



Resumo Estruturado de Vídeo Usando LLMs e MLLMs

RUI JORGE MACHADO NETO

Junho de 2025

Structured Video Summarization Using LLMs and MLLMs

Rui Neto

**A dissertation submitted in partial fulfillment of
the requirements for the degree of Master of Science,
Specialisation Area of Software Engineering**

Advisor: Dr. Nuno Pereira

Statement of Integrity

I hereby declare having that I have conducted this academic work with integrity.

I have not plagiarised or applied any form of undue use of information or falsification of results along the process leading to its elaboration.

Therefore, the work presented in this document is original and authored by me, having not previously been used for any other end. The exceptions are explicitly recognised in the section “Ethical considerations” of the first chapter. This section also states how AI tools were used and for what purpose.

I further declare that I have fully acknowledged the Code of Ethical Conduct of P.PORTO.

ISEP, Porto, 2025-06-29

Abstract

In industrial environments, operational safety and efficiency depend heavily on timely detection of anomalies. This dissertation presents a complete, structured video summarization pipeline tailored to identify anomalies in industrial settings, using recent advances in Large Language Models (LLMs) and Multimodal LLMs (MLLMs). Beyond reviewing state-of-the-art methodologies in video captioning and anomaly detection, this work delivers a practical implementation combining intelligent frame sampling, context-aware captioning using advanced MLLMs such as gpt-4.1-mini and gemini-2.5-pro, and object detection via YOLOv11.

A custom benchmark dataset with 100 Image-Question-Answer (IQA) triplets was developed to evaluate the perceptual capabilities of various MLLMs in industrial scenarios. Additionally, a novel "Model-as-a-Judge" framework was employed to assess models captioning and pipeline summarization quality beyond lexical metrics. The final pipeline achieved a summarization quality score of 0.72 and accurately detected five of six safety-critical anomalies in over an hour of our self-recorded, real-world CNC machine footage.

The research has been recognized by the scientific community, being accepted for presentation at the SASYR Symposium. These contributions advance the field of applied Artificial Intelligence (AI) for industrial safety monitoring through a robust and efficient multimodal video analysis system.

Keywords: Video Summarization, Large Language Models, Multimodal Large Language Models, Industrial Safety, Anomaly Detection

Resumo

Em ambientes industriais, a segurança operacional e a eficiência dependem fortemente da detecção atempada de anomalias. Esta dissertação apresenta uma *pipeline* completa e estruturada de sumarização de vídeo, adaptada para identificar anomalias em cenários industriais, utilizando os avanços recentes em *Large Language Models* (LLMs) e *Multimodal LLMs* (MLLMs). Além de rever as metodologias de ponta em legendagem de vídeo e detecção de anomalias, este trabalho oferece uma implementação prática que combina *frame sampling*, legendagem do contexto com recurso a MLLMs avançados como o gpt-4.1-mini e o gemini-2.5-pro, e detecção de objetos através do YOLOv11.

Foi desenvolvido um *dataset* de referência personalizado com 100 trios de Imagem-Pergunta-Resposta para avaliar as capacidades de percepção de vários MLLMs em cenários industriais. Adicionalmente, foi utilizada uma nova metodologia designada "Model-as-a-Judge" para aferir a qualidade das legendagens e da sumarização para além das métricas lexicais. A *pipeline* final alcançou uma pontuação de qualidade de sumarização de 0.72 e detetou com precisão cinco das seis anomalias críticas para a segurança, em mais de uma hora de filmagens reais de máquinas CNC gravadas por nós.

A investigação foi reconhecida pela comunidade científica, tendo sido aceite para apresentação no Simpósio SASYR. Estas contribuições representam um avanço no campo da Inteligência Artificial (IA) aplicada à monitorização da segurança industrial, através de um sistema de análise de vídeo multimodal robusto e eficiente.

Acknowledgement

I would first like to express my heartfelt gratitude to Professor Nuno Pereira for his tireless support throughout the development of this dissertation - from the bureaucratic aspects to the smallest details in the implementation of the solution.

My deepest thanks also go to my parents and my sister Sofia for their constant support during challenging moments and for their incredible understanding. A special thank you to Carolina for her patience and encouragement during a time that demanded so much from me.

Finally, I am especially grateful to my friends, particularly Simão and Nuno, for their valuable advice during this period - and throughout the many stages of my university's journey.

Could not be more grateful. Thank you all for everything!

