Міністерство освіти й науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

Автоматична генерація описів до будівель на основі зображень/ Automatic generation of building descriptions based on images

Текстова частина до кваліфікаційної роботи
за спеціальністю 122 – «Комп'ютерні науки»

Виконала: студентка 4-го року навчання,
Спеціальності 122 «Комп'ютерні науки»
Кузнецова Анна Володимирівна
Керівник Іванюк А. О.,
Доктор філософії

Рецензент _____
Кваліфікаційна робота захищена
з оцінкою _____
Секретар ЕК _____
«____» _____ 20____ р

Київ -2026

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на кваліфікаційну роботу

студентці 4 р.н. бакалаврської програми Комп’ютерні науки

Кузнецовій Анні Володимирівні

Тема: Автоматична генерація описів до будівель на основі зображень/

Automatic generation of building descriptions based on images

Зміст текстової частини до кваліфікаційної роботи:

Анотація
Вступ
Теоретичні основи
Дослідження експерименту
Результати
Висновки
Джерела

Дата видачі «____»_____ 2025 р.
Керівник _____ (підпис)
Завдання отримав _____ (підпис)

Календарний план виконання роботи

№	Назва етапу кваліфікаційного проєкту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на кваліфікаційну роботу	14.12.2025	
2.	Огляд літератури за темою роботи	грудень 2025 р. - січень 2026 р.	
3.	Збір даних	січень-лютий 2026 р.	
4.	Проведення дослідження	поч. березня 2026 р.	
5.	Аналіз проведених результатів	березень 2026 р.	
6.	Написання текстової частини курсової роботи	кін. березня - поч. квітня 2026 рр.	
7.	Оформлення презентації для захисту	травень 2026 р.	
8.	Захист кваліфікаційної роботи	2 червня 2026 р.	

Студентка Кузнецова А. В.
Керівник Іванюк А. О.
“____” _____

ЗМІСТ

1 АНОТАЦІЯ

Збереження архітектурної спадщини Києва є актуальною проблемою - історичні будівлі нерідко опиняються під загрозою знесення через відсутність належної документації. Ця робота присвячена розробці системи автоматичної генерації текстових описів будівель на основі зображень із застосуванням методів глибинного навчання та комп’ютерного зору.

У рамках дослідження було самостійно сформовано набір даних для оцінки оглянутих мультимодальних моделей без попереднього навчання та тонкого налаштування, розроблено атрибутивний спосіб оцінки ефективності текстової генерації моделей та до-навчено Qwen3-VL-2B методом низькорангової адаптації (LoRA).

Ключові слова: Архітектурні будівлі, атрибути, мультимодальні моделі, генерація тексту, тонке налаштування, LoRA.

2 ВСТУП

Щорічно в місті Києві унікальні пам’ятки історичної архітектури опиняються під загрозою знесення. Лише за 2024-2025 роки десятки пам’яток культурної спадщини були на межі повного знищення[1]. Велика проблема полягає в тому, що фактично єдиним джерелом про занедбані будинки Києва є інтерактивна мапа історичної спадщини Києва, яку створила громадська організація (ГО) “Мапа Реновації”[2].

За словами членів ГО, попри те, що об’єкти можуть очевидно підпадати під критерії пам’яток, вони водночас можуть тривалий час знаходитись без охоронного статусу, адже щоб його здобути, необхідно дуже багато документації, і треба щоб вона пройшла через Департамент охорони культурної спадщини (ДОКС) [2].

Метою цієї роботи є дослідження, оцінка ефективності мультимодальних підходів на основі різних архітектур нейронних мереж для розробки та вдосконалення автоматичного формування текстових описів зображень архітектурних пам'яток Києва. Дослідження ґрунтується на реальних даних, зібраних двома способами: парсингом інтерактивної мапи, розробленої ГО «Мапа Реновацій» [3], та геокодуванням переліку об'єктів культурної спадщини з кадастрового реєстру Департаменту охорони культурної спадщини.

Зокрема, в роботі досліджуються такі підходи: Contrastive Language-Image Pre-training (CLIP) - модель для оцінювання семантичної схожості між зображеннями та текстовими описами без додаткового донавчання; Bootstrapped Language-Image Pretraining (BLIP) - двоетапна мультимодальна архітектура, що досліджується в режимі zero-shot; Qwen3-VL-2B - компактна мультимодальна мовна модель, що також оцінюється в zero-shot режимі;

InternVL3 та Large Language-and-Vision Assistant (LLaVA) - сучасні відкриті мультимодальні архітектури, включені до порівняння як представники різних підходів до поєднання візуального та мовного компонентів.

На основі отриманих результатів було обрано модель із найвищими показниками якості. Надалі для неї було виконано тонке налаштування (fine-tuning) на зібраному датасеті, щоб забезпечити дотримання заданої структури опису зображень.

Таким чином, дане дослідження охоплює повний цикл: від збору та підготовки актуальних даних про об'єкти культурної спадщини міста Києва - до систематичного порівняння сучасних мультимодальних моделей та цілеспрямованої адаптації найефективнішої з них до конкретного прикладного завдання.

Робота виконувалась згідно послідовності кроків:

1. Збір та підготовка датасету об'єктів архітектурної спадщини міста Києва, включаючи зображення та текстові описи будівель.
2. Огляд теоретичних засад мультимодального навчання, архітектур нейронних мереж та методів генерації текстових описів зображень.
3. Реалізація та оцінка базових підходів: CLIP, BLIP та Qwen3-VL-2B у режимі zero-shot.
4. Реалізація та оцінка додаткових мультимодальних архітектур: InternVL3 та LLaVA.
5. Аналіз результатів усіх досліджуваних підходів.
6. Тонке налаштування моделі з найкращими показниками на зібраному датасеті та оцінка ефективності адаптації.

Об'єктом дослідження є процес автоматичної генерації текстових описів зображень із застосуванням методів глибинного навчання та комп'ютерного зору.

Предметом дослідження є мультимодальні моделі та архітектури нейронних мереж для генерації описів зображень архітектурних пам'яток, зокрема CLIP, BLIP, Qwen3-VL-2B, InternVL3 та LLaVA, їхня ефективність та здатність до адаптації на вузькоспеціалізованих даних.

Практична цінність отриманих результатів полягає у можливості їхнього застосування для:

- Документування архітектурної спадщини: автоматична генерація текстових описів дозволяє систематизувати та масштабувати процес каталогізації пам'яток, знижуючи залежність від ручної праці експертів.
- Підтримки реєстрів культурної спадщини: розроблений підхід може бути інтегрований у системи обліку об'єктів культурної спадщини для автоматичного поповнення та актуалізації описів.
- Збереження інформації в умовах воєнного часу. В контексті масштабних руйнувань архітектурних пам'яток України, автоматизовані інструменти опису та документування набувають особливої актуальності для фіксації об'єктів, що перебувають під загрозою знищення.

У ході виконання роботи для технічного дослідження використовувались: Google Colab із підтримкою T4 GPU[4] як основне середовище розробки та проведення експериментів; Python 3.11 як основна мова програмування; PyTorch[5] та Hugging Face Transformers[6] як головні фреймворки для завантаження та запуску мультимодальних моделей; CLIP, BLIP, Qwen3-VL-2B, InternVL3 та LLaVA як досліджувані моделі; бібліотеки pandas

та Pillow для обробки та підготовки датасету; а також інтерактивна мапа ГО «Мапа Реновацій» [3] як одне з джерел зібраних даних.

Дана робота поділена на розділи, кожен з яких охоплює важливі аспекти дослідження та порівняння мультимодальних підходів до генерації текстових описів зображень.

Розділ 3 присвячений теоретичним засадам: огляду архітектур мультимодальних нейронних мереж, принципів контрастивного навчання, методів генерації текстових описів зображень та підходів до оцінки їхньої якості.

Розділ 4 описує практичну реалізацію: збір та підготовку датасету, налаштування середовища та проведення експериментів із досліджуваними моделями в режимі zero-shot (без додаткового навчання).

Розділ 5 представляє результати експериментів, їхній аналіз, а також процес тонкого налаштування найефективнішої моделі та оцінку результатів адаптації.

Розділ 6 містить висновки дослідження з аналізом отриманих результатів та можливих напрямків подальшої роботи.

3 ТЕОРЕТИЧНІ ОСНОВИ

3.1 Image Captioning як задача

Опис зображень (Image Captioning) - це мультидисциплінарна галузь, яка поєднує технології комп'ютерного зору та обробки природної мови (NLP) для автоматичного створення описів зображень природною мовою[7]. Це завдання передбачає створення текстових описів, що відображають зміст відповідного зображення, і застосовується в різних сферах, таких як: розмітка семантичних тегів, пошук зображень, навчання дітей дошкільного віку, взаємодія між роботами, що імітують людську поведінку, візуальне надання відповідей на запитання та медична діагностика[8].

Базова архітектура для вирішення задачі image captioning побудована за принципом енкодер-декодер: енкодер, зазвичай згорткова нейронна мережа (CNN), витягує візуальні ознаки з вхідного зображення, а декодер - рекурентна нейронна мережа (RNN) або її вдосконалені варіанти LSTM та GRU, генерує текстовий опис на їхній основі[9].

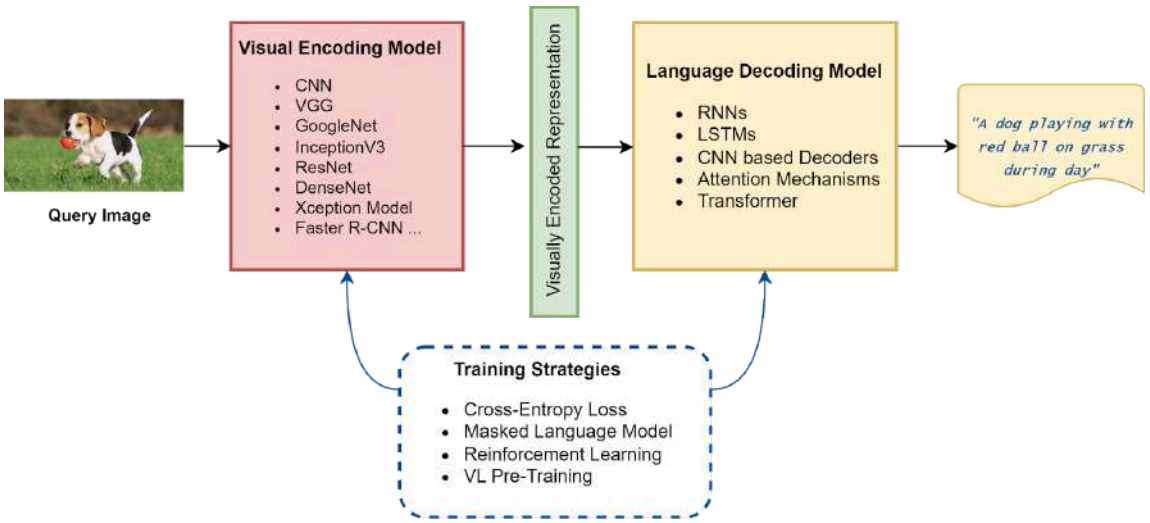

Рис. 1: Узагальнений вигляд енкодер-декодер архітектури

З часом ця архітектура зазнала значного розвитку: механізми уваги (attention layers) дозволили моделям зосереджуватись на різних регіонах зображення під час генерації кожного слова, а трансформери з механізмом self-attention вивели якість генерації описів на новий рівень[9].

З появою великих мультимодальних моделей, що поєднують потужні візуальні енкодери з великими мовними моделями, image captioning трансформувалось із вузькоспеціалізованої задачі на складову ширшого напрямку - мультимодального навчання, де межа між розумінням зображень та генерацією тексту стає дедалі менш чіткою[10]. Саме такі архітектури є предметом дослідження в даній роботі.

