

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Кваліфікаційна наукова  
праця на правах рукопису

ЗАРІЧКОВИЙ ОЛЕКСАНДР АНАТОЛІЙОВИЧ

УДК 004.42 + 004.93

**ДИСЕРТАЦІЯ**

МЕТОДИ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ РОЗМІТКИ ВІДЕОДАНИХ ДЛЯ  
ЗАДАЧ КОМП'ЮТЕРНОГО ЗОРУ

121 Інженерія програмного забезпечення

12 Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

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

\_\_\_\_\_ О.А. Зарічковий

Науковий керівник Стеценко Інна Вячеславівна, д.т.н., професор

КИЇВ – 2025

## АНОТАЦІЯ

*Зарічковий О.А.* Методи та програмне забезпечення розмітки відеоданих для задач комп'ютерного зору. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії з галузі знань 12 Інформаційні технології за спеціальністю 121 Інженерія програмного забезпечення. – Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, 2025.

Штучний інтелект є однією з найбільш поширених сфер розробок програмного забезпечення у сучасному світі інформаційних технологій. За останню декаду було досягнуто значного прогресу для задач комп'ютерного зору, зокрема, у детекції об'єктів, завдяки розвитку методів глибокого навчання та зростанню обчислювальних можливостей.

Для успішної розробки та впровадження програмних засобів на основі ШІ необхідно збирати та розмічати великі об'єми даних, що вимагає значних людських ресурсів та часу. Неякісна розмітка даних може призвести до неточних та помилкових результатів методів ШІ, що в свою чергу стає причиною помилок у результатах обчислень програмного забезпечення. Наявні інструменти розмітки даних не завжди відповідають потребам розробників програмного забезпечення з використанням ШІ, особливо в контексті роботи з великими об'ємами відеоданих, що збільшує трудовитрати процесу створення якісних наборів даних.

Наявність зазначених проблем визначає актуальне наукове завдання підвищення ефективності процесу розмітки відеоданих для задач комп'ютерного зору, яке вирішується у цій дисертаційній роботі.

**Метою дисертаційної роботи** є підвищення швидкості розмітки відеоданих у контексті завдань детекції об'єктів за рахунок вдосконалення методів та програмного забезпечення, призначеного для розмітки відеоданих.

Для досягнення мети в роботі досліджено методи навчання нейронних мереж, що підвищують точність детекції об'єктів без змін моделей чи збільшення їх параметрів, та підходи до зменшення обсягу кадрів, які обробляються в задачах комп'ютерного зору. Проведено аналіз візуально-мовних моделей для підвищення точності програмних рішень, а також сучасних інструментів і процесів розмітки зображень та відео з метою вдосконалення їх архітектури та пришвидшення обчислень. Розроблено дуальну архітектуру автоматизованої розмітки даних та програмне забезпечення, що її реалізує. Проведено експериментальне дослідження, яке доводить ефективність прийнятих рішень.

Одним із головних викликів на шляху досягнення мети є забезпечення високої якості автоматизованої розмітки даних, наслідком якої є зменшення відсотку помилок, які необхідно виправити розмітчиком (людиною). Для розв'язання цього завдання застосовані різні техніки та методи, включаючи вперше запропоновану дуальну архітектуру, метод пріоритезації даних, ітеративний метод вибору ключових кадрів та мультимодальні нейронні мережі.

Основним результатом роботи є створення дуальної архітектури програмного забезпечення для автоматизації розмітки даних та імплементація методів автоматичної розмітки відеоданих, які забезпечують високу точність розмітки та скорочення часу, необхідного для дорозмітки (уточнення розмітки, які виконуються розмітчиком після автоматизованої розмітки). Розроблені методи перевірені на реальних задачах з метою демонстрації їхньої ефективності та переваг.

**У першому розділі** розглянуто основні аспекти детекції об'єктів, розмітки даних для задач комп'ютерного зору та програмні засоби для розмітки відеоданих. Зроблено огляд різних методів детекції об'єктів, у тому числі відомі методи, такі як R-CNN, Fast R-CNN, YOLO та інші. Описано процес розмітки даних та її значення для ефективного вирішення завдань машинного навчання. Наведено особливості розмітки даних для задач детекції об'єктів та постановлено задачу дослідження.

**У другому розділі** представлено новий метод навчання моделей машинного навчання, який використовує пріоритезацією складних екземплярів даних для навчання нейронних мереж, що збільшує точність детекції об'єктів на відео за рахунок підвищення якості набору даних. Особливістю запропонованого методу є те, що він не потребує попередньої розмітки відео для вибору складних екземплярів.

**У третьому розділі** запропоновано новий ітеративний метод вибору ключових кадрів на довгих відео для узагальнення змісту відео. Запропонований метод зменшує кількість нерелевантних кадрів, які визначені на фазі попереднього відбору, шляхом ітеративного застосування моделі машинного навчання до відібраних кадрів з подальшою фільтрацією нерелевантних кадрів та сегментів.

**У четвертому розділі** представлено метод агрегації знань Attr4Vis, який спрямований на покращення використання знань, закодованих у великих лінгвістичних моделях (LLM) та мовних модальностях візуально-мовних моделей (VLM), для підвищення точності розпізнавання складних сцен на відео за рахунок генерації додаткових атрибутів по відео. Новизна підходу полягає у генерації атрибутів на невеликих сегментах відео, що дозволяє ефективно кодувати зміни атрибутів з часом та підвищує точність розпізнавання подій. Також було запропоновано новий алгоритм розширення лексикону, призначений для збільшення спектра атрибутів, пов'язаних з відеоданими, що підвищує точність опису подій текстовою моделлю.

**У п'ятому розділі** проведено огляд інструментів розмітки зображень, визначено критерії для їх оцінки та наведено детальний огляд конкретних інструментів, таких як V7, LabelBox, Keylabs, LabelImg, LabelMe, Label Studio та Computer Vision Annotation Tool (CVAT). Розглянуто архітектуру інструменту розмітки CVAT, підходи автоматизації процесу розмітки даних та визначено вимоги до програмного забезпечення. Запропоновано дуальну архітектуру програмного забезпечення для автоматизації розмітки відеоданих, описано ключові компоненти системи та їх



взаємодію. Наведено опис архітектури та основних її компонентів. Розроблено програмне забезпечення на базі запропонованої архітектури та методів.

У шостому розділі наведено опис набору даних для проведення експериментального дослідження, визначено задачі розмітки даних, детально описано організацію процесу розмітки, інших умов проведення експерименту та наведені результати експериментального дослідження. Подано результати оцінювання швидкості та якості виконання розмітки відеоданих без автоматизації та з використанням різних підходів, включаючи нульове та активне навчання, а також підхід з використанням запропонованої дуальної архітектури. Виконано порівняння різних підходів до автоматизації розмітки відеоданих. Зроблені висновки щодо досягнення поставленої мети дисертаційної роботи.

У дисертаційній роботі отримано низку **нових наукових результатів**:

- **вперше** запропоновано дуальну архітектуру програмного забезпечення для автоматизованої розмітки даних, яка, за рахунок методу адаптивно-агрегованого навчання нейромережі, забезпечує пришвидшення процесу розмітки та, на відміну від існуючих аналогів, дає змогу ефективного застосування нульового та активного навчання нейромережі для розмітки даних та більш гнучкого використання програмного забезпечення для різноманітних задач комп'ютерного зору;
- **вперше** запропоновано метод пріоритезації складних зразків для навчання нейронної мережі, який, за рахунок відбору найскладніших зразків для навчання, підвищує якість набору даних без проведення попередньої розмітки відео, внаслідок чого збільшується точність детекції об'єктів на відео, та, на відміну від існуючих підходів, базується виключно на автоматично згенерованій репрезентації даних;
- **вперше** запропоновано ітеративний метод вибору ключових кадрів на довгих відео, що дає змогу визначати ключові кадри та сегменти відео з поступовим

підвищенням точності, та, на відміну від існуючих методів, враховувати динамічно зміни контенту відео для вибору ключових кадрів, підвищуючи точність сегментації та зменшуючи обсяг відеоданих для обробки;

- **вперше** запропоновано метод агрегації знань між текстовою та візуальною частинами у візуально-мовній моделі (VLM) для обробки складних мультимодальних взаємодій, що забезпечує більш високу точність розпізнавання складних сцен на відео та їх опису у порівнянні з існуючими аналогами.

Основні результати дисертаційної роботи опубліковано у 6 публікаціях, з яких 4 статті в періодичних виданнях, що проіндексовано у базі даних Scopus, 1 публікація опублікована у фаховому виданні, включених до переліку наукових фахових видань України з присвоєнням категорії «Б», та 1 публікація у матеріалах міжнародної наукової конференції.

**Ключові слова:** інженерія програмного забезпечення; машинне навчання; нейронна мережа; згорткова нейронна мережа; машинний зір; розпізнавання; зображення; штучний інтелект; тренування моделі; інтелектуальна інформаційна система.

## ABSTRACT

Zarichkovyi O. Algorithmic software for video data annotation for computer vision tasks. – Qualifying scientific work, the manuscript.

PhD thesis in the field of knowledge 12 Information technologies in a specialty 121 Software engineering. – National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, 2025.

Artificial intelligence is one of the most prominent fields of software development in the modern world of information technologies. Significant progress has been achieved in computer vision tasks, particularly object detection, over the past decade due to the advancements in deep learning methods and the growth of computational capabilities.

For the successful development and implementation of AI-based software tools, it is necessary to collect and annotate large volumes of data, which requires considerable human resources and time. Low-quality data annotation can lead to inaccurate and erroneous AI methods, consequently resulting in errors in software computations. Current data annotation tools do not always meet the needs of software developers working with AI, especially in the context of large-scale video data annotation, which increases the labor intensity of creating high-quality datasets.

The outlined problems establish a pressing scientific task of improving the efficiency of video data annotation processes for computer vision tasks, which is addressed in this dissertation.

**The aim of the dissertation** is to enhance the speed of video data annotation in object detection tasks by improving the methods and software tools designed for video data annotation.

To achieve this aim, the study investigates neural network training methods that improve object detection accuracy without modifying models or increasing number of parameters, as well as approaches to reduce the number of frames processed by computer vision methods.

Conducted study on visual-language models to improve accuracy. A dual-architecture system for automated data annotation and its supporting software have been developed. Experimental research demonstrates the effectiveness of the proposed solutions.

One of the main challenges in achieving the aim is ensuring high-quality automated data annotation, which minimizes the percentage of errors requiring manual correction. To address this, various techniques and methods were employed, including the novel dual-architecture approach, a data prioritization method, an iterative keyframe selection method, and multimodal neural networks.

The main result of this work is the creation of a dual-architecture software system for data annotation automation and the implementation of automatic video annotation methods. These methods ensure high annotation accuracy and reduce the time needed for post-annotation refinement by annotators after automated annotation. The developed methods were tested on real-world tasks to demonstrate their efficiency and advantages.

**The first section** present the core aspects of object detection, data annotation for computer vision tasks, and video data annotation tools. A review of various object detection methods, including R-CNN, Fast R-CNN, YOLO, among others, is provided. The importance of data annotation in solving machine learning problems effectively is outlined, along with a description of data annotation for object detection tasks and the research problem.

**The second section** introduces a new method for training machine learning models. This method employs the prioritization of challenging data instances for training neural networks, enhancing object detection accuracy in videos by improving the quality of the dataset. A distinctive feature of the proposed approach is that it does not require prior annotation of videos to select difficult instances.

**The third section** proposes a novel iterative method for selecting keyframes in lengthy videos to summarize video content. The proposed method reduces the number of irrelevant frames identified during the initial selection phase by iteratively applying a machine learning model to the selected frames, followed by filtering out irrelevant frames and segments.

**The fourth section** presents the Attr4Vis knowledge aggregation method. This method aims to improve the utilization of knowledge encoded in large language models (LLMs) and the linguistic modalities of visual-language models (VLMs) to enhance the accuracy of recognizing complex scenes in videos by generating additional attributes based on the video content. The novelty of the approach lies in generating attributes for small video segments, enabling efficient encoding of attribute changes over time and improving event recognition accuracy. Additionally, a new algorithm for lexicon expansion is proposed to increase the range of attributes associated with video data, thereby improving the accuracy of event descriptions generated by text models.

**The fifth section** reviews image annotation tools, identifies criteria for their evaluation, and provides a detailed analysis of specific tools such as V7, LabelBox, Keylabs, LabelImg, LabelMe, Label Studio, and the Computer Vision Annotation Tool (CVAT). The architecture of the CVAT annotation tool is explored, including data annotation process automation approaches and software requirements. A dual-architecture system for video data annotation automation is proposed, detailing its key system components and interactions. Software based on this architecture and methods was developed.

**The sixth section** describes the dataset used for the experimental study, along with the defined data annotation tasks, a detailed explanation of the annotation process organization, other experimental conditions, and the experimental study results. The evaluation outcomes for the speed and quality of video data annotation are presented, comparing manual annotation with different approaches, including zero-shot and active learning, as well as the proposed dual architecture approach. A comparison of various approaches to video data annotation automation is conducted. Conclusions regarding the achievement of the dissertation's objectives are drawn.

In the dissertation, a number of **new scientific results** were obtained:

**For the first time**, a dual software architecture for automated data annotation has been proposed. Utilizing the method of adaptively-aggregated neural network training, this architecture accelerates the annotation process and, unlike existing counterparts, enables the effective application of zero-shot and active neural network learning for data annotation. It also allows for more flexible software utilization across various computer vision tasks.

**For the first time**, a novel method for prioritizing difficult samples for neural network training is introduced, improving dataset quality without prior video annotation and enhancing object detection accuracy. Unlike existing approaches, this approach relies solely on automatically generated data representations.

**For the first time**, an iterative method for selecting keyframes in long videos is proposed, enabling accurate identification of keyframes and segments while accounting for dynamic content changes, improving segmentation accuracy and reducing video data for processing.

**For the first time**, a method for aggregating knowledge between the textual and visual components in a visual-language model (VLM) has been proposed to model complex multimodal interactions, providing higher accuracy in recognizing complex scenes in videos and their descriptions compared to existing counterparts..

The main results of the dissertation were published in 6 scientific papers, in particular, in 4 scientific articles, which is indexed in the Scopus database, 1 article published in a scientific journal included in the list of scientific professional editions of Ukraine (category «B»), as well as 1 publication in materials of scientific and technical conferences.

**Keywords:** software engineering; machine learning; neural network; convolutional neural network; computer vision; recognition; image; artificial intelligence; model training; intelligent information system.

Список публікацій здобувача / List of publications of the applicant:

**Статті у фахових виданнях, включених до переліку наукових фахових видань України з присвоєнням категорії «Б» / the articles published in scientific journals included in the list of scientific journals of Ukraine in *category «Б»*:**

1. Зарічковий О.А. Дуальна архітектура програмного забезпечення для автоматизації розмітки даних для задач комп'ютерного зору. Міжвідомчий науково-технічний журнал «Адаптивні системи автоматичного управління». 2024. № 45 (2024). С. 109-118. DOI 10.20535/1560-8956.45.2024.313096.

**Статті у періодичних наукових виданнях, проіндексованих у базах даних Web of Science Core Collection та/або Scopus / the publication in scientific periodicals indexed in the Web of Science Core Collection and/or Scopus databases:**

2. Zarichkovyi, A., Stetsenko, I.V. (2024) 'Attr4Vis: Revisiting importance of attribute classification in Vision-Language Models for Video Recognition', *International Journal of Computing*, 23 (1), pp. 94-100. DOI 10.47839/ijc.23.1.3440.
3. Zarichkovyi, A., Stetsenko, I.V. (2023) 'Boundary Refinement via Zoom-In Algorithm for Keyshot Video Summarization of Long Sequences', *Lecture Notes on Data Engineering and Communications Technologies*, 180, pp. 344-359. DOI 10.1007/978-3-031-36115-9\_32.
4. Zarichkovyi, A., Stetsenko, I.V. (2023) 'Hard Samples Make Difference: An Improved Training Procedure for Video Action Recognition Tasks', *Lecture Notes in Networks and Systems*, 544, pp. 508-519. Springer, Cham. DOI 10.1007/978-3-031-16075-2\_36.
5. Oleksandr Zarichkovyi and Iryna Mukha. (2021) 'Approximate Training of Object Detection on Large-Scale Datasets', *Lecture Notes on Data Engineering and*

*Communications Technologies*, 83, pp. 389-400. DOI 10.1007/978-3-030-80472-5\_32.

**Публікація у матеріалах наукової конференції / the publication in the proceedings of the scientific conference:**

6. Zarichkovyi, A., Stetsenko, I.V. Improving cross-modal knowledge exploration of vision language models. Інженерія програмного забезпечення і передові інформаційні технології (Soft Tech-2024): матеріали VI Міжнародної науково-практичної конференції молодих вчених та студентів, 21-23 травня 2024 року, м. Київ, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», ФІОТ. С. 58-61.



## ЗМІСТ

Перелік умовних позначень.....	17
ВСТУП.....	18
1 Загальні положення для задачі детекції об'єктів на відео .....	27
1.1 Огляд сучасного стану методів комп'ютерного зору .....	27
1.2 Огляд методів для вирішення задачі детекції об'єктів .....	29
1.3 Порівняння методів детекції об'єктів.....	39
1.4 Фундаментальні візуально-мовні моделі машинного навчання .....	42
1.5 Детекція об'єктів на відео.....	46
1.6 Опис процесу розмітки даних для задач машинного навчання .....	49
1.7 Підходи до організації розмітки даних.....	52
1.8 Обґрунтування необхідності прискорення розмітки .....	56
1.9 Особливості розмітки даних для задач детекції об'єктів .....	58
1.10 Постановка задачі.....	60
1.11 Висновки до розділу .....	63
2 Метод пріоритезації складних зразків для навчання нейронних мереж.....	65
2.1 Методи глибокого навчання для просторово-часового моделювання .....	65
2.2 Майнінг складних негативних зразків.....	69
2.3 Метод пріоритезації складних зразків для навчання нейронних мереж ..	71
2.4 Експериментальні дослідження запропонованого підходу до вибору складних зразків.....	74
2.5 Висновки до розділу .....	78
3 Ітеративний метод вибору ключових кадрів на довгих відео .....	80
3.1 Постановка задачі вибору ключових кадрів для узагальнення відео .....	80
3.2 Методи комп'ютерного зору для узагальнення змісту відео .....	83

3.3	Запропонований ітеративний метод до вибору ключових кадрів для задачі узагальнення змісту відео .....	89
3.4	Результати експериментального дослідження ефективності запропонованого методу відбору ключових кадрів для задачі узагальнення змісту відео .....	93
3.5	Дослідження впливу ключових компонентів на якість запропонованого ітеративного методу вибору ключових кадрів для узагальнення змісту відео .....	97
3.6	Порівняння запропонованого ітеративного методу до вибору ключових кадрів з методом D-KTS при високій частоті кадрів для задач узагальнення змісту відео .....	98
3.7	Висновки до розділу .....	101
4	Мультимодальні моделі нейронних мереж для задач розпізнавання на відео... ..	103
4.1	Використання мультимодальних моделей для задач розпізнавання на відео .....	103
4.2	Запропонований підхід до використання мультимодальних моделей для задач розпізнавання на відео .....	107
4.3	Алгоритм збагачення лексикону атрибутів .....	109
4.4	Дослідження впливу запропонованого підходу для навчання мультимодальних моделей для задач розпізнавання на відео .....	112
4.5	Висновки до розділу .....	116
5	Архітектура та компоненти програмного забезпечення для автоматизації процесу розмітки відеоданих.....	118
5.1	Інструменти розмітки зображень .....	118
5.2	Порівняльний аналіз інструментів розмітки даних.....	129
5.3	Архітектура інструментів розмітки даних .....	132

5.4	Підходи до автоматизації розмітки даних в наявних інструментах розмітки даних для задач комп'ютерного зору .....	135
5.5	Вимоги до програмного засобу для розмітки з урахуванням автоматизації .....	138
5.6	Запропонована дуальна архітектура програмного забезпечення для автоматизації розмітки відеоданих .....	145
5.7	Опис архітектури розробленого програмного забезпечення.....	150
5.8	Висновки до розділу .....	152
6	Аналіз ефективності дуальної архітектури .....	154
6.1	Процес розмітки даних.....	154
6.2	Організація розмітки даних .....	159
6.3	Розмітка даних для задачі детекції об'єктів за допомогою CVAT .....	161
6.4	Результати експериментального дослідження.....	164
6.5	Порівняння різних підходів до автоматизації розмітки даних.....	172
6.6	Висновки до розділу .....	175
	ВИСНОВКИ .....	177
	СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ .....	179
	ДОДАТОК А ЧАСТИНА ПРОГРАМНОГО КОДУ РОЗРОБЛЕНОГО ВЕБЗАСТОСУНКУ .....	199
	ДОДАТОК Б. ГРАФІЧНЕ ПРЕДСТАВЛЕННЯ АРХІТЕКТУРИ РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ .....	253
	Додаток В. Вихідні дані для графіка без автоматизації процесу розмітки відеоданих .....	254
	Додаток Г. Вихідні дані для графіка з автоматизацією процесу розмітки відеоданих через підходи з нульовим навчанням .....	255

Додаток Д. Вихідні дані для графіка з автоматизацією процесу розмітки відеоданих через підходи з Активного навчання .....	256
Додаток Е. Вихідні дані для графіка з автоматизацією процесу розмітки відеоданих через запропонований підхід з дуальною архітектурою.....	257
Додаток Є. Діаграми прецедентів для ролей «Розітчик» та «Адміністратор» Інструменту розмітки .....	258
Додаток Ж. Перелік публікацій .....	260

## ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ШІ – штучний інтелект

CAGR – Compound annual growth rate – сукупні середньорічні темпи росту

CNN – Convolutional neural network – згорткова нейронна мережа

GRU – Gated recurrent units – контрольований рекурентний вузол

LLM – Large language models – великі мовні моделі

LSTM – Long-short term memory – довгострокова-короткочасна пам'ять

NMS – Non-maximum suppression – алгоритм немаксимального пригнічення

R-CNN – Region-based convolutional neural network – згорткова нейронна мережа для детекції по кандидатах

RPN – Region proposal network – нейронна мережа для генерації локацій кандидатів для детекції методами R-CNN

VLM – Vision-language models – візуально-текстові моделі

Zero-shot learning algorithm – алгоритм нульового навчання, що виконує прогнозування на даних та категоріях, які не були присутніми в навчальній вибірці даних.

## ВСТУП

**Актуальність теми.** Штучний інтелект є однією з найбільш поширених сфер розробок програмного забезпечення у сучасному світі інформаційних технологій. За останні десять років суспільство стало свідком значного прогресу в завданнях комп'ютерного зору, що відобразилося у підвищенні точності класифікації зображень на наборі даних ImageNet ILSVRC'2012 [1] або інших публічних наборах даних [2]. Цей значний прогрес був досягнутий завдяки швидкому розвитку методів глибокого навчання, експоненціальному зростанню обчислювальних можливостей комп'ютерів та більш широкому доступу дослідників до великих наборів даних [3; 4; 5].

Згідно зі звітом Gartner [6], світовий дохід від програмного забезпечення штучного інтелекту (ШІ) у 2022 році становив 62,5 мільярда доларів США, що на 21,3% перевищив показники 2021 року, а обсяг ринку штучного інтелекту склав 428 мільярдів доларів США у 2022 році. За прогнозами Gartner ринок програмного забезпечення з використанням ШІ зросте до 2025 мільярдів доларів США до 2030 року, демонструючи щорічний усереднений ріст (CAGR) на 21,6% [6].

Це створює великий попит на розробку прикладного програмного забезпечення з використанням штучного інтелекту [7]. Щоб задовольнити цей попит, лідери індустрії створили сервіси для розробки та впровадження програмного забезпечення на базі ШІ, такі як Amazon AWS SageMaker [8], Microsoft Azure AI Platform [9], IBM Watson [10], Google Cloud AI Platform [11] та безліч інших. Ці сервіси допомагають прискорити створення моделей штучного інтелекту та їх впровадження у порівнянні з розробкою програмного забезпечення ШІ з чистого аркуша [12; 13].

Однак, головним недоліком існуючих сервісів є те, що вони концентрують увагу лише на процесі створення та використанні моделей штучного інтелекту. Проте, за останні кілька років індустрія ШІ значно трансформувалася, а об'єми даних, що генеруються, особливо на площадках потокової передачі відео, швидко зростають. З цієї

причини швидкість розробки та впровадження інструментів для розмітки даних стає все більш важливішою [14; 15]. При цьому дослідження в даній галузі показують, що більше 80% часу в проєктах з розробки програмних продуктів ШІ витрачається саме на підготовку даних до обробки та розмітку даних [16], що надає можливість подальшого їх використання для навчання ШІ, включаючи нейронні та глибокі нейронні мережі, які розробляються з використанням бібліотек машинного навчання Tensorflow [17], PyTorch [18], Scikit-learn [19] тощо.

Згідно з дослідженнями [16; 20; 21], найдовшим етапом в роботі з даними є їх розмітка для подальшого використання в процесі навчання методів машинного навчання. Деякі з існуючих сервісів для задач штучного інтелекту пропонують вбудовані можливості для розмітки даних, які виконуються класичними методами без використання жодної автоматизації або з обмеженою підтримкою автоматизації [22; 23], що призводить до неефективного (повільного) процесу розмітки даних. Особливо гостро ця проблема постає для задач, які потребують великих об'ємів даних для досягнення задовільного рівня якості – автономне керування автомобілем (автопілоти), верифікація пасажирів в аеропортах (системи розпізнавання людей), покупок в магазині (магазини без кас), діагностики онкозахворювань грудної клітини та багатьох інших [24].

Проте, індустрія по розмітці даних стикається з різними проблемами, які можуть впливати на якість та швидкість роботи. Одна з головних проблем – це складність та трудомісткість процесу розмітки даних [22; 24]. Розмітка даних вимагає великої кількості людських ресурсів та часу, що за необхідності забезпечення високої точності розмітки даних може призвести до затримки здачі проєкту та збільшення його вартості [20]. Неточна або некоректна розмітка даних може призвести до незадовільної якості моделей та помилок в їх роботі, що може вплинути на програмне забезпечення та на рішення, яке приймається на основі результатів його роботи [20; 21].

Ще одна проблема з розміткою даних полягає в тому, що індустрія швидко розвивається, тому існує потреба постійно оновлювати та вдосконалювати існуючі набори. Це може бути досить складно та вимагати великої кількості ресурсів [22; 24].

Усі вищезгадані проблеми можуть вплинути на якість та швидкість розмітки даних, що створює потребу в розробці нових високоефективних та вдосконалені наявних інструментів з обробки даних в індустрії ШІ та впровадженні їх у широке коло користувачів [25; 26].

Виявлені такі технічні недоліки наявних інструментів розмітки зображень та відеоданих, які обмежують їх ефективність:

- використання лише одного методу навчання, активного або нульового, звужує спектр задач, які можуть бути оброблені цими інструментами, що знижує їх універсальність;
- брак методів для прискорення моделей активного навчання подовжує цикл навчання та затримує досягнення ними достатньо високої точності;
- рівномірний вибір кадрів у відео не враховує динамічні зміни сцени, що знижує точність автоматизованої розмітки;
- відсутність оптимальних методів до інтеграції VLM та LLM моделей у процес розмітку даних знижує точність кінцевих методів.

Дані недоліки були враховані та виправлені за рахунок використання дуальної архітектури програмного забезпечення розмітки даних, використання методу пріоритезації розмітки даних, використанням ітеративного методу до вибору ключових кадрів, та ефективної агрегації знань між знань між текстовою та візуальною частинами у візуально-мовних моделях.

Експериментальні дослідження показали, що використання вищезгаданих методів та підходів значно прискорює процес розмітки даних, що веде до зменшення вартості розмітки даних та робить застосунки на базі ШІ більш доступними ширшому колу розробників та користувачів.



Таким чином, наявність зазначених актуальних проблем визначає актуальне наукове завдання підвищення ефективності процесу розмітки відеоданих для задач комп'ютерного зору [20; 24].

**Зв'язок роботи з науковими програмами, планами, темами.** Дисертаційна робота тісно пов'язана з науковими розробками, що здійснюються на кафедрі інформатики та програмної інженерії Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Тема дисертації відповідає науковому напрямку «Інформаційні та комунікаційні технології. Глибоке навчання, великі дані (big data), нейроподібні мережі» переліку пріоритетних тематичних напрямів наукових досліджень і науково-технічних розробок на період до 31 січня року, наступного після припинення або скасування воєнного стану в Україні, затвердженого постановою Кабінету Міністрів України №476 від 30.04.2024 р. У дисертації запропоновані методи та моделі, які спрямовані на вирішення завдань, поставлених у Концепції розвитку штучного інтелекту в Україні схваленою розпорядженням № 1556-р Кабінетом Міністрів України від 2 грудня 2020 р., а саме впровадження інформаційних технологій, частиною яких є технології штучного інтелекту.

Результати наукового дослідження частково отримані здобувачем при виконанні науково-дослідної роботи «Методи та технології високопродуктивних обчислень та обробки надвеликих масивів даних» (державний реєстраційний номер 0117U000924).

**Мета і завдання дослідження.** Метою дисертаційного дослідження є підвищення швидкості розмітки відеоданих у контексті завдань детекції об'єктів за рахунок вдосконалення методів та програмного забезпечення, призначеного для розмітки відеоданих.

Для досягнення мети в роботі досліджено методи навчання нейронних мереж, що підвищують точність детекції об'єктів без змін моделей чи збільшення їх параметрів, та підходи до зменшення обсягу кадрів, які обробляються в задачах комп'ютерного зору. Проведено аналіз візуально-мовних моделей для підвищення точності програмних

рішень, а також сучасних інструментів і процесів розмітки зображень та відео з метою вдосконалення їх архітектури та пришвидшення обчислень. Розроблено дуальну архітектуру автоматизованої розмітки даних та програмне забезпечення, що її реалізує. Проведено експериментальне дослідження, яке доводить ефективність прийнятих рішень.

Одним із головних викликів на шляху досягнення мети є забезпечення високої точності автоматизованої розмітки даних та зменшення кількості помилок методу. Для розв'язання цього завдання застосовані різні підходи та методи, включаючи вперше запропоновану дуальну архітектуру, механізм пріоритезації, ітеративний процес вибору ключових кадрів та ефективної агрегації знань між знань між текстовою та візуальною частинами у візуально-мовних моделях.

Основним результатом роботи є створення інноваційної дуальної архітектури програмного забезпечення для автоматизації розмітки даних та імплементація методів автоматичної розмітки відеоданих, які забезпечують високу точність розмітки та скорочення часу, необхідного для завершення процесу розмітки. Розроблені методи перевірені на реальних задачах аналізу відеоданих з метою оцінювання їхньої ефективності та переваг.

Відповідно до поставленої мети основними завданнями дослідження є:

- розробити метод до навчання нейронних мереж, що підвищує точність методів розпізнавання по відео;
- розробити метод до зменшення обсягу оброблених кадрів для методів комп'ютерного зору;
- розробити метод по використанню візуально-мовних моделей для підвищення точності задач, які вирішують програмні засоби;
- виявити недоліки сучасних методів автоматизації розмітки в програмних засобах для розмітки зображень та відеоданих;

- розробити дуальну архітектуру автоматизованої розмітки даних за допомогою різних методів комп'ютерного зору, що прискорить процес розмітки даних;
- виконати програмну реалізацію запропонованих методів та виконати експериментальне дослідження характеристик розробленого програмного забезпечення.

**Об'єкт дослідження** – процес розмітки відеоданих для задач комп'ютерного зору.

**Предмет дослідження** – методи та моделі автоматизованої розмітки відеоданих.

**Методи дослідження.** Дослідження базується на теоретичному матеріалі, що стосується програмного забезпечення для розмітки зображень та відеоданих, процесу розмітки відеоданих, аналізу технологій ШІ для обробки відеоданих. У процесі дослідження використовувалися методи обробки зображень та теорії сигналів, статистичного аналізу та машинного навчання.

**Наукова новизна одержаних результатів** полягає у наступному:

1. **Вперше** запропоновано дуальну архітектуру програмного забезпечення для автоматизованої розмітки даних, яка, за рахунок методу адаптивно-агрегованого навчання нейромережі, забезпечує пришвидшення процесу розмітки та, на відміну від існуючих аналогів, дає змогу ефективного застосування нульового та активного навчання нейромережі для розмітки даних та більш гнучкого використання програмного забезпечення для різноманітних задач комп'ютерного зору.
2. **Вперше** запропоновано метод пріоритезації складних зразків для навчання нейронної мережі, який, за рахунок відбору найскладніших зразків для навчання, підвищує якість набору даних без проведення попередньої розмітки відео, внаслідок чого збільшується точність детекції об'єктів на відео, та, на відміну від існуючих підходів, базується виключно на автоматично згенерованій репрезентації даних.

3. **Вперше** запропоновано ітеративний метод вибору ключових кадрів на довгих відео, що дає змогу визначати ключові кадри та сегменти відео з поступовим підвищенням точності, та, на відміну від існуючих методів, враховувати динамічно зміни контенту відео для вибору ключових кадрів, підвищуючи точність сегментації та зменшуючи обсяг відеоданих для обробки.
4. **Вперше** запропоновано метод агрегації знань між текстовою та візуальною частинами у візуально-мовній моделі (VLM) для обробки складних мультимодальних взаємодій, що забезпечує більш високу точність розпізнавання складних сцен на відео та їх опису у порівнянні з існуючими аналогами.

**Практичне значення одержаних результатів.** Розроблені методи та програмні засоби сприяють значному підвищенню точності та швидкості розмітки даних для задач детекції на відео, знижуючи необхідність у ручній праці та прискорюючи процес обробки на 125% відносно підходів без автоматизації процесу розмітки даних та на 25% відносно інших підходів до автоматизації без втрати точності. Покращено використання знань, закодованих у великих лінгвістичних моделях (LLM) та мовних модальностях візуально-мовних моделей (VLM), для ініціалізації візуальних моделей та генерації семантично повних атрибутів для опису того, що відбувається на відео, підвищуючи точність розпізнавання в середньому на 4% у порівнянні з методом VIKI. Покращено процес вибору ключових кадрів на відео, що з дозволяє зменшити об'єм відеоданих для подальшої обробки в середньому в 6 разів у порівнянні з методом D-KTS. Покращено процес навчання моделей з використанням методу пріоритезації, що підвищує точність роботи методу в середньому на 1,3% в порівнянні з використанням випадкової вибірки відео з того ж розподілу даних.

**Особистий внесок здобувача.** Усі основні результати дисертаційного дослідження, які представлені до захисту, одержані автором особисто. У публікаціях,

написаних у співавторстві, здобувачеві належать наступні результати: [2] - здобувачем представлено нову архітектуру під назвою Attr4Vis, призначену для полегшення мультимодального навчання VLM, що представлено у дисертації у вигляді методу активного навчання на відео; [3] - здобувачем запропоновано вдосконалений підхід з виділенням ключових кадрів та ітеративним уточненням границь ключових кадрів, що представлено у дисертації у вигляді ітеративного методу до вибору ключових кадрів на відео; [4] – здобувачем запропоновано новітній підхід до навчання нейронних мереж на відео, який базується на виборі складних зразків та подальшим навчанням на них; [5] - здобувачем запропоновано новітній підхід до навчання нейронних мереж на зображеннях, який базується на виборі складних зразків та подальшим навчанням на них; [6] - здобувачем експериментально доведено вплив точності генерації атрибутів на точність прогнозування моделей машинного навчання, що представлено у дисертації у вигляді алгоритму збагачення лексикону атрибутів.

**Апробація результатів дисертації.** Основні результати дисертаційного дослідження доповідалися та обговорювалися на міжнародних науково-практичних конференціях: The Fourth International Conference on Computer Science, Engineering and Education Applications, ICCSEEA2021 (Kyiv, Ukraine, 2021); Intelligent Systems Conference 2022, IntelliSys 2022 (Amsterdam, The Netherlands, 2022); The 3rd International Conference on Artificial Intelligence and Logistics Engineering, ICAILE2023 (Wuhan, China, 2023); Software Engineering and Advanced Information Technology, SoftTech 2024 (Kyiv, Ukraine, 2024).

**Публікації.** Основні результати дисертаційної роботи опубліковано у 6 публікаціях, з яких 4 статті в періодичних виданнях, що проіндексовано у базі даних Scopus, 1 публікація опублікована у фаховому виданні, включених до переліку наукових фахових видань України з присвоєнням категорії «Б», та 1 публікація у матеріалах міжнародної наукової конференції.

**Структура і обсяг роботи.** Дисертаційна робота складається зі вступу, шести розділів, загальних висновків, списку використаних джерел із 211 найменувань та 8 додатків. Загальний обсяг дисертації становить 261 сторінок, з яких 161 сторінок основного тексту та містить 63 рисунків, 18 таблиць, 9 формул.

# 1 ЗАГАЛЬНІ ПОЛОЖЕННЯ ДЛЯ ЗАДАЧІ ДЕТЕКЦІЇ ОБ'ЄКТІВ НА ВІДЕО

## 1.1 Огляд сучасного стану методів комп'ютерного зору

За останнє десятиріччя відбувся значний прогрес у різних сферах ШІ, машинного навчання, штучних нейронних мереж та комп'ютерного зору. Ідеальним прикладом визначного прогресу ШІ є проєкт "Asirra" від Microsoft Research [27], запущений у 2006 році ключовою метою якого було навчитися відрізняти ботів від людей в мережі Інтернет. Користувачам ставилось дуже просте завдання: відрізнити kota від собаки за фотографією. Фотографії бралися з фотобанку тварин з притулків, який постійно зростає та оновлювався. На час створення програмного забезпечення воно визнавалось майже ідеальним, оскільки для вирішення завдання необхідно було створити загальний ШІ, що вважалось NP-повною задачею. Найкращі підходи того часу досягали точності, близької до 60%, тоді як випадкове вгадування мало точність 50%, що робило ймовірність правильного прогнозування для декількох картинок малою ймовірною. Однак, менше чим за десятиліття, найкращі алгоритми на основі методів з використання глибоких штучних нейронних мереж змогли досягти якості близької до 100%, що дозволило правильно розпізнати послідовність з 10 зображень з ймовірністю більшою за 80%, а послідовність з 50 зображень – з ймовірністю більшою за 60%. Таким чином, кількість помилок в цій складній для комп'ютера задачі зменшилась в 40 разів і стала практично нульовою. Зауважимо, що відрізнення котів від собак є лише однією з безлічі важливіших та корисніших для людства задач комп'ютерного зору.

Хоча більшість задач, пов'язаних з розумінням та класифікацією зображень вже мають достатньо високу якість, а для деяких задач комп'ютери перевершили людей, наприклад, для задачі класифікації на наборі даних ImageNet, проте все ще залишаються невирішеними фундаментальні проблеми комп'ютерного зору, які потребують

дослідження. До таких проблем належить задача детекції об'єктів на зображенні та визначення точного їх положення на зображенні [28], що є складною задачею у нічим необмеженому середовищі через різноманіття об'єктів, їх розмірів, освітлення, можливості перекриття цільових об'єктів іншими об'єктами на сцені, різного кута зйомки, тощо. Методи детекції об'єктів можуть бути використано в різних програмних продуктах, таких як класифікація зображень [29; 30], аналіз поведінки людини [31], автономне керування [32], розпізнавання обличч [33] та інші.

Одним з найбільш часто використовуваних застосувань програмних продуктів з детекції об'єктів є відеоспостереження та безпека [33]. Завдяки детекції об'єктів, системи відеоспостереження можуть автоматично виявляти небажані події, такі як вторгнення в будинок, крадіжки, агресію, та надсилати повідомлення в реальному часі для аналізу. Детекція об'єктів на відео широко застосовується також для керування автономним автомобілем [34] та в робототехніці [35]. Наприклад, в автомобільній промисловості системи детекції об'єктів можуть виявляти пішоходів, велосипедистів, інші автомобілі, що перетинають дорогу, а також розпізнавати дорожні знаки та сигнали світлофорів. Це може допомогти зменшити кількість дорожньо-транспортних пригод та забезпечити безпечний рух на дорозі.

Для детекції об'єктів на зображенні використовуються різні методи, такі як метод каскадних класифікаторів [37], SSD [38], YOLO [39], Cascade R-CNN [40] та Mask R-CNN [41], підходи базовані на використанні механізму уваги, такі як DETR [42], DINO [43], CLIP [44] та багато інших, як показали високу якість на складних наборах даних таких як PASCAL VOC [45] та MS COCO [46]. Оскільки завдання детекції об'єктів є ключовим етапом у вищезгаданих програмних продуктах, то підвищення якості детекції об'єктів має великий потенціал впливу на поліпшення продуктивності та ефективності процесів, пов'язаних з розпізнаванням об'єктів. Проте, існують значні проблеми у створенні універсальної системи детекції об'єктів, що пов'язані з вищезгаданою проблемою різноманіття об'єктів, а отже існує потреба у створенні та розмітці великих



наборів даних. У науковій літературі дослідники приділили багато уваги вирішенню цієї проблеми. Наприклад, деякі дослідження [47-50] були проведені з метою вирішення проблеми детекції об'єктів. Нові підходи до розмітки даних [51], [52] допомагають зменшити вартість розмітки даних, але все ж процес розмітки даних для задачі детекції об'єктів залишається надзвичайно трудомістким та дорогим.

## **1.2 Огляд методів для вирішення задачі детекції об'єктів**

В області комп'ютерного зору у 2014 році з'явився метод детекції об'єктів на зображенні Region-based Convolutional Neural Network (R-CNN) [47]. Цей метод вирішував важке завдання знаходження місцеположення та класифікації об'єктів на зображеннях шляхом поєднання методу генерації кандидатів об'єктів з класифікатором на базі CNN. Метод R-CNN використовує алгоритм вибіркового пошуку Selective Search [53] для визначення границь потенційних об'єктів (кандидатів) на основі перцептивного групування пікселів за кольором, текстурою та іншими атрибутами низького рівня. Ці кандидати слугували фундаментом для подальшого аналізу. Кожен запропонований кандидат далі трансформувався та нормалізувався до єдиної розмірності, яка подавалася на вхід CNN, що була попередньо навчена на різноманітних наборах даних для класифікації зображень.

Оскільки CNN класифікатор не міг відрізнити будь-який об'єкт від фону, то автори запропонували використати підхід опорних векторів (SVM), завданням якого була бінарна класифікація кожного кандидата як об'єкт інтересу, який надалі класифікувався CNN, або як фон, який відкидався з аналізу. Також автори ввели регресійну модель, що намагалася уточнити координати запропонованих кандидатів, тим самим підвищуючи точність локалізації об'єктів.

На наступному кроці метод R-CNN використовував алгоритм немаксимального пригнічення (NMS) [54]. Цей алгоритм видаляє регіони, що сильно перетинаються, залишаючи лише ті, які мали максимальні оцінки впевненості. NMS необхідний для

усунення дублікатів детекції для одного об'єкту, що забезпечує однозначний результат роботи запропонованого методу. Основні кроки методу R-CNN зображено на рисунку 1.1 [47].

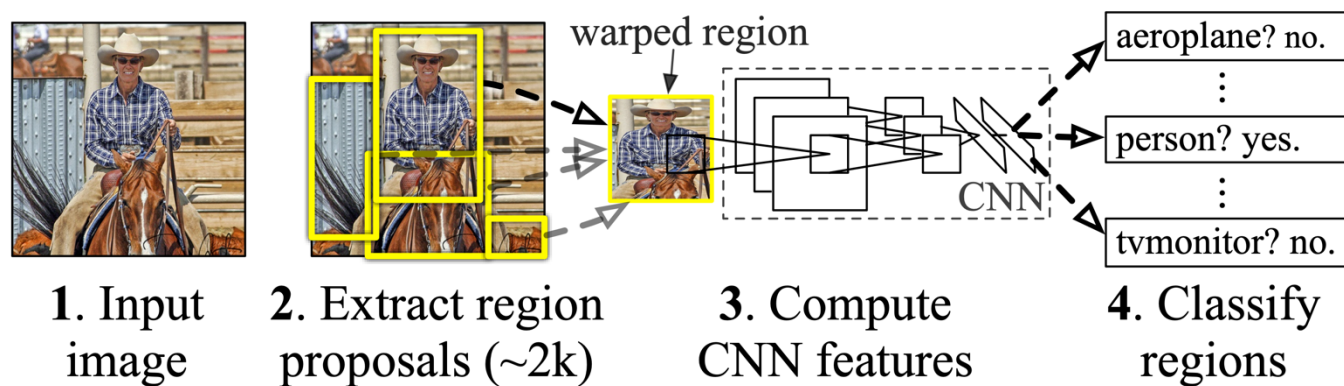


Рисунок 1.1 – Представлення основних кроків методу R-CNN [47]

Надалі метод R-CNN вдосконалювався через ітерації, відомі як Fast R-CNN [48] і Faster R-CNN [49], кожна з яких підвищує як точність методу детекції, так і його швидкодію.

Fast R-CNN [48] є модифікацією методу R-CNN та відрізняється від нього тим, що спочатку використовує згорткову нейронну мережу (CNN) для генерації мапи репрезентацій з вхідного зображення. Після цього, аналогічно до R-CNN, за допомогою алгоритму Selective Search визначаються координати кандидатів, які називають Region of Interest (ROI). Далі кожен кандидат проходить процедуру пулінгу репрезентації (ROI Pooling) з CNN, яка агрегує репрезентації для кожного кандидата у фіксований вектор. Після цього кожен кандидат проходить через повнозв'язну нейронну мережу для класифікації та регресії координат обмежувальної рамки. Вихідні дані класифікації надають розподіл ймовірностей за різними класами об'єктів, тоді як вихідні дані регресії уточнюють координати обмежувальних рамок. Ці локації потім фільтруються на основі алгоритму NMS для усунення дублікатів. Схематичне представлення кроків методу Fast R-CNN зображено на рисунку 1.2.

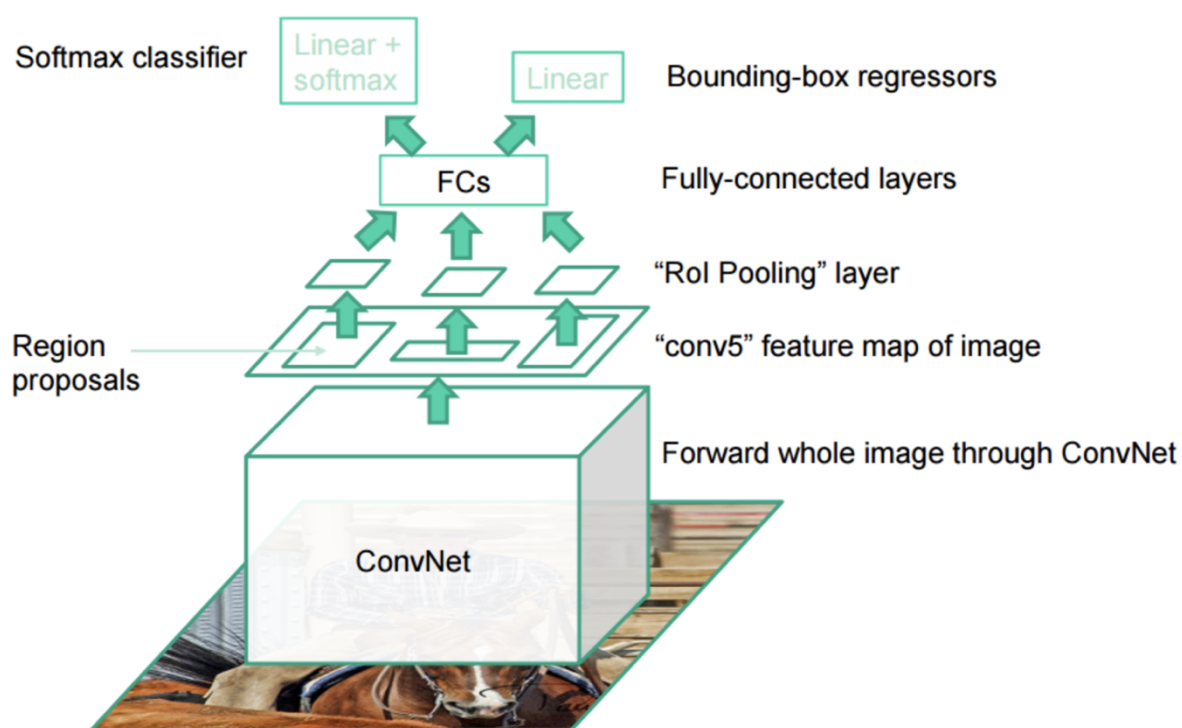


Рисунок 1.2 – Представлення основних кроків методу Fast R-CNN [48]

Основна перевага Fast R-CNN у порівнянні з оригінальним R-CNN полягає в тому, що він працює швидше завдяки об'єднанню процесу генерації мапи репрезентацій для кандидатів в одну модель, що допомагає уникнути множинного обрахунку репрезентацій для кандидатів, що пересікаються, а отже зменшити обчислювальні вимоги методу [48].

Faster R-CNN [49] — це вдосконалення методу Fast R-CNN [48], що підвищує ефективність виявлення об'єктів за рахунок введення генератора кандидатів регіонів (Region Proposal Network, RPN). Фундаментальна інновація полягає в інтеграції RPN у загальну архітектуру, що забезпечує наскрізне навчання як для генерації координат кандидатів, так і для задачі класифікації об'єктів.

Faster R-CNN починається з проходження вхідного зображення через згорткову нейронну мережу (CNN), яка виділяє об'єкти та створює мапу репрезентацій. Потім RPN використовує цю мапу репрезентацій, щоб запропонувати координати потенційних

кандидатів та відповідні оцінки того, чи знаходиться будь-який об'єкт в середині запропонованих координат. RPN, щоб ефективно генерувати кандидатів регіонів, використовує так звані якірні обмежувальні рамки, які є попередньо визначеними блоками різних масштабів та пропорцій. Запропоновані регіони з RPN об'єднуються з векторами репрезентацій, згенерованими CNN, алгоритмом об'єднання ROI Pooling, а отримані репрезентації подаються у пов'язані шари для класифікації та регресії координат об'єктів. Результатом класифікації є ймовірності приналежності до класів об'єктів, тоді як результат регресії уточнює координати обмежувальної рамки.

Під час навчання мережі, що складається з CNN, RPN, класифікатора та регресійної мережі, виконується навчання усіх мереж одночасно. Функція втрати охоплює як задачу генерації кандидатів регіонів, так і задачу виявлення об'єкта, забезпечуючи наскрізний процес оптимізації усіх компонентів методу Faster R-CNN. Схематичне представлення основних методу Faster R-CNN зображено на рисунку 1.3 [49].

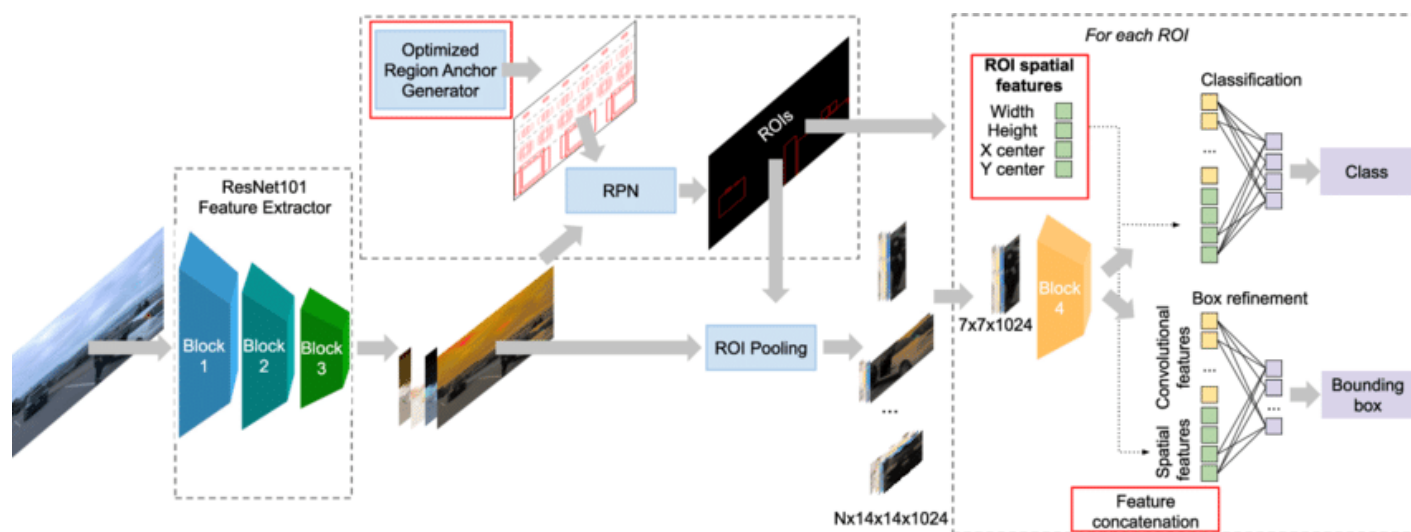


Рисунок 1.3 – Представлення основних кроків методу Faster R-CNN [49]

Mask R-CNN [41] — це вдосконалення методу Faster R-CNN [49], призначене для вирішення задачі сегментації об'єктів шляхом включення додаткової підмережі для

генерації масок об'єктів замість обмежувальних рамок. Він розширює можливості виявлення об'єктів, що забезпечує більш точне визначення меж об'єктів на зображенні. Метод базується на використанні спільної згорткової нейронної мережі (CNN) для задач детекції та сегментації.

Одночасно з гілками генерації регіонів кандидатів та виявлення об'єктів, наявними у Faster R-CNN, Mask R-CNN включає гілку маски. Ця гілка працює на карті векторів репрезентацій і призначена для прогнозування масок сегментації для кожного регіону інтересу (ROI), запропонованого мережею пропозицій регіону (RPN). Підмережа для генерації сегментації використовує CNN для бінарної класифікації місцезнаходження меж маски для кожного кандидата (ROI). Під час навчання згенеровані маски порівнюються з наявною розміткою, а функція втрат визначається як попіксельна бінарна перехресна ентропія, що дозволяє отримати високі показники якості для цієї підмережі. Представлення основних кроків методу Mask R-CNN зображено на рисунку 1.4 [41].

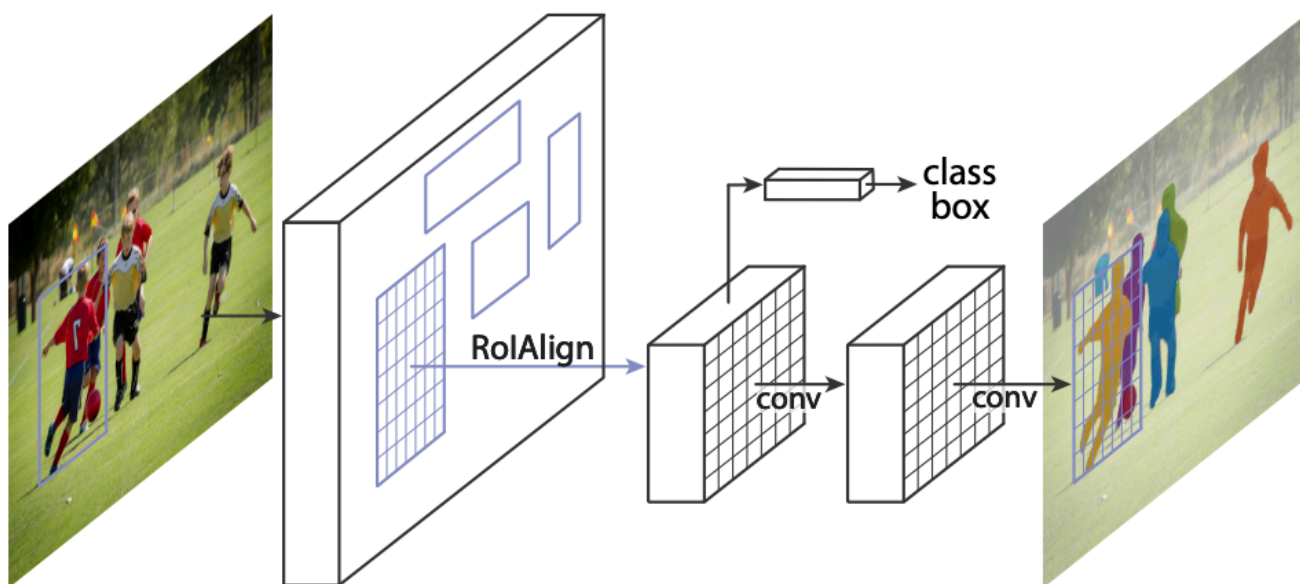


Рисунок 1.4 – Представлення основних кроків методу Mask R-CNN [41]

Аналогічно до Fast R-CNN [49] Mask R-CNN тренується наскрізно — одночасно оптимізуються підмережі RPN, класифікації об'єктів, регресії обмежувальної рамки та прогнозування попиксельних локацій масок. Ця наскрізна оптимізація гарантує, що модель навчиться виконувати завдання виявлення і сегментації об'єктів.

Mask R-CNN виявився потужною платформою для завдань, які вимагають детального просторового розуміння, наприклад, аналізу медичних зображень. Одночасне моделювання для задач детекції і сегментації сприяло вдосконаленню точності прогнозування та підвищило якість програм комп'ютерного зору, які потребують детального окреслення об'єктів.

Cascade R-CNN [40] спеціально розроблений для підвищення точності генерації регіонів об'єктів. Ця модель удосконалює Faster R-CNN [49], додаючи каскад детекторів, кожен з яких послідовно навчається, зосереджуючись на генерації локалізації об'єктів з різним рівнем точності.

В основі архітектури стоїть згортова нейронна мережа (CNN), яка відповідає за генерацію мапи репрезентації із вхідного зображення. Ці мапи репрезентацій далі подаються на каскад нейронних мереж. Кожен елемент каскаду є детектором об'єктів, навченим для виявлення об'єктів з певною точністю локалізації. Перший детектор у каскаді зосереджується на легко помітних об'єктах, як правило, із вищими оцінками достовірності. Помилкові спрацьовування з цього етапу згодом відфільтровуються, а кандидати, що залишилися, переходять до наступного етапу каскаду. Подальші етапи призначені для обробки дедалі складніших зразків, поступово підвищуючи точність локалізації об'єктів та точність детекції. Під час навчання кожен етап каскаду навчається незалежно, а помилкові спрацьовування з одного етапу стають негативними зразками для наступного етапу. Цей наскрізний підхід до навчання вдосконалює здатність детектора розрізняти об'єкт і фон, причому кожен етап сприяє підвищенню точності локалізації об'єктів.

Каскадна структура дозволяє більш деталізовано обробляти кожен з об'єктів, ефективно зменшуючи кількість помилкових спрацьовувань і підвищуючи точність локалізації. Cascade R-CNN має переваги там, де досягнення високої точності локалізації має критичне значення, наприклад у програмах, пов'язаних із завданнями безпеки, спостереження, розпізнавання критичних об'єктів тощо. Представлення основних кроків методу Cascade R-CNN зображено на рисунку 1.5 [40].

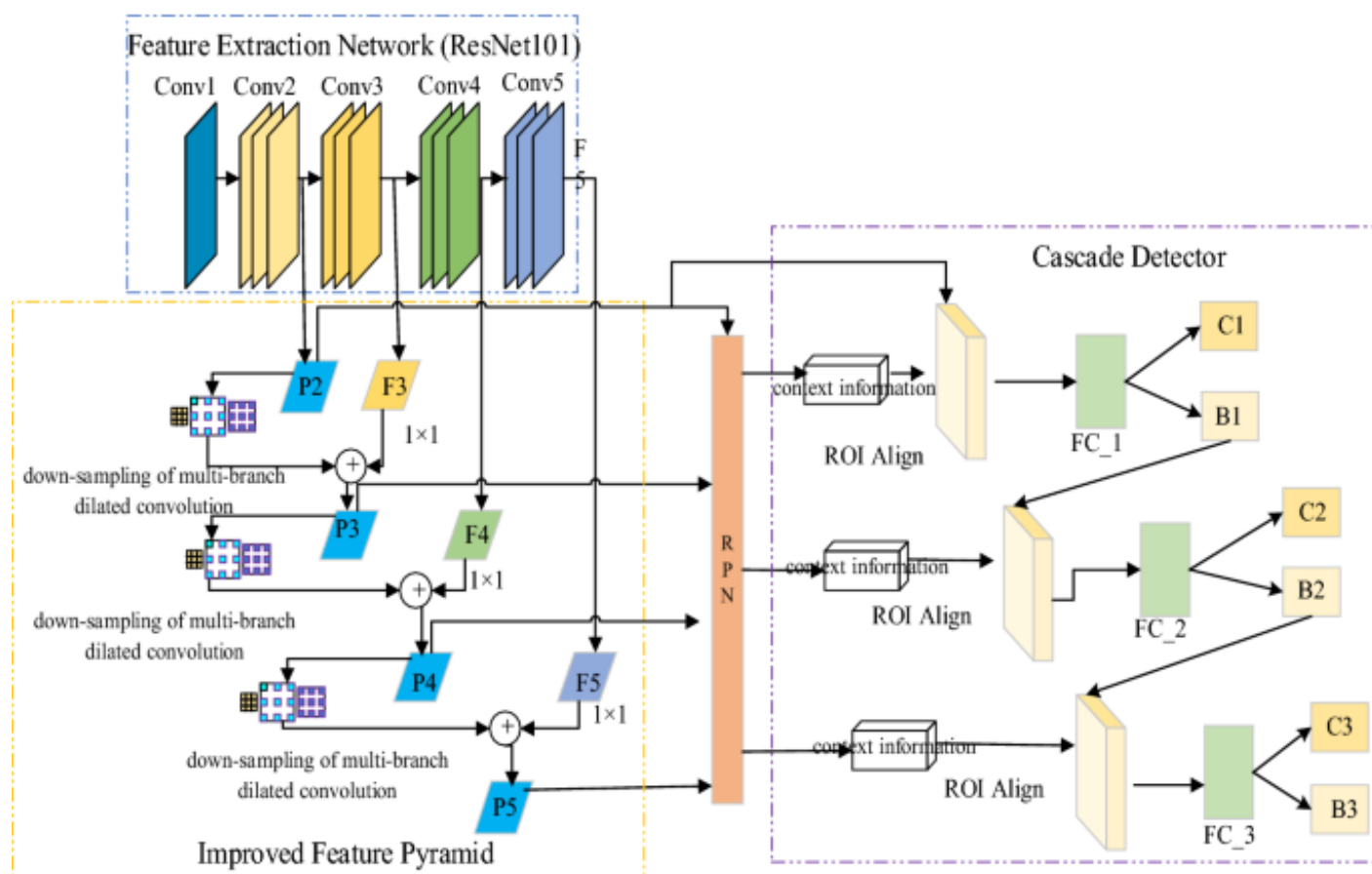


Рисунок 1.5 – Представлення основних кроків методу Cascade R-CNN [40]

Single Shot Multibox Detector (SSD) [38] спеціально розроблений для досягнення кращого балансу між швидкістю виконання детекції та точністю виявлення та локалізації об'єктів. За своєю суттю SSD представляє метод, який виконує детекцію об'єктів шляхом додавання в існуючі моделі CNN декількох додаткових згорток, що виконують задачі класифікації та регресії координат об'єктів. Ідея SSD зосереджена



навколо згорткової нейронної мережі (CNN), як правило, похідної від усталених моделей, таких як VGG або ResNet, що відповідає за генерацію мап репрезентацій. На відміну від інших методів та підходів детекції об'єктів, які використовують одну мапу репрезентацій, SSD містить декілька мап репрезентацій з різною роздільною здатністю, що дозволяє знаходити різні за розмірами об'єкти. Ці мапи репрезентацій потім використовуються для прогнозування координат об'єктів та ймовірностей класів для детекції об'єктів у різних розмірах. Кожна мапа репрезентацій сприяє захопленню об'єктів різного розміру, гарантуючи, що детектор здатний працювати з широким діапазоном масштабів за один прохід. Представлення основних кроків методу SSD зображено на рисунку 1.6 [38].