Contents

List of Figures	xiii
List of Tables	xv
List of Source Code	xvii
List of Acronyms	xix
1 Introduction	1
1.1 Context and Problem	2
1.2 Objectives	3
1.3 Ethical Considerations	3
1.4 Document Structure	4
2 Concepts and Research Method	5
2.1 Key Concepts	5
2.1.1 Large Language Models	5
2.1.2 Multimodal Large Language Models	6
2.2 Research Method	6
2.2.1 Research Questions	6
2.2.2 Data Source and Search Queries	7
2.2.3 Eligibility criteria	7
2.2.4 Data Collection Process	8
2.2.5 Results	9
3 State of the Art	11
3.1 Case Studies and Applications	11
3.1.1 MISAR (2023)	12
3.1.2 VidIL (2022)	12
3.1.3 M3SUM (2024)	12
3.1.4 AVIWT (2023)	13
3.1.5 XaiR (2024)	13
3.2 Steps in Video Summarization Frameworks	14
3.2.1 Frame Sampling	15
3.2.2 Frame-by-frame Captioning	16
3.2.3 Object Detection	18
3.2.4 External Content Integration	18
3.2.5 Temporal Awareness	22
3.2.6 Summarization	23
3.3 Datasets	25
3.3.1 Candidate Datasets	25

3.3.2	Video Games	27
3.4	Evaluation Methods and Metrics	28
3.4.1	Human Evaluation	28
3.4.2	Lexical-Based Metrics	28
3.4.3	Embedding-Based Metrics	29
3.4.4	Model-as-a-Judge	29
4	Design	31
4.1	Methodology	31
4.2	Pipeline Architecture	32
4.2.1	Pipeline Breakdown	32
4.3	Captioning Models Assessment	35
4.3.1	Dataset Modeling	35
4.3.2	Assessment Method	37
5	Implementation	39
5.1	Models Assessment	39
5.1.1	Dataset Construction	39
5.1.2	Experimental Setup and Workflow	41
5.2	Summarization Pipeline	44
5.2.1	Video Processing Module	44
5.2.2	Object Detection Module	45
5.2.3	Frame Captioning Module	45
5.2.4	Final Summarization Module	46
6	Results	49
6.1	Models Assessment Results	49
6.1.1	Overall Model Performance	49
6.1.2	Performance by Category	50
6.1.3	Performance by Capability	51
6.1.4	Qualitative Analysis	52
6.2	Pipeline Summarization Results	54
6.2.1	Summary Results	54
6.2.2	Alert Detection Results	55
7	Conclusions and Future Work	57
7.1	Conclusions	57
7.2	Future Work	58
7.2.1	Context-Aware RAG for Domain-Specific Knowledge	58
7.2.2	Fine-Tuning for Specialized Object Detection	58
7.2.3	Incorporating Audio as an Input Modality	58
	Bibliography	59

List of Figures

1.1	Number of casualties over four days of missed work in Japan.	2
2.1	Systematic flow for RQ-1 and RQ-2 papers gathering.	8
2.2	Systematic flow for RQ-3 papers gathering.	9
3.1	Draft pipeline with most common steps in video summarization frameworks.	14
3.2	Best reasoning models chart.	24
3.3	Print screen of the game "Milling Machine Simulator".	27
3.4	Print screen of the game "Lathe Machine Simulator".	28
4.1	Systematic design methodology.	31
4.2	Overview of the designed pipeline.	32
4.3	Categories and associated tasks of custom dataset.	36
4.4	Assessment method for MLLM performance.	37
5.1	Category frequency in the custom dataset.	40
5.2	Capability frequency in the custom dataset.	41
5.3	Prompt given to Judge Model for MLLM response evaluation.	42
5.4	Pipeline SSIM approach explained.	44
5.5	Visual workflow of the object detection module.	45
5.6	Dynamic prompt construction for captioning.	46
5.7	Dynamic prompt construction for summarization.	46
6.1	Overall MLLM Performance on our custom industrial dataset.	49
6.2	Average MLLM scores by category (radar chart).	50
6.3	Average MLLM scores by capability (radar chart).	51
6.4	Comparison of model responses to a Functional Reasoning question.	52
6.5	A qualitative example from the Spatial Awareness capability.	53

List of Tables

2.1	Research method results.	9
3.1	Output studies from research method.	11
3.2	Comparison of steps across different approaches.	14
3.3	Evaluation of datasets against project-specific criteria.	27
6.1	Average MLLM scores by category.	50
6.2	Average MLLM scores by capability.	51
6.3	Contexts of worker interactions with CNC machine.	54
6.4	Evaluation of video summaries.	55
6.5	Validation of the alert detection across test scenarios.	56