3.2 CLIP (Contrastive Language–Image Pre-training)

CLIP - це мультимодальна нейронна мережа, розроблена компанією OpenAI і представлена у 2021 році у роботі Radford et al. «Learning Transferable Visual Models From Natural Language Supervision» [11]. Модель навчена на масштабному датасеті з 400 мільйонів пар (зображення, текст), зібраних з відкритих інтернет-джерел [11], і демонструє здатність до узагальнення знань без додаткового навчання на цільових задачах, тобто у режимі zero-shot.

CLIP складається з двох незалежних енкодерів: візуального та текстового. Візуальний енкодер перетворює вхідне зображення на вектор у спільному embedding-просторі; в різних конфігураціях моделі він може бути реалізований як ResNet або як Vision Transformer (ViT) [11]. Текстовий енкодер на основі трансформера перетворює довільний текстовий опис на вектор у тому ж просторі.

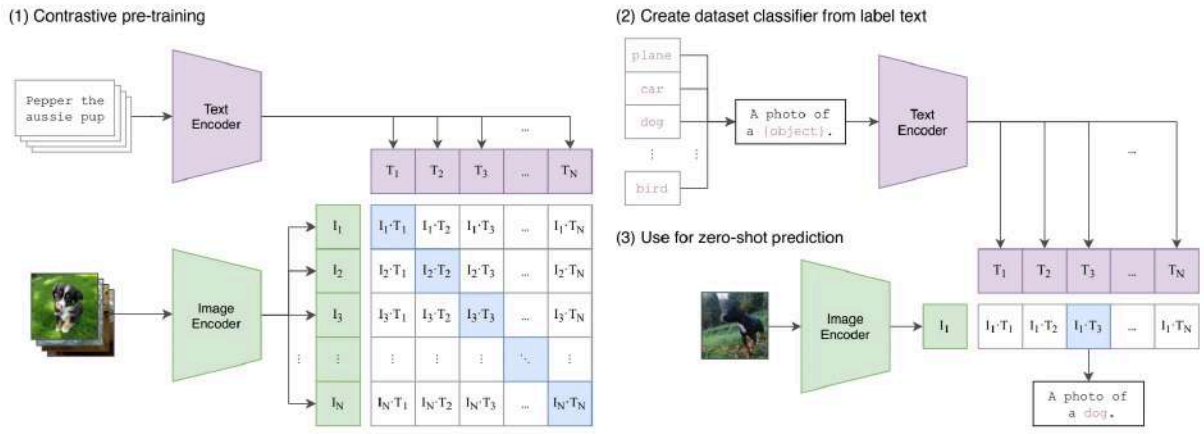

Рис. 2: Короткий опис підходу моделі CLIP

Ключова особливість архітектури полягає в тому, що обидва енкодери навчаються спільно: за допомогою контрастивного функціоналу втрат модель максимізує косинусну схожість між embeddings відповідних пар (зображення, текст) і мінімізує схожість між невідповідними парами у кожному батчі [11].

Формально, для батчу N пар (I_i, T_i) модель обчислює матрицю схожостей розміром N x N:

$$S_{ij} = \frac{f(I_i) \cdot g(T_j)}{||f(I_i)|| \cdot ||g(T_j)||} \cdot \tau$$

де $f(\cdot)$ та $g(\cdot)$ - відповідно візуальний і текстовий енкодери, τ - масштабований параметр температури. Функція втрат є симетричною крос-ентропією: діагональні елементи матриці (відповідні пари) мають бути максимальними як по рядках, так і по стовпцях [11].

Після попереднього навчання CLIP не потребує додаткового fine-tuning для вирішення нових задач [11]. У режимі zero-shot класифікації чи ранжування модель приймає зображення та набір текстових описів-кандидатів, кодує їх у спільний простір і повертає косинусні схожості між ними [12]. Це дозволяє використовувати CLIP як універсальний інструмент для семантичного зіставлення зображень і текстів без будь-якого навчання на цільових даних.

У контексті цього дослідження CLIP використовувався для кількісної оцінки якості згенерованих описів архітектурних об'єктів у режимі zero-shot. Зокрема, косинусна схожість між embedding-вектором зображення будівлі та embedding-вектором відповідного текстового опису слугувала метрикою семантичної відповідності. Цей підхід є особливо доречним для задачі image captioning, де стандартні метрики такі як Bilingual Evaluation Understudy (BLEU) чи Recall-Oriented Understudy for Gisting Evaluation (ROUGE) оцінюють лексичну, а не семантичну близькість між текстами [11]. Значення CLIP similarity, натомість, відображає, наскільки точно опис відповідає візуальному змісту зображення в семантичному просторі, що є більш інформативною мірою якості для описів, сформульованих різними словами, але з однаковим змістом.

3.3 BLIP (Bootstrapped Language-Image Pretraining)

BLIP - це фреймворк для попереднього навчання мультимодальних моделей, розроблений компанією Salesforce і представлений у 2022 році у роботі Li et al. «BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation» [13]. На відміну від більшості моделей того часу, які були оптимізовані або лише для задач розуміння зображення, або лише для генерації , BLIP об'єднує обидві парадигми у своїй єдиній архітектурі [13].

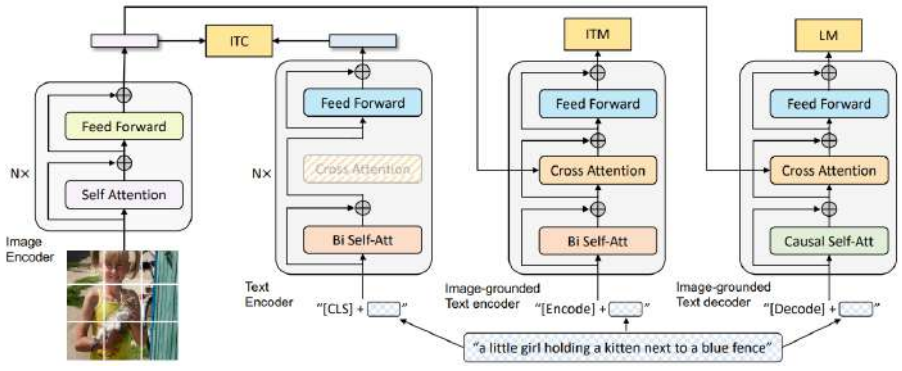

Рис. 3: Архітектура моделі BLIP перед тренуванням

На Рисунку 3 зображена пре-тренована архітектура BLIP, яка побудована на основі чотирьох взаємопов'язаних компонентах[13]:

- *Візуальний енкодер* (Vision Transformer, ViT). Перетворює вхідне зображення на послідовність patch-embeddings.
- *Текстовий енкодер* (на основі BERT). Кодує текстовий опис незалежно від зображення; використовується для контрастивного навчання зображення і тексту (ITC - Image-Text Contrastive loss), де текстові та візуальні представлення зіставляються у спільному просторі.
- *Image-grounded Text Encoder*. Також побудований на основі BERT, але доповнений шаром крос-уваги (cross-attention) між кожним трансформерним блоком, що дозволяє явно моделювати взаємодію між текстом і візуальними ознаками; використовується для задачі зіставлення зображення і тексту (ITM - Image-Text Matching), де модель бінарно класифікує, чи відповідає пара (зображення, текст) одне одному.
- *Текстовий декодер* (авторегресивна мовна модель). Генерує текст на основі візуальних ознак за допомогою однонаправленої self-attention та крос-уваги до візуального енкодера; навчений за допомогою функції втрат мовного моделювання (LM loss).

Текстовий енкодер і декодер поділяють більшість ваг, але відрізняються типом механізму уваги: енкодер використовує двонаправлену self-attention, тоді як декодер - однонаправлену, що дозволяє моделі генерувати текст послідовно[13].

Ключовою інновацією BLIP є модуль CapFilt, механізм бутстрепінгу даних, призначений для підвищення якості навчальних даних. Веб-зібрані пари зображення-текст часто містять шумові або неточні підписи, що негативно впливає на якість навчання. CapFilt вирішив цю проблему в два кроки[13].

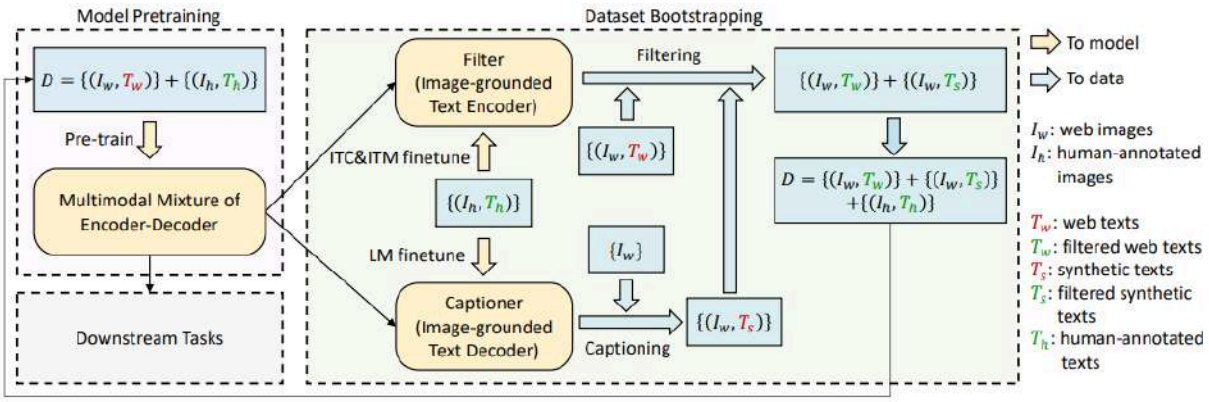

Рис. 4: Навчальний фреймворк BLIP

На рисунку 4 проілюстровано фреймворк навчання моделі BLIP та механім бутстрепінгу, що полягає у роботі двох компонентів: *captioner* (тонке налаштування декодера на датасеті Microsoft Common Objects in Context (COCO) [14]), який генерує синтетичні підписи для веб-зображень, а *filter* (також тонке налаштування енкодера на тому ж COCO) відфільтровує і оригінальні веб-підписи, і синтетично згенеровані, та видаляє ті, що не відповідають зображенню [13]. Завдяки цьому якість навчального корпусу суттєво покращується без потреби у ручній розмітці.

Для безпосередньої генерації текстових описів зображень у BLIP використовується клас “BlipForConditionalGeneration” [15]. Модель складається з візуального енкодера та текстового декодера: візуальний енкодер кодує вхідне зображення, а декодер авторегресивно генерує опис, починаючи з токена початку послідовності [BOS]. За бажанням у модель можна передати текстовий промт як початок генерації [15].

Завдяки масштабному попередньому навчанню BLIP здатний генерувати описи зображень без додаткового навчання на цільових даних [13]. У цьому режимі модель покладається виключно на знання, накопичені під час попереднього навчання, що робить її придатною для zero-shot застосування на нових доменах, зокрема на зображеннях архітектурних об'єктів.

3.4 Qwen3-VL-2B

Qwen3-VL є найновішим поколінням мультимодальних моделей у серії Qwen, розробленій командою Alibaba, та мультимодальна гілка Qwen3-VL доступна у варіантах з різним числом параметрів: від компактних моделей для локального розгортання до великих хмарних конфігурацій [16].

Qwen3-VL побудована за принципом encoder-decoder з окремим візуальним енкодером (Vision Transformer) та авторегресивним мовним декодером[16].