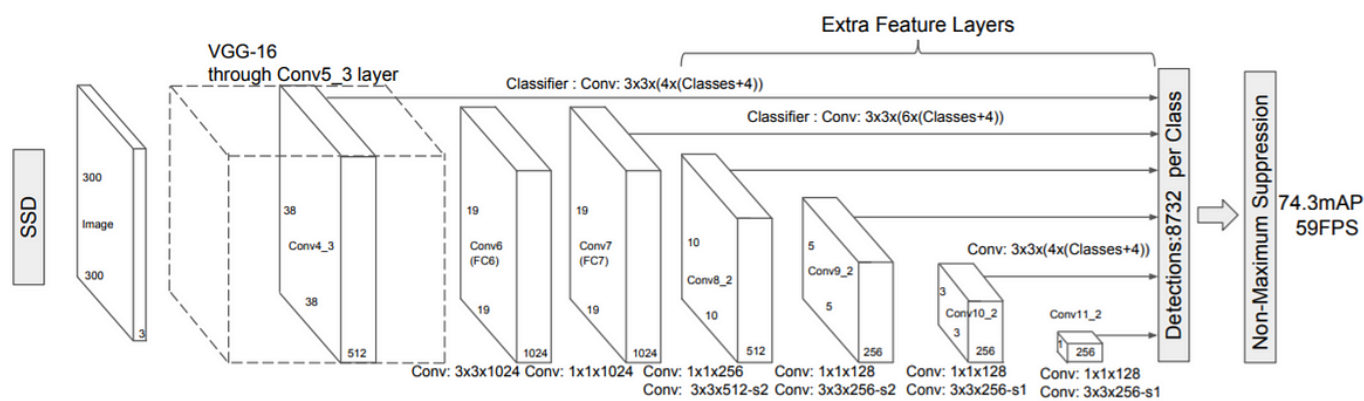


Рисунок 1.6 –Представлення основних кроків методу SSD [38]

SSD використовує набір якірних блоків для генерації локації кандидатів для об'єктів, аналогічно до RPN Faster R-CNN. Ці блоки служать еталонними шаблонами для прогнозування обмежувальних рамок і ймовірностей класу. Аналогічно до RPN Faster R-CNN локації кандидатів уточнюють регресійною мережею, щоб ті ліпше узгоджувались із реальним розташуванням об'єктів. Наявність декількох мап репрезентації, до яких застосовують свої якірні блоки, гарантує широке покриття об'єктів з різними розміром, місцезнаходженням та дистанцією від камери.



Під час прогнозування SSD генерує всі мапи репрезентації одночасно, надаючи повний набір даних для прогнозування всіх об'єктів різного розміру одночасно. Цей високопаралельний підхід допомагає моделі досягнути високої степені ефективності, дозволяючи виконувати детекцію об'єктів в реальному часі на процесорах з малою обчислювальною потужністю.

Отже, генерація мультирозмірних мап репрезентацій у поєднанні з використанням підходу якірних блоків дозволяє SSD досягати високої продуктивності з точки зору точності та швидкодії на малопотужних обчислювальних системах. Метод має переваги у сценаріях, що вимагають ефективної та точної детекції об'єктів, таких як управління робототехнікою та автономне керування автомобілем.

YOLOv10 [50], або You Look Only Once v10, є одним з найсучасніших методів детекції об'єктів, який вдосконалює нейромережу YOLO та підвищує її продуктивність порівняно з попередніми версіями. Метод зберігає концепцію одноетапного прогнозування координат об'єктів, ймовірностей для класів об'єктів і вводить додатковий показник об'єктності, що підвищує якість знайдених об'єктів та забезпечує роботу в реальному часі. YOLOv10 використовує новітні технології, зокрема Vision Transformers (ViTs), які підвищують контекстуальне розуміння об'єктів та забезпечують гнучкість моделі для адаптації до різних масштабів об'єктів на зображеннях.

Метод YOLOv10 включає вдосконалену піраміду репрезентацій для обробки об'єктів різних розмірів. Завдяки інтеграції трансформерів та покращеній структурі, метод здатний передавати характеристики між різними рівнями глибини, що підвищує здатність виявляти дрібні об'єкти навіть у складних сценах. Перевагою YOLOv10 є висока швидкість обробки зображень.

Для забезпечення точного передбачення меж об'єктів YOLOv10 використовує адаптивну систему якорів, яка налаштовується для різних категорій об'єктів на основі метрик Intersection over union (IoU). Це дозволяє моделі підвищити точність виявлення в задачах з об'єктами різних форм і розмірів, покращуючи загальні показники середньої

точності (mAP). На відміну від попередніх версій, YOLOv10 має значно вищу точність детекції завдяки мікротюнінгу, який оптимізує ваги для окремих класів об'єктів.

Навчання YOLOv10 базується на наскрізній оптимізації функції втрат, яка враховує помилки класифікації, локалізації та об'єктності. Вдосконалений процес навчання з використанням великих мовних моделей (LLMs) забезпечує генерацію додаткових атрибутів для об'єктів, що значно підвищує гнучкість моделі для детекції у складних і динамічних умовах. Представлення основних кроків методу YOLOv10 зображено на рисунку 1.7 [50].

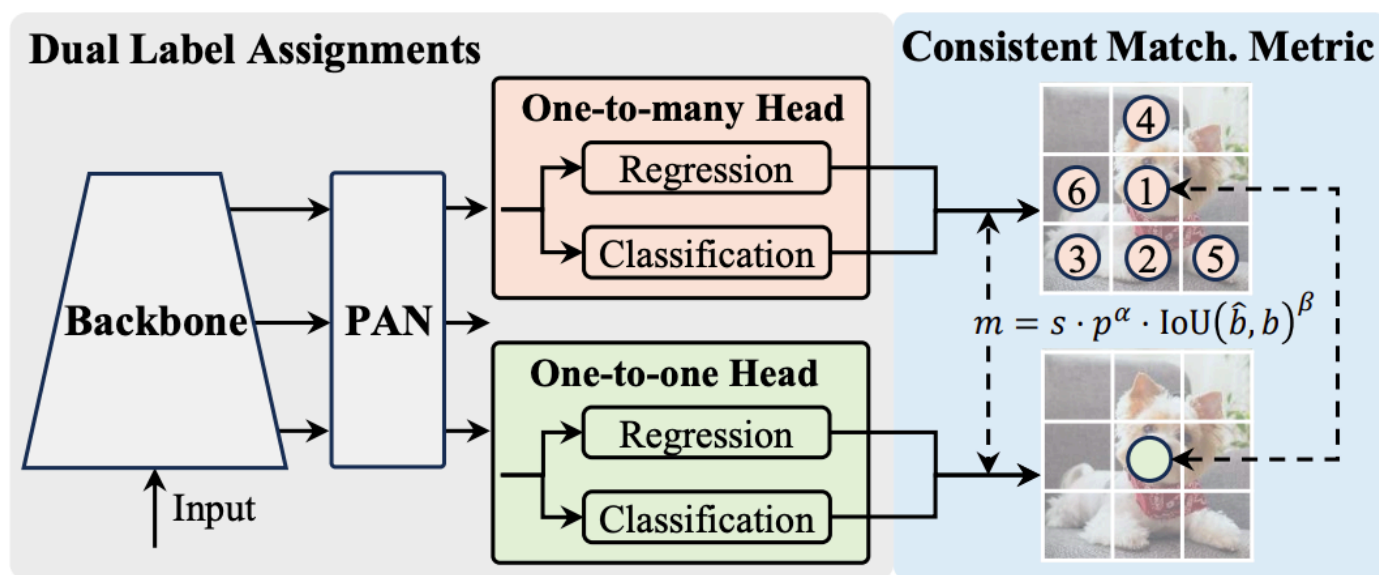


Рисунок 1.7 –Представлення основних кроків методу YOLOv10 [50]

Порівняно з іншими методами детекції, такими як R-CNN, Fast R-CNN, Faster R-CNN, SSD та Mask R-CNN, YOLO v10 демонструє оптимальний баланс між швидкістю та точністю. Наприклад, методи сімейства R-CNN базуються на багатоетапній обробці, що включає вибір регіонів, класифікацію та корекцію меж об'єктів. Це забезпечує високу точність, однак значно уповільнює процес детекції, що робить ці методи менш придатними для задач реального часу. Водночас YOLO v10, завдяки одноетапному

підходу та вдосконаленій архітектурі нейромережі, дозволяє одночасно виявляти та класифікувати об'єкти за один прохід, що суттєво зменшує час обробки.

### **1.3 Порівняння методів детекції об'єктів**

Для порівняння методів детекції об'єктів, таких як R-CNN, Fast R-CNN, Faster R-CNN, Cascade R-CNN, Mask R-CNN, SSD, та YOLO v10, було застосовано декілька ключових критеріїв. Основними параметрами, що слугували для оцінювання, стали архітектура нейромережі, швидкість обробки зображень, точність детекції, призначення для конкретних завдань, а також унікальні особливості кожного методу.

Структура та ключові компоненти кожного методу є важливим фактором, який визначає його ефективність, адаптивність і можливість інтеграції з іншими методами. Наприклад, R-CNN використовує алгоритм Selective search для пропозиції регіонів, що забезпечує точність, але значно знижує швидкість. На відміну від цього, Faster R-CNN і YOLO v10 інтегрують генерацію локацію об'єктів і їх класифікацію в єдиний процес, що значно скорочує час обробки.

Швидкість обробки є вирішальним критерієм для задач реального часу, таких як відеоспостереження, автономні транспортні засоби або мобільні додатки. Наприклад, методи SSD та YOLO v10 демонструють надзвичайно високі показники обробки (від 20 до 300 кадрів в секунду), що робить їх придатними для сценаріїв, де важливо мінімізувати затримку. Водночас, такі моделі як Mask R-CNN та Cascade R-CNN, які мають вищу точність, обробляють дані повільніше, що може обмежувати їх застосування в режимі реального часу.

Точність детекції вимірюється за допомогою середньої точності (mAP), і є критичною для забезпечення надійності детекції. Найвищу точність демонструє YOLO v10, яка досягає 80-85% mAP на наборах даних, таких як COCO, що робить її однією з найбільш точних моделей. Cascade R-CNN також забезпечує високу точність завдяки

каскадній структурі, що уточнює межі об'єктів на кожному етапі, однак це також підвищує обчислювальну складність.

Кожен метод має свої переваги в залежності від задачі. R-CNN, Fast R-CNN, Faster R-CNN і Cascade R-CNN орієнтовані головним чином на детекцію об'єктів. Mask R-CNN, у свою чергу, призначений для суміщення задач детекції і сегментації, що робить його унікальним серед інших підходів. Ця властивість дозволяє Mask R-CNN вирішувати завдання, де необхідне точне виділення меж об'єктів, такі як медичний аналіз зображень або розпізнавання об'єктів з неоднорідною структурою.

Особливості кожного методу впливають на її адаптивність та універсальність. Наприклад, метод R-CNN потребує значного часу для обробки зображень через Selective Search, тоді як Faster R-CNN інтегрує Region Proposal Network (RPN), що забезпечує значне скорочення часу. YOLO v10 забезпечує швидку обробку та високу точність завдяки оптимізованій одноетапній мережевій структурі, яка дозволяє прогнозувати об'єкти за сіткою координат, охоплюючи широкий спектр об'єктів на різних рівнях масштабування. SSD також відрізняється високою швидкістю обробки завдяки використанню одноетапного підходу, однак може поступатися точністю у складних сценах. Порівняльний аналіз методів детекції наведено в таблиці 1.1. З таблиці можна зробити висновок, що методи SSD і YOLO v10 є найбільш придатними для задач реального часу завдяки високій швидкості обробки кадрів, проте YOLO v10 також демонструє більш високу точність і здатність до масштабування, що робить його лідером серед методів детекції об'єктів для реальних застосунків. Cascade R-CNN та Mask R-CNN забезпечують вищу точність і підходять для завдань, де вимоги до точності важливіші за швидкість, наприклад для обробки статичних зображень у високотехнологічних галузях.

Таблиця 1.1 – Порівняльний аналіз різних методів детекції об'єктів

Метод	Архітек-тура	Швид-кість (FPS)	Точ-ність (mAP)	Призна-чення	Особливості
<b>R-CNN</b>	Selective Search + CNN	~2 FPS	~65%	Детекція об'єктів	Використовує Selective Search для пошуку регіонів, повільна через етапи регіональних обчислень.
<b>Fast R-CNN</b>	CNN	~7 FPS	~70%	Детекція об'єктів	Зменшив обчислювальні витрати завдяки використанню ROI Pooling; тренується кінцево.
<b>Faster R-CNN</b>	Region Proposal Network	~5-17 FPS	~75%	Детекція об'єктів	Додано Region Proposal Network (RPN) для покращення швидкості; виконує пропозицію регіонів і класифікацію в одному кроці.
<b>Cascade R-CNN</b>	Multi-Stage Faster R-CNN	~6 FPS	~80%	Детекція об'єктів	Каскадні етапи поліпшують точність через послідовне уточнення; придатний для високоточної детекції.
<b>Mask R-CNN</b>	Faster R-CNN + Mask Branch	~4-8 FPS	~75% детекція 70% сегмен-тація	Детекція + сегментація	Розширює Faster R-CNN для семантичної сегментації шляхом додавання маски об'єктів; особливо корисний для задач сегментації.
<b>SSD</b>	CNN	~20-60 FPS	~70%	Детекція об'єктів	Один прохід для регіональних пропозицій та класифікації; висока швидкість, підходить для реального часу.
<b>YOLO v10</b>	CNN + Grid	~200-300 FPS	~75%	Детекція об'єктів	Виконує детекцію в одному кроці за сітковою схемою; дуже висока швидкість, підходить для реальних додатків, значно покращений мапінг об'єктів.

## 1.4 Фундаментальні візуально-мовні моделі машинного навчання

Методи комп'ютерного зору традиційно покладалися на згорткові нейронні мережі (CNN) як основний підхід до побудови архітектури для задач розпізнавання зображень і відео [30; 55-60]. Однак успіх архітектури Transformer в обробці природної мови, прикладом якого є [61], надихнув на розробку візуального трансформеру [62], який безпосередньо застосовує Transformer до зображень, що забезпечує значне підвищення продуктивності в розпізнаванні зображень. Отже, [62] ініціював зміну парадигми в підходах до архітектур нейронних мереж для розпізнавання зображень від CNN до Transformers.

Подальші дослідження, такі як, наприклад, DeiT [63], з'явилися з метою підвищення ефективності архітектури візуального трансформеру [62]. Крім того, застосування трансформерів для розпізнавання на відео набрало обертів із появою інших моделей [64-68], які масштабують підходи з використанням архітектури Transformer на часову розмірність.

У сфері навчання візуально-мовних моделей модель CLIP [44] стала еталоном для попередньо навчених візуально-мовних моделей. CLIP використовує функцію втрат InfoNCE [69] для контрастування між зображенням та текстом, що породило декілька варіантів моделей [70-72], які поєднують різні навчальні підзадачі, включаючи зіставлення зображення та тексту, або відновлення маскованого зображення та тексту. Інше дослідження [73] включає завдання класифікації у фазу попереднього навчання візуально-мовної моделі, що призводить до значного підвищення точності порівняно з використанням крос-ентропії. Представлення основних кроків роботи методу CLIP [44] наведено на рисунку 1.8 [44].

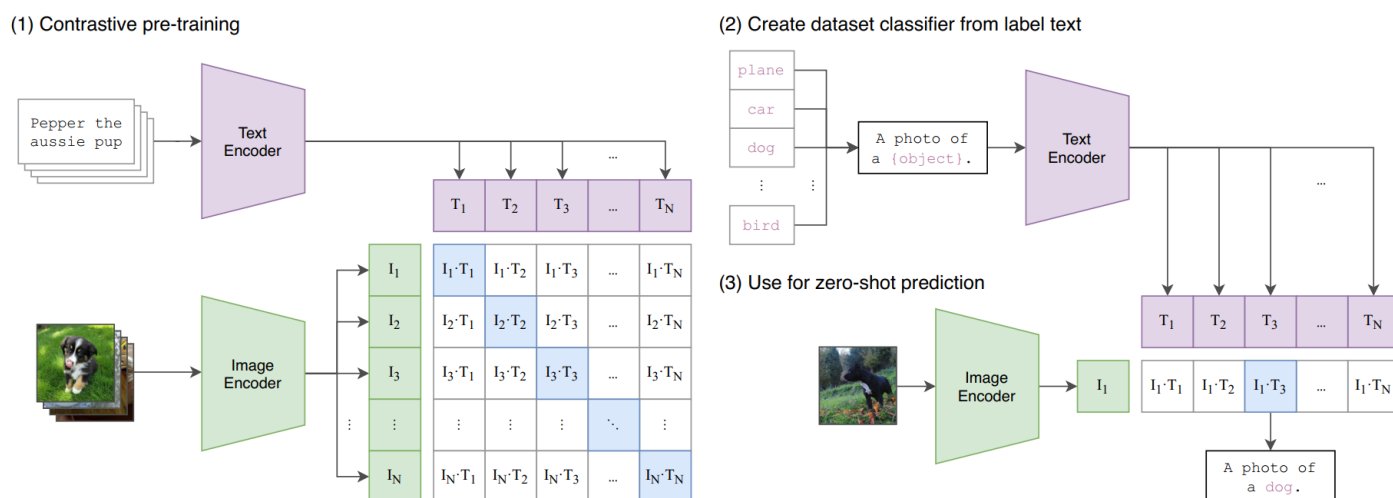


Рисунок 1.8 – Представлення основних кроків роботи методу CLIP [44] складається з двох фаз: (1) попереднього навчання моделі з контрастною функцією втрат та (2-3) використання в режимі нульового навчання.

У контексті навчання візуально-мовних моделей кілька досліджень [74-76] використовують підходи знаходження та фільтрації схожих екземплярів даних для підготовки навчальних даних. Крім того, підходи [77-79] розширюють підхід до попереднього навчання візуально-мовних моделей, запропонованого в CLIP [44], шляхом зіставлення відео та тексту, використовуючи контрастну функцію втрат, і згодом використовують подібність між відео та текстом для розпізнавання відео під час тестування нейронної мережі.

Запропонований у даному науковому дослідженні підхід досліджує ефективну передачу знань між візуальною та текстовою гілками візуально-мовних моделей з подальшим використанням в рамках класичної парадигми комп'ютерного зору.

Широко застосовувані моделі в області комп'ютерного зору зазвичай пристосовані до конкретних задач і доменів, що часто вимагає ручної розмітки наборів даних для їхнього донавчання або перенавчання. Однак останні дослідження в області комп'ютерного зору представили концепцію фундаційних моделей, які націлені на

зменшення впливу цих обмежень. Наприклад, CLIP [44] використав величезні за масштабами набори даних, які були зібрані з Інтернет та містили велику кількість зашумлених пар зображення-текст для навчання візуально-мовних моделей за допомогою контрастного навчання. Цей підхід забезпечує вивчення надійних та інваріантних репрезентацій для зображень і тексту, забезпечуючи потужні можливості нульового навчання. INTERN [80] розширює цю парадигму шляхом включення кількох етапів попереднього навчання з самоконтролем (self-supervision), використовуючи значну кількість неанотованих пар зображення-текст разом із зображеннями, які були розмічені вручну.

Florence [81] вдосконалює цей напрямок досліджень, інтегруючи уніфіковане контрастне навчання [82; 83] та складні моделі адаптації, тим самим сприяючи уніфікації та можливості використання моделі до широкого спектру завдань комп'ютерного зору в різних умовах. SimVLM [84] і OFA [85], з іншого боку, проводять навчання моделі енкодера-декодера з оптимізацією генеративних функцій втрат, забезпечуючи конкурентоспроможну продуктивність у різних мультимодальних завданнях. Крім того, CoCa [70] об'єднує контрастне навчання, подібне до CLIP [44], із генеративним навчанням, подібним до SimVLM [84].

Підходи як CoCa [70], так і Florence [81] досягли значних результатів у розпізнаванні на зображенні, що в свою чергу покращило результати на відео, особливо на таких наборах даних, як Kinetics, однак, коли справа доходить до мультимодальних завдань на відео, то такі моделі показують незадовільні результати. Тому були створені такі моделі як VIOLET [86], All-in-one [87] та LAVENDER [88], які використовують масковані текст та відео для підвищення точності мультимодального прогнозування і пропонують уніфікувати завдання через моделювання замаскованої мови. Незважаючи на те, що ці моделі успішно працюють у мультимодальних тестах, об'єм навчальних даних для цих моделей часто залишається обмеженим, що призводить до складності їх застосування у прикладних розробках з розпізнавання на відео.



На противагу цим моделям, MERLOT Reserve [89] пропонує новий шлях до навчання мультимодальних моделей, зібравши великий набір даних з 20 мільйонів пар відео-текст-аудіо для спільного навчання відео-, аудіо- та текстових репрезентації та використовуючи контрастну відповідність між модальностями.

MERLOT Reserve [89] досягає найбільшої точності не лише для задач розпізнавання на відео, але й у візуальному розумінні, що дозволяє його широке використання для прикладних задач.

Варто зазначити, що поточні візуально-мовні фундаційні моделі для відео у порівнянні з їхніми VLM аналогами, які тренувалися лише на зображеннях, демонструють обмеження у різномірності задач, до яких їх можна застосовувати, особливо в контексті точних завдань, таких як просторово-часова локалізація, що спричинено відсутністю великої кількості розмічених даних для таких моделей. Саме тому парадигма мультимодального попереднього навчання стала наріжним каменем для навчання фундаційних моделей, починаючи з розвитку попереднього навчання на парах зображення-текст, еволюціонуючи до широкомасштабного попереднього навчання на парах відео-текст з подальшим навчанням моделі для конкретних завдань [86; 89]. Основні підходи [90; 91] традиційно використовують попередньо навчені візуальні та мовні енкодери для генерації векторів властивостей для відео та тексту. Однак більш пізні підходи [92; 93] продемонстрували доцільність наскрізного навчання. Ці підходи часто охоплюють два або три завдання для попереднього навчання, включаючи моделювання замаскованої мови [88], відповідність відео-тексту [87], контрастне навчання відео-тексту [94] та моделювання замаскованого відео-тексту [95], серед інших.

На відміну від попередньо згаданих робіт, InternVideo [80] виступає як універсальна фундаційна модель для задач на відео. . Принцип навчання InternVideo наведено на рисунку 1.9 [80].

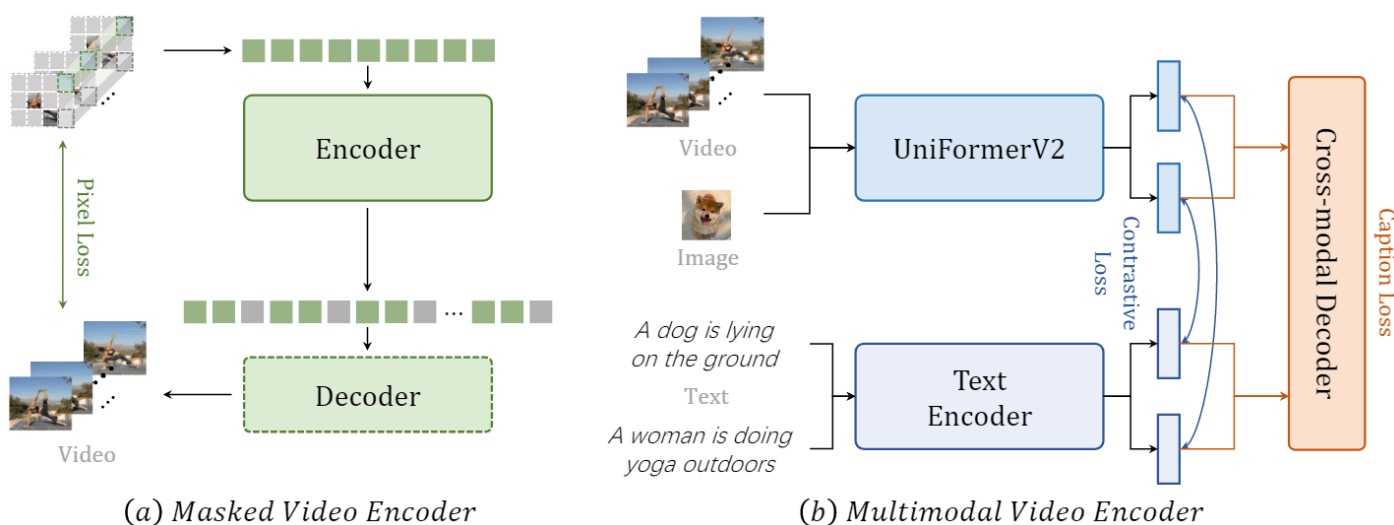


Рисунок 1.9 – Принцип навчання InternVideo [80]: (а) попереднє навчання візуального енкодера на маскованих зображеннях та (б) навчання на мультимодальних парах відео-текст

З точки зору архітектури моделі, InternVideo [80] використовує Vision Transformer (ViT) [62] як свою базову структуру, доповнену напрацюваннями з UniFormerV2 [96] і додатковими локалізаційними модулями для просторово-часового моделювання, що вдосконалює процес створення багаторівневих просторово-часових репрезентацій. InternVideo динамічно генерує вектори репрезентації від двох трансформерів через їх взаємодію, таким чином використовуючи сильні сторони генеративних і контрастних парадигм навчання, що дає точніший результат. InternVideo встановив нові еталони точності на 34 наборах даних та 10 різних задачах розпізнавання на відео.

## 1.5 Детекція об'єктів на відео

Виявлення об'єктів на відео передбачає розширення підходів детекції об'єктів по зображеннях на часову компоненту відео. Фундаментальний підхід полягає у застосуванні методів детекції об'єктів кадр за кадром, таким чином аналізуючи вміст відео, що змінюється з часом.

Відео — це, по суті, послідовність кадрів, кожен з яких представляє нерухоме зображення. Методи детекції об'єктів на основі згорткових нейронних мереж (CNN) або іншої архітектури, таких як YOLO або SSD, застосовуються незалежно до кожного кадру. Мета полягає в тому, щоб ідентифікувати та локалізувати об'єкти в кожному кадрі, подібно до того як це робиться на зображеннях.

Часова, або так звана темпоральна, інформація використовується шляхом обробки послідовності кадрів і відстеження руху виявлених об'єктів у послідовності кадрів. Алгоритми відстеження, такі як фільтр Калмана [97] або інші кореляційні фільтри, допомагають пов'язувати координати знайдених об'єктів в часі. Це відстеження дозволяє підтримувати узгодженість у розпізнаванні об'єктів, коли вони рухаються на відео (рис. 1.10) [97]. Крім того, такі алгоритми, як оптичний потік [98], можна використовувати для оцінки руху пікселів між кадрами, допомагаючи зрозуміти динаміку об'єктів та передбачити їхнє майбутнє положення. Оптичний потік є методом, що визначає видимий рух об'єктів, поверхонь і країв у візуальному полі, спричинений відносним рухом між спостерігачем і сценою. Цей темпоральний контекст підвищує точність і надійність виявлення об'єктів на відео.

Існує декілька підходів до обчислення оптичного потоку, серед яких найбільш відомі методи Гуннара-Фарнебака [99] та Лукас-Канаде [100]. Метод Гуннара-Фарнебака [99] використовує поліноміальну апроксимацію інтенсивностей пікселів для обчислення потоку, що дозволяє досягати високої точності при відносно невеликій обчислювальній складності. Метод Лукас-Канаде [100], у свою чергу, базується на припущенні про постійність інтенсивності пікселів і малих переміщеннях, розраховуючи оптичний потік шляхом мінімізації помилки між відповідними пікселями.

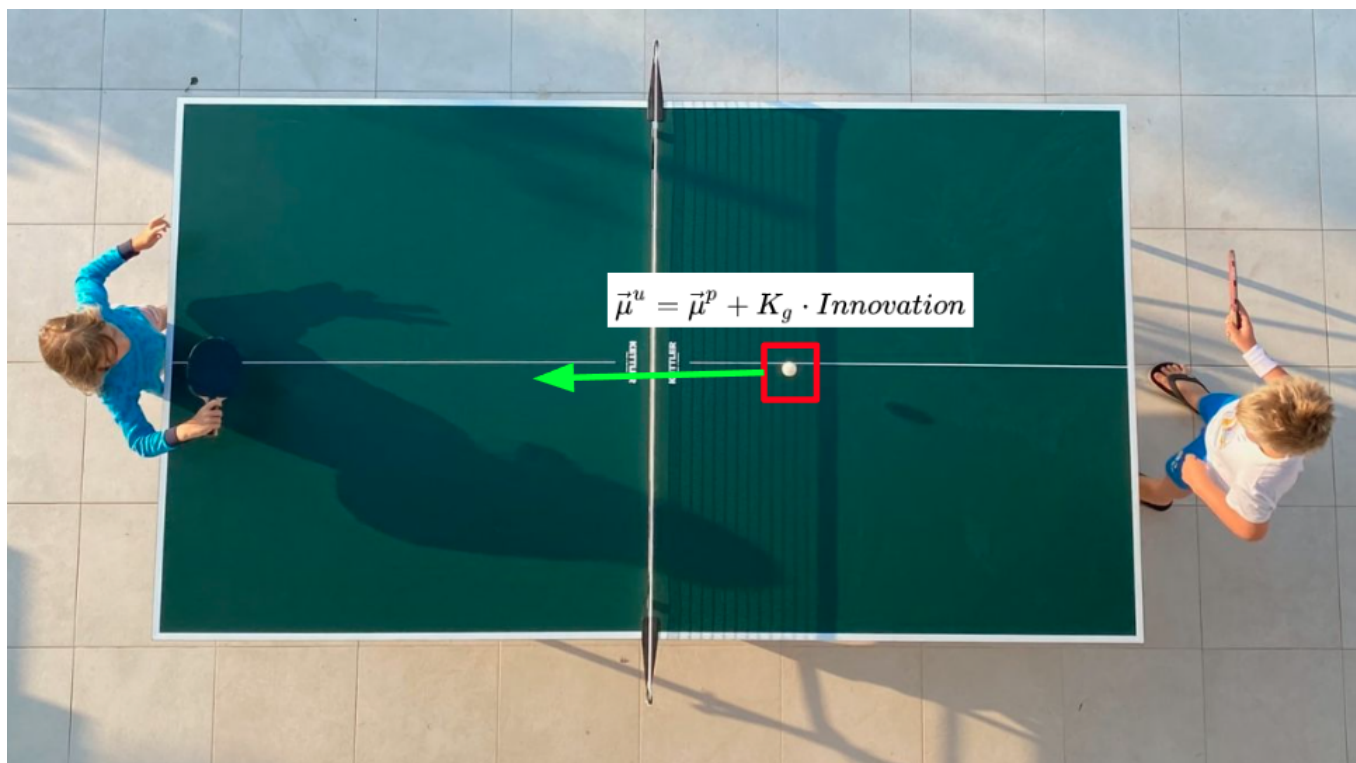


Рисунок 1.10 – Використання фільтру Калмана для відстежування об'єкта у кадрі за допомогою прогнозування місцеположення об'єкта на наступному кадрі [97]

Застосування оптичного потоку включає аналіз руху, стабілізацію відео, відстеження об'єктів, а також розширені застосування в робототехніці та комп'ютерному зорі, де необхідно розуміти та передбачати поведінку об'єктів у динамічних сценах. На рисунку 1.11 [98] представлено приклад оптичного потоку, який ілюструє векторне поле руху між двома послідовними кадрами відео. Вектори руху вказують напрямок та величину переміщення пікселів, дозволяючи детально аналізувати траєкторії рухомих об'єктів та їх взаємодії у сцені.

Для застосунків, які мають працювати в реальному часі, ефективність методу детекції об'єктів має вирішальне значення. Такі методи, як Single Shot Multibox Detector (SSD), You Look Only Once (YOLO) і Faster R-CNN, є широко використовуваними завдяки їхній здатності швидко обробляти кадри, що робить їх бажаними методами при побудові застосунків для аналізу відео.

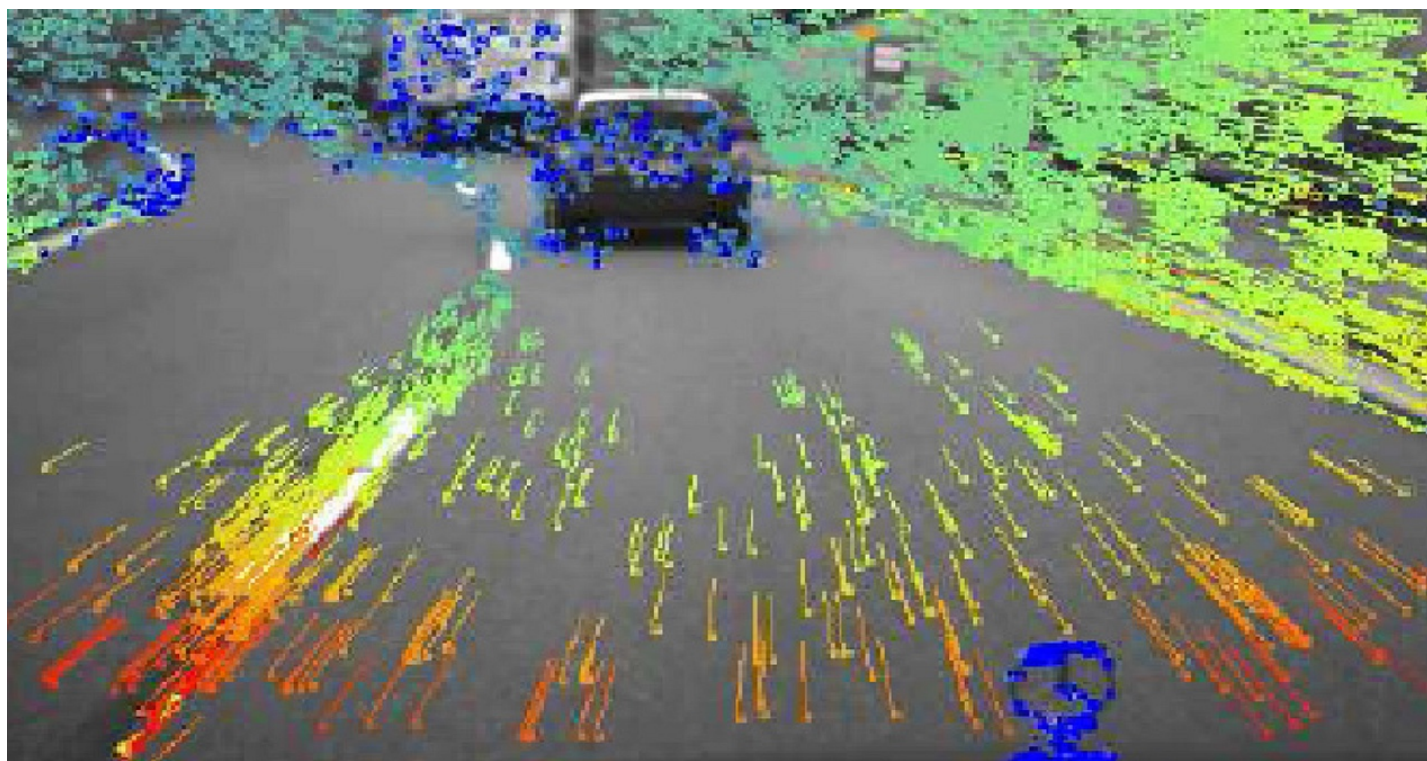


Рисунок 1.11 – Приклад оптичного потоку, що розрахований по об’єктам дорожнього трафіку [98]

Підсумовуючи, детекція об’єктів на відео передбачає застосування методів детекції об’єктів по зображеннях кожного кадру, відстеження об’єктів у часі та використання темпоральної інформації для підвищення точності та підтримки консистенції положення об’єктів у послідовності кадрів на відео. Ці методи знаходять застосування в різних областях, включаючи системи відеоспостереження, системи автономного керування транспортними засобами та різні системи взаємодії людини з комп’ютером.

## **1.6 Опис процесу розмітки даних для задач машинного навчання**

Розмітка зображень служить ключовим етапом у навчанні моделей комп’ютерного зору на основі методів машинного навчання, що полягає у генерації структурованих даних, які надалі використовуються для навчання моделі машинного навчання.

Практично кожна модель комп'ютерного зору покладається на ретельно розмічені зображення, що лежать в основі навчальних наборів даних. Ці розмічені зображення допомагають передавати навчальні дані в моделі комп'ютерного зору, які навчаються, виконувати певні завдання, наприклад, ідентифікацію чорних автомобілів Ford певного року випуску і з певними конструктивними характеристиками.

Інтеграція активного навчання в модель комп'ютерного зору покращує її адаптивність і здатність до навчання, тим самим збільшуючи її ефективність для розгортання у різних операційних умовах. Саме тому неможливо переоцінити значення вхідних даних у формуванні результатів методів машинного навчання, включаючи нейронні мережі, підходи до навчання яких орієнтовані на великі об'єми даних, що підкреслює критичну важливість отримання якісних наборів даних.

Важливо визнати, що алгоритмам комп'ютерного зору наразі не вистачає можливостей для виправлення людських помилок, допущених під час ручної розмітки даних. Тому цінність будь-якого навчального набору даних за своєю суттю пов'язана з точністю розмітки, яку вони містять.

Управління командою розмітчиків передбачає навігацію в складному ландшафті розмітки зображень, що потребує часу, навичок, розумного бюджету та застосування відповідних інструментів для забезпечення безперебійного виконання проєктів і отримання результатів, які виконують операції з даними, команди й керівників на яких можна покластися.

У широкому контексті розмітка зображень є основою успіху моделей комп'ютерного зору. Цей складний процес передбачає ручну розмітку зображень в наборі даних, що служить основою для навчання моделей комп'ютерного зору штучного інтелекту та машинного навчання. Метою розмітки зображень є точне позначення зображень, які використовуються для навчання моделі комп'ютерного зору. Розмітку зображення можна зробити повністю вручну або за допомогою автоматизації, щоб пришвидшити процес маркування.

Розмітка вручну є трудомістким процесом, оскільки вона вимагає, щоб розмітчик передивився кожен екземпляр даних і позначив її відповідною розміткою. Залежно від складності завдання та розміру набору даних цей процес може зайняти значну кількість часу, особливо якщо ви маєте справу з великим набором даних.

Використання методів автоматизації та машинного навчання, таких як активне навчання, може значно скоротити час і зусилля, необхідні для розмітки, а також підвищити точність розмітки даних. Вибираючи найбільш інформативні екземпляри даних для розмітки, активне навчання дозволяє навчати моделі машинного навчання ефективніше, не втрачаючи точність. Однак важливо зазначити, що, хоча автоматизація може бути потужним інструментом, вона не завжди може замінити людський досвід, особливо у випадках, коли завдання вимагає специфічних знань або суб'єктивного судження [101-104].

Головною метою машинного навчання є навчити модель, будь то традиційна статистична модель або сучасна нейронна мережа, виявляти та використовувати закономірності у вхідних та вихідних даних навчального набору. Вхідні та вихідні дані можуть приймати форму тексту, зображень, аудіо- чи відеофайлів. Наприклад, у сценарії, де метою є визначення тональності голосу абонента в call-центрі, модель, працюючи з вхідним набором аудіоданих, ставить за мету визначити інтонацію особи, а вихідні дані представлені мітками позитивної чи негативної тональності. Аналогічно, для відеозаписів, що реєструють порушення автомобілем ліній STOP перед світлофором, задачею може бути визначення факту порушення на основі вхідного відеоматеріалу, а вихідними даними будуть мітки виявлених порушень.

Щоб досягти успішної роботи штучного інтелекту, необхідно передусім людиною вирішити завдання підготовки даних для машинного навчання. Цей процес, відомий як розмітка даних, визначається як етап підготовки інформації, що передається моделям машинного навчання. Його результативність та якість визначають ефективність та точність роботи навчених моделей.



Застосування аналогії, що дані у сучасному світі – це нафта XXI століття, виявляється особливо точною, зокрема у контексті того, що, подібно до сирої нафти, необроблені дані залишаються малоцінним ресурсом. Перед їх використанням слід провести очистку та обробку. Це аналогічно до "нової нафти" – навіть при зборі значної кількості неструктурованих даних їх використання в розробці програмних продуктів на основі машинного навчання вимагає передусім ретельної розмітки та видалення неточних або помилкових даних. Загальний процес розмітки набору даних наведено на рисунку 1.12 [101].



Рисунок 1.12 – Процес розмітки набору даних [101]

### 1.7 Підходи до організації розмітки даних

Ураховуючи складність та трудомісткість процесу розмітки даних, організація цього етапу є важливою складовою успішного проєкту, оскільки вона визначає не тільки якість отриманих даних, але й впливає на час виконання та вартість розмітки. Аналітики Cognilytica визначають п'ять основних підходів [101] до організації процесу розмітки даних:

- формування внутрішньої команди розмітчиків даних;



- аутсорсинг завдань спеціалізованим компаніям, що займаються розміткою даних;
- використання краудсорсингових площадок для розмітки;
- створення синтетичних даних;
- автоматична розмітка даних.

Порівняльний аналіз різних підходів наведено в таблиці 1.2.

Таблиця 1.2 – Порівняльний аналіз підходів до організації розмітки даних

Підхід	Переваги	Недоліки
Внутрішня команда розмітки	Простота у відслідковуванні прогресу; можливість внесення правок у правила; швидкість комунікації з усіма зацікавленими сторонами; висока якість отриманих даних.	Є довготривалим, якщо відсутні налагоджені процеси; необхідно наймати та навчати розмітчиків даних; погано масштабується; висока вартість побудови процесу з нуля.
Аутсорсинг розмітки даних	Не потрібно налагоджувати процеси та набирати команду; простота в управлінні проектом; висока якість отриманих даних.	Висока вартість процесу; затримки в комунікації; ускладнений процес в разі необхідності внесення правок до правил.
Краудсорсингові платформи	Масштабованість процесу розмітки даних; залучення розмітчиків даних з різним досвідом; швидкість розмітки даних; низька ціна одиниці даних.	Не можливо гарантувати якість даних; необхідність в багаторазовій перевірці щоб отримати якісні дані; складність в відслідковуванні статусу розмітки; не можливо розмітити приватні дані користувачів.

Продовження таблиці 1.2

Підхід	Переваги	Недоліки
Синтетичні дані	Можливість генерувати великі об'єми даних за короткі проміжки часу; висока якість даних.	Висока вартість процесу, яка викликана необхідністю в великій кількості обчислювальних потужностей.
Автоматична розмітка даних	Масштабованість процесу; можливість розмітки великих об'ємів даних за короткі проміжки часу; простота в використанні, низька вартість одиниці даних.	Низька якість отриманої розмітки, необхідність перевірки результатуючих даних людиною.

Формування внутрішньої команди розмітки включає формування команди розмітчиків даних з постійних працівників організації. Цей підхід має ряд переваг: відстеження прогресу виявляється простим, а отримані дані характеризуються високою точністю та рівнем якості. Однак за межами великих компаній із значним штатом розмітчиків даних така організаційна модель виявляється неефективною.

Аутсорсинг завдань розмітки даних дозволяє створити тимчасову команду висококваліфікованих фахівців для виконання завдань в обмежений період часу. Зазвичай аутсорсингові компанії мають вдосконалені процеси, пов'язані з організацією найму розмітчиків даних та процесу розмітки, з навчанням розмітчиків, контролем та забезпеченням ефективної комунікації з урахуванням нововведень. Замовник має можливість обирати професіоналів для свого проєкту, враховуючи їхні навички та кваліфікацію. Цей варіант особливо привабливий для невеликих компаній, які володіють чітким розумінням щодо фінального результату, оскільки дозволяє досягти його за обмежений часовий проміжок.

Використання платформ для краудсорсингу є ефективним підходом залучення глобальної спільноти для виконання конкретних завдань. Завдяки можливості взяти на роботу з будь-якої точки світу та виконувати завдання по мірі їх доступності, цей підхід є надзвичайно швидким, ефективним та економічно доцільним. Тим не менше, платформи краудсорсингу можуть значно відрізнятися одна від одної за якістю розмітки, системою забезпечення якості та інструментами для управління проєктами та учасниками.

Синтетичні дані представляють собою стратегію для генерації нових даних, які включають атрибути, необхідні для конкретного проєкту. Одним з підходів створення таких даних є використання генеративних змагальницьких мереж (GANs). GANs використовують дві нейронні мережі - генератор і дискримінатор, які конкурують між собою у створенні реалістичних «підробок» даних та відрізненні справжніх даних від підроблених, що призводить до генерації нових даних високої якості. GANs дозволяють створювати абсолютно нові дані на основі існуючих наборів даних, роблячи їх ефективними для отримання великої кількості високоякісних даних за короткий проміжок часу. Проте використання підходів, які базуються на використанні GANs, на сьогодні вимагає значних обчислювальних ресурсів, що робить їх високовартісним підходом.

Автоматична розмітка даних визначається як процес використання скриптів для автоматизованого маркування даних. Цей процес спроможний автоматизувати широкий спектр завдань, таких як розмітка зображень та тексту, що виключає необхідність великої кількості розмітчиків даних. Комп'ютерна програма, що використовується для автоматичної розмітки, не вимагає перерви на відпочинок, що дозволяє очікувати отримання результатів набагато швидше, порівняно з ручним маркуванням даних. Однак, не зважаючи на переваги, цей підхід ще далекий від повної досконалості. Тому в практиці часто використовується комбінація програмної розмітки та спеціальної команди для забезпечення якості отриманих даних.

## 1.8 Обґрунтування необхідності прискорення розмітки

Оскільки якісна розмітка даних представляє собою ітеративний процес, що включає взаємодію з людиною – розмітчиком даних, сам цей процес виявляється витратним за часом. Згідно з аналітикою компанії Cognilytica [101], на середньостатистичному проєкті, пов'язаному з машинним навчанням, до 80% робочого часу витрачається на обробку даних та лише 20% на розробку та впровадження рішення.

Діаграма розподілу часу між різними діяльностями у сучасних проєктах з машинного навчання подана на рисунку 1.13 [101].

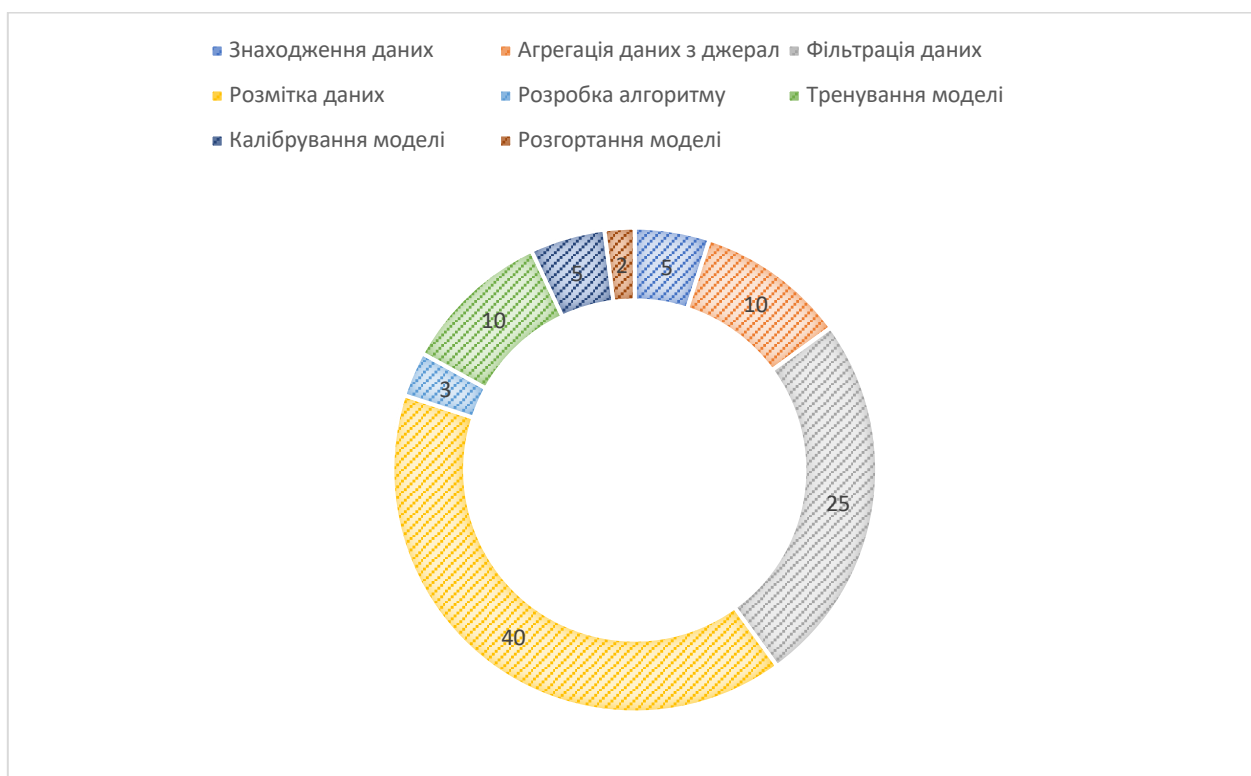


Рисунок 1.13 – Діаграма розбивки часу на різні активності в сучасних проєктах з використанням машинного навчання [101]

До активностей по роботі з даними відносяться: знаходження джерел даних, агрегація даних з цих джерел, фільтрація даних, розмітка даних та їх аугментація. До активностей, пов'язаних з навчання моделей, відносяться: розробка методів машинного

навчання, включаючи написання архітектури моделей нейронних мереж, тренування моделей, калібрування моделей та їх розгортання в робочому середовищі, будь то хмара чи персональний пристрій.

Враховуючи той факт, що створення обширних наборів даних є трудомістким та витратним процесом, технологічні гіганти здійснюють активні заходи для захоплення значної частини програмного забезпечення, пов'язаного з машинним навчанням. Це обумовлено їхньою здатністю швидко створювати об'ємні набори даних, використовуючи велику кількість розмітчиків даних, як штатних, так і контрактних, що функціонують в так званих фабриках по розмітці даних [102]. Такі фабрики являють собою централізовані установи або платформи, де великі команди розмітчиків вручну переглядають, класифікують та позначають дані для використання в моделях машинного навчання. Цей підхід дозволяє забезпечити високу якість і точність даних, необхідних для навчання складних моделей. Фабрики розмітки даних часто використовують передові інструменти та технології для автоматизації частини процесу, що додатково підвищує ефективність і продуктивність.

У зв'язку з цим для невеликих компаній та стартапів вирішення власних завдань вимагає винайдення альтернативних підходів, зокрема, використання публічних та спеціалізованих наборів даних. Вони можуть звертатися до відкритих платформ з розмітки даних, таких як Amazon Mechanical Turk або CrowdFlower, які надають доступ до широкої мережі фрілансерів для виконання різноманітних завдань з розмітки. Крім того, невеликі компанії часто розробляють нові методи напіваавтоматичної або автоматичної розмітки даних, застосовуючи активне навчання або трансферне навчання, що дозволяє значно скоротити витрати часу та ресурсів на створення якісних наборів даних.

## 1.9 Особливості розмітки даних для задач детекції об'єктів

Детекція об'єктів — це техніка комп'ютерного зору, яка передбачає виявлення та локалізацію об'єктів на зображенні чи відео. Метою детекції об'єктів є ідентифікація присутності об'єктів на зображенні чи відео та визначення їхнього просторового розташування та розміру в межах зображення. Якість розмітки відіграє вирішальну роль у детекції об'єктів, оскільки вони надають розмічені дані для навчання моделей детекції об'єктів. Точна розмітка зображень допомагає забезпечити якість і точність моделі, дозволяючи їй точно ідентифікувати та локалізувати об'єкти.

Розмітка детекції об'єктів – це систематичний процес, необхідний для навчання моделей комп'ютерного зору, що складається з декількох ключових етапів, які включають в себе ідентифікацію та обведення контурів об'єктів, що цікавлять, за допомогою прямокутних рамок, полігонів або інших фігур. Цей етап часто виконується вручну, але може бути частково автоматизований за допомогою попередньо навчальних моделей або інструментів напівавтоматичної розмітки, які прискорюють процес та зменшують кількість помилок. Після розмітки об'єктів проводиться перевірка якості. Цей етап включає в себе ретельну перевірку розмічених даних для виявлення та виправлення можливих помилок. Висока якість розмітки є критично важливою для успіху моделей комп'ютерного зору, оскільки помилки на цьому етапі можуть значно вплинути на результати навчання [103-105]. Приклад розміченого зображення наведено на рисунку 1.14 [103].

Після успішного навчання модель детекції об'єктів може бути застосована до нових зображень, які модель раніше не бачила, для ідентифікації та локалізації об'єктів у межах вивчених класів. Цей покроковий процес є основою для створення надійних і ефективних систем виявлення об'єктів у різних сферах застосування, починаючи від спостереження і закінчуючи автономними транспортними засобами.



Рисунок 1.14 – Приклад розміченого зображення з різними класами [103]

Однією з причин, чому розмітка даних для задачі детекції об'єктів є дорожчою, ніж для задачі класифікації об'єктів, є те, що детектори потребують більше інформації з кожного зображення. Для прикладу, у задачі класифікації об'єктів необхідно лише вказати, до якої категорії належить об'єкт на зображенні. У задачі ж детекції об'єктів потрібно вказати координати та розміри області, де знаходиться об'єкт. В результаті цього популярні набори даних для задачі детекції об'єктів таких як PASCAL VOC (12,000 зображень та 20 категорій) [45] та MS COCO (120,000 зображень та 80 категорій) [46], що значно менше за набір даних ImageNet [1] (1,400,000 зображень та 1,000 категорій), поява якого дала значний поштовх в розвитку сучасних методів комп'ютерного зору на основі методів глибокого машинного навчання.

Розмітка зображень полягає в позначенні зображень мітками, щоб дати моделі інформацію про зображення. Кількість міток на зображенні залежить від задачі. Для задачі детекції об'єктів може бути необхідно помітити кілька об'єктів з унікальними мітками. Вартість розмітки даних для задачі детекції об'єктів залежить від різних факторів, таких як кількість об'єктів на зображенні, кількість унікальних категорій, розмірів об'єктів, наявності перекриття об'єктів та кількості людей, що будуть розмічати дані. Середня вартість одного зображення розміченого двома людьми становить \$2 [106; 107], що призводить до мільйонних витрат для розмітки наборів даних, подібних MS COCO [46], через відсутність широкої автоматизації процесу розмітки даних.

Одним із напрямків щодо зменшення вартості розмітки даних для задачі детекції об'єктів є розробка високоефективних процесів розмітки даних [108; 109].

### 1.10 Постановка задачі

Для постановки задачі прискорення розмітки об'єктів на відео використаємо теоретико-множинний підхід. Розглянемо відео  $V$ , яке складається з послідовності  $N$  кадрів:

$$V = \{v_1, v_2, v_3, \dots, v_N\}, \quad (1.1)$$

де  $v_i$  — окремий кадр відео.

Нехай необхідно розмітити  $K$  різних типів об'єктів, яким присвоєно індекс в множині класів  $C = 1 \dots K$ . Припустимо, що на кожному кадрі  $v_i$  розмічається  $M_i$  об'єктів.

Тоді завдання розмітки передбачає позначення на кожному кадрі  $v_i$  координат об'єктів  $O_{ij}, i = 1, \dots, N, j = 1, \dots, M_i$  та ідентифікатору класу  $c_{ij}$ , до якого відноситься позначений об'єкт. Таким чином, можна визначити множину об'єктів  $O_{ij}$  на кожному кадрі:

$$O_{ij} = \{(x_{ij}, y_{ij}, w_{ij}, h_{ij}, c_{ij}) \mid x_{ij}, y_{ij}, w_{ij}, h_{ij} \in [0; 1], c_{ij} \in C\} \quad (1.2)$$



де  $x_{ij}, y_{ij}$  — відносні координати верхнього лівого кута об'єкта,  $w_{ij}, h_{ij}$  — відносні ширина та висота прямокутника, що визначає межі розміченого об'єкта,  $c_{ij}$  — клас об'єкта.

Позначимо  $O$  множину всіх об'єктів на всіх кадрах відео:

$$O = \{O_{ij} | i = 1 \dots N, j = 1 \dots M\}. \quad (1.3)$$

Тоді задача розмітки відео  $V$  полягає в розмітці розмітчиком (людиною) усіх об'єктів  $O$ . Припустимо, що розмітка відео  $V$  розмітчиком (людиною) займає  $T_{\text{розм}}$  часу. Ціль автоматизації процесу розмітки відеоданих полягає в мінімізації часу розмітки:

$$T_{\text{розм}} \rightarrow \min. \quad (1.4)$$

Для автоматизації розмітки відео використовується метод  $F$ :

$$F: V \rightarrow O', \quad (1.5)$$

який виконує відображення відео  $V$  на певний набір об'єктів  $O'$ . Оскільки якість результуючих об'єктів  $O'$  залежить від використовуваного методу автоматизації  $F$ , який може мати не ідеальну точність знаходження об'єктів, то відповідно множина  $O'$  може мати такі помилки:

- пропущений об'єкт на певному кадрі;
- знайдено та розмічено об'єкт, якого не існує (хибно позитивне спрацювання);
- знайдено та вірно розмічено об'єкт, але координати об'єкта не відповідають очікуваній точності.

Відповідно відео, які розмічені за допомогою методів автоматизації, потребують виправлення (дорозмітки) розмітчиком, що складає  $T_{\text{дорозмітки}} \leq T_{\text{розм}}$  часу. Отже, задачею даного дисертаційного дослідження є мінімізація часу  $T_{\text{дорозмітки}}$ , що можна досягнути шляхом оптимізації параметрів  $\theta$  методу автоматизації  $F$ :

$$\theta_{\min} = \underset{\theta}{\operatorname{argmin}} L(F_{\theta}(V), O), \quad (1.6)$$

де  $L$  — функція схожості двох наборів об'єктів.

В рамках даної роботи як функцію схожості використано функцію mAP задачі детекції об'єктів.

Оскільки запропонований в рамках даного дисертаційного дослідження метод автоматизації розмітки даних – це багатокomпонентне рішення, то методом оптимізації обрано жадібний підхід, який виходить з припущення, що локально оптимальні гіперпараметри методу будуть глобально оптимальними, та оптимізує параметри кожного компонента окремо.

Виконавши аналіз наявних інструментів розмітки зображень та відеоданих були виявлені наступні технічні недоліки, які обмежують їх ефективність:

- використання лише одного методу навчання, активного або нульового, звужує спектр задач, які можуть бути оброблені цими інструментами, що знижує їх універсальність;

- брак методів для прискорення моделей активного навчання подовжує цикл навчання та затримує досягнення ними достатньо високої точності;

- рівномірний вибір кадрів у відео не враховує динамічні зміни сцени, що знижує точність автоматизованої розмітки;

- відсутність оптимальних методів до інтеграції VLM та LLM моделей у процес розмітку даних знижує точність кінцевих методів.

Відповідно до поставленої оптимізаційної задачі та враховуючи вищенаведені невирішені проблеми в рамках даного дослідження необхідно вирішити такі завдання:

- розробити метод до навчання нейронних мереж, що підвищує точність методів розпізнавання по відео;

- розробити метод до зменшення обсягу оброблених кадрів для методів комп'ютерного зору;

- розробити метод по використанню візуально-мовних моделей для підвищення точності задач, які вирішують програмні засоби;

- дослідити сучасні програмні засоби для розмітки зображень та відеоданих, проаналізувати їх архітектури;

- дослідити процеси розмітки зображень та відеоданих, методи до їх пришвидшення;

- створити дуальну архітектуру автоматизованої розмітки даних за допомогою різних методів комп'ютерного зору, що прискорить процес розмітки даних для пришвидшення розмітки даних;

- виконати програмну реалізацію та експериментальне дослідження характеристик

Кожний з розроблених в дисертації методів вирішує задачу оптимізації гіперпараметрів методу автоматизації з певного боку. Метод пріоритезації розмітки відеоданих підвищує якість датасету, що використовується для тренування нейромережі, і тим сприяє підвищенню точності детекції об'єктів в окремому кадрі відео. Ітеративний метод відбору ключових кадрів підвищує якість узагальнення змісту відео і тим самим підвищує точність вибору кадрів на відео для детекції об'єктів у порівнянні з рівномірним вибором кадрів. Метод Attr4Vis підвищує якість використання візуально-мовних моделей для детекції об'єктів. Метод об'єднання результатів активного та нульового навчання збільшує точність детекції за рахунок використання більш якісної на поточному етапі розмітки моделі.

### **1.11 Висновки до розділу**

В даному розділі розглянуто основні аспекти детекції об'єктів, розмітки даних для задач комп'ютерного зору та програмні засоби для розмітки відеоданих.

Було оглянуто загальні підходи до детекції об'єктів, включаючи детальний огляд широко відомих методів детекції об'єктів, таких як R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN, Cascade R-CNN, SSD та YOLO. Порівняно переваги та недоліки методів. Розглянуто підходи до детекції об'єктів на відео та як відбувається масштабування вищезгаданих методів від задачі детекції на зображенні до детекції на відео через

детекцію об'єктів на окремих кадрах та відслідковування об'єктів через фільтр Калмана або на базі алгоритмів підрахунку оптичного потоку.

