість епох – 1000;
- Контекстне вікно – 5;
- Розмір кроку – 0.025;
- Метод оптимізації – Adam;
- Модель – пропуск-грами;
- Алгоритм тренування – негативний вибір.

Таблиця 3.4 показує приклади найближчих у натренованому просторі слів Word2Vec до запитів та їх косинусну схожість.

Таблиця 3.4 – 10 найближчих до запитуваних слів у просторі слів Word2Vec

№	Запит: «Кличко»	Схожість	Запит: «КМДА»	Схожість
1	Віталій	0.747	Київради	0.614
2	мер	0.586	депутатів	0.532
3	голова	0.458	КМР	0.518
4	голови	0.417	сайті	0.495
5	Київської	0.374	рішення	0.460
6	вимогою	0.368	ради	0.452
7	разом	0.356	відповідних	0.446
8	комісії	0.353	РДА	0.445
9	міський	0.353	КП	0.439
10	своїм	0.347	розпорядження	0.437

Більш детальний аналіз побудованого векторного простору показує, що векторні представлення слів, що зустрічаються у петиціях, побудовані зі допомогою методу Word2Vec проявляють очікувані ознаки семантичного зв’язку слів з векторами, що мають високу косинусну схожість. Отже, вони можуть бути використані для подальших експериментів у побудові векторних просторів електронних петицій.

3.3.1.2 Простір слів FastText

У процесі експериментів було обрано оптимальні гіперпараметри на структуру моделі FastText для побудови векторів слів петицій. Ними виявились:

- Розмірність векторів – 100;
- Кількість епох – 1000;
- Контекстне вікно – 5;
- Розмір кроку – 0.01;
- Метод оптимізації – Adam;
- Модель – пропуск-грами;
- Алгоритм тренування – негативний вибір;
- Мінімальна довжина символьних n-грам – 3;
- Максимальна довжина символьних n-грам – 6;

Таблиця 3.5 показує приклади найближчих у натренованому просторі слів FastText до запитів та їх косинусну схожість.

Таблиця 3.5 – 10 найближчих до запитуваних слів у просторі слів FastText

№	Запит: «Кличко»	Схожість	Запит: «КМДА»	Схожість
1	Віталій	0.724	Київради	0.577
2	голова	0.571	КМР	0.567
3	мер	0.530	депутатів	0.501
4	голови	0.478	сайті	0.486
5	Київської	0.400	рішення	0.468
6	комісії	0.394	РДА	0.464
7	разом	0.392	ради	0.446
8	КМДА	0.373	КП	0.446
9	Шановний	0.371	питання	0.445
10	Віталію	0.365	відповідних	0.430

Оскільки FastText є модифікацією Word2Vec, результати запитів - подібні, обидва методи фіксують семантичні відношення між словами, такі як ім’я та посада, або подібні установи до запитуваних слів. Однак, як згадувалося раніше, FastText

дозволяє робити запити для слів, яких немає в наборі даних для тренування, що є значною перевагою у багатьох прикладних застосуваннях векторних моделей слів. Аналогічно Word2Vec представлення слів, FastText векторний простір слів охоплює семантичні зв'язки між словами у контексті електронних петицій, але додатково фокусується на морфологічні ознаки слів, а отже і косинусова схожість між векторами слів у даному просторі це відображає.

3.3.2 Аналіз векторних просторів петицій

Для того, щоб надати якісні ознаки векторів петицій, отриманих шляхом усереднення векторів слів для вмісту петиції, у даному підрозділі наведені їх візуалізації. Такі візуалізації будуються шляхом зменшення розмірності векторів петицій з 100-вимірних до 3-вимірних і графічного зображення цього стиснутого простору. Алгоритм зменшення розмірності, який було використано, — це апроксимація та проекція рівномірного многовиду (Uniform Manifold Approximation and Projection, UMAP) [173], який працює швидше, а також краще зберігає глобальну структуру даних, ніж інші алгоритми зменшення розмірності. Ідея UMAP полягає в тому, щоб спочатку створити високовимірне графове представлення векторного простору, а потім ітеративно оптимізувати побудову нового низьковимірного графового представлення, щоб воно було структурно максимально схоже на оригінальне. Для швидкої побудови візуалізації векторних просторів зручно використати Tensorboard [174] – інструментарій візуалізації та аналізу експериментів в області машинного навчання. Tensorboard дозволяє проектувати високовимірні векторні простори в низьковимірні за допомогою UMAP, t-розподіленого представлення стохастичної близькості (t-dictributed stochastic neighbor embedding, t-SNE) або методу головних компонент (principal component analysis, PCA). Серед цих трьох опцій UMAP було обрано, оскільки він швидший, ніж t-SNE, і більш виразний, ніж PCA [175]. Після візуалізації побудовані 3-вимірні простори було досліджено вручну.

Для кількісної оцінки відмінностей у досліджуваних векторних просторах петицій далі наведено їх коефіцієнти Силуетта [176]. Дана оцінка визначається як середнє значення коефіцієнтів Силуетта для кожної точки. У свою чергу коефіцієнт Силуетта для окремої точки визначається як:

$$S = \frac{a - b}{\max(a, b)},$$

де a – середня відстань між точкою та всіма іншими точками в тому самому кластері, b – середня відстань між точкою та всіма іншими точками найближчого кластера. Для використання даної оцінки потрібно прокластеризувати досліджуваний простір. У даній роботі для цього використовується алгоритм кластеризації DBSCAN [177], адже він не потребує ручного підбору кількості кластерів, не спирається на припущення, що точки мають нормальний розподіл та є стійким до аномальних точок. Очікується, що модель з кращими кластерами покаже вищий коефіцієнт Силуетта. Коефіцієнт має діапазон від -1 до 1, при цьому моделі з високою щільністю кластерів мають оцінки ближче до 1, а з кластерами, що сильно перекриваються, ближче до 0. Таблиця 3.6 показує оцінки коефіцієнта Silhouette для векторів петицій на основі Word2Vec і FastText, прокластеризованих за допомогою DBSCAN.

Таблиця 3.6 – Коефіцієнти Силуетта для векторів документів

Модель	Коефіцієнт Силуетта
На основі Word2Vec	0.468
На основі FastText	0.004

Як зазначалося раніше, для побудови моделей векторного простору для петицій міста Києва ми використовували усереднення векторів слів, присутніх у петиції. На рис. 3.9 зображено візуалізацію векторних просторів петицій на основі Word2Vec і FastText після зменшення розмірності за допомогою UMAP.

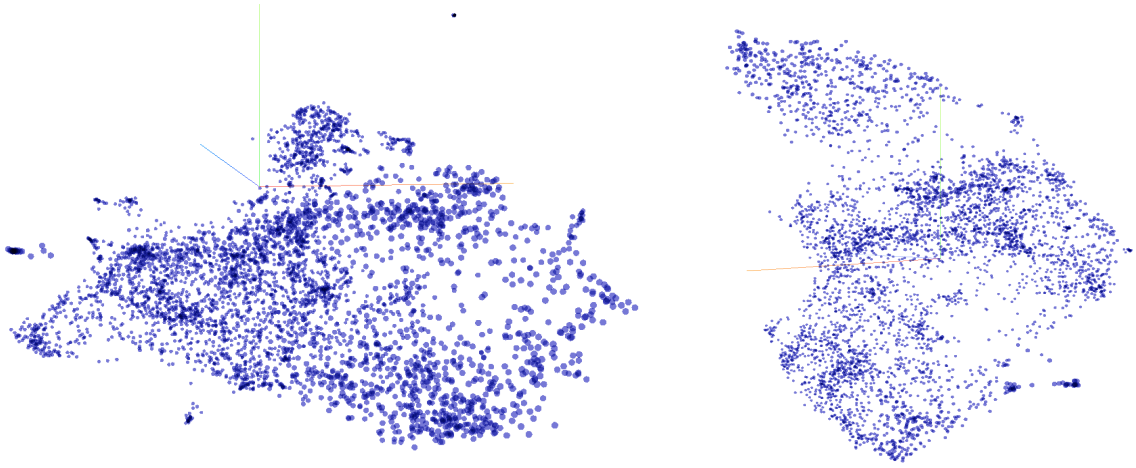

Рис. 3.9 – Візуалізація за допомогою UMAP векторних просторів петицій на основі Word2Vec (зліва) і FastText (справа)

Обидва простори демонструють структуру з багатьма кластерами, при цьому петиції, що мають різну семантику знаходяться в різних частинах простору. Модель петицій на основі Word2Vec має помітно більше розділених кластерів, ніж модель на основі FastText. Це підтверджується перерахованими раніше показниками коефіцієнта Силуетта, де модель петицій на базі Word2Vec мала набагато вищу оцінку, що означає, що кластери, присутні в її векторному просторі, є більш щільними, тоді як у моделі на основі FastText вони можуть значно перекриватися.

При детальнішому розгляді видно, що ці кластери справді семантично розділені, і кілька чітко визначених груп точок демонструють схожість у темах, які вони обговорюють. У наступному розділі обговорюється те, чим відрізняються дві побудовані моделі.

3.3.2.1 Простір петицій на основі Word2Vec

Модель на основі Word2Vec має чітко видимі кластери, які мають певне семантичне значення. Два приклади цієї поведінки зображено на рис. 3.10.

Рис. 3.10 – Візуалізація найближчих петицій на основі Word2Vec до запитів «екологічна ситуація» (зліва) та «водопостачання гарячої води» (справа)

Для демонстрації роботи моделі у таблиці 3.7 наводиться кілька запитів, для яких знаходяться найближчі петиції у наборі даних за косинусною відстанню. Наприклад, результатом запиту, який стосується екологічної ситуації в Києві, є кілька петицій про погану переробку відходів та забруднення озер, а запит про водопостачання гарячої води здебільшого повертає скарги на подібні ситуації до Київської міської ради. Загалом, якість векторних представлень петицій на основі усереднення векторів слів Word2Vec є задовільною для їх майбутнього використання у контексті трансферного навчання як вхідних даних до класифікатора документів.

3.1.1.1 Простір петицій на основі FastText

Модель на основі FastText, демонструє подібні семантичні властивості до моделі на основі Word2Vec, зрештою, їхні основні ідеї близькі. Однак загальна кількість видимих кластерів зменшується, і деякі з них чітко групуються на синтаксичному рівні замість бажаного семантичного рівня. Це можна побачити на рис. 3.11, де зліва зображений семантичний кластер, тоді як справа кластер подібних через однаковий стандартний початок точок.