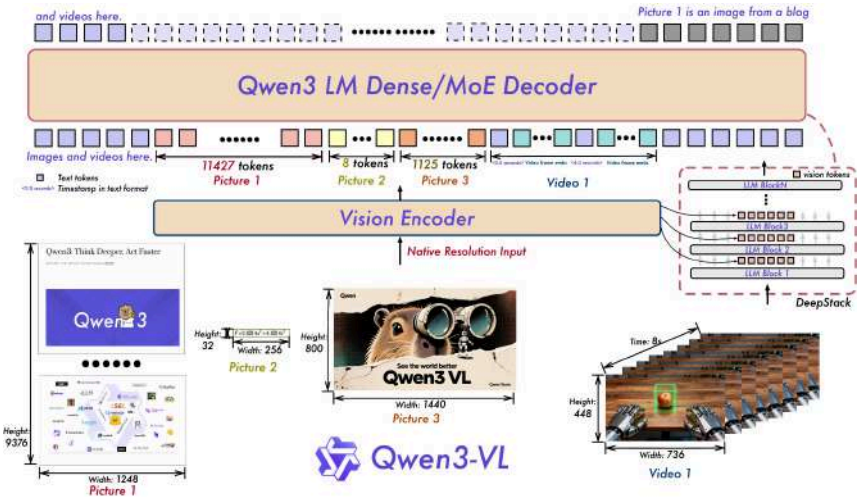

Рис. 5: Оновлення в архітектурі Qwen3-VL

Порівняно з попередніми версіями серії, Qwen3-VL запроваджує три ключові архітектурні нововведення[17], які зображені на Рисунку 5:

- *Interleaved-MRoPE* (Multimodal Rotary Position Embedding). Механізм позиційних вкладень, що рівномірно розподіляє частоти по трьох вимірах: часовому, горизонтальному та вертикальному. Це дозволяє моделі ефективніше обробляти тривалі послідовності та відеодані з довгим горизонтом.
- *DeepStack* - модуль злиття ознак із різних рівнів візуального енкодера, що дозволяє водночас враховувати як низькорівневі деталі зображення, так і високорівневу семантику, покращуючи вирівнювання між зображенням і текстом.

- *Text-Timestamp Alignment*. Механізм точного прив'язування текстових токенів до часових міток у відео, що забезпечує більш точну темпоральну локалізацію подій порівняно з попереднім підходом T-RoPE.

Qwen3-VL може демонструвати широкий спектр мультимодальних здатностей [17]: розпізнавання об'єктів, сцен та орієнтирів; розширений OCR для 32 мов із підтримкою складних умов (низьке освітлення, нахил, розмиття); просторове мислення та 2D/3D grounding; розуміння відео; а також виконання складних інструкцій у форматі діалогу (instruct-режим). Текстове розуміння моделі наближається за якістю до повністю текстових LLM, що забезпечує злиття модальностей без значних втрат[17].

У цій роботі використовувалась версія Qwen3-VL-2B-Instruct-FP8, варіант моделі з дрібнозернистою квантизацією у форматі 8-бітної числової точності з плаваючою комою та розміром блоку 128. Квантизація дозволяє суттєво скоротити обсяг оперативної та відеопам'яті, необхідний для розгортання моделі, практично без втрат у якості порівняно з оригінальною BF16-версією [17]. Це було критично важливим для проведення експериментів в умовах обмежених обчислювальних ресурсів (у даному випадку Google Colab з GPU).

3.5 Llava-1.5 (Large Language-and-Vision Assistant)

LLaVA-1.5 - це відкрита мультимодальна велика мовна модель, розроблена дослідниками Університету Вісконсін-Медісон та Microsoft Research. Вона є вдосконаленням оригінальної моделі LLaVA, запропонованої в роботі з налаштуванням візуальних інструкцій [18].

Оригінальна LLaVA вперше застосувала підхід “налаштування візуальних інструкцій” (visual instruction tuning) [18]. Концепція VIT полягає в донавчанні мультимодальної моделі на інструкційних даних, автоматично згенерованих за допомогою мовної моделі GPT-4. Ключова ідея полягає в тому, що текстовому GPT-4 надається лише текстовий опис зображення, до якого він генерує 3 різні типи пар “запитання-розгорнута відповідь”: Conversation, Detailed description, Complex reasoning, - які імітують природний діалог між людиною та зрячим асистентом. Дані, отримані таким чином, доповнюють тренувальний датасет як незалежні приклади, та дозволяють навчати модель слідувати складним інструкціям і відтворювати детальні описи на основі зображень [18].

LLaVA-1.5 використовує просту та ефективну тришарову архітектуру типу «ViT-MLP-LLM», що складається з таких компонентів [19]:

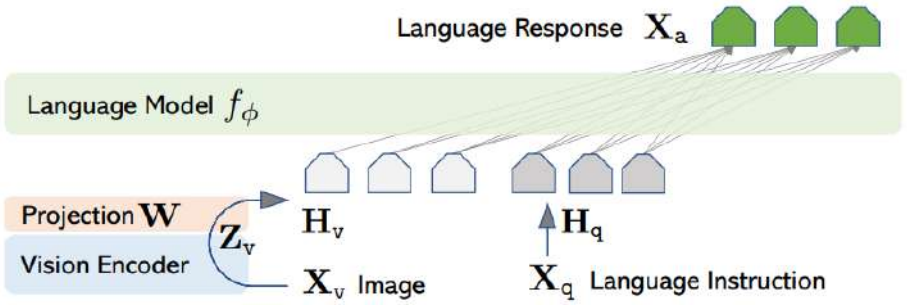

Рис. 6: Архітектура мережі LLaVA

1) *Візуальний енкодер* (CLIP ViT-L/14@336px). Для кодування зображень застосовується велика версія CLIP Vision Transformer із роздільною здатністю вхідних зображень 336×336 пікселів. Порівняно з оригінальною LLaVA, де використовувався CLIP ViT-L/14 із роздільною здатністю 224×224, підвищення до 336px суттєво покращує здатність моделі до сприйняття дрібних деталей.
2) *З'єднувальний модуль* (MLP Projector). Нововведенням LLaVA-1.5, у порівнянні з попередньою версією, є заміна простого лінійного

проєкційного шару на дворівневий багатошаровий перцептрон. Автори зазначили, що попри свою простоту, повнозв'язний MLP-конектор перевершуив складніші модулі вирівнювання [19] (наприклад, Q-Former у InstructBLIP), що були навчені на сотнях мільйонів пар зображення-текст. MLP перетворює послідовність візуальних токенів із виходу CLIP-енкодера у простір ембедингів мовної моделі.

3) *Мовна модель* (Vicuna-7B / Vicuna-13B) *як авторегресивний декодер*. Він отримує об'єднану послідовність візуальних токенів (після MLP-проєктора) та текстових токенів запиту, і генерує текстову відповідь за принципом передбачення наступного токена.

3.6 Intern VL3.5

InternVL3.5 - це сімейство відкритих мультимодальних великих мовних моделей (MLLM), розроблених командою Shanghai AI Laboratory в рамках серії InternVL. Версія InternVL3.5-4B є компактною щільною моделлю, що поєднує візуальний енкодер InternViT-300M із мовною моделлю Qwen3-4B, маючи загальну кількість параметрів близько 4,7 млрд [20].

Важливою інновацією InternVL3.5 було у використанні фреймворку Каскадного навчання з підкріпленням (Cascade RL), який покращує здатність моделі до міркування шляхом двоетапного процесу: офлайн-навчання з підкріпленням для стабільної згортності, та потім онлайн-RL для тонкого вирівнювання розподілу відповідей. Дане нововведення зумовило покращення результатів моделі до +16% в ефективності міркування у порівнянні зі своєю попередньою версією. [21].

InternVL3.5 дотримується парадигми «ViT-MLP-LLM», яка є основою всієї серії InternVL.

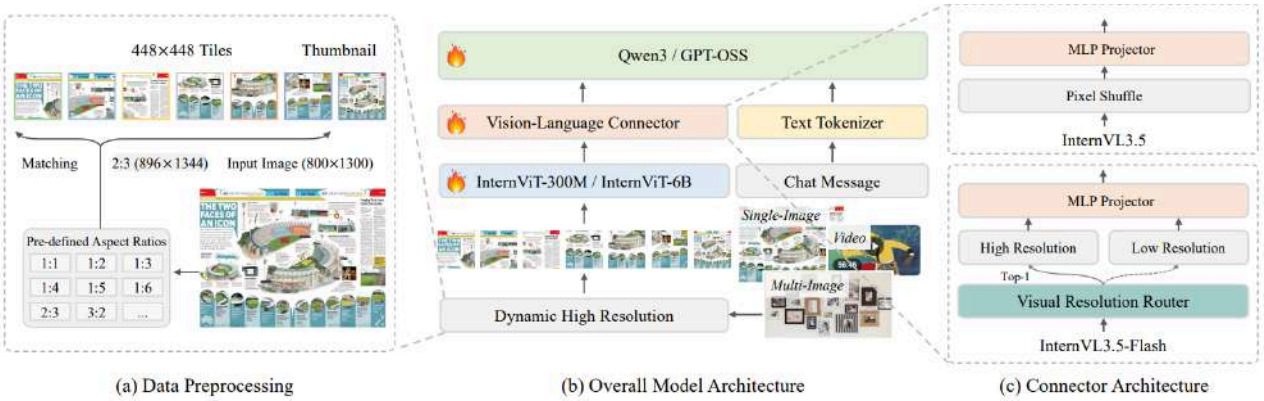

Рис. 7: Загальна архітектура InternVL3.5

На рисунку 6 зображена архітектура моделі, яка складається з трьох ключових компонентів [20]:

1) *Візуальний енкодер* (InternViT-300M). Даний енкодер є “легкою” версією InternViT, що є варіантом Vision Transformer (ViT). Він відповідає за вилучення візуальних ознак із вхідного зображення. Кожна ділянка зображення кодується у послідовність із 1024 візуальних токенів.
2) *З'єднувальний модуль* (MLP Projector). Між візуальним енкодером і мовною моделлю розміщено проєкційний шар - багатошаровий перцептрон (MLP) у поєднанні з операцією pixel shuffle. Pixel shuffle стискає 1024 візуальних токени з кожного патча до 256 токенів перед передачею до мовної моделі, зменшуючи обчислювальне навантаження без суттєвої втрати семантичної інформації [20].
3) *Мовна модель* Qwen3-4B *як декодер*. Це сучасна авторегресивна мовна модель, що генерує текстову відповідь на основі об'єднаної послідовності візуальних і текстових токенів.

Особливістю попередньої обробки зображень в InternVL3.5 є підхід Dynamic High Resolution [20]. Вхідне зображення автоматично зіставляється з набором попередньо визначених співвідношень сторін (наприклад, 1:1, 1:2, 2:3

тощо) і розбивається на плитки розміром 448×448 пікселів. Разом із плитками до моделі передається мініатюра повного зображення. Це дозволяє моделі одночасно аналізувати як дрібні деталі на рівні окремих ділянок, так і загальну композицію будівлі.

3.7 Метод донавчання низькорангової адаптації (LoRA)

Найпоширеніший підхід повного тонкого налаштування для адаптації попередньо навченої моделі до конкретної задачі передбачає оновлення всіх параметрів моделі. Проте для сучасних великих мовних і мультимодальних моделей даний спосіб є практично неможливим через надмірні витрати [22]. Крім того, повне тонке налаштування вимагає обчислення градієнтів усіх параметрів, що суттєво підвищує вимоги щодо GPU - це є критичним в умовах безкоштовного середовища виконання для даного дослідження.