Описано процес розмітки даних та її значення для ефективного вирішення завдань машинного навчання. Виявлено, що розмітка даних займає багато часу та зусиль, тому актуальною є необхідність в автоматизації та пришвидшенні процесу розмітки даних. Було наведено особливості розмітки даних для задач детекції об'єктів. Також було виконано постановку задач дослідження.

Отже, цей розділ надав огляд фундаментальних принципів та програмних засобів розмітки даних для задач комп'ютерного зору, зокрема детекції об'єктів.

## 2 МЕТОД ПРІОРИТЕЗАЦІЇ СКЛАДНИХ ЗРАЗКІВ ДЛЯ НАВЧАННЯ НЕЙРОННИХ МЕРЕЖ

### 2.1 Методи глибокого навчання для просторово-часового моделювання

Протягом останнього десятиліття було досягнуто суттєвого прогресу в розвитку методів глибокого навчання, що сприяло значному покращенню якості розв'язання задач комп'ютерного зору. Це стало особливо помітним у контексті результатів еталонних тестів класифікації зображень, таких як ILSVRC'2012 [1], а також інших загальнодоступних наборів даних [2]. Досягнення стали можливими завдяки швидкому розвитку глибоких нейронних мереж, зокрема згорткових нейронних мереж (CNN) і архітектур на базі Transformer, експоненційному збільшенню обчислювальних можливостей сучасного апаратного забезпечення, зокрема графічних процесорів, а також широкій доступності великих анотованих наборів даних, таких як ImageNet.

Згорткові нейронні мережі тривалий час залишалися домінуючим підходом у задачах візуального розпізнавання [3–5]. Мережі, такі як P3D [55], вперше продемонстрували можливість використання 3D-згорток для моделювання просторово-часових залежностей, запропонувавши архітектуру з 11 шарами. Пізніше I3D [57] показала, що ініціалізація 3D-згорток попередньо навченими вагами мережі Inception дає високу точність на наборах даних, таких як Kinetics. Роботи, зокрема S3D [56] і MVFNet [110], продемонстрували, що декомпозиція просторових і часових аспектів дозволяє досягти значного компромісу між швидкістю роботи моделі та її точністю.

Попри прогрес, використання 3D-згорток має свої обмеження, зокрема через мале рецептивне поле, що обмежує можливість ефективного моделювання довгострокових залежностей у відеоданих. Крім того, пряме заміщення 2D-згорток на 3D-аналогічні суттєво збільшує обчислювальні витрати [111]. Щоб подолати ці обмеження, почали з'являтися методи факторизації, які розділяють часові та просторові аспекти згорток, що

дозволило зменшити складність моделей і зробити їх більш придатними до застосування на практиці [55; 56].

Іншим важливим напрямком розвитку CNN стало використання механізмів уваги, що дозволяють краще моделювати взаємозв'язки між різними частинами зображення. Першою мережею, яка успішно використала механізм уваги, стала NLNet [112], яка показала, як ефективно виявляти просторові залежності між пікселями. Подальший розвиток у цій сфері був представлений GCNet [113], який спростив складний блок нелінійної уваги NLNet, замінивши його легшим і менш ресурсомістким блоком глобального контексту. Цей підхід знизив обчислювальні вимоги без втрати точності. DNL [114] запропонував ще більш гнучкий підхід, що дозволяє враховувати різні контексти для окремих пікселів у межах спільного глобального контексту, що значно покращило ефективність.

У сучасному комп'ютерному зорі дедалі частіше використовують архітектури нейромережі на основі Transformers, які зарекомендували себе як надзвичайно ефективні для задач розпізнавання зображень [62, 64, 115]. Vision Transformer (ViT) [62] стала однією з перших моделей, яка застосувала трансформери для аналізу зображень. У цій архітектурі зображення розбивається на неперекривні частини (patches), що надходять на вхід стандартного Transformer-енкодера [61]. Результати, отримані ViT на наборі даних ImageNet, перевершили показники точності моделей CNN, що мотивувало подальші дослідження.

Трансформери довели свою ефективність також у задачах відеоаналізу. Наприклад, TimeSformer [116] застосував просторово-часовий механізм уваги, що дало змогу досягти кращого балансу між швидкістю та точністю. ViViT [65] продовжив розвиток цього підходу, запропонувавши факторизацію уваги для просторових і часових компонентів, що забезпечило високу продуктивність на наборах даних, таких як Kinetics. MViT [67] розширив цю архітектуру до мультимасштабної, що підвищило продуктивність.

Нещодавно було представлено Video Swin Transformer [66] як просторово-часову адаптацію Swin Transformer [64], який є основою комп'ютерного зору загального призначення для розуміння зображення. Він базується на тій самій ієрархічній структурі та розширює механізм локальної уваги з просторової до просторово-часової модальності. Механізм уваги обчислюється на частинках відео, що не перекриваються, шляхом застосування методу ковзного вікна Swin Transformer, адаптованого для моделювання часової модальності.

На рисунку 2.1 зображено загальну архітектуру Video Swin Transformer [66] на прикладі його мініатюрної версії (Swin-T). Він приймає на вхід  $T$  кадрів, кожен з яких має розмір  $H \times W \times 3$  пікселів, що формує тензор з  $T \times H \times W \times 3$  елементів у загальному розмірі вхідних даних. Після цього вхідні дані розділяються на частини розміром  $2 \times 4 \times 4 \times 3$  елементи, які не перетинаються, та розглядаються як вхідні токени. Такий підхід перетворює вхідне зображення на послідовність  $T/2 \times H/4 \times W/4$  3D-токенів. Кожен токен представлений 96-вимірним вектором репрезентацій. Лінійний шар використовується для проєктування цих 96-вимірних векторів даних у вектор довільного вхідного розміру  $C$ .

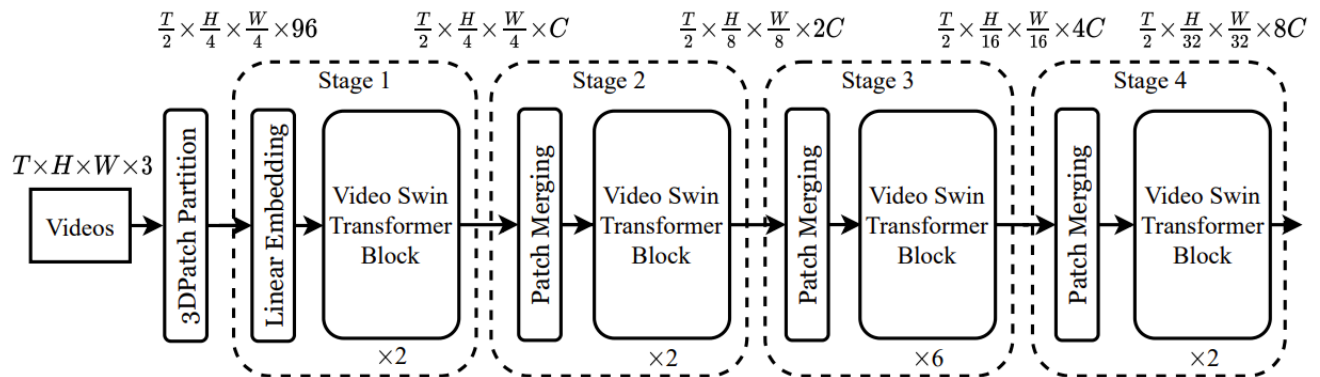


Рисунок 2.1 – Загальне представлення основних кроків методу Video Swin Transformer (версія Swin-T) [66]

У Video Swin Transformer [66] вхідні тензори по часовому виміру не зменшують, оскільки це може призвести до зменшення продуктивності, як показано в роботах [117; 118]. Це дозволяє дотримуватися архітектури та принципу роботи оригінального Swin Transformer [64], який зменшує просторовий розмір вхідних токенів наполовину після кожного етапу шляхом групування сусідніх патчів  $2 \times 2$  в один вектор репрезентацій і зменшення розміру зображення наполовину за допомогою шару лінійної проєкції. Наприклад, 8С-вимірні зображення після групування будуть спроектовані на 4С-вимірні, які утворюють вхідні маркери для наступного етапу Swin Transformer [64].

Ключовою відмінністю між Video Swin Transformer [66] та оригінальним Swin Transformer [70] є заміна мультиголового модулю уваги (MSA) мультиголовим модулем уваги зі зміщеним 3D вікном, як показано на рисунку 2.2.

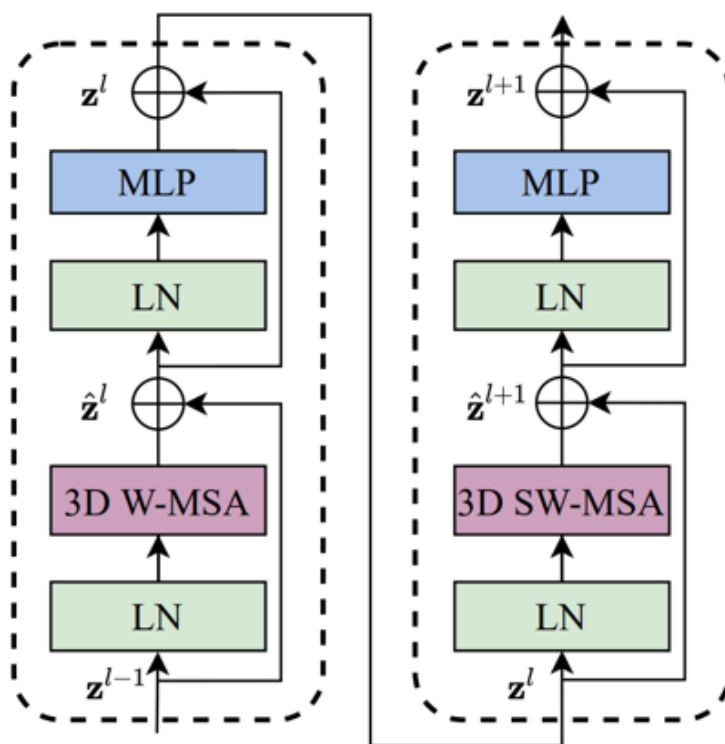


Рисунок 2.2 – Представлення основного блоку методу Video Swin Transformer [66]



Попри зосередженість науковців на вдосконаленні архітектур, важливим напрямком досліджень залишаються процедури навчання. Ефективні методи оптимізації дозволяють досягти кращого компромісу між швидкістю роботи моделей і їхньою точністю, а також підвищити стійкість до змін у вхідних даних [119; 120].

У цій дисертації зроблено акцент на вдосконаленні процедур навчання, а не лише модифікації архітектур, що може сприяти покращенню продуктивності в задачах розпізнавання відео. Саме тому, запропоновано новий підхід до навчання нейронних мереж з моделюванням темпоральної модальності, заснований на ідеї аналізу складних негативних зразків [121; 122], який вибирає найскладніші зразки даних, які не вірно класифіковані методом машинного навчання, для подальшого навчання, що найбільше сприяє підвищенню точності моделі.

Запропонований підхід підвищує точність навіть найліпших моделей на наборі даних Kinetics-400 [57] порівняно із класичним підходом до тренування нейронних мереж. На час проведення дослідження було досягнуто передовий результат на наборі даних Kinetics-400 (84,4% Top-1 і 96,5% для Top-5 точності з використанням нейронної мережі Video Swin Transformer). Наведені у розділі результати були опубліковані у публікації [123] та презентовані на міжнародній конференції Intelligent Systems Conference 2022 (IntelliSys 2022).

## **2.2 Майнінг складних негативних зразків**

Складними негативними зразками в контексті даної дисертації називаються зразки, які хибно позитивно та хибно негативно класифіковані нейронною мережею та мають найбільше значення функції втрат.

Загальна ідея майнінгу складних негативних зразків базується на простій ідеї оптимізації точності методів машинного навчання лише на найскладніших зразках із набору даних або пакету в офлайн- чи онлайн-реалізації відповідно [124]. Цей процес зображено на рисунку 2.3 [125].

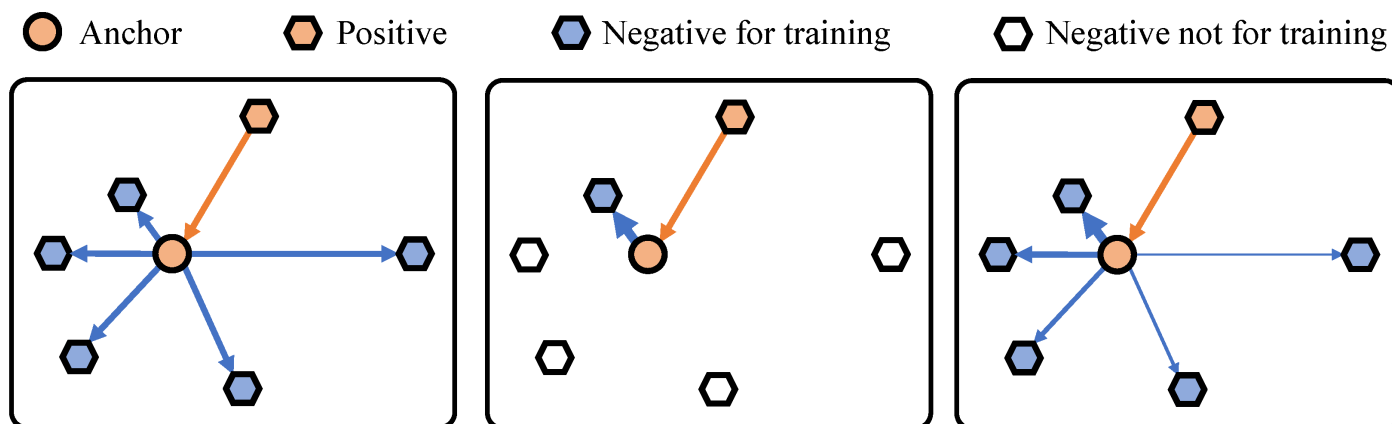


Рисунок 2.3 – Процес майнінгу складних негативних зразків [125]

З іншого боку, фокальна функція втрат дозволяє зосередитися методам машинного навчання на неправильно класифікованих зразках з високою впевненістю та зменшити вплив правильно класифікованих елементів під час оптимізації [122].

Обидва підходи [122; 124] припускають, що навчальний набір даних ідеальний і в ньому немає помилок розмітки. Це припущення не справедливе для великих наборів даних. У цьому науковому дослідженні запропоновано підхід майнінгу складних зразків, який не вимагає розмітки, а отже, не схильний до помилок в розмітці даних.

Незважаючи на значний прогрес, досягнутий у підвищенні точності моделей для різних завдань комп'ютерного зору, таких як детекція об'єктів, процеси навчання нейромережі для задач на відео продовжують використовувати класичний підхід до навчання, який оптимізує функцію втрат на базі бінарної кросентропії. У [122] було доведено, що це може призвести до недостатнього використання даних мережею, яка замість того, щоб зосереджуватися на неправильно класифікованих зразках з низькою впевненістю, починає оптимізувати переважно зразки, які вже успішно класифікуються. Візуалізація деталей цього процесу зображена на рисунку 2.4 [122].

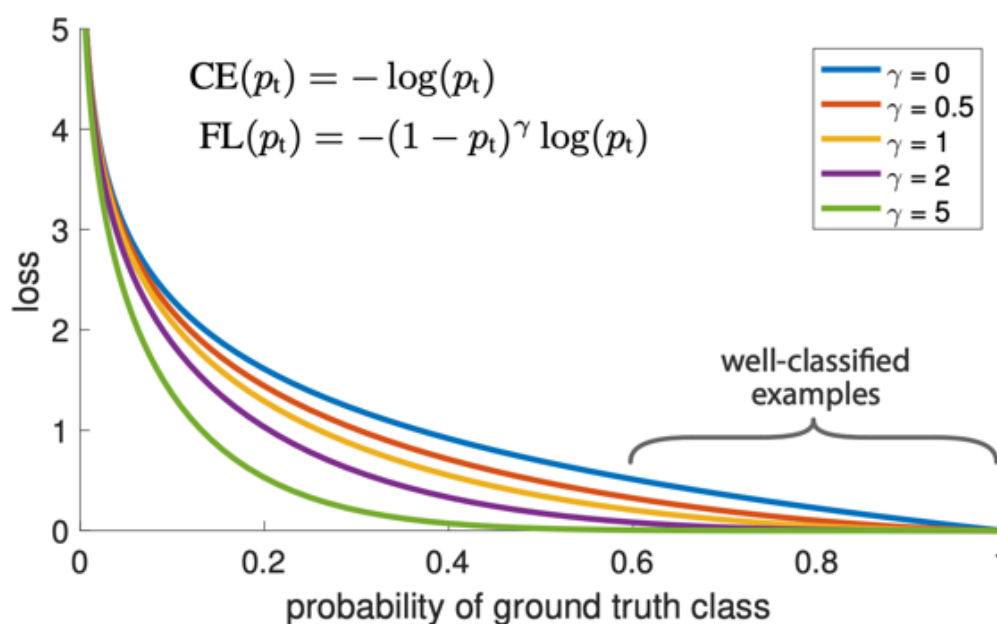


Рисунок 2.4 – Фокальна функція втрат, яка дозволяє зменшити вагу даних, які достатньо точно класифікуються нейронною мережею [122]

Схожу проблему можна спостерігати в задачах класифікації на відео, де переважає велика кількість простих для класифікації зразків, і дослідникам потрібно знайти спосіб постійно підвищувати якість методів машинного навчання на складних зразках. З практичного досвіду слідує, що фокальна функція втрат не показує більш точні результати у завданнях на відео порівняно з функцією втрат, яка базується на мінімізації кросентропії, тому потрібно знайти інший підхід до враховування інформації про складні зразки в процесі навчання нейромережі.

### 2.3 Метод пріоритезації складних зразків для навчання нейронних мереж

Загальний підхід базується на ідеї пошуку складних негативних зразків [122; 124], яка відбирає найскладніші зразки для навчання моделі, проте підхід, який розроблений в цьому дисертаційному дослідженні, пропонує використовувати відстань до кластерів даних, які можна знайти способом, що не вимагає розмітки даних, замість відбору зразків за функцією втрат.

Припустимо, що відомою є кількість кластерів  $K$  у даних, на які їх розбила нейромережа. Тоді можливо кластеризувати всі дані в ці  $K$  кластерів за допомогою методу K-means. Після цього стає можливим замінити  $T$  зразків вибірки даних, які знаходяться найближче до центрів  $K$  кластерів, на зразки, що знаходяться якнайдалі від них. Таким чином було виконано оверсемплінг складних зразків, як і у [122; 124], і замінено найпростіші зразки для навчання, як у [122].

На рівні псевдокоду запропонований підхід можна сформулювати як Лістинг 2.1.

---

**Лістинг 2.1.** Метод пріоритезації складних зразків для навчання нейронної мережі

---

**Inputs:** the set of videos  $V = \{V_1, V_2, \dots, V_n\}$ , the set of corresponding labels for each video  $L = \{L_1, L_2, \dots, L_n\}$ , that together form the dataset  $D = \{V, L\}$ , the number of training epochs  $E_{train}$ , the number of Hard Samples  $T$  selected during each epoch, pretrained model  $M$ .

---

For each epoch in  $E_{train}$ :

1. Select the number of clusters  $K$ . // *Fixed number or searched for each epoch*
  2. Cluster embeddings of data points from  $D$  to  $K$  clusters using K-means algorithm.
  3. Select  $T$  samples with the shortest distance from stage 2.
  4. Select  $T$  samples with the largest distance from stage 2.
  5. Replace samples obtained in stage 3 with samples obtained in stage 4. // *Oversampling*
  6. Train model  $M$  on dataset obtained in stage 5 for one epoch.
- 

**Outputs:** finetuned on dataset  $D = \{V, L\}$  model  $M$ .

---

Він починається з чітко визначених вхідних параметрів: набору відео  $V$ , відповідної розмітки відео  $L$ , кількості епох тренування  $E_{train}$  та кількості вибраних зразків  $T$  під час кожної епохи.

Зовнішній цикл методу виконується  $E_{train}$  разів, де  $E_{train}$  є фіксованим позитивним цілим числом. Це означає, що цикл завершиться після заданої кількості ітерацій.

Для вибору кількості кластерів  $K$  рекомендовано використовувати метод ліктя, запропонованого в [126], який заснований на ітераційному запуску методу K-means, який обирає таку кількість кластерів після якої загальна відстань між усіма зразками в наборі даних і кластерами починає зменшуватися надто повільно порівняно з попередніми ітераціями. Зауважимо, що цю процедуру можна ефективно виконати за допомогою трійкового пошуку, щоб зменшити загальне навантаження на обчислення. Метод ліктя передбачає запуск методу K-means для різних значень  $K$  (від 1 до заданого  $K_{max}$ , який рівний кількості категорій в наборі даних). Оскільки  $K_{max}$  є фіксованим числом, кількість ітерацій обмежена кількістю ітерацій методу ліктя  $R$ . Методу K-means має властивість збіжності за скінченну кількість ітерацій, що забезпечує обчислюваність методу ліктя за скінченну кількість кроків.

На другому кроці виконується кластеризація векторів ознак за допомогою методу K-means. Метод K-means збігається за скінченну кількість ітерацій, оскільки він або знаходить стабільні центроїди, або досягає максимального числа ітерацій, заданого наперед. Кількість кластерів  $K$ , кількість даних  $n$ , кількість ітерацій  $i$  та розмірність векторів даних  $d$  є фіксованими, що гарантує обчислюваність цього кроку.

Далі запропонований метод вибирає  $T$  зразків з найменшою і найбільшою відстанню до центроїдів кластерів. Сорткування відстаней і вибір зразків є детермінованими операціями зі скінченною кількістю кроків, що завжди завершується, оскільки  $T$  і кількість даних  $n$  є фіксованими. Заміна вибраних зразків (Oversampling) є прямою операцією зі складністю  $O(T)$ , яка завжди завершується за скінченну кількість кроків, що гарантує обчислюваність цього кроку.

Останнім кроком кожної ітерації є навчання моделі на сформованому наборі даних. Тренування здійснюється за фіксовану кількість кроків, що обмежена кількістю

даних  $n$ . Цей процес завжди завершується, оскільки кількість кроків тренування є скінченною.

Загальна складність методу включає всі описані кроки. Вибір кількості кластерів методом ліктя додає складність  $O(R \cdot K_{max} \cdot n \cdot i \cdot d)$ . Кластеризація після вибору  $K$  має складність  $O(K_{max} \cdot n \cdot i \cdot d)$ . Вибір  $i$  заміна зразків додають  $O(n \cdot \log_2 n + T)$ , а тренування моделі додає  $O(n \cdot M)$ , де  $M$  – складність моделі машинного навчання. Оскільки всі ці кроки виконуються  $E_{train}$  разів, загальна складність методу є  $O(E_{train} \times (R \cdot K_{max} \cdot n \cdot i \cdot d + K_{max} \cdot n \cdot i \cdot d + n \cdot \log_2 n + T + n))$ .

Оскільки всі кроки методу виконуються за скінченну кількість часу без нескінченних циклів або інших перешкод, що перешкоджають його завершенню, то алгоритм, наведений у лістингу 2.1, завжди буде завершуватися за скінченну кількість кроків і є обчислюваним.

Варто зазначити, що згідно з емпіричними спостереженнями число  $K$  можна встановити як кількість категорій у навчальному наборі даних, яка є наближенням до загальної кількості кластерів у наборі даних.

## 2.4 Експериментальні дослідження запропонованого підходу до вибору складних зразків

Усі експерименти проводилися з використанням фреймворку MMAction2 [127] і публічного репозиторія для реалізації Video Swin Transformer [66].

Під час навчання було суворо дотримано процедуру навчання, запропоновану в роботі [66]. Модель була ініціалізована із попередньо навченими вагами для нейронної мережі. Класифікатор був ініціалізований випадковими вагами з нормальним розподілом з нульовим середнім та відхиленням 0,01. Нейронна мережа була оптимізована за допомогою оптимізаційного методу AdamW [128] з початковою швидкістю навчання (learning rate)  $3 \cdot 10^{-4}$  для 30 епох, включаючи 2,5 епохи фази розігріву. Під час навчання використовується розмір вхідного набору даних 64.

Відповідно до [64], зростаючий ступінь стохастичного відкидання глибини [129] і стохастичного відкидання ваг [130] використовується для більших моделей, тобто 0,1, 0,2, 0,3 стохастичної глибини та 0,02, 0,02, 0,05 стохастичного відкидання ваг для Swin-T, Swin-S і Swin-B відповідно. Процес стохастичного викидання глибини наведено на рисунку 2.5 [131].

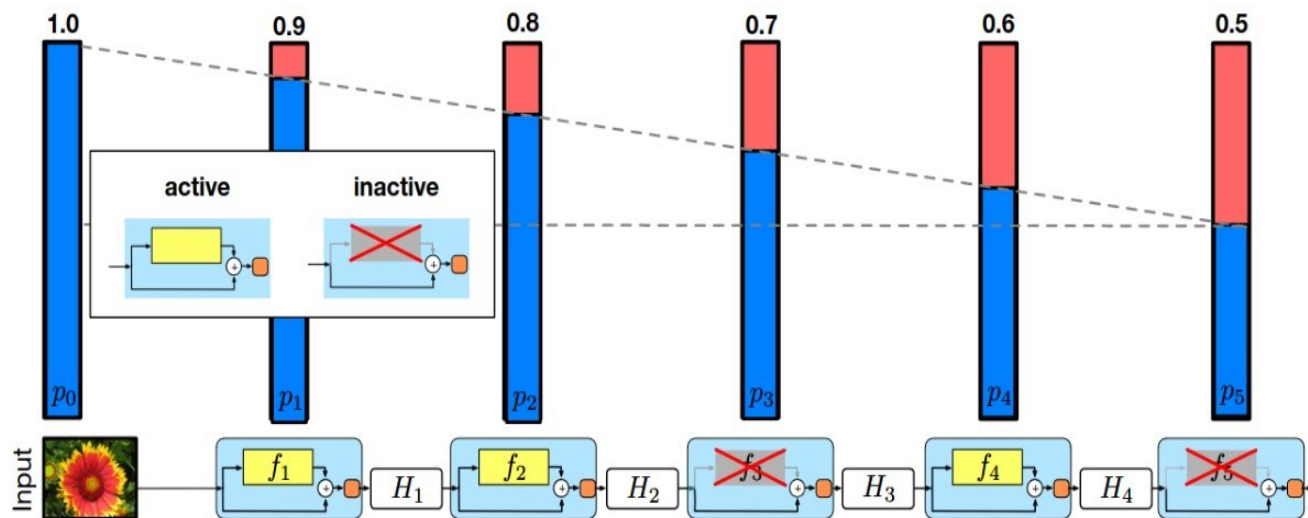


Рисунок 2.5 – Представлення основних кроків стохастичного відкидання глибини [131]

Слідуючи установленим практикам з [66] градієнти для ваг з попередньо навченої нейронної мережі були помножені на коефіцієнт  $1/10$ , що підвищує точність.

У всіх експериментах вхідні дані для моделі формуються шляхом вибірки 32-кадрового кліпу з кроком 2 і вхідною роздільною здатністю  $224 \times 224$ , що призводить до  $16 \times 56 \times 56$  вхідних токенів, і використовували параметр семплювання  $T = 0,1n$ , тобто 10% від розміру набору даних. Під час продукування результатів нейромережі, дотримуючись протоколів, запроваджених у публікації [65], було використано вікно розміром  $4 \times 3$  для обчислення оцінки роботи нейронної мережі на відео шляхом усереднення прогнозів для кожного вікна. У цьому налаштуванні відео рівномірно розділяється на 4 кліпи за часовим виміром, і для кожного кліпу виконується нарізка на 3 частини за найдовшим виміром, щоб повністю охопити відео.

У таблиці 2.1 представлено порівняння підходів на наборі даних Kinetics-400 [57].

Таблиця 2.1 – Порівняння методів аналізу відео на наборі даних Kinetics-400 (HS = Hard Samples, FL = Focal loss, HM = Hard Negative mining)

Метод	Набір даних попереднього навчання	Top -1	Top -5	Обчислювальна складність, GFLOPS	Кількість параметрів , мільйони
MVFNet [110]	-	72,0	90,0	75	61,8
I3D [57]	ImageNet-1K	72,1	90,3	108	25,0
NL I3D-101 [112]	ImageNet-1K	77,7	93,3	359	61,8
SlowFast R101+NL [120]	-	79,8	93,9	234	59,9
MViT-B, 64x3 [67]	-	81,2	95,1	455	36,6
ViT-B-VTN [62]	ImageNet-21K	78,6	93,7	4218	11,04
ViViT-L/16x2 320 [65]	ImageNet-21K	81,3	94,7	3992	310,8
Swin-B [66]	ImageNet-21K	82,7	95,5	282	88,1
Swin-L [66]	ImageNet-21K	83,1	95,9	604	197,0
Swin-B + FL	ImageNet-21K	82,8	95,5	282	88,1
Swin-L +FL	ImageNet-21K	83,2	96,0	604	197,0
Swin-B + HM	ImageNet-21K	82,3	95,0	282	88,1
Swin-L +HM	ImageNet-21K	82,5	95,2	604	197,0
Swin-B + HS	ImageNet-21K	83,5	96,0	282	88,1
<b>(Запропонований)</b>					
Swin-L +HS	ImageNet-21K	<b>84,4</b>	<b>96,5</b>	604	197,0
<b>(Запропонований)</b>					

Як показано, запропонований підхід зі зміненою процедурою навчання збільшує точність Top-1 на 1,3% та 0,8% для Swin-B і Swin-L відповідно. Крім того, варто



відзначити, що модель Swin-B з удосконаленою процедурою навчання перевершує Swin-L з класичним підходом до навчання нейронних мереж (83,5% проти 83,1% точності Top-1 відповідно для Swin-B). Це означає, що підходи до тренування моделей на основі методів машинного навчання все ще відіграють важливу роль у навчанні моделей на базі архітектури Transformer. Отримані результати узгоджуються з результатом дослідження, представленого у публікації [120].

Запропонований метод вимагає пошуку кількості кластерів даних у просторі великої розмірності, що може бути обчислювально неможливим для деяких задач і наборів даних. Було проведено додаткове дослідження впливу на точність класифікації моделлю машинного навчання від вибору кількості кластерів, встановивши кількість кластерів із фіксованим числом 50%, 75%, 100%, 125% і 150% до кількості категорій у наборі даних Kinetics-400. Крім того, було створено власну підмножину Kinetics лише зі 100 категоріями для вивчення процесу відбору на менших наборах даних. Результати представлені в таблиці 2.2.

Таблиця 2.2 – Порівняння впливу параметра кількості кластерів в методі вибору складних зразків за пріоритетом на точність класифікації моделлю Swin-B + HS

Кількість категорій ( $K$ )	TOP-1 при кількості кластерів					
	0,5 $K$	0,75 $K$	1 $K$	1,25 $K$	1,5 $K$	Метод ліктя
100	85,3%	85,6%	85,5%	85,3%	84,8%	85,8%
400	83,2%	83,4%	83,4%	83,3%	83,0%	83,5%

Як можна побачити, встановлення кількості кластерів із фіксованим числом призводить до незначного погіршення точності класифікації у порівнянні з методом ліктя, що можна пояснити додатковою рандомізацією, створеною змінною кількістю кластерів, яка змінюється від епохи до епохи. Таким чином, можна рекомендувати

встановити кількість кластерів на фіксовану кількість категорій, яка дорівнює кількості категорій у наборі даних, якщо метод ліктя незастосовний до певного набору даних.

Запропонований метод вимагає встановити кількість зображень з набору даних, які будуть видалені та замінені складними екземплярами. Було вивчено важливість видалення простих екземплярів і вплив різних пропорцій оверсемплінга складних екземплярів на результати моделей на основі методів машинного навчання. Результати дослідження представлені в таблиці 2.3. Як можна побачити, оверсемплінг складних екземплярів без видалення простих призводить до зниження точності навчених нейронних мереж, а отже підкреслює важливість заміни простих екземплярів складними.

Таблиця 2.3 – Порівняння різних стратегій оверсемплінга та видалення складних негативних зразків на точність моделей машинного навчання

Відсоток оверсемплінга складних зразків	Відсоток видалених простих зразків	Точність Top-1
0%	0%	82,7%
10%	0%	83,3%
0%	10%	82,5%
10%	10%	83,5%
20%	10%	83,3%
20%	20%	83,5%

## 2.5 Висновки до розділу

В даному розділі представлено новий метод навчання моделей машинного навчання, який використовує пріоритезацію складних екземплярів даних для навчання нейронних мереж, що призводить до підвищення точності моделі, та не вимагає додаткової розмітки. Запропонований метод опирається на відстань між вивченими центроїдами кластерів та кожною точкою даних для вибору тих даних, які знаходяться

найдалше від центрів, що призводить до вибору більш складніших точок даних і покращує точність моделі в порівнянні з існуючими аналогами.

Оскільки відстань між кожним екземпляром та центрами кластерів можна знайти способом, що не вимагає розмітки, то це усуває залежність від якості розмітки даних під час процесу відбору, а отже його можна застосовувати на зашумлених наборах даних. Відібрані таким чином складні екземпляри під час навчання піддаються процедурі оверсеплінгу, що допомагає моделі підвищити якість на задачах розпізнавання на відео. Завдяки впровадженню запропонованого підходу в процес навчання для неймереж на базі архітектури Video Swin Transformer було отримано підвищення точності на задачі розпізнавання дій на відео на Kinetics-400 (точність 84,4% топ-1 і 96,5% топ-5 точності), що дає вищу точністю у порівнянні з оригінальним підходом до навчання Video Swin Transformer.

Наведені у розділі результати були опубліковані в публікації [123] та презентовані на міжнародній конференції Intelligent Systems Conference 2022 (IntelliSys 2022).

### 3 ІТЕРАТИВНИЙ МЕТОД ВИБОРУ КЛЮЧОВИХ КАДРІВ НА ДОВГИХ ВІДЕО

#### 3.1 Постановка задачі вибору ключових кадрів для узагальнення відео

Узагальнення змісту відео є однією з ключових задач комп'ютерного зору, оскільки передбачає створення скороченої версії відео, яка фіксує найважливішу та найрелевантнішу інформацію. Процес зображено на рисунку 3.1.

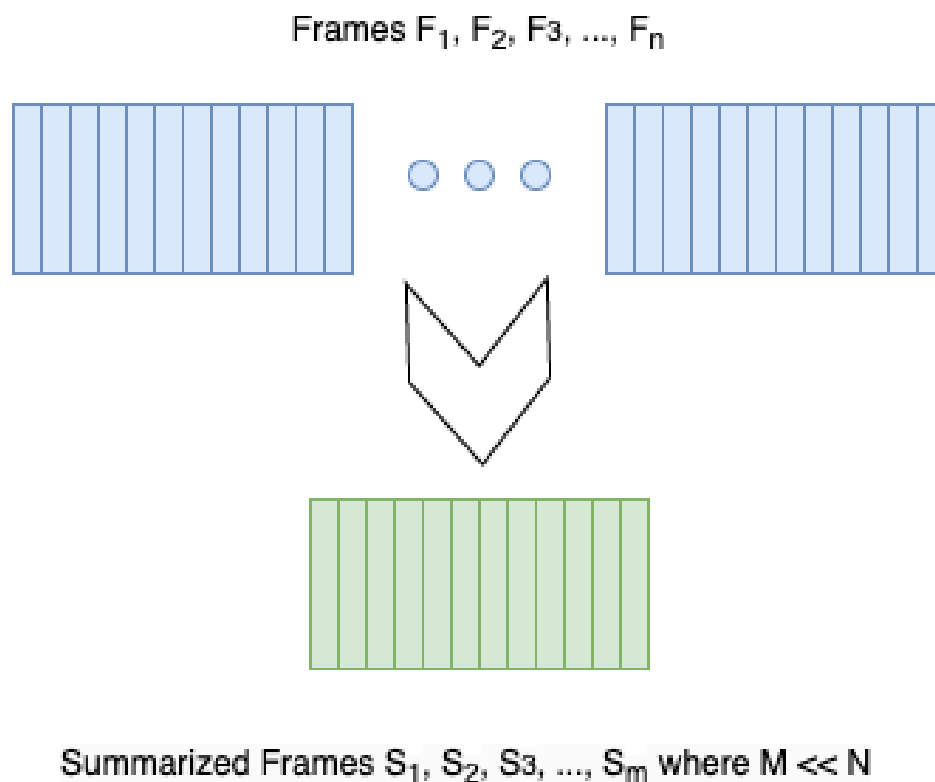


Рисунок 3.1 – Процес узагальнення змісту відео:

$N$  кадрів у відео узагальнюються до  $M$  кадрів

Є декілька причин чому задача узагальнення змісту відео є важливою у сфері комп'ютерного зору. По-перше, з експоненціальним зростанням відеоданих, доступних в мережі Інтернет [132], людям стає все важче відстежувати всі відео, які вони

переглядають або шукають для перегляду. Резюмування відео забезпечує вирішення цієї проблеми, дозволяючи користувачам отримати швидкий і стислий огляд відео, що може допомогти їм визначити чи будуть вони переглядати повне відео чи ні. По-друге, узагальнення змісту відео може підвищити ефективність пошукових систем для відео. Створюючи короткий виклад відео, пошукові системи можуть краще зіставляти пошукові запити з релевантним вмістом на відео, створюючи більш зручну для користувачів роботу [133]. По-третє, узагальнення змісту відео може підвищити доступність відеоконтенту для людей з обмеженими можливостями. Наприклад, люди з порушеннями слуху можуть використовувати підсумок відео, щоб отримати письмове резюме аудіовмісту відео. Нарешті, узагальнення змісту відео має численні практичні застосування, наприклад відеоспостереження. У цьому контексті це може допомогти співробітникам служби безпеки швидко ідентифікувати та аналізувати підозрілу діяльність на великих об'ємах відеоданих.

Загалом узагальнення змісту відео є важливим завданням у сфері комп'ютерного зору, оскільки воно може допомогти людям отримати доступ до відеовмісту, зрозуміти його та використовувати його більш зручним і ефективним способом. На додаток до переваг, згаданих вище, узагальнення змісту відео має потенціал революціонізувати спосіб споживання та взаємодії з відеовмістом [132]. Наприклад, узагальнення змісту відео можна використовувати для створення персоналізованих відео або підсумків на основі індивідуальних інтересів або вподобань. Це може дозволити людям дивитися лише найбільш релевантні та цікаві частини відео замість того, щоб переглядати все відео [133]. Узагальнення змісту відео також можна використовувати для автоматичного створення субтитрів для відео, що може зробити їх доступнішими для широкої аудиторії. Це особливо важливо для людей із вадами слуху, а також для людей, які вивчають нову мову та хочуть вдосконалити свої навички слухання та розуміння. Крім того, узагальнення змісту відео можна використовувати для створення відеоконспектів з навчальною метою. Наприклад, вчителі можуть використовувати підведення

узагальнення змісту відео, щоб створювати коротші, більш концентровані за змістом відеоролики, які охоплюють конкретні теми чи поняття, що може бути більш зручним для учнів. Потенційне застосування узагальнення змісту відео величезне, і в міру того, як технологія продовжує розвиватися, цілком ймовірно, що суспільство побачить ще більш інноваційне використання цього інструменту в області комп'ютерного зору.

Згідно з нещодавнім дослідженням літератури [132] більшість підходів до узагальнення змісту відео використовують RNN [134], LSTM [135], GRU [136] або вейвлет-перетворення [137] для моделювання часових залежностей і прогнозування важливості кадрів у відео. Однак ці підходи показали слабкі сторони, пов'язані з моделюванням довгих послідовностей і зворотного поширення сигналів. Щоб усунути ці обмеження, деякі роботи [138-140] запропонували використовувати механізми уваги для моделювання часових залежностей, що зменшує ймовірність затухання або вибуху градієнтів. Крім того, роботи [141-142] вдосконалили ці підходи шляхом моделювання часових залежностей на різних рівнях гранулярності. Існуючі рішення для узагальнення змісту відео були розроблені для вирішення проблем для конкретних наборів даних, які містять короткі послідовності. Однак реальні відеоролики, такі як відео на платформах YouTube або Facebook, як правило, мають середню тривалість, яка вимірюється десятками хвилин [144], що ускладнює застосування цих підходів. Щоб вирішити цю проблему, дослідники, як правило, аналізують довгі послідовності з малою частотою кадрів [145] або роблять численні запуски рішень на частинах відео, що можуть перекриватися [146]. Однак ці підходи мають суттєві обмеження, включаючи неточне передбачення меж узагальнення, втрату контексту, значні обчислювальні витрати та погіршення продуктивності на довгих послідовностях.

Отже, одним з завдань даного дослідження є зменшення обсягу даних для обробки, скорочення часу аналізу даних та підвищення точності обробки відеоданих для узагальнення змісту відео за рахунок знаходження ключових кадрів на довгих відео, що забезпечує точне виділення інформативних сегментів відео з мінімальними

обчислювальними витратами. Саме тому було запропоновано використання ітеративного методу до вибору ключових кадрів на довгих відео для досягнення підвищення точності розпізнавання для задачі узагальнення змісту відео, розробку якого представлено у цьому розділі.

Запропонований метод було перевірено на наборах даних TVSum [147] і SumMe [148], що показало найвищу точність на цих наборах даних – 56,2% (+0,6%) і 63,1% (+0,5%) відповідно. Такий підхід підвищує ефективність узагальнення змісту відео для реальних застосунків, які працюють з відео з платформ YouTube або Facebook, де довгі кліпи є поширеними. Результати даного дослідження було опубліковано у науковому виданні [149] та презентовано на міжнародній конференції Advances in Artificial Systems for Logistics Engineering III (ICAILE 2023).

У наступних розділах розглянуто підходи комп'ютерного зору для узагальнення змісту відео, описано чим запропонований ітеративний метод відрізняється від них, далі наведено опис методу по кроках, після цього описані експериментальні умови та наведені результати експериментального досліджень, зроблені висновки.

### **3.2 Методи комп'ютерного зору для узагальнення змісту відео**

Більшість попередніх робіт з узагальнення змісту відео використовують власну евристику для навчання без учителя. Вибираючи репрезентативні кадри для узагальнення, вони використовують оцінку важливості кадра для того, щоб визначити, які кадри є важливими або репрезентативними [150]. Ці роботи використовують навчальні дані, які складаються зі створеної людиною розміткою для кліпів. Оскільки вони можуть неявно засвоювати семантичні знання високого рівня, які використовуються людьми для створення розмітки для узагальнення змісту відео, то ці підходи до навчання з учителем часто перевершують ранні роботи, що використовували підходи без учителя [150].

Методи глибокого навчання [61; 138-143; 151-158] нещодавно стали популярними для узагальнення змісту відео, моделюючи часову залежність змінного діапазону між кадрами та навчаючись призначати пріоритет кадрам на основі розмітки. Для цього використовують CNN мережі [143] або RNN [151-153]. Їх ідея полягає в тому, щоб більш ефективно фіксувати довгострокові залежності між відеокадрами, що є важливими для розробки змістовних підсумків. Як показано у роботі [151], ця залежність може бути змодельована за допомогою двох LSTM за рахунок розгляду задачі узагальнення змісту відео як задачі структурованого прогнозування на послідовних даних.

Інші дослідження намагаються вирішити проблеми з обмеженою пам'яттю рекурентних мереж (RNN) для моделювання довгих послідовностей шляхом використання або зовнішньої пам'яті [153], або ієрархічного комбінування декількох LSTM на різних рівнях пам'яті [154]. У роботі [155] було запропоновано вбудовувати механізм уваги на основі LSTM для моделювання часових залежностей і формування нових уявлень, які дозволяють прогнозувати достатньо точне узагальнення змісту відео. У публікації [156] проблема узагальнення змісту відео розглядалась як завдання послідовного навчання та була запропонована інтеграція механізму уваги в мережу енкодера-декодера на основі LSTM. Цей механізм отримує вихідні дані енкодера та попередній прихований стан декодера та обчислює вектор із значеннями уваги, що згодом впливає на процес декодування відео.

Крім того декілька методів спрямовані на моделювання залежностей між кадрами, що використовують варіації механізму уваги, були знайдені в Transformer Networks [61]. Перший метод, опублікований у [138], поєднує в собі механізм уваги з двошаровою повністю зв'язаною мережею для регресії важливості кадру. У роботі [139] використовують ієрархічний підхід, спочатку визначаючи потенційні ключові кадри на рівні кадрів незалежно один від одного, а потім використовуючи мультиголову модель уваги, щоб оцінити їхню важливість і вибрати ключові кадри для узагальнення змісту відео. Автори публікації [157] вдосконалюють навчальний метод стандартного



механізму уваги шляхом включення етапу, який збільшує різноманітність візуального вмісту в узагальненні на відео, використовуючи обчислені значення уваги та людські розмітки. У роботі [158] представили варіацію методу з [138], яка включає додаткові представлення змісту відео, включаючи векторні репрезентації на основі CNN із рівня pool5 GoogleNet [159], навченого на наборі ImageNet, і пов'язані заміною згортки на Inflated 3D ConvNet [57], що тренувалися на наборі даних Kinetics. Ці векторні репрезентації вводяться в механізми уваги, а вихідні дані об'єднуються, щоб утворити загальний простір для представлення кадрів відео. Нарешті, представлення використовується для визначення ймовірності того, що кадр ключовий. Модель PGL-SUM [160], яка тісно пов'язана з існуючими підходами на основі уваги, фіксує залежність кадру на різних рівнях деталізації за допомогою глобальних та локальних мультиголових механізмів уваги. Крім того, на відміну від інших методів, які ігнорують послідовну природу відео, модель PGL-SUM [160] включає часову інформацію про порядковий номер кадру в відео, який є одним з ключових факторів в оцінці важливості кадру.

Деякі дослідники розробили архітектурні рішення для узагальнення змісту відео, такі як підхід на основі кластеризації, представлений у [161]. У цій роботі підхід ієрархічної кластеризації на основі графів використовується для узагальнення змісту відео шляхом вибору ключових кадрів для представлення вмісту відео. Карта вагів створюється з графа подібності кадрів, і кластери або пов'язані компоненти графа можуть бути легко виведені на мінімальному остовному дереві кадрів з картою ваг на основі ієрархічних шкал спостереження, обчислених над цим деревом.

Найближчою попередньою роботою до запропонованого у цьому дисертаційному дослідженні методу є метод часової сегментації дистильованого ядра (D-KTS), представлений у роботі [162]. D-KTS визначає межі для узагальнення змісту відео шляхом часової локалізації довгих відео. Даний процес наведено на рисунку 3.2 [162].

Input video (class: Working on a sewing project)

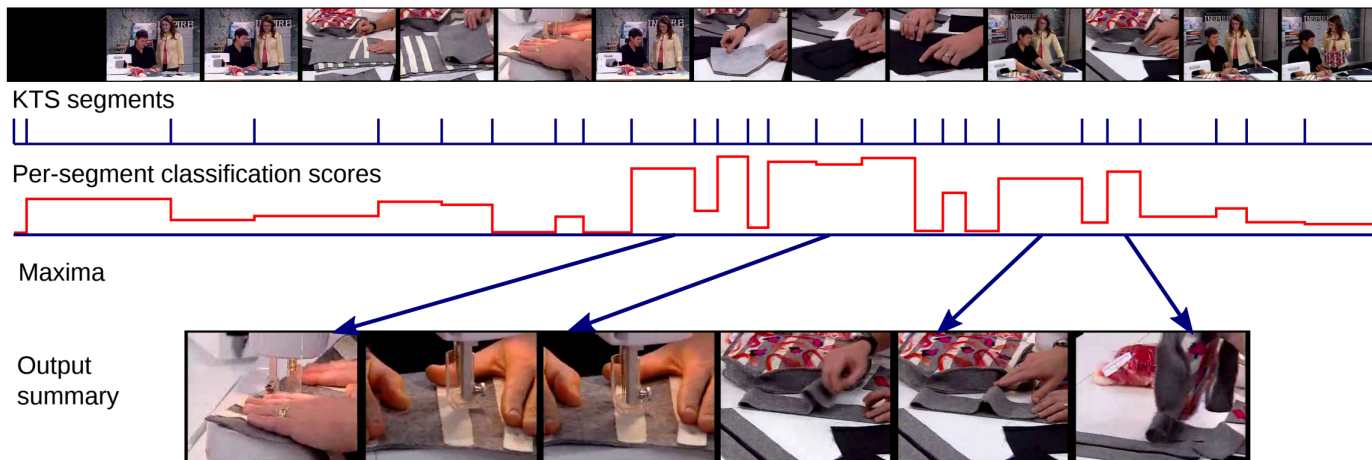


Рисунок 3.2 – Представлення основних кроків методу часової сегментації на основі ядра (KTS) для узагальнення змісту відео [162]

Існуючі рішення для узагальнення змісту відео на базі KTS [141; 160-164] зазвичай дотримуються загальної парадигми використання двох модулів для обробки довгих послідовностей: модуля попередньої обробки відео та модуля узагальнення змісту відео. Однак їхні модулі попередньої обробки покладаються на підхід із згорткою для часової сегментації (KTS) [163; 164]. Він виконує часову сегментацію на основі ядра згортки, що автоматично обирає кількість сегментів. Потім класифікатор на базі методу SVM підраховує важливості для кожного сегменту, виходячи з ваг, навчених на попередньо розміченому наборі даних. Наприкінці, підхід KTS формує підсумкове відео, яке складається із сегментів із найвищими прогнозованими оцінками важливості.

Головним недоліком KTS є витрати часу для обробки довгих відео (приблизно 68% від загального часу обробки витрачається на попередню обробку відео), оскільки обчислюються дисперсії кожного інтервалу та застосовується динамічне програмування для обчислення меж сегментів. Обчислювальна складність методу KTS становить  $O(n^2)$ , де  $n$  – кількість кадрів у відео, оскільки потрібно розглянути всі можливі інтервали та

отримати точні межі, що стає вузьким місцем для обробки довгих відео. Для роботи з 10-хвилинним відео потрібно більше ніж 8 годин обчислень при використанні базового підходу KTS.

Автори публікації [162] представили підхід D-KTS на основі методу KTS, який використовує адаптивний вибір кадрів на основі хешування (HAFS) для попередньої обробки кадрів, підвищуючи продуктивність їхнього рішення. Щоб зменшити складність обчислення ознак, вони запропонували для першого кадру записувати його репрезентацію у вигляді хеш-вектору через d-хеш [165]. Потім обчислювати хеш-вектор для кожного наступного кадру та робити порівняння з записаними векторами. Якщо відстань Хеммінга більша за порогове значення, вони замінюють запис поточним хеш-вектором і витягають вектор репрезентацій через нейронну мережу. В іншому випадку вони повторюють вектор репрезентацій кадру без обрахунку на поточному кадрі. На рисунку 3.3 [162] представлено візуалізацію підходу D-KTS з механізмом адаптивного вибору кадрів на основі хешу HAFS. За допомогою використання D-KTS і HAFS автори досягли найкращих результатів на наборах даних TvSum і SumMe за умови використання високої частоти кадрів.

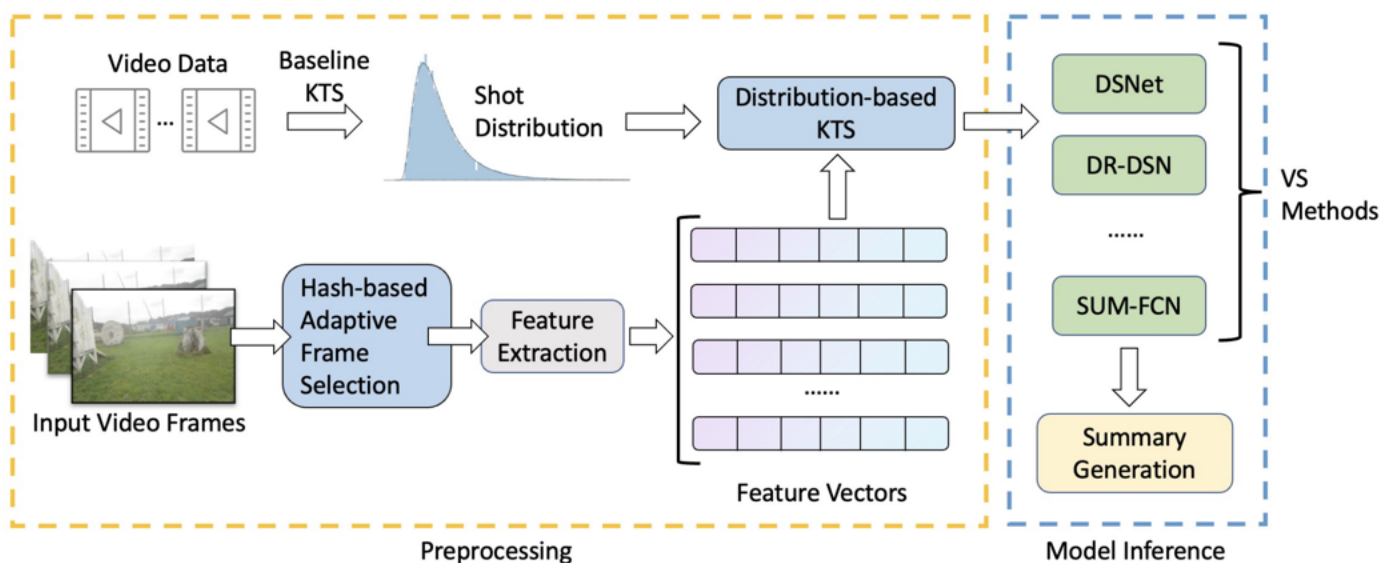


Рисунок 3.3 – Представлення основних кроків підходу D-KTS із механізмом адаптивного вибору кадрів на основі хешу HAFS [162]

Робота авторів заслуговує на особливу увагу своїми висновками про зв'язок між частотою кадрів і  $F$ -мірою, що наведено на рисунках 3.4 та 3.5 для набору даних TVSum та SumMe відповідно. Їхні дослідження показали, що існує позитивна кореляція між частотою кадрів і  $F$ -мірою, причому зниження частоти дискретизації до 1 кадру в секунду (як використовується в більшості пов'язаних робіт) призводить до негативного впливу на результат. Зокрема, автори показали, що  $F$ -міра зменшується в середньому на 20,92% для набору даних TvSum і на 20,72% для набору даних SumMe за умови зменшення частоти кадрів у відео нижче ніж 30 кадрів/с. Це важливе відкриття спонукало дослідити питання: «Чому підвищення частоти кадрів має такий позитивний вплив на  $F$ -міру?». Було припущено, що збільшення частоти кадрів призводить до більш точної ідентифікації меж сегментів на відео, що дає покращення порівняно з будь-якою евристикою, яка використовується в інших роботах [150-164].

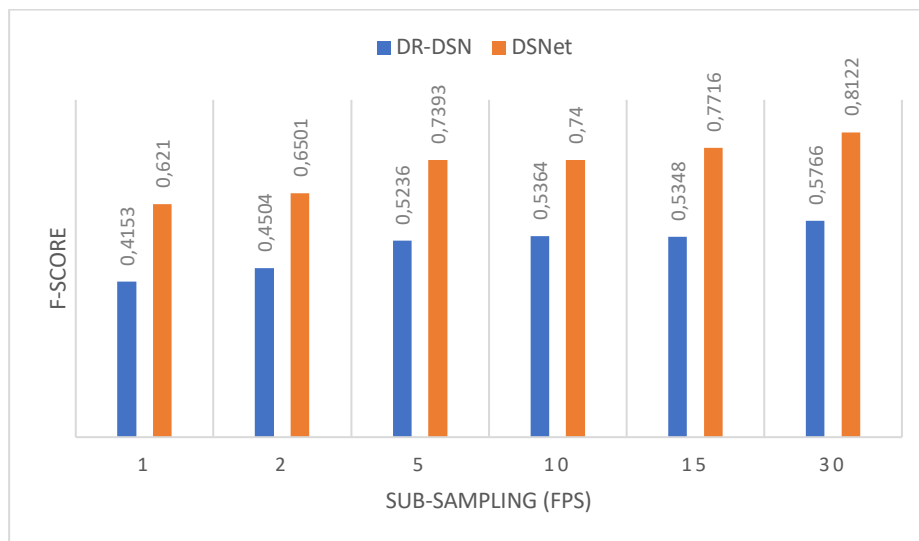


Рисунок 3.4 – Кореляція  $F$ -міри та FPS для підходу D-KTS [162] з використанням нейронних мереж DR-DSN [166] і DSNet [141] на наборі даних TVSum [147]

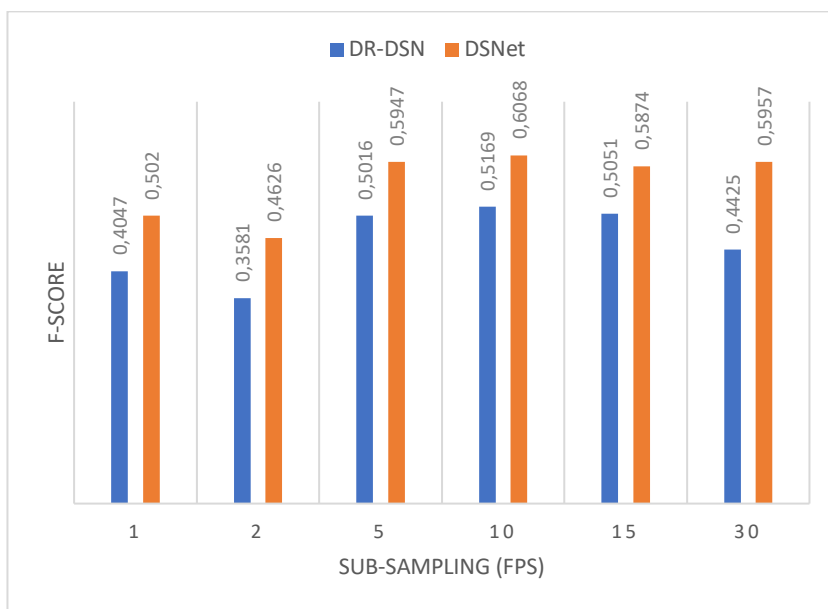


Рисунок 3.5 – Кореляція F-міри та FPS для підходу D-KTS [162] з використанням нейронних мереж DR-DSN [166] і DSNet [141] на наборі даних SumMe [148]

Варто зазначити, що запропонований у цьому дисертаційному дослідженні метод для узагальнення змісту довгих відео відрізняється від підходу D-KTS тим, що запропонованому методу не потрібні додаткові нейронні мережі для попереднього вибору кадрів для подальшого аналізу та узагальнення. У запропонованому методі використано підхід ітеративного уточнення меж узагальнень від менш до більш точних, що мінімізує розбіжності в оцінках ключових кадрів на різних етапах.

### 3.3 Запропонований ітеративний метод до вибору ключових кадрів для задачі узагальнення змісту відео

Запропонований ітеративний метод до узагальнення змісту відео складається з двох окремих фаз: попереднього вибору ключових кадрів та уточнення меж сегментів відео, по яких виконується узагальнення.

Фаза попереднього вибору ключових кадрів спрямована на визначення найбільш релевантних кадрів  $S'$  шляхом максимізації ймовірності вибору тих кадрів, які були

розмічені в наборі даних як ключові. Після цього відбувається фаза уточнення меж сегментів відео, по яких виконується резюмування шляхом відкидання нерелевантної інформації з сегментів для отримання більш точних і коротких сегментів відео. Щоб досягти цього, було запропоновано запустити метод резюмування змісту відео на вже відібраних кадрах  $S'$ , які згруповані в сегменти, але з вищою частотою, ніж це було на етапі попереднього вибору ключових кадрів, що дозволяє точніше обробити межі сегментів відео, по яких буде виконуватися резюмування.

Варто відміти, що різний розподіл даних фазою попереднього відбору ключових кадрів та фазою уточнення меж може призвести до різних оцінок впевненості моделей машинного навчання для тих самих кадрів на різних фазах. Щоб вирішити цю проблему, було запроваджено процедуру вирівнювання оцінок між різними фазами, яка спрямована на мінімізацію розбіжностей між ними. Це можна зробити ефективно, мінімізувавши евклідову відстань між ймовірностями, прогнозованими на різних етапах. Це, у свою чергу, призводить до більш високої якості моделі машинного навчання, яка може ефективно обробити дані з різних фаз методу ітеративного вибору ключових кадрів на відео. Даний процес формалізовано як Лістинг 3.1.

---

### Лістинг 3.1. Ітеративний метод узагальнення змісту відео

---

**Inputs:** keyframe selection model  $M$ , pre-defined processing sequence length  $T$ , the set of video frames  $F = \{F_1, F_2, \dots, F_n\}$ , the set of corresponding labels for each frame  $L = \{L_1, L_2, \dots, L_n\}$ , refinement padding length  $P$ , loss function  $F_{loss}$

---

1. If  $n < T$ , then pad  $F$  with  $T - n$  empty frames
2. From  $F$  subsample equal frequency  $T$  frames  $F' = \{F'_1, F'_2, \dots, F'_T\}$

*// Pre-selection*

3. Get importance scores  $S' = M(F')$

*// Border Refinement*

---

- 
4. From  $F$  subsample equally  $T$  frames  $F'' = \{F_1'', F_2'', \dots, F_T''\}$ :
    - 4.1.  $F'' = \{\}$
    - 4.2. While  $\text{len}(F'') < T$ :
      - 4.2.1. Select frame  $f$  with largest score in  $S'$ .
      - 4.2.2. Add frame  $f$  to  $F''$  and remove it from  $S'$ .
      - // Frame iteration order  $f, f-1, f+1, f-2, f+2, \dots, f-n, f+n$*
      - 4.2.3. For frames  $p$  in range from  $f - P$  to  $f + P$ :
        - 4.2.3.1. If  $p$  not in  $F''$  then add to  $F''$  and remove it from  $S'$
  5. Get importance scores  $S'' = M(F'')$ .
  - // Optional for training*
  6. Get loss  $\text{loss} = F_{\text{loss}}(S', L) + F_{\text{loss}}(S'', L) - \alpha \cdot (S'' - S')^2$ .
- 

**Outputs:** list of score  $S''$  for selecting keyframes.

---

На першому кроці перевіряється, чи  $n$  менше  $T$ . Якщо це так, то  $F$  доповнюється порожніми кадрами до довжини  $T$ . Ця операція обчислювана та виконується за скінченну кількість кроків час  $O(T)$ , оскільки максимальна кількість кадрів для доповнення обмежена  $T - n$ .

