Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Сегментація екземплярів на основі текстових запитів

(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-1

Артем Шкіль
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки

(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект

(повна назва освітньої програми)

Керівник доц. Олександра Вітько
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Шкілю Артему Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Сегментація екземплярів на основі текстових запитів

затверджена наказом університету від 19 травня 20 25 р. № 378Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20 червня 20 25 р.
3. Вихідні дані до роботи Науково технічні публікації, дослідження мультимодальних моделей машинного навчання, набори даних для сегментації екземплярів, перелік великих та візуальних мовних моделей, документація мови програмування Python

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі
2) Теоретичні дослідження
3) Практичні дослідження
4) Аналіз результатів досліджень

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	19.05.2025	виконано
2	Аналіз предметної галузі	20.05.2025	виконано
3	Проведення теоретичних досліджень	25.05.2025	виконано
4	Проведення практичних досліджень	01.05.2025	виконано
5	Аналіз результатів	03.06.2025	виконано
6	Написання пояснювальної записки	06.06.2025	виконано
7	Проходження нормоконтролю	11.06.2025	виконано
8	Попередній захист	17.06.2025	виконано
9	Захист перед ЕК	20.06.2025	

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Олександра Вітько
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 107 с., 38 рис., 5 табл., 4 дод., 25 джерел.

ВЕКТОР ПРЕДСТАВЛЕННЯ, ВЕКТОРНИЙ ПРОСТІР, ВІЗУАЛЬНИЙ ТРАНСФОРМЕР, ОРТАГОНАЛЬНИЙ БАЗИС, РЕЧЕННЄВИЙ ТРАНСФОРМЕР, СЕГМЕНТАЦІЯ ЕКЗЕМПЛЯРІВ, LLM, PYTHON, PYTORCH, VLM.

Об’єкт дослідження – задача сегментації екземплярів.

Предмет дослідження – створення підходу для сегментації екземплярів на основі текстових запитів.

Мета роботи – знаходження підходу приведення векторів представлення візуальної та текстової інформації до спільного векторного простору.

Методи дослідження – аналіз проведених досліджень на тему розробки архітектур мультимодальних моделей, теоретичні та практичні дослідження підходів приведення векторів представлення до спільного простору, аналіз результатів досліджень.

У ході цієї роботи було розглянуто три підходи приведення векторів представлення до спільного простору для розв’язання задачі сегментації екземплярів на основі текстових запитів: створення єдиної матриці трансформації, створення окремих матриць трансформації для початкових векторних просторів та використання ортогонального базису та спектральної трансформації. Було продемонстровано, що існує підхід приведення векторів до спільного простору, який використовує лише лінійні перетворення, а вибір напрямку приведення, наявність негативних пар векторів та правильна постановка задачі процесу навчання може сильно вплинути на успіх даної задачі приведення.

ABSTRACT

Bachelor’s thesis contains: 107 pp., 38 fig., 5 tabl., 4 ann., 25 references.

EMBEDDING, INSTANCE SEGMENTATION, LLM, ORTHOGONAL BASIS, PYTHON, PYTORCH, SENTENCE TRANSFORMER, VECTOR SPACE, VISUAL TRANSFORMER, VLM.

The object of research is the instance segmentation problem.

The subject of the research is the design of the solution to text prompt-based instance segmentation.

The purpose of the work is to study the approaches to transforming embeddings that represent visual and textual information to the joint vector space.

Research methods are an analysis of conducted research on the topic of multimodal model architecture design, theoretical and practical research of approaches to transforming embeddings to the joint vector space, analysis of research results.

In this thesis, three approaches to transforming embeddings to the joint vector space to solve the problem of text prompt-based instance segmentation were studied: creating single transformation matrix, creating separate transformation matrices for embeddings for each vector space and using vector embeddings on orthogonal basis and spectral transform. It was demonstrated that there exists approach to transforming embeddings to the joint vector space that uses only linear transformations, and the choice of transformation direction, existence of negative samples in the train dataset and correct task creation during training process may greatly influence the success of the given transformation problem.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 8
Вступ 9
1 Аналіз предметної галузі 11
1.1 Актуальність обраної теми 11
1.1.1 Актуальність сегментації як задачі в цілому 11
1.1.2 Актуальність використання трансформерів в алгоритмах комп'ютерного зору 13
1.1.3 Актуальність проблеми поєднання зображення та текстової інформації 17
1.2 Аналіз проведених досліджень 20
1.2.1 Будова мультимодальних моделей на прикладі VisualBERT ... 21
1.2.2 Створення спільного векторного простору для зображень та тексту на прикладі VisualBERT та CLIP-моделей 24
1.2.3 Аналіз існуючих моделей сегментації на основі запиту 26
1.3 Постановка задачі 30
2 Теоретичні дослідження 32
2.1 Опис запропонованого підходу вирішення поставленої задачі 32
2.2 Вибір моделі сегментації екземплярів 33
2.3 Вибір унімодальних моделей отримання векторів представлення 35
2.4 Вибір підходів приведення векторів до спільного простору та аспектів навчання відповідних елементів 38
2.4.1 Вибір датасету для навчання та його попередня обробка 39
2.4.2 Приведення векторів до спільного простору шляхом створення матриць трансформацій 43
2.4.3 Приведення векторів до спільного простору за допомогою ортогонального базису та спектральної трансформації 47
2.4.4 Вибір метрик для оцінки запропонованих підходів 49

2.5 Опис принципу порівняння векторів представлення тексту та зображень 51
3 Практичні дослідження 52
3.1 Робота з обраним датасетом та підготовка даних до навчання..... 52
3.2 Створення єдиної матриці трансформації 63
3.3 Створення окремих матриць трансформацій для обох векторних просторів 69
3.4 Приведення векторів до спільного простору за допомогою ортогонального базису та спектральної трансформації..... 72
4 Аналіз результатів досліджень 76
4.1 Аналіз успішного методу приведення векторів у спільний простір.. 76
4.1.1 Аналіз доцільності застосування текстових аугментацій..... 80
4.1.2 Аналіз обраного напрямку приведення векторів до спільного простору 81
4.1.3 Аналіз методу приведення на зображеннях з групами тварин . 84
4.2 Аналіз неуспішних методів приведення векторів представлення 86
4.3 Програмна реалізація повного запропонованого підходу 90
Висновки 91
Перелік джерел посилання..... 93
Додаток А Код класу моделі навчання 96
Додаток Б Код для отримання метрик моделей..... 98
Додаток В Повний код реалізації запропонованого підходу 103
Додаток Г Відомість кваліфікаційної роботи..... 107

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект;

Adam – Adaptive Moment Estimation – метод адаптивної оцінки моментів;

BERT – Bidirectional Encoder Representations from Transformer – двоспрямовані кодувальні представлення з трансформеру;

CLIP – Contrastive Language-Image Pre-Training – контрастне донавчання візуальної мови;

CLS – Class – клас;

IoU – Intersection over Union – відношення перетину до об’єднання;

LLM – Large Language Model – велика мовна модель;

RAM – Random Access Memory – оперативна пам’ять;

RCNN – Region-based Convolutional Neural Network – згорткова нейронна мережа на основі регіонів;

ResNet – Residual Network – залишкова нейронна мережа;

SAM – Segment Anything Model – модель сегментації будь-чого;

SBERT – Sentence Bidirectional Encoder Representations from Transformer – реченнєвий трансформер;

SVD – Singular Value Decomposition – сингулярний розклад матриці;

t-SNE – t-Distributed Stochastic Neighbor Embedding – t-розподілене вкладення стохастичної близькості;

VLM – Visual Language Model – візуальна мовна модель;

VRAM – Video Random Access Model – відео оперативна пам’ять.

ВСТУП

Сегментація зображення є одним з ключових завдань комп’ютерного зору. І хоча дана задача має різні підтипи [1]: семантична сегментація (присвоєння певного класу кожному пікселю зображення), сегментація екземплярів (доповнення до задачі розпізнавання об’єктів, де кожен об’єкт має чітко виражений та відокремлений від інших об’єктів регіон) та паноптична сегментація (поєднання семантичної сегментації та сегментації екземплярів, тобто кожен піксель не тільки має клас, а ще й ідентифікатор об’єкта, до регіону якого він належить) – суть проблеми однакова, а саме створення моделі, що може розбити зображення на окремі частини.

Сама задача сегментації вже не нова і детально вивчена та досліджена: уже існує безліч моделей, які здатні розпізнавати різноманітні об’єкти в зображеннях. Наприклад, датасет Microsoft COCO, який слугує бенчмарком для моделей різних задач комп’ютерного зору, в тому числі і сегментації, оцінює успіх моделей розпізнавати предмети, що належать до 91 класу, які мають широкий діапазон і включають в себе людей, тварин (кіт, собака, кінь тощо) та повсякденні предмети (дорожні знаки, електроніка тощо) [2]. Але навіть така різноманітність об’єктів не вирішує всі проблеми, що пов’язані з сегментацією. Так, моделі, навчені на строго визначеному наборі класів, не матимуть змогу сегментувати об’єкти інших класів (наприклад, модель, навчена на вище зазначеному датасеті не зможе розпізнати дерева, так як клас «дерево» не входить до переліку класів), що призводить до відсутності гнучкості. Тому сучасні дослідження в галузі сегментації зосереджуються на створенні так званих zero-shot моделей, які здатні розпізнавати об’єкти без чітко визначеного переліку класів.

Але самої такої моделі недостатньо, адже не всі об’єкти на зображенні будуть цікавити користувача. Тому є потреба у створенні моделей, які здатні розпізнавати об’єкти на основі запиту. Такі запити можуть бути подані у

великій кількості форматів, але найбільш поширений з них є саме текст, адже здебільшого людина для опису предмету використовує саме природну мову.

Одним з підходів вирішення вище окресленої задачі є порівняння текстового запиту з кожним сегментом зображення, але постає питання як порівнювати текст та зображення, адже ці джерела передачі інформації є абсолютно різними форматами. Уже є моделі для порівняння зображень з зображеннями (наприклад, так звані моделі-близнюки) та тексту з текстом (наприклад, реченнєві трансформери), які відрізняються в реалізації, але схожі за суттю: представлення тексту або зображення у вигляді одновимірного масиву чисел та знаходження пар таких масивів, які знаходяться найближче одне до одного, що і є показником «схожості». Враховуючи це, можемо сформулювати задачу порівняння тексту та зображень по-іншому: знаходження способу представлення зображень та тексту у вигляді масивів, що знаходяться в одному векторному просторі. Вирішення цієї задачі і є основною метою даної роботи.

В ході виконання роботи буде зосереджено увагу здебільшого на методах поєднання векторів в різних векторних просторах. Буде використано уже готові моделі для створення даних векторів, так як вони вже продемонстрували значні результати, коли йдеться про порівняння інформації в межах одного формату, і будуть виконані спроби поєднати їх таким чином, щоб вектори, здобуті від різних моделей, носили в собі схожу інформацію. При цьому, для побудови векторів буде використано саме моделі-трансформери, які показали здатність створювати високоякісні вектори загального призначення.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Актуальність обраної теми

На даному етапі розвитку моделей машинного навчання існує велика потреба у створенні можливостей отримання інформації з різних джерел, зокрема зображень, з використанням природної мови. Сегментація, як основна задача комп'ютерного зору, і яка знайшла застосування у багатьох сферах застосування ШІ, від будівництва (наприклад, виявлення тріщин у фундаменті) до медицини (наприклад, виявлення частин тканини, уражених інфекцією), не є виключенням. У поєднанні з природною мовою сегментація стає більш гнучкою та персоналізованою, так як дозволяє користувачам указати регіони, що їх цікавлять.

Задачі сегментації та моделювання мови окремо мають велику кількість досліджень та імплементацій, але задача їх поєднання все ще залишається відносно недослідженою. Як результат, це ускладнює ефективне використання сегментації в реальних умовах.

Таке поєднання потребує не лише індивідуальних моделей, здатних до точних передбачень, а й створення практичних підходів злиття графічної та текстової інформації. Такі підходи дадуть змогу ефективно використовувати уже створені великі zero-shot моделі, де система може відповідати на запити користувача без додаткового навчання. Надалі буде розглянуто актуальність усіх компонентів даної задачі більш детально.

1.1.1 Актуальність сегментації як задачі в цілому

Сегментація як одна із задач комп'ютерного зору є важливою частиною в багатьох системах аналізу графічних даних. Вона особливо є невід'ємною частиною систем, де просто виявлення об'єктів в зображенні недостатньо і необхідно мати уявлення про регіони зображення, що займає

лише даний об’єкт. В такому розумінні сегментацію можна розглядати як продовження задачі виявлення об’єктів, але дане визначення відповідає лише сегментації екземплярів і паноптичній сегментації. Але і семантична сегментація має широке застосування в системах для виявлення, наприклад, аномальних частин зображення. Недарма, у [3], де представлено модель сегментації U-Net – одну з перших моделей сегментації, де були використані шари згорткових нейронних мереж, розглядається саме проблема біомедичної сегментації, де задача полягає у виявленні розмежованих областей «безсмертних» клітин HeLa у зображеннях, отриманих з використанням електронних мікроскопів.

Однією з метрик актуальності задачі можна вважати кількість наукових робіт, що її розглядають, а у випадку робіт на тему машинного навчання – кількість запропонованих архітектур, які продемонстрували свою ефективність у точності та/або швидкодії. Беручи таке твердження до уваги, в якості прикладу актуальності сегментації можна взяти [4], де розглядаються різноманітні архітектури сегментації з використанням глибокого навчання.

Як зазначено в переліку ключових внесків даної роботи: в ній проведено аналіз сучасних наукових публікацій, які мають відношення до проблеми сегментації і представлено загальний огляд більше, ніж 100 алгоритмів сегментації, що були запропоновані з 2019 року. Велика кількість алгоритмів сегментації, кожен з яких має власні характерні риси, можна вважати незаперечним аргументом актуальності задачі.

З іншого боку, для розгляду актуальності задачі сегментації слід розглянути сфери, де вона є невід’ємною частиною. Основною відмінністю сегментації від інших задач комп’ютерного зору, таких як класифікація та виявлення об’єктів, можна вважати надання передбачень кожному пікселю, що дає більш деталізоване уявлення про зображення.

Таке попіксельне представлення є важливим у багатьох сферах, таких як автономне керування (для виявлення можливих перешкод), медичний

аналіз (для виявлення та відокремлення аномалій) або доповнена реальність (для створення інтерактивної сцени). Надання машині розуміння про окремі регіони зображення дає змогу вирішувати задачі, що потребують просторової обізнаності (тобто повного бачення того, що відбувається навколо) та автономного прийняття рішень на основі контексту. Це є особливо важливим, так як системи, що базуються на основі моделей машинного навчання та штучного інтелекту, повинні оперувати ефективно та інтуїтивно, не створюючи при цьому ризиків безпеці оточення.

1.1.2 Актуальність використання трансформерів в алгоритмах комп’ютерного зору

Протягом великого проміжку часу дослідження та розвитку комп’ютерного зору згорткові нейронні мережі були ключовою темою робіт та архітектур, які розглядалися.

На рисунку 1.1 подано архітектуру моделі AlexNet, що є основоположною у сучасних дослідженнях використання таких мереж, хоча ще наприкінці 80-х років XX століття було запропоновано базову систему згорткових шарів для оцифрування поштових індексів США, а в 90-х роках аналогічна система була розроблена для автоматичної обробки цифр на банківських чеках (дані задачі зараз мають назву «розпізнавання оптичних символів»).

В основі згорткових мереж лежать шари двох типів [5]:

– згорткові шари, в яких по зображенню проходить кернел (невелика матриця коефіцієнтів) з метою отримання пікселем інформації із суміжних пікселів;

– підвибіркові шари, в яких також по зображенню проходить кернел, але на відміну від згорткових шарів, дані шари слугують для зменшення розмірності зображення.

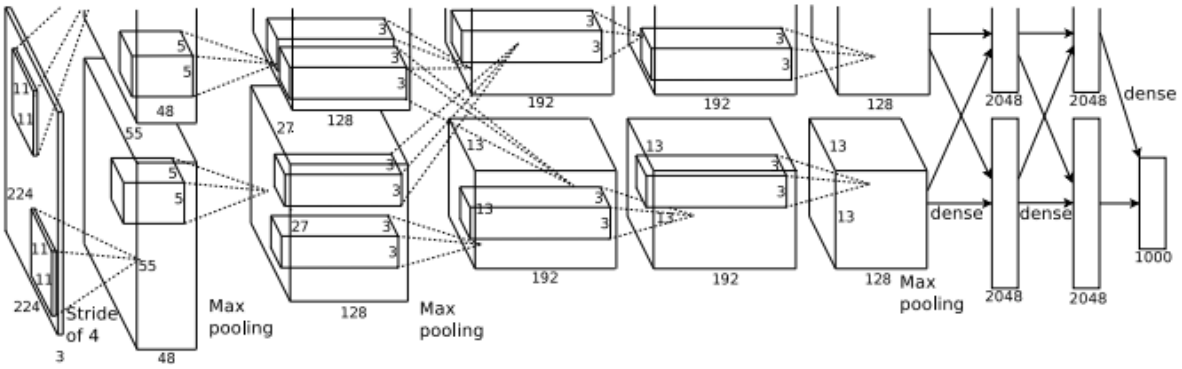

Рисунок 1.1 – Архітектура моделі AlexNet [6]

Варто зауважити, що для моделей сегментації на основі згорткових шарів також використовують надвибіркові шари, які використані для приведення зображення до оригінального розміру.

Враховуючи вище згадані основні шари згорткової нейронної мережі, можна виділити її індуктивні упередження (тобто припущення, які певна модель машинного навчання робить, і які мають виконуватися для успішного навчання даної моделі):

– трансляційна еквіваріативність: об’єкт може з’явитися на зображенні у будь-якому місці, тобто мережа «пройде» по всьому зображенню для знаходження характеристик об’єкту;

– локальність: для знаходження характеристик об’єкту в певному положенні мережа лише враховуватиме сусідні пікселі, при чому максимальна відстань таких пікселів визначається розміром кернела.

Варто зосередитися на другому індуктивному упередженні, через яке згортковій нейронній мережі буде складно поєднати характеристики, які знаходяться далеко одна від одної [7]. І хоча з проходженням зображення через мережу його розмірність зменшується завдяки підвибірковим шарам, а характеристики стають ближче одна до одної, в залежності від їх типу та специфіки реалізації підвибіркових шарів, вони все одно можуть бути втрачені.

Слід провести аналогію до задачі обробки природної мови, де перші алгоритми та нейронні мережі також мали проблему втрати можливості поєднання слів, що знаходилися далеко одне від одного.

Для вирішення цієї проблеми було запропоновано архітектуру моделі-трансформера, яка мала змогу знаходити семантичні відношення між словами, незалежно від відстані між ними. І хоча вперше ця архітектура, що з'явилася в [8], була використана для задачі машинного перекладу, вона зробила великий внесок у розвиток всіх задач обробки природної мови, від класифікації текстів та окремих слів до автоматичного написання тексту.

В основі моделі-трансформера, незалежно від проблеми, яку вона вирішує, завжди присутні такі частини:

– векторне представлення вхідних даних: текст, що подається на вхід, розбивається на токени, для кожного з яких модель створює векторне представлення; воно і буде оброблятися моделлю в подальшому;

– позиційне векторне представлення: до векторного представлення вхідних даних додається деякий вектор, який зберігає інформацію про положення токену в тексті;

– механізм «уваги»: найважливіша частина моделі-трансформера, яку слід розглянути детальніше.

Для кожної моделі даний механізм може відрізнятися, але оригінальний механізм «уваги» [8] представлено на рисунках 1.2 та 1.3. З векторного представлення токену обчислюються три матриці: запит, що містить інформацію про поточний фокус механізму, ключ, що містить інформацію про поточний токен, та значення, завдяки якому обчислюється фінальне представлення токену.

Ці матриці проходять через декілька голів механізму «уваги», що дає змогу розглянути текст з різних аспектів. Завдяки цьому для кожного токену буде отримано різні векторні представлення, які можна використати для подальшої обробки, що вже залежатиме від задачі.

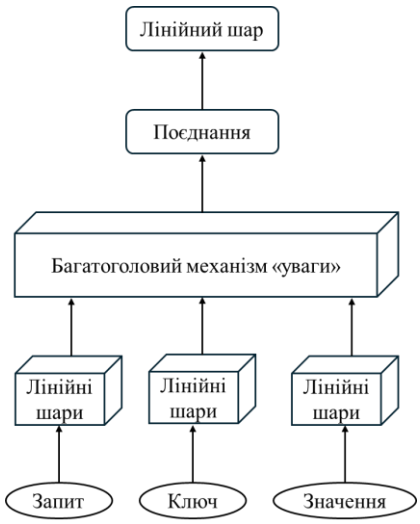

Рисунок 1.2 – Представлення механізму «уваги»

Рисунок 1.3 – Представлення однієї голови механізму «уваги»

Беручи до уваги велику популярність трансформерів у задачах обробки природної мови, це було лише справою часу, що дані принципи будуть випробувані на задачах комп’ютерного зору, починаючи з найпростішої – класифікації. Дане дослідження було проведене в [9], що уже показало свої переваги перед усталеними згортковими нейронними мережами.

Хоча реалізація деяких частин була видозмінена, сама архітектура залишилася такою самою: зображення ділиться на ділянки, для кожної

обчислюється векторне представлення (в оригінальній роботі в якості такого представлення беруть значення різних кольорових каналів усіх пікселів, «згладжені» до одновимірного вектора).

Як зазначено в [9], така обробка здебільшого нівелює локальність та трансляційну еквіваріативність згорткових нейронних мереж, так як лише лінійні шари мають такі упередження, а шари механізму «уваги» є глобальними.