Low-Rank Adaptation (LoRA) є методом параметрично-ефективного налаштування, що базується на гіпотезі про “внутрішній ранг” оновлень ваг під час адаптації моделі [22]. Попередні дослідження показали, що попередньо навчені мовні моделі мають низьку “внутрішню розмірність” і здатні ефективно навчатися навіть при проєкції у значно менший підпростір [23]. На цьому і ґрунтується метод LoRA: зміна ваг ΔW під час донавчання також має низький внутрішній ранг, і її можливо апроксимувати добутком двох малих матриць.

Математичне формулювання LoRA

Нехай $W_0 \in \mathbb{R}^{(d\times k)}$ є замороженою матрицею ваг попередньо навченої моделі. Замість прямого оновлення W_0, LoRA представляє приріст ваг через низькорангове розкладання [22]:

$$W_0 + \Delta W = W_0 + BA$$

де B ∈ ℝ^(d×r), A ∈ ℝ^(r×k), і ранг r ≪ min(d, k). Під час навчання матриця W_0 залишається замороженою і не отримує градієнтних оновлень, тоді як матриці A та B є єдиними навчуваними параметрами. Модифікований прямий прохід для входу x має вигляд [22]:

$$h = W_o x + \Delta W x = W_o x + BAx$$

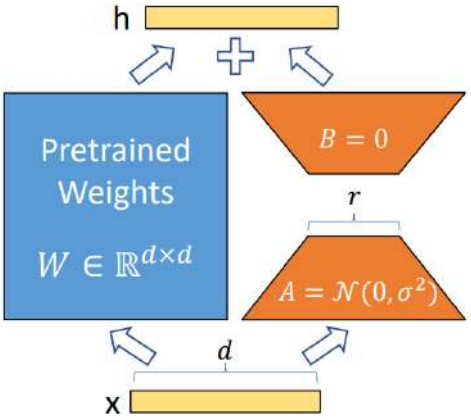

Рис. 8 Навчання матриць A та B

Матриця A ініціалізується випадковим чином із нормального розподілу, а матриця B нулями, що гарантує ΔW = 0 на початку навчання і збереження оригінальної поведінки моделі. Оновлення додатково масштабується коефіцієнтом α/r для стабілізації навчання [22].

Ефективність застосування методу LoRA визначається найбільш вдалим підбором гіперпараметрів [24]:

Параметр	Опис
r (rank)	Розмірність низькорангових матриць, зазвичай від 4 до 32. Менші значення забезпечують більше стиснення, але потенційно вони можуть знизити виразність адаптації.
lora_alpha	Коефіцієнт масштабування для шарів LoRA; зазвичай встановлюється у 2 рази більше за значення рангу. Більші значення призводять до сильнішого ефекту адаптації.

lora_dropout	Параметр, що визначає імовірність випадкового вимикання частини активацій у шарах LoRA під час навчання. Типові значення зазвичай становлять 0,05-0,1; збільшення цього параметра може сприяти зменшенню перенавчання моделі.
bias	Параметр конфігурації, що визначає, чи потрібно донавчати інші параметри зміщення під час тонкого налаштування.
target_modules	Параетр, який вказує, до яких модулів моделі слід застосувати LoRA. Більша кількість модулів забезпечує кращу адаптивність, але збільшує споживання пам’яті.

Таблиця 1 Гіперпарамтери для тонкого налаштування з LoRA

У порівнянні з повним тонким налаштуванням, підхід із LoRA має декілька суттєвих переваг.

- Ефективність використання пам’яті: вагові коефіцієнти базової моделі залишаються незмінними і можуть завантажуватися з меншою точністю, та оскільки більшість параметрів заморожена, немає потреби зберігати стани оптимізатора для них [24].
- Відсутність додаткової затримки: інференс виконується без жодних накладних витрат, адже після завершення навчання матриці A та B можуть бути злиті з вихідною матрицею $W_0 = W_0 + BA$ [23].
- Управління адаптером та модульність: матриці A та B зберігаються окремо від базової моделі і мають незначний розмір, що дозволяє легко перемикатися між задачами та обмінюватися адаптерами [23].

QLoRA

Попри суттєве скорочення параметрів для навчання моделі, метод LoRA все ще зберігає базову модель у повній точності, (зазвичай bfloat16 або float32). Для дуже великих мовних моделей це може критично обмежувати їхнє застосування в умовах обмежених ресурсів для виконання донавчання. Квантизоване донавчання методом низькорангової адаптації (QLoRA) - це метод, який “розширює” стандартне LoRA завдяки квантизації базової моделі

до 4-бітної точності [25]. Даний підхід дозволяє донавчання моделей з десятками мільярдів параметрів на одному споживчому GPU, не втрачаючи властивості якості та ефективності повного 16-бітного налаштування.

Підхід QLoRA полягає у тому, що під час навчання градієнти поширюються крізь заморожену 4-бітну квантизовану базову модель і потрапляють до адаптерів, навчених низькоранговою адаптацією [25]. Таким чином LoRA-адаптери залишаються отримують усі градієнтні оновлення та залишають повну точність, і водночас модель займає значно менше відеопам’яті.

Три ключові технічні нововведення QLoRA, які забезпечують суттєве стиснення використання пам’яті без втрати якості навчання [25]:

1. *4-bit NormalFloat* є новим типом даних, що є оптимальним для нормально розподілених ваг нейронних мереж. На відміну від стандартних 4-бітних форматів з рівномірним розподілом квантизаційних інтервалів, NF4 розміщує квантизаційні рівні відповідно до квантилів нормального розподілу. Даний підхід перевершує стандартний FP4 за точністю та забезпечує мінімальну похибку квантизації.
2. *Подвійна квантизація* - квантизація самих констант квантизації. Під час стандартної блокової квантизації кожному блоку ваг відповідає окрема константа масштабування, яка займає додаткову пам’ять. У методі QLoRA додатково квантизуються і ці константи, що дозволяє скоротити середнє використання пам’яті приблизно на 0,5 біта на параметр.
3. *Посторінкові оптимізатори* (Paged Optimizers) - це механізм управління пам’яттю, який запобігає помилкам нестачі пам'яті під час навчання на довгих послідовностях. Механізм оснований на уніфікованої пам'яті NVIDIA. Це дозволяє автоматично переносити

стани оптимізатора з GPU на CPU, якщо виникає пікове навантаження на відеопам'ять, і повертати їх назад перед кроком оновлення.

Загальний хід роботи у QLoRA відбувається відбувається таким чином: базова модель завантажується у 4-бітній точності і повністю заморожується [25]. Далі до визначених шарів трансформера, зазвичай матриці уваги, додаються LoRA-адаптери у повній точності. Під час прямого проходу відбувається квантизація “на льоту”, ваги базової моделі тимчасово деквантизуються до bfloat16 для обчислень, а потім знову квантизуються назад. Параметри базової моделі не отримують оновлень, у той час коли градієнти обчислюються лише для параметрів LoRA-адаптерів.

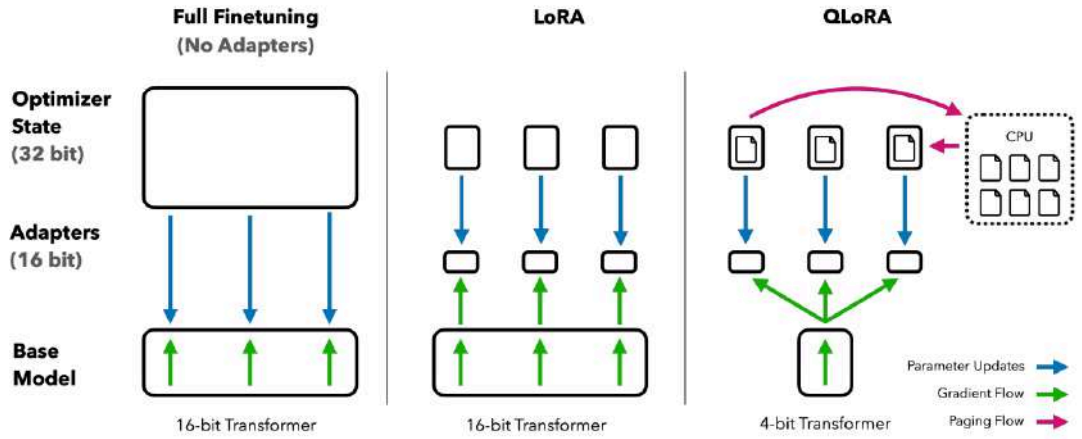

Рис. 9 Методи тонкого налаштування та їхні потреби в пам'яті

На ілюстрації 9 зображене порівняння схем роботи різних методів із донавчання моделей, зокрема переваги QLoRA в керуванні пам’яттю над стандартною LoRA завдяки квантуванню трансформера з точністю до 4 бітів та обробці пікових навантажень на пам’ять.

4 ПРАКТИЧНІ ЕКСПЕРИМЕНТИ

4.1 Збір даних

Для реалізації практичних експериментів в рамках даного дослідження було виконано поетапний збір даних у два способи.

Перший спосіб полягав у парсингу даних з інтерактивної карти ГО “Мапа Реновацій”.

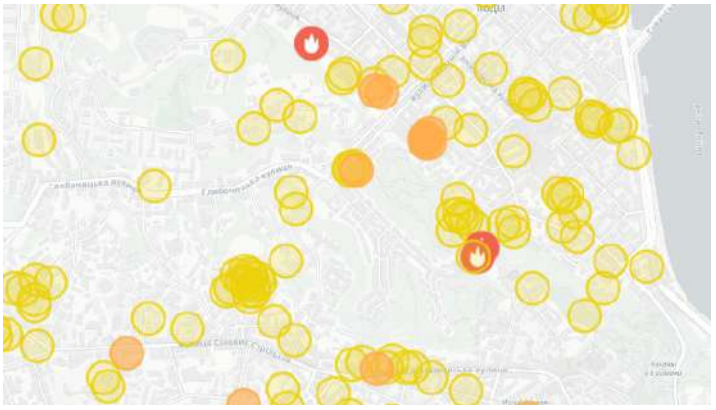

Рис. 10 Мітки із будівлями на Мапі Реновацій

На рисунку 10 зображена частина інтерактивної мапи реновацій, на якій під кожною міткою розміщена будівля в Києві, архітектурні пам’ятки, які знаходяться під потенційною загрозою знесення або потерпають від недбалого догляду.

Внаслідок парсингу даних у результаті було отримано JSON-файл із переліком пам’яток, що були розміщені на мапі, включаючи відповідну інформацію до кожної будівлі, а саме: назва, адреса, номер, id, рік побудови, рік покинутості, короткий опис та фотографія за URL-посиланням. Всього у датасеті було наявно 316 індивідуальних записів про окремі будівлі.

Рис. 11 приклад запису однієї будівлі у датасеті

Для виконання першого з практичних експериментів, оригінальні записи із будівлями були розширені додатковими полями з перекладами описів архітектури на англійську: “description_en” - для експерименту з Clip, у якого були текстові обмеження вхідних даних для енкодера, а саме відсутність підтримки української мови та ліміт на кількість токенів для вхідного тексту (77 токенів); та поле “description_en_full”, яке містить не обмежений кількістю токенів повний англомовний переклад текстових описів пам’яток.

Другий спосіб зі збору даних полягав у поетапному процесу геокодування пам’яток архітектурної спадщини міста Києва.

Перший етап полягав у комунікації з Департаментом охорони культурної спадщини КМДА, внаслідок якої було отримано Кадастровий реєстр об’єктів культурної спадщини, який не можливо було отримати безпосередньо у відкритому доступі. КРОКС - це табличний файл із переліком пам’яток у Києві, які були внесені у реєстр з атрибутами: місцезнаходження та адреса, статус, вид об’єкта (пам’ятка історії, археології, архітектури…), назва об’єкта (Біологічний інститут, житловий флігель, прибутковий будинок…), датування, охоронний номер. Оригінально табличний файл вміщав записи про 4002 об’єкти.