List of Source Code

2.1	Research query for RQ-1 and RQ-2.	7
2.2	Research query for RQ-3.	7
5.1	The structure of an entry in our custom dataset.	40
5.2	Structure of a model's evaluation output file.	43
6.1	Prompt used to guide the pipeline summaries evaluation.	54

List of Acronyms

AI	Artificial Intelligence.
API	Application Programming Interface.
AR	Augmented Reality.
BVH	BioVision Hierarchy.
C2PSA	Cross Stage Partial with Spatial Attention.
CNN	Convolutional Neural Network.
CPT	Continual Pre-training.
FDP	Fixed Detection Perspective.
FPP	First-Person Perspective.
GELAN	Generalized Efficient Layer Aggregation Network.
GPU	Graphics Processing Unit.
IQA	Image-Question-Answer.
KIPT	Knowledge Inherited Pre-training.
LLM	Large Language Model.
LoRA	Low-Rank Adaptation.
LSTM	Long Short-Term Memory.
MLLM	Multimodal Large Language Model.
NLP	Natural Language Processing.
NMS	Non-Maximum Suppression.
OCR	Optical Character Recognition.
PEFT	Parameter-Efficient Fine-Tuning.
PGI	Programmable Gradient Information.
PTS	Pre-training from Scratch.
RAG	Retrieval-Augmented Generation.
RL	Reinforcement Learning.
SPT	Simultaneous Pre-training.
SSIM	Structural Similarity Index Measure.

TAPT Task Adaptive Pre-training.

XR Extended Reality.

YOLO You Only Look Once.

Chapter 1

Introduction

In today's industrial landscape, ensuring operational safety and efficiency is paramount [16]. However, traditional reliance on human monitoring for detecting machine-related anomalies is not only costly but also inherently prone to errors of fatigue and oversight [43]. The recent advancements in Artificial Intelligence (AI), particularly Large Language Models (LLMs) and Multimodal Large Language Models (MLLMs), present an opportunity to automate and enhance these critical oversight processes.

In this dissertation, the authors address this challenge by designing, implementing, and evaluating an innovative pipeline for the structured summarization of industrial video data.

The primary goal is to automatically generate concise, coherent summaries and, most importantly, to detect operational anomalies and safety violations from real-world footage of human-machine interactions. To achieve this, a complete, end-to-end system featuring a modular architecture to process video in near-real-time was developed. The research engineered a content-aware frame selection mechanism to ensure computational efficiency, integrated a state-of-the-art YOLOv11 model for precise object detection, and used the advanced capabilities of models like gpt-4.1-mini and gemini-2.5-pro for context-aware frame captioning and final summarization.

A significant barrier in this field is the scarcity of public datasets tailored for nuanced, task-specific evaluation in industrial settings. To overcome this obstacle, the team constructed a novel benchmark dataset of 100 Image-Question-Answer (IQA) triplets, specifically designed to test MLLMs on core capabilities within the context of industrial scenarios. Confronted with the limitations of traditional evaluation metrics, a "Model-as-a-Judge" methodology was also implemented to ensure a fair and semantically accurate assessment of modern generative models.

The empirical assessment of MLLMs on the custom dataset provided a clear hierarchy of model performance, identifying gemini-2.0-flash and gpt-4.1-mini as the top performers with scores of 0.75 and 0.72 respectively on a 0-to-1 scale.

When tested on real-world footage of a CNC machine in operation, recorded by the authors, the final pipeline proved highly effective, achieving a summarization quality score of 0.72 and successfully detecting five out of six predefined operational and safety anomalies, including an engaged emergency stop button and an attempt to operate the machine with the door open.

1.1 Context and Problem

In many industries, operational disruptions caused by undetected errors or inefficiencies can lead to safety incidents and financial losses. For example, in 2018, in Japan, 127,329 people were injured in industrial accidents [16]. Figure 1.1 shows the number of casualties over four days of missed work in Japan over time [16].