На другому кроці з  $F$  із заданою частотою (тобто кожний  $k$ -ий кадр) вибираються  $T$  кадрів, що утворюють підмножину  $F'$ . Оскільки  $T$  фіксоване, кількість вибраних кадрів також фіксована, і ця операція обчислювана та завершується за скінченну кількість кроків не більшу ніж  $O(T)$ .

На третьому кроці використовується модель  $M$  для отримання значень важливості  $S'$  для кадрів у  $F'$ . Процес обробки кадрів моделлю  $M$  є обчислюваним і завершується за скінченну кількість кроків  $O(T \cdot m)$ , де  $m$  – складність обчислень за моделлю  $M$ .

На четвертому кроці виконується уточнення меж. Спочатку знову вибираються  $T$  кадрів з  $F$  із заданою частотою, утворюючи підмножину  $F''$ . Побудова набору  $F''$  відбувається за допомогою значень важливості  $S'$ . Цей процес виконується за скінченну

кількість кроків, оскільки кількість ітерацій циклу рівна  $T$ , а для кожної ітерації кількість кроків фіксована і визначена довжини вікна обробки  $P$ . Описані кроки є обчислюваними і мають оцінку складності  $O(T \cdot n)$

На п'ятому кроці модель  $M$  знову використовується для отримання значень важливості  $S''$  для кадрів у  $F''$ . Цей крок аналогічний третьому – є обчислюваним і має складність  $O(T \cdot m)$ .

На шостому кроці обчислюється функція втрат, яка враховує значення  $S'$ ,  $S''$  та мітки  $L$ . Цей розрахунок має складність  $O(n)$ .

Таким чином, загальна складність алгоритму, побудованого на основі запропонованого методу, може бути приблизно оцінена як  $O(T + T \cdot m + T \cdot n) = O(T \times (1 + m + n)) \sim O(T \times (m + n))$ . Такий алгоритм завжди завершується за скінченну кількість кроків і є обчислювально здійсненним, оскільки всі його кроки виконуються за фіксовану кількість часу без безкінечних циклів або інших перешкод, що запобігають його завершенню.

Схематичне представлення застосування даного методу наведено на рисунку 3.6. На зображенні сірим кольором представлено невібрані кадри, червоні – це кадри, вибрані для обробки на фазі попереднього відбору, жовті – це кадри з високою впевненістю моделей машинного навчання, які вибрані на фазі попереднього відбору, сині – це кадри, які обрані для етапу уточнення меж як додаткові кадри для підвищення кількості кадрів в сегментах, зеленим – це кадри, вибрані після фази уточнення меж, які є кінцевим результатом роботи методу. Числа всередині кожного кадру відображають оцінки впевненості моделей машинного навчання для цього кадру після кожної фази обробки.



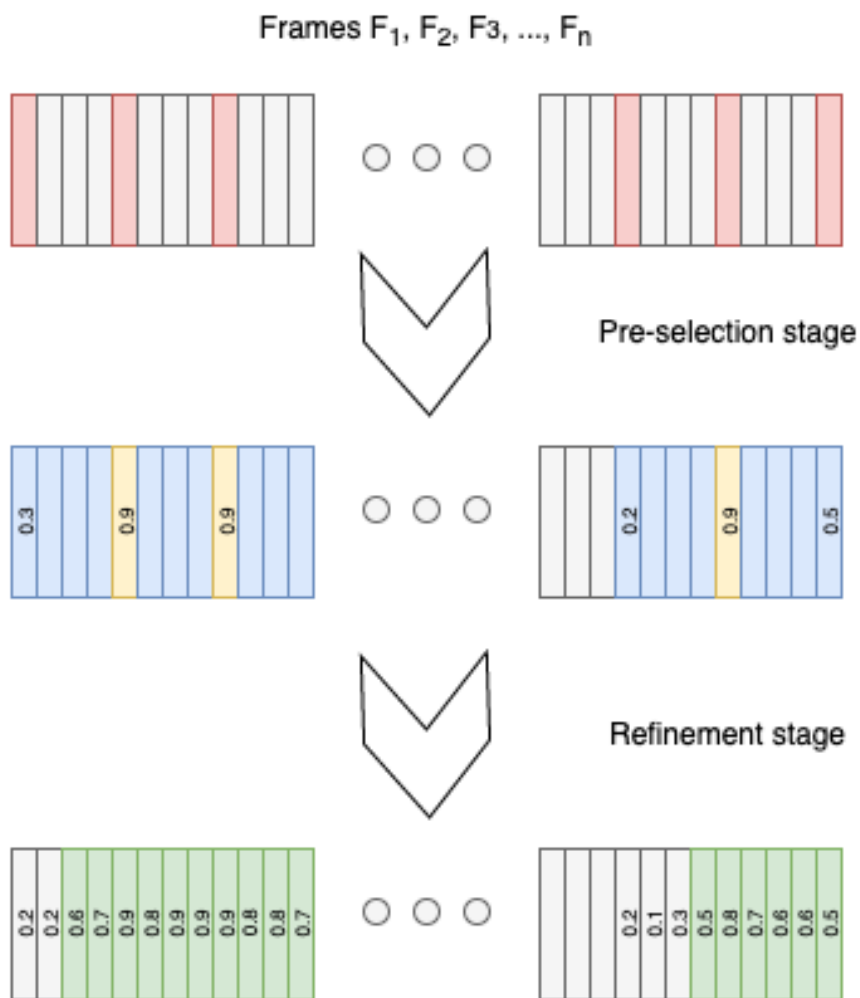


Рисунок 3.6 – Представлення основних кроків ітеративного методу узагальнення по відео

### 3.4 Результати експериментального дослідження ефективності запропонованого методу відбору ключових кадрів для задачі узагальнення змісту відео

Для порівняння запропонованого методу з попередніми роботами було проведено експерименти на чотирьох різних наборах даних: TvSum [147], SumMe [148], OVP [167] і YouTube [167]. Набори даних OVP і YouTube використовувалися головним чином для розширення навчального набору. TvSum і SumMe є єдиними наразі доступними наборами даних із відповідною розміткою для узагальнення змісту відео, але вони

залишаються відносно невеликими для навчання глибоких моделей. Основні характеристики цих наборів даних наведено в таблиці 3.1.

Таблиця 3.1 – Огляд властивостей наборів даних, які використані в експериментах для задачі узагальнення змісту відео

Набір даних	Кількість відео	Кількість з розміткою	Тип розмітки	Довжина відео, секунд		
				Мінімальна	Середня	Максимальна
SumMe [147]	25	18	Ключові сегменти	32	146	324
TvSum [148]	50	20	Ключові кадри	83	235	647
OVP [167]	50	5		46	98	209
YouTube [167]	39	5		9	196	572

Набір даних TvSum має розмітку на рівні кадрів, тоді як SumMe розмічений на рівні сегментів. Під час розмітки цих наборів даних кількість кадрів для узагальнення змісту відео обмежувалася до 15% від оригінальної довжини відео і оцінювалася на основі результатів декількох розмітчиків даних. Такий підхід дозволяє забезпечити консистентність та об'єктивність оцінювання якості розмітки, необхідної для успішного використання даних у навчанні глибоких моделей.

Для об'єктивного порівняння з попередніми дослідженнями було дотримано методики оцінки ефективності запропонованого методу узагальнення змісту відео, описану у роботі [136]. Для оцінки схожості результатів використовувалась середньо-гармонійна точність і повнота, виражена у відсотках як F-міра:

$$F = 200 \cdot p \cdot r / (p + r), \quad (3.1)$$

де  $p$  - значення точності,  $r$  - значення повноти.

Істинно позитивні, хибно позитивні та хибно негативні результати для  $F$ -міри обчислюються для кожного кадру як перекриття між розміткою та результатами обчислень моделей машинного навчання, що представлено на рисунку 3.7 [138]. Такий підхід дозволяє детально оцінити відповідність та точність роботи методу на кожному етапі обробки відео, визначаючи його ефективність і придатність для конкретних завдань комп'ютерного зору.

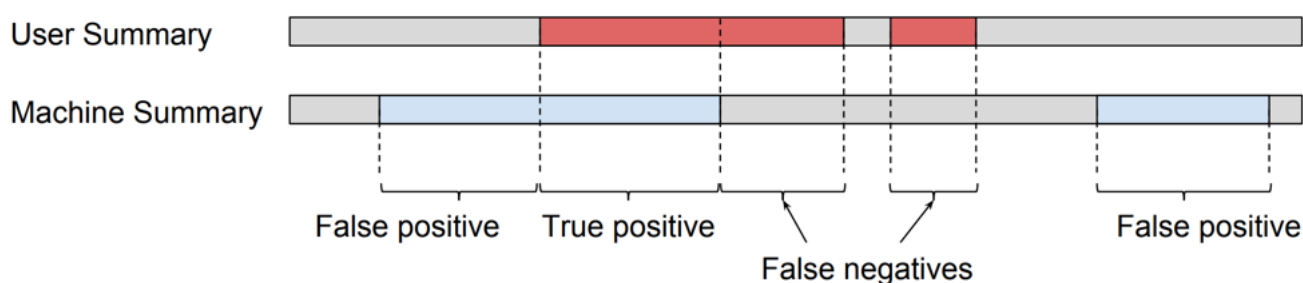


Рисунок 3.7 – Перекриття між розміткою та результатом обчислень методу – істинно позитивних, хибно позитивних та хибно негативних результатів для кожного кадру [138]

Для набору даних TvSum [147] для кожного відео  $F$ -міра обчислюється як середнє значення між результатом методу узагальнення змісту відео і розміткою кожного розмітчика даних окремо. Потім підраховується середня  $F$ -міра по всіх відео в наборі даних. Для набору даних SumMe [148], оскільки дані розмічені декількома розмітчиками, для кожного відео вибирається лише розмітка того розмітчика, яка дає найкращий результат  $F$ -міри.

Усі експерименти проводилися з використанням Python 3.9.13, PyTorch 1.13.1, CUDA 11.7, OpenCV 4.5.2 на базі операційної системи Ubuntu 20.04. Обчислювальна машина мала 8 графічних процесорів Nvidia A100 80 ГБ, 2 ЦП Intel(R) Xeon(R) Platinum 8275CL які функціонують на частоті 3,00 ГГц і системну пам'ять розміром в 1152 ГБ.

У всіх проведених дослідженнях було дотримано встановленого протоколу оцінювання моделей машинного навчання на наборах даних TvSum і SumMe. Для фази

попереднього відбору ключових кадрів було використано дискретизацію відео з частотою кадрів в 1 кадр/с. Векторне представлення кадрів було отримано з використанням моделей Video Swin Transformer [66], зокрема Swin-T. Розмір вхідних кадрів було змінено до роздільної здатності  $224 \times 224$  пікселів, що призвело до розміру вхідного тензора для моделей машинного навчання в  $16 \times 56 \times 56$  елементів.

Модель машинного навчання була ініціалізована вагами попередньо навченої моделі на наборі даних ImageNet. Голови були ініціалізовані випадковими вагами з нормальним розподілом. Оптимізацію було проведено за допомогою методу AdamW [128] з початковою швидкістю навчання  $3 \cdot 10^{-4}$ . Модель навчалася загалом 200 епох, включаючи фазу розігріву в 2,5 епохи [168]. Розмір вхідних даних під час навчання складав 64 екземпляра. Аналогічно до попередніх досліджень, градієнти ваг попередньо навченої нейронної мережі були помножені на коефіцієнт 0,1. Зростаючий рівень стохастичного відкидання глибини [129] та ваг [130] використовувався для більших моделей, як рекомендовано в [66]. Стохастичне відкидання глибини було встановлено на рівні 0,1, а стохастичне відкидання ваг було встановлено на рівні 0,02. Усі експерименти було виконано зі встановленим  $\alpha = 0,5$ .

Найліпшу модель було обрано на основі функції втрат. Усі експерименти проводилися з використанням загальнодоступної реалізації PGL-SUM [160]. У таблиці 3.2 представлено порівняння підходів до задачі узагальнення змісту відео на наборах даних SumMe [147] і TvSum [148].

Таблиця 3.2 – Порівняння різних методів для вибору ключових кадрів на задачі узагальнення змісту відео на наборах даних SumMe [147] і TvSum [148]

Метод	SumMe, F1-міра	TvSum, F1-міра
VasNet [107]	49,7%	61,4%
RR-STG [109]	53,4%	63,0%

Продовження таблиці 3.2

Метод	SumMe, F1-міра	TvSum, F1-міра
DSNet [110]	50,2%	62,1%
PGL-SUM [130]	55,6%	62,7%
D-KTS з вагами DSNet [132]	50,2%	62,3%
<b>Запропонований ітеративний метод до відбору ключових кадрів на відео</b>	<b>56,2%</b>	<b>63,2%</b>

Усі методи тестувалися з фіксованою частотою семплювання кадрів з відео, яка була виставлена в 1 кадр/с згідно з протоколами тестування на наборах даних SumMe [116] і TvSum [117]. Як можна побачити, запропонований метод досяг найкращих на момент дослідження результатів точності – 56,2% (+0,6%) і 63,1% (+0,5%) на наборах даних SumMe і TVSum. Запропонований метод перевершивши результати методів PGL-SUM [130] та D-KTS [132], що підкреслює ефективність запропонованого методу при роботі з низькою частотою кадрів і додатково буде досліджено в наступному підрозділі.

### 3.5 Дослідження впливу ключових компонентів на якість запропонованого ітеративного методи вибору ключових кадрів для узагальнення змісту відео

У рамках цього дослідження було запропоновано інноваційний підхід до узагальнення змісту відео, який включає три основні модифікації в структурі PGL-SUM [130]. По-перше, було здійснено заміну попередньо навченої нейронної мережі на Video Swin Transformer, який зарекомендував себе як ефективний екстрактор векторних репрезентацій для задач на відео. По-друге, було запропоновано метод ітеративного уточнення меж сегментів для узагальнення, що підвищує точність вибору ключових кадрів. По-третє, було введено процедуру вирівнювання оцінок нейронних мереж для ключових кадрів які відібраними на фазі попереднього відбору та фазою уточнення щоб

краще узгодити оцінки на різних етапах методу. Ефективність кожної модифікації оцінена та представлена в таблиці 3.3.

Таблиця 3.3 – Вплив різних компонентів запропонованого методу

Вдосконалення	SumMe, F1-міра	TvSum, F1-міра
PGL-SUM [130] (основа)	55,6%	62,7%
+ Video Swin Transformer	55,9% (+0,3%)	62,9% (+0,2%)
+ Ітеративний підхід	56,1% (+0,2%)	63,1% (+0,2%)
+ Вирівнювання оцінок між різними ітераціями	56,2% (+0,1%)	63,2% (+0,1%)

Результати свідчать, що приблизно 50% загального підвищення точності у порівнянні з PGL-SUM [130] досягається за рахунок використання нейронної мережі Video Swin Transformer. Решта досягається за рахунок запропонованого ітеративного методу вибору ключових кадрів та вирівнювання оцінок між різними ітераціями методу.

### **3.6 Порівняння запропонованого ітеративного методу до вибору ключових кадрів з методом D-KTS при високій частоті кадрів для задач узагальнення змісту відео**

Щоб порівняти запропонований метод із методом D-KTS [162], було збільшено частоту кадрів для фази попереднього вибору ключових кадрів та лінійно збільшено часову розмірність вхідного тензора для Video Swin Transformer [66]. Як показано на рисунках 3.8 та 3.9, запропонований метод ітеративного вибору ключових кадрів на відео перевершив метод D-KTS за умови використання меншої частоти кадрів для фази попереднього відбору ключових кадрів.

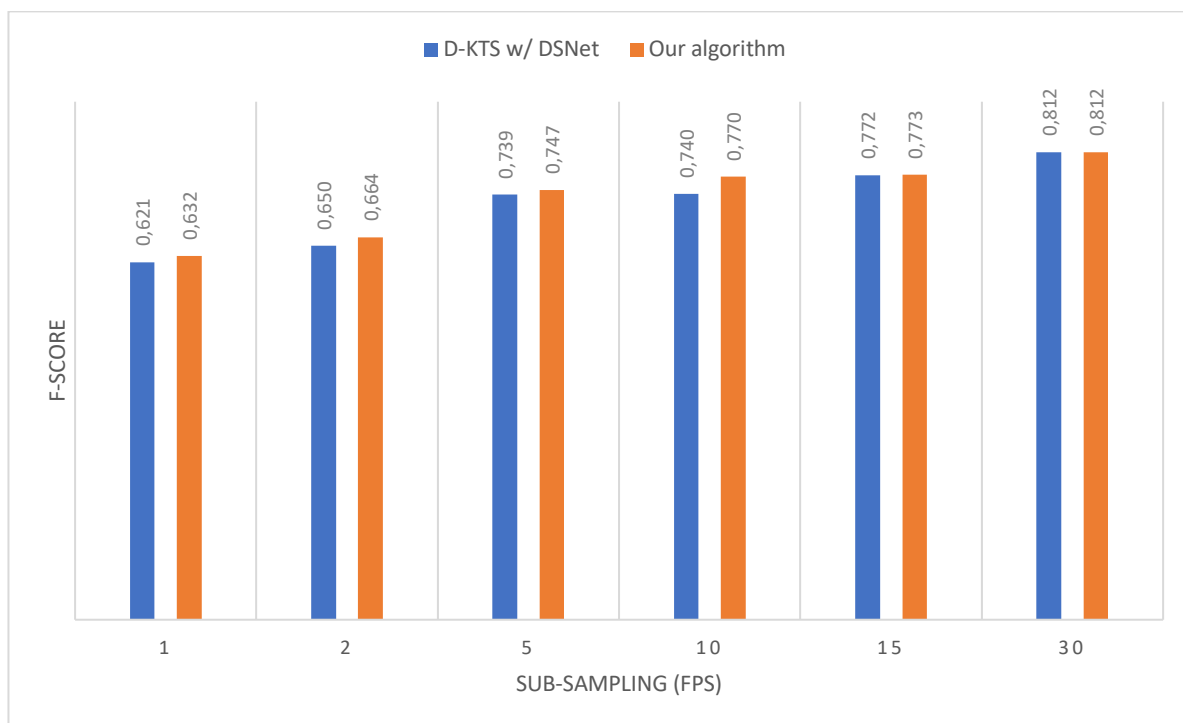


Рисунок 3.8 – Порівняння запропонованого методу з методом D-KTS [162] при вищій частоті кадрів на наборі даних TVSum [147]

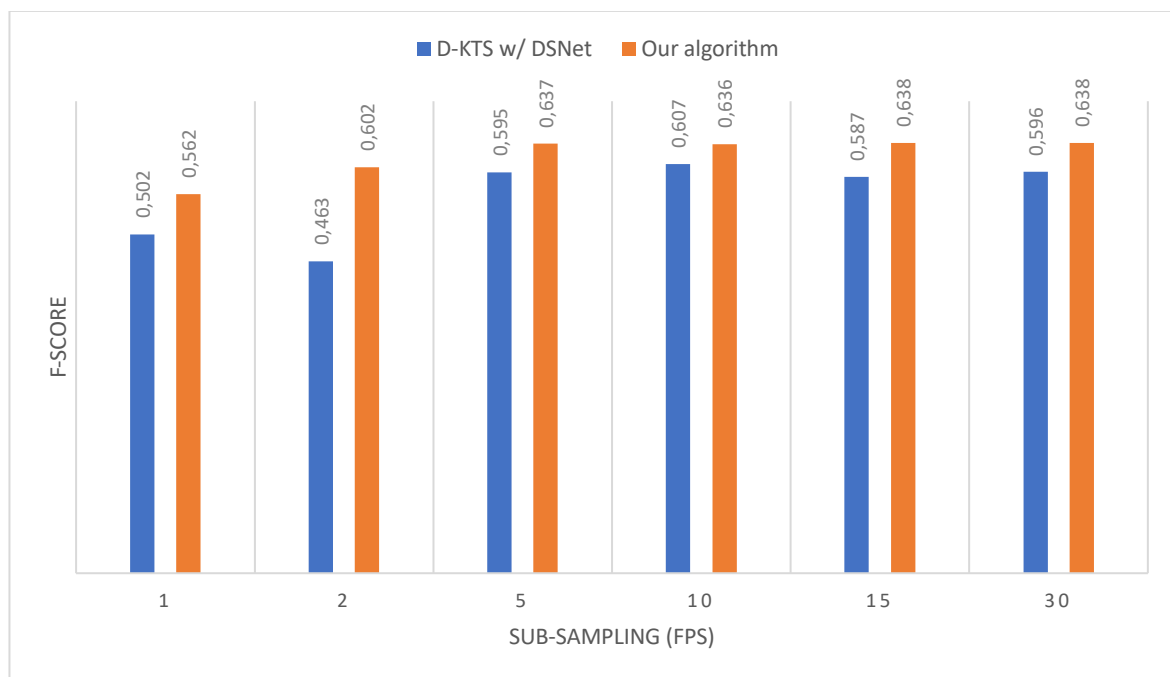


Рисунок 3.9 – Порівняння запропонованого методу з методом D-KTS [162] при вищій частоті кадрів на наборі даних SumMe [148]

Однак варто зазначити, що ця різниця зменшується та стає незначною при використанні високої частоти кадрів, такої як 15 і 30 кадрів в секунду, що призводить до різниці між запропонованим ітеративним методом по вибору ключових кадрів та D-KTS лише в +0,1% і +0,0% на наборах даних SumMe і TVSum відповідно. Це пояснюється тим фактом, що при високій частоті кадрів обидва методи мають доступ до більшої кількості кадрів, що призводить до швидкого зменшення різниці в точності виявлення меж сегментів відео. Однак, як було зазначено вище, основна мета дослідження полягала в тому, щоб надати практичне рішення для обробки довгих відео, що не призводило б до додавання складності або додаткових витрат під час фази тестування методу в реальних умовах. Запропонований метод у цьому відношенні показує кращу швидкодію, ніж метод D-KTS, описаний у роботі [162], а отже його можна використовувати з нижчою частотою кадрів і розгорнути на менших графічних прискорювачах GPU, які мають 12 ГБ відео пам'яті для обробки послідовностей довжиною в десятки хвилин, що часто зустрічається на таких платформах, як YouTube або Facebook [144]. Наведені у таблиці 3.4 дані отримані для прискорювача Nvidia A100 80GB. Саме тому можна стверджувати, що запропонований метод має переваги для розробки прикладних застосунків для узагальнення змісту відео.

Таблиця 3.4 – Залежність обсягу відеопам'яті, необхідної для виконання ітеративного методу до вибору ключових кадрів, від довжини відео.

Характеристика відео	Метод D-KTS	Ітеративний метод
1280x720, 30FPS, 1 хвилина	8GB	11GB
1280x720, 30FPS, 5 хвилин	55GB	16GB
1280x720, 30FPS, 15 хвилин	Помилка через вистачу пам'яті	21GB



Продовження таблиці 3.4

Характеристика відео	Метод D-KTS	Ітеративний метод
1280x720, 5FPS, 5 хвилин	7GB	10GB
1280x720, 5FPS, 15 хвилин	18GB	12GB
1920x1080, 5FPS, 1 хвилина	22GB	39GB
1920x1080, 5FPS, 5 хвилин	Помилка через нестачу пам'яті	44GB

### 3.7 Висновки до розділу

У даному розділі запропоновано новий ітеративний метод для узагальнення змісту відео. Запропонований метод зменшує кількість нерелевантних кадрів, які визначені на фазі попереднього відбору, шляхом ітеративного застосування моделі машинного навчання до відібраних кадрів з подальшою фільтрацією нерелевантних кадрів та сегментів.

У дослідженні додатково реалізовано вирівнювання оцінок для відібраних ключових кадрів на різних ітерація методу, що забезпечує їх узгодженість та підвищує якість запропонованого методу. У порівнянні з D-KTS, найближчим конкурентом, запропонований метод має переваги для прикладних застосунків, оскільки досягає аналогічної продуктивності за нижчої частоти кадрів, що робить його придатним до використання на електронно-обчислювальних пристроях зі слабкими графічними прискорювачами, які мають 12 ГБ відео пам'яті.

Практичні дослідження запропонованого методу показали підвищення якості узагальнення змісту відео на наборах даних SumMe та TVSum на +0,6% та +0,5% відповідно. Ці результати демонструють ефективність запропонованого методу у вирішенні задачі узагальнення змісту відео, особливо для довгих послідовностей, що робить його важливим інструментом для практичного застосування у розробці прикладних застосунків.

Результати даного дослідження опубліковано у науковому виданні [149] та презентовані на міжнародній конференції *Advances in Artificial Systems for Logistics Engineering III* (ICAILE 2023).

## **4 МУЛЬТИМОДАЛЬНІ МОДЕЛІ НЕЙРОННИХ МЕРЕЖ ДЛЯ ЗАДАЧ РОЗПІЗНАВАННЯ НА ВІДЕО**

### **4.1 Використання мультимодальних моделей для задач розпізнавання на відео**

Останніми роками багато досягнень у попередньому навчанні моделей глибокого машинного навчання в галузі обробки природної мови (NLP) викликали інтерес у спільноті розробників комп'ютерного зору. Яскравими прикладами є BERT [169] GPT [170; 171], ERNIE [172] і T5 [173], які надихнули дослідників досліджувати подібні методи комп'ютерного зору.

Візуально-мовні моделі (VLM) є результатом досліджень [169-173] в області NLP. Вони використовують великі набори даних у вигляді пар зображення-текст, що характеризуються слабкою відповідністю (відсутністю чіткої відповідності) та значним шумом, для контрастного навчання моделей машинного навчання. Таке навчання зосереджується на генерації векторних репрезентацій шляхом протиставлення позитивних і негативних пар екземплярів. Прикладами навчання моделей машинного навчання на таких даних є CLIP [44], CoCa [70] і Florence [81]. Ці візуально-мовні моделі продемонстрували надзвичайну універсальність та здатність ефективно застосовуватися до широкого спектру задач комп'ютерного зору.

Природньо, що внаслідок досягнень, представлених у роботах [44; 70; 81] для задач комп'ютерного зору, з'явилися ідеї щодо використання знань, закодованих у попередньо навчених візуально-мовних моделях, для розробки моделей розпізнавання на відео. Сучасне різноманіття досліджень у цьому напрямку можна розділити на декілька.

Прямолінійний підхід до використання знань візуально-мовних моделей, висвітлений в роботах [174; 175; 176], дотримується традиційної унімодальної

парадигми розпізнавання на відео, а попередньо навчена візуально-мовна модель використовується шляхом ініціалізації ваг нейронної моделі з енкодера VLM (Visual Encoder) та використанням згенерованих репрезентацій (Emb.) перед повнозв'язним шаром (FC) (рис. 4.1).

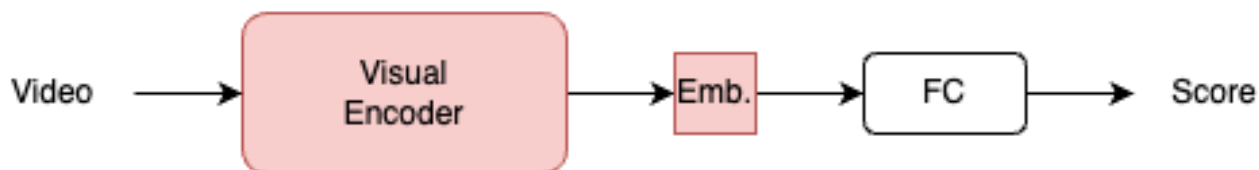


Рисунок 4.1 – Прямолінійний підхід до використання знань закодованих у візуально-мовних моделях

Альтернативний підхід до використання знань візуально-мовних моделей, описаний у публікаціях [77; 177; 178], включає VLM структуру навчання на парах даних зображення-текст, що використовує елементи природної мови (Textual Encoder), такі як назви класів для коригування результатів роботи підходів (рис. 4.2).

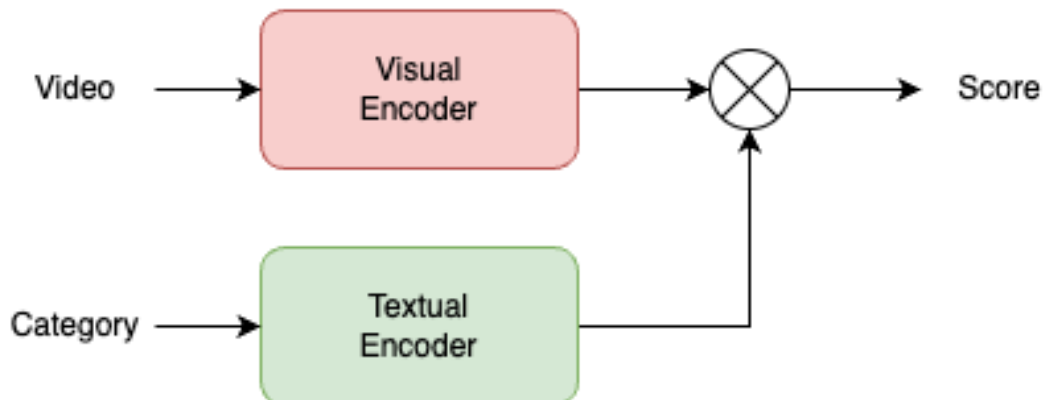


Рисунок 4.2 – Основні кроки підходів з використання знань, закодованих у відео-текстових компонентах VLM

Метод BIKE [78] висвітлює обмеження вищезазначених підходів до використання знань візуально-мовних моделей, які в основному покладалися на однонаправлене зіставлення відео з текстом. Такі обмеження ускладнили використання повного потенціалу візуально-мовних моделей для розпізнавання на відео. Саме тому метод BIKE запропонував новий підхід, який націлений на двонаправлене дослідження знань у візуальній та текстовій областях візуально-мовних моделей (Video Saliency та Attribute Feature), як показано на рисунку 4.3 [78].

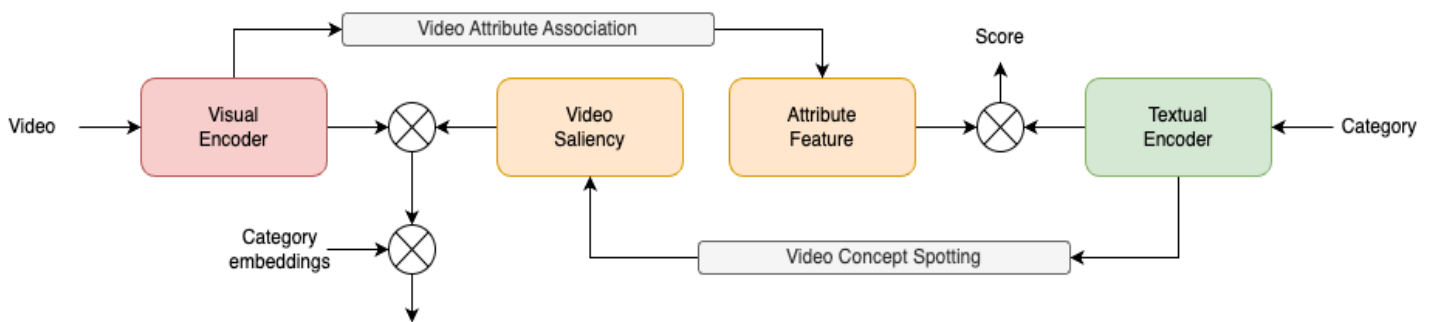


Рисунок 4.3 – Представлення основних кроків метод BIKE [78]

Підхід BIKE [78] вводить концепцію майнінгу знань у гілках Video-to-Text і Text-to-Video візуально-мовних моделей. Цей підхід включає процес генерації текстової інформації з вхідного відео, що автори називають генерацією атрибутів, а також використання опису категорій для виділення сегментів на відео, де існують дані категорії. В результаті цього двонаправленого дослідження знань досягається підвищення точності підходу BIKE [78]. Однак для досягнення цих вдосконалень потрібно внести зміни у структуру VLM, що може обмежувати використання попередньо натренованих моделей і вимагає розробки нових архітектур або адаптації існуючих для ефективного використання.

Отже, одним із завдань даного дослідження є розробка підходу по двонаправленному дослідженню знань візуально-мовних моделей, яка б не потребувала структурних змін у VLM та підвищувала точність розпізнавання.

Для досягнення цієї мети у рамках дисертаційного дослідження оптимізована структура BIKE [78] шляхом усунення взаємозалежностей між гілками Video-to-Text та Text-to-Video. Замість цього було запропоновано інтегрувати знання текстового класифікатора, аналогічно до підходу Text4Vis [179], у гілку Video-to-Text. Додатково, було розроблено процес класифікації атрибутів для сегментів відео, що перекриваються між собою. Це підходить для кодування інформації про зміни атрибутів у часі, що істотно покращує репрезентацію подій у відеоряді. Особлива увага була приділена важливості попередньо визначеного лексикону для генерації атрибутів за допомогою візуально-мовних моделей. У розділі 4.5.2 також продемонстровано як можна розширити лексикон за допомогою великих мовних моделей (LLM), таких як ChatGPT, що розширює можливості генерації атрибутів для відеоданих. Огляд запропонованого підходу представлено на рисунку 4.4, де висвітлено основні аспекти інтеграції текстових та візуальних даних.

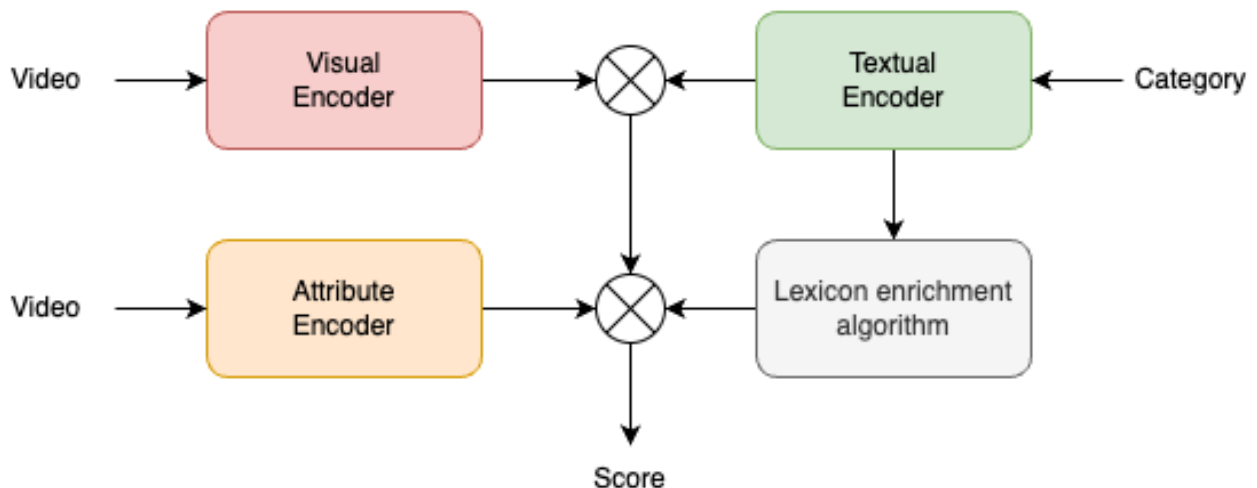


Рисунок 4.4 – Основні кроки підходу Attr4Vis

Результати запропонованого підходу Attr4Vis були опубліковані у науковому журналі International Journal of Computing [180].

## 4.2 Запропонований підхід до використання мультимодальних моделей для задач розпізнавання на відео

Підхід Attr4Vis, який розроблений у цьому дисертаційному дослідженні, схематично представлено рисунку 4.5. У наступних підрозділах буде більш детально розглянуто компоненти Text-to-Video, Video-to-Text та генерації атрибутів по відео.

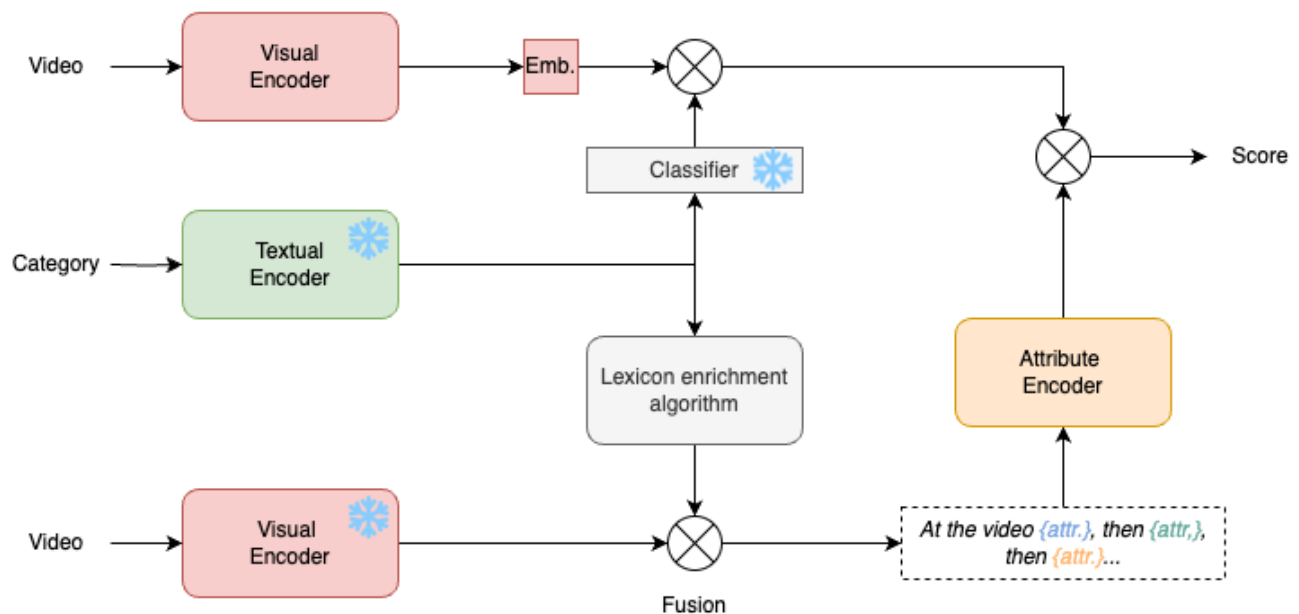


Рисунок 4.5 – Представлення основних кроків запропонованого підходу Attr4Vis

Для використання знань з текстової гілки InternVideo [80] дотримано процедуру кодування текстового класифікатора з [179], що дозволило перенести семантичні знання з тексту на візуальну модель. Ця процедура передбачає створення вагових коефіцієнтів  $W$ , які складаються з векторів репрезентацій для текстових міток набору даних  $L$ . Для набору даних та наявних міток класів  $L = \{L_1, L_2, L_3, \dots, L_N\}$ , буде згенеровано наступну матрицю вагових коефіцієнтів  $W$ :

$$W_i = \text{TextEncoder}(L_i), i = 1, 2, \dots, n, \quad (4.1)$$

де  $W_i$  -  $i$ -ий вектор-рядок матриці  $W$ ,

Отже,  $W_i$  ініціалізується з використанням результату LLM для текстових міток  $i$ -го класу. Як *TextEncoder* в експериментах, що описані у підрозділах 4.4 та 4.5, використано модель Multilingual-E5-large [82].

У запропонованому підході використано можливості нульового навчання, властиві фундаментальним візуально-мовним моделям, до яких відноситься InternVideo [80], щоб розпізнати найбільш відповідні фрази в межах попередньо визначеного лексикону та позначити їх як потенційні «Атрибути» для задачі. Цей процес розгортається таким чином: все розпочинається із застосування енкодера до невеликих фрагментів вхідного відео шляхом генерування векторів репрезентацій для кожного кадру з фрагменту відео з подальшим узагальненням їх за допомогою підрахунку середніх значень у даному фрагменті, що дозволяє отримати вектор репрезентацій для фрагменту відео. Одночасно подаємо попередньо визначену лексику в текстовий енкодер, що призводить до генерації набору атрибутів. Далі переходимо до обчислення подібності між кожним сегментом і вбудованим атрибутом, вибираючи лише ті, які показують найвищий ступінь узгодженості. Успішно визначивши ці атрибути, можна використати просту техніку злиття, яка об'єднує їх з послідовності атрибутів у коротке речення, що має форму "На відео {}, потім {}, потім {}...". Цей простий, але комплексний підхід забезпечує створення осмислених і контекстно-релевантних допоміжних атрибутів для аналізу відео.

Процес генерації атрибутів для відео зображено на рисунку 4.6.



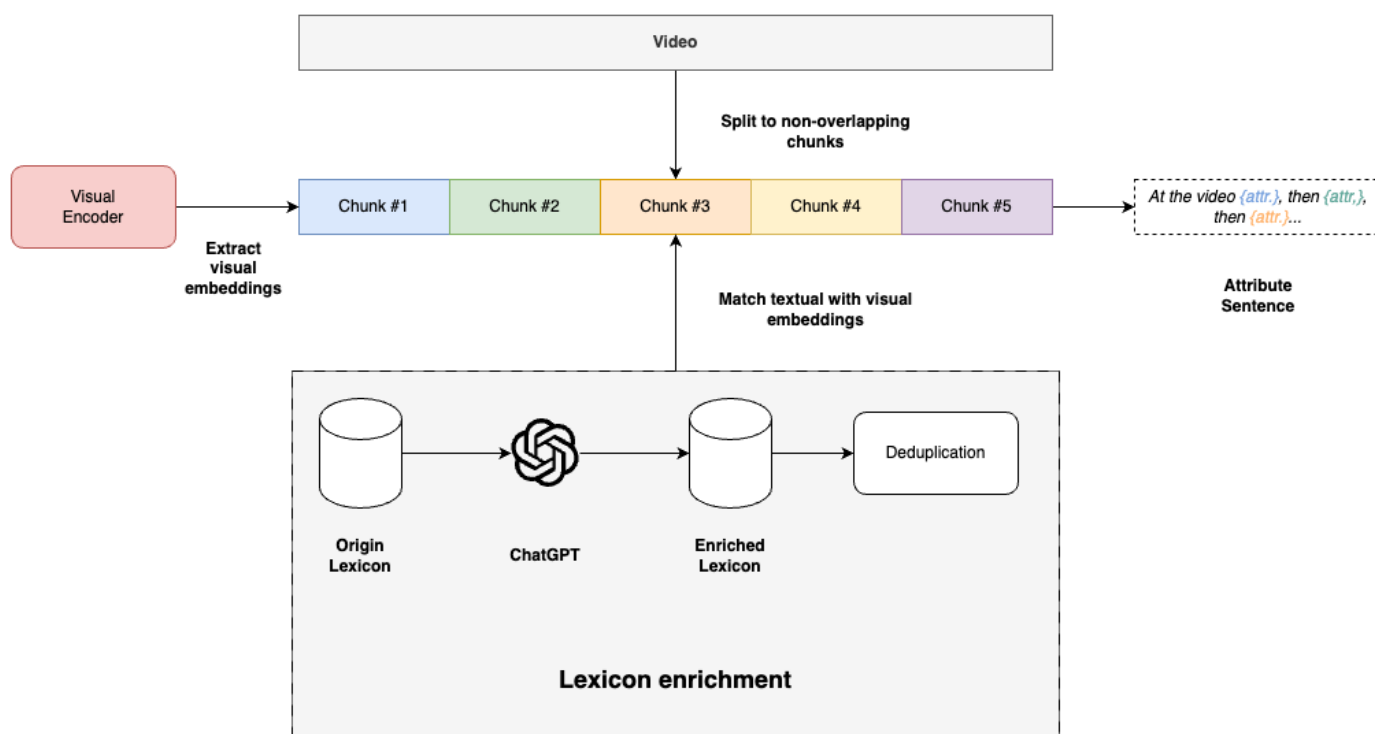


Рисунок 4.6 – Структура гілки Video-to-Text, яка для заданого вхідного відео генерує послідовність атрибутів за попередньо визначеним словником

### 4.3 Алгоритм збагачення лексикону атрибутів

Традиційно набори даних для розпізнавання на відео обмежуються певною кількістю категорій, визначених мітками для даного набору даних. У контексті структури BIKE [78] мітки з набору даних можуть використовуватись виключно для створення лексикону. Однак, як показало дослідження, підхід, який покладається виключно на цей оригінальний набір міток, накладає певні обмеження на здатність візуально-мовних моделей (VLM), таких як InternVideo [80], генерувати урізноманітненні атрибути відео. Щоб усунути це обмеження та розширити варіативність доступних міток, запропоновано розширювати лексикон шляхом використання діалогових LLM моделей, таких як ChatGPT. Завдяки цьому алгоритму збагачено лексикон, який може бути використаний VLM для генерації атрибутів на

відео. Цей підхід не тільки поліпшує описові можливості VLM, але й сприяє генерації більш детальних і контекстно-релевантних атрибутів, що сприяє підвищенню продуктивності розпізнавання на відео. Детальна формалізація запропонованої процедури збагачення лексикону наведена у вигляді псевдокоду як Лістинг 4.1:

---

**Лістинг 4.1.** Запропонований алгоритм збагачення лексикону атрибутів

---

**Inputs:** Dataset  $D$ , set dataset labels  $L = \{L_1, L_2, \dots, L_n\}$ , Textual Embedding model  $M$ , deduplication threshold  $\tau$ .

---

*// Generate enriched labels*

1.  $L_e = \{\}$  *// Initialize set of enriched labels*

2. For each  $L_i$  in  $L$ :

*// Preparing textual prompt for each label*

2.1. prompt = “You are a domain expert in the  $\{D\}$  dataset, helping develop a labeling system. Generate additional labels that will enrich  $\{L_i\}$  label of  $\{D\}$  dataset. Format output as comma separated labels.”

2.2. result = ChatGPT(prompt) *// Generate additional labels*

2.3.  $L_e.add(result)$

*// Deduplicate labels by semantic meaning*

3.  $E_e = \{M(L_i) \text{ for } L_i \text{ in } L_e\}$  *// Generate embeddings for each label*

4.  $L_o = \{\}$  *// Initialize output labels*

5. For  $e$  in  $E_e$ :

5.1. distance = cosine ( $e, L_o$ ) *// Calculate cosine distance for each pair*

5.2. if all(distance)  $< \tau$ :

5.2.1.  $L_o.add(e)$

---

**Outputs:** list of enriched labels  $L_o$

---

Алгоритм починається з чітко визначених вхідних параметрів: датасету  $D$ , набіру міток  $L = \{L_1, L_2, \dots, L_n\}$ , моделі для формування текстових векторних представлень  $M$  та порогу дедублікації  $\tau$ . Ці параметри є фіксованими та заданими наперед, що забезпечує визначеність початкових умов.

На першому кроці ініціалізується порожній набір розширених міток  $L_e$ . Ця операція обчислювана та виконується за константний час  $O(1)$ .

На другому кроці виконується цикл по кожній мітці  $L_i$  з множини  $L$ . Оскільки таких ітерацій буде  $n$ , де  $n$  — кількість міток у  $L$ , цикл має складність  $O(n)$ . На підкроці 2.1 формується текстовий запит для кожної мітки. Формування текстового запиту є операцією з постійною складністю  $O(1)$ . На підкроці 2.2 викликається модель ChatGPT для генерації додаткових міток. Час обробки запиту до ChatGPT залежить від складності самого запиту, але обмежений зверху часом на генерацію 64000 вихідних токенів, а отже складність цього підкроку буде константною  $O(1)$ . На підкроці 2.3 додавання результату в  $L_e$  виконується за амортизовану константну складність  $O(1)$ . Отже, всі операції для кроку 2 обчислювані та мають складність  $O(n)$ .

Третій крок полягає в генерації векторних представлень для кожної мітки в  $L_e$  за допомогою моделі  $M$ . Генерація векторних представлень для кожної мітки є обчислюваною та має складність  $O(m \cdot n)$ , де  $m$  — складність моделі  $M$  для обробки одного запиту.

На четвертому кроці ініціалізується порожній набір вихідних міток  $L_o$ . Ця операція обчислювана та виконується за константний час  $O(1)$ .

На п'ятому кроці виконується цикл по кожному векторному представленню  $e$  в  $E_e$ . Оскільки таких ітерацій буде  $n$ , то цикл має складність  $O(n)$ . На підкроці 5.1 обчислюється косинусна відстань між вектором  $e$  та всіма векторами в  $L_o$ . Якщо в  $L_o$  вже є  $q$  векторів, то обчислення відстані має складність  $O(q)$ . На підкроці 5.2 перевірка всіх відстаней та додавання вектора в  $L_o$  за умови, що всі відстані менші за  $\tau$ , має

складність  $O(q)$ . Амортизована складність додавання вектора до набору  $L_o \in O(1)$ . Отже, всі операції для кроку 5 обчислювані та мають складність  $O(n \cdot q)$ .

Отже, загальна складність алгоритму може бути оцінена як  $O(n + m \cdot n + n \cdot q) = O(n \times (1 + m + q)) = O(n \times (m + q))$ .

Таким чином, алгоритм завжди завершується за скінченну кількість кроків і є обчислювально здійсненним, оскільки в алгоритмі відсутні нескінченні цикли або інші перешкоди для його завершення.

#### **4.4 Дослідження впливу запропонованого підходу для навчання мультимодальних моделей для задач розпізнавання на відео**

Усі експериментальні дослідження були проведені на наборі даних Kinetics-400 [57]. Для експериментів використано візуальний енкодер з моделі InternVideo [80] для ініціалізації ваг візуальної частини VLM. Аналогічно, текстовий енкодер з InternVideo [80] використовується для ініціалізації ваг текстової частини VLM. В усіх експериментах був використаний підхід з послідовним навчанням відеоенкодера та текстового енкодера, що унеможливорює проблеми з вибухом або затуханням градієнтів під час навчання даних моделей.

Під час підготовки відеоданих було використано метод розрідженої вибірки, що включає семплювання певної кількості кадрів, позначених як  $T$  (наприклад, 8, 16, 32), що представляють все відео. У всіх експериментах було використано  $\tau = 0,85$  для видалення дублікатів після обчислень алгоритму збагачення лексикону атрибутів. Для гілки передбачення атрибутів кожне відео було розділено на 8 часових фрагментів, що не перекриваються, по 16 кадрів у кожному, та передбачено візуально-текстовою моделлю атрибут для даного фрагменту з найбільшою впевненістю моделі.

Протокол оцінки точності моделі використовує стратегію «ковзного вікна», поширений в області прогнозування на відео [71; 96; 177], що передбачає вибірку кількох кліпів з декількох кадрів на одному відео для досягнення більшої точності. З

метою порівняння з іншими підходами було використано конфігурацію тестування, що включає чотири кліпи з трьома сегментами, позначеними як «View 4×3».

У таблиці 4.1 наведено результати проведеного дослідження. Розмір вхідного тензора вказує на характеристики моделі – кількість кадрів, яку він приймає за один запуск, та їх розмірність у пікселях. Характеристики вікна вказують на кількість часових фрагментів, що не перекриваються, та кількість кадрів у кожному фрагменті. Продуктивність вказує на обчислювальну складність методу. Якщо метод запускається на декількох вікнах, то відповідна кількість вікон вказується як додатковий множник.

Таблиця 4.1 – Порівняння різних методів для детекції дій на відео на наборі даних Kinetics-400 [57]

Метод	Розмір вхідного тензора	Топ-1 (%)	Топ-5 (%)	Характеристика вікна	Обчислювальна складність, GFLOPS	Параметри, $10^6$
NL I3D-101 [181]	128×224 <sup>2</sup>	77,7	93,3	10×3	359×30	61,8
MVFNet [110]	24×224 <sup>2</sup>	79,1	93,8	10×3	188×30	-
TimeSformer-L [116]	96×224 <sup>2</sup>	80,7	94,7	1×3	2380×3	121,4
ViViT-L/16×2 [65]	32×320 <sup>2</sup>	81,3	94,7	4×3	3992×12	310,8
VideoSwin-L [66]	32×384 <sup>2</sup>	84,9	96,7	10×5	2107×50	200,0
Методи, які попередньо навчені на великих наборах даних						
ViViT-L/16×2 [65]	32×320 <sup>2</sup>	83,5	95,5	4×3	3992×12	310,8
ViViT-H/16×2 [65]	32×224 <sup>2</sup>	84,8	95,8	4×3	8316×12	647,5
TokenLearner-L/10 [182]	32×224 <sup>2</sup>	85,4	96,3	4×3	4076×12	450
MTV-H [183]	32×224 <sup>2</sup>	85,8	96,6	4×3	3706×12	-
CoVeR [184]	16×448 <sup>2</sup>	87,2	-	1×3	-	-
Методи на базі VLM, які попередньо навчені на великих наборах даних						
CoCa ViT-giant [70]	6×288 <sup>2</sup>	88,9	-	-	-	2100
VideoPrompt ViT-B/16 [77]	16×224 <sup>2</sup>	76,9	93,5	-	-	-

Продовження таблиці 4.1

Метод	Розмір вхідного тензора	Тор-1 (%)	Тор-5 (%)	Хара ктери стика вікна	Продук тивніст ь, GFLOP S	Парам етри, 10 <sup>6</sup>
ActionCLIP ViT-B/16 [178]	32×224 <sup>2</sup>	83,8	96,2	10×3	563×30	141,7
Florence [81]	32×384 <sup>2</sup>	86,5	97,3	4×3	-	647
ST-Adapter ViT-L/14 [175]	32×224 <sup>2</sup>	87,2	97,6	3×1	2750×3	-
AIM ViT-L/14 [176]	32×224 <sup>2</sup>	87,5	97,7	3×1	3736×3	341
EVL ViT-L/14 [185]	32×224 <sup>2</sup>	87,3	-	3×1	2696×3	-
EVL ViT-L/14 [185]	32×336 <sup>2</sup>	87,7	-	3×1	9098×3	-
X-CLIP ViT-L/14 [56]	16×336 <sup>2</sup>	87,7	97,4	4×3	3086×12	-
Text4Vis ViT-L/14 [179]	32×336 <sup>2</sup>	87,8	97,6	1×3	3829×3	230
BIKE ViT-L/14 [78]	32×336 <sup>2</sup>	88,6	98,3	4×3	3728×12	230
InternVideo-T [80]	32×336 <sup>2</sup>	91,1	99,0	4×3	1434×12	1300
<b>Attr4Vis (Запропонований)</b>	<b>32×336<sup>2</sup></b>	<b>91,5</b>	<b>99,2</b>	<b>4×3</b>	<b>1434×12</b>	<b>1300</b>

Як видно з таблиці 4.1, запропонований метод досягає більшої точності розпізнавання на відео у порівнянні з існуючими методами і при цьому потребує менше обчислень для досягнення результатів.

Проведене експериментальне дослідження (таблиця 4.2) підкреслює важливість запропонованого методу для генерації декількох атрибутів для аналізу відео. Причиною цього є те, що даний підхід дозволяє ефективно кодувати з високою деталізацією зміни атрибутів в часі, що в свою чергу призводить до підвищення точності векторних репрезентацій, згенерованих мовними моделями, та підвищення якості розпізнавання подій.

Таблиця 4.2 – Вплив прогнозування декількох атрибутів на точність.

Підхід	Views	Top-1 (%)
Без прогнозування атрибутів	-	91,1%
Прогнозування одного атрибуту	1×16	91,3% (+0,2%)
Прогнозування декількох атрибутів	4×16	91,4% (+0,3%)
	8×16	91,5% (+0,4%)
	16×16	91,5% (+0,4%)

Аналіз запропонованого алгоритму збагачення лексикону на якість розпізнавання на відео у порівнянні з підходом, який використовує категорії з ImageNet [1] та Kinetics-400 [57], наведено у таблиці 4.3.

Таблиця 4.3 – Вплив різних лексиконів на точність моделі

Лексикон	Top-1 (%)
Без прогнозування атрибутів	91,1%
ImageNet-1K [1]	90,3% (-0,8%)
Kinetics-400 [57]	91,4% (+1,1%)
Запропонований алгоритм збагачення лексикону	91,5% (+0,1%)

З таблиці 4.3 можна зробити висновок, що використання довільного лексикону (ImageNet-1K [1]) призводить до погіршення якості на 0,8%, але використання лексикону з набору даних, на якому виконується прогнозування (Kinetics-400), підвищує якість на 1,1% навіть за умови, що кількість атрибутів менша (400 у порівнянні з 1000).

Додатково було досліджено важливість аспекту ініціалізації класифікатора в гілці Text-to-Video, кроку, який відіграє ключову роль у оптимізації структури VIKE [152]. Зміна підходу до ініціалізації класифікатора в гілці Text-to-Video дозволила відмовитися від механізму Video Concept Spotting – однієї з найскладніших частин архітектури VIKE

[152], що дозволило таким чином спростити архітектуру BIKE [152] та уможливити інтеграцію запропонованого підходу в існуючі архітектури візуально-мовних моделей без змін у базових візуально-мовних моделях. Результати аналізу з використанням різних ініціалізацій класифікатора, базових архітектур візуально-мовних моделей, прогнозування атрибутів та запропонованого алгоритму збагачення лексикону наведено в таблиці 4.4.

Таблиця 4.4 – Вплив різних компонентів в гілці Text-to-Video на якість підходу

<b>Зміна</b>	<b>Тор-1 (%) та приріст відносно попередньої зміни</b>
BIKE [152]	88,6%
Змінено нейромережу на InternVideo-T [80]	90,6% (+2,0%)
Додано Text4Vis initialization [179]	90,8% (+0,2%)
Прибрано Video Concept Spotting з BIKE[78]	91,2% (+0,4%)
Додано прогнозування декількох атрибутів	91,4% (+0,2%)
Додано алгоритм збагачення лексикону	91,5% (+0,1%)

#### 4.5 Висновки до розділу

У даному розділі представлено інноваційний підхід Attr4Vis, який націлений на вдосконалення передачі знань між візуальними та текстовими модальностями візуально-мовних (VLM) нейронних мереж для підвищення точності розпізнавання на відео. Підкреслено важливість генерації декількох атрибутів для підвищення точності роботи методу. Доведено експериментально, що цей підхід дозволяє ефективно кодувати зміни атрибутів з часом, сприяючи таким чином більш точному представленню та розпізнаванню подій. Представлено новий алгоритм збагачення лексикону, призначений для розширення масиву атрибутів, пов'язаних із відеоданими. Це розширення збільшує



можливості для розпізнавання атрибутів у наборах відеоданих, тим самим поліпшуючи описові можливості візуально-мовних моделей. Крім того, було досліджено важливість аспекту ініціалізації класифікатора в гілці Text-to-Video та видалення механізму Video Concept Spotting, що дозволило оптимізувати структуру підходу BIKE [78] та підвищити якість розпізнавання.

Запропонований підхід Attr4Vis, об'єднуючи всі вищезгадані інновації, досягає точності 91,5% Top-1 на наборі даних Kinetics-400, що є більш високим результатом у порівнянні з існуючими підходами. Запропонований підхід має широкі перспективи для вивчення передачі знань між візуальною та текстовою модальностями VLM моделей та одночасно підвищує точність розпізнавання атрибутів.

Дані результати були опубліковані у науковому журналі International Journal of Computing [180].

## **5 АРХІТЕКТУРА ТА КОМПОНЕНТИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСУ РОЗМІТКИ ВІДЕОДАНИХ**

### **5.1 Інструменти розмітки зображень**

У сфері технологій комп'ютерного зору інструменти розмітки зображень мають ключове значення, суттєво сприяючи точному маркуванню та позначенню тегоми даних.

Організації, які прагнуть ефективно керувати своїми даними, мають доступний широкий набір програмних засобів розмітки зображень такі як V7, LabelBox, Keylabs, LableMe та інші. Функціональні можливості цих інструментів охоплюють різноманітні функції, такі як позначення зображень і відео, текстові розмітки та позначення аудіо.

Точність набуває першочергового значення в проєктах розмітки зображень через її значний вплив на якість наборів даних та ефективність моделей машинного навчання. Отже, пріоритетність використання потужних і точних інструментів розмітки стає обов'язковою.

Автоматизований інструмент розмітки V7 [186] спрощує процес розмітки зображень для таких завдань, як сегментація зображень, класифікація зображень і класифікація дій на відео тощо. V7 забезпечує зручний інтерфейс (рис. 5.1) і низку інструментів для розмітки, щоб спростити процес розмітки та підвищити точність. За допомогою V7 користувачі можуть коментувати набори даних, керувати розміткою, тренувати моделі та співпрацювати з членами команди в межах однієї платформи.