Першим чином було виконано фільтрацію табличних даних: відкинуто записи із порожніми значеннями, пам’ятки, що не мають архітектурного змісту (лише історичні пам’ятки, монументальне мистецтво: надгробки та кладовища, і природні ландшафти без будівель), знесені та інші нерелевантні для виконання задачі об’єкти.

Після першочергової фільтрації даних, залишилось приблизно 1758 об’єктів. Наступним етапом було витягування координат для пам’яток. Для цього було написано автоматизований скрипт в AppsScript із використанням Google Geocoding Api, який брав для основи координат адресу місцезнаходження об’єктів. В результаті табличні дані розширились додатковою колонкою із відповідними координатами для кожної будівлі, що залишились у записі.

Наступним етапом для фінального формування метаданих виконувалась поетапна обробка табличних даних:

1. Класифікація за типом будівлі. Кожен запис пам’ятки отримав мітку класу на основі ключових слів у полях назви та місцезнаходження. Об’єкт відносився до однієї із категорій: релігійна, навчальна, медична, адміністративна, промислова, культурна, житлова будівля тощо.
2. Парсинг датування. Оригінальне поле про датування об’єктів містило неструктурований текст, наприклад “2-га пол. XIX ст.”, “1883–1885”, “поч. XX ст.” тощо. Парсер нормалізував ці рядки до трьох числових колонок: year_mid, year_min, year_max, - і відповідні мітки точності (exact, range, century, decade). Для цього було реалізовано словник конвертації кирилічних та латинських римських цифр та набір регулярних виразів для розпізнавання модифікаторів (чверть, третина, половина, початок/кінець/середина).

3. Класифікація виду охорони. На основі поля “Вид об'єкта” визначалась приналежність об’єкту до однієї з категорій: architecture, architecture_urbanism або archaeology.

Результати даної обробки допомогли сформувати фінальний CSV файл з метаданими для формування майбутнього цілісного датасету.

Маючи сформовані метадані, був перехід для останнього етапу формування датасету - отримання фотографій для архітектурних пам’яток. Для реалізації цього використовувався Street View Static API від Google, що базувався на попередньо взятих координатах об’єктів та “витягував” за ними зображення. Всього на кожен об’єкт було взято по 4 зображення з різних ракурсів - в результаті було отримано 6948 зображень. Після цього було необхідно зробити фінальну чистку та прибрати нерелевантні зображення та об’єкти, для яких його отримати було неможливо, що у фінальному етапі дало 1965 зображень для 885 об’єктів.

object_id	name	address	district	type	year	year_precision	status	lat	lon	num_images	image_paths
1	Навчальна будівля	Амосова, 9	солом	architecture	1896	exact	М	50.4245305	30.5038401	4	dataset/images/obj_
2	Релігійна будівля	Андріївська, 1/1 (рі	под	architecture_urbanis	1891	range	М	50.460596	30.520113	4	dataset/images/obj_
3	Прибуткова / комер	Андріївська, 2/12	под	architecture_urbanis	1896	range	Щ	50.4600838	30.520612	4	dataset/images/obj_
4	Археологічна спору	Андріївська, 7	под	archaeology	1150	century	М	50.4615412	30.5219917	4	dataset/images/obj_
5	Прибуткова / комер	Андріївська, 9	под	architecture_urbanis	1910	exact	М	50.4618482	30.522495	4	dataset/images/obj_

Рис. 12 Приклад зображення будівлі та метаданих із датасету

Для виконання третього експерименту із донавчанням мовної моделі методом LoRA було вирішено знайти додаткове джерело відкритих даних, щоб збільшити кількість тренувальних зразків. Ми використали дані, які використовувались для проекту OpenFacades [26]. Датасет налічує 19443 унікальних зображень будівель разом із метаданими з атрибутивними мітками: "building_type", "building_age", "floors", "surface_material", "construction_material".

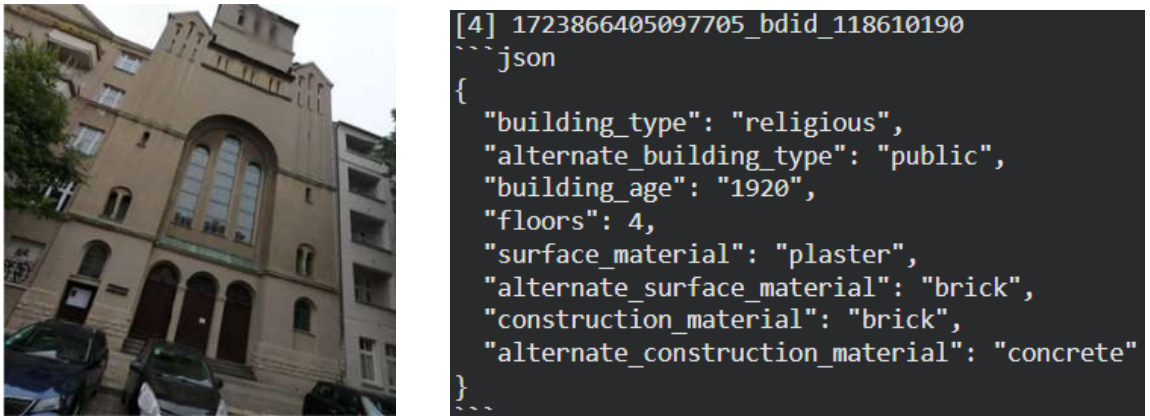

Рис. 13 Приклад зображення будівлі з датасету OpenFacades

4.2 Zero-shot експерименти

Мета zero-shot досліджень або експериментів з мультимодальними моделями без будь-якого донавчання на цільовому датасеті полягала у визначенні найкращої та найбільш підходящої моделі для майбутнього тонкого налаштування з LoRA на тренувальних даних. Найбільш задовільна модель обиралась за принципом балансування: висока точність та водночас доступний розмір для обмеженого середовища виконання. Тестова вибірка для всіх експериментів є фіксованою: seed=42 для 64 об'єктів.

Clip

Перший експеримент проводився із Clip у конфігурації “openai/clip-vit-base-patch32”, завантаженої через бібліотеку Hugging Face Transformers. Як вхідні дані для Clip надавалися зображення за URL та еталонний текстовий опис англійською, який був попередньо доданий у поле датасету “description_en”. Векторні представлення зображення та тексту отримувались окремо через відповідні енкодери моделі й нормалізувались до одиничної норми; міра відповідності між ними обчислювалась як косинусна подібність між цими векторами. Отримана оцінка після проведення експерименту з Clip могла відобразити, наскільки добре передтренована модель семантично може узгодити візуальний контент будівлі з її відповідним описом.

Blip

Для першої zero-shot генерації описів використовувалась модель “Salesforce/blip-image-captioning-large”, завантажена через класи BlipProcessor та BlipForConditionalGeneration також з бібліотеки Hugging Face Transformers (HFT). Препроцесор поєднує обробку зображень через “BlipImageProcessor” та токенізацію на основі BERT. Експеримент проводився у двох режимах: безумовна генерація - без будь-якого текстового входу, коли модель самостійно розпочинала опис із токена [BOS], та умовна генерація - з коротким текстовим підказом "a historic building". Генерація здійснювалась методом пошуку з п'ятьма променями (num_beams=5, max_new_tokens=50).

Qwen3-VL-2B

Наступний експеримент проводився з моделлю “Qwen/Qwen3-VL-2B-Instruct”, завантаженою через класи “Qwen3VLForConditionalGeneration” та “AutoProcessor” з бібліотеки HFT.

Зображення та текстовий промпт подавались у форматі chat-template через метод apply_chat_template; на відміну від інших моделей, Qwen3-VL приймає URL зображення безпосередньо у полі повідомлення без попереднього завантаження через PIL. Для обмеження споживання пам'яті препроцесор налаштовувався через параметри min_pixels та max_pixels, що дозволило запускати модель у середовищі Colab без перевищення ліміту GPU-пам'яті. Генерація здійснювалась у режимі жадібного декодування (do_sample=False, max_new_tokens=80), а токени вхідного промту зрізались з виведення перед декодуванням. Промт, який використовувався для всіх оглянутих моделей у zero-shot експерименті зображений на Рисунку 14.

```
PROMPT = (
    "You are an architectural analyst. Describe the building in the image in one to two sentences (30–60 words). "
    "Follow this structure: (1) number of stories and massing, (2) construction material and facade treatment, "
    "(3) architectural style or period, (4) approximate era of construction, "
    "(5) current condition or notable functional detail if clearly visible. "
    "Do not name specific people or owners. Do not speculate beyond what is visually apparent. "
    "If the building is ruined or unfinished, state that explicitly. "
    "Use neutral, factual architectural language. "
    "Example: 'Four-story brick income house in the Neo-Renaissance style, built in the early 20th century; "
    "the facade features ornate plasterwork pilasters and a decorative cornice. "
    "The ground floor shows signs of later commercial alteration.'"
)
```

Рис. 14 Єдиний промт для zero-shot експериментів

Llava-1.5

Третій zero-shot експеримент проводився з моделлю LLaVA-1.5 у конфігурації “llava-hf/llava-1.5-7b-hf”, завантаженої через HFT за допомогою класу “LlavaForConditionalGeneration”. Модель завантажувалась у форматі bfloat16 з автоматичним розподілом шарів по доступних пристроях (device_map="auto").

На відміну від сучасніших мультимодальних архітектур, LLaVA-1.5 не підтримує стандартний механізм apply_chat_template, тому запит формувався вручну у фіксованому форматі USER: <image>\n{prompt}\nASSISTANT:, де спеціальний токен <image> замінювався процесором на патч-ембедінги зображення розміром 336×336.

Зображення завантажувались за URL після чого модель у режимі жадібної генерації продукувала архітектурний опис будівлі. Для отримання лише згенерованої частини відповіді з виведення зрізались токени вхідного промпту.

InternVL3.5

Останній експеримент проводився з моделлю InternVL3.5 у конфігурації “OpenGVLab/InternVL3_5-4B-HF”, завантаженої через класи “AutoModelForImageTextToText” та “AutoProcessor” також з HFT.

Модель завантажувалась у форматі bfloat16 з автоматичним розподілом шарів. На відміну від LLaVA-1.5, InternVL3.5 підтримує стандартний інтерфейс “apply_chat_template”, що дозволило передавати за URL зображення безпосередньо у структурованому повідомленні без ручного формування промпту. Запит будувався у форматі розмови з роллю користувача, де зображення та текстова інструкція передавались як окремі елементи контенту; процесор самостійно завантажував і кодував зображення під час токенізації. Генерація здійснювалась у режимі жадібного декодування, і токени промту зрізались з виведення перед декодуванням.

4.3 Донавчання моделі методом LoRA

В результаті проведення попередніх експериментів з дослідженням ефективності мультимодальних моделей, було прийнято рішення зупинитись на використанні Qwen3-VL-2B-Instruct для процесу тонкого налаштування. Дана модель показала високий результат ефективності в генерації текстових описів для будівель у поєднанні із доступною компактністю, що задовольняє використане обмежене середовище виконання серед досліджуваних архітектур інших моделей.

У даній роботі ціль дослідження із тонким налаштуванням LoRA полягала в спробі навчити модель дотримуватись заданого шаблону й формату виводу, і також вказувати про неможливість визначення атрибуту, коли зображення його не має.