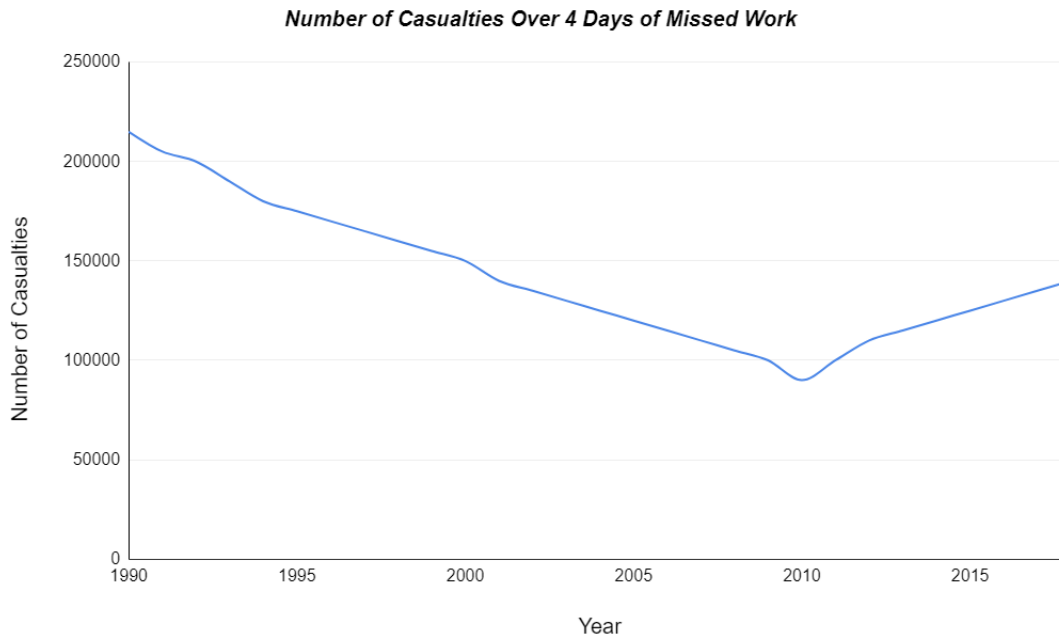


Figure 1.1: Number of casualties over four days of missed work in Japan [16].

Also, companies that actively minimize human errors and workplace incidents, consistently achieve significant long-term cost savings by reducing expenses related to accidents [8]. But training personnel to monitor these events consistently and accurately is not only expensive but also fails to eliminate errors due to fatigue and oversight. Automated detection of machine-related issues in industrial contexts can be an effective way to improve operational safety. For manufacturing and insurance companies, using factory-installed sensors - such as video capture devices, sensors, and machine monitoring tools - to analyze operational data could mean the difference between seamless operations and costly downtime.

Recent deep learning models have revolutionized a various number of industries, including video surveillance [43]. Surveillance systems can now analyze large amounts of data quickly and accurately, reducing the burden on human operators [43] and detecting possible errors using less resources. This research seeks to take advantage of the capabilities of LLMs and MLLMs to create structured video summaries adapted to industrial applications. These summaries are designed to capture the logical sequence of events and highlight anomalies. The method proposed builds on the recent success of machine learning models in integrating textual and visual data to produce meaningful outputs.

1.2 Objectives

The global goal of this dissertation is to address the challenge of analyzing industrial video data to enhance operational safety. This is achieved by using the capabilities of recent LLMs and MLLMs to generate structured, context-aware video summaries. These summaries condense long footage into actionable insights that support error detection and preventive safety interventions. This goal is supported by the following specific objectives:

- Survey State-of-the-Art Techniques

Conduct a systematic review of the latest methods in video summarization using LLMs and MLLMs, particularly those relevant to industrial safety, video captioning, and cognitive assistance. This includes examining techniques such as few-shot video-language processing and zero-shot video understanding.

- Explore and Construct Evaluation Datasets

Identify existing public datasets suitable for industrial scenarios assessment or simulate relevant industrial environments to produce custom data. Where needed, construct and annotate datasets tailored to evaluating the reasoning and perceptual capabilities of modern machine learning models in safety settings.

- Design, Implement, and Assess a Summarization Pipeline

Develop a modular system that generates summaries using cutting-edge models. Evaluate the system's effectiveness through both quantitative metrics and qualitative assessments, emphasizing accuracy and the detection of potential anomalies during operation.