Таблиця 3.7 – 3 найближчі петиції у просторі Word2Vec та їх косинусна схожість до запитів

№	Запит: «Екологічний стан Києва у Дарницькому районі»	Схожість	Запит: «відсутнє постачання гарячої води»	Схожість
1	Вже багато років екологічна ситуація в Дарницькому районі м. Києва, та інших районах міста, відповідно, є катастрофічно-критичною: нестерпний сморід в нічний та ранковий час доби як результат недолугої діяльності сміттєпереробного заводу «Енергія»...	0.69	В Оболонському районі, по вулицях Дубровицька 5, 7, 3 вже майже місяц відсутнє постачання гарячої води. Чіткої відповіді, на питання мешканців цих будинків "коли буде вода" від Київенерго та районного ЖКХ не має...	0.841
2	заборонено купання в 50 озерах і ставках, зокрема Дідоровських і Мишеловських ставках … в Дарницькому районі, ... Купання в цих озерах не рекомендується через незадовільні проби води. Комунальне підприємство «Плесо» заборонило пляжний відпочинок через невідповідність санітарним нормам: за результатами санітарно-мікробіологічних досліджень проб води…	0.651	Не перший рік виникають нарікання на температуру постачання гарячої води. Едина можливість додати холодну воду до гарячої виникае...	0.835
3	Мабуть, кожен, хто проживає у Дарницькому районі міста Києва має можливість «насолоджуватись» ароматом, який «дарує» нам БОРТНИЦЬКА СТАНЦІЯ АЕРАЦІЇ ТА СМІТТЄСПАЛЮВАЛЬНИЙ ЗАВОД "ЕНЕРГІЯ". Та, крім неприємного запаху, підприємства також несуть загрозу здоров'ю КОЖНОГО З НАС...	0.642	Оскільки послуги з опалення та постачання гарячої води надаються монополістами потрібен важіль тиску на постачальників для унеможливлення значного завищення тарифів. Таким засобом впливу стане можливість встановлення індивідуальний систем опалення та підігріву води у будь-яких багатоповерхових будинках.	0.818

У таблиці 3.8 також наводяться два приклади запитів до моделі на основі FastText. Перший запит — це лише назва проспекту, і модель змогла знайти семантично подібні петиції, які здебільшого стосуються процесу перейменування

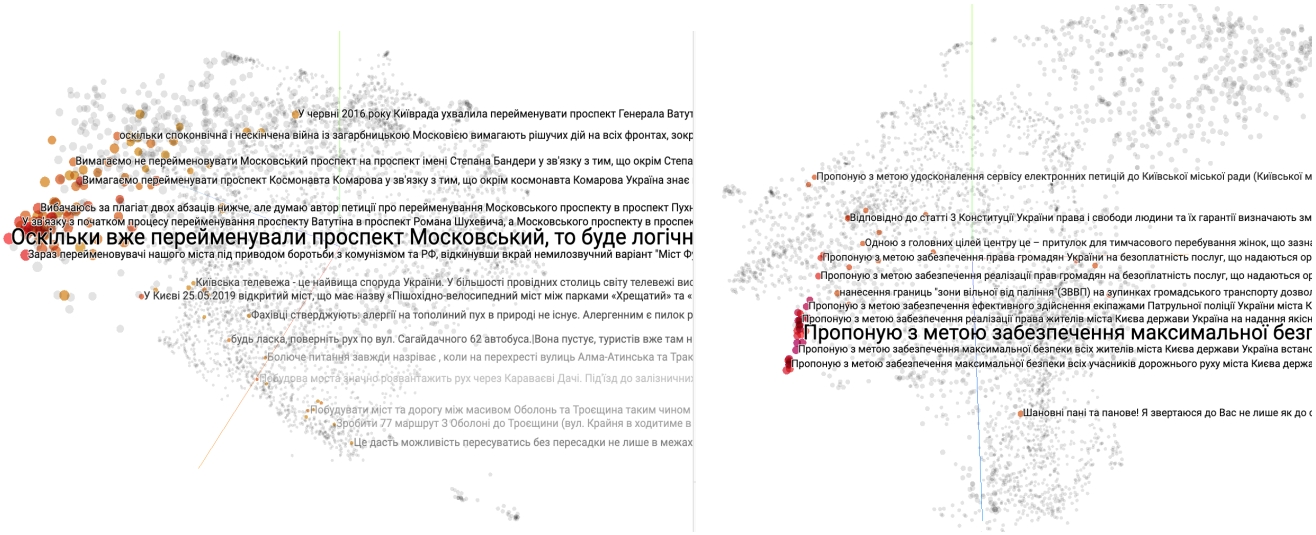

Рис. 3.11 – Візуалізація найближчих петицій на основі FastText до запитів «Московський проспект» (зліва) та «пропоную» (справа)

конкретного (з запиту) або інших проспектів. Другий приклад показує, що модель також може знаходити синтаксично подібні петиції. Загалом, якість цієї моделі менш придатна для використання в подальших семантично значущих завданнях, ніж модель на основі Word2Vec.

3.1.1 Простір синонімів

Для того щоб доповнити векторний простір документів інформацією зі словника синонімів, варто спочатку побудувати векторні представлення вузлів графа словника синонімів української мови. Так як моделюється абсолютно кожен вузол даного графу, фільтрація низькочастотних у природній мові слів як етап передобробки не потрібна. Очікувана якість навчених представлень синонімів полягає в тому, що зв’язані компоненти в графі повинні бути близькі з точки зору обраної функції відстані, тоді як вузли, які існують в окремих підграфах, знаходяться далі один від одного у побудованому просторі. Обраний критерій відстані між векторами – косинусна відстань для відповідності аналізу векторних представлень слів та документів на основі тексту. Більше того, враховуючи особливості подальшого

Таблиця 3.8 – 3 найближчі петиції у просторі FastText та їх косинусна схожість до запитів

№	Запит: «Проспект Московський»	Схожість	Запит: «Пропоную з метою забезпечення»	Схожість
1	У зв'язку з початком процесу перейменування проспекту Ватутіна в проспект Романа Шухевича, а Московського проспекту в проспект Степана Бандери - було б логічно перейменувати Московський міст, що з'єдує ці два проспекти, в Троєщинський міст.	0.74	Пропоную з метою забезпечення ефективного здійснення екіпажами Патрульної поліції України міста Києва держави Україна своїх посадових обов'язків забезпечити кожен з них сертифікованими приладами - лазерними радарами вимірювання швидкості TruCam	0.654
2	Назва Московського проспекту в Оболонському і Московському районах Києва не є історичною. Її було надано 2003 року як дружній політичний жест тодішнього міського голови Києва О.Омельченка щодо його московського колеги Ю.Лужкова...	0.69	Пропоную з метою забезпечення ефективного здійснення екіпажами Патрульної поліції України міста Києва держави Україна своїх посадових обов'язків забезпечити кожен з них сертифікованими приладами - алкотестерами Драгер	0.643
3	Назва "Проспект "Правди"" (Виноградар) підпадає під Закон про декомунізацію. Окрім того, при перейменуванні його на проспект Павла Шеремета буде символічно, що бічним до цього проспекта є проспект Георгія Гонгадзе - також загиблого київського журналіста.	0.651	Просимо заборонити рух великовагового транспорту територією міста в денний час, з метою зменшення руйнування асфальтового покриття та з метою поліпшення організації дорожнього руху і його безпеки, поліпшення екологічного стану та підвищення пропускної спроможності вулично-шляхової мережі м. Києва	0.606

злиття векторів документів та векторів словника синонімів, гіперпараметр розмірності вихідного вектора регулюється розмірністю побудованих у попередній секції векторів документів та був налаштований у 100.

Гіперпараметри алгоритму Node2Vec, які виявились оптимальними при проведенні експериментів:

- Довжина випадкової прогулянки – 5;

- Кількість прогулянок – 100;
- Крок навчання – 0.1;
- Контекстне вікно – 5;
- p (гіперпараметр повернення) – 1;
- q (гіперпараметр виходу) – 1;
- Кількість епох навчання – 5;
- Метод оптимізації – Adam;
- Модель – пропуск-грами;
- Алгоритм тренування – негативний вибір 50 точок.

Час навчання даного методу становив 3 години. Так як оптимальними значеннями гіперпараметрів p та q виявились 1, метод еквівалентний застосуванню DeepWalk.

Оскільки вся концепція побудови представлень вузлів для графа синонімії полягала в тому, щоб отримати щільні представлення вузлів, які кодують синонімію, відстані між синонімічними словами повинні бути значно меншими, ніж між випадковими словами. Побудовані за допомогою Node2Vec вектори вузлів показують цю властивість. Наприклад, косинусна відстань між словами «вулиця» та «проспект» дорівнює 0.121, а відстань між словами «вулиця» та «автомобіль» — 0.735. У результаті якісного аналізу результуючих векторів вузлів можна помітити, що ці вектори мають вищі значення дистанцій, а отже є більш відмінними, ніж вектори слів Word2Vec на основі петицій, що має сенс, оскільки вони були навчені на добре визначеній за своєю природою графовій структурі замість реальних текстів, де синонімія може бути відстежена лише через спільні контексти, які дуже обмежені в малоресурсних середовищах.

Представлення документів на основі представлень вузлів графа синонімів також демонструють багатообіцяючу поведінку з коефіцієнтом Силуетта рівним 0.863. Це вказує на те, що векторний простір документів на основі графа синонімів може бути ефективно прокластризованим. Більше того, оскільки цей показник значно

відрізняється від показника векторного простору петицій на основі Word2Vec/FastText, це може бути свідченням того, що їх поєднання може покращити результати класифікації документів.

3.2 Порівняльний аналіз методів класифікації петицій

Після того, як було побудовано векторні простори документів та векторні простори графу синонімів можна переходити до їх злиття та використання в якості ознак для класифікації документів. В якості методу класифікації було обрано просту нейронну мережу – багатошаровий персептрон, адже він дозволяє робити прогноз на основі нелінійних поєднань вхідних ознак. Враховуючи щільність векторного простору, а також різну природу побудованих представлень, дана його особливість робить його кращим інших методів. Інші нелінійні методи класифікації, наприклад методи на основі дерев, працюють гірше з такими щільними представленнями та були виключені з дослідження [178].

Для перевірки ефективності запропонованих методів та їх порівняння з іншими методами класифікації документів при обробці природних мов у малоресурсних середовищах було розділено повний передоброблений набір петицій до Київради на дві підмножини: набір даних для навчання (75% вибірки) та набір даних для тестування (25% вибірки). Набір даних для тестування використовується для обчислення усіх метрик описаних у даному розділі. Враховуючи дисбаланс класів, наявний у оригінальному наборі даних, було вирішено використати стратифікований розділ, так щоб розподіл класів як у навчальному, так і в наборі даних для тестування були подібними. З тієї ж причини основною метрикою, на яку орієнтується дане дослідження є зважена F1-міра, а не точність, яка чутлива до дисбалансу залежних змінних. Зважена F1-міра відрізняється від звичайної тим, що замість використання середнього як функції агрегації F1-мір окремих класів, вона використовує зважене на основі частоти попадання у набір даних для тестування середнє.