Навчання здійснювалось за допомоги підходу QLoRA. Базова модель завантажувалась у 4-бітній квантизації у форматі NF4 (BitsAndBytesConfig, bnb_4bit_use_double_quant=True), після чого поверх неї накладались адаптери LoRA з початковими параметрами:

- r=16,
- lora_alpha=16,
- learning_rate=1e-4,
- batch=8,
- epoch=2
- lora_dropout=0.05.

У процесі тонкого налаштування дані гіперпараметри могли поетапно змінюватись згідно їхнього впливу на результати навчання моделі.

Адаптери застосовувались до семи проєкційних матриць трансформерних блоків: q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj.

Для збільшення кількості тренувальних зразків використовувався зовнішній загальнодоступний датасет “seshing/openfacades-dataset”, публічні дані з фасадами будівель із Hugging Face Hub [27].

Датасет завантажувався у форматі JSONL, де кожен запис до відповідної будівлі містив пару “запитання-відповідь” у форматі розмови з LLM. Як метадані для навчання було використано записи, які відбирались за конкретним префіксом (“TARGET_QUESTION_PREFIX” у коді), що вказував на структурований, а не вільний опис фасаду будівлі. Після чого з

відфільтрованої множини описів випадковим чином обирались 1000 або 2000 (в залежності від попереднього результату експерименту) зразків, зберігаючи seed=42. В кінці вибірка даних ділилась у співвідношенні 90/10 на тренувальну та тестову частини.

Промт для навчання структурованому виводу моделі із явно визначеними значеннями атрибутів зображений на Рисунку 15.

```
PROMPT = (
    'You are an architectural analyst. Examine the building facade in the image carefully. '
    'Describe what you observe using exactly these attributes:\n'
    '- Building type: (one of: apartments, house, retail, office, hotel, industrial, religious, education, public, garage)\n'
    '- Alternate building type: (another option from the same list, or "not visible")\n'
    '- Approximate age: (a 4-digit year, e.g. "1921")\n'
    '- Number of floors: (a numeric value)\n'
    '- Surface material: (one of: brick, wood, concrete, metal, stone, glass, plaster)\n'
    '- Alternate surface material: (another option from the same list, or "not visible")\n'
    '- Construction material: (one of: brick, wood, concrete, steel, other)\n'
    '- Alternate construction material: (another option from the same list, or "not visible")\n'
    'IMPORTANT rules:\n'
    '1. If an attribute cannot be determined from the image, write "not visible".\n'
    '2. "Alternate" fields are for secondary options — if only one is visible, write "not visible".\n'
    '3. Do not speculate beyond what is visually apparent.\n'
    '4. Write only the attribute list, no extra commentary.'
)
```

Рис. 15 Промт для тонкого налаштування з LoRA

Обробку даних було реалізовано через самостійно написаний колатор VMSCollator, а саме дані завантажувались із локального диску та передавались у форматі “chat-template” разом із текстовим промптом. Для оцінки результатів навчання моделі використовувалась функція втрат, яка обчислювалась лише на токенах відповіді. Самі токени промпту маскувались значенням -100. Процес навчання проводився експериментально від 2 до 3 епох, та значення швидкості навчання також варіювалось від 1e-4 до 5e-5 од.б.р. У процесі значення батчу серед параметрів не змінювалось - 8, (сам розмір батчу 1 з акумуляцією градієнту у 8 кроків).

Для тренування моделі використовувався графічний процесор NVIDIA L4. Для тренування було використано тип даних bf16 (Brain Floating Point 16), що дозволило зменшити споживання пам’яті без суттєвої втрати точності обчислень.

Для спостереження динаміки навчання моделі було реалізовано колбек “LossLoggerCallback”, що фіксував тренувальні та валідаційни втрати на кожному кроці логування. За результатами навчання фінальні адаптери LoRA зберігались окремо від базової моделі, та також будувався графік кривих втрат.

4.4 Атрибутивний метод оцінки ефективності моделей

У ході виконання практичних експериментів для оцінки ефективності мовних моделей спершу використовувались такі метрики як: BLEU, METEOR та ROUGE-L. Дані метрики є стандартними для оцінки задач текстово генерації, і ґрунтуються на точному лексичному збігу між референсним описом та згенерованим текстом моделі [28]. Проте в контексті виконання цієї дипломної роботи вищезгадані метрики є суттєво обмеженими у використанні через семантичні розбіжності між форматами виводу в моделей. Зокрема, у практичних дослідженнях моделі генерують текст у структурованому стилі, націленому на формат документацій, коли оригінальні описи архітектурних об’єктів були представлені у формі вільного тексту.

Для забезпечення більш реалістичної та змістовної оцінки роботи моделей було реалізовано атрибутивний метод оцінювання, що полягає в екстракції структурованих архітектурних атрибутів з двох текстових описів: референсу та генерації із подальшим порівнянням між собою.

Після донавчання шляхом низькорангової адаптації модель Qwen3-VL-2B-Instruct безпосередньо надає відповідь у структурованому JSON-форматі, тому для “витягування” атрибутів із згенерованого тексту здійснювалось завдяки прямому парсингу результату без допомоги зовнішніх інструментів. Коли як для вільного від структури еталонного тексту виконати екстракцію лише парсингом було неможливо. Для екстракції референсів було

залучено Groq API з моделлю LLaMA-3.3-70B. Для цього модель отримала інструкції у промті з переліком допустимих значень для кожного атрибуту та правил конвертації наближених часових згадок (десятиліть та епох) у числові значення року. Промт з інструкцією для екстракції атрибутів зображений на Рисунку 16:

```
REF_EXTRACTION_PROMPT = """Extract architectural attributes from the building description below.
Return ONLY a valid JSON object with exactly these keys. No explanation, no markdown.

Keys and allowed values:
- "building_type": list of strings from {building_types} (empty list if not mentioned)
- "building_age": integer (4-digit year) or null if year/decade/era is not mentioned at all.
  Conversion rules for approximate mentions:
  - Exact year (e.g. "1921", "built in 1935") → use as-is
  - Decade (e.g. "1920s", "the 1970s") → use the first year of the decade (1920, 1970)
  - "early Xth century" → X*100 - 90  (e.g. "early 20th century" → 1910)
  - "mid Xth century"   → X*100 - 50  (e.g. "mid 20th century"   → 1950)
  - "late Xth century"  → X*100 - 10  (e.g. "late 19th century"  → 1890)
  - "post-war"          → 1950
  - "Soviet-era" alone (no decade given) → 1960
  - If only a broad era like "19th century" with no qualifier → use X*100 - 50 (midpoint)
- "floors": integer (number of floors) or null if not mentioned
- "surface_material": list of strings from {surface_materials} (empty list if not mentioned)
- "construction_material": list of strings from {construction_materials} (empty list if not mentioned)

Text: "{text}"
```

Рис. 16 Промт для екстракції значень атрибутів із референсів

Відповідно до промту доналаштованої Qwen3-VL-2B, було визначено п'ять основних атрибутів для оцінювального порівняння текстових описів, що були поділені на два типи:

- *Листові атрибути*, (що обчислювались через Precision, Recall, F1): тип будівлі, матеріал поверхні, будівельний матеріал. Кожен листовий атрибут міг приймати кілька значень одночасно.
- *Скалярні атрибути*: рік будівництва, що оцінювався через точність із допуском +-10 років; та поверховість, що рахувала точний збіг, (із об'єднанням значень більше п’яти в один клас.

Також сумарно ще відбувався обчислення метрики macro average, що включала листові атрибути F1 разом із скалярною точністю. В межах даної атрибутивної оцінки додатково була порахована статистика із відсотком зразків, де атрибути взагалі були вказані, а де першочергово ні.

Допуск із похибкою в 10 років для року будівництва був зумовлений тим, що еталонні описи часто мали наближені дати, наприклад "1920-ті роки" або "початок XX століття", які під час витягування конвертувались у відповідне значення десятиліття чи епохи, згідно правилам екстракції у промті.

В результаті виконання оцінки атрибутивним методом було звернуто увагу на реалістичні обмеження даного підходу, зокрема те, що цей метод залежить від якості LLM-екстракції з еталонних описів, яка не є абсолютно точною. Також велика частина оцінювальних атрибутів могла бути відсутньою, особливо в описах-референсах. Як протидія, такі зразки були виключені з підрахунку відповідної метрики, що було відображено у статистиці покриття атрибутів у даних.

Необхідно зазначити, що набори атрибутів для атрибутивної оцінки (і відповідно промт) zero-shot експериментів та донавченої Qwen3-VL-2B відрізняються, що було зумовлено суттєвою відмінністю у форматі виводу моделей. У zero-shot експерименті моделі отримували промт, що передбачав генерацію вільного текстового опису будівлі, тому оцінювані атрибути (поверховість, ера, стан, матеріал, стиль) витягувались із неструктурованого тексту за допомогою LLM-екстракції. А тонке налаштування Qwen3-VL-2B проводилось на промті із явно визначеними категоріями атрибутів, відповідно і оцінювання цих категорій було можливо реалізувати парсингом структурованого виводу. Таким чином, таблиці атрибутивної оцінки для двох режимів не є прямо порівняльними, а такими, що відображають якість роботи кожної групи моделей у межах власного формату виводу.

5 РЕЗУЛЬТАТИ

5.1 Результати Clip-експерименту 1

Загальна статистика підрахунку косинусу подібності між векторами у результаті проведення першого експерименту з Clip на 316 зразках (з них 64 тестових) архітектурних об’єктів із “Мапи Реновацій” наведена нижче:

- Середнє значення (mean): 0.2605
- Медіана (median): 0.2645
- Стандартне відхилення (std): 0.0298
- Мінімальне значення (min): 0.1959
- Максимальне значення (max): 0.3271

Також графіки розподілу подібності векторів зображені на Рисунку 17

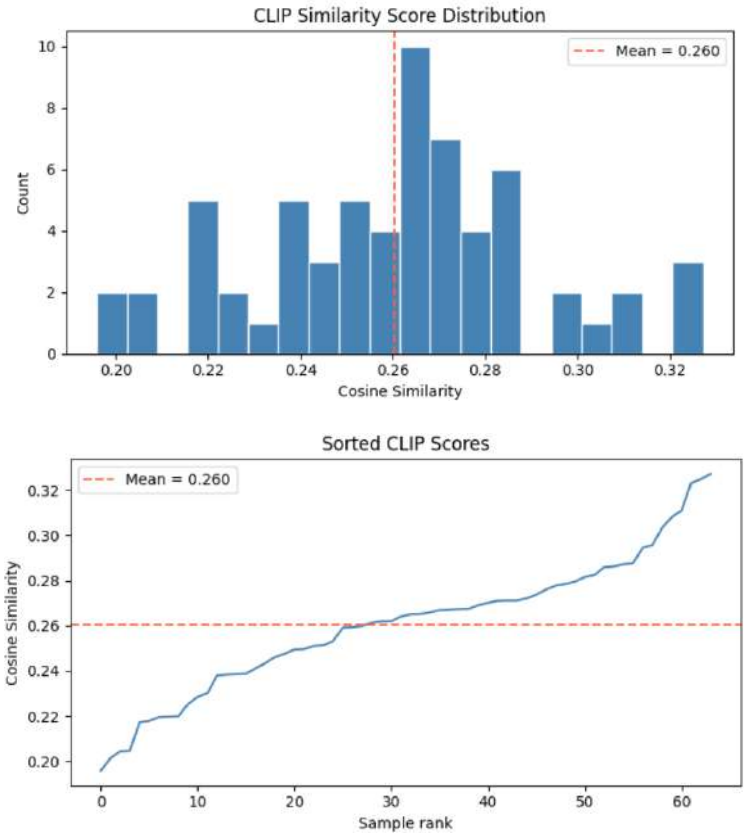

Рис. 17: Графіку із розподілом подібності та оцінкою експерименту Clip

Ілюстрації на Рисунку 15 підкріплюють загальну статистику розподілу значень. Другий графік є ранжованим розподілом, який показує, що оцінки рівномірно охоплюють діапазон від від 0.20 до до 0.33 без різких стрибків. Це може означати відсутність явних кластерів серед зразків, і модель робила оцінку відносно рівномірно. Загалом Clip стабільно і помірно обчислював косинусну подібність між зображеннями будівель та текстовими описами без виразних аномалій у даних.