1.3 Ethical Considerations

This project focus on using LLMs and MLLM. Therefore, key tasks include experimenting with datasets of public videos, evaluating state-of-the-art AI models, and building a prototype capable of generating structured video summaries. Throughout these activities, ethical responsibility will be integral to the project's design and execution. To ensure privacy, no organizational data will be collected. Instead, the project will rely on personal or public datasets, ensuring adherence to principles of data minimization and transparency. Data sources will be documented. The AI models used will be critically evaluated for fairness and non-bias to ensure ethical and transparent outputs. Special attention will be given to aligning the system's development in a way that prevents misuse.

The solution of this dissertation extensively utilizes AI models. However, for the writing of this document, AI tools were used solely for proofreading and linguistic refinement. They were not used to generate ideas, arguments, or content.

Although this project is a proof of concept, the collection of images could lead to issues related to the privacy of workers, making it necessary to address these aspects thoroughly. These practices will be aligned with established ethical standards and guidelines, including the *Code of Good Practices and Conduct of the Polytechnic of Porto*¹, the *Code of Ethics and Deontology of the Portuguese Engineering Association*², and the principles of the *European AI Act*³.

¹<https://diariodarepublica.pt/dr/detalhe/despacho/11171-2020-148327696>

²https://www.ordemengenharios.pt/fotos/editor2/regulamentos/codigo_ed.pdf

³<https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai>

- Code of Good Practices and Conduct (Polytechnic of Porto)
Articles 2.f and 2.m emphasize respect for dignity, privacy, integrity, and rigor. These will guide data safeguarding and ensure high academic standards in the representation of results.
- Portuguese Engineering Association Code of Ethics
Section 1 emphasizes safety and responsibility, principles that align with the system's goal of detecting and mitigating risks during industrial operations. Section 2.5 advises avoiding harm, which will be addressed by rigorously testing AI models to minimize biases and potential misuse.
- European AI Act
Principles of data minimization and transparency will guide the handling of datasets. Articles 10.g and 10.h on fairness and transparency will guide the evaluation and refinement of AI models to ensure ethical outputs.

It is important to note that portions of the state of the art review in this dissertation are adapted from a prior academic report but have been substantially revised and expanded.

1.4 Document Structure

To ensure clarity and a logical flow, this dissertation is organized into the following chapters:

- Chapter 1 - Introduction: This chapter introduces the research by outlining the problem of industrial video data analysis, the objectives of the study, and the overall scope. It discusses the potential of LLMs and MLLMs to address these challenges and details the ethical considerations that guided the research.
- Chapter 2 - Concepts and Research Method: This chapter establishes the foundational concepts. It also outlines the systematic research methodology used to identify the key literature that informs the state of the art review.
- Chapter 3 - State of the Art: Provides a comprehensive review of existing literature and case studies in video summarization. It explores current methods for integrating LLMs and MLLMs together, highlighting the most advanced techniques in the field.
- Chapter 4 - Design: This chapter details the design of the proposed video summarization pipeline. It covers the overall architecture and the methodology for assessing the captioning models, including the creation of a new dataset and the evaluation method.
- Chapter 5 - Implementation: This chapter describes the practical implementation of the designed system. It covers the construction of the custom dataset for model assessment and the development of the end-to-end video summarization pipeline, detailing the tools and workflows used.
- Chapter 6 - Results: This chapter presents the empirical findings from the models' assessment and the pipeline's performance on the real-world test videos. It includes a quantitative analysis of model performance and the pipeline's effectiveness in generating summaries and detecting anomalies.
- Chapter 7 - Conclusions and Future Work: This final chapter summarizes the research, reflecting on the key findings and achievements. It also discusses the limitations of the current work and proposes promising directions for future research and development.

Chapter 2

Concepts and Research Method

This chapter defines the key concepts used throughout the document and outlines the research methodology employed to identify the papers that inform the state of the art discussion in Chapter 3.

2.1 Key Concepts

An important first step is to clarify the difference between LLMs and MLLMs. While both represent cutting-edge advances in AI, they differ fundamentally in their design, capabilities, and application domains.

2.1.1 Large Language Models

LLMs represent a class of deep learning models that have revolutionized Natural Language Processing (NLP) [30]. They are defined as large-scale, pre-trained, statistical language models built upon neural network architectures, most notably the Transformer [30]. Their defining characteristic is their immense scale, with modern LLMs containing tens to hundreds of billions of parameters, and being pre-trained on massive, web-scale texts in a self-supervised fashion [30]. The primary function of an LLM is to process, understand, and generate coherent and contextually relevant text.