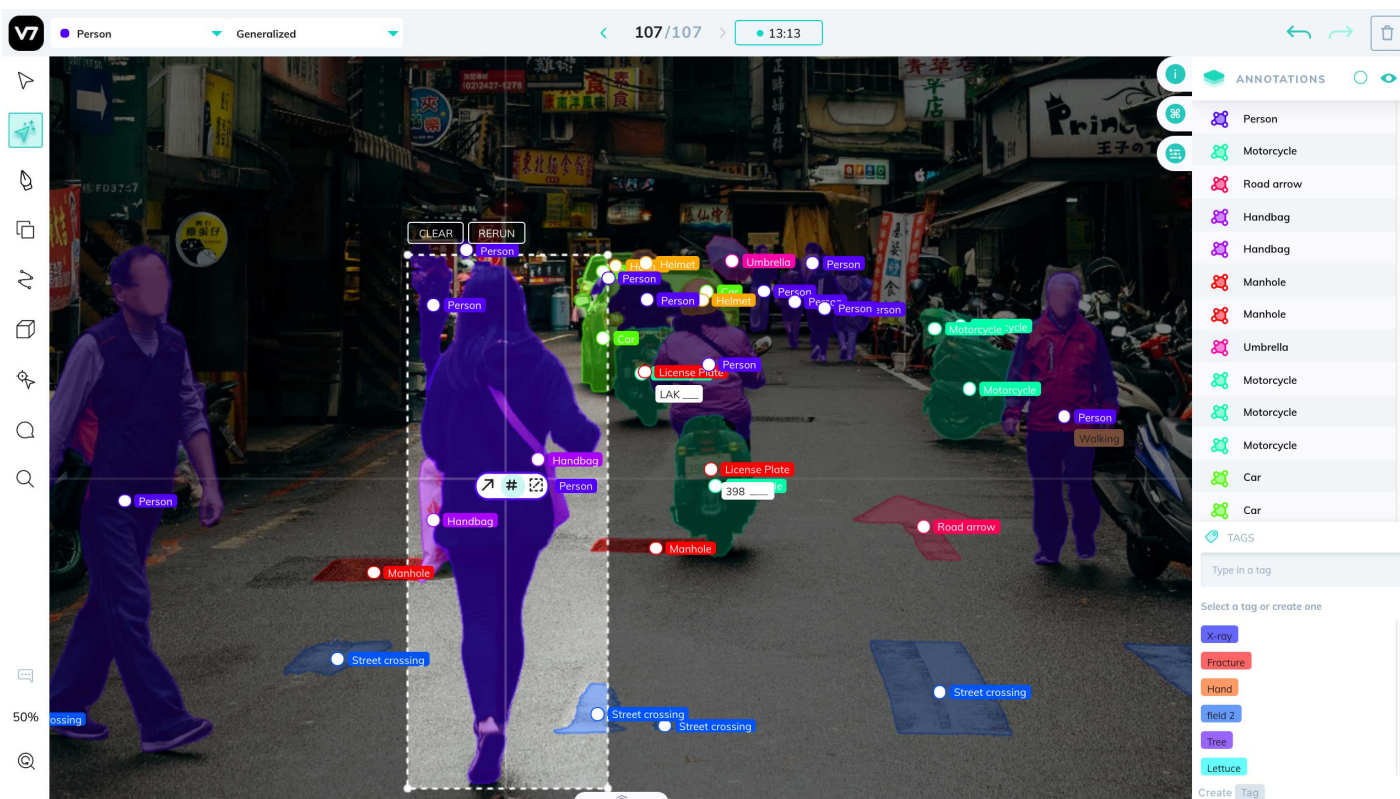


Рисунок 5.1 – Користувачський інтерфейс V7 [186]

До переваг програмного засобу V7 відносяться [187]:

- можливість розмічати зображення для заданих класів та навчати моделі для заданих класів;
- можливість точно й швидко розмітити будь-які об'єкти за рахунок використання інструментів для розмітки, таких як обмежувальні рамки, багатокутники, ключові точки та каркасні інструменти для виконання різноманітних завдань щодо дій з об'єктами;
- можливість розмічати об'єкти та регіони за допомогою багатокутника, який дає змогу точно описати складні форми та межі будь-якого об'єкту, та використовувати такі функції, як приближення зображення та міжрівнева синхронізація розмітки;

- можливість автоматизованого створення семантичної сегментації на зображеннях на основі інформації, введеної користувачем;
- можливості керування даними, включаючи керування наборами даних, контроль версій та завантаження даних у різних форматах;
- можливості для спільної роботи над розміткою набору даних для кількох розмітчиків за рахунок відстежування індивідуальної продуктивності, аналізу виконаної розмітки та поширення відгуків усіх учасників процесу;
- широкий спектр форматів даних зображень і розмітки, включаючи набори даних магнітно-резонансної томографії, у популярних форматах, таких як COCO, Pascal VOC і YOLO;
- інтеграція з хмарними платформами, такими як AWS, Google Cloud і Azure, що дає змогу використовувати масштабованість і ресурси, які пропонують ці платформи.

Отже, V7 має найвищі показники ефективності, функціональності та форматування даних. Проте, цей програмний засіб дозволяє розмітити лише 100 одиниць даних безоплатно, що є значним обмеженням для даного дисертаційного дослідження, тому V7 не був використаний.

LabelBox — це платформа для розмітки даних, зокрема для потреб машинного навчання та проєктів комп'ютерного зору, що пропонує універсальний та інтуїтивно зрозумілий інтерфейс (рис. 5.2 [188]). Цей програмний засіб підтримує різні формати даних, такі як JPEG, PNG, BMP і TIFF, забезпечуючи сумісність із широким діапазоном файлів зображень. Платформа надає низку функцій, призначених для підвищення ефективності та точності розмітки. Однією з особливостей є здатність підтримувати маркування як для зображень, так і для відео, враховуючи різноманітні модальності візуальних даних. Крім того, LabelBox містить функції текстової розмітки, що дозволяє

розмічати текстові елементи в межах даних. Цей підхід до розмітки має ключове значення для вирішення складних завдань машинного навчання.

На додаток до цих функцій, LabelBox містить можливості семантичної сегментації, що дозволяє детально окреслити та розмітити межі об'єктів на зображеннях. Це полегшує створення високоякісних наборів даних, необхідних для навчання складних моделей машинного навчання.

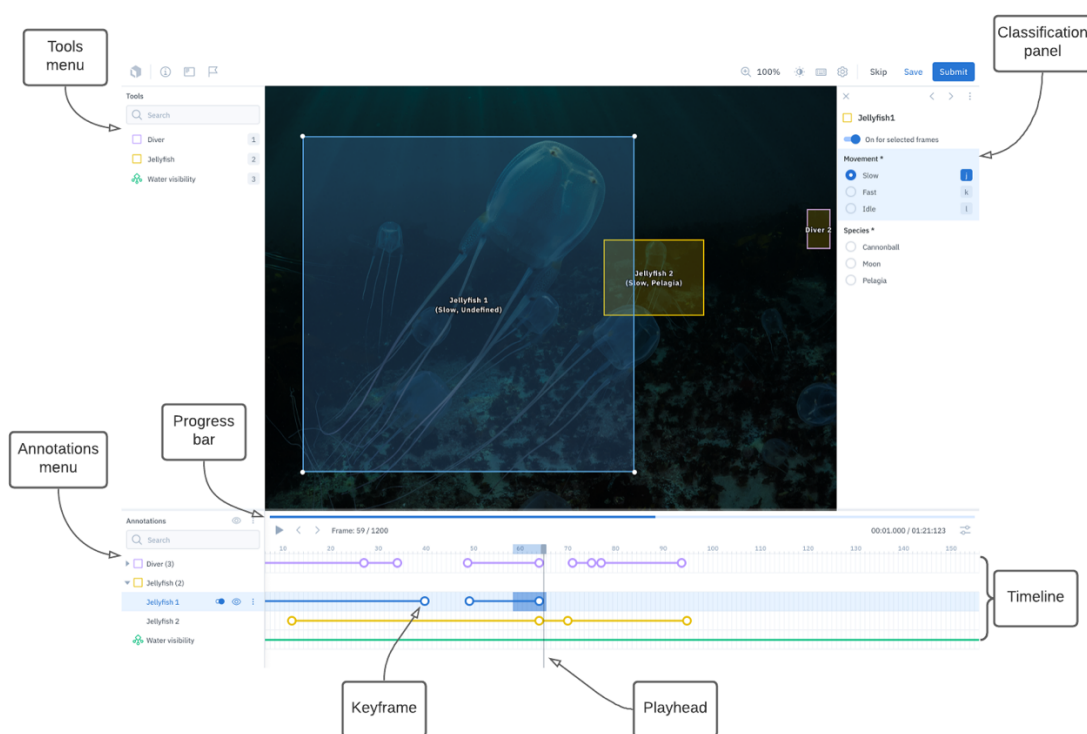


Рисунок 5.2 – Користувацький інтерфейс інструменту LabelBox [188]

Інструменти LabelBox для спільної роботи та керування проектами сприяють постійному спілкуванню та координації між різними розмітчиками даних та зацікавленими сторонами. При цьому платформа надає пріоритет точності створення розмітки. Отже, дана платформа є однією з найбільш ефективних, функціональних та чітко структурованих інструментів, відповідаючи потребам створення моделей різних масштабів. Проте, наявні значні обмеження щодо безкоштовного використання

LabelBox, які полягають у ліміті розмітки даних у 10000 одиниць, які, без сумніву, представляють суттєве обмеження для досліджень, що є предметом даної дисертації. Тому LabelBox не був використаний.

Програмний засіб для розмітки даних Keylabs [189] пропонує низку функцій для проведення процесу розмітки. Користувацький інтерфейс Keylabs наведено на рисунку 5.3 [189]. Keylabs вирізняється ефективністю, функціональністю та форматуванням даних, створюючи оптимальні умови для створення та вдосконалення моделей машинного навчання в різних масштабах.

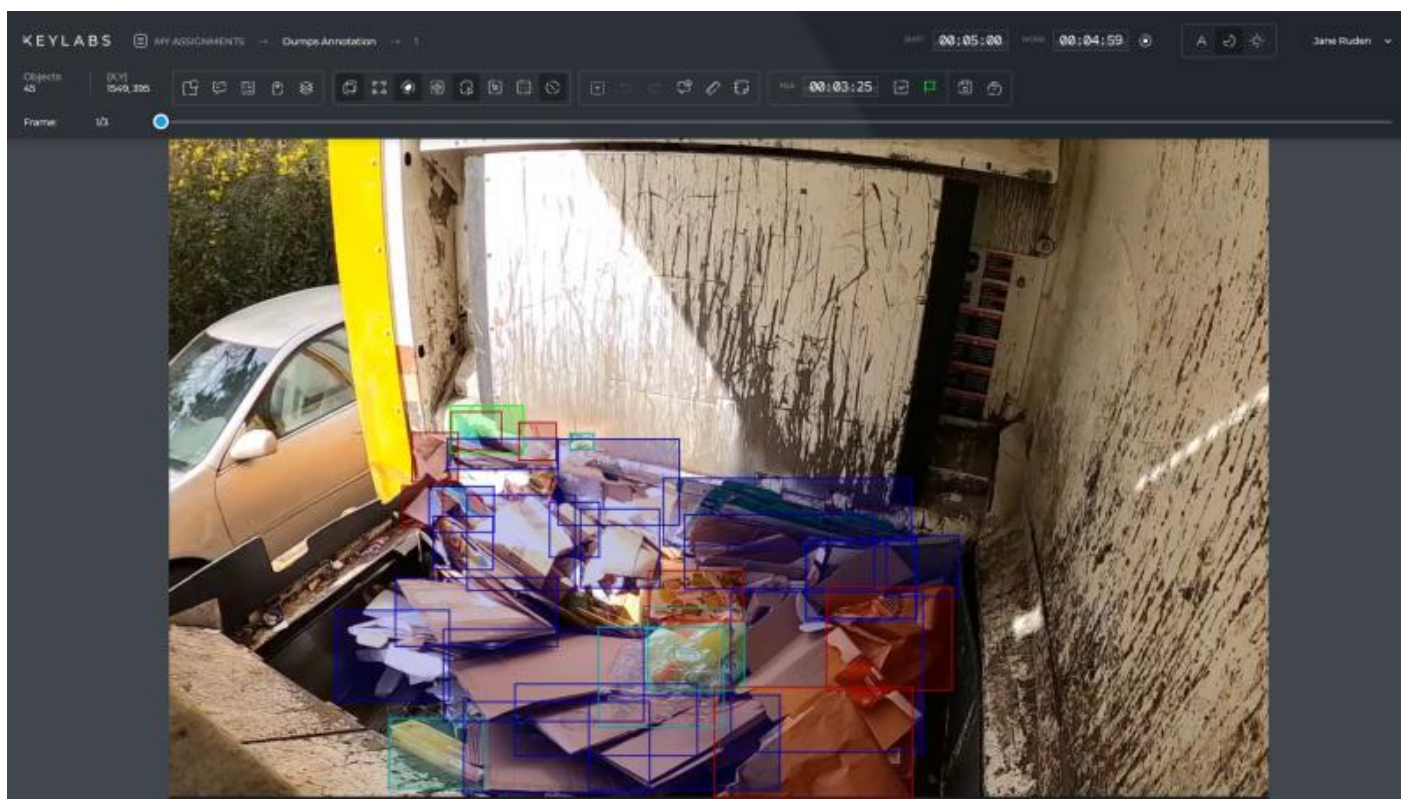


Рисунок 5.3 – Користувацький інтерфейс інструмента Keylabs [189]

З точки зору функцій, програмний засіб Keylabs демонструє високі показники ефективності у виконанні завдань семантичної сегментації, особливо застосовних у сфері медичної візуалізації. Цей програмний засіб надає можливість розмічати широкий спектр типів даних, включаючи зображення, відео, тексти та аудіо. Keylabs підтримує

різноманітні формати даних, такі як JPEG, PNG, BMP і TIFF, полегшує процес створення розмітки без необхідності попереднього перетворення даних. Крім того, точність та якість створеної розмітки є достатньо високими. Спеціалізуючись на завданнях сегментації, що вимагають високої точності, Keylabs має переваги у сфері розмітки медичних зображень. Однак слід визнати, що застосовність Keylabs та його економічна ефективність можуть залежати від характеру розмітки.

Таким чином, можна зробити висновок, що програмний засіб Keylabs має посередні результати щодо функціональності, ефективності та форматування результатів роботи. Відсутність можливості користуватися ним без попередньої покупки робить його таким, що не підходить для використання в рамках дисертаційного дослідження.

Програмний засіб LabelImg [190] представляє собою безкоштовний відкритий інструмент призначений для проведення розмітки зображень та виділення об'єктів на них, який реалізований на мові програмування Python. Результат розмітки може бути збережений у вигляді XML-файлів, аналогічних формату PASCAL VOC. LabelImg забезпечує можливість створення обрамляючих прямокутників для розмітки об'єктів, використовуючи графічний інтерфейс Qt, який проілюстровано на рисунку 5.4 [190].

Початок роботи з LabelImg передбачає певний рівень технічної обізнаності, зокрема необхідними є навички використання командного рядка. Для встановлення LabelImg потрібно ввести у командному рядку команду для пакетного менеджера `pip` "`pip install labelImg`". Після встановлення, керування LabelImg здійснюється через командний рядок.



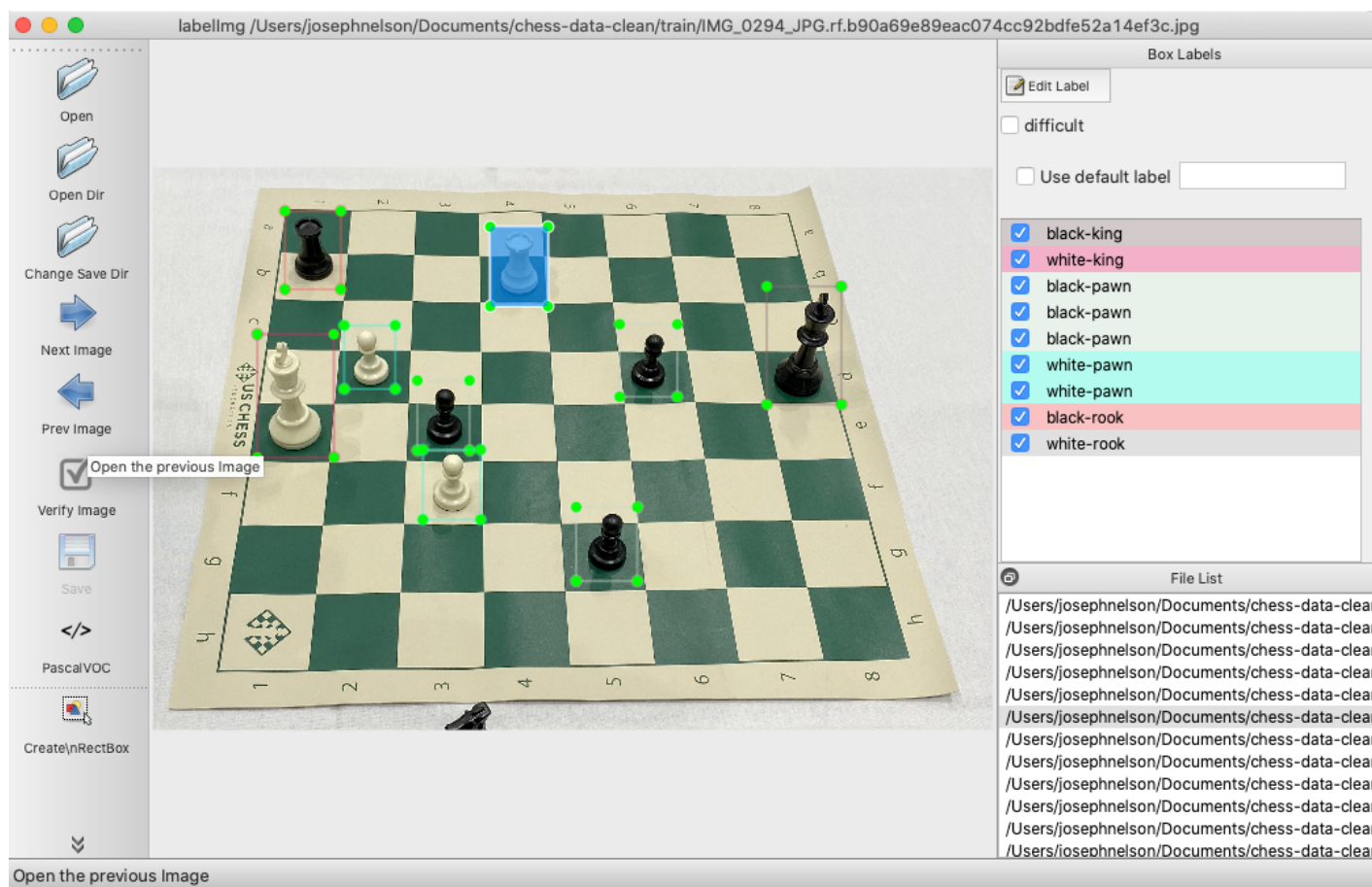


Рисунок 5.4 – Користувачький інтерфейс інструменту LabelImg [190]

Програмний засіб LabelImg є ефективним інструментом для невеликих компаній, оскільки дозволяє розпочати процес розмітки даних без створення складної інфраструктури та працевлаштування персоналу, спроможного утримувати її в робочому стані. Інтерфейс LabelImg відзначається інтуїтивністю, що полегшує початок процесу розмітки даних. Однак серед недоліків даного інструмента можна зазначити відсутність системи версіонування розмітки та складнощі при розмітці зображень із великою кількістю категорій або об'єктів.

Таким чином, даний програмний засіб отримує найнижчі показники з функціональності, ефективності розмітки та кількості підтримуваних форматів даних. Проте, на відміну від вищезгаданих інструментів, він безкоштовний, що робить його одним із кандидатів на використання в рамках дисертаційного дослідження.



Програмний засіб LabelMe [191] розроблений як графічний інструмент із відкритим кодом, переважно написаним на мові програмування Python, що позиціонує його як універсальне рішення для розмітки об'єктів на зображеннях. Однією з його переваг є здатність зберігати розмітку в файлах XML, узгоджуючи їх із форматом, що нагадує стандарт PASCAL VOC.

Загальний дизайн (рис. 5.5 [191]), і функціонал LabelMe спрямовані на оптимізацію процесу розмітки, особливо у проєктах, для яких швидкий початок розмітки даних має вирішальне значення. Визначною особливістю LabelMe є здатність створювати обмежувальні рамки, полегшуючи розмітку об'єктів через графічний інтерфейс, що реалізований засобами Qt. Цей графічний інтерфейс забезпечує зручну інтуїтивно зрозумілу взаємодію дослідників та розмітчиків даних.

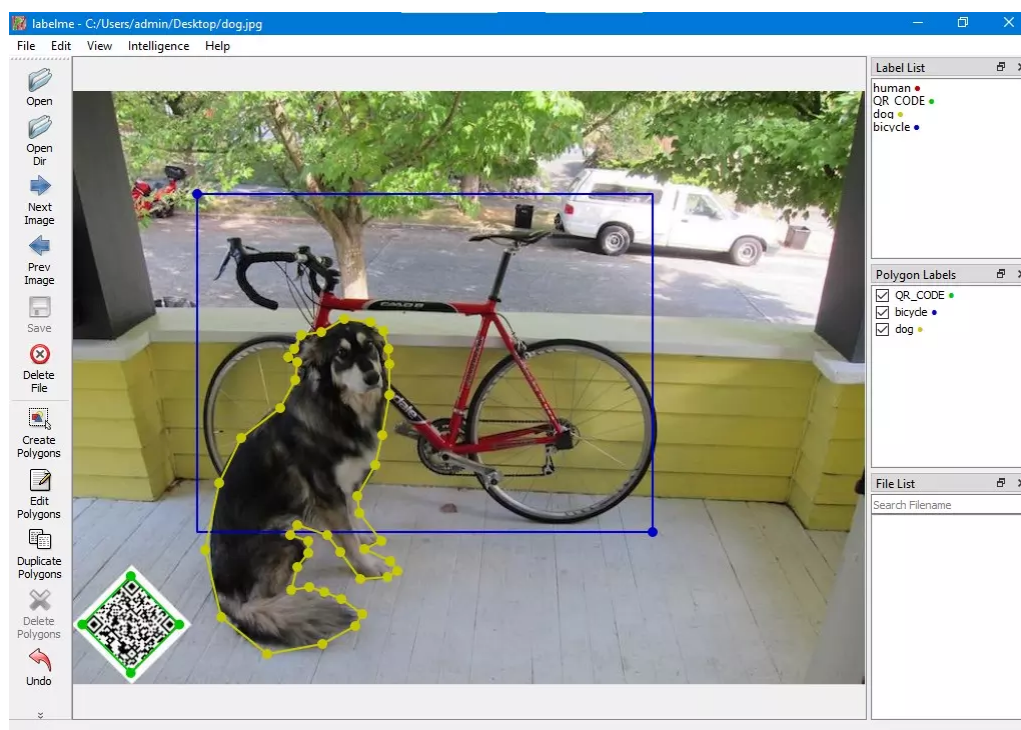


Рисунок 5.5 – Користувацький інтерфейс інструменту LabelMe [191]

Щоб розпочати використання LabelMe, потрібен базовий рівень технічних навичок, включаючи знання командного рядка. Інструмент можна легко розгорнути за допомогою менеджера пакунків Python pip, за допомогою команди «pip install LabelMe». Після встановлення LabelMe стає доступним через командний рядок. LabelMe виділяється своєю простотою, що дає змогу швидко розуміти його функціональні можливості та ініціювати розмітку даних. Разом з тим, слід визнати певні обмеження, зокрема відсутність контролю версій та проблеми з розміткою зображень із значною кількістю категорій об'єктів. Незважаючи на ці обмеження, LabelMe є достатньо ефективним інструментом розмітки, полегшуючи робочий процес розмітки зображень.

Виходячи з усього вищезгаданого можна зробити висновок, що LabelMe має посереднє значення функціональності та вузький список підтримуваних форматів, особливо відео, проте є повністю безкоштовним. Оскільки функціональності даного інструменту не достатньо, щоб організувати ефективний процес розмітки відеоданих, то він не був використаний.

Програмний засіб Label Studio [192] є комплексною платформою для розмітки даних із відкритим вихідним кодом. Цей програмний засіб вирізняється універсальністю та надійним набором функцій, слугує для різних типів розмітки, включаючи класифікацію зображень, детекцію та сегментацію об'єктів, та надає широкий набір форматів розмітки, такі як JSON, XML та інші, що полегшує інтеграцію в різноманітні конвеєри та робочі процеси машинного навчання. Ключова перевага Label Studio полягає у підтримці активного навчання, надаючи засоби для розумного вибору та розмітки тих даних, які важливі для підвищення точності моделі машинного навчання. Це сприяє тому, що процес розмітки керується даними, що розмічаються, оптимізуючи використання ресурсів і прискорюючи процес навчання моделі.

На додаток, Label Studio вирізняється масштабованістю, що дозволяє ефективно розмічати великі набори даних. Ця масштабованість доповнюється надійною та розширюваною архітектурою, що дає змогу адаптувати інструмент до вимог проєкту.

Крім того, відданість Label Studio принципам відкритого вихідного коду сприяє співпраці, заохочуючи користувачів обмінюватися робочими процесами, розширеннями та найкращими практиками.

Інтерфейс Label Studio, який зображено на рисунку 5.6 [192], пропонує інтуїтивно зрозуміле середовище, яке сприяє ефективному проведенню розмітки даних.

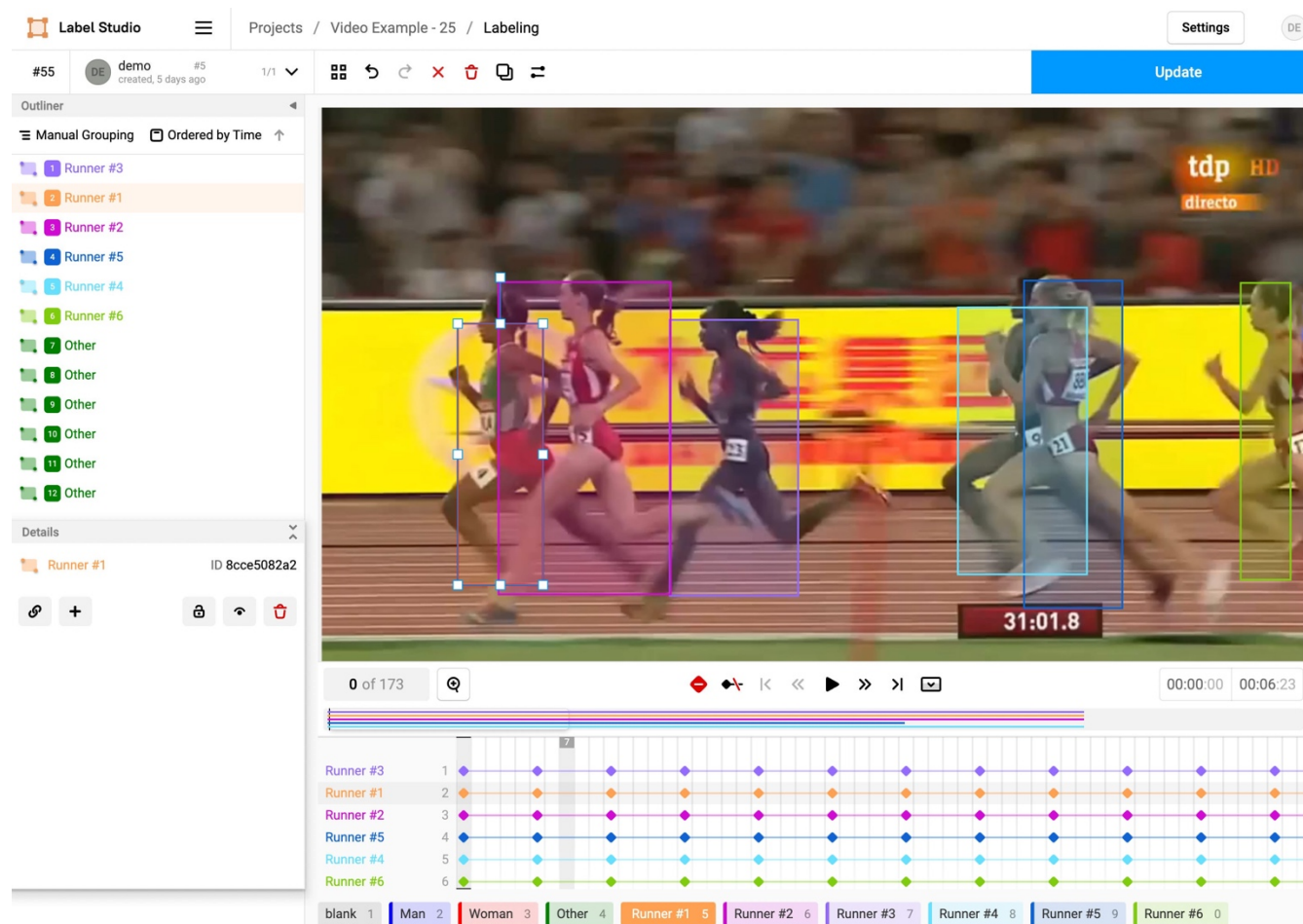


Рисунок 5.6 – Користувачський інтерфейс інструменту Label Studio [192]

Окрім основних функцій інструментів розмітки, програмний засіб Label Studio містить також інструменти для спільної роботи, які дозволяють кільком розмітчикам працювати одночасно. Універсальність платформи додатково підсилюється підтримкою

завдань текстової розмітки, забезпечуючи застосовність інструменту для різних типів модальностей.

Підсумовуючи, Label Studio постає як комплексний інструмент розмітки, який пропонує універсальне та зручне середовище для розмітки різноманітних типів даних. Його підтримка різних форматів розмітки, функції спільної роботи, можливості активного навчання та масштабованість сприяють його статусу передового інструменту в галузі комп'ютерного зору та машинного навчання.

Виходячи з усього вищезгаданого можна зробити висновок, що Label Studio підходить для даного дисертаційного дослідження. Проте, частково закритий вихідний код Label Studio робить його менш бажаною альтернативною у порівнянні з іншими інструментами.

Програмний засіб Computer Vision Annotation Tool (CVAT) [193] розроблений для полегшення розмітки зображень і відео та адаптований до завдань комп'ютерного зору. Він підтримує різноманітні формати даних, забезпечуючи сумісність із різними розширеннями зображень і відеоданих, та пропонує низку особливих функцій, які можна застосовувати як для розмітки зображень, так і відео. Програмний засіб підтримує обмежувальні рамки, багатокутники та ключові точки, забезпечуючи гнучкість у захопленні різноманітних візуальних елементів. Крім того, CVAT сприяє розподіленому процесу створення розмітки, дає змогу кільком розмітчикам працювати одночасно, сприяє ефективній командній роботі та швидкому створенню набору даних. З точки зору сумісності форматів даних, програмний засіб підтримує усі широко використовувані формати.

Програмний засіб CVAT має інтуїтивно зрозумілий інтерфейс, який спрощує процес створення розмітки. Він підтримує як ручну, так і автоматичну розмітку даних. CVAT розширюється та налаштовується, що дає змогу адаптувати інструмент до конкретних потреб проекту та сприяє його використанню у різноманітних проєктах комп'ютерного зору. Користувацький інтерфейс CVAT наведено на рисунку 5.7 [193].

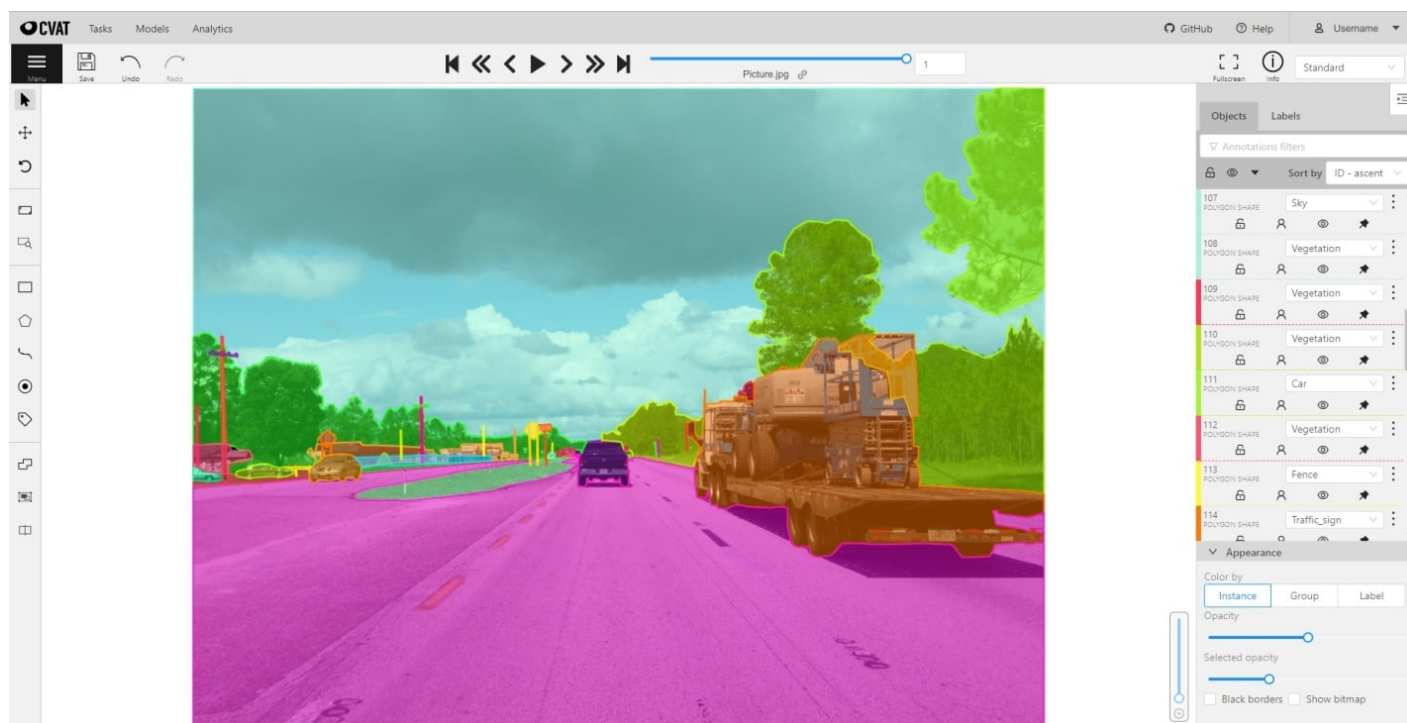


Рисунок 5.7 – Користувачський інтерфейс інструменту CVAT [193]

Отже, CVAT отримує найвищі характеристики з функціональності, ефективності, підтримки різноманітних форматів та оплати використання. Тому було обрано цей програмний засіб для виконання завдань дисертаційного дослідження.

## 5.2 Порівняльний аналіз інструментів розмітки даних

Порівняльний аналіз програмних засобів розмітки даних V7, LabelBox, Keylabs, LabelImg, LabelMe, Label Studio та CVAT, який наведено у таблиці 5.1, демонструє значну різницю в їх функціональних можливостях та архітектурних підходах, що істотно впливає на ефективність розмітки, гнучкість використання, підтримку командної роботи та здатність до інтеграції з алгоритмами машинного навчання.

Таблиця 5.1 – Порівняльний аналіз інструментів розмітки даних для задач комп'ютерного зору

<b>Критерій</b>	<b>V7</b>	<b>Label Box</b>	<b>Key Labs</b>	<b>Label Img</b>	<b>LabelMe</b>	<b>Label Studio</b>	<b>CVAT</b>
Платформа	Веб	Веб	Веб	Десктоп (Python)	Десктоп (Python)	Веб, Десктоп	Веб
Підтримка типів даних	Зображення, Відео, 3D	Зображення, Відео	Зображення, Відео	Тільки зображення	Тільки зображення	Зображення, Відео, Текст	Зображення, Відео
Інтеграція ML моделей	Так, інтеграція з V7 Darwin	Так, через API	Ні	Ні	Ні	Так, через API	Так, через API
Формати експорту даних	COCO, Pascal VOC, YOLO, JSON	COCO, Pascal VOC, YOLO, JSON	Pascal VOC, JSON	Pascal VOC, YOLO	Pascal VOC, JSON	COCO, Pascal VOC, YOLO, JSON, CSV	COCO, Pascal VOC, YOLO, JSON
Підтримка командної роботи	Так	Так	Так	Ні	Ні	Так	Так
Масштабованість	Висока	Висока	Середня	Низька	Низька	Висока	Висока
Гнучкість кастомізації	Низька	Середня	Низька	Висока	Висока	Висока	Висока
Підтримка анотацій для відео	Так	Так	Так	Ні	Ні	Так	Так
Відкрита ліцензія	Ні	Ні	Ні	Так	Так	Так	Так (Apache License 2.0)

Ключовою відмінністю, яка виділяє CVAT серед конкурентів, є його безкоштовний доступ і відкритий код під ліцензією Apache License 2.0, що дозволяє

широке використання та модифікацію інструменту, зокрема для академічних проєктів та малих команд, обмежених у фінансових ресурсах. Це надає CVAT значну перевагу у порівнянні з комерційними платформами, такими як V7 та LabelBox, де деякі інструменти можуть бути недоступні без попередньої оплати.

Крім того, програмний засіб CVAT забезпечує інтеграцію з моделями машинного навчання, що є важливою особливістю, оскільки така інтеграція дозволяє суттєво автоматизувати та прискорити процес розмітки. Підтримка таких платформ, як OpenVINO та TensorFlow, дає змогу CVAT не лише обробляти дані швидше, але й робить його конкурентоспроможним інструментом для великих проєктів, де необхідне масштабування обробки даних і постійне залучення моделей для автоматичної розмітки зображень та відео. Важливим фактором є також підтримка відеорозмітки, яку забезпечує CVAT, що дає йому перевагу перед такими десктопними інструментами, як LabelImg та LabelMe, які обмежені лише підтримкою зображень. CVAT дозволяє ефективно розмічати довгі відео з використанням анотацій на різних кадрах, що є важливим для задач комп'ютерного зору, де об'єкти можуть змінювати своє положення в часі, і потрібен високий рівень контролю над усіма аспектами анотацій.

Ще однією значною перевагою CVAT є його масштабованість та підтримка командної роботи, що дає можливість великим проєктам розподіляти задачі між кількома учасниками, зберігаючи єдиний формат даних та стандарти якості. У порівнянні з інструментами LabelBox та Keylabs, які також підтримують командну роботу, CVAT відзначається більшою гнучкістю завдяки можливості адаптації коду, що дозволяє реалізовувати специфічні функції для кожного окремого проєкту. Висока гнучкість кастомізації є однією з ключових характеристик, що вирізняє CVAT серед інших інструментів, дозволяючи адаптувати інтерфейс та функціонал інструменту під специфічні вимоги користувачів.

Таким чином, CVAT представляє собою багатofункціональний інструмент з широким спектром можливостей, орієнтованим на розв'язання складних задач розмітки



даних з високою точністю. Здатність до інтеграції з ML-моделями, підтримка командної роботи, гнучкість кастомізації та доступність для користувачів роблять його надзвичайно ефективним вибором для використання у великих проєктах, які вимагають високого рівня автоматизації та мають великий обсяг даних для обробки.

### 5.3 Архітектура інструментів розмітки даних

Оскільки більшість інструментів розмітки даних схожі один на одного, то тип архітектури таких застосунків наведемо на прикладі CVAT [193]. Цей інструмент розмітки комп'ютерного зору демонструє детально розроблену архітектуру програмного забезпечення (рисунок 5.8 [193]), що вдало поєднує ключові принципи розробки програмного забезпечення для розмітки даних у сфері комп'ютерного зору.

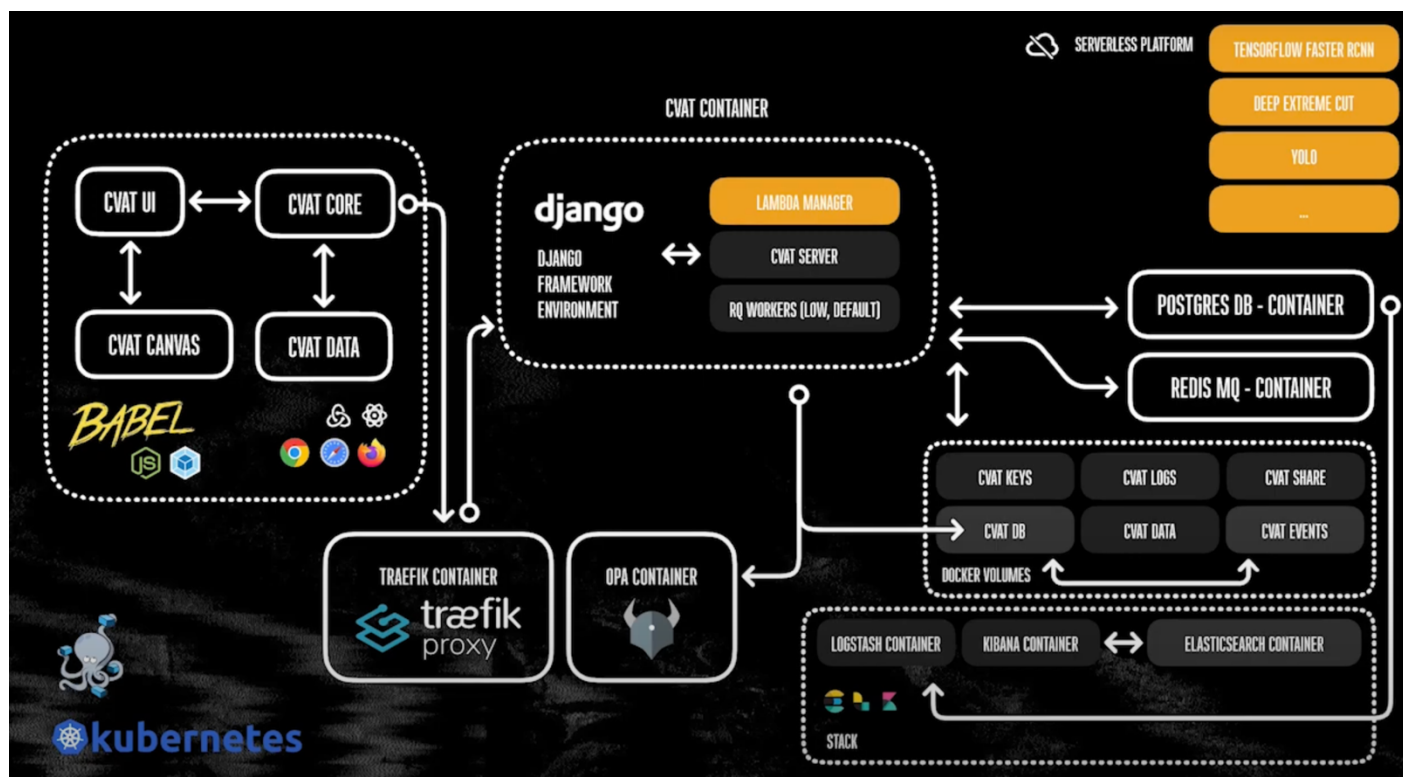


Рисунок 5.8 – Архітектура інструменту CVAT [193]



З точки зору програмного забезпечення архітектура CVAT складається з кількох компонентів, кожен з яких сприяє функціональності та ефективності інструменту:

- CVAT UI: інтерфейс користувача для взаємодії з інструментом;
- CVAT Core: основна логіка програми;
- CVAT Canvas: компонент для роботи з візуальними даними;
- CVAT Data: система для управління даними;
- Babel для трансформації JavaScript коду;
- Django-сервер;
- Lambda Manager: компонент для управління асинхронними задачами;
- CVAT Server: серверна частина, яка обробляє запити від користувачів;
- RQ Workers: сервіси для обробки асинхронних задач;
- Postgres DB - Container: контейнер для бази даних Postgres;
- Redis MQ - Container: контейнер для черги повідомлень Redis;
- Traefik Container: контейнер для реверс-проксі Traefik;
- OPA Container: контейнер для політик Open Policy Agent;
- Logstash Container: контейнер для обробки логів Logstash;
- Kibana Container: контейнер для візуалізації даних Kibana;
- Elasticsearch Container: контейнер для пошуку та аналітики Elasticsearch;
- Docker Volumes: зберігання даних у вигляді Docker томів, що включають CVAT Keys, CVAT Logs, CVAT Share, CVAT DB, CVAT Data, CVAT Events;
- CVAT Keys: управління ключами доступу;
- CVAT Logs: логи системи;
- CVAT Share: спільний доступ до даних;
- CVAT DB: база даних CVAT;
- CVAT Data: дані для розмітки;
- CVAT Events: події системи.

В основі CVAT орієнтована на мікросервіси архітектура, що окреслює окремі, слабо пов'язані компоненти, які інкапсулюють певні функціональні можливості, сприяючи модульності, масштабованості та зручності в підтримці програмного забезпечення. Внутрішні мікросервіси відповідають за керування процесом розмітки, включаючи зберігання даних, автентифікацію користувачів, керування проєктами та оркестрування завдань розмітки. Передача даних у CVAT полегшується системою керування базою даних, яка базується на технології реляційної бази даних PostgreSQL. Це забезпечує структуроване та організоване зберігання розмічених даних, забезпечуючи швидкий пошук і маніпуляції з даними. Інтерфейс CVAT створено з використанням сучасних вебтехнологій з акцентом на інтуїтивно зрозумілий інтерфейс користувача. Використовуючи фреймворки, такі як React.js та Angular.js, інтерфейс динамічно взаємодіє з сервером через чітко визначені API. Графічний інтерфейс користувача розроблено з урахуванням потреб розмітчиків даних, забезпечуючи малювання багатокутників та створення обмежувальної рамки.

Архітектура CVAT приділяє значну увагу керуванню користувачами та безпечному доступу до інструменту розмітки та даних розміщених на ньому. Крім того, CVAT використовує технології контейнеризації, такі як Docker, що забезпечує безпроблемне і нескладне розгортання в різноманітних обчислювальних середовищах. Інструменти оркестровки контейнерів, такі як Kubernetes, можна використовувати для оптимізації керування та масштабування екземплярів CVAT. З погляду оптимізації продуктивності, CVAT включає такі підходи, як кешування та асинхронну обробку. Використання стратегій кешування зменшує надлишкові обчислення, тоді як асинхронна обробка дозволяє паралельно виконувати неблокуючі завдання.

Підсумовуючи, архітектура інструменту розмітки CVAT втілює найліпші підходи в побудові програмного забезпечення, у тому числі мікросервісну архітектуру, контейнеризацію, контроль доступу на основі ролей та адаптивні інтерфейсні технології. Така архітектура забезпечує застосовність інструменту для різноманітних сценаріїв

розмітки, масштабованість, безпеку та зручність обслуговування, таким чином створюючи умови для найбільш ефективного рішення серед інструментів розмітки комп'ютерного зору. У ході дослідження було вирішено зосередити увагу на архітектурі CVAT через її виняткову комплексність та адаптивність. Завдяки своїм різноманітним компонентам архітектура CVAT демонструє як правильно поєднувати модульність, масштабованість та зручність у підтримці програмного забезпечення для створення успішного інструменту розмітки даних. Однією з основних причин вибору CVAT є відкритість вихідного коду та доступність для аналізу. Відкритий код CVAT дає змогу детально дослідити внутрішню структуру та принципи роботи системи. Це дало можливість зрозуміти, як кожен компонент архітектури взаємодіє з іншими та як вони разом забезпечують функціональність та ефективність інструменту. Особлива увага приділена архітектурі CVAT також тому, що вона застосовує сучасні підходи до розробки програмного забезпечення, такі як мікросервіси, контейнеризація та асинхронна обробка. Ці підходи дозволили розробити систему, яка є не тільки ефективною, але й легко масштабованою та адаптованою до змін у вимогах користувачів та обчислювальних середовищах.

#### **5.4 Підходи до автоматизації розмітки даних в наявних інструментах розмітки даних для задач комп'ютерного зору**

Підходи до автоматизації процесу розмітки даних можна поділити на наступні категорії:

- мануальний процес розмітки даних, який характеризується відсутністю автоматизації;
- використання попередньо навчених методів комп'ютерного зору для автоматизації процесу розмітки даних, які містять широко розповсюджені категорії або використання моделей нульового навчання;

- використання підходів активного донавчання моделей комп'ютерного зору на результатах розмітки даних.

Мануальний підхід до розмітки даних для задач комп'ютерного зору, що зображений на рисунку 5.9, передбачає, що всі зображення, які наявні в наборі даних, будуть переглянуті людиною, яка відмітить усі наявні на зображенні категорії.

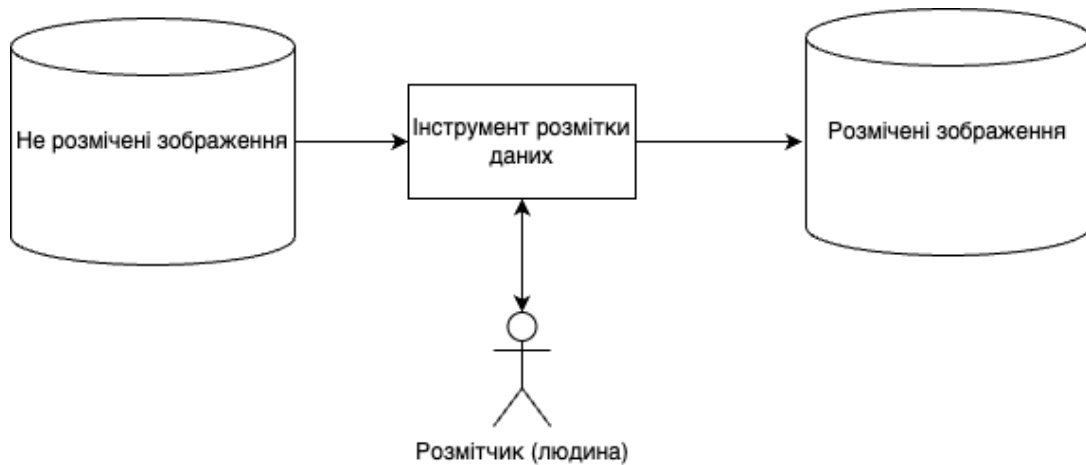


Рисунок 5.9 – Мануальний підхід до розмітки

Мануальний підхід можна оптимізувати шляхом використання методів комп'ютерного зору, які піддаватимуться навчанню на вже розмічених даних (процес донавчання на розмічених даних) та поступово здатні будуть надавати рекомендації стосовно виявлення того чи іншого об'єкта на зображенні. У рамках такого підходу користувачеві необхідно буде лише підтверджувати припущення методу стосовно присутності об'єкта, замість того, щоб самостійно розмічати його. Альтернативно, можливо використовувати методи комп'ютерного зору з нульовим навчання, що дозволяє їх широке використання на даних, на яких такі моделі не навчалися, але недоліком таких моделей є нижча точність у порівнянні з аналогами, які донавчаються. Такий процес розмітки даних представлений схематично на рисунку 5.10. Зазначений підхід на даний момент є переважним у популярних інструментах для розмітки даних.

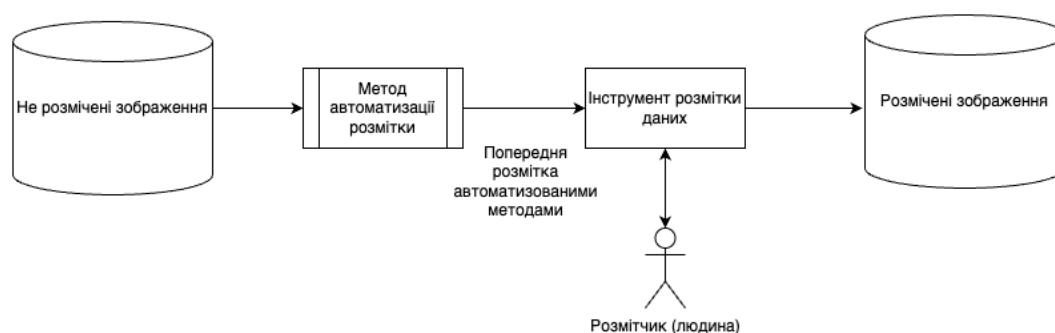


Рисунок 5.10 – Підхід з використанням попередньо навчених методів машинного навчання для пришвидшення процесу розмітки даних

Застосування активного навчання моделей на основі результатів вже розмічених даних дозволяє системі визначати ті дані, які вимагають додаткової уваги та розмітки, з метою підвищення точності моделей. Під час активного навчання моделі визначають неоднозначні чи складні приклади, що можуть збагатити варіативність об'єктів та сцен в розміченому наборі даних. Процес активного навчання може бути побудований на основі різноманітних стратегій, таких як вибір прикладів, які викликають високий рівень невизначеності для моделі, або врахування прикладів, що знаходяться на межі між різними категоріями. Це дозволяє моделі фокусуватися на тих областях, де її точність може бути поліпшена.

Навчання на зразках, що вибрані активним чином, сприяє пришвидшенню процесу розмітки даних, оскільки модель самостійно визначає пріоритетні області для подальшої розмітки. Цей підхід, як результат, дозволяє зменшити зусилля та час, необхідний для створення високоякісних розмічених наборів даних. Деталізацію цього процесу можна знайти на схематичному зображенні, поданому на рисунку 5.11.



Рисунок 5.11 – Підхід з використанням методів активного донавчання моделей на результатах розмічених даних

## 5.5 Вимоги до програмного засобу для розмітки з урахуванням автоматизації

Для формулювання основних принципів та структури програмної системи, спрямованої на автоматизацію процесу розмітки відеоданих, належить детально розглянути і розмежувати функціональні та нефункціональні вимоги [194]. Функціональні вимоги включають в себе докладний опис поведінки програмного забезпечення, який містить основні характеристики та функції, що повинні бути реалізовані. Це механізми, які дають змогу користувачам взаємодіяти з програмним забезпеченням відповідно до визначеної мети. Нефункціональні вимоги, з іншого боку, визначають якості програмної системи, такі як її безпека, надійність, продуктивність, масштабованість, зручність у використанні тощо, що відповідають вимогам використання.

Досягнення мети автоматизації процесу розмітки відеоданих передбачає вибір та врахування підходів до автоматизації, які мають бути впроваджені цією програмною системою. У процесі розробки системи автоматизації потрібно враховувати особливості вхідної інформації, яка надходить у форматі mp4.

Викладемо функціональні та нефункціональні вимоги для програмного забезпечення, призначеного для автоматизації процесу розмітки відеоданих, з

обов'язковим врахуванням пріоритетності кожної конкретної вимоги. Під час уточнення пріоритетів вимог будемо керуватися шкалою, яку використовують у публікації [198]. За такою шкалою визначено чотири рівні пріоритетів для оцінювання важливості вимог:

- «Must have» (система повинна задовольняти вимозі);
- «Should have» (бажано, щоб система задовольняла вимозі);
- «Could have» (система може задовольняти вимозі);
- «Won't have this time» (система може не задовольняти вимозі).

Оскільки в рамках дисертаційної роботи будемо формулювати базові вимоги до програмної системи, що призначена для автоматизації процесу розмітки відеоданих, зокрема, використовуючи дуальну архітектуру розмітки даних, пропонується застосовувати модифікований підхід до визначення пріоритетів вимог. Пропонується поділяти усі функціональні та нефункціональні на незалежні та залежні вимоги. Залежна вимога відрізняється від незалежної вимоги тим, що її значущість проявляється лише у випадку виконання іншої вимоги, від якої вона залежить.

Тоді можна визначити шкалу для пріоритезації вимог до програмної системи таким чином:

- «Обов'язково»;
- «Обов'язково за умови»;
- «Бажано»;
- «Можливо».

Пріоритет «Обов'язково» відзначається як еквівалент пріоритету «Must have» і може бути визначений як для незалежної, так і для залежної вимоги.

Пріоритет «Обов'язково за умови» рекомендується використовувати для ситуацій, коли встановлюється обов'язкова залежна вимога. Наприклад, вимога «Забезпечення узгодження декодованих кадрів» є залежною від вимоги «Забезпечення декодування відео в кадри». У випадку, якщо базова вимога не виконується (не є обов'язковою), це призводить до втрати сенсу для залежної вимоги.

Пріоритет «Бажано» виступає як еквівалент пріоритету «Should have» і може бути визначений як для незалежної, так і для залежної вимоги.

Пріоритет «Можливо» є аналогом пріоритету «Could have» і може бути визначений як для незалежної, так і для залежної вимоги.

У контексті визначення базових вимог до програмної системи нецільовим є застосування пріоритету «Won't have this time». Тому у запропонованій шкалі відсутній еквівалент для цього пріоритету.

Базові функціональні вимоги до програмного забезпечення, призначеного для автоматизації процесу розмітки відеоданих сформульовано у таблиці 5.2. Нефункціональні вимоги до програмного забезпечення, призначеного для автоматизації процесу розмітки відеоданих сформульовано у таблиці 5.3. Модулі системи, що забезпечують роботу з даними, наведені на рисунку 5.12.

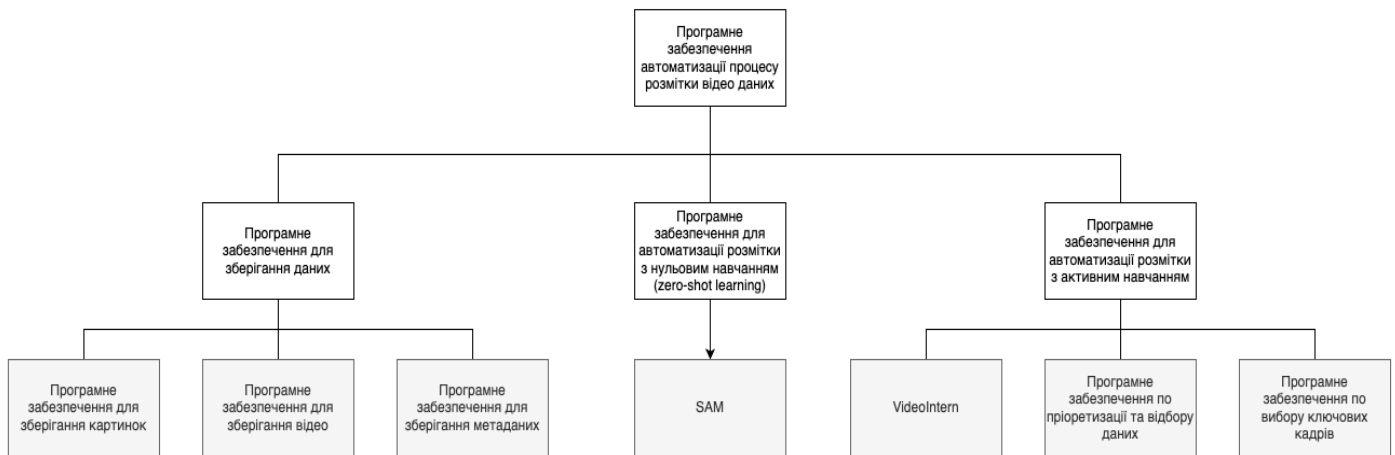


Рисунок 5.12 – Модулі системи, що відповідають за роботу з даними



Таблиця 5.2 – Базові функціональні вимоги до програмного забезпечення призначеного для автоматизації процесу розмітки відеоданих

Код	Вимога	Пріоритет
ФВ1	Запровадити автентифікацію та авторизація користувачів. Впровадити безпечні механізми автентифікації користувачів.	Обов'язково
ФВ2	Забезпечити призначення різних рівні доступу для різних ролей користувачів	Обов'язково (за умови виконання ФВ1)
ФВ3	Забезпечити підтримку імпорту відеоданих у різноманітних форматах (наприклад, MP4, AVI)	Обов'язково
ФВ4	Забезпечити інтуїтивно зрозумілий та зручний інтерфейс для ефективної розмітки об'єктів та регіонів інтересу на відео	Обов'язково
ФВ5	Забезпечити декодування відео в кадри	Обов'язково
ФВ6	Забезпечити синхронізацію кадрів між клієнтом та сервером	Обов'язково (за умови виконання ФВ5)
ФВ7	Забезпечити інтеграцію моделей з нульовим навчанням (zero-shot learning) в інструмент розмітки даних	Обов'язково
ФВ8	Забезпечити інтеграцію моделей активного навчання в інструмент розмітки даних	Обов'язково (за умови виконання ФВ7)
ФВ9	Забезпечити інтеграцію методу пріоритезації та відбору даних для розмітки в інструмент розмітки даних	Обов'язково (за умови виконання ФВ8)

Продовження таблиці 5.2.

Код	Вимога	Пріоритет
ФВ10	Забезпечити інтеграцію фундаційних моделей комп'ютерного зору для відео (InternVideo [ <b>Error! Reference source not found.</b> ])	Обов'язково (за умови виконання ФВ8)
ФВ11	Забезпечити взаємодію декількох розмітчиків даних над одним набором даних	Обов'язково
ФВ12	Забезпечити можливість генерації аналітичної інформації про хід розмітки, витрачений час та точність	Можливо
ФВ13	Забезпечити можливість ручного перемикання між моделями нульового та активного навчання	Можливо

Таблиця 5.3 – Нефункціональні вимоги до програмного забезпечення призначеного для автоматизації процесу розмітки відеоданих

Код	Вимога	Пріоритет
НФВ1	Забезпечити розробку інтерфейсу користувача з урахуванням підходу, орієнтованого на користувача, забезпечуючи інтуїтивність і легкість використання для користувачів із різними технічними знаннями	Можливо
НФВ2	Забезпечити ефективну обробку великих наборів відеоданих, забезпечуючи адаптивну розмітку навіть за значного навантаження даних і одночасних дій декількох користувачів	Бажано
НФВ3	Забезпечити високу надійність системи, забезпечити зведення до мінімуму будь-яких простоїв та забезпечувати постійну доступність для підтримки безперервних процесів розмітки	Можливо

Продовження таблиці 5.3.

Код	Вимога	Пріоритет
НФВ4	Розроблене програмне забезпечення має бути сумісним з різними операційними системами та веб-браузерами, забезпечуючи доступність на різних платформах і пристроях, якими зазвичай користуються розмітчики даних.	Можливо
НФВ5	Забезпечити користувачам гнучкість в налаштуванні моделей машинного навчання для підтримки автоматизацій розмітки дозволяючи адаптувати програмне забезпечення до конкретних вимог проєкту та складності завдань розмітки	Бажано
НФВ6	Забезпечити дотримання правил захисту даних і конфіденційності наданих користувацьких шляхом забезпечення дотримання таких стандартів, як GDPR та інших відповідних політик управління даними	Можливо
НФВ7	Програмне забезпечення має бути розроблено з урахуванням зручності обслуговування, сприяючи легкому оновленню, виправленню помилок і впровадженню нових функцій, що не буде викликати перебоїв у поточних проєктах розмітки .	Бажано
НФВ8	Забезпечити захист конфіденційних відеоданих і інформації користувача через застосування надійних заходів безпеки, зокрема шифрування даних,.	Бажано

Передбачається, що із системою можуть взаємодіяти дві категорії користувачів, яких умовно названо «Розмітчик даних» та «Адміністратор».

Діаграма прецедентів для «Розмітчика даних», наведена на рисунку 5.13. Вона ілюструє процес роботи розмітчика даних у контексті машинного навчання. Центральним актором виступає розмітчик даних, діяльність якого розгалужується на чотири напрямки: перегляд даних, класифікація об'єктів, розмітка даних та збереження

результатів розмітки даних. Процес перегляду даних включає в себе перегляд відео або зображень. Цей етап пов'язаний з визначенням областей інтересу, що дозволяє фокусувати увагу на релевантних елементах даних.

Класифікація об'єктів є наступним ключовим аспектом роботи розмітчика. Цей процес складається з двох підпроцесів: визначення категорій об'єктів та призначення об'єктів до відповідних категорій. Розмітка даних включає створення нової розмітки, перевірку точності автоматично розмічених даних, а також редагування існуючої та виправлення помилок автоматичної розмітки даних. Цей етап є критичним для забезпечення якості та надійності вхідних даних для методів машинного навчання. Завершальним етапом є збереження результатів розмітки даних, що забезпечує можливість подальшого використання та аналізу розмічених даних.

У сфері розмітки даних на відео адміністратор проєкту виконує роль головного організатора і наглядача за всім процесом розмітки. Діаграма прецедентів, наведена на рисунку 5.14, ілюструє обов'язки адміністратора системи розмітки даних, які включають три напрямки його діяльності: моніторинг продуктивності розмітки даних, управління доступами до системи, адміністрування автоматизації процесу розмітки даних.