На Рисунку 18 показані по два зображення, для яких оцінка косинусної подібності сягала найвищого та найнижчого результату відповідно.

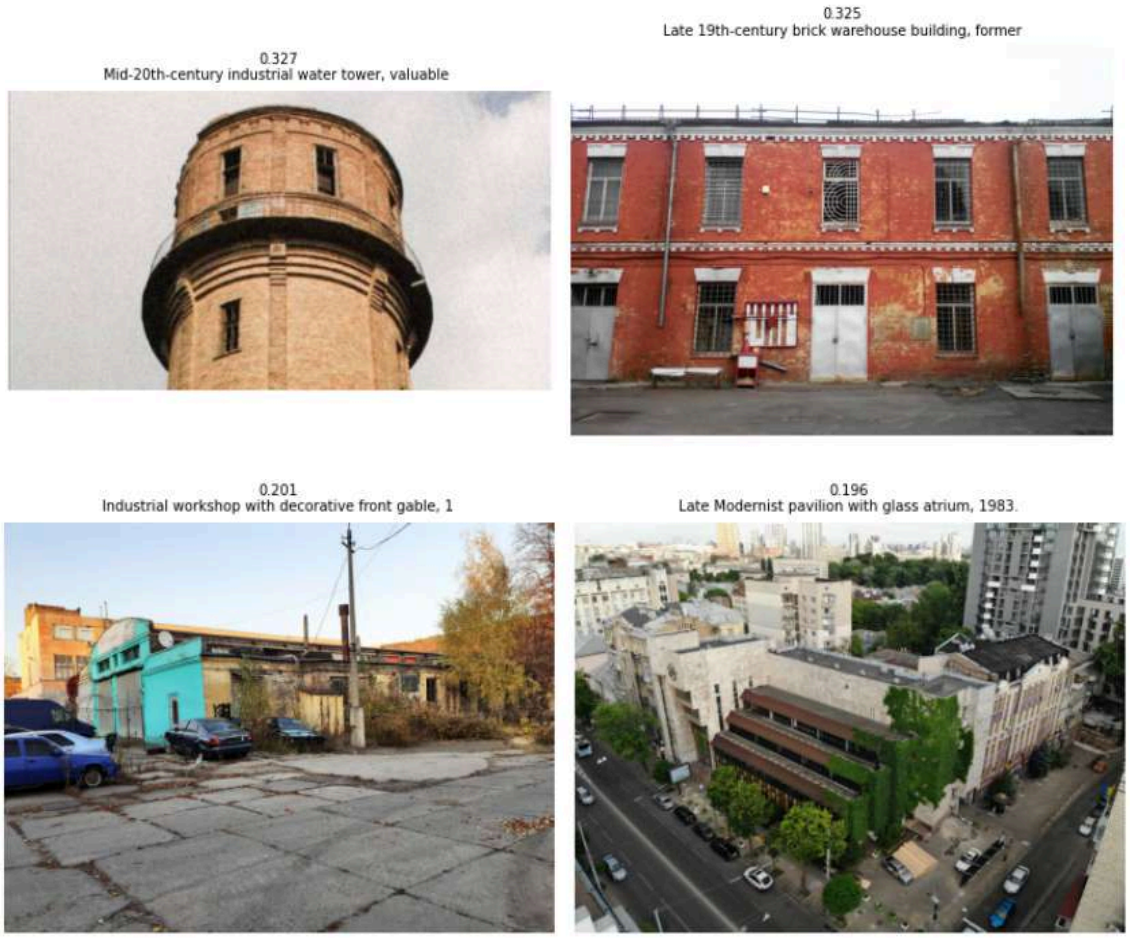

Рис. 18 Зображення із найвищою і найнижчою оцінками подібності

З ілюстрацій на Рисунку 18 можна зробити висновок, що кращі результати подібності отримали більш якісні та характерні зображення, коли фотографії

нижче на рисунку мають нижчу оцінку подібності, імовірно через більшу кількість сторонніх об’єктів та нижчу якість на фото.

Загалом у ході експерименту з Clip важливим результатом було виявлення ключових обмежень моделі у контексті досліджуваної задачі: відсутність підтримки української мови та критична лімітованість текстового енкодера у 77 токенів. Це спонукало адаптації текстових описів вхідних даних, а саме їхній переклад на англійську та розширення датасету додатковим полем “description_en”, що враховувало кількісне обмеження токенів.

Варто також зауважити, що виявлені обмеження у роботі моделі могли впливати на наслідки даного практичного експерименту - середні значення обчислених подібностей текстового та візуального вбудованих векторів загалом показали не високі результати оцінки (із середнім значенням 0.2605).

5.2 Результати zero-shot експериментів 2

Blip

Приклад результату генерації текстового опису для зображення будівлі з id “1BqN3” без додаткового навчання моделлю Blip-1 зображений на Рисунку 19.


```
"caption_free": "there is a white car parked in front of an old building",
"caption_prompted": "a historic building with a car parked in front of it",
"ground_truth": "Two-story brick residential building, pre-revolutionary.",
```


Рис. 19 Приклад роботи zero-shot Blip

Як видно із ілюстрації на Рисунку 19, Blip більше була націлена на генерацію опису для загальної сцени зображення: погоду, транспорт та інші зовнішні чинники - замість архітектурних характеристик будівлі. Через це результат роботи ефективності моделі не був високим, що й також відобразилось на метриках оцінки:

- "rouge1_free": 0.1145, "rouge1_prompted": 0.133;
- "rouge2_free": 0.0117, "rouge2_prompted": 0.0115;
- "rougeL_free": 0.1028, "rougeL_prompted": 0.1118;
- "bertscore_f1_free": 0.384,
- "bertscore_f1_prompted": 0.4373;

Qwen3-VL-2B

Приклад генерації тексту моделлю Qwen3-VL-2B-Instruct для однакової будівлі, що і на Рисунку 19:

- "generated": "Two-story building with a flat roof and a central arched entrance, constructed of brick and plaster. The facade features decorative ironwork on the windows and a simple awning. The building shows signs of age and partial decay, with visible graffiti and a small, uncompleted section on the ground floor."

З наведеного прикладу можна побачити велику різницю в розгорнутості та точності генерації опису. Попри кращу якість генерації тексту, n-gram метрики все одно не могли врахувати семантичну різницю між форматами виводу моделі та еталонного опису будівлі, що й відповідно не вплинуло позитивно на підвищення метрик для оцінки ефективності:

- "BLEU-1": 0.0786,
- "BLEU-4": 0.0008,
- "METEOR": 0.1161,

- "ROUGE-L": 0.11;

Llava-1.5

Приклад генерації текстового опису для тієї ж самої будівлі, але за допомоги Llava1.5-7B-HF:

- "generated": "The building is a four-story brick structure with a white facade, featuring a decorative cornice and ornate plasterwork pilasters. The architectural style appears to be Neo-Renaissance, and the building was constructed in the early 20th century. The ground floor shows signs of later commercial alteration."

На прикладі можна побачити, що Llava-1.5 також здатна генерувати злагоджені та розгорнуті текстові описи будівель. Проте варто звернути увагу, що дана модель неодноразово суттєво помиляється від фактичного зображення будівлі. Зокрема в даному прикладі Llava-1.5 явно неправильно вказала на кількість поверхів (чотири замість двох), наближений рік побудови (20-те століття замість 1890-тих або 19-те століття) та стиль (неоренесанс замість дореволюційної архітектури).

Як і попередньо, стандартні метрики не відобразили реальної різниці у похибках між генеративною відповіддю моделі та референсом:

- "BLEU-1": 0.0878,
- "BLEU-4": 0.0032,
- "METEOR": 0.1167,
- "ROUGE-L": 0.1114,

InternVL3.5

Приклад генерації опису для того ж самого зображення для будівлі моделлю InternVL3.5:

- "generated": "The two-story building features a simple, rectangular massing with a beige facade and white trim. It has barred windows and a mix of exposed brick and plaster, suggesting early 20th-century construction. The ground floor includes an arched passageway and a small commercial entrance with a striped awning, showing signs of wear and graffiti."

У відповіді модель надала змістовну, розгорнуту та переважно правильні деталі із зображенні, включаючи кількість поверхів, фасад, матеріали та стан будівлі. Попри очевидну вищу якість генерації тексту, n-gram метрики не змогли надати суттєво вищі оцінки для ефективності моделі:

- "BLEU-1": 0.083,
- "BLEU-4": 0.0016,
- "METEOR": 0.1245,
- "ROUGE-L": 0.1226;

Окрім стандартних метрик на отриманих відповідях моделей був застосований атрибутивний метод оцінки, попередньо описаний у практичному розділі, що зміг дати більш реалістичних погляд на результати ефективності генерацій оглянутих мовних моделей.

Для zero-shot експериментів обчислювались такі атрибути: *скалярні* (кількість поверхів, ера побудови, стан будівлі) та *листові* (матеріал та стиль архітектури). Таблиця із оцінюваним порівнянням моделей наведена нижче.

	stories	era	condition	material			style			macro avg
				precision	recall	F1	precision	recall	F1	
Blip	0.0	0.0	0.1429	0.1667	0.1667	0.1667	0.0	0.0	0.0	0.0619
Qwen3-VL	**0.5882**	**0.25**	**0.2273**	**0.619**	**1.0**	**0.7381**	**0.05**	**0.05**	**0.05**	**0.3707**

Llava-1.5	0.2353	0.2542	0.2727	0.625	0.6875	0.6458	0.1667	0.1667	0.1667	0.315
InternVL3.5	**0.7778**	**0.4167**	**0.3043**	**0.5625**	**0.75**	**0.625**	**0.25**	**0.3125**	**0.2708**	**0.4789**

Таблиця 2 Атрибутивна оцінка всіх досліджених архітектур

Статистика покриття, тобто відсоток вказаних атрибутів у відповідях моделі, допоміг обгрунтувати деякі результати генерації моделей. Особливо у випадку моделі Blip: через високу концентрацію на загальній сцені зображення, а не архітектури будівлі, Blip майже не генерував вказані атрибути, включаючи кількість поверхів (атрибути були вказані в 1.6% відповідях) та еру побудови (1.6% також) - тобто модель скоріше не намагалась навіть зачепитися за архітектурні характеристики. У такому випадку найнижчі результати для Blip є більш зрозумілими.

У Qwen3-VL-2B стовідсоткове покриття атрибутів сягало майже всюди, окрім ери побудови (64.1%) та стилю (50%). Для всіх моделей “найважчим” атрибутом для ідентифікації виявився стиль, як і для InternVL3.5, яка на даному атрибуті має найнижче покриття, у порівнянні з іншими - 82.8%. Можна зробити висновок, що стиль є найскладнішим атрибутом для визначення через суб’єктивність та складну категорію, і додатково недостатню кількість прикладів у референсах.