A key phenomenon associated with this scale is the manifestation of "emergent abilities" - capabilities that are not present in smaller-scale models but appear as model size and training data increase. These abilities include [2]:

- In-context learning: An LLM can learn to perform a new task from just a few examples provided within the prompt at inference time, without any updates to its weights.
- Instruction following: After a phase of instruction tuning, models can generalize to follow natural language instructions for new tasks without requiring explicit examples.
- Complex reasoning: Chain-of-thought prompting, where the model can break down a complex problem into intermediate reasoning steps to arrive at a final answer.

Despite these powerful cognitive capabilities, LLMs are fundamentally unimodal systems. They are trained exclusively on text data and can only perceive and process information in the form of text tokens [30]. This renders them inherently "blind" to all other modalities, such as images, audio, or video. This core limitation is the primary catalyst for the development of MLLMs, which seek to augment the powerful reasoning engine of the LLM with perceptual capabilities [54].

2.1.2 Multimodal Large Language Models

MLLMs represent a more generalized and ambitious class of AI. MLLMs are defined as sophisticated models designed to seamlessly process, integrate, and generate information across a diverse spectrum of data types, which can include not only text and images but also audio, video, and even more specialized data like biological signals [54].

The fundamental goal of MLLMs is to achieve a more human-like understanding of the world by joining information from these multiple sources. They are capable of accepting and reasoning over interleaved text and image inputs to perform tasks that require deep cross-modal understanding. The development of MLLMs is seen as a crucial pathway toward more general-purpose and adaptive AI systems [54].

The typical MLLM architecture is composed of three key modules [54]:

- **Modality Encoders (the senses):** These are specialized, pre-trained models that serve as the sensory organs of the model. Each encoder is responsible for a specific modality; for example, a vision encoder processes images, while an audio encoder processes sound waves. These encoders receive raw sensory signals and transform them into high-level feature representations.
- **Modality Interface/Adapter (the interpreter):** This is arguably the most critical and actively researched component of the MLLM architecture. The adapter's function is to serve as an interpreter, aligning the different modalities by translating the feature representations generated by the encoders into a "language" or format that the LLM "brain" can comprehend. The effectiveness of this translation is paramount to the overall performance of the model.
- **LLM Backbone (the brain):** At the core of the MLLM is a powerful, pre-trained LLM. This LLM does not perceive the world directly. Instead, it functions as the central reasoning engine of the system. It receives the processed feature representations from the various modality encoders and uses its vast parametric knowledge to understand the multimodal input, follow instructions, and generate a coherent output.

2.2 Research Method

The current section details the systematic research approach undertaken to identify and analyze relevant literature. It includes the research questions, search queries, eligibility criteria, and the data collection process. The findings provide a foundation for the subsequent analysis of case studies and their steps in industrial video summarization.

2.2.1 Research Questions

To guide the systematic review, the following research questions were formulated:

- RQ-1: What are the methods for applying LLMs and MLLMs to video summarization tasks, such as video captioning and cognitive assistants?
- RQ-2: Do existing pre-trained models performing approaches such as few-shot video-language processing, unsupervised language-guided summarization, and zero-shot video understanding help circumvent the need to train them?
- RQ-3: What datasets exist for industrial human action recognition and machine operation validation?

2.2.2 Data Source and Search Queries

Data Source

*Google Scholar*¹ was selected as the primary data source due to its comprehensive aggregation of academic libraries, journals, and publications. It provides access to a diverse collection of reviewed articles, conference papers, and other scholarly materials, making it a reliable and extensive resource for state-of-the-art researches.

Search Queries

The search queries were meticulously designed to capture a wide range of relevant literature for each research question.

RQ-1 and RQ-2

```
multimodal summarization
OR ((few-shot video-language
    OR unsupervised language-guided video summarization
    OR multimodal large language models
    OR zero-shot video understanding)
AND (video captioning
    OR cognitive assistant
    OR video-understanding tasks))
```

Listing 2.1: Research query for RQ-1 and RQ-2.

RQ-3

```
intitle:dataset
AND industrial context
AND multimodal
AND (human action recognition
    OR human-robot
    OR human-machine)
AND (temporal
    OR procedural)
```

Listing 2.2: Research query for RQ-3.

2.2.3 Eligibility criteria

RQ-1 and RQ-2

Inclusion Criteria:

- Reviewed articles and conference papers published later than 2022, motivated by the release of ChatGPT in that year, which triggered a dramatic shift in AI research.
- Studies focusing on multimodal summarization, LLMs and MLLMs.

Exclusion Criteria:

- Articles not written in English.
- Studies lacking sufficient experimental or methodological details.