Моніторинг продуктивності розмітки даних включає два варіанти використання. Перший варіант – це аналіз звітів про продуктивність, що передбачає перегляд щоденних або щотижневих звітів та аналіз кількості розмічених даних. Другий варіант – це виявлення проблем у продуктивності, що включає визначення збоїв або затримок у процесі та визначення причин низької продуктивності. Управління доступами до системи складається з двох компонентів. Налаштування прав доступу передбачає визначення та розподіл ролей у системі, а також оновлення списку користувачів для ролей.

Моніторинг доступів включає перевірку журналів доступу до інструментів та виявлення й усунення невідповідностей. Адміністрування автоматизації процесу розмітки даних є третім напрямком. Цей процес включає запуск методів автоматичної

розмітки, що охоплює як запуск моделей активного навчання, так і запуск моделей нульового навчання. Крім того, адміністратор відповідає за забезпечення навчального набору для моделей машинного навчання, що включає розширення набору даних для навчання та запуск навчання моделей машинного навчання.

У наступному підрозділі представлено розроблену архітектуру програмного забезпечення для автоматизації процесу розмітки відеоданих, що задовольняє вище визначеним вимогам (див. таблиці 5.1, 5.2), та забезпечує варіанти використання, визначені відповідно до ролей користувачів (див. додаток Є).

## **5.6 Запропонована дуальна архітектура програмного забезпечення для автоматизації розмітки відеоданих**

Здебільшого кожен інструмент для розмітки даних надає можливість автоматизації з використанням або попередньо навченої моделі, або за допомогою активного навчання на даних, які вже були розмічені. Однак жоден з них не пропонує поєднання цих підходів, що є вагомим недоліком, оскільки кожен інструмент має свої переваги та обмеження на різних етапах проєкту з розмітки даних.

По-перше, підходи, які базуються на попередньо навчених моделях, використовують моделі з нульовим навчанням (zero-shot learning), такі як SAM [195] (рис. 5.13 [196]).

Ці моделі навчені на обширних наборах даних та здатні впізнавати об'єкти в різноманітних сценаріях. Моделі з нульовим навчанням часто відзначаються великим розміром, що зазвичай коливається від декількох десятків до сотень мільйонів параметрів. Крім того, такі моделі демонструють посередні результати при роботі з рідкісними типами даних, наприклад, розмітку ядр клітин людських тканин. У той самий час ці моделі можуть показувати високі результати якості для поширених категорій об'єктів, таких як людина. Отже, зазначений підхід відмінно підходить на початкових

етапах розмітки даних, особливо тоді, коли для підходів активного навчання ще не накопичено достатньо даних для досягнення високих результатів після навчання.

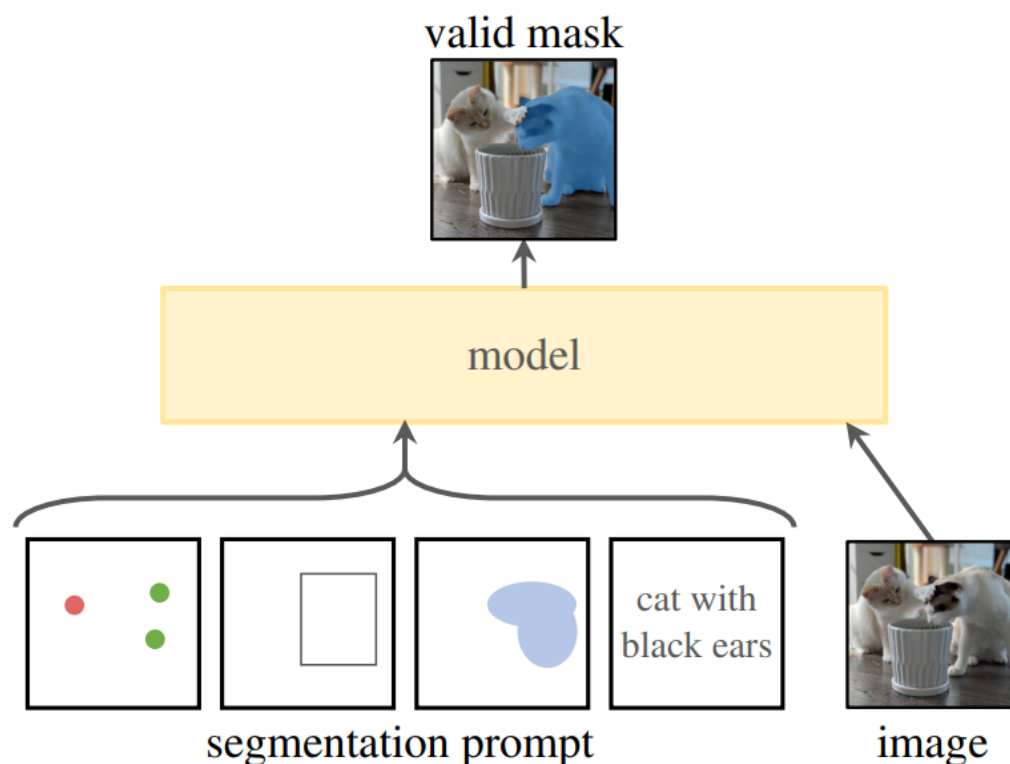


Рисунок 5.13 – SAM – підхід з нульовим навчанням (zero-shot learning) [196]

На противагу, підходи активного навчання можуть використовувати менші за кількістю параметрів моделі, що можуть містити від сотень тисяч до декількох мільйонів параметрів. Це робить їх застосування економічно доцільним у проєктах, що вимагають розмітки великих об’ємів даних, але при цьому вони мають значний недолік – таким підходам необхідний певний час на те, щоб накопичити достатню кількість даних для навчання таких моделей та їх вихід до рівня моделей з нульовим навчанням (zero-shot learning). В контексті даної роботи опираємося на модель VideoIntern [80]. Саме тому підхід з активним навчанням підходить для середньої та пізньої стадії процесу розмітки

даних, коли є вже певна кількість розмічених даних та можливо навчити модель активного навчання на цих даних.

Як можна побачити, вищезгадані підходи мають протилежні сильні та слабкі сторони та підходять для використання на різних стадіях розмітки даних. Тому в рамках даної дисертаційної роботи запропоновано підхід, який об'єднує обидва підходи в єдину систему шляхом моніторингу якості розмітки даних обома підходами та прийняттям рішення щодо того, з якої моделі буде використовуватися розмітка та донавчання методу активного навчання [197]. Формалізоване представлення запропонованого адаптивно-агрегованого методу у вигляді псевдокоду наведено як Лістинг 5.1:

---

**Лістинг 5.1.** Адаптивно-агрегований метод навчання нейромережі

---

**Inputs:** Dataset  $D$ , dataset labels  $L = \{L_1, L_2, \dots, L_n\}$ , model for active learning  $A$ , model for zero-shot learning  $Z$ , quality function mAP

---

```

1. a, b, control = 1, 1, 0 // Initialize controlling values
2. mode = Zero // Initialize operating mode
// Process dataset in batches
3. while batch in D:
    3.1. results = A(batch)
    // All control times check if active learning is not better than zero-shot approach
    3.2. if control == 0:
        3.2.1. results_zero = Z(batch)
        3.2.2. if mode == Active
            3.2.2.1 Fix annotation in results
            3.2.2.2 Train algorithm of Active learning on data from 2.2.2.1
        3.2.3. else:
            2.2.3.1 Fix annotation in results_zero
            2.2.3.1 Train algorithm of Active learning on data from 2.2.3.1
        3.2.4. anno = D.get_annotation(batch)
        3.2.5. if mAP(results, anno) > mAP(results_zero, anno):

```

---

---

2.2.5.1. control = b

2.2.5.2. a, b = b, a + b

2.2.5.3. mode = Active

3.2.6. else:

2.2.6.1. a, b, control = 1, 1, 0

2.2.6.2. mode = Zero

3.3. else:

2.3.1. control = control – 1

2.3.2. yield results

---

**Outputs:** annotated dataset  $D$ , finetuned on dataset  $D$  active learning model  $A$

---

Оскільки моделі мають різну вартість використання, то це вплинуло на алгоритм, який мінімізує кількість використання моделей з нульовим навчанням (zero-shot learning) шляхом пропуску обробки вхідних даних з інтервалами між послідовними застосуваннями, довжина яких контролюється числами Фібоначчі:

$$F_1 = 1, F_2 = 1, F_{n+2} = F_n + F_{n+1}, n = 1, 2, 3, \dots \quad (5.1)$$

Послідовність Фібоначчі обрана для експонентного зменшення частоти викликів моделі з нульовим навчанням, що доведено формулою Біне:

$$F_n = \frac{\phi^n - (-\phi)^{-n}}{\phi - (-\phi)^{-1}} \approx \frac{\phi^n}{\sqrt{5}}, \quad (5.2),$$

де  $\phi$  – приймає значення золотого перетину  $\phi = \frac{1 + \sqrt{5}}{2}$ .

Метод починається з ініціалізації керуючих значень a, b, та control, що виконується за константний час  $O(1)$ . Метод обробляє датасет частинами з кількістю зразків в кожній  $K \leq n$ . Кількість частин залежить від обсягу пам'яті, використовуюваного для обчислень. На кожній ітерації для кожної частини з датасету  $D$  виконується обчислення моделі активного навчання  $A$ , що має складність  $O(a)$ .

Раз на певну кількість кроків, що контролюється послідовністю Фібоначчі (кроки 2.2.5.2 та 2.2.6.1), виконується порівняння якості методів активного  $A$  та нульового



навчання  $Z$ . Для цього додатково виконується обрахунок методу нульового навчання  $Z$ . Припустимо, що він має складність  $O(z)$ . Далі відбувається порівняння якості методів, що відбувається за допомогою метрики mAP, складність обчислення якої складає  $O(C^2 \log_2 C)$ , де  $C$  – кількість об'єктів на відео.

Загальна складність методу залежить від якості моделей активного та нульового навчання, що буде змінювати кількість викликів до моделей, але в найгіршому випадку, якщо метод нульового навчання буде завжди точнішим за метод активного навчання, то складність можна оцінити як  $O(K \times (a + z + C^2 \log_2 C))$ .

Оскільки всі операції на кожному кроці виконуються за кінцевий час і жоден цикл не є нескінченним, можна зробити висновок, що метод завжди завершується за скінченну кількість кроків. Це забезпечує його обчислювальну здійсненність, тобто відсутність безкінечних циклів або інших перешкод для його успішного завершення.

Схематичне представлення процесу роботи інструменту розмітки з використанням дуальної архітектури наведено на рисунку 5.14.

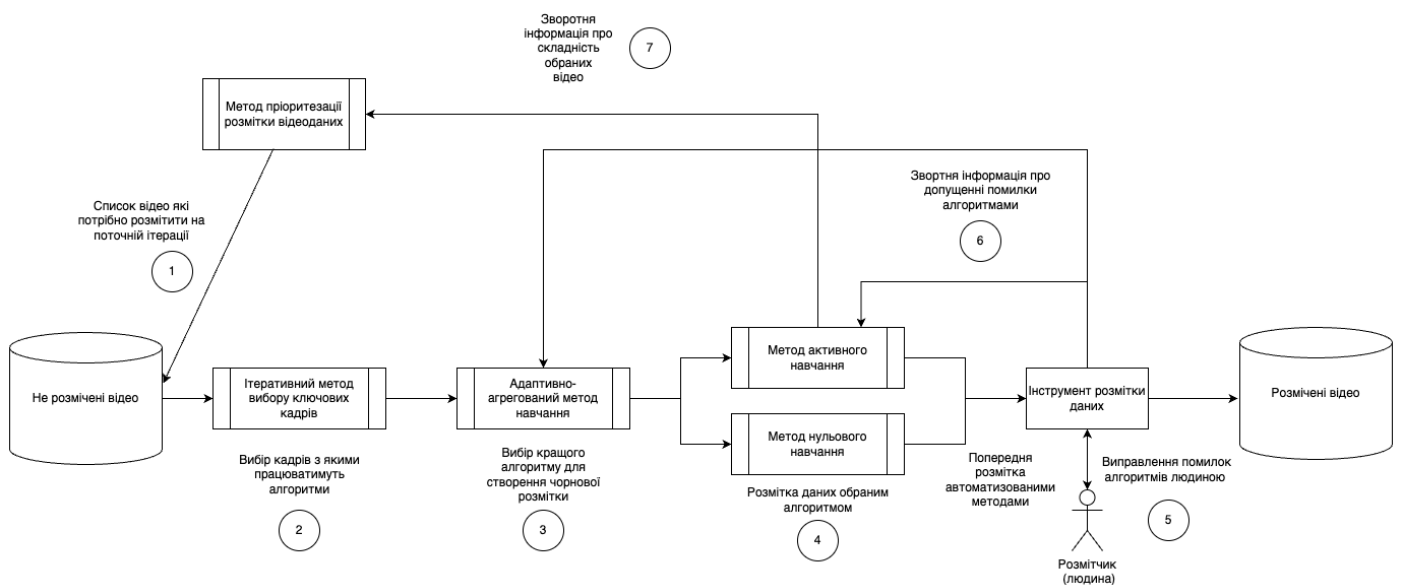


Рисунок 5.14 – Схематичне представлення процесу роботи інструменту розмітки з використанням дуальної архітектури

На рисунку 5.14 наведені наступні процеси:

- 1) визначення списку відео з найбільшою складністю для методу детекції зображень за допомогою методу пріоритезації;
- 2) вибір ключових кадрів на відео, на яких будуть застосовуватися алгоритми;
- 3) вибір методу, який має найвищу точність розмітки на даному етапі;
- 4) виконання розмітки відібраних даних за допомогою методів активного навчання та нульового навчання (за необхідності);
- 5) розмітчик даних аналізує розмітку, отриману від методу автоматизації розмітки; зберігає ту розмітку, яка вірна; видаляє хибну розмітку; виконує розмітку пропущених об'єктів;
- 6) інформація про правки, внесені розмітчиком даних, відправляється на метод активного навчання для подальшої мінімізації помилок;
- 7) інформація про реальну складність обраних зображень передається від методу активного навчання до методу пріоритезації процесу розмітки даних.

Варто зауважити, що кроки 3, 5 та 6 в даному процесі є критичними, оскільки система постійно поліпшується за рахунок зворотного зв'язку на кроці 5, а отже, ті зображення, які були раніше складними, могли стати простими для методу детекції об'єктів і навпаки. Саме тому потрібно адаптувати механізм пріоритезації процесу розмітки даних після кожної частини розмічених даних, щоб він залишався актуальним.

## **5.7 Опис архітектури розробленого програмного забезпечення**

Розроблене програмне забезпечення для автоматизації процесу розмітки – це веб-застосунок, побудований за клієнт-серверною та мікросервісною архітектурами, частина коду якого представлена в додатку А.

Діаграма розгортання розробленого програмного забезпечення наведена на рисунку 5.15.

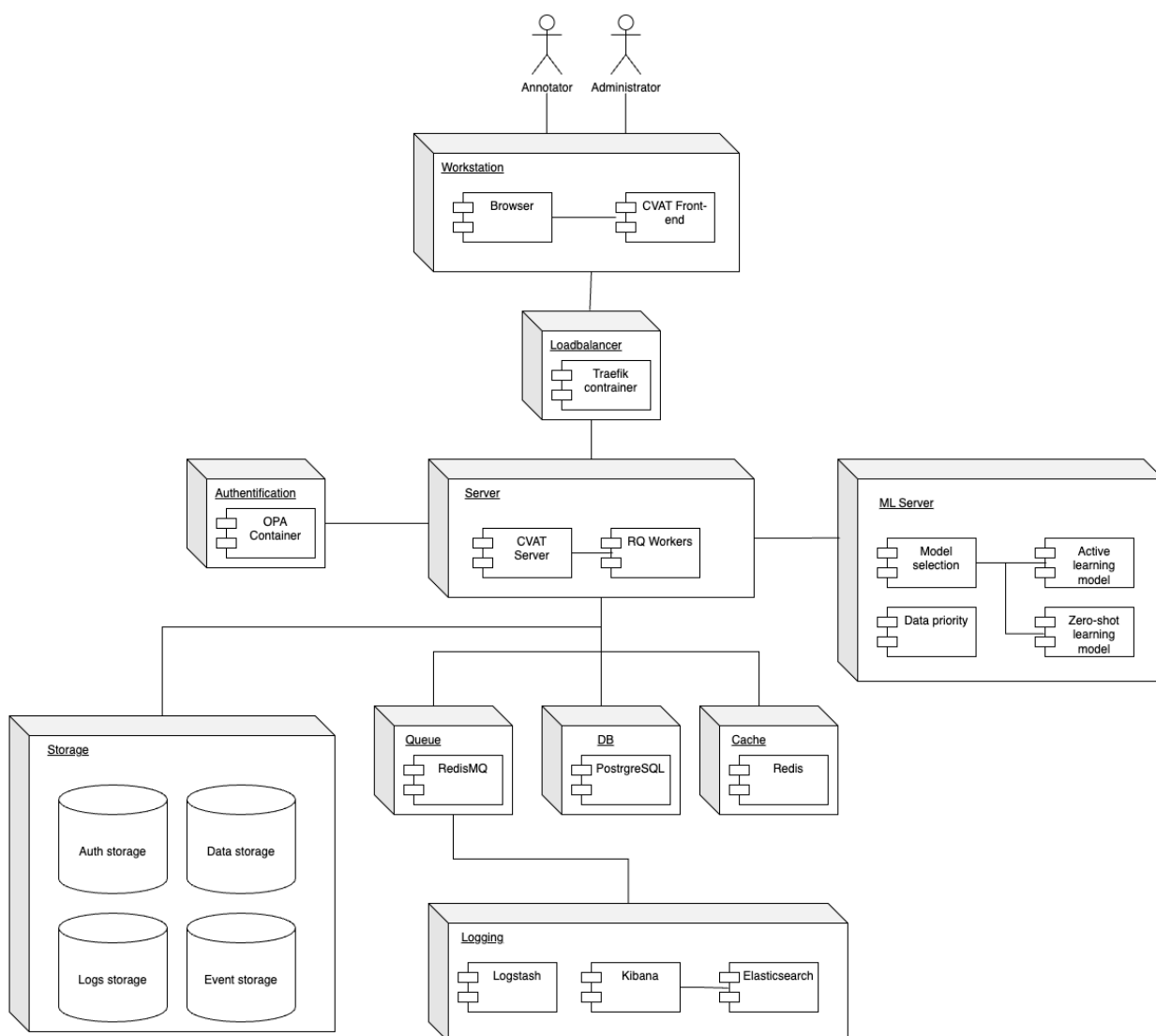


Рисунок 5.15 – Діаграма розгортання розробленого програмного забезпечення

Візуалізація архітектури програмного забезпечення наведена в додатку Б. Вона складається з наступних частин, які оркеструються на базі платформи Docker та Docker Compose [198]:

- клієнт розроблений на базі React [199];
- серверна частина клієнту розроблена на базі Python Django [200];
- балансування трафіку виконано на базі Traefik [201];

- баз даних на базі PostgreSQL [202] для зберігання даних, розмітки, налаштувань системи, логів та додаткових метаданих;
- кешування на базі Redis [203];
- черги подій на базі RedisMQ [203];
- системи аутентифікації на базі контейнерів OPA [204];
- системи управління моделями машинного навчання на базі Nuclio [205];
- системи управління пріоритетом розмітки;
- системи управління моделями активного навчання;
- системи управління моделями нульового навчання (zero-shot learning);
- системи логування та аналітики на базі Elasticsearch [206].

## 5.8 Висновки до розділу

У даному розділі були розглянуті ключові аспекти архітектури програмного забезпечення для автоматизації розмітки відеоданих. Перш за все, було проведено аналіз наявних програмних засобів для розмітки зображень та відео, таких як V7, LabelBox, Keylabs, LabelImg, LabelMe, Label Studio та CVAT, з метою визначити програмний засіб для поставлених завдань у рамках цієї дисертаційної роботи. Таким засобом визначено CVAT.

Детально проаналізовано архітектуру програмного засобу автоматизованої розмітки відеоданих CVAT, що дозволило вивчити його побудову з точки зору інженерії програмного забезпечення. Особливу увагу приділено аспектам архітектури, які забезпечують високу ефективність та гнучкість під час роботи з розміченими даними.

Далі в розділі розглянуто функціональні та нефункціональні вимоги до системи для автоматизації процесу розмітки відеоданих. Визначено основні функції, які повинно виконувати розроблене програмне забезпечення, зокрема, швидкодія та масштабованість.

Запропоновано дуальну архітектуру програмного забезпечення для автоматизації розмітки відеоданих, описано ключові компоненти системи та їх взаємодію [197]. Наведено опис архітектури та основних її компонентів.

У результаті дослідження в цьому розділі визначені основні вимоги до програмного забезпечення для автоматизації процесу розмітки відеоданих та розроблена його дуальна архітектура, що стала основою для створення інструменту для розмітки відеоданих. Розроблено програмне забезпечення на базі запропонованої архітектури та розроблених методів: метод вибору складних зразків для пріоритезації процесу розмітки даних, метод ітеративного вибору ключових кадрів для вибору кадрів з якими будуть працювати моделі машинного навчання, метод Attr4Vis для преднавчання методу активного навчання.

## 6 АНАЛІЗ ЕФЕКТИВНОСТІ ДУАЛЬНОЇ АРХІТЕКТУРИ

### 6.1 Процес розмітки даних

Відповідно до дослідження, представленого в Bloomberg [108], весь процес розмітки даних для проєктів, що використовують машинне навчання, можна розглядати як комплексний процес, що складається з дев'яти ключових етапів:

- постановка мети проєкту з розмітки даних та визначення основних зацікавлених сторін, що прописується у проєктній документації;
- декомпозиція мети проєкту на цілі та формування етапів розмітки даних;
- планування дорожньої карти розмітки даних;
- вибір розмітчиків даних;
- вибір утиліти для розмітки даних;
- створення керівних принципів;
- навчання розмітчиків даних;
- управління процесом розмітки даних;
- використання розмічених даних.

Процес формалізації мети розмітки даних розпочинається з визначення основних зацікавлених сторін, які, як правило, включають:

- проєктний менеджер (і/або продуктовий менеджер): особа, що визначає прикладне застосування проєкту, розуміє його потенційний вплив на бізнес, встановлює пріоритети функціям на основі потреб користувачів та може мати інші доменні знання;
- головний інженер: відповідає за впровадження рішення, виступає замовником розмітки даних та надає інформацію щодо технічної стратегії та доцільності прийнятих рішень/компромісів в процесі розмітки даних;
- менеджер по розмітці даних: відповідальний за нагляд за проєктом розмітки, обирає та керує робочою силою, забезпечує якість та узгодженість даних.

Після визначення основних зацікавлених сторін необхідно встановити кінцеву мету, що допоможе учасникам проєкту забезпечити спрямованість всіх зусиль на досягнення цієї мети. Останнім кроком даного етапу є вибір способу регулярного та чіткого спілкування між зацікавленими сторонами на всіх етапах проєкту. Це дозволить зберегти фокус на кінцевій меті проєкту. Цей крок важливий незалежно від того, чи бере участь у проєкті одна команда чи декілька команд, розташованих в різних куточках планети.

Далі учасники повинні визначити характеристики даних, які підлягають розмітці, метод відбору даних та формат розмітки. Аналіз даних визначає важливий початковий етап процесу формалізації проєкту. Розуміння особливостей даних, їх обмежень, закономірностей та крайових випадків дозволяє учасникам проєкту приймати обґрунтовані рішення стосовно необхідного типу розмітки. Після усвідомлення характеристик даних, сторонам необхідно вирішити, як вибрати підвибірку даних для розмітки та чи необхідна попередня обробка даних перед розміткою. Також зацікавлені сторони повинні визначити набір міток, які підлягають розмітці. Цей процес сприяє конкретизації задачі розмітки та уточненню обсягу роботи, сприяючи ефективній реалізації проєкту з розмітки даних.

Далі настає етап планування дорожньої карти розмітки даних, що виступає інструментом для чіткого визначення очікувань зацікавлених сторін, виявлення обмежень та залежностей, які відзначають великий вплив на фінансові витрати та загальний успіх проєкту. Дорожня карта передбачає представлення чітких термінів із докладною інформацією про проєкт та ключові етапи. При визначенні термінів потрібно враховувати достатній час для розробки керівних принципів та навчання робочої сили.

Після цього запускається процес вибору розмітчиків даних. Цей етап вимагає глибокого розуміння різноманітних аспектів, таких як вимоги до конфіденційності та безпеки даних, складність самого проєкту, а також обмеження, накладені бюджетом. Вибором розмітчиків повинен займатися спеціалізований менеджер по розмітці даних.

Необхідно зауважити, що це рішення ніколи не повинно прийматися виключно проєктним менеджером чи головним інженером, оскільки воно визначає не тільки технічні аспекти, але й конфіденційність, безпеку та ефективність роботи всього колективу розмітчиків даних.

Далі відбувається процес відбору програмного забезпечення для виконання розмітки даних. Зацікавленим сторонам слід звернути увагу на такі аспекти:

- спрямованість на стандартну інфраструктуру;
- гарантія конфіденційності даних та легкий доступ розмітчиків до необхідної інформації;
- оцінка технічного списку задач, пов'язаного з імплементацією обраної утиліти;
- можливості підтримки функцій управління проєктами;
- можливість налаштування користувацького інтерфейсу та вимоги до досвіду користувача для інструменту розмітки та інтерфейсу проєкту.

Після цього здійснюється розробка керівних принципів, які націлені на охоплення типових питань та визначення правил для розмітки даних у ході щоденних обов'язків. Одна особа має взяти на себе відповідальність за створення та оновлення цих принципів. Це сприяє внутрішній узгодженості, усуває суперечливий зміст та забезпечує зрозумілість. Менеджер проєкту розмітки є найбільш компетентною особою для розробки інструкцій з розмітки, оскільки він володіє розумінням набору даних, кінцевої мети та технічних вимог.

Незважаючи на те, що відповідальність за розробку та оновлення керівних принципів покладається на одну особу, кінцеві рекомендації мають виникати в результаті співпраці всіх зацікавлених сторін. Менеджер проєкту розмітки повинен координувати роботу всіх учасників проєкту розмітки даних, щоб гарантувати чіткість та узгодженість вказівок, що мають враховати особливості набору даних.

Після розробки керівних принципів проводиться пілотне тестування для перевірки їх ефективності, залучаючи тестувальників, які розмічають великий обсяг даних. Після



колективного огляду результатів вирішується, чи необхідно вносити зміни до рекомендацій.

Остаточні версії керівних принципів після всіх виправлень розміщуються в інструменті для розмітки даних, де розмітчики можуть зручно звертатися до них під час виконання своїх завдань. Виконавці проєкту мають уважно розглядати будь-які модифікації керівних принципів, оскільки вони можуть вплинути на вартість та терміни виконання проєкту.

Далі відбувається інструктаж розмітчиків даних, спрямований на забезпечення адекватного їх розуміння проєкту та високоякісної розмітки. Графік роботи над проєктом повинен враховувати визначений період підготовки, тривалість якого обумовлена обраною групою розмітчиків даних, складністю завдання та вибраним методом забезпечення якості даних. Активне сприйняття запитань від розмітчиків даних під час тренування є елементом для вирішення можливих непорозумінь або невизначеності в інструкціях.

Процес роботи розмітника даних, який розмічає декілька різних категорій об'єктів обмежувальними рамками на відео, є складною і багатоступеневою задачею, що вимагає точності та систематичності. Цей процес можна формалізувати, використовуючи математичний апарат теорії множин для опису і маніпуляції об'єктами розмітки.

На початковому етапі розмітник даних визначає множину кадрів  $F = \{f_1, f_2, f_3, \dots, f_n\}$ , де  $n$  – загальна кількість кадрів у відео. Кожен кадр  $f_i$  аналізується на наявність об'єктів, що належать до різних категорій. Нехай  $C = \{c_1, c_2, c_3, \dots, c_k\}$  є множиною категорій об'єктів, де  $k$  – кількість різних категорій, які підлягають розмітці.

Для кожного кадру  $f_i$  та кожної категорії  $c_j$  розмітник визначає множину об'єктів  $O_{ij} = \{o_{ij1}, o_{ij2}, o_{ij3}, \dots, o_{ijm}\}$ , де  $m$  – кількість об'єктів категорії  $c_j$  на кадрі  $f_i$ . Кожен об'єкт  $o_{ijs}$  представляється обмежувальною рамкою, яка математично визначається як множина координат  $b_{ijs}$ , що описують її межі. Наприклад, обмежувальна рамка може

бути представлена як пара множин координат кутів:  $b_{ijs} = ((x_{min}, y_{min}), (x_{max}, y_{max}))$ , де  $(x_{min}, y_{min})$ , і  $(x_{max}, y_{max})$  – координати верхнього лівого і нижнього правого кутів відповідно.

Теорія множин використовується для управління та маніпуляції цими обмежувальними рамками. Операція об'єднання  $\cup$  дозволяє комбінувати рамки, що перекриваються або розташовані близько одна до одної, утворюючи єдину множину точок, що представляє загальну область покриття. Операція перетину  $\cap$  використовується для визначення спільних областей між рамками, що може свідчити про присутність об'єктів у спільному просторі. Операція різниці  $\setminus$  дозволяє відокремити одну обмежувальну рамку від іншої, видаляючи непотрібні області, такі як фон, який не є частиною об'єкта.

Процес розмітки об'єктів на відео вимагає динамічного оновлення обмежувальних рамок у кожному кадрі з урахуванням руху об'єктів. Це забезпечується за допомогою методів відстеження, який аналізує зміну положення об'єктів і відповідно коригує координати рамок у реальному часі. Для кожного об'єкта  $o_{ijk}$  визначається траєкторія руху  $T_{jk}$ , яка представляється як множина координат рамок у послідовності кадрів  $T_{js} = \{b_{1js}, b_{2js}, b_{3js}, \dots, b_{njs}\}$ .

Аналіз результатів розмітки здійснюється за допомогою теорії множин, що дозволяє оцінити точність розмітки, виявити можливі помилки та запропонувати методи вдосконалення. Це включає порівняння розмічених множин об'єктів з реальними даними та аналіз відхилень.

Таким чином, задачу розмітки об'єктів обмежувальними рамками на відео, що включає декілька різних категорій об'єктів, можна формалізовано описати за допомогою теорії множин, що забезпечує коректність постановки цієї складної задачі.

Після завершення процесу розмітки, зацікавлені сторони зобов'язані провести аналіз виконаної роботи з метою визначення необхідності додаткової розмітки даних.

Враховуючи, що процес розмітки додаткових даних буде вимагати додаткових ресурсів у вигляді часу, фінансів та зусиль, детальне обговорення та обґрунтування цього рішення усіма зацікавленими особами є обов'язковим етапом. Якщо якість та обсяг розмічених даних відповідають вимогам та очікуванням усіх сторін, то проєкт можна вважати успішно завершеним, інженери можуть приступати до етапу побудови моделей машинного навчання на основі накопичених даних.

## 6.2 Організація розмітки даних

Для розмітки даних на наборі даних YouTube Bounding Boxes [207] необхідно створити формалізований опис задачі розмітки даних з метою перевірки пришивдшення процесу розмітки різними підходами автоматизації.

Для проведення експериментальних досліджень на наборі даних YouTube Bounding Boxes необхідно знайти 10 досвідчених розмітчиків, щоб забезпечити достатню різноманітність в даних експериментів для подальшого аналізу. Був обраний підхід з залученням колег, які працюють з аналізом зображень та відеоданих, оскільки вони мають достатньо фундаментальних знань в області машинного навчання, штучного інтелекту, та комп'ютерного зору, а отже будуть вимагати мінімального навчання для реалізації мети експериментів.

Для постановки даних експериментів створено правила розмітки даних, базуючись на правилах розмітки даних MS COCO [46]. Метою розмітки є визначення трьох категорій об'єктів у відео з набору даних YouTube Bounding Boxes [208], а саме: люди, транспортні засоби та звірі. Розмітка здійснюється шляхом використання прямокутних обмежуючих рамок.

Категорія "людина" включає всіх людей, незалежно від статі, віку чи видимості частини тіла. Рамка повинна охоплювати все тіло людини, навіть якщо частина тіла не видима. Варіації позицій, такі як стояння, сидіння чи лежання, повинні враховуватися.

Категорія "транспорт" охоплює всі типи транспортних засобів, включаючи автомобілі, вантажівки, автобуси, мотоцикли, велосипеди тощо. Рамка повинна повністю охоплювати весь транспортний засіб, включаючи будь-який вантаж або додаткове обладнання, яке є частиною транспортного засобу. Якщо транспортний засіб частково закритий іншим об'єктом, рамка повинна охоплювати тільки видиму частину.

Категорія "звірі" включає всіх тварин, незалежно від виду. Рамка повинна повністю охоплювати тіло тварини. Якщо тварина частково закрита іншим об'єктом, рамка повинна охоплювати тільки видиму частину.

Рамки повинні бути максимально точними, охоплюючи об'єкт з мінімальними зазорами. Не слід розмічати об'єкти, які важко ідентифікувати через низьку якість відео або сильне заступання іншими об'єктами. Якщо об'єкт частково закритий іншим об'єктом, розмічайте тільки видиму частину об'єкта, і уникайте дублювання розміток одного і того ж об'єкта.

Процес розмітки починається з відкриття відео в обраному інструменті для розмітки. Кадри для розмітки обираються з інтервалом в одну секунду, якщо відео не зазначене іншим чином. Далі потрібно нанести прямокутні обмежуючі рамки на об'єкти відповідно до категорій і зберегти результати розмітки у форматі, що підтримується інструментом (наприклад, JSON або XML).

Якщо виникають сумніви щодо категоризації або правил розмітки, потрібно консультуватись з менеджером проєкту розмітки для уточнення, що забезпечить максимально можливу точність та акуратність при розмітці об'єктів. Будь-які зміни або доповнення до правил розмітки повинні бути затверджені менеджером проєкту розмітки та повідомлені всім розмітчикам у письмовій формі.

Після завершення розмітки, інженер-дослідник проводить перевірку розмічених даних на відповідність правилам. Надання зворотного зв'язку розмітчикам допомагає виправити помилки або покращити якість розмітки.

Вищенаведені правила допоможуть забезпечити єдину стандартизацію процесу розмітки даних, що дозволить провести об'єктивний аналіз ефективності різних підходів до автоматизації.

В рамках поставлених експериментів було проаналізовано 10 відео з набору даних YouTube Bounding Boxes в розрізі різних умов щодо автоматизації процесу розмітки даних:

- без автоматизації (базовий варіант);
- з автоматизацією на основі підходу нульового навчання за допомогою методу SAM [195];
- з автоматизацією на основі методів активного навчання;
- з автоматизацією на основі дуальної архітектури.

Результати розмітки використані для аналізу процесу розмітки даних без та з використанням автоматизації процесу розмітки різними підходами, оцінювання швидкості розмітки кожного екземпляру даних та вимірювання прискорення процесу розмітки відео даних різними підходами.

### **6.3 Розмітка даних для задачі детекції об'єктів за допомогою CVAT**

CVAT [193] являє собою односторінковий веб-застосунок, в рамках якого інтегровано весь необхідний функціонал, що сприяє простоті та зручності користування сервісом для розмітки даних як досвідчених розмітчиків даних, так і початківців. На рисунку 6.1 наведено приклад інтерфейсу користувача з відео, який буде використовуватись для розмітки відео.

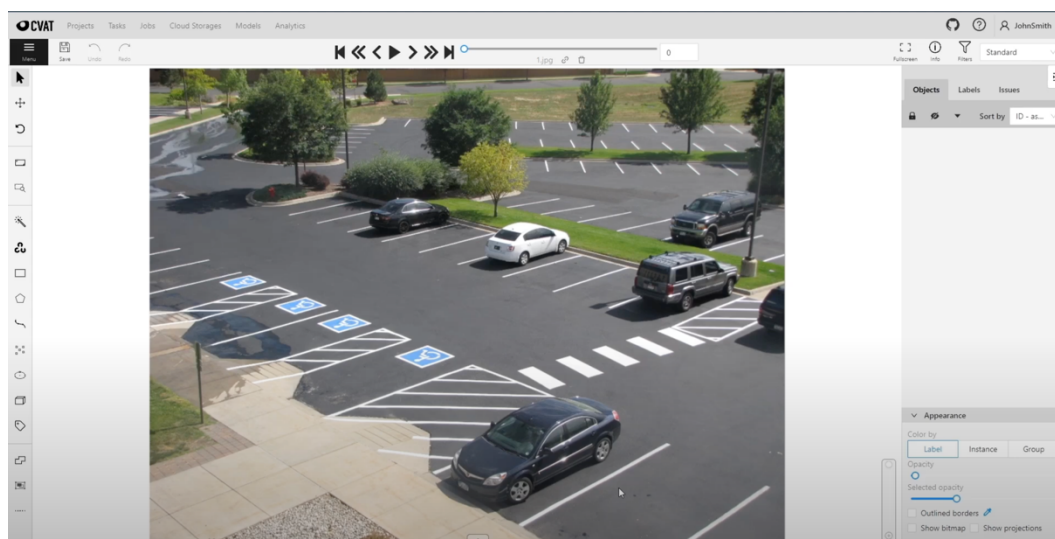


Рисунок 6.1 – Початок роботи з відео в інструменті розмітки даних CVAT

Основною частиною функціоналу веб застосунку є можливість завантаження зображень та відео для подальшої розмітки. Крім того, користувач отримує можливість проведення розмітки обраного об'єкта чи об'єктів на завантажених даних. Приклад підготовки до розмітки відеоданих наведено на рисунку 6.2.

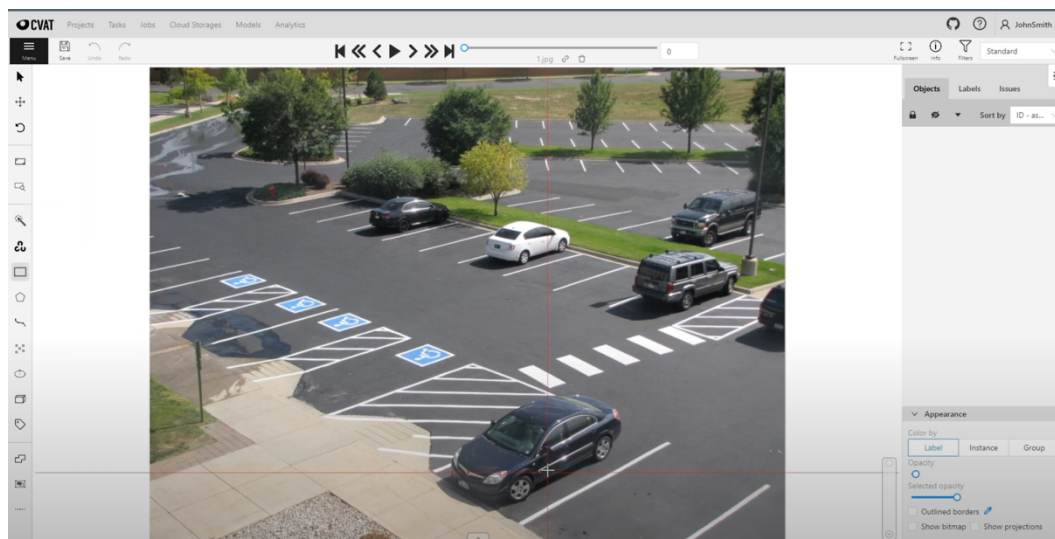


Рисунок 6.2 – Підготовка до розмітки обраного об'єкту в інструменті розмітки CVAT

Для забезпечення максимальної точності розмітки передбачено функцію наближення зображення, що наведено на рисунку 6.3.

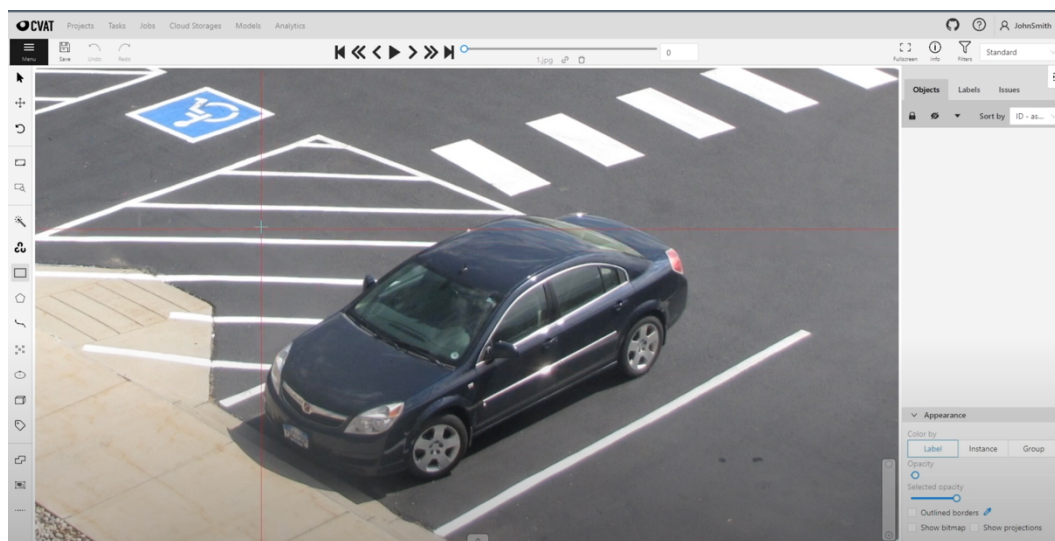


Рисунок 6.3 – Наближення зображення для збільшення точності розмітки обраного об'єкта на відео в інструменті розмітки CVAT

Зафіксовані результати розмітки можливо зручно зберегти за допомогою відповідної опції у веб-застосунку, що наведено на рисунку 6.4.

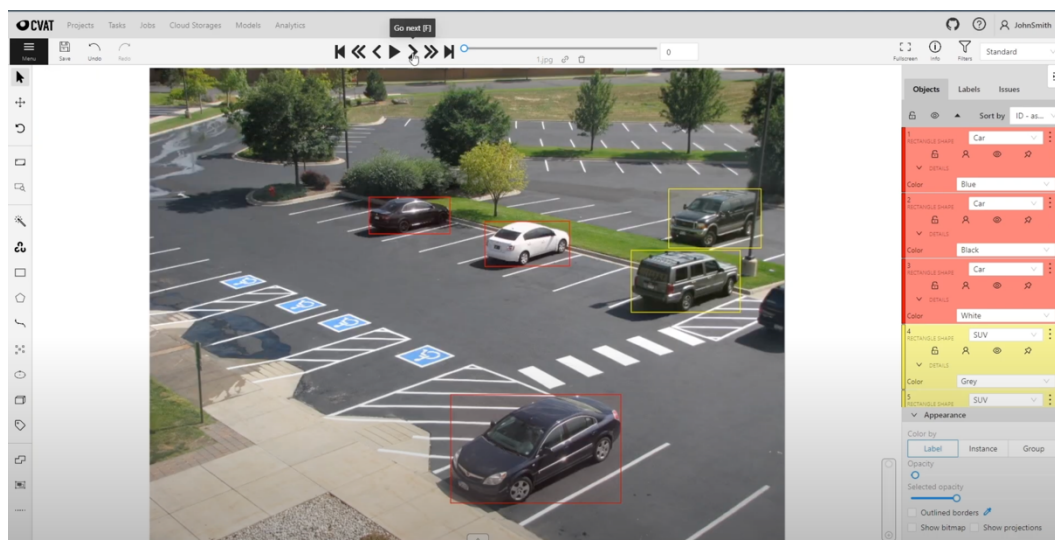


Рисунок 6.4 – Результат розмітки відеоданих в інструменті розмітки CVAT

## 6.4 Результати експериментального дослідження

Для постановки експериментів був використаний набір даних YouTube Bounding Boxes [207]. Набір даних YouTube Bounding Boxes — це колекція розмічених відеокадрів, призначених для детекції об'єктів і дослідження якості локалізації об'єктів на відео методами машинного навчання, який розроблений як розширення набору даних YouTube-8M [208]. Цей набір даних зосереджений на тому, щоб надати якісну розмітку обмежувальних рамок навколо об'єктів у відеокадрах.

Набір даних охоплює 4803 класи, надаючи можливість класифікувати та розпізнавати різні об'єкти, сцени та дії на відео. Набір даних складається з 240000 відео на яких розмічені 5600000 об'єктів. Набір даних YouTube Bounding Boxes є цінним ресурсом для дослідників і практиків, які працюють над завданнями комп'ютерного зору, зокрема тими, що стосуються виявлення об'єктів і локалізації їх на відео.

Для проведення експериментів був використаний метод з нульовим навчанням SAM [195] та метод детекції об'єктів на базі InternVideo [80], який побудований на базі mmaction2 [127] та має наступні характеристики:

- базова модель для методу детекції – Attr4Vis [180];
- попередньо навчена на 13 мільйонах відео;
- метод детекції об'єктів – Cascade R-CNN [40];
- базовий розмір вхідного зображення – 1600x900;
- кількість стадій детекції об'єктів – 3 [40];
- генератор ознак – FPN [209];
- генератор регіонів інтересу – GA-RPN [210];
- оптимізатор – LookAhead [211].

В запропонованій дуальній архітекторі використано запропонований метод пріоритезації даних [123], метод вибору ключових кадрів [149], та нейронну мережу, навчену методом Attr4Vis [180] як базову модель для методу детекції.



З метою проведення детальних експериментів по розмітці даних та оцінці ефективності різних підходів до автоматизації, була створена група з 10 навчених розмітчиків даних. Ці учасники займались розміткою 10 відібраних відео, які були піддані різним умовам автоматизації. Для забезпечення об'єктивності та точності оцінок для кожного учасника експерименту був обчислений час, витрачений на розмітку. Результатом кожного конкретного експерименту є час розмітки, який обчислювався на основі усіх учасників та усіх відео.

Медіанний час розмітки відео без автоматизації кожним із розмітчиків даних наведено на рисунку 6.5.

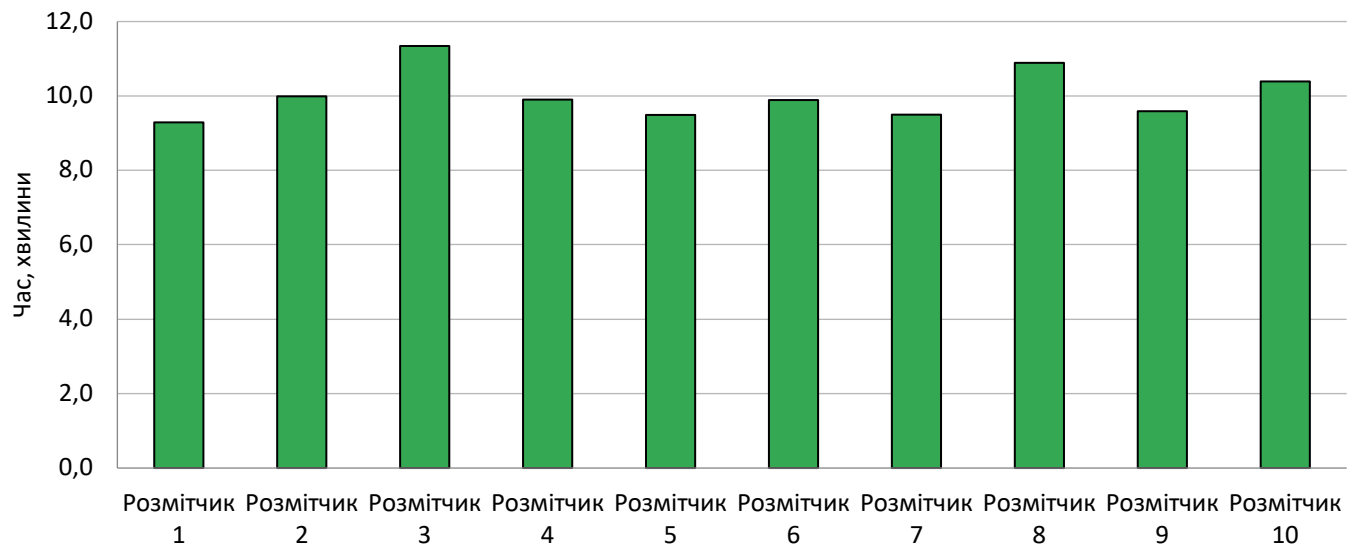


Рисунок 6.5 – Медіанний час розмітки відео без автоматизації

Детальна інфографіка по часу розмітки кожного відео кожним розмітчиком даних наведено на рисунку 6.6.

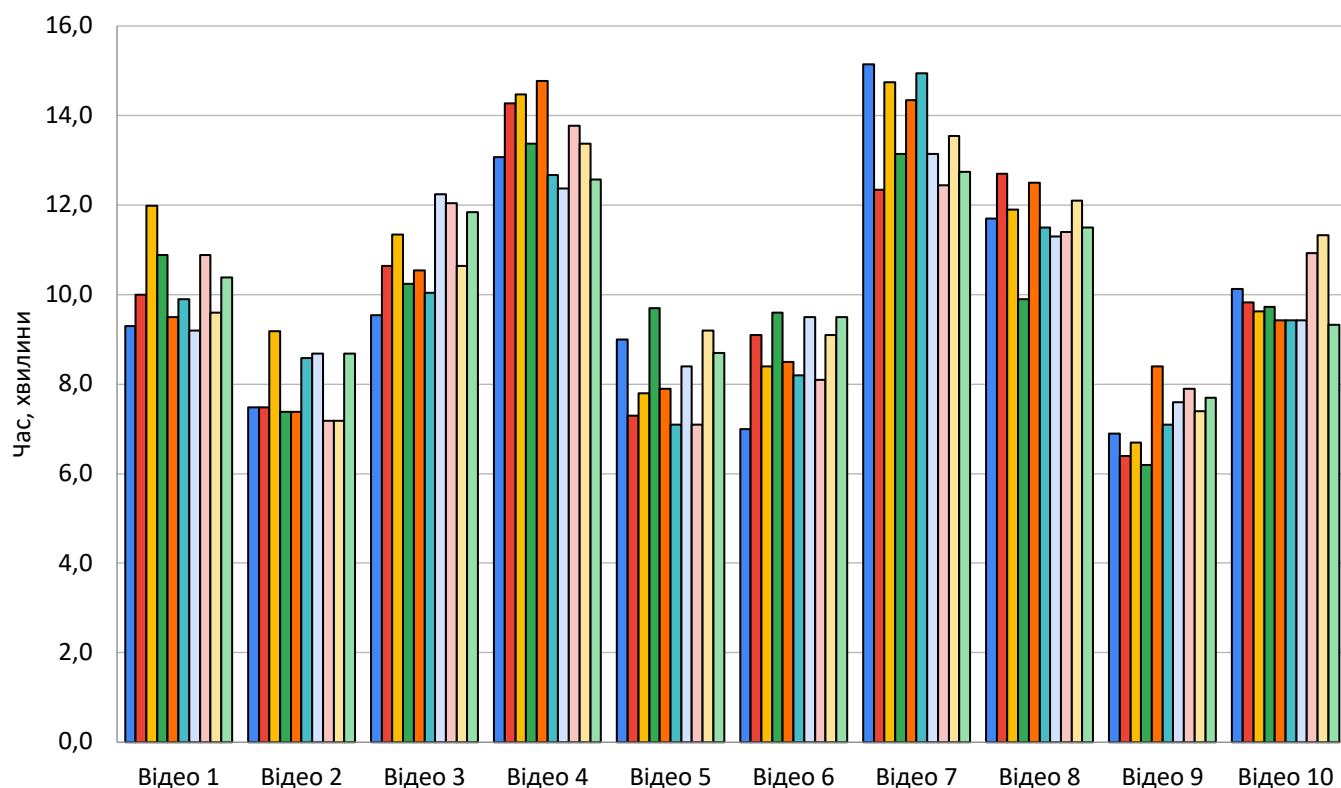


Рисунок 6.6 – Інфографіка по часу, витраченому на розмітку кожного відео кожним розмітчиком даних без автоматизації

Вихідні дані для рисунків 6.5 та 6.6 наведено в додатку В. З цих результатів можемо зробити висновок, що медіанний час розмітки заданих відео без автоматизації складав 9,7 хвилин. Середнє значення та стандартне відхилення розподілу відповідно склали 10,2 та 2,3 хвилин відповідно, 95-відсотковий довірчий інтервал – (9,7; 10,6).

Автоматизація за рахунок використання підходу з нульовим навчанням виконана з використанням моделі SAM [195]. Кожному розмітчику даних надавалися попередньо розмічені відео з об'єктами, знайденими SAM [195] на кожному кадрі, а розмітчику потрібно було виправити створену підходом розмітку та додати пропущені об'єкти. Медіанний час розмітки відео з автоматизацією підходом нульового навчання для всіх

розмітчиків даних наведено на рисунку 6.7 та детальний час розмітки кожного відео наведено на рисунку 6.8.

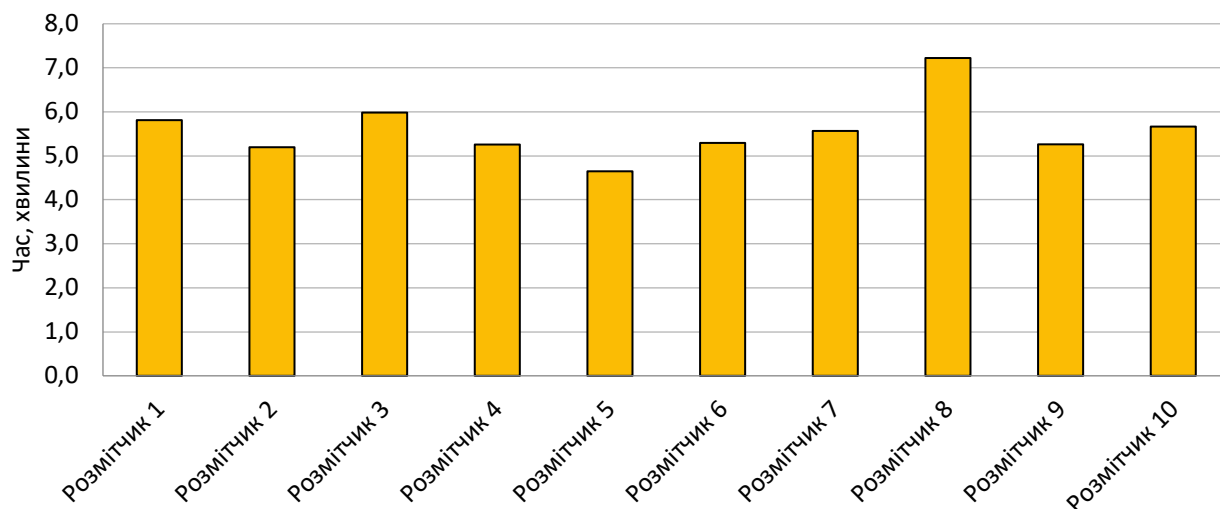


Рисунок 6.7 – Медіанний час розмітки відео з автоматизацією підходом з нульовим навчанням

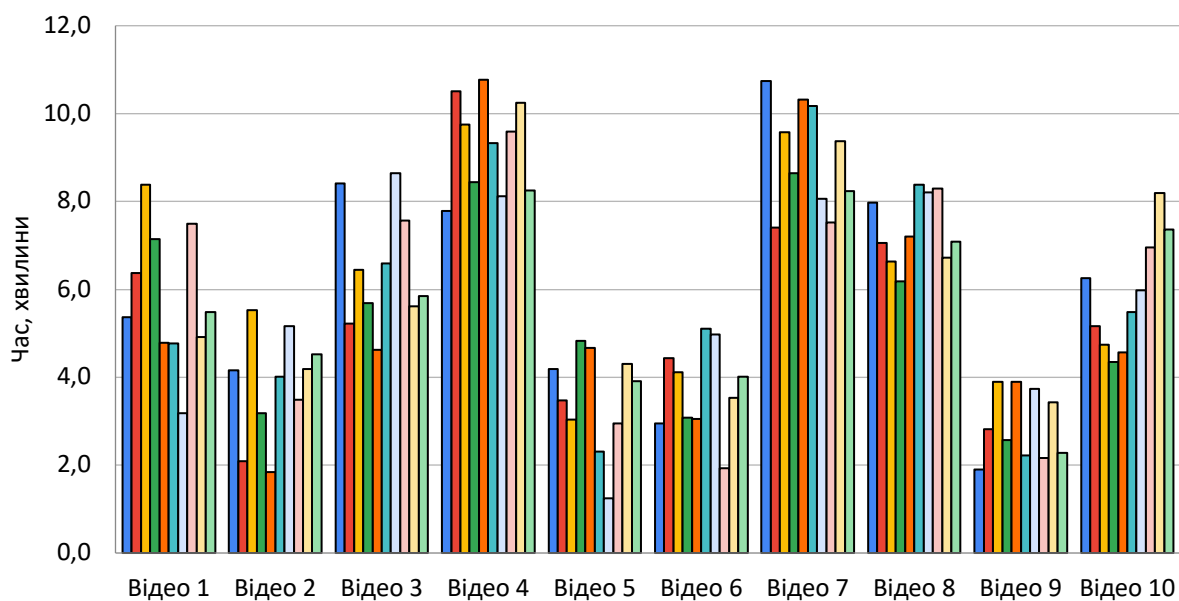


Рисунок 6.8 – Інфографіка по часу, затраченому на розмітку кожного відео кожним розмітчиком даних, для автоматизації з підходом нульового навчання

Вихідні дані для рисунків 6.7 та 6.8 наведено в додатку Г. З цих рисунків можемо зробити висновок, що медіанний час розмітки обраних відео підходом з використанням методів нульового навчання складав 5,4 хв, що є прискоренням на 79% відносно підходу без використання автоматизації процесу розмітки даних. Середнє значення та стандартне відхилення розподілу складають 5,8 та 2,5 хвилин відповідно, 95-відсотковий довірчий інтервал – (5,2; 6,3).

Оскільки підходи активного навчання потребують певного часу на те, щоб вийти на оптимальну якість роботи, то в рамках даного дисертаційного дослідження використано підхід, в якому кожне наступне відео розмічається моделлю активного навчання з більшою кількістю опрацьованих даних.

Тобто відео 1 було розмічене без автоматизації, оскільки модель активного навчання була навчена на 0 відео. Відео 2 – розмічене з автоматизацією після навчання моделі на 1000 відео, відео 3 – на 2000 відео, і тд. Для проведення експерименту було обрано крок в 1000 відео як оптимальну опцію, що відповідає приблизно двом тижням роботи 10 розмітчиків даних.

Медіанний час розмітки відео з автоматизацією підходом активного навчання для усіх розмітчиків даних наведено на рисунку 6.9. Як було вище згадано, ефективність підходів активного навчання росте з кожною ітерацією перенавчання, тому на рисунку 6.10 наведено порівняння медіанного часу розмітки кожним розмітчиком даних кожного відео з використанням підходів активного та нульового навчання для автоматизації процесу розмітки. Детальна інфографіка по часу розмітки кожного відео кожним розмітчиком наведено на рисунку 6.11. Вихідні дані для рисунків 6.9, 6.10, та 6.11 наведено в додатку Д.