Наостанок, варто зауважити, що використання трансформерів для задачі сегментації також має певні специфічні переваги:

– уникнення інших компонент, специфічних до завдання: для більшості згорткових нейронних мереж сегментації також додано інші компоненти (наприклад, просторові якорі в MaskRCNN), але використання універсальних трансформерів дає можливість уникнути їх додавання, що спрощує модель;

– узагальнення завдання сегментації: згорткові нейронні мережі створені для одного з підтипів сегментації (U-Net для задачі семантичної сегментації, MaskRCNN для задачі сегментації екземплярів), а використання трансформерів дає можливість створити універсальну сегментаційну модель, що може бути використана для всіх типів даної задачі одночасно.

1.1.3 Актуальність проблеми поєднання зображення та текстової інформації

Хоча дана робота розглядає проблему сегментації екземплярів на основі текстового запиту, варто також розглянути актуальність більш загальної проблеми, а саме поєднання зображення та текстової інформації.

Людина сприймає інформацію про оточення за допомогою п’яти органів чуття: зору, слуху, нюху, смаку та дотику – при чому часто для отримання повної картини про певний об’єкт, явище або поняття вона має

поєднувати інформацію з різних органів, а також інформацію, подану в різних форматах (наприклад, хоча текст і зображення сприймаються людиною за допомогою зору, ці засоби передачі інформації є абсолютно різними). Зокрема, у [10] провідний американський психолог Альберт Меєрабіан зазначає, що у спілкуванні 7% загального сенсу передається вербально, 38% – через різні аспекти голосу (гучність, тональність тощо), а 55% – невербально (через мову жестів, вирази обличчя тощо). Проводячи аналогію до штучного інтелекту, який часто розробляють для імітації людського, можна побачити, що для досить складних задач часто недостатньо обробляти інформацію лише в одному форматі, адже вона часто подається за допомогою поєднання декількох, наприклад:

– тексту та зображення: інфографіка подає інформацію у вигляді графіків, до яких додається текстове пояснення; в підручниках та наукових матеріалах часто подають зображення певного об’єкта з текстовими поясненнями його складових;

– тексту та аудіо: в онлайн-комунікації через месенджери окрім текстових повідомлень часто використовуються «голосові повідомлення», тобто запис голосу людини; більшість пісень мають музичний супровід, створений за допомогою музичних інструментів або синтезаторів, та слова, які виконавець співає під даний супровід;

– зображення та аудіо: у лекціях та презентаціях доповідачі часто використовують візуальну інформацію як доповнення усної; у сучасному медіапросторі набули популярності так звані подкасти, до яких автори часто додають зображення для додаткового залучення слухачів;

– тексту, зображення та аудіо: більшість відео, що уже є поєднанням зображення та аудіо, також мають субтитри, які можуть нести додаткову текстову інформацію.

Варто зауважити, що чим більше форматів інформації має обробляти штучний інтелект, тим складніше інженеру ШІ розробити таку модель, так як вона має коректно обробляти інформацію в усіх комбінаціях форматів,

тобто мають бути враховані випадки, коли інформація з одного або декількох форматів відсутня. Надалі буде зосереджено увагу лише на аналізі поєднання інформації, поданої у вигляді тексту та зображення.

Проблема поєднання текстової та візуальної інформації вже не нова: існують різноманітні задачі, де дані формати поєднуються різними способами. Наприклад, в уже згаданому вище датасеті Microsoft COCO [2] присутні інструменти для оцінки задачі опису зображення, де на вхід подається рисунок, а на виході отримується його текстовий опис (рисунок 1.4). Таким чином, для моделей даної задачі, вхідні дані є візуальними, а вихідні подані у вигляді тексту. Варто зауважити, що і протилежна задача, тобто задача, де за текстовим описом модель створює зображення, також має неабияку актуальність.

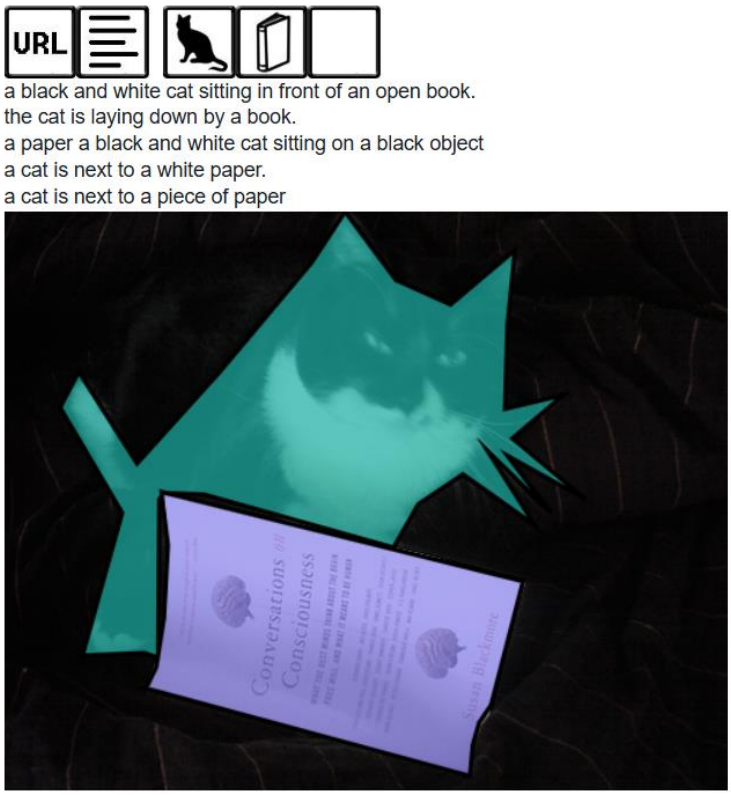

Рисунок 1.4 – Приклад зображення датасету Microsoft COCO з текстовим описом

Але варто розглянути інший спосіб поєднання текстової та візуальної інформації – задачі, в яких вхідні дані подаються в обох форматах одночасно. Постає питання, як порівнювати інформацію зображень та тексту. Уже створена достатня кількість моделей для порівняння текстових даних (наприклад, векторні представлення, витягнуті з моделі-трансформера, можна використати для порівняння різних параграфів; такі моделі називають реченнєвими трансформерами) та зображень (моделі-близнюки, які порівнюють векторні представлення, отримані або від візуальних трансформерів, або завдяки проходженню зображення через згорткові та підвибіркові шари) окремо, але дані моделі створюють вектори в різних просторах, які неможливо порівнювати без застосування додаткових шляхів перетворення числових даних. Саме тому існує потреба у створенні засобів приведення даних, взятих з різних форматів, у спільний простір, що даватиме можливість семантичного поєднання інформації з різних джерел.

1.2 Аналіз проведених досліджень

Хоча задача поєднання графічної та текстової інформації є новою та не такою дослідженою, як інші сфери ШІ, уже існують дослідження на дану тематику. Вони мають різні теми, але їх мета здебільшого однакова – створити підхід для використання семантичної інформації, видобутої з різних джерел.

Говорячи про проблему поєднання інформації з різних джерел, її можна об’єднати в задачу кросмодального поєднання представлення, де така інформація буде приведена до спільного простору, в якому «схожість» вимірюється об’єктивною метрикою.

Надалі будуть розглянуті деякі підходи поєднання такої інформації, як в загальному розумінні, так і для задачі сегментації.

1.2.1 Будова мультимодальних моделей на прикладі VisualBERT

Дослідження, в яких піднімається проблема поєднання інформації з різних форматів, зазвичай зосереджуються на задачі створення мультимодальних моделей. В контексті машинного навчання моделі розділяють за кількістю форматів, які вони можуть обробляти, на унімодальні та мультимодальні. Унімодальні моделі можуть обробляти інформацію лише в одному її представленні (наприклад, лінійні та логістичні регресори для табличних даних, згорткові нейронні мережі для зображень, рекурентні нейронні мережі для текстових або аудіо даних), в той час як мультимодальні моделі мають змогу поєднувати інформацію з різних джерел. І хоча детальні структури архітектур мультимодальних моделей відрізняються в залежності від способу поєднання та задачі, які вони вирішують, загальна структура таких моделей представлена на рисунку 1.5.

Зокрема, в таких моделях можна виділити наступні частини:

– вхідні дані в різних форматах: на вхід до моделі подаються дані з різних джерел, при побудові моделі також варто врахувати випадки, коли дані з одного або декількох джерел відсутні (якщо такі випадки можливі в даній предметній галузі);

– унімодальні моделі: інформація в кожному форматі проходить через власну унімодальну модель, що витягує семантично важливу інформацію з даного джерела;

– часткові шари злиття: після окремої обробки кожного джерела інформація з декількох (але не всіх одразу) джерел може поєднуватися і подаватися у вигляді спільного вектору;

– загальний шар злиття: векторні представлення всіх джерел поєднуються в одне спільне.

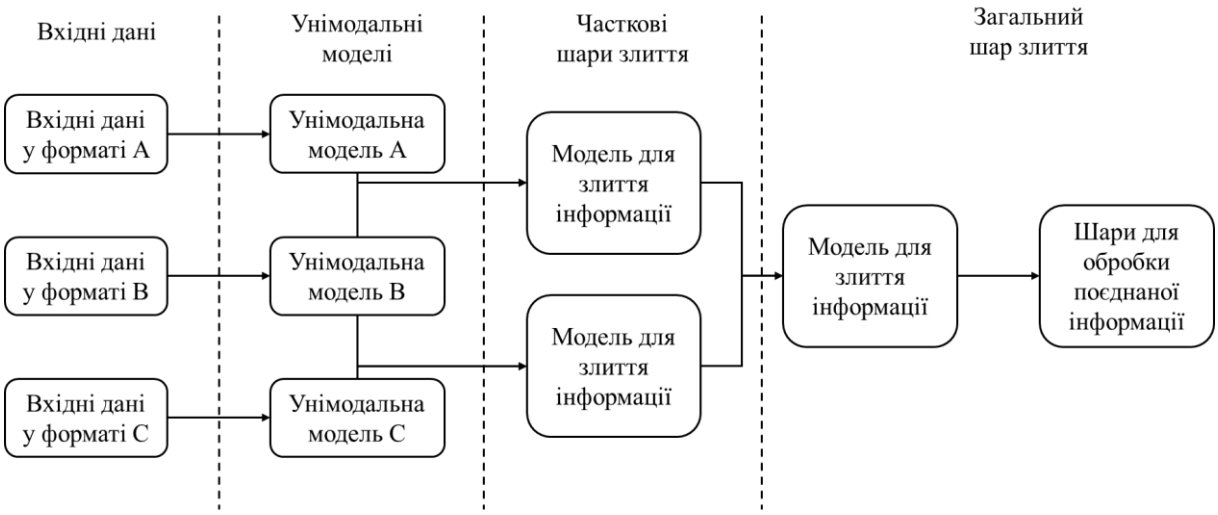

Рисунок 1.5 – Загальна структура мультимодальних моделей

Таким чином, наприкінці модель отримує єдине представлення інформації, зібране з різних джерел, і яке може бути опрацювано в залежності від задачі, що вона має вирішувати.

Однією такою задачею використання мультимодальних систем для розв’язання проблеми поєднання графічної та текстової інформації є задача отримання відповіді на візуальне запитання (Visual Question Answering), де модель отримує зображення та текстове запитання і має дати правильну відповідь на поставлене питання. Прикладом такої моделі є VisualBERT [11], яка бере за основу принципи, закладені в звичайну BERT-модель [12]. І звичайний BERT, і VisualBERT для векторного представлення використовують суму трьох векторів: вектора токена, вектора сегмента та вектора позиції, але інформація, що представлена в даних векторах, відрізняється. Опис такої інформації подано в таблиці 1.1. Завдяки такій структурі подання інформації у векторах VisualBERT має можливість включати в текстове речення графічну інформацію, при чому саме такий підхід дає змогу включати зображення в різні позиції речення, а також включати в речення декілька зображень.

Таблиця 1.1 – Інформація, що представлена в різних векторах представлення інформації в моделях BERT та VisualBERT

Вектор представлення	Опис інформації, що модель BERT містить в даному векторі	Опис інформації, що модель VisualBERT містить в даному векторі
Вектор токена	Вектор представлення деякого токена зі словника усіх можливих токенів.	Для тексту: вектор токена, аналогічний вектору токена в BERT. Для зображення: вектор представлення деякого сегмента зображення, що було отримане за допомогою згорткових шарів.
Вектор сегмента	Текст ділиться на два речення (наприклад, питання-відповідь). Вектор зберігає інформацію, до якого речення даний токен належить.	Який тип даних представляє даний вектор: текстовий токен чи сегмент зображення.
Вектор позиції	Позиція токена в тексті	Позиція токена в тексті або позиція сегмента в зображенні
Загальний вектор представлення	Сума векторів токена, сегмента та позиції	

Такий підхід також має ще одну перевагу: можливість використання звичайної BERT-моделі в якості основи відповідної мультимодальної моделі. Як зазначено в [11], на початку процесу навчання мультимодальної моделі було взято параметри відповідної BERT-моделі, що пришвидшує

процес навчання (так як не має потреби навчати її «з нуля»). З іншого боку, процес навчання все ще є досить складним, оскільки обидві моделі мають різні специфіки навчання: для обох моделей спільною є задача так званого замаскованого моделювання мови (деякі токени в реченні замасковані, і перед моделлю стоїть задача передбачити токен, що має стояти замість токена-маски), але VisualBERT також необхідно навчити на завданні, пов’язаному із зображеннями. Це може бути, наприклад, уже згадане завдання опису зображення.

Було розглянуто принцип побудови мультимодальних моделей, зокрема побудови моделі з текстовими та графічними вхідними даними. Тепер слід розглянути інші способи поєднання інформації з даних джерел, зокрема, для задачі сегментації.

1.2.2 Створення спільного векторного простору для зображень та тексту на прикладі VisualBERT та CLIP-моделей

Необхідно розглянути спосіб поєднання текстової та графічної інформації, представлений у VisualBERT, з боку проблеми порівняння лише тексту та лише зображення. Для цього варто спочатку розглянути реченнєві BERT-моделі, задача яких порівнювати текстові речення між собою [13]. Обидва речення, які необхідно порівняти, паралельно проходять через BERT-модель та підвибірковий шар, а отриманий вектор представлення можна порівнювати тим чи іншим способом. Варто зауважити, що є декілька способів витягу єдиного вектора представлення для всього речення: використання вектора представлення спеціального токену [CLS], що вбудований в BERT-модель, обчислення середнього значення векторів всіх токенів або обчислення вектора-максимуму. Оскільки VisualBERT може створити вектори представлення для речень з зображеннями, можна замінити BERT на VisualBERT для порівняння семантичної інформації тексту та зображення між собою. Таким чином, вектори представлення для

обох форматів знаходяться в одному векторному просторі, що і дає можливість порівняння.

Але також слід розглянути інший спосіб створення спільного векторного простору, що дає змогу порівнювати зображення та текст між собою, і який досліджується в [14]. І хоча дане дослідження зосереджується на класифікації без обмеження множини класів (що є проблемою класичних класифікаційних моделей), даний процес можна легко видозмінити для задачі семантичного порівняння. Схему такого порівняння зображено на рисунку 1.6.

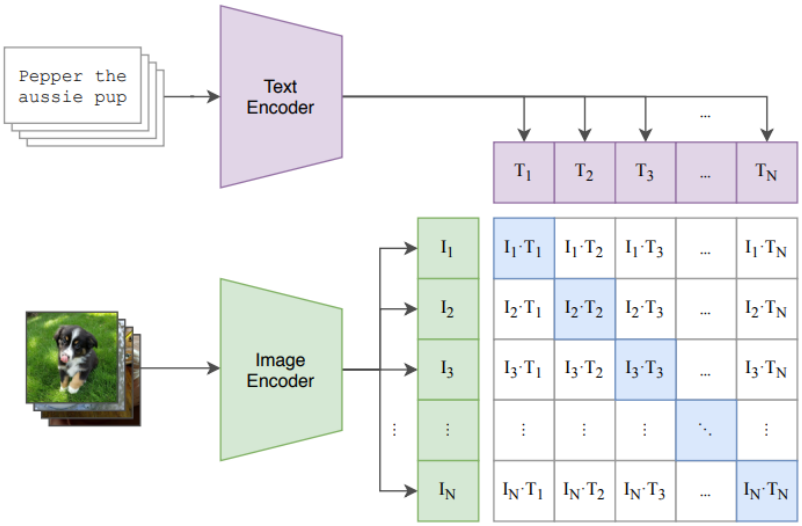

Рисунок 1.6 – Процес навчання спільного векторного простору [14]

У моделях, названих авторами CLIP-моделями, текст та рисунки проходять через окремі унімодальні енкодери (в дослідженні для тексту використовуються трансформери, а для зображення автори порівнюють результати ResNet-моделей та візуальних трансформерів), які створюють векторні представлення однакового розміру. Вони приводяться до спільного простору шляхом лінійної трансформації (матричного множення векторів на матрицю ваг). Таким чином, отримано вектори, що можна порівнювати між собою.

Даний спосіб створення спільного векторного простору має декілька переваг. По-перше, лінійне приведення векторів представлення текстового опису та зображень спрощує процес навчання. Якщо для навчання VisualBERT необхідно було створювати окремі функції втрат для «надання» алгоритму-оптимізатору настанов порівнювати семантичні характеристики текстових та графічних даних, для навчання CLIP-моделей достатнє використання уже створених функцій (у дослідженні використовується функція втрат крос-ентропії, так як воно зосереджено на проблемі класифікації, але можливе використання функцій для проблеми порівняння, наприклад функція втрат, що використовується в моделях-близнюках). Але все одно такий підхід не вирішує всіх проблем, так як для його успішного використання унімодальні моделі-енкодери мають на виході обчислювати вектори однакового розміру, що практично унеможливує використання уже навчених моделей, оскільки буде дуже складно їх підібрати. І хоча можливо знайти способи приведення векторів до однакового розміру: наприклад, створення окремої матриці ваг для кожної моделі або використання техніки зниження розмірності вектору з більшим розміром – це додає до складності моделі та не гарантує, що семантична інформація збережеться після застосування таких підходів. А говорячи про процес навчання «з нуля», самі автори роботи зазначають про складності у навчанні енкодерів та показують, що модель трансформатора природної мови із 63 мільйонами параметрів (яка вже використовує вдвічі більше обчислень, аніж відповідна модель ResNet-50 для зображень) вчиться розпізнавати класи, представлені в датасеті ImageNet втричі повільніше, ніж набагато простіша базова модель [14].

1.2.3 Аналіз існуючих моделей сегментації на основі запиту

Наостанок, слід розглянути уже існуючі дослідження проблеми поєднання текстової та графічної інформації в контексті

сегментації. Проблема сегментації на основі запиту як окреме завдання машинного навчання було вперше представлене у [15], де її ціль була описана, як отримання коректного регіону сегментації, маючи який-небудь запит. Запит просто вказує на те, що сегментувати в зображенні: наприклад, запит може включати просторову чи текстову інформацію, що ідентифікують об’єкт. Слід звернути більшу увагу на те, що автори не ставлять обмеження у форматах запиту. Власне, у дослідженні зазначено, що він може бути у вигляді:

– набору позитивних та негативних точок: множини точок, де будь-яка точка може вказувати на область, що треба включити в регіон сегментації (позитивна точка) або з нього виключити (негативна точка);

– обмежувального прямокутника: координат прямокутника, який обмежує положення об’єкта на зображенні (варто зауважити, що така проблема отримання регіону, маючи обмежувальний прямокутник екземпляру, є частиною архітектури моделі сегментації екземплярів Mask-RCNN, де для всіх отриманих обмежувальних прямокутників від моделі виявлення об’єктів, модель виділяє пікселі всередині прямокутника, які належать до об’єкту);

– тексту: опису об’єкта, для якого необхідно знайти регіон сегментації;

– регіону (маски): набору пікселів, де зображення знаходиться; автори дослідження назвали даний тип запиту «щільним», а решту – «розрідженими», ця класифікація запитів має значення, оскільки від цього змінюється процес обробки запитів моделлю, яку автори запропонували як базову.

Базова модель, запропонована у дослідженні, була названа SAM, вона отримує коректний регіон сегментації шляхом створення спеціального шару, який поєднує вектор представлення зображення разом з вектором представлення запиту. Загалом, модель має наступну структуру:

– енкодер зображення: все зображення проходить через спеціальний шар візуального трансформера, з якого отримується вектор представлення (самі способи отримання вектору з трансформера зазначені у підрозділі 1.1.2);

– згортковий шар для щільного запиту: якщо модель отримала на вході запит у вигляді регіону, дана маска (що представлена у вигляді двовимірного масиву значень пікселів, кожен з яких зберігає інформацію про те, чи входить даний піксель в регіон) проходить через згорткові та підвибіркові шари, а отриманий вектор представлення «додається» до вектору представлення зображення;

– енкодер розріджених запитів: запити, представлені іншими способами, проходять через CLIP-модель (для текстових запитів) або шар позиційного кодування (для просторових запитів);

– декодер регіону сегментації: шар, де поєднуються вектори представлення зображення та розріджених запитів, а на виході отримується множина кортежів регіонів сегментації та передбачення метрики IoU для прогнозування відсотку «покриття» регіоном об’єкта.

Сам шар поєднання векторів представлення подано на рисунку 1.7, з нього видно, що вектори проходять через різні типи механізму «уваги»: «самоуваги запиту», «уваги запиту до зображення» та «уваги зображення до запиту».