Далі порівнюються запропонований підхід у різних варіаціях імплементації з іншими методами класифікації та обробки природної мови у малоресурсному

середовищі. Також для визначення ефекту розроблених методів надаються результати базових методів без урахування малоресурсності середовища. До них відносяться: представлення документів за допомогою мішка слів, TF-IDF, усереднення Word2Vec векторів слів та усереднення FastText векторів слів. Наступним класом методів, які порівнюються з запропонованим є інші методи обробки природної мови у малоресурсному середовищі. Малоресурсність середовища у даному випадку накладає обмеження на методи, які можна застосувати. Так, для методів міжмовної проекції та багатомовних моделей мов відсутні багатомовні набори даних для тренування з українською мовою; для методів віддаленого нагляду та адаптації домену моделі мови у прикладній області відсутні нерозмічені дані, які можна було б розмітити псевдо-розміткою чи адаптувати; метанавчання та змагальні дискримінатори вимагають глибокі нейромережеві підходи, які не працюють за такої малої кількості даних. Тому інші методи обробки природної мови у малоресурсних середовищах, які можна використати у даній ситуації є методи доповнення даних. Враховуючи наявність словника синонімів, даний підхід полягає в тому, що до набору даних петицій додаються дублікати, але з заміненими на синоніми словами. Це надає два потенційних джерела покращення моделі – (1) розширюється розмір даних для навчання, (2) з’являється зв’язок між петиціями через схожі контексти замінених слів-синонімів.

Для обмеження потенційного впливу випадковості ініціалізації параметрів чи гіперпараметрів алгоритму класифікації документів на результуючі метрики було виконано пошук гіперпараметрів для кожного з порівнюваних методів. Отримані метрики наведені на рис. 3.12, 3.13 в таблиці 3.9. З неї видно, що запропонований метод злиття векторних представлень документів на основі векторних представлень слів та вузлів графа здатен покращити якість класифікації документів порівняно з аналогічними методами на 2-3%. При цьому злиття на основі конкатенації стабільно показує нижчі результати, ніж злиття на основі зваженої суми за умови однакових векторів. Серед різних методів векторного представлення словника синонімів

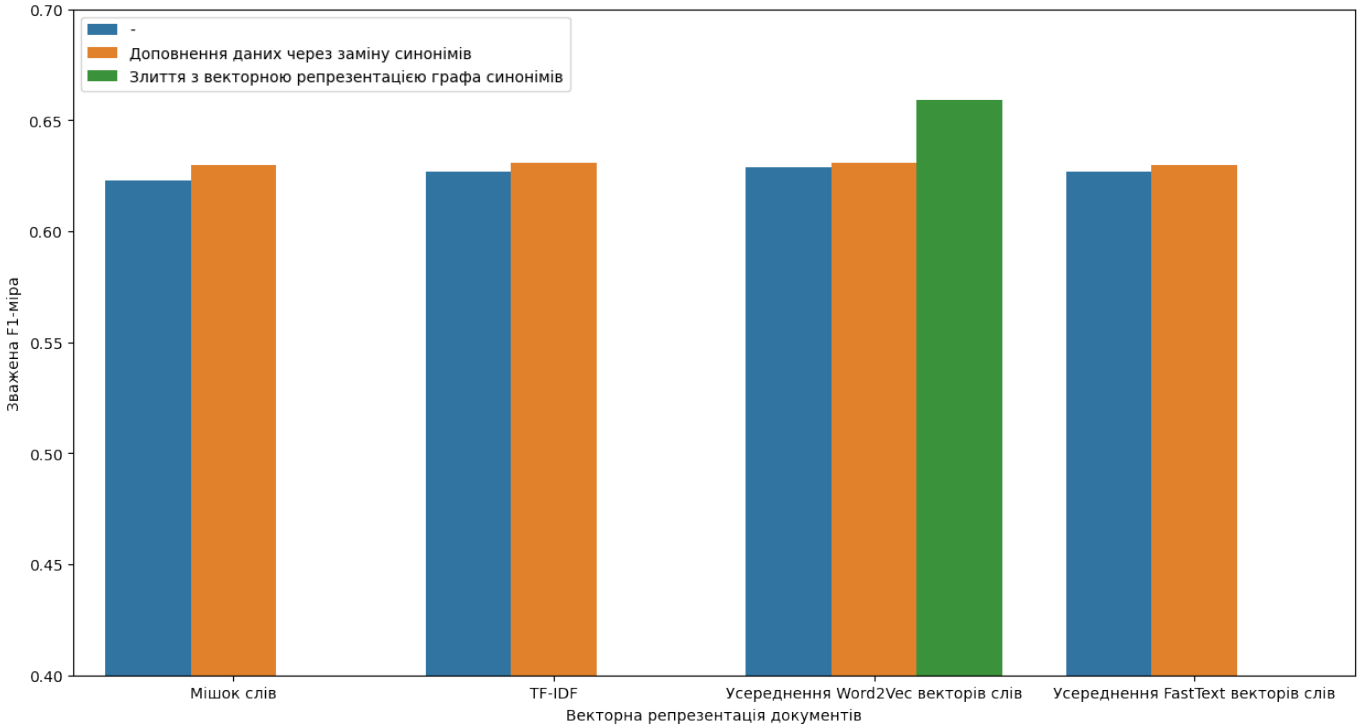

Рис. 3.12 – Порівняння зваженої F1-міри запропонованого методу та інших методів обробки природної мови у малоресурсних середовищах

найкраще показав себе метод Node2Vec, який працює на основі випадкових блукань. Його аналог у категорії методів на основі випадкових блукань – Walklets – спрацював менш якісно, але все ще краще ніж методи інших категорій. Алгоритми побудови векторного представлення графів на основі факторизації мають велику часову складність, а отже при словнику великих розмірів можуть бути непрактичними. Відтак, у даному дослідженні при розмірі графу описаному у підрозділі 3.1.2 методи HOPE, LLE, Лапласівські проекції та GraRep не змогли завершити своє виконання на кроці побудови векторних представлень вузлів графу. При цьому методи HOPE, LLE та Лапласівські проекції мають часову складність пропорційну кількості ребер, яких у графі синонімів значно більше, ніж вузлів. І хоч складність GraRep залежить від кількості вузлів, дана залежність – кубічна, а отже його робота повільна у великих графах. З методів побудови векторних представлень графу на основі факторизації в даному середовищі можна застосувати факторизацію графу (2.3.1.1), адже її часова складність менша ніж аналогічних методів. Даний метод при злитті з векторним

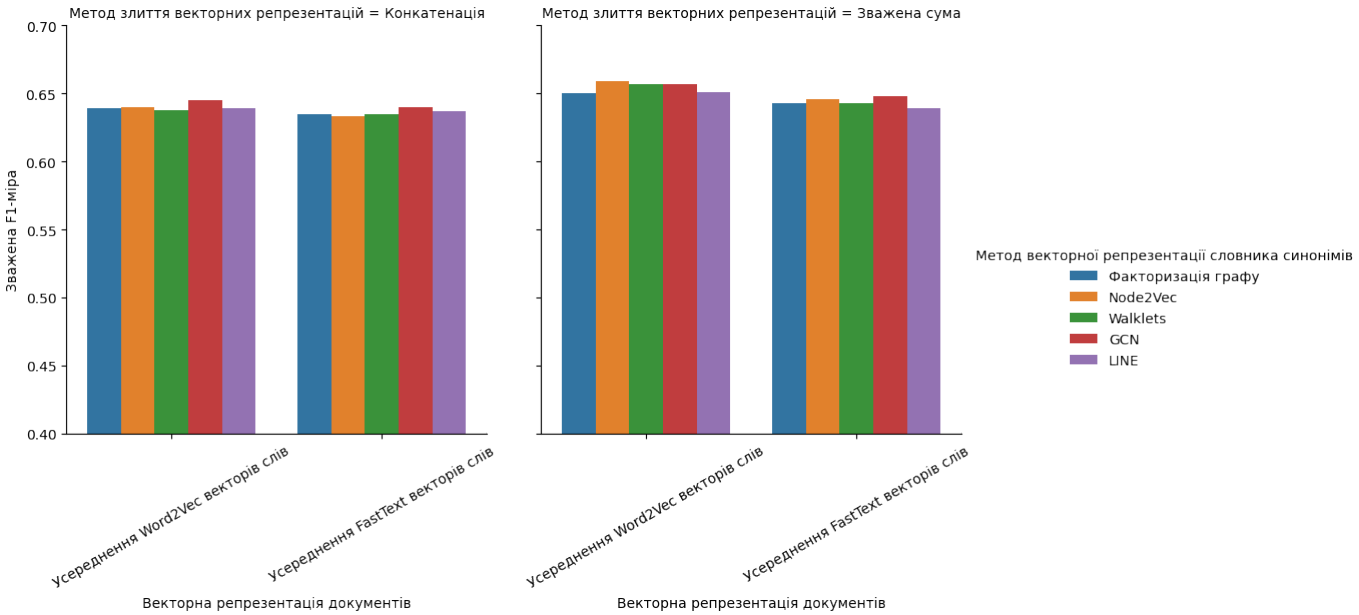

Рис. 3.13 – Порівняння зваженої F1-міри різних способів реалізації запропонованого методу

представленням документів зміг додати близько 1% F1-мірі. Серед методів на основі глибокого навчання для експериментів було обрано графові згорткові мережі, адже він є узагальненням та покращенням альтернатив у своїй категорії. Він не зміг показати високих результатів при злитті представлень. Можливе пояснення цьому – те, що граф синонімів високо кластеризований та має набір не глибоких сильнозв'язних груп, а отже при його моделюванні важливіше розуміти локальну, аніж глобальну, структуру. Останнім методом, що порівнювався є LINE. Через те, що це ітераційний алгоритм, який явно фокусується лише на близькостях першого та другого порядків, він зміг показати результат кращий ніж GCN, а також GraRep при роботі зі словником синонімів української мови. При цьому, так як він випадково відбирає зразки зв’язків для моделювання, його швидкодія значно краща, ніж алгоритмів факторизації графу з подібною часовою складністю.