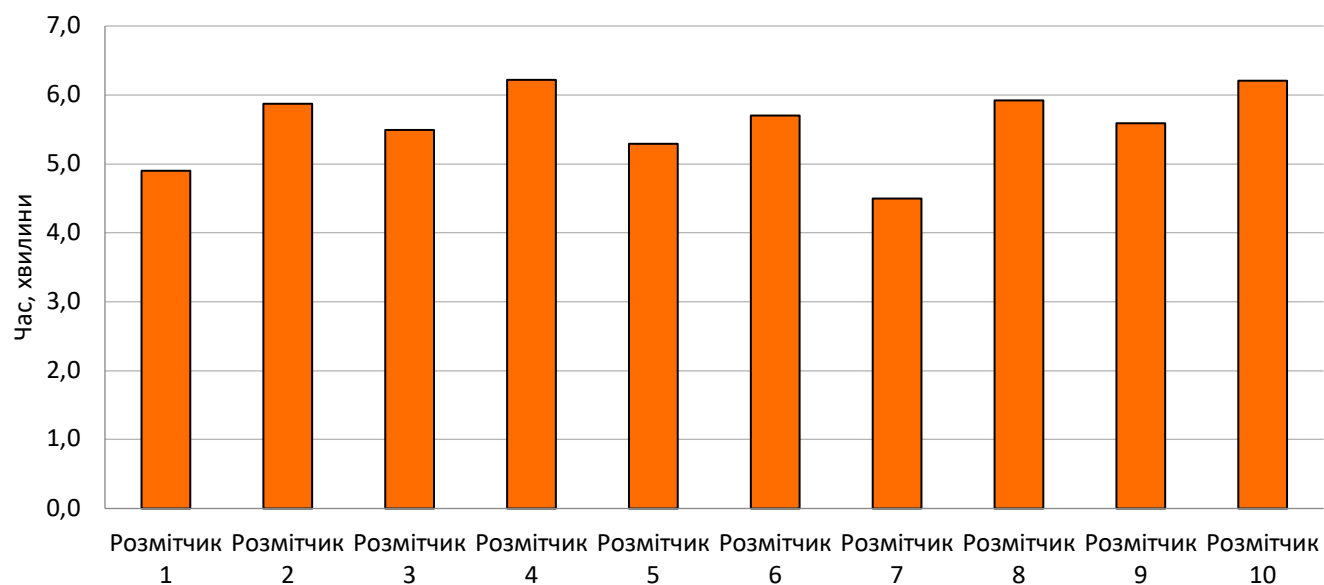


Рисунок 6.9 – Медіанний час розмітки відео з автоматизацією підходом з активним навчанням

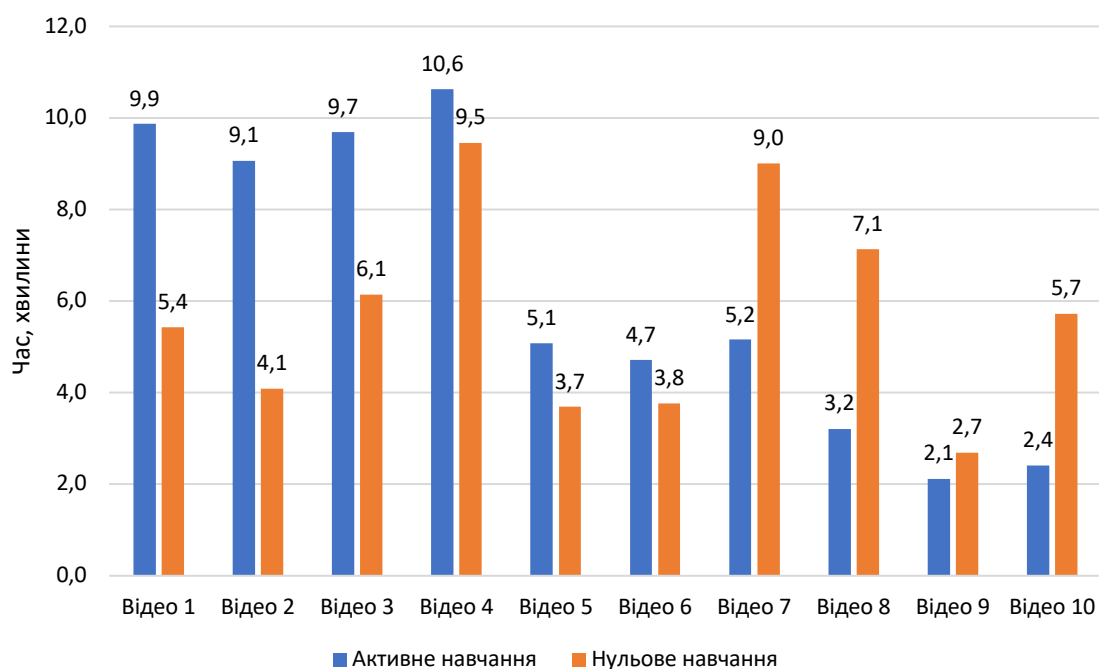


Рисунок 6.10 – Порівняння медіанного часу розмітки кожного відео для методів нульового та активного навчання

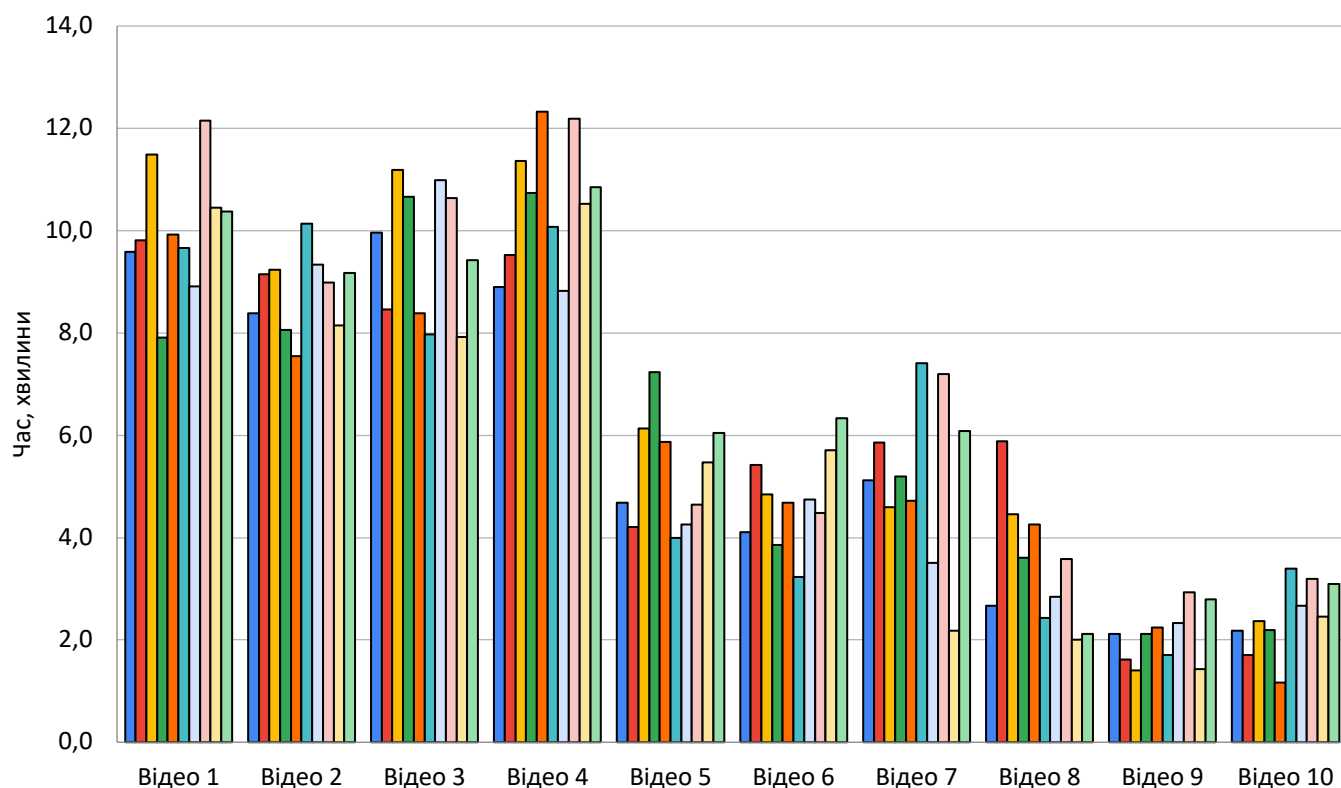


Рисунок 6.11 – Інфографіка по часу, затраченому на розмітку кожного відео кожним розмітчиком даних, для автоматизації з підходом нульового навчання

З наведених результатів можемо зробити висновок, що медіанний час розмітки обраних відео підходом з використанням методів активного навчання склав 5,8 хвилин, тобто швидше на 62% відносно підходу без використання автоматизації процесу розмітки даних, але він виявився повільнішим, ніж підхід з нульовим навчанням на 8%. Середнє значення та стандартне відхилення розподілу складають відповідно 6,2 та 3,3 хвилин відповідно, 95-відсотковий довірчий інтервал – (5,6; 6,8).

Відповідно до підходів з попередніх підрозділів було проведено експеримент по автоматизації розмітки відеоданих з запропонованою дуальною архітектурою. Медіанний час розмітки відео наведено на рисунку 6.12. Детальна інфографіка по часу

розмітки кожного відео кожним розмітчиком даних наведено на рисунку 6.13. Вихідні дані для рисунків 6.12 та 6.13 наведено в додатку Е.

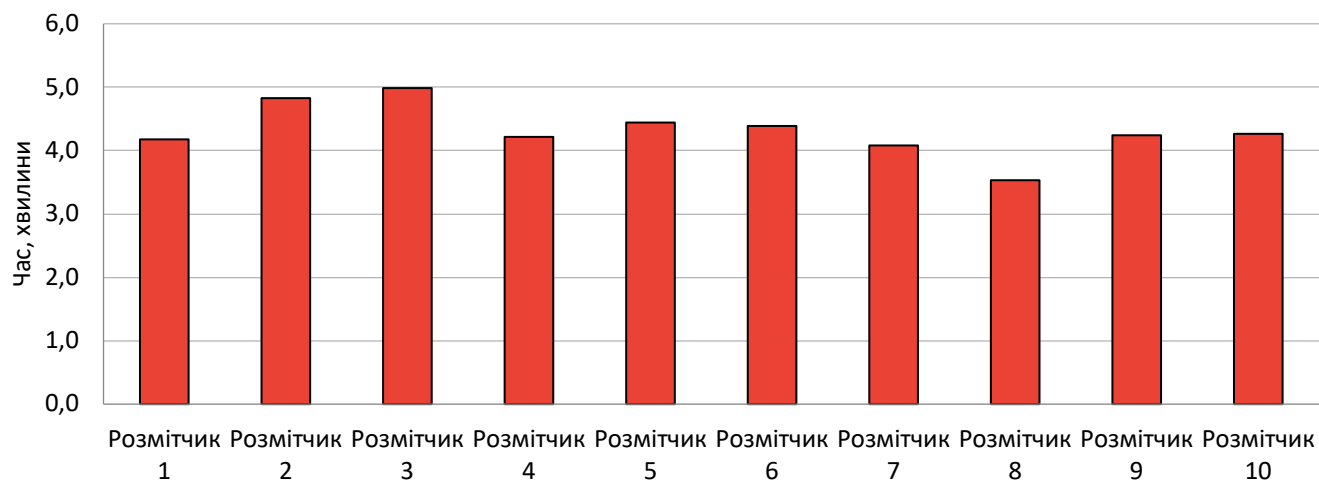


Рисунок 6.12 – Медіанний час розмітки відео з автоматизацією запропонованим підходом з дуальною архітектурою

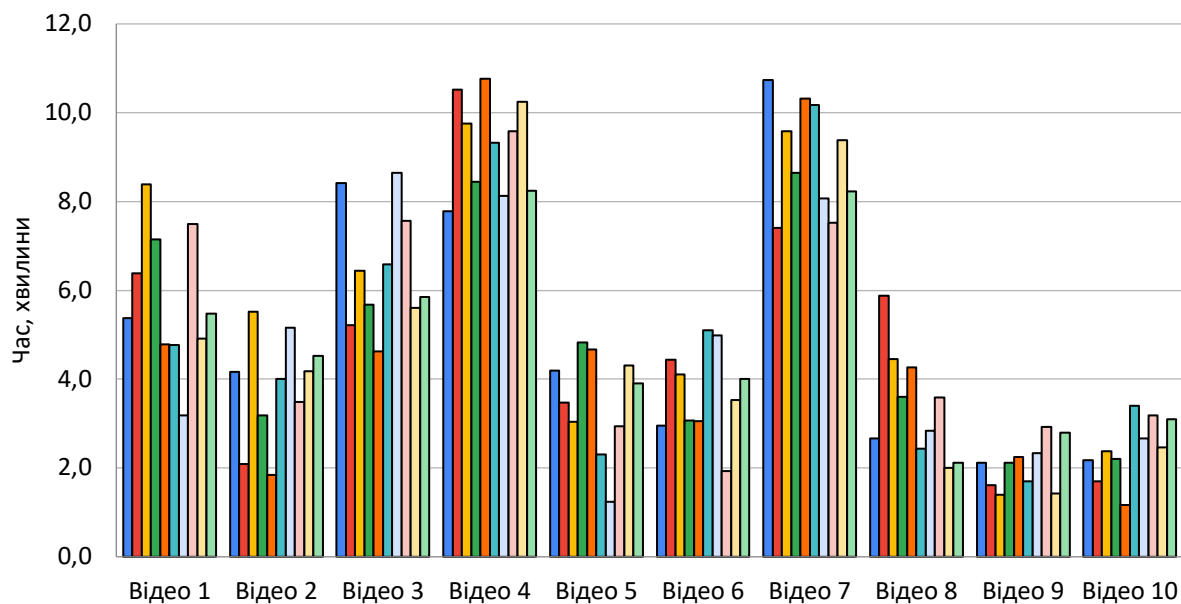


Рисунок 6.13 – Інфографіка по часу, затраченому на розмітку кожного відео кожним розмітчиком даних, для автоматизації з запропонованим підходом дуальної архітектури

З наведених результатів можемо зробити висновок, що медіанний час розмітки обраних відео запропонованим підходом з використанням дуальної архітектури склав 4,3 хвилини, тобто швидше на 125% відносно підходу без використання будь-яких підходів по автоматизації процесу розмітки даних, на 25% відносно підходу з використанням підходу нульового навчання та на 34% відносно підходу з використанням активного навчання. Середнє значення та стандартне відхилення розподілу – 4,9 та 2,7 хвилин відповідно. 95-відсотковий довірчий інтервал – (4,4; 5,5).

## 6.5 Порівняння різних підходів до автоматизації розмітки даних

Порівняльний аналіз усіх підходів наведено на рисунку 6.14.

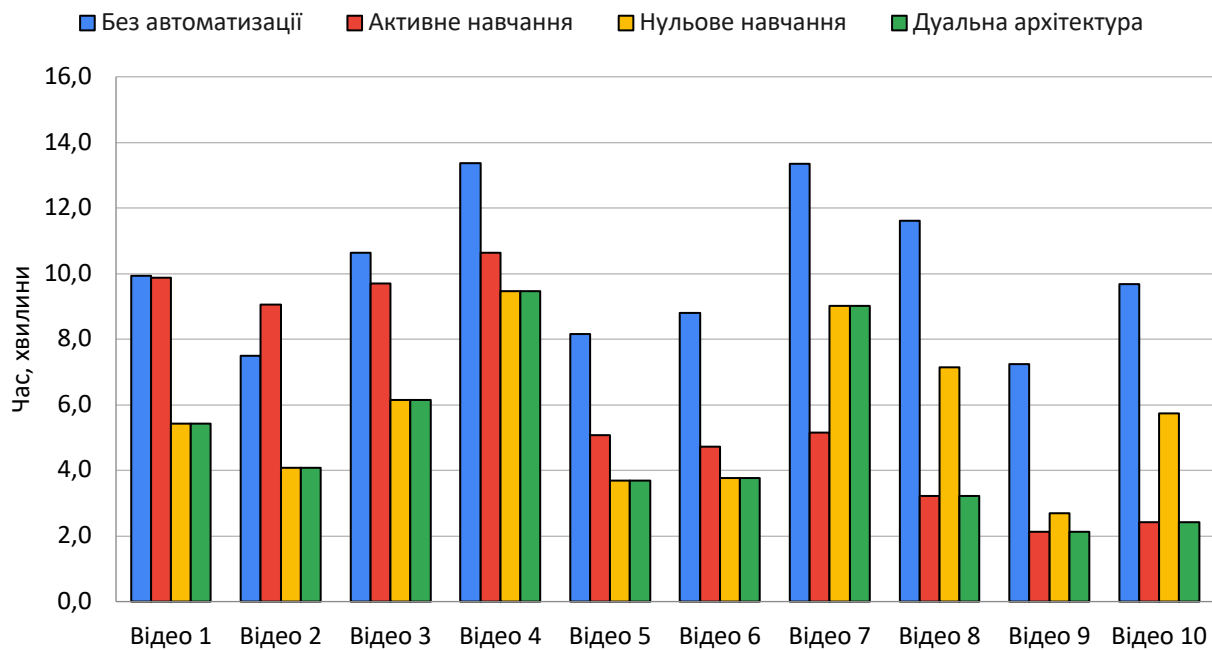


Рисунок 6.14 – Порівняння медіанного часу розмітки кожного відео різними підходами

З наведених результатів можна зробити висновок, що запропонований підхід поводить себе як підходи з нульовим навчанням на ранніх етапах розмітки, але по мірі того як накопичується все більша кількість даних, доступних для методів активного



навчання, починає поводитися як метод активного навчання, що дозволяє ще більше зменшити час на розмітки даних, тим самим поєднуючи переваги обох підходів та усуваючи їх недоліки.

Основні статистичні характеристики експериментальних результатів наведено в таблиці 5.1.

Таблиця 5.1 – Основні статистичні характеристики проведених експериментів

Характеристика	Без автоматизації	З автоматизацією		
		підходом активного навчання	підходом нульового навчання	дуальним підходом
Медіанний час, хв	9,7	5,8	5,4	4,3
Середній час, хв	10,2	6,2	5,8	4,9
Стандартне відхилення, хв	2,3	3,3	2,5	2,7
Межа похибки, хв	0,45	0,65	0,53	0,53
95% довірчий інтервал	(9,7; 10,6)	(5,6; 6,8)	(5,2; 6,3)	(4,4; 5,5)
p-value відносно підходу без автоматизації для 5% критичної області	-	$6,38 \cdot 10^{-19}$	$9,03 \cdot 10^{-29}$	$1,07 \cdot 10^{-33}$

Для оцінки статистичної значимості отриманих результатів було застосовано t-критерій Стюдента. Результати, наведені в таблиці 5.2, показали, що порівняння всіх підходів автоматизації з відсутністю автоматизації дало статистично значиму різницю з

p-value значно меншим за 0,05. Зокрема, порівняння підходу активного навчання з відсутністю автоматизації дало p-value  $6,38 \cdot 10^{-19}$ , підходу нульового навчання –  $9,03 \cdot 10^{-29}$ , а дуальної архітектури –  $1,07 \cdot 10^{-33}$ , що свідчить про більш високу ефективність підходів з автоматизацією у порівнянні з підходом без автоматизації.

Таблиця 5.2 – Статистична значимість отриманих результатів за t-критерієм Стьюдента

		Без автома- тизації	З автоматизацією		
			підходом активного навчання	підходом нульового навчання	дуальним підходом
Без автоматизації		-	$6,38 \cdot 10^{-19}$	$9,03 \cdot 10^{-29}$	$1,07 \cdot 10^{-33}$
З автома- тизацією	підходом активного навчання	$6,38 \cdot 10^{-19}$	-	0,2955	0,0241
	підходом нульового навчання	$9,03 \cdot 10^{-29}$	0,2955	-	0,0338
	дуальним підходом	$1,07 \cdot 10^{-33}$	0,0241	0,0338	-

Додатковий аналіз показав, що порівняння між активним та нульовим навчанням не дало статистично значимої різниці (p-value = 0,2955), що вказує на подібну ефективність цих двох підходів. Однак, порівняння підходів активного навчання з дуальною архітектурою (p-value = 0,0241) та підходів нульового навчання з дуальною архітектурою (p-value = 0,0338) показало значущу різницю, що свідчить про перевагу дуальної архітектури над іншими підходами автоматизації.

Загалом, проведений аналіз демонструє, що впровадження автоматизації значно підвищує продуктивність, особливо при використанні дуальної архітектури, яка

показала найнижчий середній час виконання завдань та статистично значущу перевагу над іншими методами автоматизації.

## **6.6 Висновки до розділу**

У розділі наведено детальний опис експериментального дослідження ефективності розмітки відеоданих без та з використанням автоматизації процесу розмітки та виконано статистичний аналіз результатів експериментів. Описано використовуваний набір даних для експериментів та умови проведення експериментів. Зокрема, надано формалізований опис процесу розмітки даних та визначені підходи до організації цього процесу. Також здійснено постановку задачі експериментів, яка включає мету проєкту розмітки даних, визначення зацікавлених сторін, створення дорожньої карти розмітки, вибір розмітчиків та інструменту для розмітки, створення керівних принципів розмітки та управління процесом розмітки.

У дослідженні розмітки даних для задач детекції об'єктів за допомогою інструменту розмітки CVAT, описані умови проведення експериментів та наведені результати експериментів для розмітки відеоданих без та з автоматизацією з використанням підходів нульового та активного навчання, а також дуального підходу. Результати експериментів показали, що медіанний час розмітки обраних відео запропонованим дуальним підходом склав 4,3 хвилини. Це забезпечило прискорення на 125% у порівнянні з підходом без автоматизації, на 25% у порівнянні з підходом нульового навчання та на 34% у порівнянні з підходом активного навчання. Виконаний статистичний аналіз свідчить, що впровадження автоматизації значно підвищує продуктивність, особливо при використанні дуального підходу, який показав найнижчий середній час виконання завдань та статистично значиму перевагу над іншими методами автоматизації.

Таким чином, мету по пришвидшенню процесу розмітки об'єктів на відеоданих, поставлену в рамках цієї дисертаційної роботи, досягнуто. Розроблений дуальний підхід довів свою ефективність, зменшуючи суттєво час, необхідний для розмітки, і підвищуючи продуктивність процесу розмітки.

## ВИСНОВКИ

У дисертаційній роботі вирішено актуальне наукове завдання підвищення швидкості розмітки відеоданих у контексті завдань детекції об'єктів за рахунок розробки методів та програмного забезпечення для розмітки відеоданих, що підвищують точність розмітки даних, виконану автоматизованими засобами, та зменшують час, необхідний на дорозмітку людиною.

У результаті дисертаційного дослідження отримано такі наукові і практичні результати:

1. Запропоновано дуальну архітектуру програмного забезпечення для автоматизованої розмітки даних, яка, за рахунок методу адаптивно-агрегованого навчання нейромережі, забезпечує пришвидшення процесу розмітки та, на відміну від існуючих аналогів, дає змогу ефективного застосування нульового та активного навчання нейромережі для розмітки даних та більш гнучкого використання програмного забезпечення для різноманітних задач комп'ютерного зору. Розроблені методи та програмні засоби сприяють підвищенню швидкості розмітки даних для задач детекції на відео, знижуючи час, витрачений на розмітку даних, на 125% відносно підходів без автоматизації розмітки даних та на 25% відносно інших підходів автоматизації.
2. Запропоновано метод пріоритезації розмітки відеоданих для навчання нейронної мережі, який, за рахунок відбору найскладніших зразків для навчання, підвищує якість набору даних без проведення попередньої розмітки відео, внаслідок чого збільшується точність детекції об'єктів на відео, та, на відміну від існуючих підходів, базується виключно на автоматично згенерованій репрезентації даних. Завдяки впровадженню запропонованого підходу у процес навчання для нейромереж на базі архітектури Video Swin Transformer було отримано підвищення точності на задачі

розпізнавання дій на відео на Kinetics-400 (точність 84,4% топ-1 і 96,5% топ-5 точності), що підвищує точність роботи методу в середньому на 1,3% у порівнянні з використанням випадкової вибірки відео з того ж розподілу даних у порівнянні з оригінальним підходом до навчання Video Swin Transformer.

3. Запропоновано ітеративний метод вибору ключових кадрів на відео, що дає змогу визначати ключові кадри та сегменти відео з поступовим підвищенням точності, та, на відміну від існуючих методів, враховувати динамічно зміни контенту відео для вибору ключових кадрів, підвищуючи точність сегментації та зменшуючи обсяг відеоданих для обробки. Практичні дослідження запропонованого методу показали підвищення якості узагальнення змісту відео на наборах даних SumMe та TVSum на +0,6% та +0,5% відповідно та зменшення об'єму відеоданих для подальшої обробки в середньому в 6 разів у порівнянні з методом D-KTS.
4. Запропоновано метод агрегації знань між текстовою та візуальною частинами у візуально-мовній моделі (VLM) для обробки складних мультимодальних взаємодій, що забезпечує більш високу точність розпізнавання складних сцен на відео та їх опису у порівнянні з існуючими аналогами. Запропонований підхід Attr4Vis, об'єднуючи всі вищезгадані інновації, досягає точності 91,5% Top-1 на наборі даних Kinetics-400, що є більш високим результатом у порівнянні з існуючими підходами. Запропонований підхід має перспективи для вивчення передачі знань між візуальною та текстовою модальностями VLM моделей та, водночас, підвищує точність розпізнавання атрибутів.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Russakovsky, O., Deng, J., Su, H. et al. (2015). ImageNet Large Scale Visual Recognition Challenge. *Int J Comput Vis* 115, 211-252.
2. Papers with Code. “Image Classification”, 2024. Available at: <http://paperswithcode.com/task/image-classification>, last accessed 2024/07/13
3. Lecun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, vol. 521, no. 7553, 436-444.
4. Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*. MIT Press, <http://www.deeplearningbook.org>
5. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.
6. Gartner. “Gartner Forecasts Worldwide Artificial Intelligence Software Market to Reach \$62 Billion in 2022”, 2021. Available at: <https://www.gartner.com/en/newsroom/press-releases/2021-11-22-gartner-forecasts-worldwide-artificial-intelligence-software-market-to-reach-62-billion-in-2022>, last accessed 2024/07/13
7. Зарічковий, О. А. Алгоритмічне забезпечення для розмітки надвеликих об’ємів даних для задачі детекції об’єктів методами комп’ютерного зору : магістерська дис. : 121 Інженерія програмного забезпечення / Зарічковий Олександр Анатолійович. - Київ, 2021. - 119 с.
8. Amazon Web Services. “Amazon SageMaker”, 2024. Available at: <https://aws.amazon.com/sagemaker>, last accessed 2024/07/13
9. Microsoft Azure. “Microsoft Azure AI Platform”, 2024. Available at: <https://azure.microsoft.com/en-us/overview/ai-platform>, last accessed 2024/07/13

10. IBM. “IBM Watson”, 2024. Available at: <https://www.ibm.com/watson>, last accessed 2024/07/13
11. Google Cloud. “ Innovate faster with enterprise-ready AI, enhanced by Gemini models ”, 2024. Available at: <https://cloud.google.com/vertex-ai/>, last accessed 2024/07/13
12. MIT Technology Review. “Embracing the rapid pace of AI”, 2021. Available at: <https://www.technologyreview.com/2021/05/19/1025016/embracing-the-rapid-pace-of-ai/>, last accessed 2024/07/13
13. Sharma A., Virmani T., Pathak V., et al. Artificial Intelligence-Based Data-Driven Strategy to Accelerate Research, Development, and Clinical Trials of COVID Vaccine. *Biomed Res Int.* 2022 Jul 6;2022:7205241. doi: 10.1155/2022/7205241. PMID: 35845955; PMCID: PMC9279074
14. Wang, S., Ye, Q. (2021). Deep learning based video data annotation. *Information Fusion*, 72, 179-190
15. Li, Y., Song, H., Huang, W. (2020). An efficient video annotation method based on deep learning. *Journal of Visual Communication and Image Representation*, 72, 102009
16. Lionbridge Aurora AI. “ Lionbridge Aurora AI”, 2024. Available at: <https://www.lionbridge.com/>, last accessed 2024/07/13
17. TensorFlow. “ An end-to-end platform for machine learning”, 2024. Available at: <https://www.tensorflow.org>, last accessed 2024/07/13
18. PyTorch. “PyTorch GET STARTED ”, 2024. Available at: <https://pytorch.org>, last accessed 2024/07/13
19. Scikit-learn. “Scikit-learn Machine Learning in Python”, 2024. Available at: <https://scikit-learn.org/stable>, last accessed 2024/07/13
20. Hossain, M. S., Muhammad, G., Hasan, M. M. (2020). The Challenges of Data Annotation for Machine Learning: A Review. *CoRR*, vol. abs/2004.03705, <https://arxiv.org/abs/2004.03705> (2020)



21. Yin, X., Zhang, X., Chen, L. (2018). Data Annotation in the Age of Deep Learning. *IEEE Intelligent Systems*, 33(4), 9-19
22. Marge, M., Rostami, M. (2019). The Future of Data Annotation: Challenges and Opportunities. *CoRR*, vol. abs/1903.01585, <https://arxiv.org/abs/1903.01585> (2019)
23. Gao, X., Zhang, Z., Lu, X. (2019). The Importance of Data Annotation in Machine Learning. *IEEE Access*, 7, 9650-9661
24. Zhou, X., Zhu, X., Liu, Y. (2017). Annotating Data for Machine Learning: A Review of Current Strategies and Future Directions. *IEEE Transactions on Big Data*, 3(4), 380-404
25. Sigma A. I. (2022). Building Scalable Data Annotation Strategy. Available at: <https://sigma.ai/scalable-data-annotation-strategy/>, last accessed 2024/07/13
26. DataScienceCentral.com - Big Data News and Analysis. “Annotation Strategies for Computer Vision Training Data”, 2022. Available at: <https://www.datasciencecentral.com/annotation-strategies-for-computer-vision-training-data/>, last accessed 2024/07/13
27. Golle, P. (2008). Machine learning attacks against the asirra CAPTCHA. *Proceedings of the ACM Conference on Computer and Communications Security*. 535-542. 10.1145/1455770.1455838.
28. Kobatake H. and Yoshinaga Y. (1996). Detection of spicules on mammogram based on skeleton analysis. *IEEE Trans. Med. Imag.*, 15(3), 235–245.
29. Jia Y., Shelhamer E., Donahue J., et al. (2014). Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*.
30. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *NIPS*, 2012
31. Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, “Realtime multi-person 2d pose estimation using part affinity fields,” in *CVPR*, 2017

32. C. Chen, A. Seff, A. L. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in ICCV, 2015
33. P. K. Mishra and G. P. Saroha, "A study on video surveillance system for object detection and tracking," 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2016, pp. 221-226
34. Ghasemieh A., Kashef R. (2022). 3D object detection for autonomous driving: Methods, models, sensors, data, and challenges. *Transportation Engineering*, volume 8, 100115. ISSN 2666-691X. doi: 10.1016/j.treng.2022.100115
35. A. Coates and A. Y. Ng (2010). Multi-camera object detection for robotics. 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 2010, pp. 412-419, doi: 10.1109/ROBOT.2010.5509644
36. Lecron, F., Benjelloun, M., Mahmoudi, S. (2012). Descriptive Image Feature for Object Detection in Medical Images. In: Campilho, A., Kamel, M. (eds) *Image Analysis and Recognition. ICIAR 2012. Lecture Notes in Computer Science*, vol 7325. Springer, Berlin, Heidelberg. doi: 10.1007/978-3-642-31298-4\_39
37. P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, Kauai, HI, USA, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990517
38. Liu, W. et al. (2016). SSD: Single Shot MultiBox Detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) *Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science()*, vol 9905. Springer, Cham. doi: 10.1007/978-3-319-46448-0\_2
39. Redmon J., Divvala S., Girshick R., Farhadi A. (2015). You Only Look Once: Unified, Real-Time Object Detection. *CoRR*, vol. abs/1506.02640, <http://arxiv.org/abs/1506.02640>
40. Cai Z. and Vasconcelos N. (2018). Cascade R-CNN: Delving into high quality object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*.

41. He K., Gkioxari G., Dollar P., and Girshick R. B. (2017). Mask R-CNN. In ICCV, pp. 2980–2988.
42. Carion, N., Massa, F., Synnaeve, G., Usunier, N., et al. (2020). End-to-End Object Detection with Transformers. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12346. Springer, Cham. doi: 10.1007/978-3-030-58452-8\_13
43. Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, et al. Emerging Properties in Self-Supervised Vision Transformers. CoRR, vol. abs/2104.14294, <http://arxiv.org/abs/2104.14294> (2021)
44. A. Radford, J. W. Kim, C. Hallacy, et al. Learning Transferable Visual Models From Natural Language Supervision. Proceedings of the 38th International Conference on Machine Learning, PMLR 139:8748-8763, 2021
45. M. Everingham, L. J. V. Gool, C.K. I. Williams, et al. The pascal visual object classes (VOC) challenge. International Journal of Computer Vision, 88(2):303–338, 2010
46. Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In ECCV, pp. 740-755, 2014
47. R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in CVPR, 2014
48. R. Girshick, Fast R-CNN, in: ICCV '15: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV) December 2015, pp. 1440 – 1448 <https://doi.org/10.1109/ICCV.2015.169>
49. S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards realtime object detection with region proposal networks,” in NIPS, 2015, pp. 91–99.
50. W. Ao, C. Hui, L. Lihao, C. Kai, L. Zijia, H. Jungong, D. Guiguang. YOLOv10: Real-Time End-to-End Object Detection. CoRR, vol. abs/2405.14458, <http://arxiv.org/abs/2405.14458> (2024)

51. Dim P. Papadopoulos, Jasper R. R. Uijlings, Frank Keller, and Vittorio Ferrari. We don't need no bounding-boxes: Training object class detectors using only human verification. In CVPR, pp. 854–863, 2016
52. D. P. Papadopoulos, J. R. R. Uijlings, F. Keller, and V. Ferrari. Extreme clicking for efficient object annotation. In ICCV, pp. 4940–4949, 2017
53. J. R. Uijlings, K. E. Sande, T. Gevers, and A. W. Smeulders. 2013. Selective Search for Object Recognition. *Int. J. Comput. Vision* 104, 2 (September 2013), 154–171. <https://doi.org/10.1007/s11263-013-0620-5>
54. P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 2010
55. Qiu, Z., Yao, T., and Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5533–5541 (2017)
56. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 305–321 (2018)
57. J. Carreira et al., “Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset,” in 2017 IEEE Conf. on Comp. Vision and Pattern Rec., pp. 4724–4733
58. He, K.; Zhang, X.; Ren, S.; and Sun, J. Deep residual learning for image recognition. In CVPR, 770–778. 2016
59. Simonyan, K.; and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. 2014
60. Ioffe, S.; and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 448–456. PMLR. 2015
61. A. Vaswani et al., “Attention is All You Need”, in 31st Int. Conf. on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 6000–6010

62. Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. International Conference on Learning Representations (2021)
63. Medium. “DeiT — Training Data-Efficient Image Transformer & distillation through attention, Facebook AI -ICML’21”. Available at: <https://medium.com/aiguys/deit-training-data-efficient-image-transformer-distillation-through-attention-facebook-ai-9b60aea3da07>, last accessed 2024/07/13
64. Liu, Z., Lin, Y., Cao, Y., et al.: Swin transformer: Hierarchical vision transformer using shifted windows. CoRR, vol. abs/2103.14030, <http://arxiv.org/abs/2103.14030> (2021)
65. Arnab, A.; Dehghani, M.; Heigold, G.; Sun, C.; Lucic, M.; and Schmid, C. Vivit: A video vision transformer. In ICCV, 6836–6846. 2021
66. Liu, Z., Ning, J., Cao, Y., Wei, Y., et al: Video Swin Transformer. CoRR, vol. abs/2106.13230, <http://arxiv.org/abs/2106.13230> (2021)
67. Fan, H., Xiong, B., Mangalam, K., et al.: Multiscale vision transformers. CoRR, vol. abs/2104.11227, <http://arxiv.org/abs/2104.11227> (2021)
68. Bertasius, G.; Wang, H.; and Torresani, L. Is SpaceTime Attention All You Need for Video Understanding? In ICML, 813–824. PMLR. 2021
69. Van den Oord, A.; Li, Y.; and Vinyals, O. Representation learning with contrastive predictive coding. arXiv eprints, arXiv–1807. 2018
70. J. Yu, Z. Wang, V. Vasudevan, et al. Coca: Contrastive captioners are image-text foundation models. arXiv preprint arXiv:2205.01917, 2022
71. Jia, C.; Yang, Y.; Xia, Y.; et al. Scaling up visual and vision-language representation learning with noisy text supervision. In ICML, 4904–4916. PMLR. 2021
72. Li, J.; Li, D.; Xiong, C.; and Hoi, S. Blip: Bootstrapping language-image pre-training for unified visionlanguage understanding and generation. In ICML. 2022

73. Yang, J.; Li, C.; Zhang, P.; Xiao, B.; Liu, C.; Yuan, L.; and Gao, J. Unified contrastive learning in image-text-label space. In CVPR, 19163–19173. 2022
74. Wang, L.; Tong, Z.; Ji, B.; and Wu, G. TDN: Temporal difference networks for efficient action recognition. In CVPR, 1895–1904. 2021
75. Zhao, S.; Zhu, L.; Wang, X.; and Yang, Y. CenterCLIP: Token Clustering for Efficient Text-Video Retrieval. SIRIR. 2022
76. Luo, H.; Ji, L.; Zhong, M.; et al. CLIP4Clip: An empirical study of CLIP for end to end video clip retrieval and captioning. *Neurocomputing*, 508: 293–304. 2022
77. C. Ju, T. Han, K. Zheng, Y. Zhang, and W. Xie. Prompting visual-language models for efficient video understanding. In ECCV, pp. 105–124. Springer, 2022
78. Wu, W., Wang, X., Luo, H. et al. Bidirectional Cross-Modal Knowledge Exploration for Video Recognition with Pre-trained Vision-Language Models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023
79. Wu, W.; Luo, H.; Fang, B.; Wang, J.; and Ouyang, W. Cap4Video: What Can Auxiliary Captions Do for Text-Video Retrieval? In CVPR. 2023
80. Y. Wang, K. Li, Y. Li, et al. InternVideo: General Video Foundation Models via Generative and Discriminative Learning. *arXiv preprint arXiv:2212.03191*. 2022
81. L. Yuan, D. Chen, Yi-Ling Chen, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021
82. Wang, L. and Yang, N. and Huang, X. et al. Text Embeddings by Weakly-Supervised Contrastive Pre-training. *arXiv preprint arXiv:2212.03533*. 2022
83. J. Yang, C. Li, P. Zhang, et al. Unified contrastive learning in image-text-label space. In CVPR, 2022
84. Z. Wang, J. Yu, A. Wei Yu, et al. SimVLM: Simple visual language model pretraining with weak supervision. In ICLR, 2022
85. P. Wang, A. Yang, R. Men, et al. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In ICML, 2022

86. Tsu-Jui Fu, L. Li, Z. Gan, K. Lin, et al. Violet: End-to-end video-language transformers with masked visual-token modeling. arXiv preprint arXiv:2111.12681, 2021
87. A. J. Wang, Yixiao Ge, Rui Yan, et al. All in one: Exploring unified video-language pre-training. arXiv preprint arXiv:2203.07303, 2022
88. Linjie Li, Zhe Gan, Kevin Lin, et al. Lavender: Unifying video-language understanding as masked language modeling. arXiv preprint arXiv:2206.07160, 2022
89. R. Zellers, J. Lu, X. Lu, et al. MERLOT reserve: Neural script knowledge through vision and language and sound. In CVPR, 2022
90. C. Sun, A. Myers, C. Vondrick, et al. Videobert: A joint model for video and language representation learning. ICCV, 2019
91. L. Zhu and Y. Yang. Actbert: Learning global-local video-text representations. CVPR, 2020
92. J. Lei, L. Li, L. Zhou, et al. Less is more: Clipbert for video-and-language learning via sparse sampling. In CVPR, 2021
93. M. Bain, A. Nagrani, G. Varol, and A. Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In ICCV, 2021
94. H. Xu, G. Ghosh, P.-Y. Huang, et al. Videoclip: Contrastive pre-training for zero-shot video-text understanding. arXiv preprint arXiv:2109.14084, 2021
95. X. Hu, Z. Gan, J. Wang, et al. Scaling up vision language pre-training for image captioning. In CVPR, 2022
96. A. Miech, J.-B. Alayrac, L. Smaira, et al. End-to-end learning of visual representations from uncurated instructional videos. In CVPR, 2020
97. R., E., Kalman. (1960). A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering, 82(1):35-45. doi: 10.1115/1.3662552
98. B. K.P. Horn and B. G. Schunck. 1980. Determining Optical Flow. Technical Report. Massachusetts Institute of Technology, USA.

99. Farneback, Gunnar. "Two-frame motion estimation based on polynomial expansion." Scandinavian conference on Image analysis. Springer, Berlin, Heidelberg, 2003
100. Lucas, Bruce D., and Kanade, T.. "An iterative image registration technique with an application to stereo vision." Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI), 1981
101. Cognilytica. "Cognilytica: AI & Data Best Practices Training & Certification", 2024. Available at: <https://www.cognilytica.com/>, last accessed 2024/07/13
102. The Washington Post. "Hottest job in China's hinterlands: Teaching AI to tell a truck from a turtle", 2019. Available at: <https://www.washingtonpost.com/business/2019/09/26/hottest-job-chinas-hinterlands-teaching-ai-tell-truck-turtle/>, last accessed 2024/07/13
103. Encord. "The Complete Guide to Image Annotation for Computer Vision", 2022. Available at: <https://encord.com/blog/image-annotation-guide/>, last accessed 2024/07/13
104. NV5. "Annotate Images for Object Detection", 2023. Available at: <https://www.nv5geospatialsoftware.com/docs/AnnotateImagesForObjectDetection.html>, last accessed 2024/07/13
105. Labelbox. "Best practices for successful image annotation", 2023. Available at: <https://labelbox.com/guides/image-annotation/>, last accessed 2024/07/13
106. Lionbridge AI. "How Much Do Image Annotation Services Cost?", 2019. Available at: <https://lionbridge.ai/articles/how-much-do-image-annotation-services-cost/>, last accessed 2019/05/13
107. "Pricing | Data Labelling Service", 2024. Available at: <https://cloud.google.com/ai-platform/data-labeling/pricing>, last accessed 2024/07/13
108. T. Tseng, A. Stent, D, Maida. Best Practices for Managing Data Annotation Projects. CoRR, vol. abs/2009.11654, <http://arxiv.org/abs/2009.11654> (2020)



109. Lionbridge AI. “5 Approaches to Data Labeling for Machine Learning Projects”, 2019. Available at: <https://lionbridge.ai/articles/5-approaches-to-data-labeling-for-machine-learning-projects/>, last accessed 2019/05/13
110. Wu, W., He, D., Lin, T., et al.: MVFNet: Multi-View Fusion Network for Efficient Video Recognition. CoRR, vol. abs/2012.06977, <http://arxiv.org/abs/2012.06977> (2020)
111. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. Proceedings of the IEEE international conference on computer vision, pp. 4489–4497 (2015)
112. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7794–7803 (2018)
113. Cao, Y., Xu, J., Lin, S., Wei, F., Hu, H.: Gcnet: Non-local networks meet squeeze excitation networks and beyond. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops (2019)
114. Yin, M., Yao, Z., Cao, Y., et al.: Disentangled non-local neural networks. Proceedings of the European conference on computer vision (2020)
115. Touvron, H., Cord, M., Douze, M., et al.: Training data-efficient image transformers & distillation through attention. CoRR, vol. abs/2012.12877, <http://arxiv.org/abs/2012.12877> (2020)
116. G. Bertasius, H. Wang, and L. Torresani. Is space-time attention all you need for video understanding? In ICML, pp. 813–824. PMLR, 2021
117. Feichtenhofer, C., Fan, H., Malik, J., and He, K.: Slowfast networks for video recognition. Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6202–6211 (2019)
118. Li, X., Zhang, Y., Liu, C., et al.: Vidtr: Video transformer without convolutions. CoRR, vol. abs/2104.11746, <http://arxiv.org/abs/2104.11746> (2021)

119. Zhang, X., Wang, Q., Zhang, J., Zhong, Z.: Adversarial AutoAugment. CoRR, vol. abs/1912.11188, <http://arxiv.org/abs/1912.11188> (2019)
120. Wightman, R., Touvron, H., Jégou, H.: ResNet strikes back: An improved training procedure in timm. CoRR, vol. abs/2110.00476, <http://arxiv.org/abs/2110.00476> (2021)
121. Robinson, J., Chuang, C., Sra, S., Jegelka, S.: Contrastive Learning with Hard Negative Samples. CoRR, vol. abs/2010.04592, <http://arxiv.org/abs/2010.04592> (2020)
122. Lin, T., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal Loss for Dense Object Detection. CoRR, vol. abs/1708.02002, <http://arxiv.org/abs/1708.02002> (2017)
123. Zarichkovyi, A., Stetsenko, I.V. (2023) ‘Hard Samples Make Difference: An Improved Training Procedure for Video Action Recognition Tasks’, Lecture Notes in Networks and Systems, 544, pp. 508-519. Springer, Cham.
124. Zhengxia, Z., Keyan, C., Zhenwei, S., Yuhong, G., Jieping, Y.: Object Detection in 20 Years: A Survey. CoRR, vol. abs/1905.05055, <http://arxiv.org/abs/1905.05055> (2019)
125. Bolun, C., Pengfei, X., Shangxuan, T.: Center Contrastive Loss for Metric Learning. CoRR, vol. abs/2308.00458, <http://arxiv.org/abs/2308.00458> (2023)
126. D. Marutho, S. Hendra Handaka, E. Wijaya and Muljono, "The Determination of Cluster Number at k-Mean Using Elbow Method and Purity Evaluation on Headline News," 2018 International Seminar on Application for Technology of Information and Communication, Semarang, Indonesia, 2018, pp. 533-538, doi: 10.1109/ISEMANTIC.2018.8549751
127. Github. Chen, K., Wang, J., Pang, J., et al.: MMAAction2: open-source toolbox for video understanding. Available at: <https://github.com/open-mmlab/mmaaction2>, last accessed 2024/07/13
128. Kingma, D. P., Ba, J.: Adam: A method for stochastic optimization. CoRR, vol. abs/1412.6980, <http://arxiv.org/abs/1412.6980> (2014)

129. Huang, G., Sun, Y., Liu, Z., Sedra, D., Weinberger, K. Q.: Deep networks with stochastic depth. European conference on computer vision, pp. 646–661, Springer (2016)
130. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*. 15. 1929-1958
131. Github. “How does DropPath's implementation relate to the paper?”. Available at: <https://github.com/huggingface/pytorch-image-models/discussions/593>, last accessed 2024/07/13
132. Evlampios, A., Eleni, A., Alexandros, I.M., Vasileios, M., Ioannis, P.: Video Summarization Using Deep Neural Networks. *CoRR*, vol. abs/2101.06072, <http://arxiv.org/abs/2101.06072> (2021)
133. M. Fei et al. Memorable and rich video summarization. *J. Vis. Commun. Image Represent*. (2017)
134. T.-J. Fu et al., “Attentive and Adversarial Learning for Video Summarization,” in *IEEE Winter Conf. on Applic. of Comp. Vision HI*, USA: IEEE, 2019, pp. 1579–1587
135. S. Hochreiter et al., “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997
136. K. Cho et al., “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation,” in *2014 Conf. on Empirical Methods in Natural Language Processing*. Doha, Qatar: ACL, 2014, pp. 1724–1734
137. J. Kavitha, P. Arockia Jansi Rani, "Design of a Video Summarization Scheme in the Wavelet Domain Using Statistical Feature Extraction", *IJIGSP*, vol.7, no.4, pp.60-67, 2015.DOI: 10.5815/ijigsp.2015.04.07
138. J. Fajtl et al., “Summarizing Videos with Attention” in *Asian Conf. on Comp. Vision 2018 Workshops*. Cham: Springer Int. Publishing, 2018, pp. 39–54
139. Y.-T. Liu et al., “Learning Hierarchical Self-Attention for Video Summarization” in *2019 IEEE Int. Conf. on Image Processing*. IEEE, 2019, pp. 3377–3381

140. W. Zhu, Y. Han, J. Lu and J. Zhou, "Relational Reasoning Over Spatial-Temporal Graphs for Video Summarization" in *IEEE Transactions on Image Processing*, vol. 31, pp. 3017-3031, 2022, doi: 10.1109/TIP.2022.3163855
141. W. Zhu, J. Lu, J. Li and J. Zhou, "DSNet: A Flexible Detect-to-Summarize Network for Video Summarization," in *IEEE Transactions on Image Processing*, vol. 30, pp. 948-962, 2021, doi: 10.1109/TIP.2020.3039886 (2020)
142. W. Zhu, J. Lu, Y. Han, J. Zhou. Learning multiscale hierarchical attention for video summarization. *Pattern Recognition*, Volume 122, 2022, 108312, ISSN 0031-3203, <https://doi.org/10.1016/j.patcog.2021.108312> (2021)
143. M. Rochan et al., "Video Summarization Using Fully Convolutional Sequence Networks," in *Europ. Conf. on Comp. Vision 2018*. Cham: Springer Int. Publishing, 2018, pp. 358– 374 (2018)
144. Statista. "L. Ceci. Statista. Average YouTube video length as of December 2018, by category", 2023. Available at: [www.statista.com/statistics/1026923/youtube-video-category-average-length](http://www.statista.com/statistics/1026923/youtube-video-category-average-length), last accessed 2024/07/13
145. S. Sah, S. Kulhare, A. Gray, S. Venugopalan, E. Prud'Hommeaux and R. Ptucha, "Semantic Text Summarization of Long Videos," 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, USA, 2017, pp. 989-997, doi: 10.1109/WACV.2017.115
146. Wenyi, L., Wenjing, L.: Sliding Window Recurrent Network for Efficient Video Super-Resolution. *CoRR*, vol. abs/2208.11608, <http://arxiv.org/abs/2208.11608> (2022)
147. Song, Y., Vallmitjana, J., Stent, A., Jaimes, A.: Tvsum: Summarizing web videos using titles. In: *Proceedings of the IEEE CVPR*. pp. 5179–5187 (2015)
148. Gygli, M., Grabner, H., Riemenschneider, H., Van Gool, L.: Creating summaries from user videos. In: *Proceedings of the ECCV*. pp. 505–520. Springer (2014)

149. Zarichkovyi, A., Stetsenko, I.V. (2023) ‘Boundary Refinement via Zoom-In Algorithm for Keyshot Video Summarization of Long Sequences’, *Lecture Notes on Data Engineering and Communications Technologies*, 180, pp. 344-359
150. H. Fu et al. Self-attention binary neural tree for video summarization. *Pattern Recognit. Lett.* (2021)
151. B. Zhao et al., “Hierarchical Recurrent Neural Network for Video Summarization,” in *25th ACM Int. Conf. on Multimedia* NY, USA: ACM, 2017, pp. 863–871 (2017)
152. B. Zhao et al., “HSA-RNN: Hierarchical Structure-Adaptive RNN for Video Summarization”, in *2018 IEEE Conf. on Comp. Vision and Pattern Rec.*, pp. 7405–7414 (2018)
153. L. Feng et al., “Extractive Video Summarizer with Memory Augmented Neural Networks”, in *26th ACM Int. Conf. on Multimedia*. NY, USA: ACM, 2018, pp. 976–983 (2018)
154. J. Wang et al., “Stacked Memory Network for Video Summarization”, in *27th ACM Int. Conf. on Multimedia*. NY, USA: ACM, 2019, pp. 836–844 (2019)
155. L. Lebron Casas et al., “Video Summarization with LSTM and Deep Attention Models”, in *25th Int. Conf. on Multimedia Modeling*. Cham: Springer Int. Publishing, 2019, pp. 67–79 (2019)
156. Z. Ji et al., “Deep Attentive and Semantic Preserving Video Summarization”, *Neurocomputing*, vol. 405, pp. 200–207, 2020 (2020)
157. P. Li et al., “Exploring Global Diverse Attention via Pairwise Temporal Relation for Video Summarization”, *Pattern Recognition*, vol. 111, no. 107677, 2021
158. J. Ghauri et al., “Supervised Video Summarization Via Multiple Feature Sets with Parallel Attention,” in *2021 IEEE Int. Conf. on Multimedia and Expo*. CA, USA: IEEE, 2021, pp. 1–6
159. C. Szegedy et al., “Going Deeper with Convolutions”, in *2015 IEEE Conf. on Comp. Vision and Pattern Rec.*, pp. 1–9

160. E. Apostolidis, G. Balaouras, V. Mezaris and I. Patras, "Combining Global and Local Attention with Positional Encoding for Video Summarization", 2021 IEEE International Symposium on Multimedia (ISM), Naples, Italy, 2021, pp. 226-234, doi: 10.1109/ISM52913.2021.00045
161. Belo, L., Caetano, C., Patrocínio J., Zenilton, Guimarães, S. (2016). Summarizing video sequence using a graph-based hierarchical approach. *Neurocomputing*. 173. 1001-1016. 10.1016/j.neucom.2015.08.057
162. X. Ke, B. Chang, H. Wu, F. Xu and S. Zhong, "Towards Practical and Efficient Long Video Summary", ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, Singapore, 2022, pp. 1770-1774, doi: 10.1109/ICASSP43922.2022.9746911
163. D. Potapov, M. Douze, Z. Harchaoui, and C. Schmid, "Category-Specific Video Summarization", in *Computer Vision – ECCV 2014*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, Eds., vol. 8694, pp. 540–555. Springer International Publishing, Cham, 2014
164. Z. Lei, K. Sun, Q. Zhang, and G. Qiu, "User video summarization based on joint visual and semantic affinity graph," in *Proceedings of the 2016 ACM Workshop on Vision and Language Integration Meets Multimedia Fusion*, New York, NY, USA, 2016, p. 45–52, Association for Computing Machinery
165. Hacker Factor. "The hacker factor blog", 2013, Available at: <http://www.hackerfactor.com/blog/?/archives/529-Kind-of-Like-That.html>, last accessed 2024/07/13
166. K. Zhou, Y. Qiao, and T. Xiang, "Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018, vol. 32

167. De Avila, S.E.F., Lopes, A.P.B., da Luz Jr, A., de Albuquerque Ara'ujo, A.: Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters* 32(1), 56–68 (2011)
168. Leslie N. Smith: Cyclical Learning Rates for Training Neural Networks. CoRR, vol. abs/1506.01186, <https://arxiv.org/abs/1506.01186> (2015)
169. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018
170. T. Brown, B. Mann, N. Ryder, M. Subbiah, et al. Language models are few-shot learners. *NeurIPS*, 33:1877–1901, 2020
171. A. Radford, J. Wu, R. Child, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019
172. Z. Zhang, X. Han, Z. Liu, et al. Ernie: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1441–1451, 2019
173. C. Raffel, N. Shazeer, A. Roberts, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020
174. Z. Lin, S. Geng, R. Zhang, et al. Frozen clip models are efficient video learners. In *ECCV*, pp. 388–404. Springer, 2022
175. J. Pan, Z. Lin, X. Zhu, et al. St-adapter: Parameter-efficient image-to-video transfer learning. In *NeurIPS*, 2022
176. T. Yang, Y. Zhu, Y. Xie, et al. Aim: Adapting image models for efficient video understanding. In *ICLR*, 2023
177. B. Ni, H. Peng, M. Chen, et al. Expanding language-image pretrained models for general video recognition. In *ECCV*, pp. 1–18. Springer, 2022
178. M. Wang, J. Xing, and Yong Liu. Actionclip: A new paradigm for video action recognition. *arXiv preprint arXiv:2109.08472*, 2021

179. W. Wu, Z. Sun, and W. Ouyang. Revisiting classifier: Transferring vision-language models for video recognition. In AAAI, 2023
180. Zarichkovyi, A., Stetsenko, I.V. (2024) ‘Attr4Vis: Revisiting importance of attribute classification in Vision-Language Models for Video Recognition’, International Journal of Computing, 23 (1), pp. 94-100
181. J. Carreira, E. Noland, A. Banki-Horvath, et al. A short note about kinetics600. arXiv preprint arXiv:1808.01340, 2018
182. M. Ryoo, A.J. Piergiovanni, A. Arnab, et al. Tokenlearner: Adaptive space-time tokenization for videos. NeurIPS, 34:12786– 12797, 2021
183. S. Yan, X. Xiong, A. Arnab, et al. Multiview transformers for video recognition. In CVPR, pp. 3333–3343, 2022
184. B. Zhang, J. Yu, C. Fifty, et al. Co-training transformer with videos and images improves action recognition. arXiv preprint arXiv:2112.07175, 2021
185. Simonyan, K.; and Zisserman, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. 2014
186. V7. “V7 | The AI Data Engine for Computer Vision & Generative AI”, 2024. Available at: <https://www.v7labs.com/>, <https://www.v7labs.com/>
187. Toolify.ai. “Enhance Your Visual Content with V7 Annotation Tool”, 2023. Available at: <https://www.toolify.ai/ai-news/enhance-your-visual-content-with-v7-annotation-tool-77266>, last accessed 2024/07/13
188. “Labelbox | Data-centric AI Platform for Building & Using AI”, 2023. Available at: <https://labelbox.com/>
189. “Keylabs: Data Annotation Tool - Data Annotation Platform”, 2023. Available at: <https://keylabs.ai/>
190. LabelImg. “LabelImg”, 2024. Available at: <https://github.com/qaprosoft/labelImg>, last accessed 2024/07/13



191. LabelMe. “LabelMe. The Open annotation tool”, 2024. Available at: <http://labelme.csail.mit.edu/Release3.0/>, last accessed 2024/07/13
192. Label Studio io. “Label Studio: Open Source Data Labeling”, 2024. Available at: <https://labelstud.io/>, last accessed 2024/07/13
193. Github. “Computer Vision Annotation Tool (CVAT)”, 2024. Available at: <https://github.com/opencv/cvat>, last accessed 2024/07/13
194. Saher N., Baharom F., Romli R. A Review of Requirement Prioritization Techniques in Agile Software Development. Proceedings of the Knowledge Management International Conference (KMICe), 2018. P. 242-247
195. Kirillov A., Mintun E., Ravi N., Mao H., Rolland C., Gustafson L., Xiao T., Whitehead S., Berg A., Lo W., Dollár P., Girshick R.. (2023). Segment Anything. 3992-4003. 10.1109/ICCV51070.2023.00371
196. Medium. “Segment Anything” — An Overview. 2023. Available at: <https://medium.com/@ghadi.alhajj/segment-anything-model-an-overview-118905735135>, last accessed 2024/07/13
197. Зарічковий О.А. Дуальна архітектура програмного забезпечення для автоматизації розмітки даних для задач комп’ютерного зору. Міжвідомчий науково-технічний журнал «Адаптивні системи автоматичного управління». 2024. № 45 (2024). С. 109-118. doi: 10.20535/1560-8956.45.2024.313096
198. Docker. “Docker: Accelerated Container Application Development”, 2024. Available at: <https://www.docker.com/>, last accessed 2024/07/13
199. React. “React”, 2024. Available at: <https://react.dev/>, last accessed 2024/07/13
200. Django Project. “Django: The web framework for perfectionists with deadlines”, 2024. Available at: <https://www.djangoproject.com>, last accessed 2024/07/13
201. Traefik. “Traefik Labs: Say Goodbye to Connectivity Chaos”, 2024. Available at: <https://traefik.io>, last accessed 2024/07/13

202. PostgreSQL. “PostgreSQL”: The world's most advanced open source database. 2024. Available at: <https://www.postgresql.org/>, last accessed 2024/07/13
203. Redis. “Redis”. 2024. Available at: <https://redis.io/>, last accessed 2024/07/13
204. Open Policy Agent. “Open Policy Agent”, 2024. Available at: <https://www.openpolicyagent.org>, last accessed 2024/07/13
205. Nuclio. “Nuclio: Serverless Platform for Automated Data Science”, 2024. Available at: <https://nuclio.io>, last accessed 2024/07/13
206. Elasticsearch. “Elasticsearch: The Official Distributed Search & Analytics”, 2024. Available at: <https://www.elastic.co>, last accessed 2024/07/13
207. E. Real, J. Shlens, S. Mazzocchi, et al. YouTube-BoundingBoxes: A Large High-Precision Human-Annotated Data Set for Object Detection in Video. CoRR, vol. abs/1702.00824, <http://arxiv.org/abs/1702.00824> (2017)
208. S. Au-El-Haija, N. Kothari, J. Lee, et al. YouTube-8M: A Large-Scale Video Classification Benchmark. CoRR, vol. abs/1609.08675, <http://arxiv.org/abs/1609.08675> (2016)
209. T.-Y. Lin, P. Dollar, R. B. Girshick, et al. Feature pyramid networks for object detection. In CVPR, pp. 936–944, 2017
210. J. Wang, K. Chen, S. Yang, C. C. Loy, D. Lin. “Region Proposal by Guided Anchoring”. CoRR, vol. abs/1901.03278, <http://arxiv.org/abs/1901.03278> (2019)
211. M. R. Zhang, J. Lucas, G. Hinton, J. Ba. “Lookahead Optimizer: k steps forward, 1 step back”. CoRR, vol. abs/1907.08610, <http://arxiv.org/abs/1907.08610> (2019)

## ДОДАТОК А

### ЧАСТИНА ПРОГРАМНОГО КОДУ РОЗРОБЛЕНОГО ВЕБЗАСТОСУНКУ

HardSampleMining.py

```
import torch
import torch.nn as nn
from torch.utils.data import DataLoader, Dataset
from transformers import VideoSwinModel, VideoSwinConfig
from sklearn.cluster import KMeans
from sklearn.metrics import pairwise_distances_argmin_min
import numpy as np
```

```
class VideoDataset(Dataset):
    def __init__(self, videos, labels):
        self.videos = videos
        self.labels = labels

    def __len__(self):
        return len(self.videos)

    def __getitem__(self, idx):
        return self.videos[idx], self.labels[idx]
```

```
class HardSampleMining:
    def __init__(self, videos, labels, epochs, num_samples, device='cuda'):
```

```
"""
```

Initializes the HardSampleMining class with the dataset and training parameters.

Parameters:

videos (list): List of video data points.

labels (list): List of labels corresponding to the videos.

epochs (int): Number of training epochs.

num\_samples (int): Number of hard samples to select each epoch.

device (str): Device to use for training ('cuda' or 'cpu').

```
"""
```

```
self.videos = np.array(videos)
```

```
self.labels = np.array(labels)
```

```
self.epochs = epochs
```

```
self.num_samples = num_samples
```

```
self.dataset = list(zip(self.videos, self.labels))
```

```
self.device = device
```

```
self.model = self.initialize_model()
```

```
self.criterion = nn.CrossEntropyLoss()
```

```
self.optimizer = torch.optim.Adam(self.model.parameters(), lr=1e-4)
```

```
def initialize_model(self):
```

```
"""
```

Initializes the Video Swin Transformer Large model.

Returns:

nn.Module: The Video Swin Transformer model.

```
"""
```

```

config = VideoSwinConfig.from_pretrained('microsoft/videomae-large')
model = VideoSwinModel(config)
model.to(self.device)
return model

```

```

def train_model(self, train_data):

```

```

    """

```

```

    Trains the model on the given dataset for one epoch.

```

```

    Parameters:

```

```

    train_data (list): List of training data points.

```

```

    """

```

```

        train_loader = DataLoader(VideoDataset(*zip(*train_data)), batch_size=4,
shuffle=True)

```

```

        self.model.train()

```

```

        for videos, labels in train_loader:

```

```

            videos = videos.to(self.device)

```

```

            labels = labels.to(self.device)

```

```

            outputs = self.model(videos).logits

```

```

            loss = self.criterion(outputs, labels)

```

```

            self.optimizer.zero_grad()

```

```

            loss.backward()

```

```

            self.optimizer.step()

```

```

print(f"Training loss: {loss.item()}")

def get_embeddings(self, data):
    """
    Generates embeddings for the given data points.

    Parameters:
    data (list): List of data points to generate embeddings for.

    Returns:
    np.array: Array of embeddings for the data points.
    """
    self.model.eval()
    embeddings = []

    with torch.no_grad():
        for video in data:
            video_tensor = torch.tensor(video).unsqueeze(0).to(self.device)
            embedding = self.model(video_tensor).pooler_output
            embeddings.append(embedding.cpu().numpy().flatten())

    return np.array(embeddings)

def hard_sample_mining(self):
    """
    Performs hard sample mining and trains the model over multiple epochs.
    """

```

```

for epoch in range(self.epochs):
    print(f"Epoch {epoch + 1}/{self.epochs}")
    embeddings = self.get_embeddings(self.videos)
    k = self.determine_k(embeddings)
    kmeans = KMeans(n_clusters=k).fit(embeddings)
    distances = kmeans.transform(embeddings)

    hard_samples = self.select_hard_samples(distances)
    train_data = self.oversample(hard_samples)

    self.train_model(train_data)

def determine_k(self, embeddings):
    """
    Determines the number of clusters K using the Elbow method.

    Parameters:
    embeddings (np.array): Array of embeddings.

    Returns:
    int: The number of clusters.
    """
    distortions = []
    K = range(1, 400)

    for k in K:
        kmeans = KMeans(n_clusters=k).fit(embeddings)

```

```
distortions.append(sum(np.min(pairwise_distances_argmin_min(embeddings,
kmeans.cluster_centers_)[1]) ** 2))
```

```
# Finding the elbow point
```

```
k_optimal = np.diff(distortions, 2).argmax() + 2
```

```
return k_optimal
```

```
def select_hard_samples(self, distances):
```

```
    """
```

```
    Selects hard samples based on distances from cluster centers.
```

```
    Parameters:
```

```
    distances (np.array): Array of distances from cluster centers.
```

```
    Returns:
```

```
    list: List of indices of selected hard samples.
```

```
    """
```

```
    closest_indices = np.argsort(distances, axis=0)[:self.num_samples].flatten()
```

```
    furthest_indices = np.argsort(distances, axis=0)[-self.num_samples:].flatten()
```

```
    return closest_indices.tolist() + furthest_indices.tolist()
```

```
def oversample(self, hard_sample_indices):
```

```
    """
```

```
    Oversamples the dataset by replacing samples with selected hard samples.
```

```
    Parameters:
```

```
    hard_sample_indices (list): List of indices of selected hard samples.
```



Returns:

list: Oversampled training dataset.