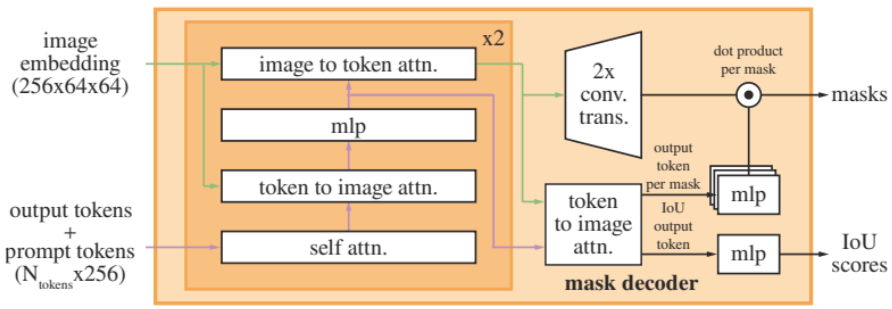

Рисунок 1.7 – Схема шару декодеру регіону сегментації моделі SAM [15]

Хоча автори пропонують використання тексту в якості запиту для сегментації зображення, вони зазначають, що модель підтримує лише короткі запити та найкраще працює з текстовим описом у поєднанні з просторовими запитами. Саме на просторових запитах і побудовані офіційні демонстраційні представлення SAM.

У [16] акцентується увага саме на проблемі сегментації на основі текстового запиту. У даному дослідженні розглядається підхід додавання до SAM-моделі шару раннього злиття візуальної інформації, який поєднує необроблене зображення з текстом, аналогічно до принципу поєднання у моделі VisualBERT, а отриманий вектор представлення проходить через шар проєкції до розміру вектора запиту, який можна подати на вхід до SAM-моделі (схематично архітектуру зображено на рисунку 1.8). Варто зауважити, що шари SAM, через які проходять вектори запитів (енкодер розріджених запитів, декодер регіону сегментації) змінюють свої ваги під час навчання, а решта шарів залишаються незмінними (так званий підхід «заморожування» шарів під час донавчання моделі).

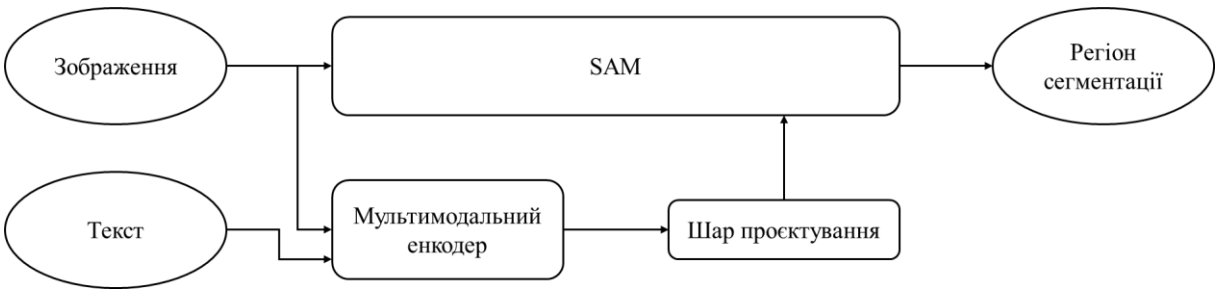

Рисунок 1.8 – Схема архітектури SAM-моделі з раннім злиттям візуальної інформації для текстових запитів

Хоча даний підхід додавання текстової інформації і використовує принцип «заморожування» деяких шарів та використовує уже досліджені архітектури, все одно велику кількість параметрів необхідно або донавчати, або навчати «з нуля», з чого випливає проблема необхідності великої

кількості навчальних даних та достатньої кількості обчислювальних ресурсів.

1.3 Постановка задачі

З аналізу уже проведених досліджень видно, що проблема поєднання текстової та графічної інформації має стратегії вирішення, особливо для задачі сегментації. Такі способи мають деякі аспекти, що ускладнюють використання відповідних моделей для доопрацювання, серед яких вже було зазначено:

– неможливість поєднання уже навчених моделей: деякі моделі, такі як CLIP, мають можливість поєднання лише векторів представлення однакових розмірів, що зменшує кількість можливих пар унімодальних моделей для обробки тексту та зображення;

– велика кількість параметрів, що необхідно оптимізувати під час процесу навчання: SAM з раннім злиттям візуальної інформації та VisualBERT мають досить великі за об’ємом шари, що повинні бути доопрацьовані, а це потребує більшого за розміром датасету (інакше виникне проблема перенавчання, коли модель «запам’ятовує» шум з навчальних даних, роблячи погані передбачення для нових) та більше обчислювальних ресурсів.

З іншого боку, проблема доопрацювання моделі може виникнути для специфічних предметних галузей, де уже готові моделі не вирішують задачі, наприклад, тому що не враховують необхідну для даної галузі семантичну інформацію. Для таких випадків розмір доступних датасетів зазвичай досить малий, тому розробники мають або штучно збільшувати розмір вибірки шляхом аугментацій, або шукати прості у навчанні стратегії. З [11] можна зробити висновок, що можливо підібрати простий підхід поєднання графічної та текстової інформації, так як автори дослідження не помітили

різниці в ефективності тренувань між двома версіями використання лінійних та нелінійних підходів поєднання векторів.

Таким чином, за мету даного дослідження слід взяти наступне: знайти стратегію сегментації на основі текстового запиту, в якому можливе використання уже доступних моделей обробки текстової та візуальної інформації, а також моделей сегментації. При цьому для витягу семантичної інформації доцільно використати моделі-трансформери, так як проаналізовані дослідження показали, що вони «збирають» семантичну інформацію, яку можливо використати для порівняння.

2 ТЕОРЕТИЧНІ ДОСЛІДЖЕННЯ

2.1 Опис запропонованого підходу вирішення поставленої задачі

Враховуючи поставлену вище задачу, необхідно створити підхід, в якому буде використано уже навчені моделі для частин задач, які потребують великої кількості обчислень. Такі частини включають в себе:

– задачу отримання регіонів сегментації: існує можливість використати уже готові моделі, так як вони навчені на великій кількості різноманітних навчальних даних; окрім цього, уже існують zero-shot моделі сегментації, що здатні виділити всі можливі об’єкти у зображені без додаткових вказівок;

– задачу представлення зображення у вигляді вектора: уже існують моделі (візуальні трансформери, VisualBERT, CLIP), з яких можливо витягнути вектори представлення візуальної інформації;

– задачу представлення тексту у вигляді вектора: задача семантичного порівняння текстових речень уже достатньо досліджена, тому існує багато відповідних моделей, наприклад, реченнєві трансформери;

– задачу порівняння векторів: існує велика кількість методів та підходів порівняння векторів представлення з метою отримання величини, що слугуватиме об’єктивною метрикою їх «схожості».

Але вектори представлень, отримані від різних моделей, знаходитимуться в різних векторних просторах, тому перш ніж порівнювати вектори між собою, їх необхідно привести до спільного простору. Такий підхід приведення, а також коректну стратегію порівняння приведених векторів між собою, і слід знайти. Оскільки дана частина є задачею розробки моделі машинного навчання, її слід розбити на окремі під задачі, а саме:

– вибір навчального датасету, який покриває всі аспекти моделі, яку необхідно навчити, та задачі, яку вона повинна розв’язати;

– підготовка навчального датасету для навчання, що включає в себе розбиття на навчальну, валідаційну та тестову вибірки, очистка датасету та застосування підходящих аугментацій;

– розробка та навчання моделей;

– вибір метрик, які слугуватимуть результатом успіху моделі у розв’язанні поставленої задачі.

Таким чином, маємо повний запропонований підхід до вирішення задачі сегментації на основі текстових запитів, який схематично представлений на рисунку 2.1.

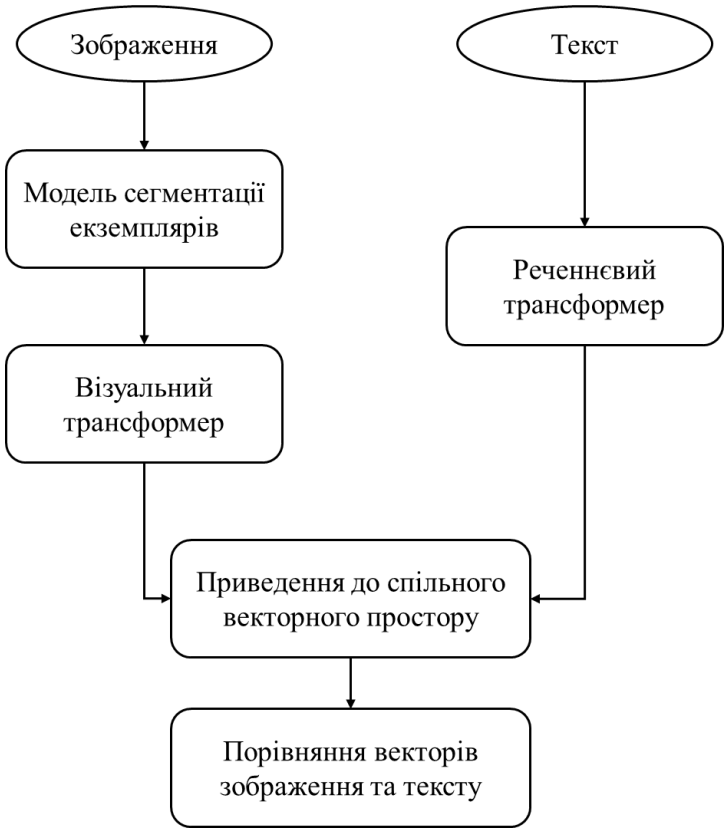

Рисунок 2.1 – Схема запропонованого підходу

2.2 Вибір моделі сегментації екземплярів

Сегментація екземплярів є однією з основних моделей, яка застосовується в запропонованому підході, так як вона виокремлює у

зображенні регіони, які потім будуть семантично порівнюватися з текстовими запитами. Кожен екземпляр слугує кандидатом, який може відповідати запиту користувача. Успіх всього підходу залежить від здатності моделі давати точні сегменти, які можуть потім бути успішно представлені у вигляді векторів, адже для отримання семантично багатих векторів представлення необхідно мати чіткі регіони об’єкту.

Як уже було зазначено в підрозділі 1.1.2, використання моделей сегментації на основі трансформерів є рішенням кращим, аніж використання звичайних моделей на основі згорткових шарів. На відміну від згорткових моделей, які зазвичай залежать від усталених рецептивних полів (які є доволі вузькими, так як кернели в таких моделях є дуже малими), моделі на основі трансформерів обробляють зображення з використанням механізмів «уваги», які охоплюють глобальний контекст всього зображення. Такі архітектури мають велику перевагу у генеруванні векторів, які є контекстно обізнаними та структурно узгодженими по всьому регіону зображення. Більше того, як було видно з розділу 1.2, для завдань, які передбачають обробку інформації в різних форматах, моделі-трансформери підходять більше всього.

Однією з передових моделей-трансформерів для задачі сегментації є модель MaskFormer, яка використовує підхід, що вміло перетворює усталені моделі класифікації кожного пікселя в моделі класифікації маски регіону [17].

Модель, схема якої представлена на рисунку 2.2, складається з трьох частин:

– попіксельний модуль: використання принципів згорткових нейронних мереж для генерації рис зображення меншої розмірності, з яких потім обчислюються вектори представлення для кожного пікселя зображення;

– модуль-трансформер: використання стандартної архітектури трансформера-декодера [8] для обчислення векторів представлення

кожного сегменту; варто зауважити, що кількість потенційних сегментів визначається кількістю запитів, які також є предметом навчання;

– модуль сегментації: використання лінійного класифікатора та мультишарового перцептрону для створення вектора представлення кожного сегменту, а також передбачення його класу (варто зауважити, що кількість класів, передбачення для яких робить даний модуль, становить на один більше, ніж кількість класів у навчальному датасеті, так як додається клас «не об’єкт», який присвоюється сегменту, що є лише заднім планом зображення).

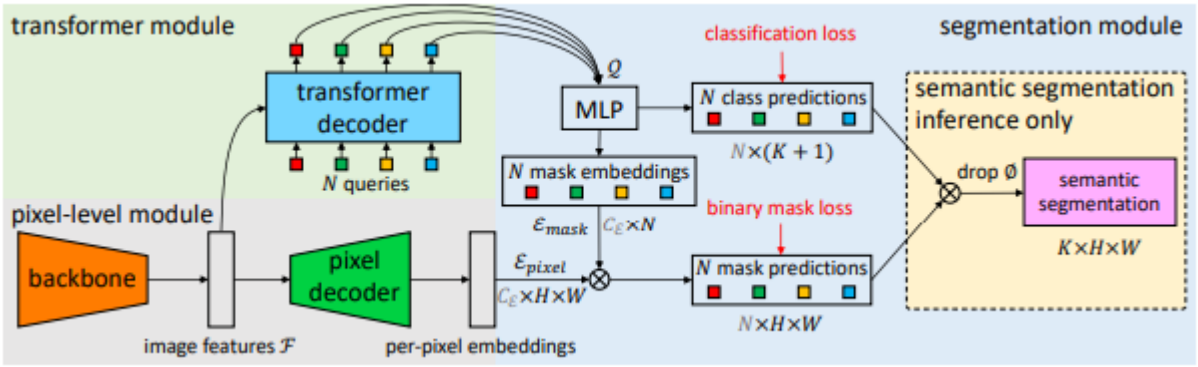

Рисунок 2.2 – Схема моделі сегментації MaskFormer [17]

Але у використанні моделі MaskFormer є обмеження, пов’язане з тим, що вона витягує лише екземпляри визначених класів; MaskFormer не є zero-shot моделлю, які можуть витягувати всі об’єкти зображення.

Таким чином, при виборі певної імплементації моделі варто переконатися, що об’єкти, які вона повинна витягати, входять в перелік її класів.

2.3 Вибір унімодальних моделей отримання векторів представлення

Маючи сегменти екземплярів, можна перейти до наступного кроку запропонованого підходу, а саме отримання векторів представлення тексту

та сегментів зображення окремо з використанням унімодальних моделей. Ці моделі повинні генерувати такі вектори представлення, які зберігають семантичну інформацію, необхідну для порівняння векторів між собою.

Враховуючи вище сказаний критерій, необхідно обрати моделі, в яких порівняння векторів між собою є однією із задач, використаних під час навчання. Для унімодальної моделі обробки текстових запитів буде доцільно використати реченнєві трансформери, а саме SentenceBERT [13].

Така архітектура базується на звичайній BERT-моделі, а навчання відбувається за трьома цілями:

– ціль класифікації: вектори представлення об’єднуються один з одним, а також з їх абсолютною різницею; отриманий вектор множиться з навченими вагами, згідно з формулою:

$$Y = \mathrm{softmax}(W_t(u, v, |u - v|)), \qquad (2.1)$$

де u та v – вектори, що порівнюються;

W_t – навчена матриця ваг розміром $3n \times k$ (n – розмірність вектору представлення, k – кількість класів);

$\mathrm{softmax}(x)$ – активаційна софтмакс-функція;

– ціль регресії: для векторів обчислюється метрика, яка слугує числовою характеристикою «схожості» двох речень;

– ціль триплетів: на відміну від попередніх двох цілей, в ній використовуються три речення: речення-якір, позитивне речення (речення, яке є схожим з реченням-якорем) та негативне речення (речення, яке суттєво відрізняється від речення-якоря); в даній цілі мінімізується відстань між реченням-якорем та позитивним реченням, а також максимізується відстань між реченням-якорем та негативним реченням.

З усіх описаних вище цілей запропонованого підходу найбільше всього стосується ціль регресії, яка детальніше описана на рисунку 2.3.

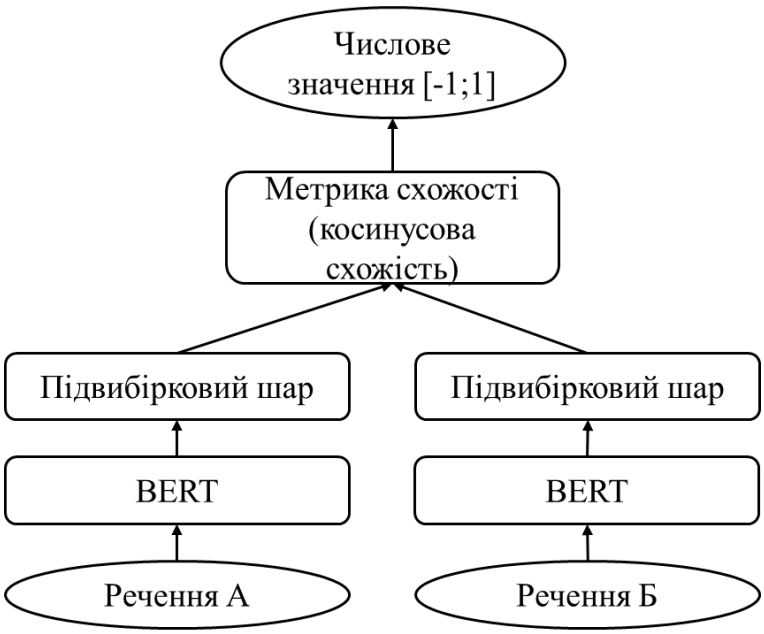

Рисунок 2.3 – Схема використання моделі SBERT для цілі регресії

Таким чином, отримані вектори можна використовувати для встановлення, наскільки два речення «схожі» між собою.

Тепер необхідно обрати аналогічну модель для створення векторів представлення з сегментів екземплярів, що були отримані з моделі сегментації.

Оскільки для обробки текстової інформації було обрано модель, засновану на BERT, для обробки графічних даних доцільно обрати аналог BERT-моделі, який обробляє зображення, наприклад трансформер, поданий у [9]. Він має структуру, схожу з тією, що використовується в BERT та SBERT, а тому можна використати аналогічні принципи.

Наприклад, для отримання вектору представлення всього речення в BERT існує спеціальний токен [CLS], який беруть як один із способів представлення всього тексту.

У візуальному трансформері також є аналогічний токен [class] (на рисунку 2.4 він представлений індексом 0), який можливо взяти для представлення всього зображення.

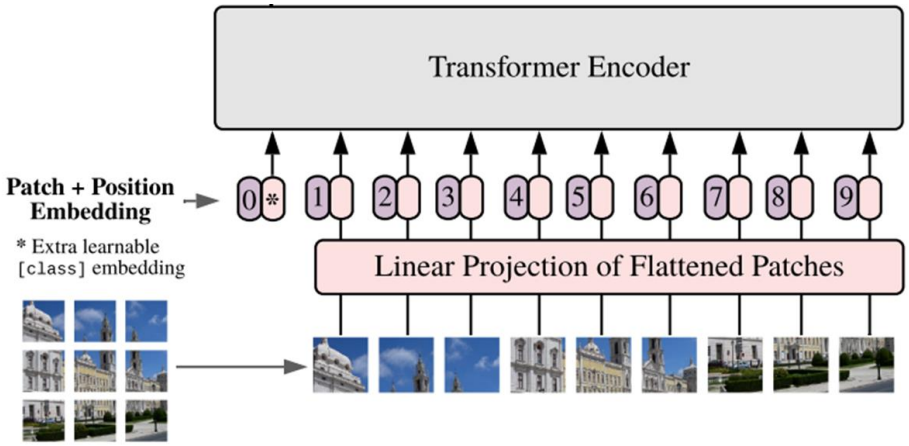

Рисунок 2.4 – Схема візуального трансформера без голівки класифікації [9]

Але у використанні такого трансформера існує певний ризик, пов’язаний із ціллю навчання даної моделі, так як в дослідженні трансформер використовувався для класифікації, чого може бути недостатньо для представлення семантично багатого для порівняння вектора. Тому використання такої моделі може призвести до провального на практиці підходу вирішення поставленої задачі. В такому випадку буде доцільно:

– замість вектора представлення токену [class] брати середнє значення векторів представлення всіх інших токенів (можливо, вони міститимуть у собі інформацію, необхідну для порівняння);

– використати модель іншої імплементації, яка явно навчалася на задачі регресії або триплетів, описаної вище;

– донавчити модель на задачі регресії або триплетів.

2.4 Вибір підходів приведення векторів до спільного простору та аспектів навчання відповідних елементів

Після проходження сегментів екземплярів та текстового запиту через візуальний та реченнєвий трансформери відповідно, буде отримано вектори представлення, які мають зберігати інформацію, що можна використати для

порівняння, але в чистому вигляді вони не порівнюються, так як різні моделі виводять вектори в різних векторних просторах. Тому далі необхідно розробити певний підхід приведення до спільного простору.

Надалі буде розглянуто усі аспекти створення даного підходу та принципи його подальшої практичної реалізації. Так як підходи базуватимуться на засадах створення певної моделі машинного навчання, необхідно спочатку обрати датасет, на якому можливо практично дослідити успіх запропонованих підходів, розробити схему попередньої обробки даних, а також обрати метрики, які можна застосувати для перевірки точності даних підходів.

2.4.1 Вибір датасету для навчання та його попередня обробка

Перш ніж обрати датасет для навчання моделі приведення векторів представлення у спільний простір, необхідно розробити певні вимоги, які описуватимуть характеристики обраного датасету, виходячи із поставленої задачі. Перелік вимог, а також їх пояснення, наведено нижче:

– об’єкти в датасеті повинні мати достатню кількість характеристик, за якими їх можна описати: візуальний та реченнєвий трансформери створюють вектори представлення великої розмірності, що мають змогу зберігати інформацію-опис об’єкта з різних сторін; таким чином об’єкти в датасеті повинні бути такими, які природньо мають велику кількість властивостей, за якими вони можуть бути описані;

– об’єкти в датасеті повинні бути схожими за видом (під видом мається на увазі, що об’єкти повинні належати до спільних класів), але при цьому відрізнятися одним або декількома характеристиками: так як вектори представлення можуть зберігати дані про властивості об’єкта, що вони представляють, об’єкти в датасеті повинні належати до одного класу або схожих класів, але мати різні характеристики, щоб вони (а отже і їх вектори представлення) суттєво відрізнялися;

– для кожного об’єкта в датасеті повинна зберігатися інформація про його регіон: так як першою і головною задачею загального запропонованого підходу є розробка алгоритму сегментації, датасет повинен мати еталонні регіони для кожного екземпляра; і хоча сам процес сегментації не є ціллю навчання, перевірка обраної моделі сегментації на аспект точності виведення сегментів є також важливою.