Таблиця 3.9 – Результати класифікації документів

Векторне представлення документів	Метод обробки природної мови у малоресурсному середовищі	Метод векторного представлення словника синонімів	Метод злиття векторних представлень	Зважена F1-міра
Мішок слів	-	-	-	0.623
TF-IDF	-	-	-	0.627
Усереднення Word2Vec векторів слів	-	-	-	0.629
Усереднення FastText векторів слів	-	-	-	0.627
Мішок слів	Доповнення даних через заміну синонімів	-	-	0.630
TF-IDF	Доповнення даних через заміну синонімів	-	-	0.631
Усереднення Word2Vec векторів слів	Доповнення даних через заміну синонімів	-	-	0.631
Усереднення FastText векторів слів	Доповнення даних через заміну синонімів	-	-	0.630
Усереднення Word2Vec векторів слів	Злиття з векторним представленням графа синонімів	Факторизація графу	Конкатенація	0.639
Усереднення Word2Vec векторів слів	Злиття з векторним представленням графа синонімів	Факторизація графу	Зважена сума	0.650
Усереднення FastText векторів слів	Злиття з векторним представленням графа синонімів	Факторизація графу	Конкатенація	0.635

Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів	Факторизація графу	Зважена сума	0.643
Усереднення векторів слів	Word2Vec	Злиття з векторним представленням синонімів	Node2Vec	Конкатенація	0.640
Усереднення векторів слів	Word2Vec	Злиття з векторним представленням синонімів	Node2Vec	Зважена сума	**0.659**
Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів	Node2Vec	Конкатенація	0.633
Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів	Node2Vec	Зважена сума	0.646
Усереднення векторів слів	Word2Vec	Злиття з векторним представленням синонімів	Walklets	Конкатенація	0.638
Усереднення векторів слів	Word2Vec	Злиття з векторним представленням синонімів	Walklets	Зважена сума	0.657
Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів	Walklets	Конкатенація	0.635
Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів	Walklets	Зважена сума	0.643
Усереднення векторів слів	Word2Vec	Злиття з векторним представленням синонімів	GCN	Конкатенація	0.645

Усереднення векторів слів	Word2Vec	Злиття з векторним представленням синонімів графа	GCN	Зважена сума	0.657
Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів графа	GCN	Конкатенація	0.640
Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів графа	GCN	Зважена сума	0.648
Усереднення векторів слів	Word2Vec	Злиття з векторним представленням синонімів графа	LINE	Конкатенація	0.639
Усереднення векторів слів	Word2Vec	Злиття з векторним представленням синонімів графа	LINE	Зважена сума	0.651
Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів графа	LINE	Конкатенація	0.637
Усереднення векторів слів	FastText	Злиття з векторним представленням синонімів графа	LINE	Зважена сума	0.639

Різниця між матрицями невідповідностей базової моделі (усереднення Word2Vec векторів слів, без додаткової інформації) та найкращої отриманої моделі (на основі зваженої суми Word2Vec і Node2Vec) показана на рис. 3.14. Позитивні значення по діагоналі та від'ємні значення на недіагональних позиціях відповідають покращенню класифікації певних класів. Базуючись на розподілі анотацій у наборі даних, можна прийти до висновку, що покращена модель змогла вловити нюанси в низькочастотних класах, тоді ж як класи, що представлені вдосталь, стали прогнозуватись краще лише незначною мірою.

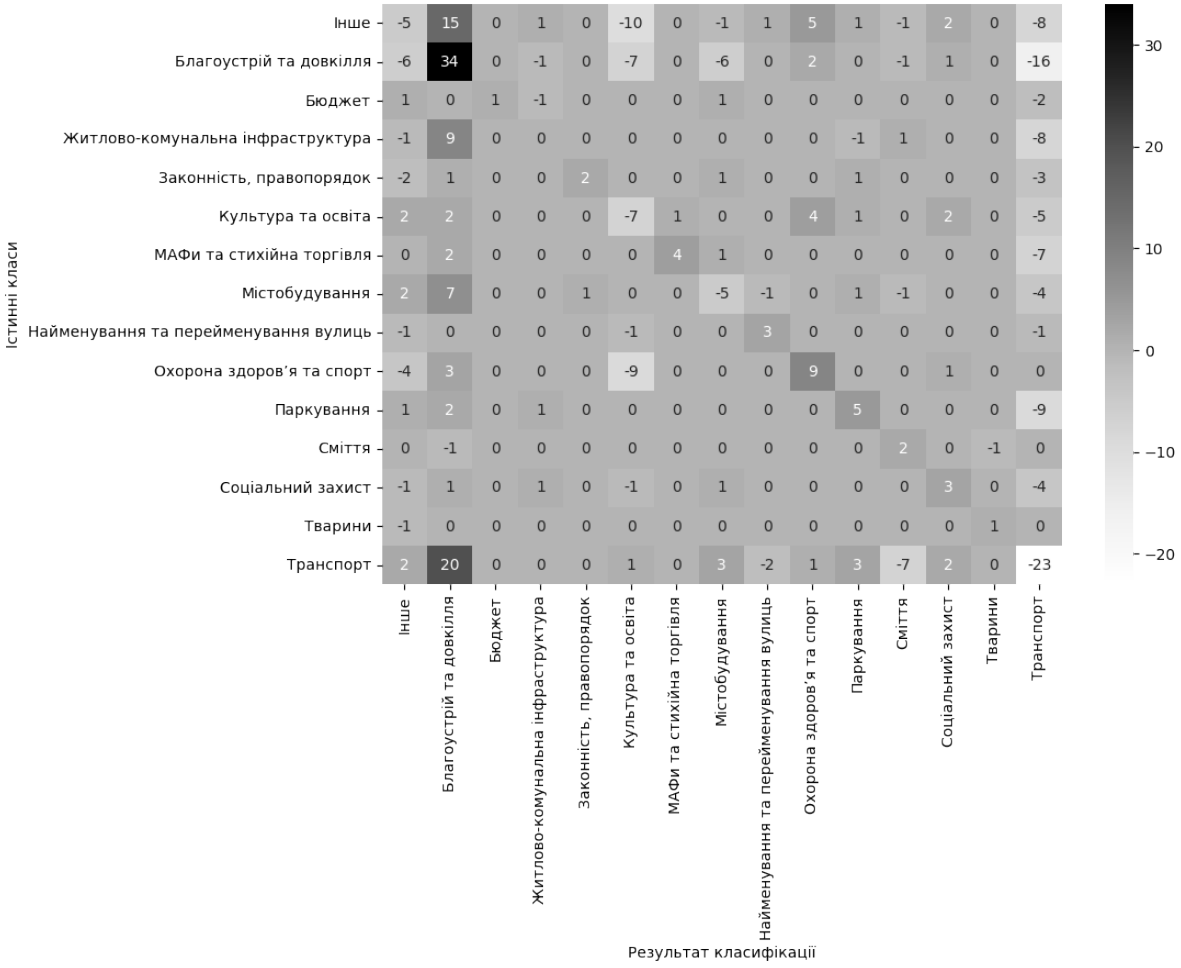

Рис. 3.14 – Різниця матриць невідповідностей базової моделі та моделі на основі зваженої суми Word2Vec і Node2Vec для класифікації петицій

3.3 Аналіз гіперпараметрів *a* та *β* запропонованого методу

Оскільки значення *a* та *β* мають великий вплив на ефективність запропонованого методу, далі наводиться аналіз їх оптимальних значень. Для того щоб їх знайти було запущено сітковий алгоритм пошуку з повним перезапуском навчання та валідації. Сітка значень *a* та *β* складалась зі значень від -1.5 до 1.5 з кроком 0.02, що результує у 22500 експериментів. Результати представлені на рис. 3.15. Менші абсолютні значення *β* разом із більшими абсолютними значеннями *a* призводять до вищих зважених F1-мір у побудованих експериментах, що відповідає припущенню, що інформація про синоніми є лише покращенням базових векторів. Більше того, одного векторного простору синонімів недостатньо, щоб

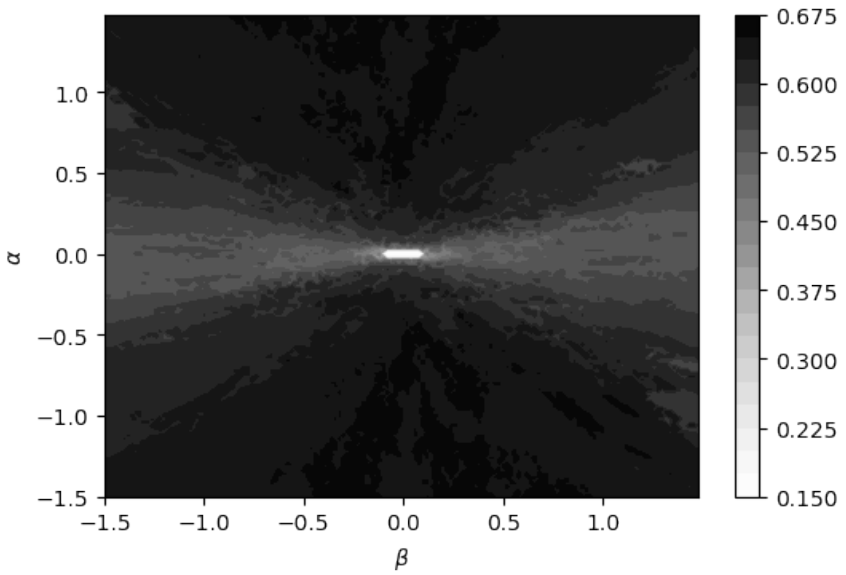

Рис. 3.15 – Зважені F1-міри при зміні *a* та *β* гіперпараметрів

повністю замінити класичні представлення документів, оскільки їм не вистачає конкретності та деталей, які присутні в практичних застосуваннях. Наприклад, назви вулиць не можуть бути охоплені представленнями синонімів, оскільки для них не існує синонімів. Це можна побачити в області простору, де *a* близький до нуля, а *β* змінюється. Абсолютний максимум метрик, які були досягнуті під час експериментів, знаходиться у точці *a* = 1,4 і *β* = 0,14. На рис. 3.16 також наводяться результати вищезгаданого сіткового пошуку з іншою метрикою – макро F1-мірою, яка не

враховує нерівномірність представлення класів у наборі даних, тобто не зважує класи відповідно до частоти їх наявності у наборі даних для тестування, але простір оптимальних гіперпараметрів процесу злиття через зважену суму виглядає подібним чином.

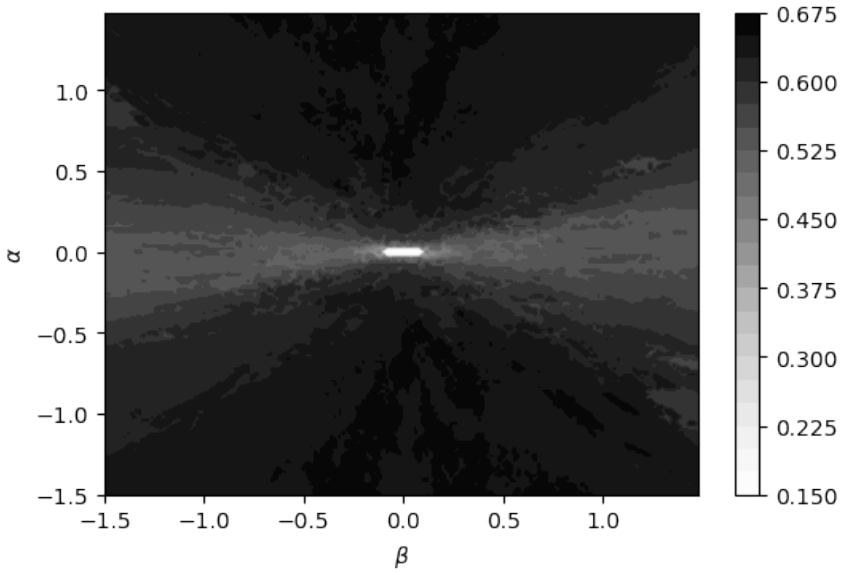

Рис. 3.16 – Макро F1-міри при зміні *a* та *ß* гіперпараметрів

Одним з можливих пояснень позитивних результатів даного дослідження може бути просто ефект регуляризації через шум, який подається як представлення синонімів. Однак дослідження показали, що це не так з двох причин: (1) експерименти з додаванням шуму до базових представлень документів не покращили результати; (2) якби це була регуляризація, пов'язана з шумом, ми б побачили симетричний спад метрик у області $a = 1.0$ при зміні гіперпараметра β.