¹<https://scholar.google.com/> (accessed 2025-06-29)

RQ-3

Inclusion Criteria:

- Reviewed articles and conference papers published later than 2020.

Exclusion Criteria:

- Articles not written in English.
- Studies lacking sufficient experimental or methodological details.

2.2.4 Data Collection Process

The data collection process closely followed the PRISMA guidelines to ensure a systematic and transparent review. The process involved three key steps.

1. Identification: Conducting searches using the predefined queries in Google Scholar.
2. Screening: Filtering the results based on titles and abstracts to exclude irrelevant and duplicated studies; and assessing the full text of selected studies against the inclusion and exclusion criteria.
3. Inclusion: Finalizing the selection of studies for analysis and inclusion in the review.

Figure 2.1 depicts the systematic flow for gathering papers to address RQ-1 and RQ-2.

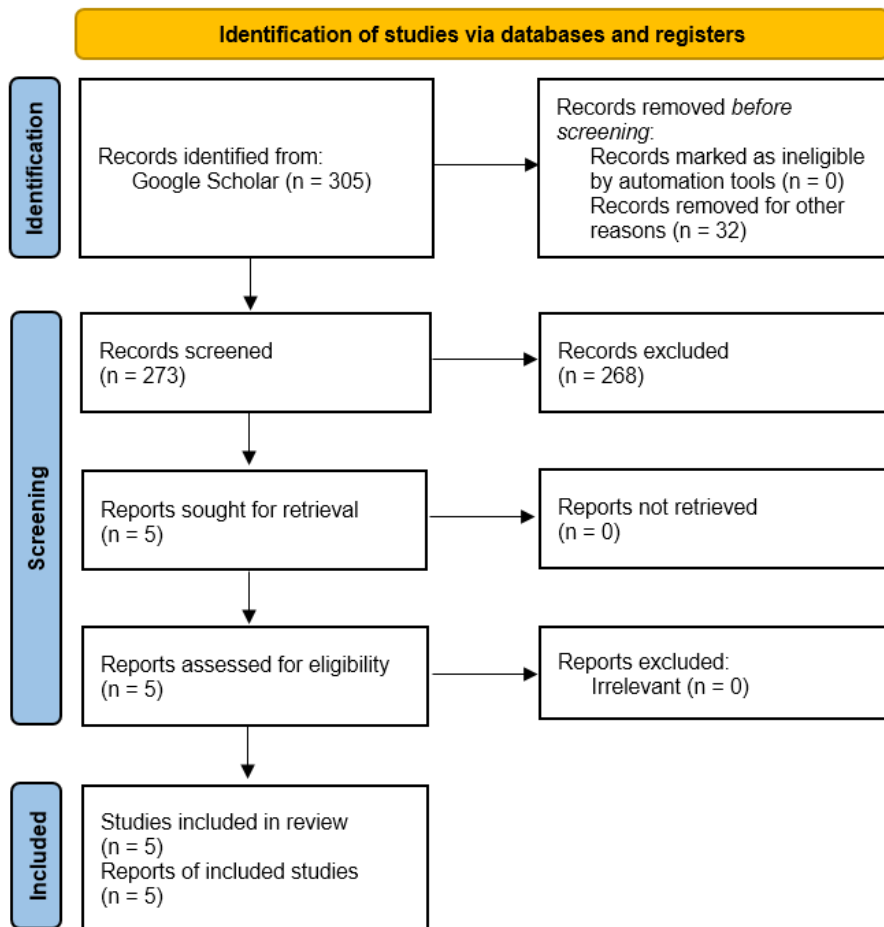


Figure 2.1: Systematic flow for RQ-1 and RQ-2 papers gathering.

Figure 2.2 depicts the systematic flow for gathering papers to answer RQ-3.