Існує два підходи вибору датасету, а саме:

– вибір великого датасету з різноманіттям об’єктів, наприклад Microsoft COCO [2];

– вибір меншого, більш вужчого за кількістю типів об’єктів, датасету.

Перший підхід вибору датасету не є оптимальним для поставленої задачі, так як велике різноманіття типів не може гарантувати, що об’єкти одного типу будуть достатньо різними один від одного. Також, підрозділ 1.2.2 показав, що можливо створити лінійний принцип приведення векторів до спільного простору, тому малий розмір датасету не є недоліком. Таким чином, слід обрати датасет, який спеціалізується на окремому підтипі об’єктів.

Враховуючи все вище сказане, для навчання моделі приведення векторів представлення у спільний простір було обрано The Oxford-IIIT Pet Dataset, який містить у собі зображення котів та собак [18]. Даний датасет є ідеальним, оскільки:

– тварин в датасеті можна описати багатьма властивостями: вік, порода, колір хутра, властивості обличчя тощо;

– маючи велику кількість описових властивостей, тварини в датасеті відрізняються принаймні однією з них;

– датасет має також анотацію регіонів, які є ідеальними для перевірки обраної моделі сегментації екземплярів.

Загалом, датасет містить зображення 2371 кота та 4978 собак, які поділяються на 12 та 25 різних порід відповідно. Кожна порода

представлена в датасеті двомастами зображеннями, окрім винятків, наведених в таблиці 2.1.

Таблиця 2.1 – Винятки кількості зображень порід котів та собак у датасеті

Порода кота чи собаки	Кількість зображень
Боксер	199
Англійський кокер-спанієль	196
Кеесхонд	199
Ньюфаундленд	196
Шотландський тер’єр	199
Стаффордський бультер’єр	189
Абіссінська	198
Британський короткошерста	184
Мейнкун	190
Сіамський	199

Таким чином, датасет можна вважати збалансованим відносно порід, але незбалансованим відносно видів тварин: зображень собак приблизно вдвічі більше, ніж зображень котів.

Хоча датасет і має еталонні сегменти екземплярів, він не має їх текстових описів, що також є невід’ємною частиною поставленої задачі. Тому попередня обробка датасету в основному полягатиме у створенні описів природною мовою. Очевидно, їх ручне створення є найбільш ідеальним способом, але велика кількість рисунків та часове обмеження не дають такої змоги. Тому для створення даних описів будуть застосовані уже розроблені моделі машинного навчання, які можуть обробляти текст та зображення та видавати текст природною мовою.

Процес створення текстових описів доцільно розбити на два етапи. Спочатку необхідно обробити зображення та витягнути з нього опис

окремих характеристик, застосувавши для цього модель VLM. Перелік властивостей, якими можна описати тварин, наведено нижче:

– тип: чи є тварина котом чи собакою;

– вік: текстовий опис віку тварини;

– колір: колір хутра або шкіри (для тварин без хутра);

– поза: опис пози, в якій тварина стоїть на зображенні;

– дія: дія, яку тварина виконує на зображенні;

– обличчя: опис різних характеристик обличчя тварини, такі як колір очей, розмір та форма вух тощо;

– емоція: текстовий опис емоційного стану тварини.

Другим етапом є створення опису природною мовою на основі характеристик. Так як для цього не потрібно обробляти саме зображення, можливе використання LLM моделей, вони є унімодальними та більш спроможними для створення зрозумілих описів.

Оскільки використання статистичних моделей ШІ для створення опису може створити такі, які мають спільну структуру (так як моделі створення тексту природної мови базуються на спільному принципі послідовного створення токенів, який полягає в обчисленні ймовірностей, що кожен окремий токен в словнику буде наступним, та вибору токену словника з найбільшою ймовірністю), слід також застосувати засоби аугментації описів для створення більш різноманітних текстових даних.

У [19] розглядається аугментація тексту на основі чотирьох підходів:

– заміна синонімами: деякі слова в реченні будуть замінені відповідними синонімами;

– випадкова вставка: в певному положенні в реченні буде вставлене певне слово;

– випадкова заміна: положення двох випадкових слів буде змінене між собою;

– випадкове видалення: деяке слово в реченні буде видалене.

Автори дослідження також розглядають процес вибору двох параметрів: гіперпараметру α, який позначає частку слів, що будуть піддаватися аугментації, та кількості речень n_{aug}, створених для одного і того ж об'єкта. На основі експериментів автори дослідження створили рекомендовані значення параметрів, що залежать від розміру навчальної вибірки та приведені в таблиці 2.2. Варто зауважити, що самі числові значення параметрів на основі таблиці буде обрано під час практичної реалізації процесу навчання.

Таблиця 2.2 – Рекомендовані значення параметрів аугментації тексту

Розмір навчальної вибірки	α	n_{aug}
500 або менше об'єктів	0,05	16
Від 500 до 2000 об'єктів	0.05	8
Від 2000 до 5000 об'єктів	0,1	4
Більше 5000 об'єктів	0,1	4

2.4.2 Приведення векторів до спільного простору шляхом створення матриць трансформацій

Перший принцип приведення векторів представлення до спільного векторного простору буде заснований на методі розв'язання схожої проблеми обробки природної мови, а саме машинний переклад слів різними мовами. Слова в різних мовах матимуть різні вектори представлення, тому постає задача створення такої матриці трансформації R, для якої справджується наступна рівність:

$$Y \approx RX, \tag{2.2}$$

де X та Y є векторними представленнями одного слова в різних мовах.

Варто зауважити, що після приведення вектори не обов’язково повинні бути однаковими – головне, щоб вони були якомога ближче один до одного (саме тому у формулі 2.2 було використано знак «приблизно дорівнює»). Сам процес навчання формально можна представити у вигляді схеми, яка зображена на рисунку 2.5 (у схемі параметр ε позначає допустиму похибку, а α – гіперпараметр навчання). Нескладно помітити, що даний алгоритм є примітивним алгоритмом навчання за допомогою градієнтного спуску.

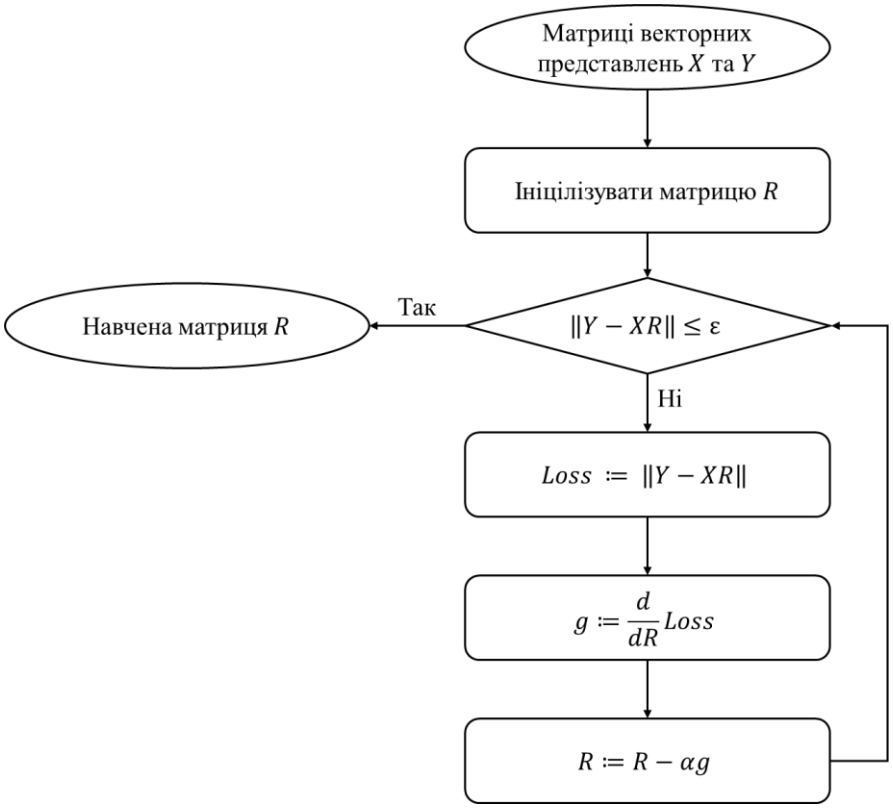

Рисунок 2.5 – Схема отримання матриці трансформації для задачі машинного навчання

Даний алгоритм навчання матриці трансформації доцільно застосувати і для задачі приведення векторів до спільного простору. Але при цьому варто зробити декілька змін. По-перше, метод градієнтного спуску є

найелементарнішим методом навчання; він простий в застосуванні, але має декілька недоліків:

– даний метод повільно сходиться та може «коливатися» навколо оптимуму;

– успіх методу сильно залежить від гіперпараметру швидкості навчання: якщо вона замала, сходження відбувається дуже повільно, якщо завелика – метод може не зійтись;

– швидкість навчання буде однаковою для всіх вимірів навчального простору, що буде проблемою, якщо в одному вимірі треба зробити більший «крок».

Для вирішення цих недоліків буде доцільно використати метод стохастичної оптимізації Adam [20], який вводить так звані величини моменту першого та другого порядків. Навчальні параметри при цьому змінюються за наступними формулами:

$$m_t = \frac{\beta_1 m_{t-1} + (1-\beta_1) g_t}{1-\beta_1^t}, \tag{2.3}$$

$$v_t = \frac{\beta_2 v_{t-1} + (1-\beta_2) g_t^2}{1-\beta_2^t}, \tag{2.4}$$

$$\theta_t = \theta_{t-1} - \frac{\alpha m_t}{\sqrt{v_t} + \varepsilon}, \tag{2.5}$$

де t – порядковий номер ітерації;

β_1, β_2, α, ε – гіперпараметри експоненціального розпаду моментів першого та другого порядків, швидкість навчання та мала константа відповідно (в якості значень автори рекомендують брати 0,9; 0,999; 0,001 та 10^{-8} відповідно);

θ_t – навчальні параметри на t-ій ітерації;

m_t, v_t – імпульс першого та другого порядку відповідно.

По-друге, як видно на рисунку 2.3, для порівняння векторів представлення в SBERT використовується косинусова «схожість», яка чисельно дорівнює косинусу кута γ між двома векторами $\vec{a}$ та $\vec{b}$ [21]:

$$\cos\gamma = \frac{\vec{a}\vec{b}}{|\vec{a}||\vec{b}|}. \tag{2.6}$$

Тому буде доцільно використати косинусову «схожість» в якості функції втрат для навчання.

В результаті буде отримано єдину матрицю, яка приводить вектори представлення текстових описів до простору векторів представлення зображень. Але також в якості окремого підходу варто спробувати знайти окремі матриці трансформації $R_{\text{тек}}$ та $R_{\text{зоб}}$, щоб привести вектори до спільного штучного простору, де вектори, які представляють одну і ту саму інформацію в різних форматах, були однаковими. Саме тому в цьому випадку варто застосувати в якості функції втрат середньоквадратичну похибку:

$$Loss = \frac{\sum_{i=0}^{N}\sum_{j=0}^{D}\left|Y_{i,j}^{\text{тек}} - Y_{i,j}^{\text{зоб}}\right|^2}{ND}, \tag{2.7}$$

$$Y^{\text{тек}} = R_{\text{тек}}X_{\text{тек}}, \tag{2.8}$$

$$Y^{\text{зоб}} = R_{\text{зоб}}X_{\text{зоб}}, \tag{2.9}$$

де $X_{\text{тек}}$, $X_{\text{зоб}}$ – вектори представлень текстових описів та сегментів зображень відповідно;

N – розмір вибірки;

D – розмірність векторів в спільному просторі.

2.4.3 Приведення векторів до спільного простору за допомогою ортогонального базису та спектральної трансформації

Вище було описано два підходи створення спільного векторного простору, обидва були засновані на створенні матриць трансформації. Варто також спробувати більш складний підхід, заснований на проведених дослідженнях. Для цього було розглянуто [22], в якому пропонується підхід створення векторного простору для процесу навчання з учителем (де треба встановити відображення між вхідними та вихідними даними) та без учителя (де вхідні дані можна природньо поділити на кластери, які і треба знайти).

Процес навчання даного підходу навчання з учителем має наступні етапи:

– для обох векторних просторів з використанням SVD обчислюються матриці власних векторів просторів U_x та U_y [23]; з використанням цих власних векторів обчислюються нові вектори представлення P_x та P_y:

$$P_x = U_x^T X, \tag{2.10}$$

$$P_y = U_y^T Y; \tag{2.11}$$

– нові векторні представлення об’єднуються в один вектор Z з урахуванням ступеня масштабування α; для нового векторного простору також обчислюється матриця власних векторів простору U_z та нові вектори представлення P_z методами, описаними нижче:

$$Z = \begin{bmatrix} P_x \\ \alpha P_y \end{bmatrix}, \tag{2.12}$$

$$P_z = U_z^T Z; \tag{2.13}$$

– наостанок, необхідно знайти матрицю трансформації A_c між векторами представлення P_x та P_y:

$$P_z = A_c P_x. \tag{2.14}$$

Саме матриця A_c є об'єктом навчання в даному підході, який можна знайти за допомогою алгоритму Adam, описаного в підрозділі 2.4.2, та середньоквадратичної похибки згідно з формулою 2.7.

Маючи навчену матрицю A_c, а також матриці власних векторів просторів U_x, U_y та U_z, вектори у спільному просторі можна знайти за принципом, оберненим тому, що було описано вище, а саме:

– обчислити спільний вектор представлення Z'

$$Z' = U_z A_c P_x'; \tag{2.15}$$

– розділити його на два вектори P_x' та P_y';

– обчислити оригінальний вектор представлення Y:

$$Y' = U_y P_y' . \tag{2.16}$$

Варто зауважити, що при розрахунку обернених формул 2.15 та 2.16 було враховано, що матриці власних векторів просторів ортогональні [21], а для такого вектора U справджується наступне:

$$U^T = U^{-1} . \tag{2.17}$$

2.4.4 Вибір метрик для оцінки запропонованих підходів

Наостанок, маючи обраний датасет та підходи вирішення задачі приведення векторів у спільний простір, необхідно обрати метрики, які будуть використані у порівнянні успіху підходів. Для цього варто зосередитися на глобальній задачі, яку вирішують підходи приведення, а саме знаходження семантичної відповідності між текстовою та графічною інформацією. В даному випадку під семантичною відповідністю слід вважати відповідність між зображенням екземпляру та його текстовим описом. Таким чином, необхідно знайти чисельну метрику, яка оцінюватиме успіх знаходження зображення для відповідного опису.

Спочатку розглянемо метрики, які використовуються для задач класифікації, так як окрім текстових описів кожен екземпляр має клас: тип тварини (кіт або собака) та порода. Якщо розглядати задачу з боку вибору екземпляру з правильною міткою типу тварини, задача перетворюється на бінарну класифікацію. Беручи «кіт» як позитивну мітку (це доцільно зробити, так як у підрозділі 2.4.1 було зазначено, що обраний датасет є незбалансованим до класу «кіт»), а «собака» як негативну, успіх майбутньої моделі можливо представити у вигляді матриці помилок, макет якої представленої на рисунку 2.6. Найпоширеніша метрика для задачі класифікації є точність (accuracy), яка чисельно дорівнює відношенню правильно визначених точок до їх загальної кількості в датасеті:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2.18}$$

де TP та TN – кількості точок з правильно визначеними позитивними та негативними мітками відповідно;

FP та FN – кількості точок, для яких модель допустилася помилок.

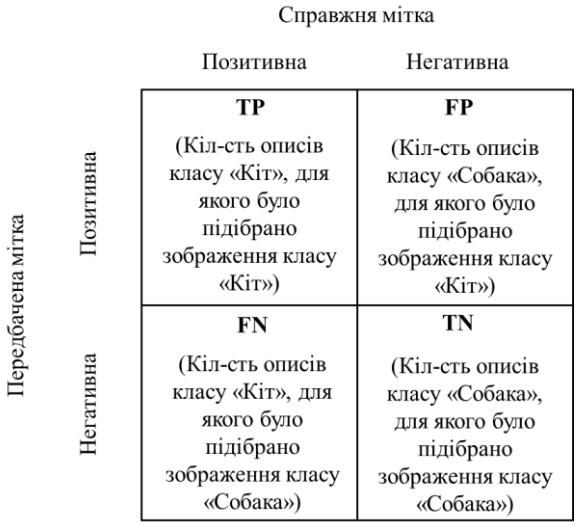

Рисунок 2.6 – Схема матриці помилок бінарної класифікації

Але дана метрика є неоптимальною для незбалансованих датасетів, для них використовують метрику F1-score, яка є середнім гармонічним між чіткістю (precision) та відкликом (recall):

$$\mathrm{F1} = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}; \tag{2.19}$$

$$\mathrm{Precision} = \frac{TP}{TP + FP}; \tag{2.20}$$

$$\mathrm{Recall} = \frac{TP}{TP + FN}. \tag{2.21}$$

Якщо в якості мітки брати породу тварини, задача стає проблемою мультикласової класифікації, для якої можливо побудувати матрицю помилок та обчислити точність (так як відносно порід датасет є збалансованим), для кожної породи окремо також можливо обчислити F1-score за формулами 2.19 – 2.21, беручи в якості позитивної мітки дану породу, а решту порід вважати негативними.

Розгляд метрик класифікації не дає повної можливості аналізу успіху підходу приведення векторів представлення до спільного простору, так як не враховує випадки, коли зображення, підібране до опису, має правильну мітку, але не відповідає опису характеристик екземпляру. В такому випадку можна скористатися методами перевірки «схожості» інформації, яка знаходиться в одному форматі, а саме текстовий опис, зображення для якого треба підібрати, та текстовий опис підібраного зображення, за принципом, схематично описаним на рисунку 2.3. Для отримання загальної метрики доцільно взяти середнє значення всіх чисельних метрик «схожості» для кожної пари текстових описів у вибірці.

2.5 Опис принципу порівняння векторів представлення тексту та зображень

Маючи вектори представлення текстових описів та зображень екземплярів у спільному просторі, останнім кроком запропонованого підходу є порівняння векторів представлення тексту та зображень для отримання одного екземпляру, що відповідає текстовому опису. Загальний алгоритм порівняння можна описати наступним чином:

– для вектору представлення зображення кожного екземпляру необхідно обчислити «схожість», наприклад косинусову, між вектором даного екземпляру та вектором текстового опису;

– обрати екземпляр, для якого «схожість» з текстовим описом є найбільшою;

При цьому також можливо обрати певне порогове значення, при якому екземпляр відповідатиме текстовому опису тільки тоді, якщо «схожість» більше, ніж дане значення. Інакше вважатимемо, що зображення не містить екземпляру, що відповідає опису.

3 ПРАКТИЧНІ ДОСЛІДЖЕННЯ

У попередньому розділі було запропоновано підхід до задачі сегментації екземплярів на основі текстового запиту та було описано кожну з його частин, а також основні положення методів вибору датасету для навчання та оцінки успіху.

Нижче буде викладено загальний порядок практичної реалізації кожного з етапу підходу, при чому особливу увагу буде зосереджено на етапі приведення векторів до спільного простору, так як саме цей етап є частиною підходу, що необхідно створити методами машинного навчання.

Всі практичні дослідження, окрім анотації екземплярів за допомогою статистичних моделей природної мови, було проведено на єдиній обчислювальній машині, яка має процесор Intel одинадцятого покоління, з 16-ма гігабайтами RAM, а також з графічним процесором Nvidia RTX 3050Ti з 4-ма гігабайтами VRAM. А для анотації екземплярів було розгорнуто моделі LLM та VLM на хмарному сервісі Azure AI Foundry.

Для дослідження всі скрипти було написано мовою програмування Python.

3.1 Робота з обраним датасетом та підготовка даних до навчання

Після завантаження датасету була розглянута його структура, до якої входив набір зображень та набір відповідних анотацій. Анотації сегментації представлені у вигляді псевдозображень з трьома каналами (рисунок 3.1), які відповідали за регіон екземпляру, його контури та задній план відповідно.

Породу тварини зазначено у назві зображення, а вид тварини можна визначити за принципом: якщо назва зображення починається з великої літери – тварина є котом, інакше – собакою.

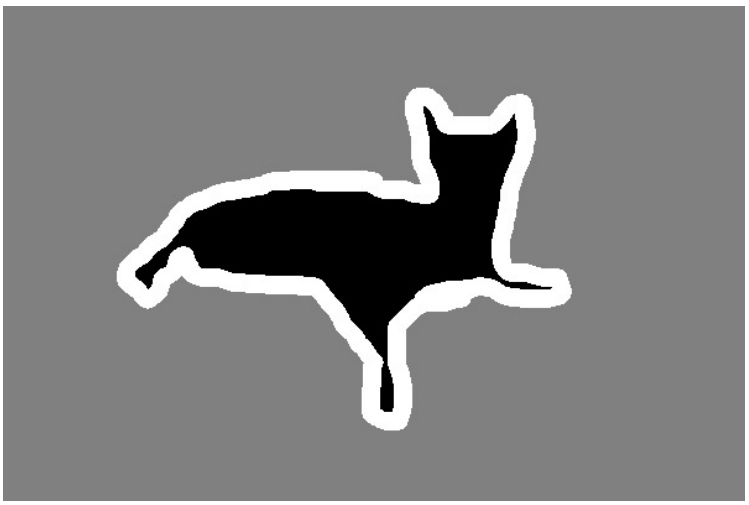

Рисунок 3.1 – Візуальний приклад анотації сегментації

Наступним кроком практичної реалізації було завантаження обраної моделі сегментації екземплярів. В якості імплементації було обрано модель «maskformer-swin-base-coco» від автора «facebook» з набору моделей, представлених на платформі HuggingFace. Оскільки датасет має анотовані сегменти для кожного екземпляра, було проведено аналіз успіху сегментації. Для цього було обрано метрику IoU, яка обчислюється за формулою:

$$\text{IoU} = \frac{|P \cap T|}{|P \cup T|}, \tag{3.1}$$

де P – множина пікселів, що належать до передбаченого сегменту;

T – множина пікселів, що належать до сегменту-анотації.