Як практична рекомендація, додатково наводяться оптимальні гіперпараметри самого багатошарового перцептрону як класифікатора петицій. Саме за цих умов було отримано найвищі показники зваженої F1-міри:

- Кількість прихованих шарів – 1 шар з 100 нейронів;
- Функція активації – логістична;
- Крок навчання – адаптивний зі стартом в 0.01 і зменшенням у 5 разів кожні дві епохи, коли функція втрат не зменшується;

- Розмір батчу - 200;
- Рання зупинка (валідація на додатково відкладених 20% тренувальних даних);
- Метод оптимізації – Adam;
- $ß_1$-параметр методу оптимізації – 0.9;
- $ß_2$-параметр методу оптимізації – 0.999.

3.4 Статистична значимість результату

Моделі зазвичай оцінюються за допомогою методів повторної вибірки, таких як k-кратна перехресна перевірка, на основі якої обчислюються середні метрики, або з зафіксованим набором даних для тестування. Хоча ці підходи прості та ефективні, іноді на їх основі важко зробити висновок про ефективність моделі, оскільки при малих абсолютних значеннях зміни метрики не зрозуміло, чи є різниця між оцінками істинна, чи це результат статистичної випадковості.

Методи статистичної перевірки гіпотез призначені для вирішення цієї проблеми та кількісної оцінки вірогідності спостереження певних даних, враховуючи припущення, що вони були взяті з одного розподілу. Якщо це припущення або нульова гіпотеза відхиляється, це говорить про те, що різниця в даних та метриках є статистично значущою.

Процес вибору тесту для оцінки статистичної значущості результатів побудови моделей було описано у [179]. Одним із висновків даного аналізу є те, що двовибірковий t-критерій для залежних вибірок побудованих на основі повторів [180] не може бути використаний для аналізу відмінностей між двома моделями, адже основні припущення даного тесту про незалежність порушуються при генерації повторних випадкових виборок. Тому в даних ситуаціях рекомендується застосування Хі-квадрат тесту за методом МакНемара [181], який використовується для парних номінальних даних. На вхід тесту надається таблиця невідповідностей 2×2 з дихотомічною ознакою, у якій існують узгоджені пари суб’єктів. На основі цього

визначається, чи рівні граничні частоти рядка та стовпця (тобто чи є «гранична однорідність»).

Для перевірки статистичної значущості результату побудованої моделі, далі проводиться Хі-квадрат тесту за методом МакНемара. Сформульована нульова гіпотеза *H0* – результати класифікації документів базовим методом (на основі Word2Vec представлень) та запропонованим методом на основі словникової інформації є однаковими. Відповідно, альтернативна гіпотеза – ці результати неоднакові, а так як існує покращення в метриках, запропонований метод – дієвий. Таблиця невідповідностей 2x2 на вхід тесту наведена у таблиці 3.10.

Таблиця 3.10 – Таблиця невідповідностей для Хі-квадрат тесту за методом МакНемара

		Класифікація запропонованим методом	
		правильна	помилкова
Класифікація базовим методом	правильна	757	59
	помилкова	83	350

Результуюче значення статистики тесту рівне 4.056, що відповідає p-значенню 0.044 (менше рівня статистичної значущості alpha 0.05), а отже тест надає підстави відхилити нульової гіпотези про рівність результатів класифікації. При цьому, так як тестова метрика виросла, можна констатувати, що відмінність результатів полягає у покращенні результатів класифікації.

3.5 Висновки до розділу

Третій розділ дисертації було присвячено опису експериментальних результатів імплементації запропонованого методу, його порівнянню з іншими методами обробки природної мови у малоресурсному середовищі, а також порівнянню різних варіантів самої імплементації на прикладі задачі класифікації петицій до Київської міської ради

за темами. Показано, що доповнення векторних представлень документів векторними представленнями словника синонімів здатне покращити роботу класифікатора більш ніж на 2%. При цьому було досліджено набір особливостей побудови та роботи даного методу. Статистична значущість отриманих результатів була підтверджена Хі-квадрат тестом за методом МакНемара з рівнем статистичної значущості 0.05.

Експерименти показали, що у малоресурсному середовищі можна побудувати векторні простори документів на основі методів Word2Vec та FastText, а потім методами злиття доповнити їх векторним простором синонімів мови. При цьому метод на основі Word2Vec показав себе краще для моделювання петицій, оскільки він фіксував більше семантичних зв'язків між текстами петицій на відміну від синтаксичних, які фіксувались моделлю на основі FastText. Це сталося тому, що FastText за своєю природою працює на рівні підслова, і хоча це корисно для отримання векторів невідомих символьних послідовностей, процес усереднення векторів слів при побудові вектора документу прибирає цю перевагу. Кількісні результати аналізу побудованих просторів показують, що модель на основі Word2Vec отримує вищі значення коефіцієнту Силуетта, а отже краще підходить для подальшої кластеризації та створює більш щільні кластери, ніж модель на основі FastText.

Серед методів побудови векторних представлень графу, для словника синонімів найбільш оптимальним за необхідними ресурсами для побудови та впливом на результат класифікації виявився Node2Vec. При цьому, більшість методів на основі факторизації графу, такі як HOPE, LLE, Лапласівські проекції та GraRep, показали себе непрактичними для побудови представлення графу синонімів української мови через свою алгоритмічну складність. Інші методи, а саме Walkelts, Графові згорткові мережі та LINE, теж можуть бути застосовані для покращення роботи методів обробки природної мови у малоресурсному середовищі, хоч і показали менші значення F1-міри, ніж Node2Vec при класифікації петицій.

Для злиття векторних представлень документів та вузлів графу було досліджено два підходи – на основі конкатенації та на основі зваженої суми. У всіх експериментах

краще себе проявив метод на основі зваженої суми, а отже при класифікації документів запропонованим методом рекомендовано використовувати саме його. Єдиним його недоліком є необхідність підбору додаткових гіперпараметрів – вагів кожного з представлень, але у малоресурсному середовищі цей процес не є занадто затратним і може бути проведений у автоматичному режимі.

Запропоновані моделі можна використовувати як частину автоматизованих рішень для аналізу петицій. Вони надають кожній петиції числове представлення, що кодує її семантичне значення, та може бути включене у систему класифікації. Такі системи можуть допомогти визначити ставлення громадян до певних подій, групувати та знаходити дублікати петицій з однаковим наміром або передбачати, чи набере певна петиція достатньо голосів для розгляду.

ВИСНОВКИ

Дослідження в дисертаційній роботі було зосереджено на задачі обробки природної мови у малоресурсних середовищах. Його основними результатами є:

1. аналіз методів обробки природної мови у малоресурсних середовищах. Попри стрімкий розвиток методів обробки природної мови, малоресурсні середовища все ще маловивчені, а вибір методів, що дозволяють у них працювати, обмежений. Більшість з них працюють на припущеннях, що наявні додаткові нерозмічені дані, експертні знання чи альтернативні моделі, якими можна доповнити наявні ресурси. У багатьох малоресурсних середовищах такі припущення не виконуються, а отже пошук нових джерел корисних сигналів для моделей та розробка методів їх ефективнішого використання здатні відкрити нові можливості для автоматизації роботи з текстовими даними;
2. запропоновано метод класифікації документів при обробці природної мови у малоресурсному середовищі, який через використання векторних представлень словників надає внутрішнім моделям додатковий мовний сигнал для більш точної класифікації. Ідея методу у злитті векторних представлень документів та векторних представлень словника, який приймає форму графа, що відрізняється від традиційних способів поєднання словникової інформації через доповнення даних. Основним припущенням розробленого методу є доступність мовних словників синонімів, що може не виконуватись в усіх малоресурсних середовищах, наприклад, у мовах, де лінгвістична наука та спільнота все ще розвиваються, але є більш простим припущенням, ніж у інших методів обробки природних мов у малоресурсних середовищах, таких як наявність додаткових даних для розмітки чи великих моделей мов натренованих у багаторесурсному середовищі. У ході експериментальних досліджень було показано, що запропонований метод дозволяє покращити якість класифікації документів на 2-3% порівняно з моделями без словникової інформації, а також іншими методами обробки природної мови у малоресурсних середовищах. Даний

результат є статистично значущим, що було встановлено за допомогою статистичного Хі-квадрат тесту за методом МакНемара з рівнем статистичної значущості 0.05;

3. запропоновано векторну модель словника синонімів української мови, яку можна перевикористати за допомогою трансферного навчання у інших завданнях обробки природної мови у малоресурсних середовищах, а отже розвинено область трансферних моделей української мови;
4. сформульовані рекомендації з підбору компонентів, параметрів та гіперпараметрів запропонованого методу для вирішення задачі класифікації документів. Таким чином можна покращити якість класифікації при обробці природної мови у малоресурсному середовищі з малими витратами на модифікацію поточних рішень та здатністю перевикористовувати векторні представлення словника для вирішення інших задач. Серед компонентів найважливішими є методи побудови графових представлень словника синонімів, де найвищі значення тестової метрики отримав метод Node2Vec, а також методи злиття векторних представлень, де відповідно краще себе проявив метод злиття на основі зваженої суми.

Практичне значення отриманих результатів відображається у тому, що розроблений метод дозволяє підвищити якість систем класифікації документів у малоресурсних середовищах. Таким чином розробники даних систем можуть зменшити час та витрати на розробку, адже вища якість системи буде досягнута з меншою кількістю розмітки, доповнення якої може бути не доступним, або вимагати додаткових часових чи фінансових інвестицій. Також було розроблено векторні представлення слів у словнику синонімів української мови, які можна перевикористовувати за допомогою трансферного навчання при створенні програмних систем у інших прикладних областях. Було представлено набір даних для класифікації тем петицій, націлений на тестування методів обробки природної мови у малоресурсному середовищі. Це дозволить продовжити дослідження області

класифікації документів при обробці природної мови у малоресурсному середовищі та мати базові метрики для перевірки пропонованих методів та моделей. Документи написані українською мовою та мають вузьку урбаністичну спеціалізацію, що робить набір даних відмінним від корпусів загального призначення. У ході експериментів було розроблено систему класифікації петицій до Київської міської ради за темами, яка дозволяє автоматично пропонувати тему петиції при ручній розмітці, що може суттєво скоротити час на аналіз петицій. Наявність рішення на основі трансферного навчання розмежовує процес кодування мовної синонімії та процес навчання моделей класифікації документів, а отже робить зручним перевикористання даної додаткової інформації при вирішенні інших задач.

Достовірність отриманих результатів забезпечується проведенням аналізу наукових праць учених в області обробки природної мови, логікою та чітко поставленими завданнями, опрацюванням великої кількості статистичних та аналітичних матеріалів, наукових концепцій щодо класифікації документів при обробці природної мови у малоресурсних середовищах. Більше того, вона підтверджується шляхом порівняння з аналогічними методами обробки природних мов у малоресурсному середовищі, опублікованих у сучасній науковій літературі, а також аналізом статистичної значущості результатів запропонованого методу.