З результатів описаних у Таблиці 2 та найбільших відсотків покриттів атрибутів, можна побачити що найкращі результати в ефективності генерації описів будівель показали дві моделі - Qwen3-VL-2B та InternLV3.5-4B. Зокрема найвищий macro average сягнув для другої моделі (0.4789). Проте хоч дана модель і має найкращі показники, вона вимагала б значно більш потужного середовища виконання для процесу тонкого налаштування, навіть враховуючи застосування підходу низькорангової адаптації. Попри нижчий macro_avg, Qwen-2B показала порівнянні результати по кількості поверхів та

навіть кращі результати по ідентифікації матеріалів будівлі (F1=0.7381) за InternLV3.5 при вдвічі меншому розмірі. Тому для донавчання було обрано саме Qwen3-VL-2B-Instruct через доступні обчислювальні потужності та достатньо високу якість генерації тексту.

5.3 Результати LoRA-експерименту 3

Результати тонкого налаштування методом низькорангової адаптації Qwen3-VL-2B і досліджені гіперпараметри наведені в Таблиці 3.

Run	R	α	Epoch	LR	Batch	Min Train Loss	Min Eval Loss	Примітки
1	16	16	2	1e-4	8	0.4390	0.4743	GPU: T4, зразки: 1000, SF=1
2	16	32	2	1e-4	8	0.3619	0.4688	зміна: SF=2
3	16	32	2	1e-4	8	0.3365	0.6330	зміна: зразки=2000
4	16	32	3	5e-5	8	0.2882	0.7468	зразки: 2000, зміна: GPU L4
5	**16**	**32**	**2**	**1e-4**	**8**	**0.3065**	**0.6451**	GPU: L4, зміна: зразки 1000
6	16	32	2	1e-4	8	0.3201	0.7467	GPU: L4, зміна: зразки 2000

Таблиця 3 Гіперпарамтери донавчання з LoRA

У процесі експериментів послідовно змінювався один або декілька параметрів із метою покращення якості навчання. Першочергово було перевірено вплив коефіцієнту масштабування LoRA (SF = α/R), збільшення SF з 1 до 2 показало позитивний вплив на eval loss, тому цей параметр залишався незмінним у подальших запусках. Наступні експерименти налаштування були спрямовані на дослідження впливу обсягу навчальних даних (у кількості зразків від 1000 до 2000) та зниження LR для запобігання перенавчанню. Після чого також досліджувалась поведінка моделі при навчанні на GPU L4,

що зумовило відповідні зміни в конфігурації (зокрема, увімкнення формату bfloat16).

Попри те, що запуск 2 продемонстрував найнижче значення eval loss (0.4688), він проводився в умовах іншої конфігурації масштабування та на GPU T4. Запуски 3, 4 та 6 показали суттєвий розрив між train та eval loss, що свідчило про перенавчання моделі. Запуск 5, проведений на GPU L4 з 1000 навчальними зразками та стандартними гіперпараметрами (LR=1e-4, epochs=2), продемонстрував найкращий баланс між train (0.3065) та eval loss (0.6451) серед L4-запусків, що й зумовило його вибір для майбутнього інференсу.

Збережені чекпоінти були використані для інференесу моделі та її запуску на тестових даних для порівняння із проведеними практичними експериментами. Приклад генерації тексту для донавченої Qwen3-VL-2B для попередньої будівлі з id “1BqN3”:


```
{
    "building_type": "house",
    "alternate_building_type": "apartments",
    "building_age": "1950",
    "floors": 2,
    "surface_material": "plaster",
    "alternate_surface_material": "concrete",
    "construction_material": "wood",
    "alternate_construction_material": "brick"
}
```


Як і на даному прикладі, модель навчилась генерувати текстовий опис для будівель у структурованому форматі, націленому на інформаційне документування, для усіх зображень будівлі.

Результати атрибутивної оцінки ефективності для донавченої Qwen3-VL, включаючи зміни у виводі для атрибутів (скалярні: кількість поверхів та приблизний вік побудови; і листові: тип будівлі, матеріал поверхні, будівельний матеріал) наведені у Таблиці 4.

	floors	build age	building type			surface material			construction material			macro avg
			precision	recall	F1	precision	recall	F1	precision	recall	F1	
Донавчена Qwen3-VL	0.9167	0.3167	0.2909	0.5152	0.3648	0.3889	0.7778	0.5185	0.4375	0.875	0.5833	**0.54**

Таблиця 4 Атрибутивна оцінка для до-налаштованої Qwen3-VL

Виходячи із результатів атрибутивної оцінки зображеної в Таблиці 4, донавчання Qwen3-VL-2B допомогло підвищити точність для атрибутів: кількість поверхів (0.5882→0.9167) та приблизному році побудови (0.25→0.3167), зокрема саме значення macro average підвищилось з 0.3707 до 0.54. Також модель навчилась генерувати змістовні та структуровані відповіді, вдало розпізнаючи архітектурні особливості будівель.

6 ВИСНОВКИ

У ході виконання кваліфікаційної роботи було виконано детальне дослідження чотирьох мультимодальних моделей для автоматизованої генерації текстових описів зображень архітектурних пам'яток Києва. Також було самостійно сформовано два датасети та розроблено власний метод оцінки ефективності моделей за допомоги екстракції визначених атрибутів. Після результатів із практичними експериментами, було обрано Qwen3-VL-2B для тонкого налаштування шляхом низькорангової адаптації, щоб краще адаптувати модель для вузькоспеціалізованої задачі документування культурної спадщини. В результаті донавчання моделі методом LoRA було досягнуто підвищення розпізнавання атрибутів кількості поверхів та наближеного року будівництва, і також значення macro average.

Перспективами подальшої роботи є розширення тренувального датасету більш якісними метаданими для референсів та інтеграція розробленого підходу з реєстровими системами ДОКС для автоматизованого поповнення облікової документації пам'яток.

ВИКОРИСТАННЯ ІНСТРУМЕНТІВ ШТУЧНОГО ІНТЕЛЕКТУ

У процесі написання текстової частини було використано Claude (Anthropic) для структурування розділу 3 із теоретичними основами, вдосконалення та покращення академічного формулювання самостійно написаного тексту авторкою.

Фрагменти програмного коду були частково згенеровані також Claude, зокрема для очистки і фільтрації метаданих: парсинг років будівництва об’єктів з використанням регулярних виразів. Усі згенеровані зразки коду та змістовного матеріалу були ретельно переглянуті та доопрацьовані авторкою.

ДЖЕРЕЛА

1. Kashtan NEWS. (2025, 3 січня). Пам'ятки, які ми втратили: найгучніші архітектурні скандали Києва 2024 року. https://www.kashtan.news/pam-iatky-iaki-my-vtratyly-nayhuchnishi-arkhitekturni-skandaly-kyieva-2024-roku/
2. Аргумент. (б. д.). Феномен знищення історичного Києва: що це і як його зупинити. https://argumentua.com/stati/fenomen-znishchennya-istorichnogo-kieva-shcho-tse-i-yak-iogo-zupiniti
3. Renovation Map. (б. д.). Карта пам'яток Києва [Інтерактивна карта]. https://renovationmap.org/map/?region=kyiv
4. McCormick, C. (2024, 23 квітня). Google Colab GPUs: features and pricing. https://mccormickml.com/2024/04/23/colab-gpus-features-and-pricing/
5. PyTorch Team. (б. д.). PyTorch: An open-source machine learning framework [Програмна бібліотека]. https://pytorch.org/
6. Hugging Face. (б. д.). Transformers documentation. https://huggingface.co/docs/transformers/index
7. ScienceDirect. (б. д.). Image captioning [Довідковий ресурс]. https://www.sciencedirect.com/topics/computer-science/image-captioning
8. Rahman, M. M., Uzzaman, A., Sami, S. I., Khatun, F., & Bhuiyan, M. A. (2024). A comprehensive construction of deep neural network-based encoder–decoder framework for automatic image captioning systems. IET Image Processing, 18(14), 4778-4798. https://doi.org/10.1049/ipr2.13287
9. Singh, H., Sharma, A., & Pant, M. (2024). Pixels to prose: Understanding the art of image captioning. arXiv:2408.15714. https://arxiv.org/abs/2408.15714

10. Rahman, M. M., Uzzaman, A., Sami, S. I., Khatun, F., & Bhuiyan, M. A. (2024). A comprehensive construction of deep neural network-based encoder–decoder framework for automatic image captioning systems. IET Image Processing, 18(14), 4778–4798. https://doi.org/10.1049/ipr2.13287
11. Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. arXiv:2103.00020. https://arxiv.org/abs/2103.00020
12. OpenAI. (2021). CLIP: Connecting text and images [Програмний репозиторій]. GitHub. https://github.com/openai/CLIP
13. Li, J., Li, D., Xiong, C., & Hoi, S. (2022). BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. arXiv:2201.12086. https://arxiv.org/abs/2201.12086
14. Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2015). Microsoft COCO: Common objects in context. arXiv:1405.0312. https://arxiv.org/abs/1405.0312
15. Hugging Face. (2022). BLIP - Transformers documentation. https://huggingface.co/docs/transformers/model_doc/blip
16. Qwen Team. (2025). Qwen3 technical report. arXiv:2505.09388. https://arxiv.org/abs/2505.09388
17. Hugging Face. (2025). Qwen/Qwen3-VL-2B-Instruct-FP8 - Model card. https://huggingface.co/Qwen/Qwen3-VL-2B-Instruct-FP8
18. Liu, H., Li, C., Wu, Q., & Lee, Y. J. (2023a). Visual instruction tuning. Advances in Neural Information Processing Systems (NeurIPS 2023). arXiv:2304.08485. https://arxiv.org/abs/2304.08485
19. Liu, H., Li, C., Li, Y., & Lee, Y. J. (2023b). Improved baselines with visual instruction tuning. Proceedings of the IEEE/CVF Conference on

Computer Vision and Pattern Recognition (CVPR 2024). arXiv:2310.03744. https://arxiv.org/abs/2310.03744

20. Chen, Z., Wang, W., Gao, Z., et al. (2025). InternVL3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. arXiv:2508.18265. https://arxiv.org/abs/2508.18265
21. Hugging Face. (2025). OpenGVLab/InternVL3_5-4B - Model card. https://huggingface.co/OpenGVLab/InternVL3_5-4B
22. Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2022). LoRA: Low-rank adaptation of large language models. Proceedings of the International Conference on Learning Representations (ICLR 2022). arXiv:2106.09685. https://arxiv.org/abs/2106.09685
23. Aghajanyan, A., Zettlemoyer, L., & Gupta, S. (2020). Intrinsic dimensionality explains the effectiveness of language model fine-tuning. arXiv:2012.13255. https://arxiv.org/abs/2012.13255
24. Hugging Face. (б. д.). Fine-tuning LLMs [Навчальний ресурс]. https://huggingface.co/learn/llm-course/chapter11/4
25. Dettmers, T., Pagnoni, A., Holtzman, A., & Zettlemoyer, L. (2023). QLoRA: Efficient finetuning of quantized LLMs. Advances in Neural Information Processing Systems (NeurIPS 2023). arXiv:2305.14314. https://arxiv.org/abs/2305.14314
26. Liang, X., Xie, J., Zhao, T., Stouffs, R., & Biljecki, F. (2025). OpenFACADES: An open framework for architectural caption and attribute data enrichment via street view imagery. arXiv:2504.02866. https://arxiv.org/abs/2504.02866
27. Hugging Face. (б. д.). seshing/openfacades-dataset [Набір даних]. https://huggingface.co/datasets/seshing/openfacades-dataset
28. Selvam, A. (2024, 7 серпня). LLM evaluation metrics - BLEU, ROUGE and METEOR explained. Medium.

https://avinashselvam.medium.com/llm-evaluation-metrics-bleu-rogue-and-meteor-explained-a5d2b129e87f