Програмно, обчислення метрики було реалізовано так, як наведено в лістингу 3.1.

Лістинг 3.1 – Програмна реалізація обчислення IoU

```
# змінна pred_mask містить маску моделі
# gt_mask містить маску-анотації
pred_mask = np.array(best_result["mask"]) == 255
```

Продовження лістингу 3.1

```
pred_mask_size = np.sum(pred_mask)
iou = np.sum(gt_mask & pred_mask) \
      / np.sum(gt_mask | pred_mask)
```

Основні висновки наведено нижче, а гістограму IoU метрик зображено на рисунку 3.2:

– середній IoU для всього датасету становить 0,79;

– середній IoU для зображень котів становить 0,77;

– середній IoU для зображень собак становить 0,80;

– серед усіх зображень модель не змогла знайти жодного екземпляра на 190 зображеннях або на 2,57% кількості зображень всього датасету.

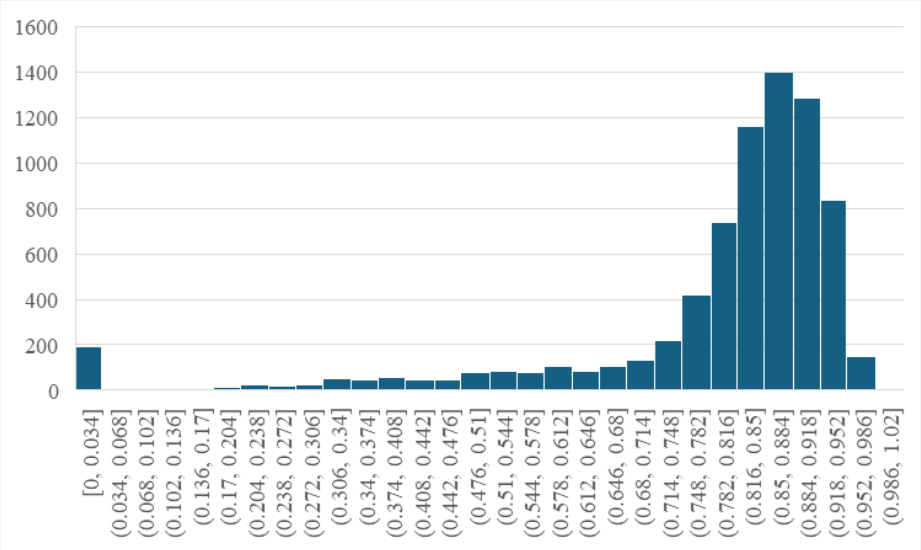

Рисунок 3.2 – Гістограма розподілу метрик IoU

Можемо зробити висновок, що було підібрано вдалу модель, яка вміє знаходити собак та котів з однаковим успіхом.

Після даної перевірки були створені описи кожного екземпляра в датасеті. Спочатку було витягнуто окремі характеристики тварин із екземплярів, які було отримано після запуску сегментаційної моделі. Для

цього було обрано VLM-модель Phi-3.5-vision-instruct, яка може в якості вхідних даних приймати зображення й текст та повертати відповідь природною мовою (рисунок 3.3).

Рисунок 3.3 – Порівняння моделі Phi-3.5-vision-instruct з іншими схожими моделями Azure AI Foundry

Для успішного виконання моделлю природної мови задачі необхідно створити хороший системний запит, тобто запит, в якому описуються інструкції. Згідно з [24], в ньому необхідно зазначити:

– роль моделі, де описано тип задачі, що необхідно виконати;

– основні інструкції, в якому описано всі аспекти та процедури, як само необхідно вирішити задачу;

– обмеження, в яких сказано, чого треба уникати;

– формат вихідного результату.

Для виконання задачі витягання характеристик із зображень було створено наступний системний запит, перекладений українською: «Тобі

буде надано зображення тварини. Надай його детальний опис, а також його виду та рис. Ти повинна надавати результат тільки у форматі словника JSON. Даний словник повинен включати наступні риси:

– тип: тип тварини;
– вік: чи є тварина дитиною, чи дорослим;
– колір: точний опис окрасу тварини;
– поза: опис пози тварини;
– дія: дія, яку тварина виконує на зображенні;
– обличчя: детальний опис рис обличчя, включно з описом очей та вух тварини;
– емоція: емоція, яку тварина виражає.

Якщо ти не можеш надати опис деяких з характеристик, зазначених вище, дай значення null цим атрибутам». В даному запиті надані інструкції створювали результат у форматі JSON словника з метою представлення різних характеристик у зрозумілому як для людини, так і для моделі природної мови форматі. Даний словник було оброблено за допомогою функції-помічника, код для якої представлено в лістингу 3.2.

Лістинг 3.2 – Код функції-помічника для обробки результату VLM-моделі

```
def clean_response(text: str) -> str:
     text = text.replace("'", '"')
     open_bracket = text.find("{")
     close_bracket = len(text) - text[::-1].find("}")
     if open_bracket == -1 \
     or close_bracket == 1 \
     or open_bracket >= close_bracket:
          return text
     else:
          trimmed_text = text[open_bracket:\
          close_bracket + 1]
          return trimmed_text
```

Після отримання характеристик було проведено оцінку якості результатів. Серед проблемних зон можна виокремити наступні:

– некоректний формат: модель в якості результату видавала словник, який неможливо автоматично перетворити в JSON формат через наступні причини: відсутність фігурних дужок, існування додаткових символів (наприклад, коментарів, які не підтримуються форматом), невірне використання лапок;

– ігнорування інструкції про формат відповіді: інколи модель надавала результат у форматі природної мови, здебільшого це були деякі галюцинації, тобто текст, який не має логічного чи фактичного сенсу;

– додавання сторонніх характеристик: інколи модель додавала до словника ознаки, яких не було в переліку, або в якості значення деяких коректних ознак створювала додатковий список.

Загалом серед 7389 екземплярів, отриманих від моделі сегментації, 6381 екземпляр був придатним для подальшої обробки. Розподіл ознак, які були присутні для кожного екземпляру, наведено на рисунку 3.4.

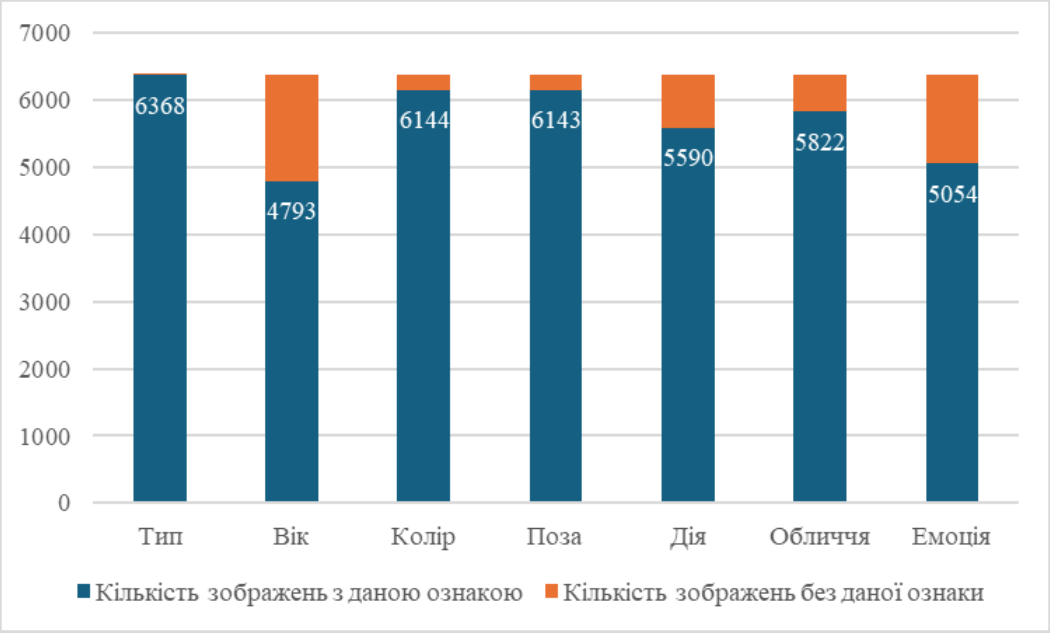

Рисунок 3.4 – Розподіл ознак екземплярів

Наступним кроком є створення описів на основі витягнутих характеристик. Для цього було використано LLM-модель Phi-4, що є провідною моделлю, навченою на основі поєднання синтетичних датасетів, даних від вручну фільтрованих публічних доменів та академічних книг. Вона не поступається в точності іншим моделям, але завдяки меншому розміру (14 мільярдів параметрів) досягає вищої швидкості генерації нових токенів, що показано на рисунку 3.5.

Рисунок 3.5 – Порівняння моделі Phi-4 з іншими схожими моделями Azure AI Foundry

Враховуючи принципи створення системних запитів, зазначені вище, для моделі було створено наступний системний запит, що описує підхід до створення опису для кожного екземпляра: «Ти асистент магазину домашніх тварин. Твоя робота полягає в написанні описів тварин на фото. Тобі буде надано словник у форматі JSON з основними рисами тварини. Ти повинен

вивести опис тварини, лише спираючись на наданий опис. Ти повинен виводити лише опис тварини. Твій опис повинен бути не більше, ніж три речення». Такий запит дає вказівки щодо характеру роботи та дає обмеження щодо результату.

Далі необхідно перейти до процесу створення векторів представлення, на основі яких і буде навчено моделі машинного навчання для приведення до спільного векторного простору. Дану задачу слід поділити на дві: створення векторів для зображень екземплярів та текстових описів.

Почнемо зі створення векторів для екземплярів. Як було описано в розділі 2.3, для них буде використано блок звичайного візуального трансформеру, а саме модель «vit-base-patch16-224-in21k» від автора «google» з набору моделей, представлених на платформі HuggingFace. Для зберігання векторів було створено скрипт, який представлений у лістингу 3.2. Дана модель створює вектори представлення розмірністю 768.

Варто зауважити, що з останнього прихованого стану (властивість «last_hidden_state») витягується вектор з індексом [0, 0], що означає вектор першого екземпляру (так як функція пропускає зображення один за одним послідовно) першого токена (так як саме перший токен є спеціальним токеном [class] на рисунку 2.4, що зберігає загальну інформації про зображення).

Лістинг 3.3 – Функція для створення векторів представлення сегментів екземплярів

```
def main(image_folder: str, embedding_folder: str):
    # PROCESSOR зберігає модель для передобробки зображень
    # MODEL зберігає модель візуального трансформера
    image_folder_path = Path(image_folder)
    embedding_folder_path = Path(embedding_folder)
    all_images = list(image_folder_path.glob("*"))
    for image_file in tqdm(all_images):
        name = image_file.name[:-4]
```

Продовження лістингу 3.3

```
image = Image.open(image_file)
gray_image = image.convert("L")
bbox = gray_image.getbbox()
cropped_image = image.crop(bbox)
processor_params = {
     "images": cropped_image,
     "return_tensors": "pt"
}
inputs = PROCESSOR(**processor_params)
outputs = MODEL(**inputs)
embedding = outputs\
     .last_hidden_state[0, 0]
embedding = embedding.cpu().detach().numpy()
file_name = embedding_folder_path / f"{name}.npy"
np.save(file_name, embedding)
```

Далі необхідно створити вектори для текстових описів. Але спочатку треба розділити датасет на навчальну, валідаційну та тестову вибірки. Для створення вибірки було використано функцію «train_test_split» бібліотеки «scikit learn», яку було викликано двічі: для поділу на навчальну вибірку та таку, яку модель не бачитиме під час навчання, та подальший поділ другої на валідаційну та тестову. Поділ був стратифікований по відношенню до порід, тобто кожна вибірка мала приблизно однакову кількість зображень однієї породи.

Слід зауважити, що розподіл зображень у навчальній, валідаційній та тестовій вибірках складає 4356, 933 та 934 зображення відповідно.

Тепер можна перейти до створення векторів представлення для текстових описів. Для опису тварин валідаційної та тестової вибірки вектори було створено на основі тексту без змін, а для зображень тварин навчальної вибірки – з використанням аугментації описів, принципи якої було описано в підрозділі 2.4.1.

З урахуванням інформації, поданої в таблиці 2.2, та кількості екземплярів у навчальній вибірці, було обрано в якості значень параметрів аугментації α та n_{aug} значення 0,1 та 4 відповідно. Практична реалізація аугментації мала один нюанс: оскільки вік був одним із характеристик опису тварини, дуже часто в текстах зустрічалося слово «adult» (в даному контексті переклад українською – «дорослий»). Дане слово має декілька визначень, найбільш підходяще з яких: людина чи тварина, яка виросла до повного розміру та сили (синонімами для такого визначення будуть слова «grown-up», «mature» тощо). Але слово має і інше визначення: фільми, журнали та книжки для дорослих, які показують оголених людей та статеві акти, і не є підходящими для дітей (синонімом для такого визначення буде слово «pornographic»).

Реалізація аугментації заміни синонімами, що базується на усталених словниках, користувалася саме другим поняттям, що не підходить для даної задачі, тому було використано іншу реалізацію, що підбирає синоніми на основі семантичного контексту для більш чіткого розуміння та визначення правильного пояснення слів.

Після створення аугментованих текстів для створення векторів представлення було використано модель «all-MiniLM-L6-v2» сімейства «sentence-transformers», які створюють вектори розмірності 384, та написано відповідну функцію, подану в лістингу 3.4.

Лістинг 3.4 – Функція для створення векторів представлення текстових описів

```
def main(
     prompt_folder: str,
     embedding_folder: str,
     split_file: str
):
     # MODEL зберігає модель реченнєвого трансформера
     prompt_folder_path = Path(prompt_folder)
```

Продовження лістингу 3.4

```
folder_path = Path(embedding_folder)
with open(split_file, 'r') as f:
          train_files = json.load(f)['train']
     all_prompts = list(prompt_folder_path.glob("*"))
     for prompt_file in tqdm(all_prompts):
          name = prompt_file.name[:-4]
          with open(prompt_file, 'r') as file:
               prompt = file.read().strip()
          if name in train_files:
               embedding = []
               for _ in range(4):
                    prompt_aug = augment_prompt(prompt)
                    encoding = MODEL.encode(
                         prompt_aug,
                         convert_to_tensor=True
                    ).cpu().numpy()
                    embedding.append(encoding)
               embedding = np.stack(embedding, axis=0)
          else:
               model_params = {
                    "sentences": prompt,
                    "convert_to_tensor": True
               }
               embedding = MODEL.encode(**model_params)
               embedding = embedding.cpu().numpy()\
                    .reshape(1, -1)
          embed_file = folder_path / f"{name}.npy"
          np.save(embed_file, embedding)
```

Наприкінці було отримано безпосередні вектори представлення для зображень та описів, які мають відповідні назви, тому маємо змогу перейти до процесу навчання підходів приведення векторів до спільного простору.

3.2 Створення єдиної матриці трансформації

Тепер варто переходити до процесу безпосереднього навчання, починаючи з підходу створення єдиної матриці трансформації. Для цього буде використано фраємворк PyTorch та його доповнення PyTorch Lightning, що прискорює процес написання коду, інкапсулюючи частини, які є спільними для всіх процесів навчання. Перш ніж запускати даний процес, необхідно створити екземпляр класу датасету та його завантажувач, який розбиває весь набір даних на пакети.

Для створення датасету достатньо створити власний клас, який наслідується від класу Dataset. В ньому необхідно перевантажити два методи: «__len__», який рахує загальний розмір вибірки, та «__getitem__», який повертає готову для навчання єдину точку вибірки за заданим індексом. Перевантаження останнього методу представлено на лістингу 3.5; дана функція повертає кортеж двох масивів – векторів представлення текстового опису та зображення.

Лістинг 3.5 – Перевантаження методу «__getitem__»

```
def __getitem__(self, index):
     file_name = f"{self.names_list[index]}.npy"
     img_embed = np.load(self.images / file_name)
     prompt_embeds = np.load(self.prompts / file_name)
     num_prompt = randint(0, prompt_embeds.shape[0] - 1)
     prompt_embed = prompt_embeds[num_prompt]
     dtype=torch.float32
     return torch.tensor(prompt_embed, dtype=dtype),\
          torch.tensor(img_embed, dtype=dtype)
```

Для навчання необхідно створити клас моделі, який наслідується від класу «LightningModule» та повинен мати наступні методи:

– «forward», в якому необхідно імплементувати прямий хід моделі;

– «configure_optimizers», в якому необхідно зазначити, який оптимізатор використовується (тут буде використано оптимізатор Adam);

– «training_step» та «validation_step», в яких описано, як саме має відбуватися процес навчання та валідації.

Код для даних методів однаковий для всіх підходів, його детальніше представлено у додатку А. Тепер слід розглянути матрицю трансформації, яку необхідно створити. Дана матриця R буде використана так, як представлено на формулі 2.2, тому, оскільки матриця векторів представлення текстових описів X має розмірність n x 384 (n – кількість векторів), а матриця векторів представлення зображень Y – n x 768, то матриця трансформації R повинна мати розмір 384 x 768. В такому випадку в програмній реалізації треба ініцілізувати матрицю трансформації наступним чином:

```
nn.Parameter(torch.randn(input_dim, output_dim))
```

А метод «forward» матиме наступний вид:

```
def forward(self, x):
    return torch.matmul(x, self.transformation_matrix)
```

На рисунку 3.6 показано криву навчання, тобто зміну функції втрат з плином часу (синя лінія позначає криву на навчальній вибірці, помаранчева – на валідаційній). З даної кривої видно, що навчання, яке в цілому тривало 236 епох, було стабільним.

Після навчання слід перейти до обчислення метрик на тестовій вибірці для перевірки успіху моделі. Для цього спочатку варто створити клас-інтерфейс для отримання результату від моделей, в якому слід визначити два методи для повернення оброблених векторів представлення тексту та зображень окремо. Дані методи, програмну реалізацію яких подано у лістингу 3.6, повертатимуть вектори без обробки, а в подальшому клас

кожного методу окремо наслідуватиме клас-інтерфейс, в якому за потреби буде перевантажено методи обробки векторів.

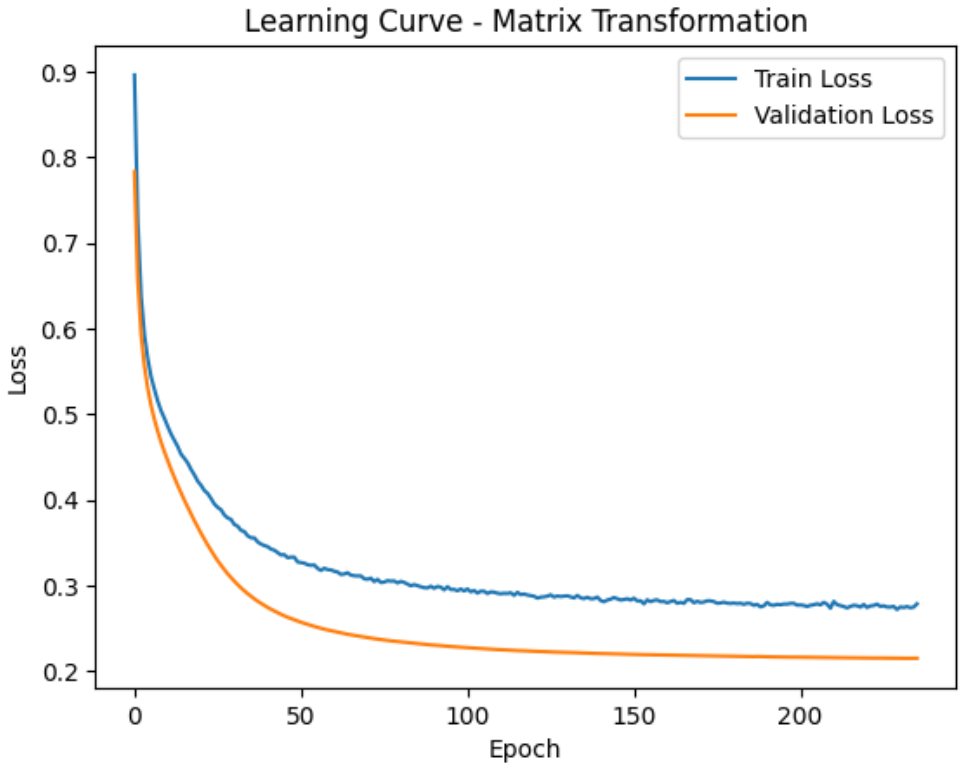

Рисунок 3.6 – Крива навчання для підходу створення єдиної матриці трансформації

Лістинг 3.6 – Код класу-інтерфейсу для отримання результату від моделей приведення векторів до одного простору

```
class InferencePipeline:
    def __init__(self):
        pass

    def encode_image(self, embedding: np.ndarray):
        return embedding

    def encode_text(self, embedding: np.ndarray):
        return embedding
```

Тепер для даного методу слід створити клас, в якому буде перевантажено метод опрацювання текстових даних, де кожен вектор множитиметься на матрицю трансформації:

```
def encode_text(self, embedding):
    return np.dot(embedding, self.matrix)
```

Створений клас можливо використати для отримання метрик успіху підходу приведення векторів до спільного простору шляхом створення єдиної матриці трансформації. Детальний код отримання даних метрик та графіків наведено у додатку Б. З матриці помилок на рисунку 3.7 видно, що модель ідеально розрізнила зображення котів від зображень собак. Також, обрахувавши метрики F1-score для кожної породи, можна помітити, що модель здебільшого змогла ідеально відділити породи одна від одної з деякими винятками, які показані на рисунку 3.8.