Дане дослідження у майбутньому можна розвивати у кількох напрямках. По-перше, можна досліджувати злиття з векторними представленнями інших типів лінгвістичної інформації, які можуть бути виражені графом, наприклад, етимологічні або тлумачні словники. Якщо буде доведено, що вони теж мають позитивний вплив на ефективність класифікації документів, наступним логічним кроком буде об’єднання кількох представлень на основі графів і перевірка їх кумулятивного ефекту. По-друге, дослідження можна спрямувати на розробку нових методів побудови представлень графів, які використовують особливості структури мовних словників такі як велику кількість кластерів та неглибокі зв’язки. Також оскільки розмір словників синонімів є відносно стабільним, це дослідження може оцінити час,

необхідний для створення таких векторних представлень, так як методи, засновані на випадковому блуканні, зазвичай займають багато часу на навчання і дуже чутливі до гіперпараметрів. І останній, але не менш важливий напрям, даний підхід можна перевірити у інших умовах з низькими ресурсами, наприклад, при використанні мов відмінних від української, але все ще з обмеженою присутністю в Інтернет-джерелах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] M. A. Hedderich, L. Lange, H. Adel, J. Strötgen, and D. Klakow, “A Survey on Recent Approaches for Natural Language Processing in Low-Resource Scenarios,” in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 2545–2568.

[2] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. Cambridge, MA, USA: MIT Press, 1999.

[3] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Pearson Prentice Hall, 2009.

[4] J. Howard and S. Ruder, “Universal Language Model Fine-tuning for Text Classification,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 328–339.

[5] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[6] R. Collobert, J. Weston, J. Com, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural Language Processing (Almost) from Scratch,” *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.

[7] R. Collobert and J. Weston, “A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning,” in *Proceedings of the 25th International Conference on Machine Learning*, 2008, pp. 160–167.

[8] Н. Дарчук, “Автоматичний синтаксичний аналіз текстів корпусу української мови,” *Українське мовознавство*, no. 43, pp. 11–19, 2013.

[9] А. Б. Романюк and З. М. Палій, “Дослідження алгоритмів автоматичного вирівнювання текстів англо-української мовної пари,” *Вісник Національного університету «Львівська політехніка» «Комп’ютерні системи проектування. Теорія і практика»*, no. 651, pp. 207–215, 2009.

[10] Є. А. Карпіловська, “Вступ до прикладної лінгвістики: комп’ютерна лінгвістика. Підручник,” *Донецьк ТОВ {\guillemotleft}Юго-Восток, Лтд*, 2006.

[11] Н. В. Бардіна and Н. В. Бардина, “Сучасні проблеми прикладної лінгвістики,” 2004.

[12] В. І. Перебийніс, *Статистичні методи для лінгвістів. Вид. 2.: Посібник*. Нова Книга, 2002.

[13] Т. Грязнухіна and Т. Любченко, “Електронний словник синонімів як засіб системного опису синонімічних відношень у лексичній системі української мови,” *Людина. Комп’ютер. Комунікація: збірник наукових праць*, pp. 56–60, 2017.

[14] B. Plank and Ž. Agić, “Distant Supervision from Disparate Sources for Low-Resource Part-of-Speech Tagging,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 614–620.

[15] J. Wei and K. Zou, “EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 2019, pp. 6382–6388.

[16] G. G. Şahin and M. Steedman, “Data augmentation via dependency tree morphing for low-resource languages,” *Proc. 2018 Conf. Empir. Methods Nat. Lang. Process. EMNLP 2018*, pp. 5004–5009, 2020.

[17] A. Lauscher, V. Ravishankar, I. Vuli´cvuli´c, and G. Glavaš, “From Zero to Hero: On the Limitations of Zero-Shot Language Transfer with Multilingual Transformers,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 4483–4499.

[18] R. Zhang *et al.*, “Improving Low-Resource Cross-lingual Document Retrieval by Reranking with Deep Bilingual Representations,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3173–3179.

[19] S. Ruder, “The 4 Biggest Open Problems in NLP.” [Online]. Available:

https://ruder.io/4-biggest-open-problems-in-nlp/. [Accessed: 05-Dec-2021].

[20] W. John Hutchins, "The Georgetown-IBM Experiment Demonstrated in January 1954," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3265, pp. 102–114, 2004.

[21] J. Weizenbaum, "ELIZAa computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, no. 1, pp. 36–45, Jan. 1966.

[22] R. C. Schank, N. Goldman, C. J. R. Iii, and C. Rissbeck, "MARGIE: Memory Analysis Response Generation, and Inference on English," *IJCAI*, 1973.

[23] J. R. Meehan, "TALE-SPIN, An Interactive Program that Writes Stories," *IJCAI*, 1977.

[24] W. G. Lehnert, "A conceptual theory of question answering," in *Proceedings of the 5th international joint conference on Artificial intelligence-Volume 1*, 1977.

[25] C. Christianson, J. Duncan, and B. Onyshkevych, "Overview of the DARPA LORELEI Program," *Mach. Transl. 2018 321*, vol. 32, no. 1, pp. 3–9, Jun. 2018.

[26] V. Berment, "Méthodes pour informatiser les langues et les groupes de langues « peu dotées ». (Methods to computerize 'little equipped' languages and groups of languages)," *Univ. Joseph-Fourier-Grenoble I Diss.*, 2004.

[27] D. Prys, "The BLARK Matrix and its relation to the language resources situation for the Celtic languages," in *Strategies for developing machine translation for minority languages*, 2006.

[28] T. Thongtan and T. Phienthrakul, "Sentiment Classification Using Document Embeddings Trained with Cosine Similarity," *ACL 2019 - 57th Annu. Meet. Assoc. Comput. Linguist. Proc. Student Res. Work.*, pp. 407–414, 2019.

[29] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning Word Vectors for Sentiment Analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 2011, pp. 142–150.

[30] "Ethnologue: Languages of the World." [Online]. Available:

https://www.ethnologue.com/. [Accessed: 05-Dec-2021].

[31] "LRE Map." [Online]. Available: http://portal.elda.org/en/catalogues/lre-map/. [Accessed: 05-Dec-2021].

[32] E. Bender, "The #BenderRule: On Naming the Languages We Study and Why It Matters," *The Gradient*, 2019.

[33] D. R. Mortensen, "Low-Resource NLP." [Online]. Available: http://demo.clab.cs.cmu.edu/algo4nlp20/slides/low-resource-nlp.pdf. [Accessed: 05-Dec-2021].

[34] P. Lison and J. Tiedemann, "OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2016, pp. 923–929.

[35] R. Lowe, N. Pow, I. V. Serban, and J. Pineau, "The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems," *SIGDIAL 2015 - 16th Annu. Meet. Spec. Interes. Gr. Discourse Dialogue, Proc. Conf.*, pp. 285–294, 2015.

[36] S. Zhang, E. Dinan, J. Urbanek, A. Szlam, D. Kiela, and J. Weston, "Personalizing Dialogue Agents: I have a dog, do you have pets too?," *ACL 2018 - 56th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf. (Long Pap.*, vol. 1, pp. 2204–2213, 2018.

[37] Chen Hongshen, Liu Xiaorui, Yin Dawei, and Tang Jiliang, "A Survey on Dialogue Systems," *ACM SIGKDD Explor. Newsl.*, vol. 19, no. 2, pp. 25–35, Nov. 2017.

[38] L. Perez and J. Wang, "The Effectiveness of Data Augmentation in Image Classification using Deep Learning," Dec. 2017.

[39] O. Büyük and L. M. Arslan, "Learning from mistakes: Improving spelling correction performance with automatic generation of realistic misspellings," *Expert Syst.*, vol. 38, no. 5, p. e12692, Aug. 2021.

[40] J. Raiman and J. Miller, "Globally normalized reader," *EMNLP 2017 - Conf. Empir. Methods Nat. Lang. Process. Proc.*, pp. 1059–1069, 2017.

[41] X. Dai and H. Adel, "An Analysis of Simple Data Augmentation for Named Entity

Recognition," in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 3861–3867.

[42] K. Gulordava, P. Bojanowski, E. Grave, T. Linzen, and M. Baroni, "Colorless green recurrent networks dream hierarchically," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018. Association for Computational Linguistics*, 2018, pp. 1195–1205.

[43] M. Fadaee, A. Bisazza, and C. Monz, "Data augmentation for low-Resource neural machine translation," *ACL 2017 - 55th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf. (Long Pap.*, vol. 2, pp. 567–573, 2017.

[44] S. Kobayashi, "Contextual augmentation: Data augmentation bywords with paradigmatic relations," *NAACL HLT 2018 - 2018 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 2, pp. 452–457, 2018.

[45] C. Vania, Y. Kementchedjhieva, A. Søgaard, and A. Lopez, "A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages," in *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, 2020, pp. 1105–1116.

[46] J. Min, R. T. Mccoy, D. Das, E. Pitler, and T. Linzen, "Syntactic Data Augmentation Increases Robustness to Inference Heuristics," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 2339–2352.

[47] O. Bojar and A. Tamchyna, "Improving Translation Model by Monolingual Data," in *Proceedings of the Sixth Workshop on Statistical Machine Translation*, 2011, pp. 330–336.

[48] V. Kumar, A. Ai, A. Choudhary, and E. Cho, "Data Augmentation Using Pre-trained Transformer Models," in *Proceedings of the Second Workshop on Life-long Learning for Spoken Language Systems Data Augmentation Using Pre-trained Transformer Models*, 2020.

[49] M. Yasunaga, J. Kasai, and D. Radev, “Robust multilingual part-of-speech tagging via adversarial training,” *NAACL HLT 2018 - 2018 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, pp. 976–986, 2018.

[50] D. Yarowsky, G. Ngai, and R. Wicentowski, “Inducing Multilingual Text Analysis Tools via Robust Projection across Aligned Corpora,” in *Proceedings of the First International Conference on Human Language Technology Research*, 2001.

[51] R. Eskander, S. Muresan, and M. Collins, “Unsupervised Cross-Lingual Part-of-Speech Tagging for Truly Low-Resource Scenarios,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 4820–4831.

[52] T. Mayer and M. Cysouw, “Creating a massively parallel Bible corpus,” in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC’14)*, 2014, pp. 3158–3163.

[53] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, “Distant supervision for relation extraction without labeled data,” in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009, pp. 1003–1011.

[54] X. Deng and H. Sun, “Leveraging 2-hop Distant Supervision from Table Entity Pairs for Relation Extraction,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 410–420.

[55] P. Lison, J. Barnes, A. Hubin, and S. Touileb, “Named Entity Recognition without Labelled Data: A Weak Supervision Approach,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 1518–1533.

[56] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, “Snorkel: rapid training data creation with weak supervision,” *VLDB J.*, vol. 29, no. 2, pp. 709–730, 2020.

[57] S. Li, J. Graça, and B. Taskar, “{W}iki-ly Supervised Part-of-Speech Tagging,” in

Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 2012, pp. 1389–1398.