"""

hard\_samples = [self.dataset[i] for i in hard\_sample\_indices]

return self.dataset + hard\_samples

FrameSelection.py

import torch

import torch.nn as nn

from torch.utils.data import DataLoader, Dataset

class KeyframeSelectionModel(nn.Module):

def \_\_init\_\_(self):

super(KeyframeSelectionModel, self).\_\_init\_\_()

# Define your model layers here

def forward(self, x):

# Define the forward pass

return x

class VideoDataset(Dataset):

def \_\_init\_\_(self, frames, labels):

self.frames = frames

self.labels = labels

def \_\_len\_\_(self):

```
return len(self.frames)
```

```
def __getitem__(self, idx):
    return self.frames[idx], self.labels[idx]
```

```
class FrameSelectionAndTraining:
```

```
    def __init__(self, model, sequence_length, padding_length, loss_function, alpha,
device='cuda'):
```

```
        """
```

Initializes the FrameSelectionAndTraining class with the keyframe selection model and parameters.

Parameters:

model (nn.Module): Keyframe selection model.

sequence\_length (int): Pre-defined processing sequence length.

padding\_length (int): Refinement padding length.

loss\_function (function): Loss function.

alpha (float): Regularization parameter.

device (str): Device to use for training ('cuda' or 'cpu').

```
        """
```

```
        self.model = model.to(device)
```

```
        self.sequence_length = sequence_length
```

```
        self.padding_length = padding_length
```

```
        self.loss_function = loss_function
```

```
        self.alpha = alpha
```

```
        self.device = device
```

```
        self.optimizer = torch.optim.Adam(self.model.parameters(), lr=1e-4)
```

```

def pad_frames(self, frames):
    """
    Pads frames to the pre-defined sequence length.

    Parameters:
    frames (list): List of video frames.

    Returns:
    list: Padded list of frames.
    """
    if len(frames) < self.sequence_length:
        padding = [torch.zeros_like(frames[0])] * (self.sequence_length - len(frames))
        frames.extend(padding)
    return frames

def subsample_frames(self, frames, num_samples):
    """
    Subsamples frames to the desired number of samples.

    Parameters:
    frames (list): List of video frames.
    num_samples (int): Number of samples to subsample.

    Returns:
    list: Subsampled list of frames.
    """

```

```
indices = np.linspace(0, len(frames) - 1, num_samples).astype(int)
return [frames[i] for i in indices]
```

```
def get_importance_scores(self, frames):
```

```
    """
```

Gets importance scores from the model for the given frames.

Parameters:

frames (list): List of video frames.

Returns:

torch.Tensor: Importance scores.

```
    """
```

```
    self.model.eval()
```

```
    with torch.no_grad():
```

```
        frames_tensor = torch.stack(frames).to(self.device)
```

```
        scores = self.model(frames_tensor)
```

```
    return scores
```

```
def border_refinement(self, frames, scores):
```

```
    """
```

Performs border refinement on the frames based on importance scores.

Parameters:

frames (list): List of video frames.

scores (torch.Tensor): Importance scores.

Returns:

list: Refined list of frames.

"""

```
refined_frames = []
```

```
scores = scores.cpu().numpy()
```

```
indices = np.argsort(scores)[::-1]
```

```
while len(refined_frames) < self.sequence_length:
```

```
    for idx in indices:
```

```
        if idx not in refined_frames:
```

```
            refined_frames.append(frames[idx])
```

```
            if len(refined_frames) >= self.sequence_length:
```

```
                break
```

```
            for p in range(idx - self.padding_length, idx + self.padding_length + 1):
```

```
                if 0 <= p < len(frames) and p not in refined_frames:
```

```
                    refined_frames.append(frames[p])
```

```
                    if len(refined_frames) >= self.sequence_length:
```

```
                        break
```

```
return refined_frames[:self.sequence_length]
```

```
def train(self, frames, labels):
```

"""

Trains the model on the given frames and labels.

Parameters:

frames (list): List of video frames.

labels (list): List of labels corresponding to the frames.

```

"""
    padded_frames = self.pad_frames(frames)
    subsampled_frames = self.subsample_frames(padded_frames,
self.sequence_length)
    subsampled_scores = self.get_importance_scores(subsampled_frames)

    refined_frames = self.border_refinement(padded_frames, subsampled_scores)
    refined_scores = self.get_importance_scores(refined_frames)

    labels_tensor = torch.tensor(labels).to(self.device)

    loss = self.loss_function(subsampled_scores, labels_tensor) + \
        self.loss_function(refined_scores, labels_tensor) - \
        self.alpha * ((refined_scores - subsampled_scores) ** 2).mean()

    self.optimizer.zero_grad()
    loss.backward()
    self.optimizer.step()

    print(f"Training loss: {loss.item()}")

```

EnrichLexicon.py

```
import openai
```

```
import torch
```

```
from transformers import AutoModel, AutoTokenizer
```

```
import numpy as np
```

```
from scipy.spatial.distance import cosine
```

```

# Set your OpenAI API key
openai.api_key = 'YOUR_OPENAI_API_KEY'

class TextualEmbeddingModel:
    def __init__(self, model_name='intfloat/multilingual-e5-large'):
        """
        Initializes the TextualEmbeddingModel with a pre-trained model.

        Parameters:
        model_name (str): Name of the pre-trained model.
        """
        self.tokenizer = AutoTokenizer.from_pretrained(model_name)
        self.model = AutoModel.from_pretrained(model_name)
        self.model.eval()

    def get_embedding(self, text):
        """
        Generates an embedding for the given text.

        Parameters:
        text (str): The text to generate an embedding for.

        Returns:
        np.array: The generated embedding.
        """
        inputs = self.tokenizer(text, return_tensors='pt', truncation=True, padding=True)

```

```

    with torch.no_grad():
        outputs = self.model(**inputs)
        return outputs.last_hidden_state.mean(dim=1).squeeze().numpy()

def chatgpt_prompt(prompt):
    """
    Sends a prompt to the ChatGPT API and gets the response.

    Parameters:
    prompt (str): The prompt to send to ChatGPT.

    Returns:
    list: A list of generated labels.
    """
    response = openai.Completion.create(
        model="text-davinci-003",
        prompt=prompt,
        max_tokens=150,
        temperature=0.7,
        n=1,
        stop=None
    )
    result = response.choices[0].text.strip()
    return [label.strip() for label in result.split(',')]

def deduplicate_labels(enriched_labels, model, threshold):
    """

```



Deduplicates labels based on semantic meaning using cosine similarity.

Parameters:

enriched\_labels (list): List of enriched labels.

model (TextualEmbeddingModel): Textual embedding model.

threshold (float): Deduplication threshold.

Returns:

list: List of deduplicated labels.

"""

```
embeddings = [model.get_embedding(label) for label in enriched_labels]
```

```
unique_labels = []
```

```
unique_embeddings = []
```

```
for i, emb in enumerate(embeddings):
```

```
    is_unique = True
```

```
    for u_emb in unique_embeddings:
```

```
        if cosine(emb, u_emb) < threshold:
```

```
            is_unique = False
```

```
            break
```

```
    if is_unique:
```

```
        unique_labels.append(enriched_labels[i])
```

```
        unique_embeddings.append(emb)
```

```
return unique_labels
```

```
def main(dataset, labels, model, threshold):
```

```
"""
```

Main function to generate enriched labels and deduplicate them.

Parameters:

dataset (str): Name of the dataset.

labels (list): List of labels.

model (TextualEmbeddingModel): Textual embedding model.

threshold (float): Deduplication threshold.

Returns:

list: List of deduplicated labels.

```
"""
```

```
enriched_labels = set()
```

```
for label in labels:
```

```
    prompt = f'You are a domain expert in the {dataset} dataset, helping develop a
labeling system. Generate additional labels that will enrich {label} label of {dataset} dataset.
Format output as comma separated labels.'
```

```
    result = chatgpt_prompt(prompt)
```

```
    enriched_labels.update(result)
```

```
deduplicated_labels = deduplicate_labels(list(enriched_labels), model, threshold)
```

```
return deduplicated_labels
```

DualArch.py

```
import torch
```

```
import numpy as np
```

```
from segment_anything import SamPredictor
```

```
from internvideo import InternVideoPredictor
```

```
class ActiveLearningModel:
```

```
    def __init__(self):
```

```
        """
```

```
        Initializes the ActiveLearningModel using the InternVideo model.
```

```
        """
```

```
        self.model = InternVideoPredictor() # Initialize the InternVideo model
```

```
    def predict(self, batch):
```

```
        """
```

```
        Predicts using the active learning model.
```

```
        Parameters:
```

```
        batch (list): The batch of data.
```

```
        Returns:
```

```
        list: The predictions.
```

```
        """
```

```
        # Placeholder for actual prediction logic
```

```
        return self.model.predict(batch)
```

```
class ZeroShotModel:
```

```
    def __init__(self):
```

```
        """
```

```
        Initializes the ZeroShotModel using the SAM model.
```

```
        """
```

```

self.model = SamPredictor() # Initialize the SAM model

def predict(self, batch):
    """
    Predicts using the zero-shot learning model.

    Parameters:
    batch (list): The batch of data.

    Returns:
    list: The predictions.
    """
    # Placeholder for actual prediction logic
    return self.model.predict(batch)

def process_dataset(dataset, labels, active_model, zero_shot_model, map_function):
    """
    Processes the dataset using active learning and zero-shot learning models.

    Parameters:
    dataset (list): The dataset.
    labels (list): The dataset labels.
    active_model (ActiveLearningModel): The active learning model.
    zero_shot_model (ZeroShotModel): The zero-shot learning model.
    map_function (function): The quality function (mAP).

    Yields:

```

list: The predictions.

```
"""
```

```
a, b, control = 1, 1, 0
```

```
mode = "Zero"
```

```
for batch in dataset:
```

```
    results = active_model.predict(batch)
```

```
    if control == 0:
```

```
        results_zero = zero_shot_model.predict(batch)
```

```
        if mode == "Active":
```

```
            yield results
```

```
        else:
```

```
            yield results_zero
```

```
        anno = zero_shot_model.predict(batch)
```

```
        if map_function(results, anno) > map_function(results_zero, anno):
```

```
            control = b
```

```
            a, b = b, a + b
```

```
            mode = "Active"
```

```
        else:
```

```
            a, b, control = 1, 1, 0
```

```
            mode = "Zero"
```

```
    else:
```

```
        control -= 1
```

```
        yield results
```

Dockerfile

```
ARG PIP_VERSION=22.0.2
```

```
ARG BASE_IMAGE=ubuntu:22.04
```

```
FROM ${BASE_IMAGE} as build-image-base
```

```
RUN apt-get update && \
```

```
    DEBIAN_FRONTEND=noninteractive apt-get --no-install-recommends install -yq \
```

```
        curl \
```

```
        g++ \
```

```
        gcc \
```

```
        git \
```

```
        libgeos-dev \
```

```
        libldap2-dev \
```

```
        libsasl2-dev \
```

```
        make \
```

```
        nasm \
```

```
        pkg-config \
```

```
        python3-dev \
```

```
        python3-pip \
```

```
&& rm -rf /var/lib/apt/lists/*
```

```
ARG PIP_VERSION
```

```
ENV PIP_DISABLE_PIP_VERSION_CHECK=1
```

```
RUN --mount=type=cache,target=/root/.cache/pip/http \
```

```
    python3 -m pip install -U pip==${PIP_VERSION}
```

```
# We build OpenH264, FFmpeg and PyAV in a separate build stage,
```

```
# because this way Docker can do it in parallel to all the other packages.
```

FROM build-image-base AS build-image-av

# Compile Openh264 and FFmpeg

ARG PREFIX=/opt/ffmpeg

ARG PKG\_CONFIG\_PATH=\${PREFIX}/lib/pkgconfig

ENV FFMPEG\_VERSION=4.3.1 \

OPENH264\_VERSION=2.1.1

WORKDIR /tmp/openh264

RUN curl -sL

[https://github.com/cisco/openh264/archive/v\\${OPENH264\\_VERSION}.tar.gz](https://github.com/cisco/openh264/archive/v${OPENH264_VERSION}.tar.gz) --output - | \

tar -zx --strip-components=1 && \

make -j5 && make install-shared PREFIX=\${PREFIX} && make clean

WORKDIR /tmp/ffmpeg

RUN curl -sL [https://ffmpeg.org/releases/ffmpeg-\\${FFMPEG\\_VERSION}.tar.gz](https://ffmpeg.org/releases/ffmpeg-${FFMPEG_VERSION}.tar.gz) --

output - | \

tar -zx --strip-components=1 && \

./configure --disable-nonfree --disable-gpl --enable-libopenh264 \

--enable-shared --disable-static --disable-doc --disable-programs --

prefix="\${PREFIX}" && \

make -j5 && make install && make clean

COPY

utils/dataset\_manifest/requirements.txt

/tmp/utils/dataset\_manifest/requirements.txt

```
# Since we're using pip-compile-multi, each dependency can only be listed in
# one requirements file. In the case of PyAV, that should be
# `dataset_manifest/requirements.txt`. Make sure it's actually there,
# and then remove everything else.
```

```
RUN grep -q '^av==' /tmp/utils/dataset_manifest/requirements.txt
```

```
RUN sed -i '/^av==/!d' /tmp/utils/dataset_manifest/requirements.txt
```

```
# Work around https://github.com/PyAV-Org/PyAV/issues/1140
```

```
RUN pip install setuptools wheel 'cython<3'
```

```
RUN --mount=type=cache,target=/root/.cache/pip/http \
    python3 -m pip wheel --no-binary=av --no-build-isolation \
    -r /tmp/utils/dataset_manifest/requirements.txt \
    -w /tmp/wheelhouse
```

```
# This stage builds wheels for all dependencies (except PyAV)
```

```
FROM build-image-base AS build-image
```

```
COPY cvat/requirements/ /tmp/cvat/requirements/
```

```
COPY                                utils/dataset_manifest/requirements.txt
```

```
/tmp/utils/dataset_manifest/requirements.txt
```

```
# Exclude av from the requirements file
```

```
RUN sed -i '/^av==/d' /tmp/utils/dataset_manifest/requirements.txt
```

```
ARG CVAT_CONFIGURATION="production"
```



```

RUN --mount=type=cache,target=/root/.cache/pip/http \
  DATUMARO_HEADLESS=1 python3 -m pip wheel --no-deps \
  -r /tmp/cvat/requirements/${CVAT_CONFIGURATION}.txt \
  -w /tmp/wheelhouse

```

```

FROM golang:1.20.5 AS build-smokescreen

```

```

RUN git clone --depth=1 -b v0.0.4 https://github.com/stripe/smokescreen.git
RUN cd smokescreen && go build -o /tmp/smokescreen

```

```

FROM ${BASE_IMAGE}

```

```

ARG http_proxy
ARG https_proxy
ARG no_proxy="nuclio,${no_proxy}"
ARG socks_proxy
ARG TZ="Etc/UTC"

```

```

ENV TERM=xterm \
  http_proxy=${http_proxy} \
  https_proxy=${https_proxy} \
  no_proxy=${no_proxy} \
  socks_proxy=${socks_proxy} \
  LANG='C.UTF-8' \
  LC_ALL='C.UTF-8' \
  TZ=${TZ}

```

```

ARG USER="django"
ARG CVAT_CONFIGURATION="production"
ENV
DJANGO_SETTINGS_MODULE="cvat.settings.${CVAT_CONFIGURATION}"

# Install necessary apt packages
RUN apt-get update && \
    DEBIAN_FRONTEND=noninteractive apt-get --no-install-recommends install -yq \
        bzip2 \
        ca-certificates \
        curl \
        git \
        libgeos-c1v5 \
        libgl1 \
        libgomp1 \
        libldap-2.5-0 \
        libpython3.10 \
        libsasl2-2 \
        nginx \
        p7zip-full \
        poppler-utils \
        python3 \
        python3-distutils \
        python3-venv \
        supervisor \
        tzdata \
    && ln -fs /usr/share/zoneinfo/${TZ} /etc/localtime && \

```

```
dpkg-reconfigure -f noninteractive tzdata && \
rm -rf /var/lib/apt/lists/* && \
echo 'application/wasm wasm' >> /etc/mime.types
```

```
# Install smokescreen
```

```
COPY --from=build-smokescreen /tmp/smokescreen /usr/local/bin/smokescreen
```

```
# Add a non-root user
```

```
ENV USER=${USER}
```

```
ENV HOME /home/${USER}
```

```
RUN adduser --shell /bin/bash --disabled-password --gecos "" ${USER}
```

```
ARG CLAM_AV="no"
```

```
RUN if [ "$CLAM_AV" = "yes" ]; then \
```

```
    apt-get update && \
```

```
    apt-get --no-install-recommends install -yq \
```

```
        clamav \
```

```
        libclamunrar9 && \
```

```
    sed -i 's/ReceiveTimeout 30/ReceiveTimeout 300/g' /etc/clamav/freshclam.conf
```

```
&& \
```

```
    freshclam && \
```

```
    chown -R ${USER}:${USER} /var/lib/clamav && \
```

```
    rm -rf /var/lib/apt/lists/*; \
```

```
fi
```

```
# Install wheels from the build image
```

```
RUN python3 -m venv /opt/venv
```

```

ENV PATH="/opt/venv/bin:${PATH}"
ARG PIP_VERSION
ARG PIP_DISABLE_PIP_VERSION_CHECK=1

RUN python -m pip install -U pip==${PIP_VERSION}
RUN                                     --mount=type=bind,from=build-
image,source=/tmp/wheelhouse,target=/mnt/wheelhouse \
    --mount=type=bind,from=build-image-
av,source=/tmp/wheelhouse,target=/mnt/wheelhouse-av \
    python -m pip install --no-index /mnt/wheelhouse/*.whl /mnt/wheelhouse-av/*.whl

ENV NUMPROCS=1
COPY --from=build-image-av /opt/ffmpeg/lib /usr/lib

# These variables are required for supervisord substitutions in files
# This library allows remote python debugging with VS Code
ARG CVAT_DEBUG_ENABLED
RUN if [ "${CVAT_DEBUG_ENABLED}" = 'yes' ]; then \
    python3 -m pip install --no-cache-dir debugpy; \
fi

# Install and initialize CVAT, copy all necessary files
COPY cvat/nginx.conf /etc/nginx/nginx.conf
COPY --chown=${USER} components /tmp/components
COPY --chown=${USER} supervisord/ ${HOME}/supervisord
COPY    --chown=${USER}    wait-for-it.sh    manage.py    backend_entrypoint.sh
wait_for_deps.sh ${HOME}/

```

```

COPY --chown=${USER} utils/ ${HOME}/utils
COPY --chown=${USER} cvat/ ${HOME}/cvat
COPY --chown=${USER} rqscheduler.py ${HOME}

```

```

ARG COVERAGE_PROCESS_START

```

```

RUN if [ "${COVERAGE_PROCESS_START}" ]; then \
    echo "import coverage; coverage.process_startup()" > \
/opt/venv/lib/python3.10/site-packages/coverage_subprocess.pth; \
fi

```

```

# RUN all commands below as 'django' user

```

```

USER ${USER}
WORKDIR ${HOME}

```

```

RUN mkdir -p data share keys logs /tmp/supervisord static

```

```

EXPOSE 8080

```

```

ENTRYPOINT ["/backend_entrypoint.sh"]

```

```

Dockerfile.mmaction2

```

```

ARG PYTORCH="1.8.1"

```

```

ARG CUDA="10.2"

```

```

ARG CUDNN="7"

```

```

FROM pytorch/pytorch:${PYTORCH}-cuda${CUDA}-cudnn${CUDNN}-devel

```

```

ENV TORCH_CUDA_ARCH_LIST="6.0 6.1 7.0+PTX"

```

```

ENV TORCH_NVCC_FLAGS="-Xfatbin -compress-all"

```

```
ENV CMAKE_PREFIX_PATH="$(dirname $(which conda))/../"
```

```
# fetch the key refer to https://forums.developer.nvidia.com/t/18-04-cuda-docker-image-is-broken/212892/9
```

```
RUN apt-key adv --fetch-keys
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/3bf863cc.pub
b 32
```

```
RUN apt-key adv --fetch-keys
https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86_64/7fa2af80.pub
```

```
RUN apt-get update && apt-get install -y git ninja-build libglib2.0-0 libsm6 libxrender-dev libxext6 ffmpeg \
```

```
&& apt-get clean \
```

```
&& rm -rf /var/lib/apt/lists/*
```

```
# Install MMCV
```

```
RUN pip install openmim
```

```
RUN mim install mmengine mmcv
```

```
# Install MMAction2
```

```
RUN conda clean --all
```

```
RUN git clone https://github.com/open-mmlab/mmdetection2.git /mmdetection2
```

```
WORKDIR /mmdetection2
```

```
RUN mkdir -p /mmdetection2/data
```

```
ENV FORCE_CUDA="1"
```

```
RUN git checkout main
```

```
RUN pip install cython --no-cache-dir
```

```
RUN pip install --no-cache-dir -e .
```

```
dist_test.sh
```

```
#!/usr/bin/env bash
```

```
set -x
```

```
CONFIG=$1
```

```
CHECKPOINT=$2
```

```
GPUS=$3
```

```
NNODES=${NNODES:-1}
```

```
NODE_RANK=${NODE_RANK:-0}
```

```
PORT=${PORT:-29500}
```

```
MASTER_ADDR=${MASTER_ADDR:-"127.0.0.1"}
```

```
PYTHONPATH="$(dirname $0)/..":$PYTHONPATH \
```

```
# Arguments starting from the forth one are captured by ${@:4}
```

```
python -m torch.distributed.launch --nnodes=$NNODES --node_rank=$NODE_RANK
--master_addr=$MASTER_ADDR \
    --nproc_per_node=$GPUS --master_port=$PORT $(dirname "$0")/test.py $CONFIG
$CHECKPOINT --launcher pytorch ${@:4}
```

```
dist_train.sh
```

```
#!/usr/bin/env bash
```

```
set -x
```

```
CONFIG=$1
```

```
GPUS=$2
```

```
NNODES=${NNODES:-1}
```

```
NODE_RANK=${NODE_RANK:-0}
```

```
PORT=${PORT:-29500}
```

```
MASTER_ADDR=${MASTER_ADDR:-"127.0.0.1"}
```

```
PYTHONPATH="$(dirname $0)/..":$PYTHONPATH \
```

```
python -m torch.distributed.launch --nnodes=$NNODES --node_rank=$NODE_RANK
--master_addr=$MASTER_ADDR \
    --nproc_per_node=$GPUS --master_port=$PORT "$(dirname "$0")/train.py
$CONFIG --launcher pytorch ${@:3}
```

```
# Any arguments from the third one are captured by ${@:3}
```

```
test.py
```

```
# Copyright (c) OpenMMLab. All rights reserved.
```

```
import argparse
```

```
import os
```

```
import os.path as osp
```

```
from mmengine.config import Config, DictAction
```

```
from mmengine.runner import Runner
```

```
from mmaction.registry import RUNNERS
```

```
def parse_args():
```



```

parser = argparse.ArgumentParser(
    description='MMAAction2 test (and eval) a model')
parser.add_argument('config', help='test config file path')
parser.add_argument('checkpoint', help='checkpoint file')
parser.add_argument(
    '--work-dir',
    help='the directory to save the file containing evaluation metrics')
parser.add_argument(
    '--dump',
    type=str,
    help='dump predictions to a pickle file for offline evaluation')
parser.add_argument(
    '--cfg-options',
    nargs='+',
    action=DictAction,
    help='override some settings in the used config, the key-value pair '
    'in xxx=yyy format will be merged into config file. If the value to '
    'be overwritten is a list, it should be like key="[a,b]" or key=a,b '
    'It also allows nested list/tuple values, e.g. key="[(a,b),(c,d)]" '
    'Note that the quotation marks are necessary and that no white space '
    'is allowed.')
parser.add_argument(
    '--show-dir',
    help='directory where the visualization images will be saved.')
parser.add_argument(
    '--show',
    action='store_true',

```

```

        help='whether to display the prediction results in a window.')
    parser.add_argument(
        '--interval',
        type=int,
        default=1,
        help='visualize per interval samples.')
    parser.add_argument(
        '--wait-time',
        type=float,
        default=2,
        help='display time of every window. (second)')
    parser.add_argument(
        '--launcher',
        choices=['none', 'pytorch', 'slurm', 'mpi'],
        default='none',
        help='job launcher')
    parser.add_argument('--local_rank', '--local-rank', type=int, default=0)
    args = parser.parse_args()
    if 'LOCAL_RANK' not in os.environ:
        os.environ['LOCAL_RANK'] = str(args.local_rank)
    return args

def merge_args(cfg, args):
    """Merge CLI arguments to config."""
    # ----- visualization -----
    if args.show or (args.show_dir is not None):

```

```

assert 'visualization' in cfg.default_hooks, \
    'VisualizationHook is not set in the `default_hooks` field of '\
    'config. Please set `visualization=dict(type="VisualizationHook")`'

cfg.default_hooks.visualization.enable = True
cfg.default_hooks.visualization.show = args.show
cfg.default_hooks.visualization.wait_time = args.wait_time
cfg.default_hooks.visualization.out_dir = args.show_dir
cfg.default_hooks.visualization.interval = args.interval

# ----- Dump predictions -----
if args.dump is not None:
    assert args.dump.endswith(('pkl', 'pickle')), \
        'The dump file must be a pkl file.'
    dump_metric = dict(type='DumpResults', out_file_path=args.dump)
    if isinstance(cfg.test_evaluator, (list, tuple)):
        cfg.test_evaluator = list(cfg.test_evaluator)
        cfg.test_evaluator.append(dump_metric)
    else:
        cfg.test_evaluator = [cfg.test_evaluator, dump_metric]

return cfg

def main():
    args = parse_args()

```

```

# load config
cfg = Config.fromfile(args.config)
cfg = merge_args(cfg, args)
cfg.launcher = args.launcher
if args.cfg_options is not None:
    cfg.merge_from_dict(args.cfg_options)

# work_dir is determined in this priority: CLI > segment in file > filename
if args.work_dir is not None:
    # update configs according to CLI args if args.work_dir is not None
    cfg.work_dir = args.work_dir
elif cfg.get('work_dir', None) is None:
    # use config filename as default work_dir if cfg.work_dir is None
    cfg.work_dir = osp.join('./work_dirs',
                             osp.splitext(osp.basename(args.config))[0])

cfg.load_from = args.checkpoint

# build the runner from config
if 'runner_type' not in cfg:
    # build the default runner
    runner = Runner.from_cfg(cfg)
else:
    # build customized runner from the registry
    # if 'runner_type' is set in the cfg
    runner = RUNNERS.build(cfg)

```

```

# start testing
runner.test()

if __name__ == '__main__':
    main()

train.py
# Copyright (c) OpenMMLab. All rights reserved.
import argparse
import os
import os.path as osp

from mmengine.config import Config, DictAction
from mmengine.runner import Runner

from mmaction.registry import RUNNERS

def parse_args():
    parser = argparse.ArgumentParser(description='Train a action recognizer')
    parser.add_argument('config', help='train config file path')
    parser.add_argument('--work-dir', help='the dir to save logs and models')
    parser.add_argument(
        '--resume',
        nargs='?',
        type=str,

```

```

const='auto',
help='If specify checkpoint path, resume from it, while if not '
'specify, try to auto resume from the latest checkpoint '
'in the work directory.')
parser.add_argument(
    '--amp',
    action='store_true',
    help='enable automatic-mixed-precision training')
parser.add_argument(
    '--no-validate',
    action='store_true',
    help='whether not to evaluate the checkpoint during training')
parser.add_argument(
    '--auto-scale-lr',
    action='store_true',
    help='whether to auto scale the learning rate according to the '
    'actual batch size and the original batch size.')
parser.add_argument('--seed', type=int, default=None, help='random seed')
parser.add_argument(
    '--diff-rank-seed',
    action='store_true',
    help='whether or not set different seeds for different ranks')
parser.add_argument(
    '--deterministic',
    action='store_true',
    help='whether to set deterministic options for CUDNN backend.')
parser.add_argument(

```

```

'--cfg-options',
nargs='+',
action=DictAction,
help='override some settings in the used config, the key-value pair '
'in xxx=yyy format will be merged into config file. If the value to '
'be overwritten is a list, it should be like key="[a,b]" or key=a,b '
'It also allows nested list/tuple values, e.g. key="[(a,b),(c,d)]" '
'Note that the quotation marks are necessary and that no white space '
'is allowed.')
```

```

parser.add_argument(
    '--launcher',
    choices=['none', 'pytorch', 'slurm', 'mpi'],
    default='none',
    help='job launcher')
parser.add_argument('--local_rank', '--local-rank', type=int, default=0)
args = parser.parse_args()
if 'LOCAL_RANK' not in os.environ:
    os.environ['LOCAL_RANK'] = str(args.local_rank)

return args

```

```

def merge_args(cfg, args):
    """Merge CLI arguments to config."""
    if args.no_validate:
        cfg.val_cfg = None
        cfg.val_dataloader = None

```

```

    cfg.val_evaluator = None

    cfg.launcher = args.launcher

    # work_dir is determined in this priority: CLI > segment in file > filename
    if args.work_dir is not None:
        # update configs according to CLI args if args.work_dir is not None
        cfg.work_dir = args.work_dir
    elif cfg.get('work_dir', None) is None:
        # use config filename as default work_dir if cfg.work_dir is None
        cfg.work_dir = osp.join('./work_dirs',
                                osp.splitext(osp.basename(args.config))[0])

    # enable automatic-mixed-precision training
    if args.amp is True:
        optim_wrapper = cfg.optim_wrapper.get('type', 'OptimWrapper')
        assert optim_wrapper in ['OptimWrapper', 'AmpOptimWrapper'], \
            f'`--amp` is not supported custom optimizer wrapper type '\
            f'`{optim_wrapper}`.'
        cfg.optim_wrapper.type = 'AmpOptimWrapper'
        cfg.optim_wrapper.setdefault('loss_scale', 'dynamic')

    # resume training
    if args.resume == 'auto':
        cfg.resume = True
        cfg.load_from = None
    elif args.resume is not None:

```



```

    cfg.resume = True
    cfg.load_from = args.resume

# enable auto scale learning rate
if args.auto_scale_lr:
    cfg.auto_scale_lr.enable = True

# set random seeds
if cfg.get('randomness', None) is None:
    cfg.randomness = dict(
        seed=args.seed,
        diff_rank_seed=args.diff_rank_seed,
        deterministic=args.deterministic)

if args.cfg_options is not None:
    cfg.merge_from_dict(args.cfg_options)

return cfg

def main():
    args = parse_args()

    cfg = Config.fromfile(args.config)

# merge cli arguments to config
    cfg = merge_args(cfg, args)

```

```

# build the runner from config
if 'runner_type' not in cfg:
    # build the default runner
    runner = Runner.from_cfg(cfg)
else:
    # build customized runner from the registry
    # if 'runner_type' is set in the cfg
    runner = RUNNERS.build(cfg)

# start training
runner.train()

if __name__ == '__main__':
    main()

webpack.config.cjs
// Copyright (C) 2020-2022 Intel Corporation
// Copyright (C) 2023 CVAT.ai Corporation
//
// SPDX-License-Identifier: MIT

const path = require('path');
const CopyPlugin = require('copy-webpack-plugin');

const webConfig = {

```

```

target: 'web',
mode: 'production',
devtool: 'source-map',
entry: {
  'cvat-core': './src/api.ts',
},
output: {
  path: path.resolve(__dirname, 'dist'),
  filename: '[name].[contenthash].min.js',
  library: 'cvatCore',
  libraryTarget: 'window',
},
resolve: {
  extensions: ['.ts', '.js'],
  fallback: {
    url: false,
  },
},
module: {
  rules: [
    {
      test: /\.ts?$/,
      exclude: /node_modules/,
      use: {
        loader: 'babel-loader',
        options: {
          plugins: [

```

```

        '@babel/plugin-proposal-class-properties',
        '@babel/plugin-proposal-optional-chaining',
    ],
    presets: ['@babel/preset-env', '@babel/typescript'],
    sourceType: 'unambiguous',
  },
},
},
],
},
plugins: [
  new CopyPlugin({
    patterns: [
      {
        from: '../cvat-data/src/ts/3rdparty/avc.wasm',
        to: 'assets/3rdparty/',
      },
    ],
  }),
],
};

```

```
module.exports = webConfig;
```

```
package.json
```

```

{
  "name": "cvat-core",

```

```

"version": "14.0.0",
"type": "module",
"description": "Part of Computer Vision Tool which presents an interface for client-
side integration",
"main": "src/api.ts",
"scripts": {
  "build": "webpack",
  "test": "jest --config=jest.config.cjs --coverage",
  "type-check": "tsc --noEmit",
  "type-check:watch": "yarn run type-check --watch"
},
"author": "CVAT.ai",
"license": "MIT",
"browserslist": [
  "Chrome >= 63",
  "Firefox > 102",
  "not IE 11",
  "> 2%"
],
"devDependencies": {
  "@babel/preset-typescript": "^7.23.3",
  "babel-jest": "^29.7.0",
  "babel-plugin-transform-import-meta": "^2.2.1",
  "jest": "^29.5.0",
  "jest-config": "^29.5.0",
  "jest-environment-jsdom": "^29.5.0",
  "jest-junit": "^6.4.0"
}

```

```

    },
    "dependencies": {
      "@types/lodash": "^4.14.191",
      "axios": "^1.6.0",
      "axios-retry": "^4.0.0",
      "cvat-data": "link:../cvat-data",
      "detect-browser": "^5.2.1",
      "error-stack-parser": "^2.0.2",
      "form-data": "^4.0.0",
      "js-cookie": "^3.0.1",
      "json-logic-js": "^2.0.1",
      "lodash": "^4.17.21",
      "platform": "^1.3.5",
      "quickhull": "^1.0.3",
      "store": "^2.0.12",
      "tus-js-client": "^3.0.1"
    }
  }
}

```

babel.config.json

```

{
  "presets": [
    [
      "@babel/preset-env",
      {
        "targets": {
          "node": "current"
        }
      }
    ]
  ]
}

```

```

    }
  }
],
"@babel/preset-typescript"
],
"plugins": [
  "babel-plugin-transform-import-meta"
]
}

```

test.config.json

```

// Copyright (C) 2019-2022 Intel Corporation
// Copyright (C) 2023 CVAT.ai Corporation
//
// SPDX-License-Identifier: MIT

```

```
const { defaults } = require('jest-config');
```

```

module.exports = {
  testEnvironment: 'jsdom',
  coverageDirectory: 'reports/coverage',
  coverageReporters: ['json', ['lcov', { projectRoot: '..' }]],
  moduleFileExtensions: [...defaults.moduleFileExtensions, 'ts', 'tsx'],
  reporters: ['default', ['jest-junit', { outputDirectory: 'reports/junit' }]],
  testMatch: ['**/tests/**/*.cjs'],
  testPathIgnorePatterns: ['/node_modules/', '/tests/mocks/*'],
  automock: false,

```

```
};
```

```
jest.config.json
```

```
// Copyright (C) 2019-2022 Intel Corporation
```

```
// Copyright (C) 2023 CVAT.ai Corporation
```

```
//
```

```
// SPDX-License-Identifier: MIT
```

```
const { defaults } = require('jest-config');
```

```
module.exports = {
```

```
  testEnvironment: 'jsdom',
```

```
  coverageDirectory: 'reports/coverage',
```

```
  coverageReporters: ['json', ['lcov', { projectRoot: '..' }]],
```

```
  moduleFileExtensions: [...defaults.moduleFileExtensions, 'ts', 'tsx'],
```

```
  reporters: ['default', ['jest-junit', { outputDirectory: 'reports/junit' }]],
```

```
  testMatch: ['**/tests/**/*.cjs'],
```

```
  testPathIgnorePatterns: ['/node_modules/', '/tests/mocks/*'],
```

```
  automock: false,
```

```
};
```

```
canvas.ts
```

```
// Copyright (C) 2019-2022 Intel Corporation
```

```
// Copyright (C) 2022-2023 CVAT.ai Corporation
```

```
//
```

```
// SPDX-License-Identifier: MIT
```



```

import {
  DrawData, MergeData, SplitData, GroupData,
  JoinData, SliceData, MasksEditData,
  InteractionData as _InteractionData,
  InteractionResult as _InteractionResult,
  CanvasModel, CanvasModelImpl, RectDrawingMethod,
  CuboidDrawingMethod, Configuration, Geometry, Mode,
  HighlightSeverity as _HighlightSeverity, CanvasHint as _CanvasHint,
} from './canvasModel';
import { Master } from './master';
import { CanvasController, CanvasControllerImpl } from './canvasController';
import { CanvasView, CanvasViewImpl } from './canvasView';

import '../scss/canvas.scss';
import pjson from '../package.json';

const CanvasVersion = pjson.version;

interface Canvas {
  html(): HTMLDivElement;
  setup(frameData: any, objectStates: any[], zLayer?: number): void;
  setupIssueRegions(issueRegions: Record<number, { hidden: boolean; points:
number[] }>): void;
  setupConflictRegions(clientID: number): number[];
  activate(clientID: number | null, attributeID?: number): void;
  highlight(clientIDs: number[] | null, severity: HighlightSeverity | null): void;
  rotate(rotationAngle: number): void;

```

```
focus(clientID: number, padding?: number): void;  
fit(): void;  
grid(stepX: number, stepY: number): void;
```

```
interact(interactionData: InteractionData): void;  
draw(drawData: DrawData): void;  
edit(editData: MasksEditData): void;  
group(groupData: GroupData): void;  
join(joinData: JoinData): void;  
slice(sliceData: SliceData): void;  
split(splitData: SplitData): void;  
merge(mergeData: MergeData): void;  
select(objectState: any): void;
```

```
fitCanvas(): void;  
bitmap(enable: boolean): void;  
selectRegion(enable: boolean): void;  
dragCanvas(enable: boolean): void;  
zoomCanvas(enable: boolean): void;
```

```
mode(): Mode;  
cancel(): void;  
configure(configuration: Configuration): void;  
isAbleToChangeFrame(): boolean;  
destroy(): void;
```

```
readonly geometry: Geometry;
```

```
}
```

```
class CanvasImpl implements Canvas {
```

```
    private model: CanvasModel & Master;
```

```
    private controller: CanvasController;
```

```
    private view: CanvasView;
```

```
    public constructor() {
```

```
        this.model = new CanvasModelImpl();
```

```
        this.controller = new CanvasControllerImpl(this.model);
```

```
        this.view = new CanvasViewImpl(this.model, this.controller);
```

```
    }
```

```
    public html(): HTMLDivElement {
```

```
        return this.view.html();
```

```
    }
```

```
    public setup(frameData: any, objectStates: any[], zLayer = 0): void {
```

```
        this.model.setup(frameData, objectStates, zLayer);
```

```
    }
```

```
    public setupIssueRegions(issueRegions: Record<number, { hidden: boolean; points:
number[] }>): void {
```

```
        this.model.setupIssueRegions(issueRegions);
```

```
    }
```

```
    public setupConflictRegions(clientID: number): number[] {
```

```
    return this.view.setupConflictRegions(clientID);
}

public fitCanvas(): void {
    this.model.fitCanvas(this.view.html().clientWidth, this.view.html().clientHeight);
}

public bitmap(enable: boolean): void {
    this.model.bitmap(enable);
}

public selectRegion(enable: boolean): void {
    this.model.selectRegion(enable);
}

public dragCanvas(enable: boolean): void {
    this.model.dragCanvas(enable);
}

public zoomCanvas(enable: boolean): void {
    this.model.zoomCanvas(enable);
}

public activate(clientID: number | null, attributeID: number | null = null): void {
    this.model.activate(clientID, attributeID);
}
```

```
public highlight(clientIDs: number[] | null, severity: HighlightSeverity | null = null):  
void {  
    this.model.highlight(clientIDs, severity);  
}  
  
public rotate(rotationAngle: number): void {  
    this.model.rotate(rotationAngle);  
}  
  
public focus(clientID: number, padding = 0): void {  
    this.model.focus(clientID, padding);  
}  
  
public fit(): void {  
    this.model.fit();  
}  
  
public grid(stepX: number, stepY: number): void {  
    this.model.grid(stepX, stepY);  
}  
  
public interact(interactionData: InteractionData): void {  
    this.model.interact(interactionData);  
}  
  
public draw(drawData: DrawData): void {  
    this.model.draw(drawData);
```

```
}
```

```
public edit(editData: MasksEditData): void {  
    this.model.edit(editData);  
}
```

```
public split(splitData: SplitData): void {  
    this.model.split(splitData);  
}
```

```
public group(groupData: GroupData): void {  
    this.model.group(groupData);  
}
```

```
public join(joinData: JoinData): void {  
    this.model.join(joinData);  
}
```

```
public slice(sliceData: SliceData): void {  
    this.model.slice(sliceData);  
}
```

```
public merge(mergeData: MergeData): void {  
    this.model.merge(mergeData);  
}
```

```
public select(objectState: any): void {
```

```
        this.model.select(objectState);
    }

    public mode(): Mode {
        return this.model.mode;
    }

    public cancel(): void {
        this.model.cancel();
    }

    public configure(configuration: Configuration): void {
        this.model.configure(configuration);
    }

    public isAbleToChangeFrame(): boolean {
        return this.model.isAbleToChangeFrame();
    }

    public get geometry(): Geometry {
        return this.model.geometry;
    }

    public destroy(): void {
        this.model.destroy();
    }
}
```

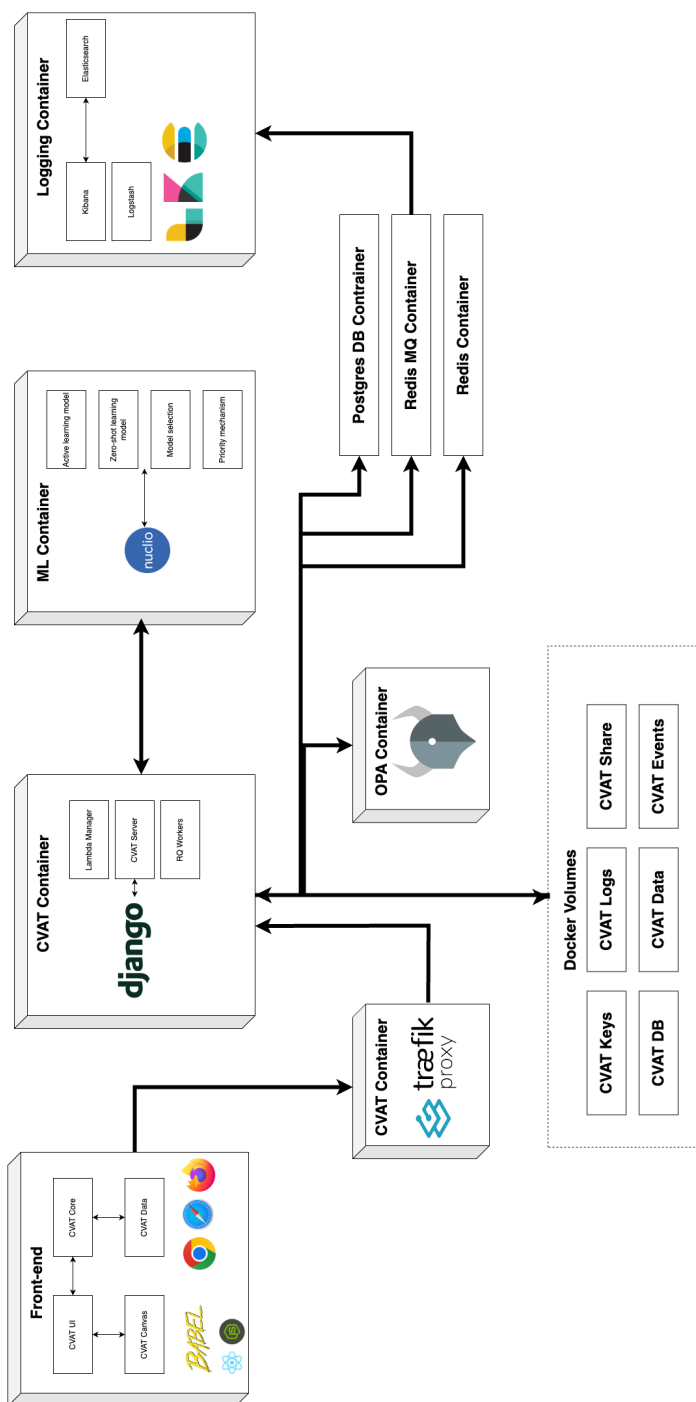
```
export type InteractionData = _InteractionData;
export type CanvasHint = _CanvasHint;
export type InteractionResult = _InteractionResult;
export type HighlightSeverity = _HighlightSeverity;

export {
    CanvasImpl as Canvas, CanvasVersion, RectDrawingMethod,
    CuboidDrawingMethod, Mode as CanvasMode,
};
```



## ДОДАТОК Б.

### ГРАФІЧНЕ ПРЕДСТАВЛЕННЯ АРХІТЕКТУРИ РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ



# ДОДАТОК В.

## ВИХІДНІ ДАНІ ДЛЯ ГРАФІКА БЕЗ АВТОМАТИЗАЦІЇ

### ПРОЦЕСУ РОЗМІТКИ ВІДЕОДАНИХ

	Розміт. 1	Розміт. 2	Розміт. 3	Розміт. 4	Розміт. 5	Розміт. 6	Розміт. 7	Розміт. 8	Розміт. 9	Розміт. 10
Відео 1	9,3	10,0	12,0	10,9	9,5	9,9	9,2	10,9	9,6	10,4
Відео 2	7,5	7,5	9,2	7,4	7,4	8,6	8,7	7,2	7,2	8,7
Відео 3	9,5	10,6	11,3	10,2	10,5	10,0	12,2	12,0	10,6	11,8
Відео 4	13,1	14,3	14,5	13,4	14,8	12,7	12,4	13,8	13,4	12,6
Відео 5	9,0	7,3	7,8	9,7	7,9	7,1	8,4	7,1	9,2	8,7
Відео 6	7,0	9,1	8,4	9,6	8,5	8,2	9,5	8,1	9,1	9,5
Відео 7	15,1	12,3	14,7	13,1	14,3	14,9	13,1	12,4	13,5	12,7
Відео 8	11,7	12,7	11,9	9,9	12,5	11,5	11,3	11,4	12,1	11,5
Відео 9	6,9	6,4	6,7	6,2	8,4	7,1	7,6	7,9	7,4	7,7
Відео 10	10,1	9,8	9,6	9,7	9,4	9,4	9,4	10,9	11,3	9,3

# ДОДАТОК Г.

## ВИХІДНІ ДАНІ ДЛЯ ГРАФІКА З АВТОМАТИЗАЦІЄЮ ПРОЦЕСУ РОЗМІТКИ ВІДЕОДАНИХ ЧЕРЕЗ ПІДХОДИ З НУЛЬОВИМ НАВЧАННЯМ

	Розміт. 1	Розміт. 2	Розміт. 3	Розміт. 4	Розміт. 5	Розміт. 6	Розміт. 7	Розміт. 8	Розміт. 9	Розміт. 10
Відео 1	5,4	6,4	8,4	7,1	4,8	4,8	3,2	7,5	4,9	5,5
Відео 2	4,2	2,1	5,5	3,2	1,8	4,0	5,2	3,5	4,2	4,5
Відео 3	8,4	5,2	6,4	5,7	4,6	6,6	8,6	7,6	5,6	5,8
Відео 4	7,8	10,5	9,8	8,4	10,8	9,3	8,1	9,6	10,2	8,2
Відео 5	4,2	3,5	3,0	4,8	4,7	2,3	1,2	2,9	4,3	3,9
Відео 6	3,0	4,4	4,1	3,1	3,1	5,1	5,0	1,9	3,5	4,0
Відео 7	10,7	7,4	9,6	8,6	10,3	10,2	8,1	7,5	9,4	8,2
Відео 8	8,0	7,1	6,6	6,2	7,2	8,4	8,2	8,3	6,7	7,1
Відео 9	1,9	2,8	3,9	2,6	3,9	2,2	3,7	2,2	3,4	2,3
Відео 10	6,2	5,2	4,7	4,3	4,6	5,5	6,0	7,0	8,2	7,4

**ДОДАТОК Д.**

**ВИХІДНІ ДАНІ ДЛЯ ГРАФІКА З АВТОМАТИЗАЦІЄЮ**

**ПРОЦЕСУ РОЗМІТКИ ВІДЕОДАНИХ ЧЕРЕЗ ПІДХОДИ З**

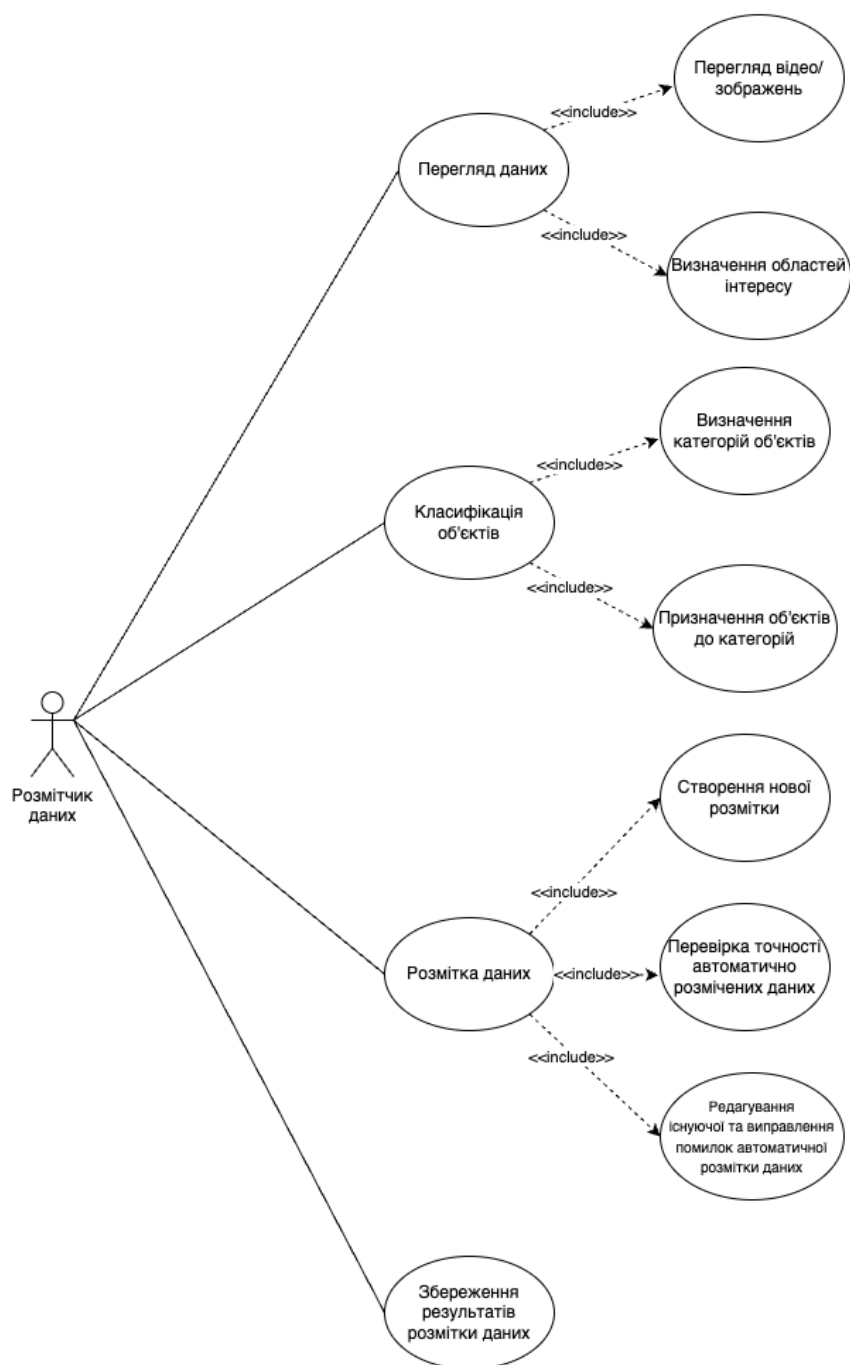
**АКТИВНОГО НАВЧАННЯ**

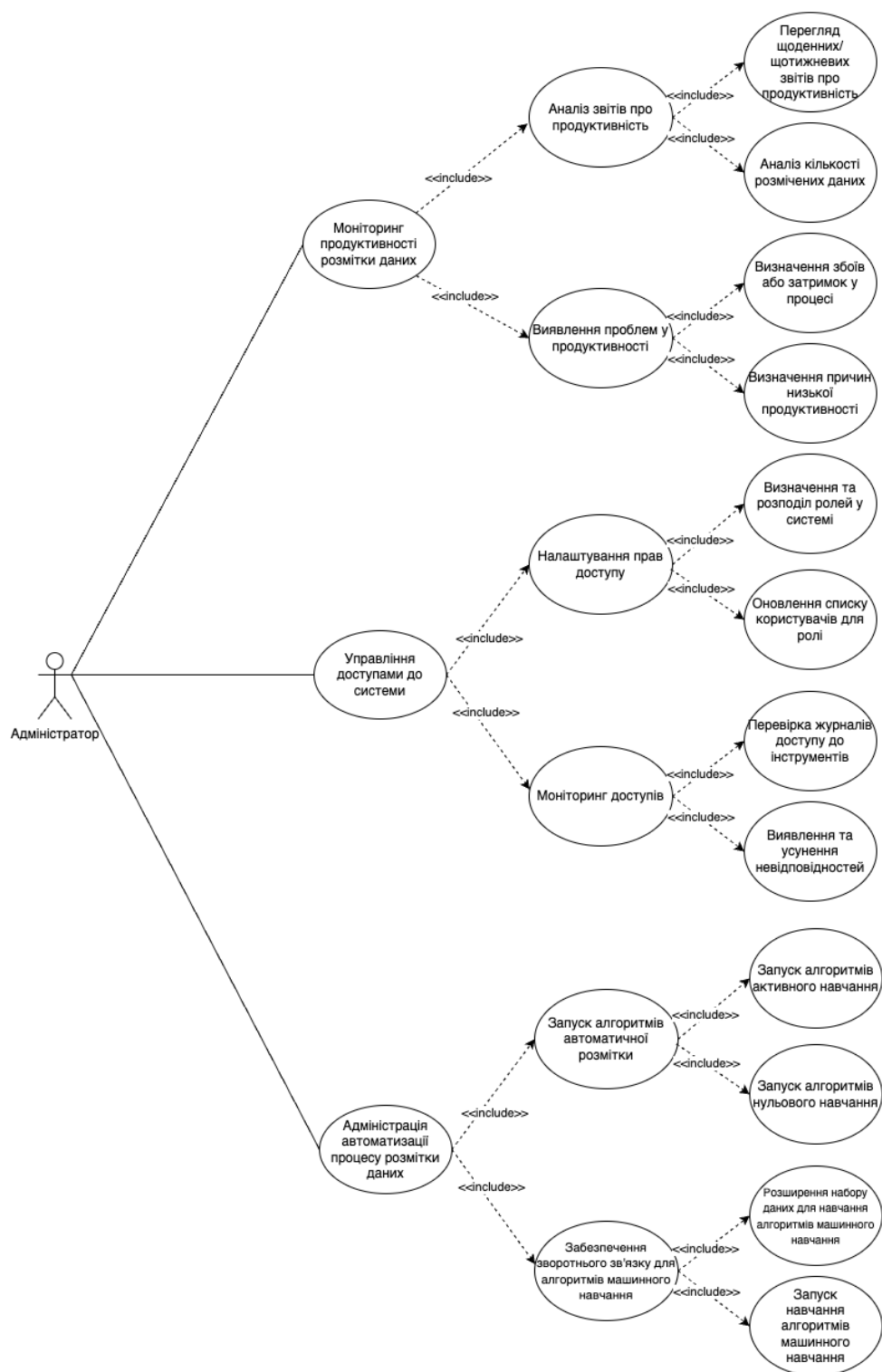
	Розміт. 1	Розміт. 2	Розміт. 3	Розміт. 4	Розміт. 5	Розміт. 6	Розміт. 7	Розміт. 8	Розміт. 9	Розміт. 10
Відео 1	9,6	9,8	11,5	7,9	9,9	9,7	8,9	12,1	10,4	10,4
Відео 2	8,4	9,1	9,2	8,1	7,5	10,1	9,3	9,0	8,1	9,2
Відео 3	10,0	8,5	11,2	10,7	8,4	8,0	11,0	10,6	7,9	9,4
Відео 4	8,9	9,5	11,4	10,7	12,3	10,1	8,8	12,2	10,5	10,8
Відео 5	4,7	4,2	6,1	7,2	5,9	4,0	4,3	4,6	5,5	6,0
Відео 6	4,1	5,4	4,8	3,9	4,7	3,2	4,7	4,5	5,7	6,3
Відео 7	5,1	5,9	4,6	5,2	4,7	7,4	3,5	7,2	2,2	6,1
Відео 8	2,7	5,9	4,5	3,6	4,3	2,4	2,8	3,6	2,0	2,1
Відео 9	2,1	1,6	1,4	2,1	2,2	1,7	2,3	2,9	1,4	2,8
Відео 10	2,2	1,7	2,4	2,2	1,2	3,4	2,7	3,2	2,5	3,1

# ДОДАТОК Е. ВИХІДНІ ДАНІ ДЛЯ ГРАФІКА З АВТОМАТИЗАЦІЄЮ ПРОЦЕСУ РОЗМІТКИ ВІДЕОДАНИХ ЧЕРЕЗ ЗАПРОПОНОВАНИЙ ПІДХІД З ДУАЛЬНОЮ АРХІТЕКТУРОЮ

	Розміт. 1	Розміт. 2	Розміт. 3	Розміт. 4	Розміт. 5	Розміт. 6	Розміт. 7	Розміт. 8	Розміт. 9	Розміт. 10
Відео 1	5,4	6,4	8,4	7,1	4,8	4,8	3,2	7,5	4,9	5,5
Відео 2	4,2	2,1	5,5	3,2	1,8	4,0	5,2	3,5	4,2	4,5
Відео 3	8,4	5,2	6,4	5,7	4,6	6,6	8,6	7,6	5,6	5,8
Відео 4	7,8	10,5	9,8	8,4	10,8	9,3	8,1	9,6	10,2	8,2
Відео 5	4,2	3,5	3,0	4,8	4,7	2,3	1,2	2,9	4,3	3,9
Відео 6	3,0	4,4	4,1	3,1	3,1	5,1	5,0	1,9	3,5	4,0
Відео 7	10,7	7,4	9,6	8,6	10,3	10,2	8,1	7,5	9,4	8,2
Відео 8	2,7	5,9	4,5	3,6	4,3	2,4	2,8	3,6	2,0	2,1
Відео 9	2,1	1,6	1,4	2,1	2,2	1,7	2,3	2,9	1,4	2,8
Відео 10	2,2	1,7	2,4	2,2	1,2	3,4	2,7	3,2	2,5	3,1

## ДОДАТОК Є. ДІАГРАМИ ПРЕЦЕДЕНТІВ ДЛЯ РОЛЕЙ «РОЗІТЧИК» ТА «АДМІНІСТРАТОР» ІНСТРУМЕНТУ РОЗМІТКИ





## ДОДАТОК Ж. ПЕРЕЛІК ПУБЛІКАЦІЙ

**Статті у фахових виданнях, включених до переліку наукових фахових видань України з присвоєнням категорії «Б» / the articles published in scientific journals included in the list of scientific journals of Ukraine in *category «Б»*:**

1. Зарічковий О.А. Дуальна архітектура програмного забезпечення для автоматизації розмітки даних для задач комп'ютерного зору. Міжвідомчий науково-технічний журнал «Адаптивні системи автоматичного управління». 2024. № 45 (2024). С. 109-118. DOI 10.20535/1560-8956.45.2024.313096.

**Статті у періодичних наукових виданнях, проіндексованих у базах даних Web of Science Core Collection та/або Scopus / the publication in scientific periodicals indexed in the Web of Science Core Collection and/or Scopus databases:**

2. Zarichkovyi, A., Stetsenko, I.V. (2024) 'Attr4Vis: Revisiting importance of attribute classification in Vision-Language Models for Video Recognition', *International Journal of Computing*, 23 (1), pp. 94-100. DOI 10.47839/ijc.23.1.3440.
3. Zarichkovyi, A., Stetsenko, I.V. (2023) 'Boundary Refinement via Zoom-In Algorithm for Keyshot Video Summarization of Long Sequences', *Lecture Notes on Data Engineering and Communications Technologies*, 180, pp. 344-359. DOI 10.1007/978-3-031-36115-9\_32.
4. Zarichkovyi, A., Stetsenko, I.V. (2023) 'Hard Samples Make Difference: An Improved Training Procedure for Video Action Recognition Tasks', *Lecture Notes in Networks and Systems*, 544, pp. 508-519. Springer, Cham. DOI 10.1007/978-3-031-16075-2\_36.
5. Oleksandr Zarichkovyi and Iryna Mukha. (2021) 'Approximate Training of Object Detection on Large-Scale Datasets', *Lecture Notes on Data Engineering and*



*Communications Technologies*, 83, pp. 389-400. DOI 10.1007/978-3-030-80472-5\_32.

**Публікація у матеріалах наукової конференції / the publication in the proceedings of the scientific conference:**

6. Zarichkovyi, A., Stetsenko, I.V. Improving cross-modal knowledge exploration of vision language models. Інженерія програмного забезпечення і передові інформаційні технології (Soft Tech-2024): матеріали VI Міжнародної науково-практичної конференції молодих вчених та студентів, 21-23 травня 2024 року, м. Київ, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», ФІОТ. С. 58-61.