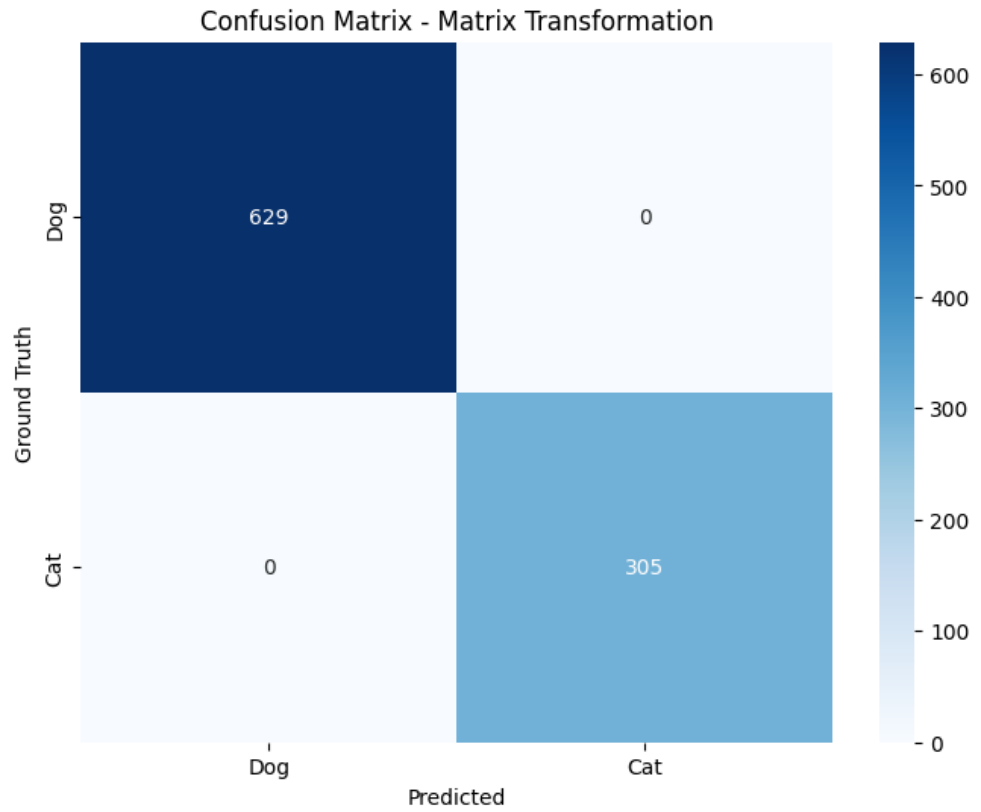

Рисунок 3.7 – Матриця помилок для єдиної матриці трансформації

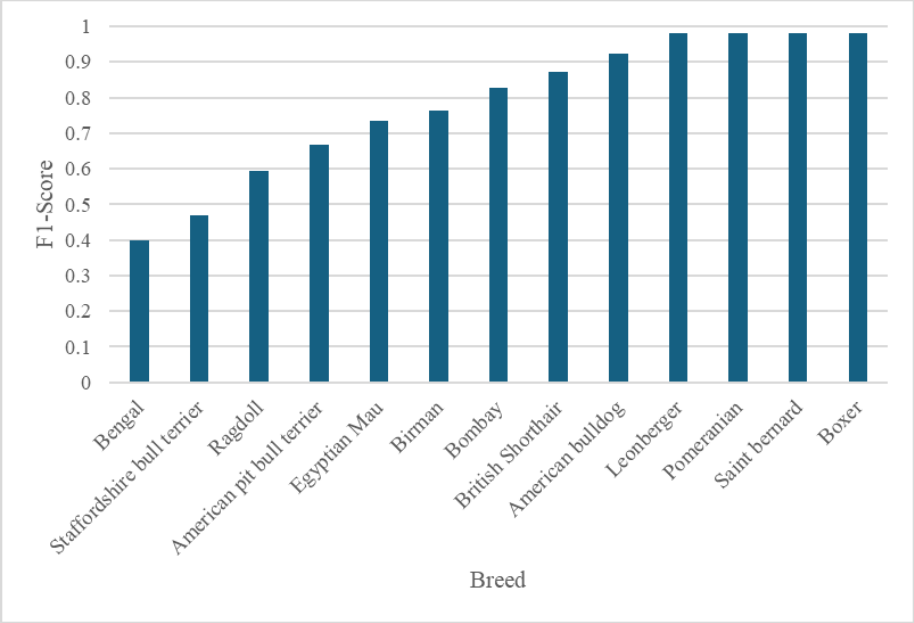

Рисунок 3.8 – Метрики F1-score для кожної породи, в яких дана метрика для єдиної матриці трансформації не дорівнювала одиниці

З рисунку 3.8 видно, що є декілька порід, з якими модель впоралася найгірше. Розглянемо кожну з даних порід та результати окремо. Даний аналіз представлено в таблиці 3.1.

Таблиця 3.1 – Аналіз порід, з якими модель єдиної матриці трансформації погано впоралася

Порода	F1-score	Аналіз помилок
1	2	3
Бенгальський кіт	0,4	Більшість зображень плуталися з породою кота єгипетський мау, при чому лише з одним зображенням. Решта зображень плуталися з зображеннями тієї ж самої породи.
Стаффордський бультер’єр	0,47	Модель плутала зображення собак порід стаффордського та американського бультер’єрів.

Продовження таблиці 3.1

1	2	3
Реґдол	0,54	Модель плутала зображення котів порід реґдол та бірмен.
Американський пітбультер'єр	0,67	Помилки, аналогічні помилкам для стаффордського бультер'єру.
Єгипетський мау	0,74	Помилки, аналогічні помилкам для бенгальського кота.
Бірмен	0,76	Помилки, аналогічні помилкам для реґдола.

Наостанок, на рисунку 3.9 зображено розподіл метрик «схожості» між еталонними описами та описами зображень, які були найближчими до тексту-запиту. Середнє значення та медіана такої метрики становлять 0,8 та 0,81 відповідно, з чого можна зробити висновок про успіх застосування підходу приведення векторів до спільного простору шляхом створення єдиної матриці трансформації.

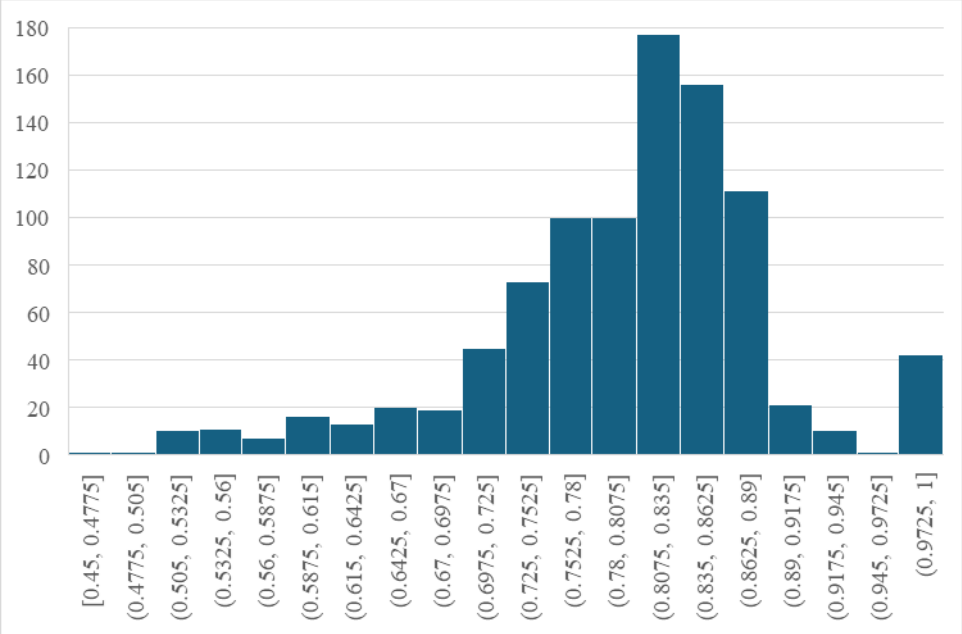

Рисунок 3.9 – Розподіл метрик «схожості» для моделі матриці єдиної трансформації

3.3 Створення окремих матриць трансформацій для обох векторних просторів

Після створення єдиної матриці трансформації слід перейти до другого підходу приведення векторів представлення до спільного простору, в якому для обох просторів векторів текстових описів та зображень екземплярів створюється окрема матриця трансформації. В результаті усі вектори приведені до штучно створеного простору.

Спочатку необхідно визначитися з розмірами матриць. Так як створюється третій векторний простір, маємо вибір його розмірності. При виборі було враховано наступне:

– для проміжних шарів у нейронних мережах зазвичай беруть розміри, які є степенями двійки, щоб використовувати усю можливу внутрішню пам’ять машини, на яких проводяться навчання та обробка;

– слід взяти такий розмір, який був меншим, ніж розмір найбільш розмірного простору. Таким чином можна зменшити розмірність багато розмірного векторного простору без втрати семантичної інформації.

Враховуючи вище перелічене, в якості розміру нового простору було взято значення 512. Враховуючи це, а також розміри уже існуючих просторів, матриці трансформацій $R_{\text{тек}}$ та $R_{\text{зоб}}$ матимуть розміри 384 x 512 та 768 x 512 відповідно.

Після визначення розміру матриць варто перейти до створення класу даної моделі, як показано у додатку А, де треба перевантажити метод «forward», який поданий в лістингу 3.7.

Лістинг 3.7 – Код перевантаженого методу «forward» для підходу створення окремих матриць трансформації

```
def forward(self, prompt, img):
    y_img = torch.matmul(img, self.image_matrix)
    y_prompt = torch.matmul(prompt, self.prompt_matrix)
    return y_prompt, y_img
```

На рисунку 3.10 подано криву навчання для даної моделі. Порівнюючи її з кривою навчання методу створення єдиної матриці, яку подану на рисунку 3.6, бачимо, що початкове значення функції втрат значно більше, ніж у попередньому методі, що можна пояснити різними функціями (при створенні єдиної матриці трансформації було використано косинусову «схожість», а при створенні окремих матриць – середньоквадратичну похибку). Також навчання було майже вдвічі довшим: навчання даного підходу тривало 462 епохи.

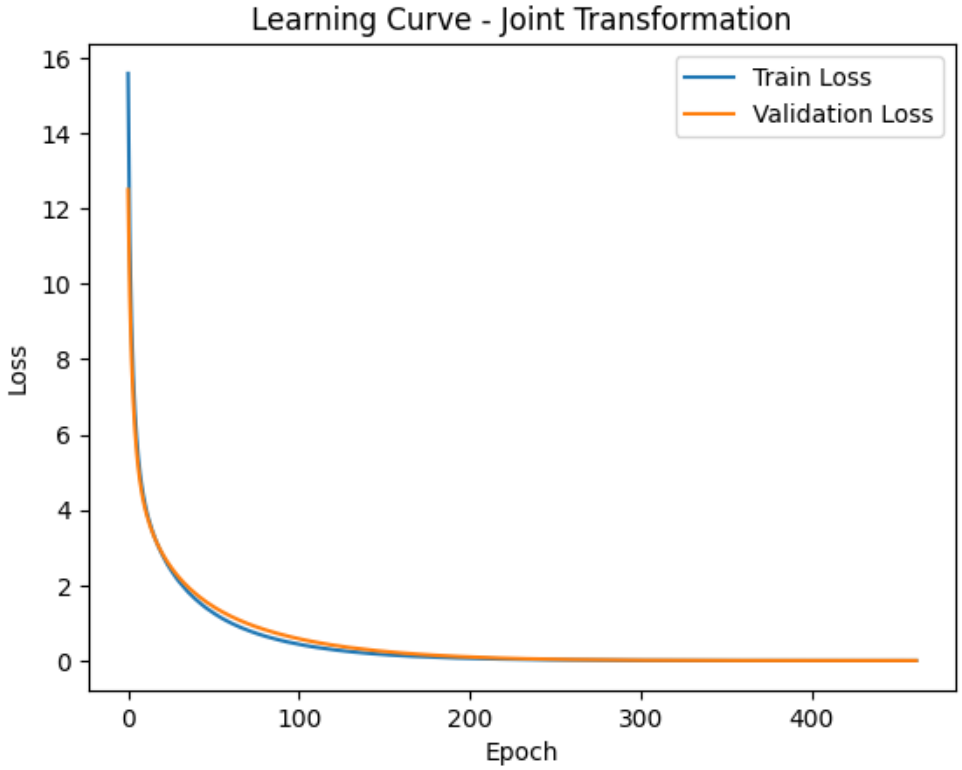

Рисунок 3.10 – Крива навчання для підходу створення окремих матриць трансформацій

Аналізуючи матрицю помилок на рисунку 3.11, уже можемо побачити, що навчання було неуспішним, так як здебільшого всі передбачені зображення є собаками (це ймовірно пов’язано з тим, що клас «собака» сильно переважає в розмірі клас «кіт»).

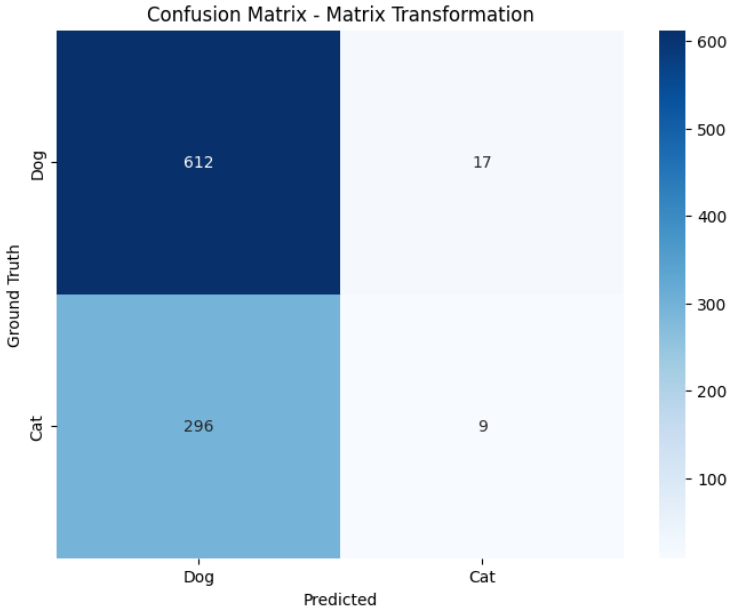

Рисунок 3.11 – Матриця помилок для нового штучного векторного простору

А з розподілу метрик «схожості» між еталонними описами та описами зображень (середнє значення та медіана становлять 0,52 та 0,49 відповідно), представленого на рисунку 3.12, стає очевидно, що даний підхід не зміг створити векторний простір, в якому можливий пошук семантично схожої інформації.

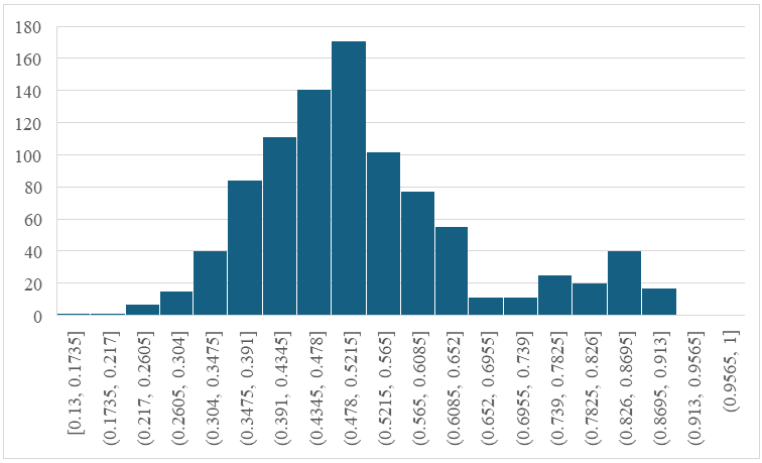

Рисунок 3.12 – Розподіл метрик «схожості» для моделі окремих матриць трансформацій

Немає сенсу проводити подальший аналіз результатів, так як він не принесе жодної дослідницької користі. Тому варто перейти до наступного підходу приведення векторів представлення до спільного векторного простору.

3.4 Приведення векторів до спільного простору за допомогою ортогонального базису та спектральної трансформації

Останній підхід приведення до спільного векторного простору буде зроблений за допомогою ортогонального базису та спектральної трансформації, що було розглянуто з теоретичної частини у підрозділі 2.4.3.

Спочатку необхідно обчислити власні вектори просторів U_x та U_y, а також власні вектори спільного простору U_z (лістинг 3.8). Для цього буде використано всю навчальну вибірку одразу (це можливо зробити, так як розмір вибірки відносно малий). Матриці власних векторів буде обчислено за допомогою SVD з використанням модуля методів лінійної алгебри «linalg» стандартної бібліотеки «NumPy». Варто зауважити, що вектори представлення текстових описів буде помножено на чотири, так як довжина даних векторів в середньому у чотири рази менша, ніж довжини векторів представлення зображень екземплярів.

Лістинг 3.8 – Код обчислення матриць власних векторів просторів

```
x = np.array(x).T
y = np.array(y).T
u_x, _, _ = np.linalg.svd(x, full_matrices=True)
u_y, _, _ = np.linalg.svd(y, full_matrices=True)
p_x = np.matmul(u_x.T, x)
p_y = np.matmul(u_y.T, y) * 4
z = np.concatenate((p_x, p_y), axis=0)
u_z, _, _ = np.linalg.svd(z, full_matrices=True)
```

Згідно з формулою 2.14, матриця трансформації A_c є тією, яку треба створити за допомогою методів машинного навчання. Тому для цього було розроблено клас моделі, де в перевантаженому методі «forward» відбувається обчислення всіх проміжних матриць у формулах 2.10–2.13; дану програмну реалізацію представлено в лістингу 3.9.

Лістинг 3.9 – Код перевантаженого методу «forward» для підходу з використанням ортогонального базису та спектральної трансформації

```
def forward(self, img, prompt):
     x = img.T
     y = prompt.T
     p_x = torch.matmul(self.u_x.T, x)
     p_y = torch.matmul(self.u_y.T, y) * 4
     z = torch.cat((p_x, p_y), dim=0)
     p_z_true = torch.matmul(self.u_z.T, z)
     p_z_pred = torch.matmul(
          self.projection_matrix, p_x
     )
     return p_z_true.T, p_z_pred.T
```

На рисунку 3.13 подано криву навчання для даної моделі, яке тривало 179 епох. Видно, що хоча навчання і було швидшим, крива навчання сильно схожа до кривої моделі окремих матриць трансформації, з чого уже можна зробити припущення про провал підходу.

А з рисунка 3.14, на якому зображено матрицю помилок для задачі бінарної класифікації, видно, що модель не змогла успішно знайти підходящі зображення для текстових описів котів. Таким чином, F1-score дорівнює нулю, а це свідчить про неуспішність навчання даного підходу. Підкріплює цей висновок також розподіл метрик «схожості» (середнє значення та медіана становлять 0,46 та 0,45 відповідно), що поданий на рисунку 3.15.