[58] G. Karamanolakis, D. Hsu, and L. Gravano, *Cross-Lingual Text Classification with Minimal Resources by Transferring a Sparse Teacher*. 2020.

[59] H. Wang, B. Liu, C. Li, Y. Yang, and T. Li, "Learning with noisy labels for sentence-level sentiment classification," in *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, 2020, pp. 6286–6292.

[60] G. Karamanolakis, S. Mukherjee, G. Zheng, and A. H. Awadallah, "Self-Training with Weak Supervision," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 845–863.

[61] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *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)*, 2019, pp. 4171–4186.

[62] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Trans. Assoc. Comput. Linguist.*, vol. 5, pp. 135–146, 2017.

[63] B. Heinzerling and M. Strube, "BPEmb: Tokenization-free Pre-trained Subword Embeddings in 275 Languages," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[64] Y. Zhu, B. Heinzerling, I. Vulić, M. Strube, R. Reichart, and A. Korhonen, "On the Importance of Subword Information for Morphological Tasks in Truly Low-Resource Languages," in *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 2019, pp. 216–226.

[65] J. Jungmaier, N. Kassner, and B. Roth, "Dirichlet-Smoothed Word Embeddings for

Low-Resource Settings," in *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 3560–3565.

[66] A. Vaswani *et al.*, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, 2017, vol. 30.

[67] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," Jul. 2019.

[68] J. C. B. Cruz and C. Cheng, "Evaluating Language Model Finetuning Techniques for Low-resource Languages," Jun. 2019.

[69] C. Raffel, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer.," *J. Mach. Learn. Res.*, vol. 21, pp. 140:1-140:67, 2020.

[70] O. Melamud, M. Bornea, and K. Barker, "Combining Unsupervised Pre-training and Annotator Rationales to Improve Low-shot Text Classification," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3884–3893.

[71] K. Bhattacharjee *et al.*, "To BERT or Not to BERT: Comparing Task-specific and Task-agnostic Semi-Supervised Approaches for Sequence Tagging," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 7927–7934.

[72] J. O. Alabi, K. Amponsah-Kaakyire, D. I. Adelani, and C. Espã, "Massive vs. Curated Embeddings for Low-Resourced Languages: the Case of YorùbáYor`Yorùbá and Twi," in *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, 2020, pp. 2754–2762.

[73] L. Lange, H. Adel, and J. Strötgen, "On the Choice of Auxiliary Languages for Improved Sequence Tagging," in *Proceedings of the 5th Workshop on Representation Learning for NLP*, 2020, pp. 95–102.

[74] A. Conneau *et al.*, "Unsupervised Cross-lingual Representation Learning at Scale," in *Proceedings of the 58th Annual Meeting of the Association for Computational*

Linguistics, 2020, pp. 8440–8451.

[75] R. Hvingelby, A. B. Pauli, M. Barrett, C. Rosted, L. M. Lidegaard, and A. Søgaard, "DaNE: A Named Entity Resource for Danish," in *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 4597–4604.

[76] T.-Y. Hsu, C.-L. Liu, and H. Lee, "Zero-shot Reading Comprehension by Cross-lingual Transfer Learning with Multi-lingual Language Representation Model," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 5933–5940.

[77] B. Muller, B. Sagot, and D. Seddah, "Can Multilingual Language Models Transfer to an Unseen Dialect? A Case Study on North African Arabizi," May 2020.

[78] J. Hu, S. Ruder, A. Siddhant, G. Neubig, O. Firat, and M. Johnson, "XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalisation," in *International Conference on Machine Learning. PMLR*, 2020.

[79] S. Ruder, I. Vulić, and A. Søgaard, "A survey of cross-lingual word embedding models," *J. Artif. Intell. Res.*, vol. 65, pp. 569–631, May 2019.

[80] A. Joulin, P. Bojanowski, T. Mikolov, H. Jégou, and E. Grave, "Loss in Translation: Learning Bilingual Word Mapping with a Retrieval Criterion," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 2979–2984.

[81] S. Cao, N. Kitaev, and D. Klein, "Multilingual Alignment of Contextual Word Representations," in *International Conference on Learning Representations*, 2019.

[82] S. Wu and M. Dredze, "Are All Languages Created Equal in Multilingual BERT?," in *Proceedings of the 5th Workshop on Representation Learning for NLP*, 2020, pp. 120–130.

[83] S. Gururangan *et al.*, "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 8342–8360.

[84] E. Alsentzer *et al.*, “Publicly Available Clinical BERT Embeddings,” in *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, 2019, pp. 72–78.

[85] C. Alt, M. Hübner, and L. Hennig, “Fine-tuning Pre-Trained Transformer Language Models to Distantly Supervised Relation Extraction,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 1388–1398.

[86] J. Lee *et al.*, “BioBERT: a pre-trained biomedical language representation model for biomedical text mining,” *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, Feb. 2020.

[87] I. Beltagy, K. Lo, and A. Cohan, “{S}ci{BERT}: A Pretrained Language Model for Scientific Text,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3615–3620.

[88] A. Friedrich *et al.*, “The {SOFC}-Exp Corpus and Neural Approaches to Information Extraction in the Materials Science Domain,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 1255–1268.

[89] H. Xu, B. Liu, L. Shu, and P. Yu, “{D}om{BERT}: Domain-oriented Language Model for Aspect-based Sentiment Analysis,” in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 1725–1731.

[90] R. Aharoni and Y. Goldberg, “Unsupervised Domain Clusters in Pretrained Language Models,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7747–7763.

[91] D. Kiela, C. Wang, and K. Cho, “Dynamic meta-embeddings for improved sentence representations,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, 2018, pp. 1466–1477.

[92] L. Lange, H. Adel, J. Strötgen, and D. Klakow, “FAME: Feature-Based Adversarial Meta-Embeddings for Robust Input Representations,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 8382–8395.

[93] I. J. Goodfellow *et al.*, "Generative Adversarial Nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 2014, pp. 2672–2680.

[94] T. Gui, Q. Zhang, H. Huang, M. Peng, and X. Huang, "Part-of-Speech Tagging for {T}witter with Adversarial Neural Networks," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2411–2420.

[95] P. Liu, X. Qiu, and X. Huang, "Adversarial Multi-task Learning for Text Classification," in *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 2017, vol. 1, pp. 1–10.

[96] J. Kasai, K. Qian, S. Gurajada, Y. Li, L. Popa, and P. G. Allen, "Low-resource Deep Entity Resolution with Transfer and Active Learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5851–5861.

[97] D. Grießhaber, N. T. Vu, and J. Maucher, "Low-Resource Text Classification Using Domain-Adversarial Learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11171 LNAI, pp. 129–139, Oct. 2018.

[98] J. T. Zhou *et al.*, "Dual Adversarial Neural Transfer for Low-Resource Named Entity Recognition," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3461–3471.

[99] J.-K. Kim, Y.-B. Kim, R. Sarikaya, and E. Fosler-Lussier, "Cross-Lingual Transfer Learning for {POS} Tagging without Cross-Lingual Resources," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2832–2838.

[100] X. Chen, Y. Sun, B. Athiwaratkun, C. Cardie, and K. Weinberger, "Adversarial Deep Averaging Networks for Cross-Lingual Sentiment Classification," *Trans. Assoc. Comput. Linguist. vol. 6*, pp. 557–570, 2018.

[101] L. Lange, A. Iurshina, H. Adel, and J. Strötgen, "Adversarial Alignment of Multilingual Models for Extracting Temporal Expressions from Text," in *Proceedings of the 5th Workshop on Representation Learning for NLP*, 2020, pp. 103–109.

[102] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, vol. 70, pp. 1126–1135.

[103] M. Yu *et al.*, "Diverse Few-Shot Text Classification with Multiple Metrics," in *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2018, vol. 1, pp. 1206–1215.

[104] X. Chen, A. Ghoshal, Y. Mehdad, L. Zettlemoyer, and S. Gupta, "Low-Resource Domain Adaptation for Compositional Task-Oriented Semantic Parsing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020, pp. 5090–5100.

[105] Z.-Y. Dou, K. Yu, and A. Anastasopoulos, "Investigating Meta-Learning Algorithms for Low-Resource Natural Language Understanding Tasks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 2019, pp. 1192–1197.

[106] T. Bansal, R. Jha, T. Munkhdalai, and A. McCallum, "Self-Supervised Meta-Learning for Few-Shot Natural Language Classification Tasks," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 522–534.

[107] Y. Huang, J. Feng, S. Ma, X. Du, and X. Wu, "Towards Low-Resource Semi-Supervised Dialogue Generation with Meta-Learning," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 4123–4128.

[108] A. Rahimi, Y. Li, and T. Cohn, "Massively Multilingual Transfer for {NER}," in *Proceedings of the 57th Annual Meeting of the Association for Computational*

Linguistics, 2019, pp. 151–164.

[109] R. Socher, J. Bauer, C. D. Manning, and A. Y. Ng, "Parsing with Compositional Vector Grammars," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2013, pp. 455–465.

[110] R. Socher *et al.*, "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1631–1642.

[111] Y. M. Bishop, S. E. Fienberg, and P. W. Holland, *Discrete multivariate analysis: theory and practice*. Springer Science \& Business Media, 2007.

[112] R. Bellman, "On the theory of dynamic programming," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 38, no. 8, p. 716, 1952.

[113] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 2013.

[114] E. Tutubalina, S. Nikolenko, E. Tutubalina, and S. Nikolenko, "Demographic Prediction Based on User Reviews about Medications," *Comput. y Sist.*, vol. 21, no. 2, pp. 227–241, 2017.

[115] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[116] Z. S. Harris, "Distributional Structure," *Distrib. Struct. WORD*, vol. 10, no. 3, pp. 146–162, 1954.

[117] H. P. Luhn, "A Statistical Approach to Mechanized Encoding and Searching of Literary Information," *IBM J. Res. Dev.*, vol. 1, no. 4, pp. 309–317, 1957.

[118] G. Salton, E. A. Fox, and H. Wu, "Extended Boolean information retrieval," *Commun. ACM*, vol. 26, no. 11, pp. 1022–1036, Nov. 1983.

[119] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A Convolutional Neural Network for Modelling Sentences," in *Proceedings of the 52nd Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), 2014, pp. 655–665.

[120] R. Socher, E. Huang, J. Pennin, C. D. Manning, and A. Ng, "Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection," in *Advances in Neural Information Processing Systems*, 2011, vol. 24.

[121] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu, "Towards Universal Paraphrastic Sentence Embeddings," in *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2015.

[122] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and Word2vec for text classification with semantic features," in *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)*, 2015, pp. 136–140.

[123] S. Arora, Y. Liang, and T. Ma, "A Simple but Tough-to-Beat Baseline for Sentence Embeddings," in *International conference on learning representations*, 2017.

[124] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Syst.*, vol. 151, pp. 78–94, Jul. 2018.

[125] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed Large-Scale Natural Graph Factorization," in *Proceedings of the 22nd International Conference on World Wide Web*, 2013, pp. 37–48.

[126] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric Transitivity Preserving Graph Embedding," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1105–1114.

[127] C. F. Van Loan, "Generalizing the Singular Value Decomposition," *SIAM J. Numer. Anal.*, vol. 13, no. 1, pp. 76–83, Dec. 1976.

[128] R. S. T. and S. L. K., "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science (80-.).*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.

[129] S. Cao, W. Lu, and Q. Xu, "GraRep: Learning Graph Representations with Global Structural Information," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 2015, pp. 891–900.

[130] M. Belkin and P. Niyogi, “Laplacian Eigenmaps for Dimensionality Reduction and Data Representation,” *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, 2003.

[131] M. E. J. Newman, “A measure of betweenness centrality based on random walks,” *Soc. Networks*, vol. 27, no. 1, pp. 39–54, Jan. 2005.

[132] F. Fouss, A. Pirotte, J. Renders, and M. Saerens, “Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation,” *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 355–369, 2007.

[133] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online learning of social representations,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 701–710.

[134] A. Grover and J. Leskovec, “Node2vec: Scalable Feature Learning for Networks,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 855–864.

[135] B. Perozzi, V. Kulkarni, H. Chen, and S. Skiena, “Don’t Walk, Skip! Online Learning of Multi-Scale Network Embeddings,” in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, 2017, pp. 258–265.

[136] Y. Bengio, A. Courville, and P. Vincent, “Representation Learning: A Review and New Perspectives,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Jun. 2013.

[137] D. Wang, P. Cui, and W. Zhu, “Structural Deep Network Embedding,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1225–1234.

[138] S. Cao, W. Lu, and Q. Xu, “Deep Neural Networks for Learning Graph Representations,” *Proc. AAAI Conf. Artif. Intell.*, vol. 30, no. 1 SE-Technical Papers: Machine Learning Applications, Feb. 2016.

[139] T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” in *5th International Conference on Learning*

Representations, ICLR 2017 - Conference Track Proceedings, 2016.

[140] S. F. Mousavi, M. Safayani, A. Mirzaei, and H. Bahonar, "Hierarchical graph embedding in vector space by graph pyramid," *Pattern Recognit.*, vol. 61, pp. 245–254, Jan. 2017.

[141] X. Zhao, A. Chang, A. Das Sarma, H. Zheng, and B. Y. Zhao, "On the Embeddability of Random Walk Distances," *Proc. VLDB Endow.*, vol. 6, no. 14, pp. 1690–1701, Sep. 2013.

[142] X. Wei, L. Xu, B. Cao, and P. S. Yu, "Cross View Link Prediction by Learning Noise-Resilient Representation Consensus," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 1611–1619.

[143] Y. Lin, Z. Liu, and M. Sun, "Knowledge Representation Learning with Entities, Attributes and Relations," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 2016, pp. 2866–2872.

[144] L. F. R. Ribeiro, P. H. P. Savarese, and D. R. Figueiredo, "struc2vec: Learning Node Representations from Structural Identity," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, vol. Part F129685, pp. 385–394.

[145] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-Scale Information Network Embedding," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 1067–1077.

[146] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, 2012, vol. 25.

[147] Y. Bengio, R. Ducharme, and P. Vincent, "A Neural Probabilistic Language Model," in *Advances in Neural Information Processing Systems*, 2001, vol. 13.

[148] C. Zhang, Z. Yang, X. He, and L. Deng, "Multimodal Intelligence: Representation Learning, Information Fusion, and Applications," *IEEE J. Sel. Top. Signal Process.*, vol. 14, no. 3, pp. 478–493, Mar. 2020.

[149] C. G. M. Snoek, M. Worring, and A. W. M. Smeulders, “Early versus Late Fusion in Semantic Video Analysis,” in *Proceedings of the 13th Annual ACM International Conference on Multimedia*, 2005, pp. 399–402.

[150] В. И. Левенштейн, “Двоичные коды с исправлением выпадений, вставок и замещений символов,” in *Доклады Академии наук*, 1965, vol. 163, no. 4, pp. 845–848.

[151] B. Nojavanasghari, D. Gopinath, J. Koushik, T. Baltrušaitis, and L.-P. Morency, “Deep Multimodal Fusion for Persuasiveness Prediction,” in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, 2016, pp. 284–288.

[152] V. Vielzeuf, A. Lechervy, S. Pateux, and F. Jurie, “Centralnet: a multilayer approach for multimodal fusion,” in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, p. 0.

[153] B. Zhou, Y. Tian, S. Sukhbaatar, A. Szlam, and R. Fergus, “Simple Baseline for Visual Question Answering,” Dec. 2015.

[154] J. Perez-Rua, V. Vielzeuf, S. Pateux, M. Baccouche, and F. Jurie, “MFAS: Multimodal Fusion Architecture Search,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6959–6968.

[155] C. Liu *et al.*, “Progressive Neural Architecture Search,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, vol. 11205 LNCS, pp. 19–35.

[156] “What is Exploratory Data Analysis? | IBM.” [Online]. Available: https://www.ibm.com/cloud/learn/exploratory-data-analysis. [Accessed: 05-Dec-2021].

[157] L. Hagen, “Content analysis of e-petitions with topic modeling: How to train and evaluate LDA models?,” *Inf. Process. Manag.*, vol. 54, no. 6, pp. 1292–1307, Nov. 2018.

[158] M. Asher, C. L. Bandeira, and V. Spaiser, “Assessing the effectiveness of e-petitioning through Twitter conversations,” in *Political Studies Association Annual Meeting 2017*,

2017.

[159] L. Hagen, T. M. Harrison, Ö. Uzuner, W. May, T. Fake, and S. Katragadda, “E-petition popularity: Do linguistic and semantic factors matter?,” *Gov. Inf. Q.*, vol. 4, no. 33, pp. 783–795, Oct. 2016.

[160] “Головне управління статистики м.Києва - Населення (1990-2020pp.).” [Online]. Available: http://www.kiev.ukrstat.gov.ua/p.php3?c=527&lang=1. [Accessed: 05-Dec-2021].

[161] “City Mayors: The 500 largest European cities (1 to 100).” [Online]. Available: http://www.citymayors.com/features/euro_cities1.html. [Accessed: 05-Dec-2021].

[162] “Запрацював сайт петицій до Київради - DreamKyiv.” [Online]. Available: http://dreamkyiv.com/zapushheno-sajt-petytsij-kyyivrady/. [Accessed: 05-Dec-2021].

[163] G. Pant, P. Srinivasan, and F. Menczer, “Crawling the Web,” *Web Dyn.*, pp. 153–177, 2004.

[164] D. Kouzis-Loukas, *Learning Scrapy*. Packt Publishing Ltd., 2016.

[165] P. Geneves, “Course-e XPath Language.” October, 2012.

[166] “Polite Crawler.” [Online]. Available: https://courses.cs.washington.edu/courses/cse454/02au/polite-crawler.htm. [Accessed: 05-Dec-2021].

[167] “Офіційний сайт Української мови.” [Online]. Available: https://ukrainskamova.com/. [Accessed: 05-Dec-2021].

[168] Sociological Service “Razumkov Centre,” “Consolidation of the Ukrainian society: ways, challenges, prospects,” 2016. [Online]. Available: https://razumkov.org.ua/uploads/journal/eng/NSD165-166_2016_eng.pdf. [Accessed: 05-Dec-2021].

[169] “Language Detection Library for Java.” [Online]. Available: https://www.slideshare.net/shuyo/language-detection-library-for-java. [Accessed: 05-Dec-2021].

[170] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, “Bag of Tricks for Efficient Text

Classification," in *Proceedings of the 15th Conference of the {E}uropean Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 2017, pp. 427–431.

[171] "stop-words · PyPI." [Online]. Available: https://pypi.org/project/stop-words/. [Accessed: 05-Dec-2021].

[172] R. Rehurek and P. Sojka, "Software framework for topic modelling with large corpora," in *In Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*, 2010.

[173] L. McInnes, J. Healy, N. Saul, and L. Großberger, "UMAP: Uniform Manifold Approximation and Projection," *J. Open Source Softw.*, vol. 3, no. 29, 2018.

[174] Mart\'\in~Abadi *et al.*, "{TensorFlow}: Large-Scale Machine Learning on Heterogeneous Systems." 2015.

[175] "Understanding UMAP." [Online]. Available: https://pair-code.github.io/understanding-umap/. [Accessed: 05-Dec-2021].

[176] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, 1987.

[177] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226–231.

[178] V. Marivate *et al.*, "Investigating an Approach for Low Resource Language Dataset Creation, Curation and Classification: Setswana and Sepedi," in *Proceedings of the first workshop on Resources for African Indigenous Languages*, 2020, pp. 15–20.

[179] T. G. Dietterich, "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, Oct. 1998.

[180] Student, "The Probable Error of a Mean," *Biometrika*, vol. 6, no. 1, pp. 1–25, Jun. 1908.

[181] Q. McNemar, “Note on the sampling error of the difference between correlated proportions or percentages,” *Psychometrika*, vol. 12, no. 2, pp. 153–157, 1947.

ДОДАТОК 1. СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ ТА ВІДОМОСТІ ПРО АПРОБАЦІЮ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЇ

Публікації у виданнях, проіндексованих у Scopus

[1] R. Shaptala and G. Kyselov, “Enhancing document representations with synonyms graph node embeddings,” *J. Theor. Appl. Inf. Technol.*, vol. 100, no. 1, pp. 70–80, Jan. 2022.

Статті у наукових фахових виданнях України

[2] Р. Шаптала і Г. Кисельов, «Метод злиття багатомодальних векторних представлень слів у малоресурсному середовищі», *ВОТТП*, вип. 1, с. 174–179, Бер. 2023.

[3] Р. Шаптала і Г. Кисельов, “Класифікація текстових документів з використанням доповнення векторних представлень документів графовими представленнями елементів словника синонімів,” *Інформаційні технології та суспільство*, вип. 3 (5), с. 49–55, Січ. 2023.

[4] Р. Шаптала і Г. Кисельов, “Огляд методів злиття векторних представлень,” *Телекомунікаційні та інформаційні технології*, вип. 4 (77), с. 84–89, 2022.

[5] R. V. Shaptala and G. D. Kyselev, “Using graph embeddings for Wikipedia link prediction,” *Bull. Natl. Tech. Univ. “KhPI”. Ser. Syst. Anal. Control Inf. Technol.*, vol. 0, no. 1 SE-INFORMATION TECHNOLOGY, pp. 48–52, Jul. 2019.

[6] Shaptala R.V. and Kyselov G.D., “Vector space models of Kyiv city petitions,” *Sci. notes Taurida Natl. V.I. Vernadsky Univ. Ser. Tech. Sci.*, vol. 32, no. 1, pp. 169–177, 2021.

Наукові праці, які засвідчують апробацію матеріалів дисертації та інші публікації

[7] A. Samvelyan, R. Shaptala, and G. Kyselov, “Exploratory data analysis of Kyiv city petitions,” in *2020 IEEE 2nd International Conference on System Analysis Intelligent Computing (SAIC)*, 2020, pp. 1–4.