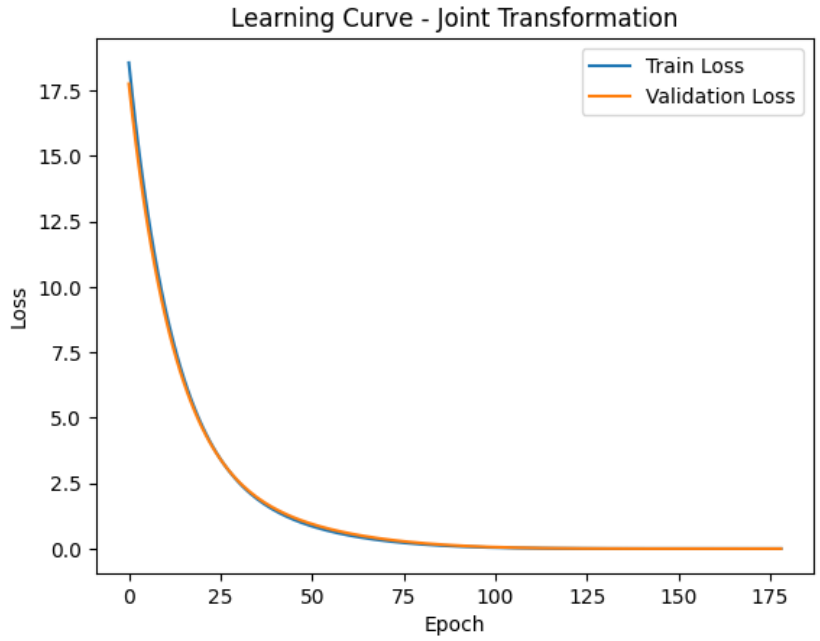

Рисунок 3.13 – Крива навчання для підходу з використанням ортогонального базису та спектральної трансформації

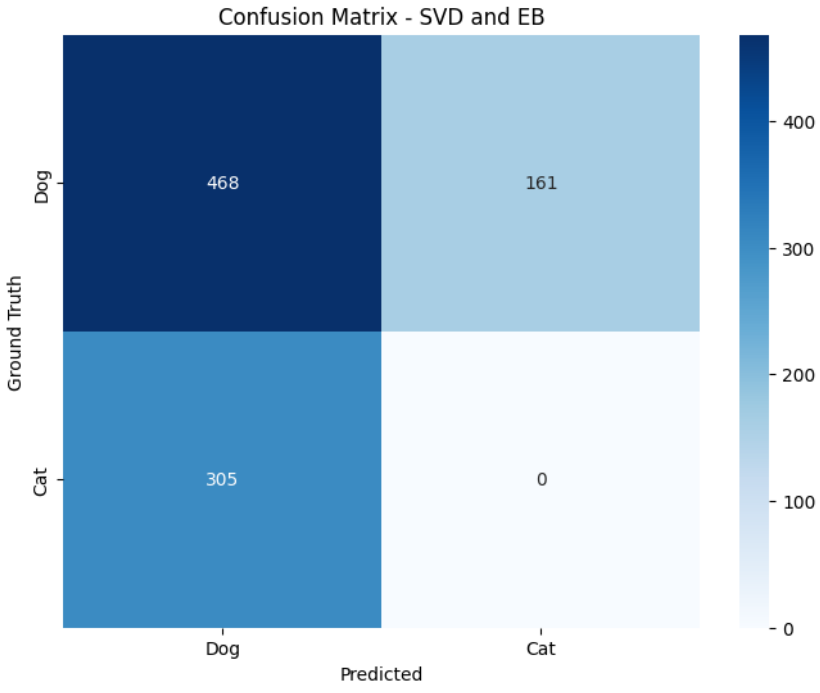

Рисунок 3.14 – Матриця помилок для підходу з використанням ортогонального базису та спектральної трансформації

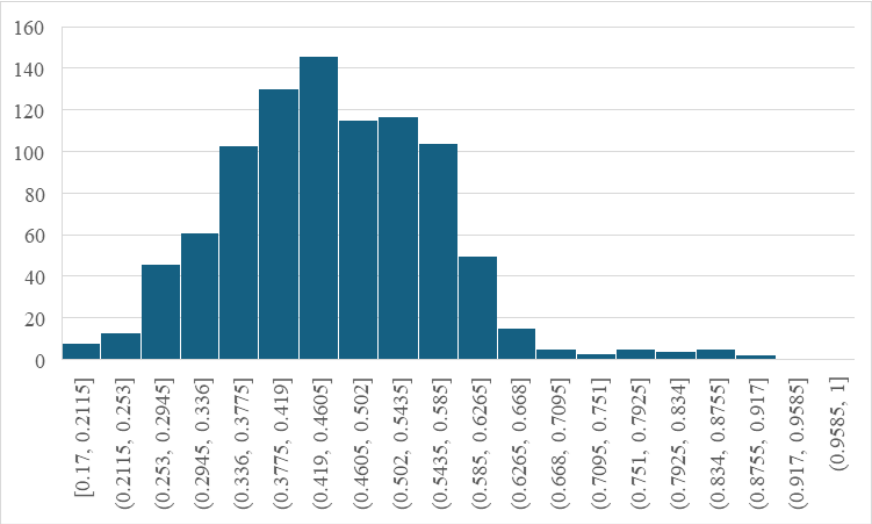

Рисунок 3.15 – Розподіл метрик «схожості» для підходу з використанням ортогонального базису та спектральної трансформації

Було проаналізовано усі три методи приведення векторів представлення до спільного простору, з яких лише метод створення єдиної матриці трансформацій виявився успішним, а методи створення окремих матриць та з використанням ортогонального базису та спектральної трансформації – ні. Слід провести детальний аналіз усіх методів та спробувати пояснити, чому деякі з них були провальними.

4 АНАЛІЗ РЕЗУЛЬТАТІВ ДОСЛІДЖЕНЬ

Вище було описано процес виконання практичних досліджень побудови моделі сегментації екземплярів на основі текстових запитів. Здебільшого було зосереджено увагу на розробці підходів приведення векторів представлення зображень та тексту в спільний простір. Було розглянуто три підходи: створення єдиної матриці трансформації, створення окремих матриць для векторних просторів та з використанням ортогонального базису та спектральної трансформації – з яких лише один виявився успішним.

Далі буде проведено аналіз досліджених практично методів. Для підходів, які виявилися провальними, буде наведено можливі способи вирішення причин, через які підходи виявились неуспішними. Наостанок, буде практично реалізовано остаточну схему запропонованого підходу сегментації екземплярів на основі текстових запитів, представлену на рисунку 2.1.

4.1 Аналіз успішного методу приведення векторів у спільний простір

З усіх трьох підходів приведення векторів у спільний простір, що було досліджено теоретично та реалізовано практично, лише один метод, а саме метод створення єдиної матриці трансформації, виявився успішним. Варто дослідити векторний простір, що породжується даним методом. Але для цього варто спочатку візуалізувати початкові векторні простори з метою наочного представлення їх форм, а також патернів у них. Візуалізація необробленого простору не є можливою, так як їх розмірність (768 та 384 для просторів векторів представлення текстових описів та екземплярів зображень відповідно) сильно перевищує розмірність простору, яке людське око здатне сприймати. Водночас, розбиття усього простору на окремі осі також не має сенсу, оскільки вони самі по собі не несуть жодного

семантичного значення – лише у поєднанні усіх осей вектори мають сенс. Таким чином слід застосувати методи зменшення розмірності просторів, які дають можливість зменшити кількість осей до прийнятного для візуалізації числа (зазвичай нова кількість осей становить два або три для візуалізації векторів на площині або у просторі відповідно) з мінімізацією втрати інформації.

Для візуалізації просторів слід застосувати підхід t-SNE, який є нелінійним способом зменшення розмірності, що враховує як локальний, так і глобальний контекст кожної точки при побудові малорозмірного простору. В якості практичної реалізації було взято клас «TSNE» з пакету «manifold» стандартної бібліотеки машинного навчання мовою програмування Python «sklearn». Візуалізації векторних просторів зображень та тексту представлено на рисунках 4.1 та 4.2 відповідно. На них різні кольори позначають різні породи тварин, а тип визначається стилем точки: «круг» позначає кота, «хрестик» – собаку.

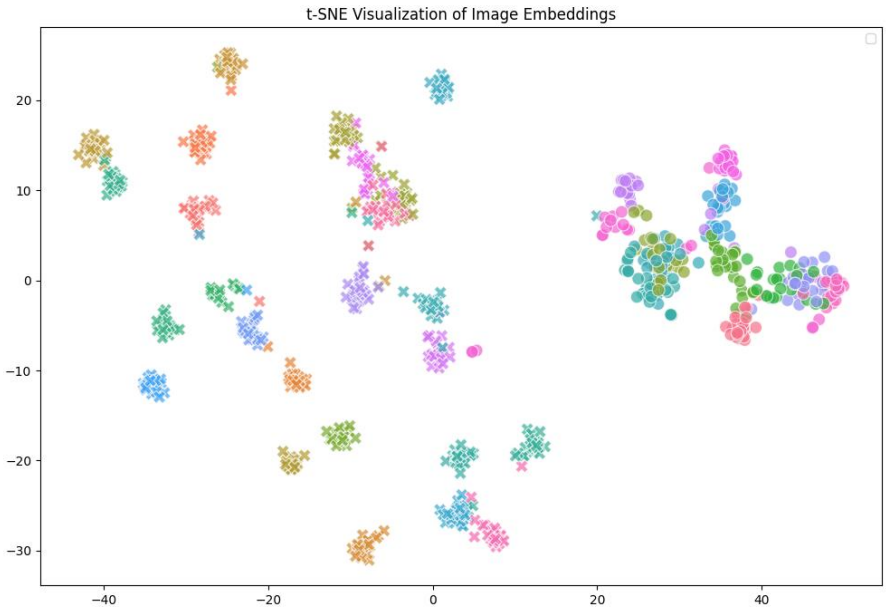

Рисунок 4.1 – Двовимірне представлення початкового векторного простору екземплярів зображень

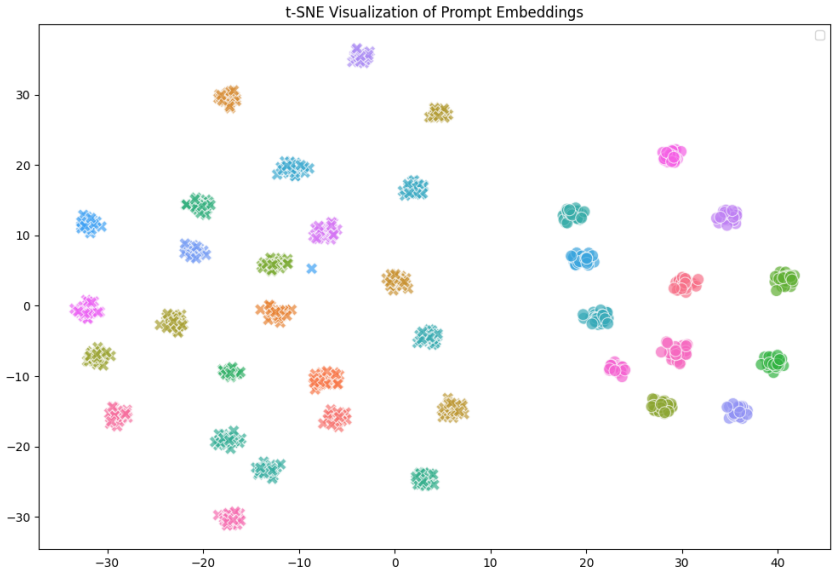

Рисунок 4.2 – Двовимірне представлення початкового векторного простору текстових описів

Можемо помітити, що на обох просторах вектори представлень, що належать тваринам однієї породи, утворюють кластери, при чому у просторі текстових описів вони більш відокремлені один від одного. Таким чином, можемо сформулювати умову, що спільний векторний простір має зберегти утворені кластери для того, щоб підхід приведення, в результаті якого був створений даний спільний простір, був успішним.

Для перевірки цієї умови слід розглянути візуалізацію спільного простору, яка представлена на рисунку 4.3 («кругом» позначено вектор представлення зображення, «хрестиком» – вектор представлення тексту). На ньому видно, що хоча кластери і збережено, дані кластери інформації в різних форматах знаходяться віддалено один від одного.

Це можна пояснити, розглянувши графік розподілу довжин векторів представлення в спільному просторі (рисунок 4.4), на якому чітко видно, що довжини векторів представлення тексту набагато менші, ніж довжини векторів представлення зображень.

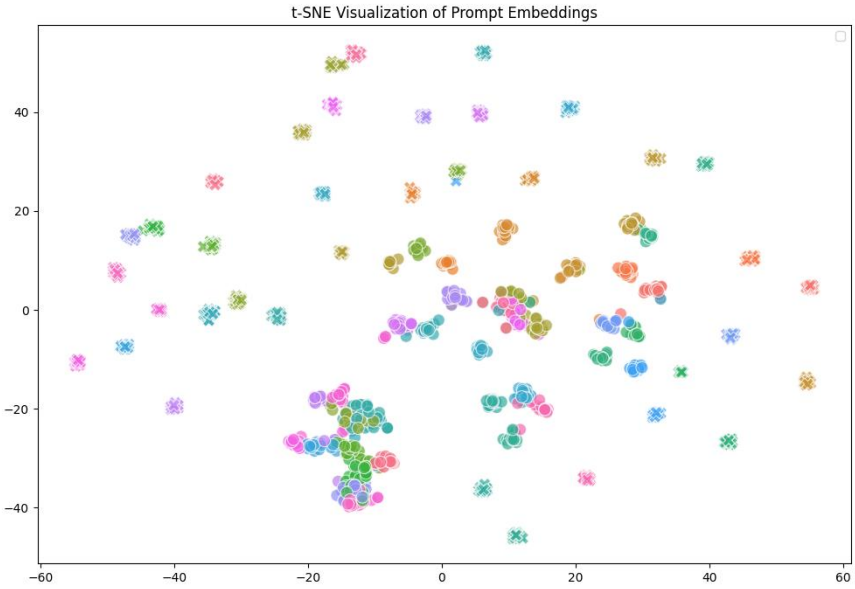

Рисунок 4.3 – Двовимірне представлення спільного векторного простору, утвореного за допомогою єдиної матриці трансформації

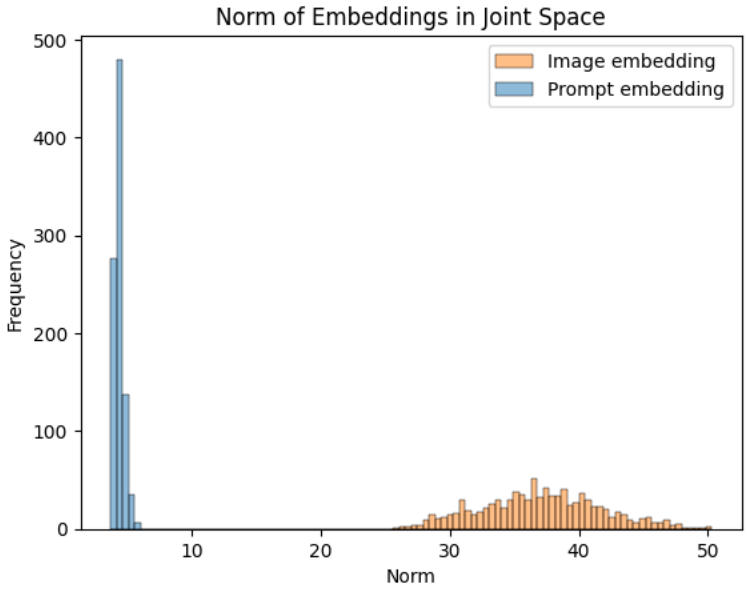

Рисунок 4.4 – Розподіл довжин векторів спільного простору, утвореного за допомогою єдиної матриці трансформації

Такий результат можна пояснити метрикою «схожості», яку було використано як функцію втрат при навчанні матриці трансформації. Косинусова «схожість», яка представлена на формулі 2.6, для обрахунку

спирається лише на кут між векторами: чим менше кут, тим схожі за семантичним значенням вектори один до одного. Таким чином, при навчанні модель не мала потреби у приведенні векторів до однієї довжини. Такий результат показує, що якщо модель була навчена на основі метрики косинусової «схожості», дана метрика також повинна бути використана для знаходження пар відповідностей при використанні моделі в подальшому.

4.1.1 Аналіз доцільності застосування текстових аугментацій

Для навчання усіх підходів приведення векторів до спільного простору, в тому числі і для методу створення єдиної матриці трансформації, було використано аугментовані текстові описи, а самі принципи аугментації було описано у підрозділі 2.4.1 та розділі 3.1.

Вони спиралися на [19], в якому експериментально було показано, що для задачі класифікації текстів дана аугментація збільшувала точність приблизно від двох до п’яти відсотків (в залежності від типу моделі та розміру навчальної вибірки).

Проте розміри речень у описах значно перевищували кількість слів, що були змінені, а для енкодингу даних описів були використані моделі, які навчалися на великих текстових корпусах. Тому варто перевірити, чи принесла аугментація текстових даних користь у даній задачі.

Для цього було навчено аналогічну матрицю єдиної трансформації на неаугментованих текстових описах. В результаті, після 237 епох навчання було завершене, а на рисунку 4.5 представлено розподіл метрик «схожостей» між еталонними описами та описами зображень (середнє значення та медіана становлять 0,81 та 0,82 відповідно).

Порівнюючи дані результати з початковими результатами (навчання матриці єдиної трансформації на аугментованих текстових даних тривало 236 епох, середнє значення та медіана метрик «схожостей» становило 0,8 та 0,81 відповідно), можемо зробити висновок, що

аугментація не принесла видимої користі при практичній реалізації даного методу.

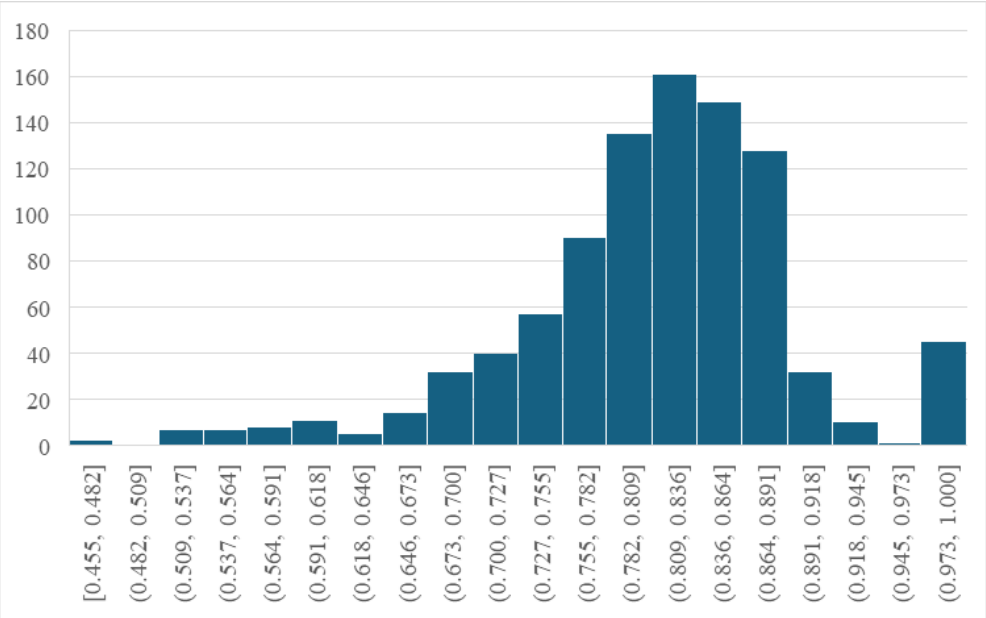

Рисунок 4.5 – Розподіл метрик «схожості» для моделі матриці єдиної трансформації, навченої на неаугментованих текстових описах

4.1.2 Аналіз обраного напрямку приведення векторів до спільного простору

Даний метод приведення векторів представлення до спільного простору полягає у приведенні одного векторного простору до іншого за допомогою єдиної матриці трансформації. В даній реалізації саме векторний простір представлень текстових описів приводиться до векторного простору представлень екземплярів зображень. Таким чином, взаємне розміщення векторів представлення текстових даних «імітуватиме» взаємне розміщення векторів представлення візуальної інформації. Але з рисунків 4.1 та 4.2 видно, що у просторі текстових описів існують більш відокремлені один від одного кластери векторів, які зазвичай відносяться до зображень тварин з

різними породами. В якості пояснення такої чіткішої відокремленості кластерів можна зробити два припущення:

– через те, що текстові описи були згенеровані з використанням LLM- та VLM-моделей, дані кластери занадто штучно розмежовані один від одного, що не притаманне зображенням, так як вони не є штучними;

– для створення векторів представлення текстових описів було використано модель SBERT, задача якої і полягає у порівнянні різних текстів між собою, в той час як для створення векторів зображень екземплярів було використано звичайний візуальний трансформер, який не був явно навчений з метою порівняння, саме тому він не зміг так чітко виокремити кластери.

Тому варто проаналізувати обраний напрямок приведення шляхом порівняння із зворотнім. Для цього було створено клас-інтерфейс для отримання результату від моделей приведення векторів до одного простору в зворотному напрямку, в якому змінюються саме вектори представлення зображень, а не текстових описів:

```
def encode_image(self, embedding):
    return np.dot(embedding, self.matrix)
```

На рисунку 4.6 видно, що у зворотному напрямку приведення метрики F1-score для порід значно покращилися, це свідчить про те, що модель більш часто обирала відповідні зображення тварин з однаковою породою. Також слід розглянути новий спільний векторний простір, який представлено на рисунку 4.7 («кругом» позначено вектор представлення зображення, «хрестиком» – вектор представлення тексту). На ньому видно, що хоча деякі кластери векторів представлення екземплярів зображень і є більш віддалені від інших векторів, все одно кластери векторів представлення текстових описів є більш відмежовані один від одного, тобто покращення метрик F1-score для порід не можна пояснити кращим розмежуванням кластерів векторів у новому просторі. Скоріше за все, дане покращення метрик доцільно пояснити тим, що порівнюються вектори у просторі текстових

описів, які були чіткіше розмежовані один від одного від початку, що і дало покращення у відокремленні тварин різних порід одна від одної.

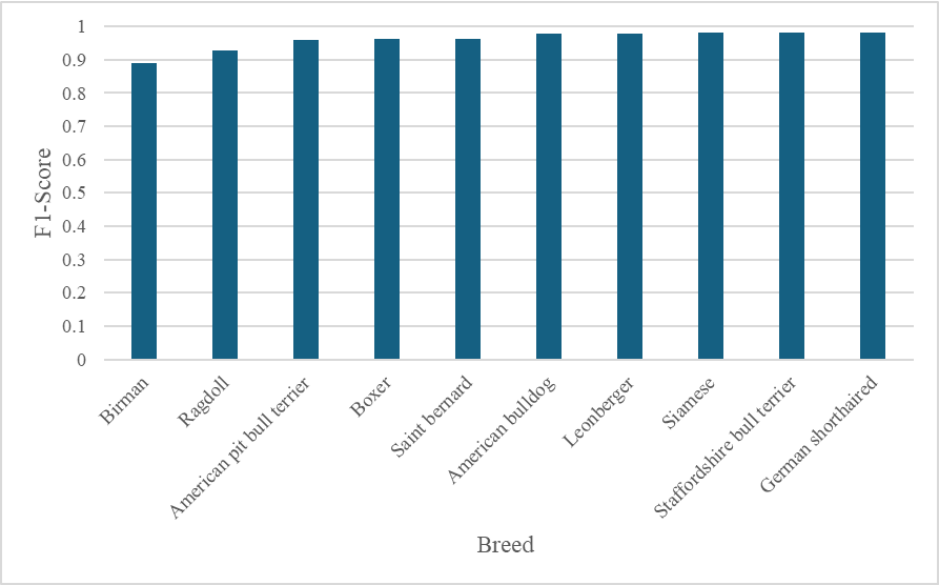

Рисунок 4.6 – Оновлені метрики F1-score для кожної породи, в яких дана метрика для єдиної матриці трансформації не дорівнювала одиниці

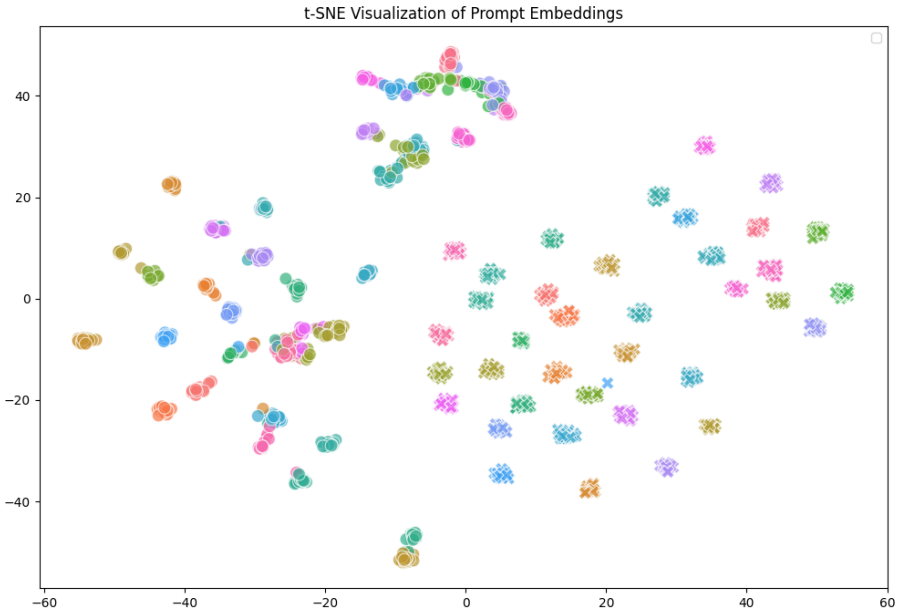

Рисунок 4.7 – Двовимірне представлення спільного векторного простору, утвореного за допомогою зворотного напрямку приведення

4.1.3 Аналіз методу приведення на зображеннях з групами тварин

Датасет, на якому було навчено моделі підходу приведення векторів представлення до спільного простору, здебільшого складається із зображень лише однієї тварини на рисунку, але дані методи було навчено саме для сегментації екземплярів, яка використовується для обробки зображень, на яких присутні одразу декілька об’єктів, тому слід проаналізувати, як створений метод приведення впорається на рисунках з багатьма тваринами.

Для цього було відібрано 29 зображень, на яких розташовано одразу декілька тварин різного роду. Опис зображень, а також результати виконання методу приведення векторів до спільного простору за допомогою єдиної матриці трансформації, приведено в таблиці 4.1.

Таблиця 4.1 – Результати приведення векторів до спільного простору на зображеннях з групами тварин

Опис зображення	Кількість виявлених екземплярів	Середня метрика «схожості»
1	2	3
Два кота, що стоять поруч	2	1,00
Людина тримає в руках собаку та гладить кота	2	1,00
Собака та кіт, що лежать поруч	2	1,00
Група собак та котів, що сидять поруч	5	0,82
Собака «обнюхує» лежачого кота	1	1,00
Собака та кіт дивляться одне на одного з ворожнечею	2	1,00
Велика кількість котів та собак стоять на груповому фото	15	0,70

Продовження таблиці 4.1

1	2	3
Три кота та собака лежать поруч	3	0,91
Собака та кіт «обнюхують» одне одного	2	1,00
Група котів у русі	4	0,88
Колаж тварин із їх власниками	3	0,76
П’ять котів стоять на кам’яній плиті	5	0,92
Кіт сидить на лежачій собаці	2	1,00
Людина тримає кота і собаку	2	1,00
Людина в оточенні двох собак та кота	3	1,00
Кіт та собака лежать разом	2	1,00
Кошеня та цуценя лежать разом	2	1,00
Група безхатніх собак	4	0,66
Троє собак несуть довгу гілку	3	0,86
Ряд собак та котів на білому фоні	8	0,65
Кошеня «нападає» на собаку	2	0,62
Група котів та собак у русі	7	0,71
Ряд собак різних порід	15	0,85
Кіт та собака, що стоять поруч	2	1,00
Група кошенят, що стоять в ряд	3	0,75
Група котів, що лежать разом	4	0,66

З таблиці видно, що модель в середньому вдало впоралася із задачею на зображеннях з багатьма тваринами, але вона може плутати екземпляри між собою, якщо їх дуже багато на одному зображенні, що можна пояснити тим, що чим більше екземплярів на одному зображенні, тим менш деталізованими кожен з них стає, а це в свою чергу призводить до створенню менш семантично багатих векторів представлення.

4.2 Аналіз неуспішних методів приведення векторів представлення

Проаналізувавши успішний метод приведення векторів у спільний простір, варто перейти до аналізу методів, які були провальними, а саме методи створення окремих матриць трансформації та за допомогою ортогонального базису та спектральної трансформації. Їх провал не можливо пояснити тим, що складність методів не відповідає складності задачі, так як попередній успішний метод є менш складним, ніж решта провальних. Тому слід проаналізувати їх результати, розглянувши створені векторні простори, які представлені для методів створення окремих матриць трансформації та з використанням ортогонального базису та спектральної трансформації на рисунках 4.8 та 4.9 відповідно («кругом» позначено вектор представлення зображення, «хрестиком» – вектор представлення тексту).

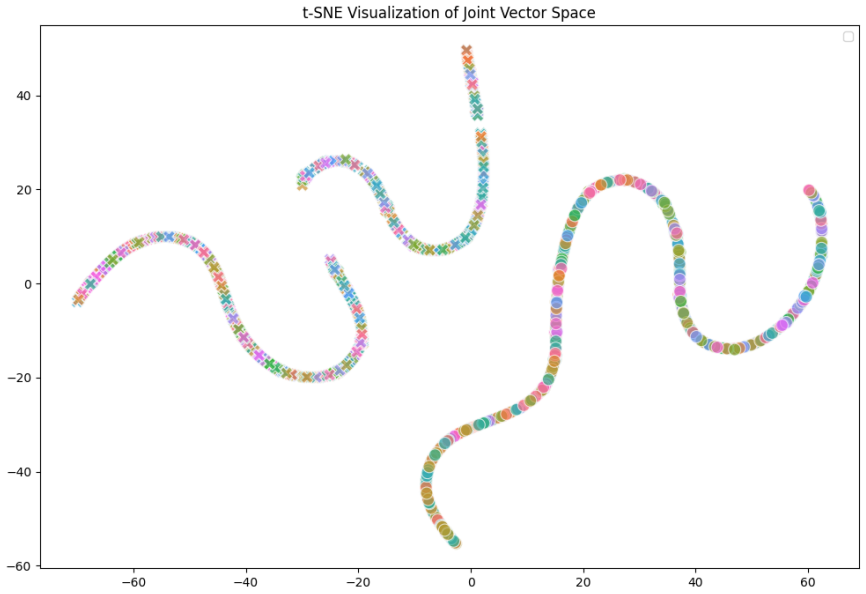

Рисунок 4.8 – Двовимірне представлення спільного векторного простору, утвореного за допомогою окремих матриць трансформації

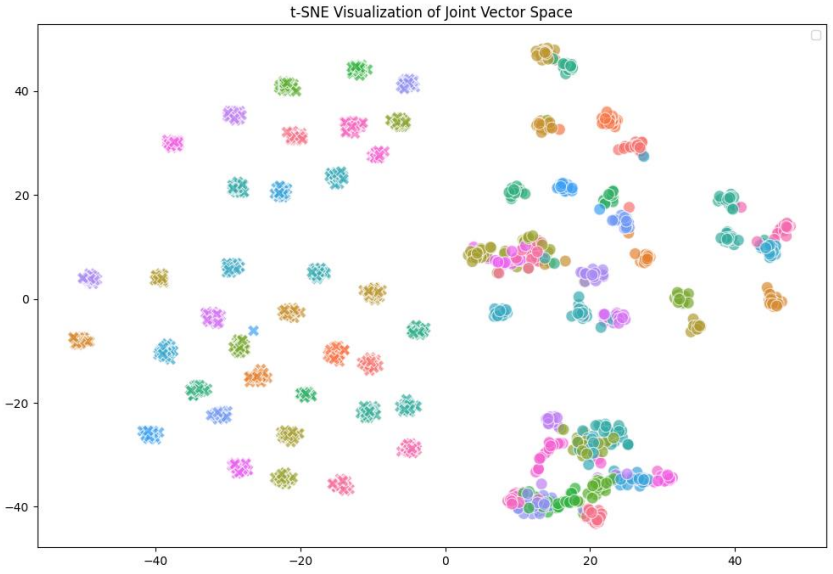

Рисунок 4.9 – Двовимірне представлення спільного векторного простору, утвореного з використанням ортогонального базису та спектральної трансформації

На візуальних представленнях спільних векторних просторів можна спостерігати різні структури. Таким чином, доцільно зробити висновок, що методи мають різні причини провалу, аналіз яких треба провести окремо.

З рисунка 4.8 видно, що у новому векторному просторі не збереглися кластери векторів представлення інформації, що належать екземплярам тварин однакових порід, так як це було для методу створення єдиної матриці трансформації. Таким чином, можна зробити висновок про те, що у методі створення окремих матриць трансформації новий простір повністю втратив семантичну інформацію, необхідну для створення пар відповідності між векторами, що належать одному і тому ж екземпляру.

Для створення стратегії розв’язання даної проблеми варто розглянути задачу метричного навчання, яка за мету ставить задачу знаходження векторного простору, де існує метрика «схожості» (зазвичай, евклідова відстань або косинусова «схожість»), яка показує, наскільки схожі дві точки, що представлені у вигляді вектора [25]. Не важко помітити, що задача

приведення до спільного векторного простору належить до задач метричного навчання.

В залежності від типу даних, на яких навчаються моделі метричного навчання, їх можна поділити на три типи:

– методи метричного навчання на основі попарних даних, тобто коли датасет представлено у вигляді пар векторів;

– методи на основі «проксі», тобто такі, які використовують псевдокласовий представник (який і називають як «проксі») для формулювання остаточної оптимізації;

– методи на основі регуляризації, тобто такі, які використовують допоміжну інформацію в процесі навчання.

З розглянутих методів, метод окремих матриць трансформації найближче належить до методу метричного навчання на основі попарних даних. Існує декілька запропонованих функцій втрат для таких моделей, серед яких є функція контрастної втрати, яка полягає у розрахунку відстаней між парами векторів:

$$\text{Loss} = \begin{cases} \left\|X_i - X_j\right\|^2, & \text{якщо } y_i = y_j \\ \left|\alpha - \left\|X_i - X_j\right\|^2\right|, & \text{якщо } y_i \neq y_j \end{cases}, \tag{4.1}$$

де X_i та X_j є векторами представлення з класами y_i та y_j відповідно;

α – границя, яка показує, наскільки віддаленими мають бути вектори, що представляють семантично різну інформацію.

З даної формули бачимо, що під час навчання повинна бути присутня інформація про семантично віддалені точки, які не були частиною датасету під час навчання окремих матриць трансформації.

Таким чином, для спроби вирішення проблеми, що призвела до провалу методу створення окремих матриць трансформації, слід додати до навчального датасету негативні пари векторів, щоб модель мала розуміння

про те, які вектори слід вважати схожими, а які – діаметрально протилежними. Це можливо зробити, використовуючи породи як класи: в якості негативної пари вважати вектори, що належать тваринам різних порід (даний процес має назву негативного майнінгу).

Тепер слід перейти до аналізу методу використання ортогонального базису та спектральної трансформації. На рисунку 4.9 бачимо, що даний підхід зберіг загальну форму обох векторних просторів при приведенні, але все одно метод не зміг порівнятися з методом створення єдиної матриці трансформації. Можемо зробити припущення, що під час створення передбачень було використано непідходящу метрику «схожості»: під час використання методу на тестовій вибірці було використано евклідову відстань, що є неоптимальним, так як з рисунку 4.9 видно, що векторні представлення даних, які знаходяться в різних форматах, віддалені. Але при практичній перевірці дане припущення виявилось хибним. Таким чином, можемо зробити інше припущення, що під час навчання модель не мала інформації, необхідної для встановлення відповідності.

Для вирішення даної проблеми слід розглянути алгоритм навчання методу, представленого в лістингу 3.9. На ньому видно, що процес навчання знаходить оптимальні значення для матриці A_c за формулою 2.14, в якій знаходиться відповідність векторів представлення у об’єднаному векторному просторі. Але під час обчислення передбачень використовуються вектори представлення в окремих просторах. Тому для вирішення проблеми, імовірно, необхідно під час обчислення значення функції втрат використати саме вектори представлення, що явно обчислюються при створенні передбачень, згідно з процесом, описаним у формулах 2.15–2.16.

Але при цьому постає питання: чи не виникне проблема, пов’язана зі спробою створити окремі вектори представлення, а саме втрата новим векторним простором необхідної для порівняння семантичної інформації. Це важко передбачити аналітично, тому необхідно провести практичні

дослідження на дану тему та, за необхідності, заповнити навчальну вибірку негативними парами векторів і змінити функцію втрат так, як це було описано вище.

4.3 Програмна реалізація повного запропонованого підходу

Наостанок, слід описати програмну реалізацію повного запропонованого у даній роботі підходу до вирішення задачі сегментації екземплярів на основі текстових запитів, схема якого представлена на рисунку 2.1. Для цього слід розбити процес на чотири функціональні частини:

– створення сегментів: пропуск зображення через модель сегментації екземплярів для створення усіх сегментів, які будуть потенційними вихідними даними алгоритму;

– створення векторів представлення сегментів екземплярів: пропуск усіх потенційних сегментів через візуальний трансформер для створення векторів представлення у початковому просторі та через модель приведення векторів до спільного простору;

– створення вектору представлення текстового опису: пропуск запиту-тексту через реченнєвий трансформер та через модель приведення векторів до спільного простору;

– знаходження підходящого сегменту: порівняння усіх векторів представлення сегментів із вектором представлення текстового опису та вибір того екземпляру, який має найбільшу метрику «схожості».

Повний програмний код, що реалізує запропонований підхід, наведено у додатку В. У ньому функції «get_segments», «get_image_embedding» та «get_prompt_embedding» є проміжними реалізаціями перших трьох функціональних частин, а функція «get_prompted_segment» є основною: вона отримує на вході зображення та запит та видає маску сегменту відповідного екземпляру.

ВИСНОВКИ

В результаті цієї роботи було розроблено підхід до вирішення задачі сегментації екземплярів на основі текстового запиту. Детально було розглянуто процес приведення векторів представлення інформації, поданої у вигляді зображень екземплярів та текстових описів у спільний векторний простір.

В результаті теоретичного дослідження було обрано моделі сегментації екземплярів, а також унімодальні моделі створення векторів представлення. Для приведення векторів представлення у спільних векторний простір було розглянуто три підходи: створення єдиної матриці трансформації, створення окремих матриць трансформації для початкових векторних просторів та використання ортогонального базису та спектральної трансформації для створення відображення векторів з одного простору в інший.

В результаті практичних досліджень та їх аналізу було показано, що існує підхід створення спільного векторного простору, що використовує лише лінійні перетворення. Підхід створення єдиної матриці трансформації виявився найбільш успішним. При цьому аналіз візуального представлення векторного простору та результатів успіху застосування підходу показали, що аугментація текстових даних не дає жодних змістовних переваг, а вибір напрямку, в якому будуть трансформовані вектори, може вплинути на результат остаточної моделі.

Підхід створення окремих матриць трансформації для початкових векторних просторів був неуспішним, імовірно, через відсутність негативних пар векторів представлення у навчальній вибірці. Використання ортогонального базису та спектральної трансформації хоч і створило векторний простір, що зберіг форму початкових, але все одно виявився провальним, імовірно, через некоректний підхід до навчання коефіцієнтів для даного підходу.

В рамках даного дослідження було повністю розглянуто запропонований підхід до вирішення задачі сегментації екземплярів на основі текстового запиту. Було послідовно описано всі дії, які необхідно виконати для створення методів приведення векторів представлення до одного простору на принципах застосування машинного навчання. Для практичної реалізації дій було створено програмний код з метою отримання результатів з уже навчених моделей сегментації та отримання початкових векторів представлення, а також з метою створення моделей машинного навчання та для подальшого обчислення метрик. Для аналізу методів приведення до спільного простору було створено їх візуалізації за допомогою методик зменшення розмірності. Для всіх неуспішних методів було зроблено припущення про причини їх провалу, а успішних підхід приведення було використано в розробці програмної реалізації повного запропонованого підходу.

Загалом, результати дослідження можна вважати задовільними, так як було показано, що запропонований підхід до вирішення задачі сегментації екземплярів на основі текстового запиту з використанням лише лінійних трансформацій векторних просторів є практично можливим. При цьому існує також можливість подальшого дослідження даної теми, яке полягатиме у теоретичному та практичному аналізі припущень, зроблених для пояснення неуспішності деяких з розглянутих методів приведення векторів до спільного простору.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Szeliski R. Computer vision: algorithms and applications. 2nd ed. Springer, 2022.

2. Microsoft COCO: common objects in context / T.-Y. Lin et al. *Computer vision – ECCV 2014*. 2014. P. 740–755. URL: https://doi.org/10.1007/978-3-319-10602-1_48 (date of access: 25.05.2025).

3. Ronneberger O., Fischer P., Brox T. U-Net: convolutional networks for biomedical image segmentation. *Medical image computing and computer-assisted intervention – MICCAI 2015*. 2015. P. 234–241. URL: https://doi.org/10.1007/978-3-319-24574-4_28 (date of access: 29.04.2025).

4. Image segmentation using deep learning: a survey / S. Minaee et al. *IEEE transactions on pattern analysis and machine intelligence*. 2022. P. 3523–3542. URL: https://doi.org/10.1109/TPAMI.2021.3059968 (date of access: 29.04.2025).

5. Terziyan V., Vitko O. Causality-Aware convolutional neural networks for advanced image classification and generation. *Procedia computer science*. 2023. P. 495–506. URL: https://doi.org/10.1016/j.procs.2022.12.245 (date of access: 08.05.2025).

6. Krizhevsky A., Sutskever I., Hinton G. E. ImageNet classification with deep convolutional neural networks. *Advances in neural information processing systems 25 (NIPS 2012)*. 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (date of access: 29.04.2025).

7. Streltsov O., Terziyan V., Vitko O. Discover and explore weak causality and causal disposition in images for smart manufacturing tasks. *Procedia computer science*. 2025. P. 187–198. URL: https://doi.org/10.1016/j.procs.2025.01.082 (date of access: 08.05.2025).

8. Attention is all you need / A. Vaswani et al. *Advances in neural information processing systems 30 (NIPS 2017)*. 2017. URL: https://doi.org/10.48550/arXiv.1706.03762 (date of access: 24.05.2025).

9. An image is worth 16x16 words: transformers for image recognition at scale / A. Dosovitskiy et al. 2021. URL: https://doi.org/10.48550/arXiv.2010.11929 (date of access: 24.05.2025).

10. Mehrabian A. Silent messages. Wadsworth, 1971. 152 p.

11. VisualBERT: a simple and performant baseline for vision and language / L. H. Li et al. 2019. URL: https://doi.org/10.48550/arXiv.1908.03557 (date of access: 05.05.2025).

12. BERT: pre-training of deep bidirectional transformers for language understanding / J. Devlin et al. *Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019. P. 4171–4186. URL: https://doi.org/10.18653/v1/N19-1423 (date of access: 05.05.2025).

13. Reimers N., Gurevych I. Sentence-BERT: sentence embeddings using siamese bert-networks. 2019. URL: https://doi.org/10.48550/arXiv.1908.10084 (date of access: 24.05.2025).

14. Learning transferable visual models from natural language supervision / A. Radford et al. *Proceedings of the 38th international conference on machine learning*. 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (date of access: 07.05.2025).

15. Segment anything / A. Kirillov et al. *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*. 2023. P. 4015–4026. URL: https://doi.org/10.48550/arXiv.2304.02643 (date of access: 07.05.2025).

16. EVF-SAM: early vision-language fusion for text-prompted segment anything model / Y. Zhang et al. 2024. URL: https://doi.org/10.48550/arXiv.2406.20076 (date of access: 07.05.2025).

17. Cheng B., Schwing A. G., Kirillov A. Per-Pixel classification is not all you need for semantic segmentation. *Advances in neural information processing systems 34 (neurips 2021)*. 2021. URL: https://doi.org/10.48550/arXiv.2107.06278 (date of access: 24.05.2025).

18. Cats and dogs / O. M. Parkhi et al. *IEEE conference on computer vision and pattern recognition*. 2012. URL: https://www.robots.ox.ac.uk/~vgg/publications/2012/parkhi12a/parkhi12a.pdf (date of access: 25.05.2025).

19. Wei J., Zou K. EDA: easy data augmentation techniques for boosting performance on text classification tasks. 2019. URL: https://doi.org/10.48550/arXiv.1901.11196 (date of access: 04.06.2025).

20. Kingma D. P., Lei Ba J. L. Adam: a method for stochastic optimization. *Published as a conference paper at ICLR 2015*. 2015. URL: https://doi.org/10.48550/arXiv.1412.6980 (date of access: 25.05.2025).

21. Тєвяшев А. Д., Литвин О. Г. Вища математика у прикладах та задачах. Ч.1. Лінійна алгебра і аналітична геометрія. Диференціальне числення функцій однієї змінної. 2-ге вид. Харків: ХНУРЕ : Фактор, 2004. 592 с.

22. Lu L. Y. Fast machine learning method with vector embedding on orthonormal basis and spectral transform. 2013. URL: https://doi.org/10.48550/arXiv.2310.18424 (date of access: 25.05.2025).

23. Zhang Z. The singular value decomposition, applications and beyond. 2015. URL: https://doi.org/10.48550/arXiv.1510.08532 (date of access: 25.05.2025).

24. Prompt engineering for generative AI: future-proof inputs for reliable AI outputs. O'Reilly Media, Incorporated, 2024.

25. Deep metric learning for computer vision: a brief overview / D. D. Mohan et al. 2023. URL: https://doi.org/10.48550/arXiv.2312.10046 (date of access: 06.04.2025).