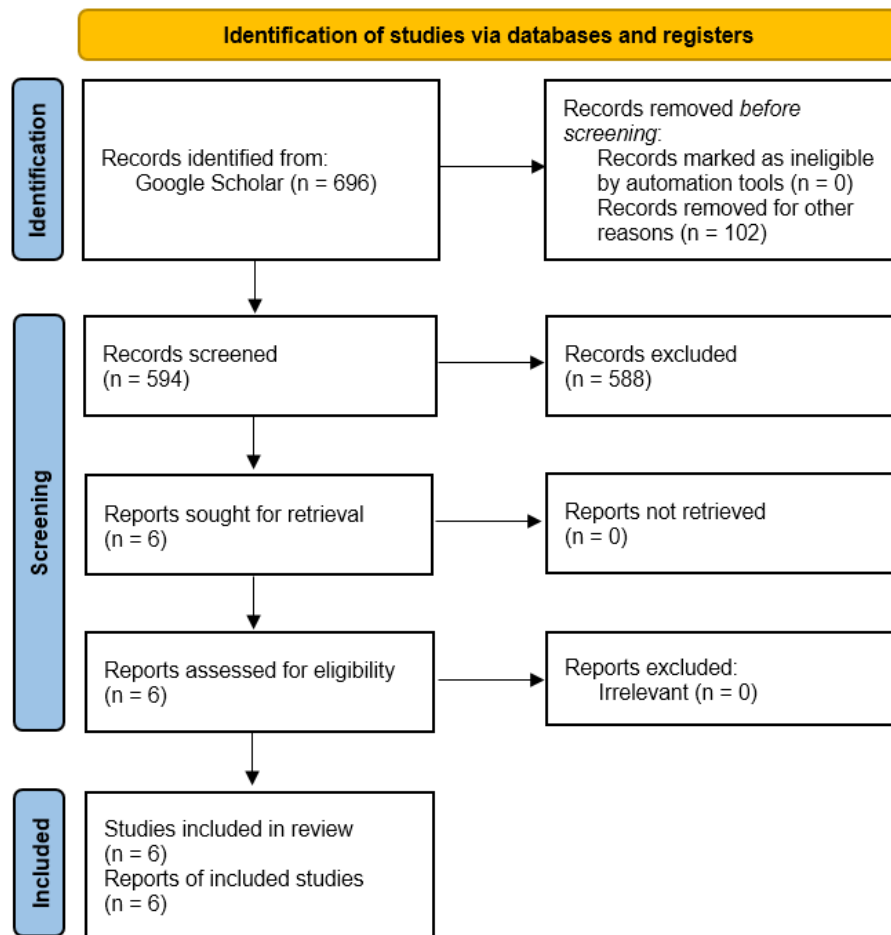


Figure 2.2: Systematic flow for RQ-3 papers gathering.

2.2.5 Results

The most relevant case studies and datasets identified through the systematic review are detailed in Chapter 3. Table 2.1 summarizes the key papers resulting from the research. In addition, a snowballing approach was employed to identify further references cited throughout the document.

Table 2.1: Research method results.

Research Questions	Papers
RQ-1, RQ-2	[4], [49], [48], [3], [42]
RQ-3	[7], [38], [34], [5], [58], [52]

Chapter 3

State of the Art

This Chapter presents a review of recent case studies and their approaches and methodologies, derived from the output papers of the research method, addressed in Section 2.2. It emphasizes the pipeline steps, approaches, and techniques integrated in the case studies, and discusses the available datasets to evaluate the prototype.

3.1 Case Studies and Applications

The development of a robust video summarization system hinges on understanding existing approaches and their implementation in similar contexts. In this section, we delve into notable studies that employ pipelines similar to the one aimed to do. The main objective of analyzing these case studies is to uncover the steps and technologies that have proven effective in addressing the challenges of structured video summarization.

Each case study is examined for its pipeline structure, detailing the key steps. The insights drawn from these applications will inform critical steps, ensuring that the proposed system follows best practices.

Five studies were chosen and they are presented in Table 3.1. These studies were the output of the research method described in Section 2.2. They approach video summarization as a structured pipeline, which closely matches the methodological requirements of our project.

Table 3.1: Output studies from research method. We assigned the name AVIWT for ease of reference throughout the document, as the original authors did not provide an official name.

Paper Title	Framework Name	Source	Year
MISAR: A Multimodal Instructional System with Augmented Reality	MISAR	[4]	2023
Language Models with Image Descriptors are Strong Few-Shot Video-Language Learners	VidIL	[49]	2022
M3SUM: A Novel Unsupervised Language-guided Video Summarization	M3SUM	[35]	2024
A Video Is Worth 4096 Tokens: Verbalize Videos to Understand Them in Zero Shot	AVIWT	[3]	2023
XaiR: An XR Platform that Integrates Large Language Models with the Physical World	XaiR	[42]	2024

3.1.1 MISAR (2023)

The MISAR system [4] uses Augmented Reality (AR) to enhance task performance through multimodal integration of visual, auditory, and linguistic data. The core innovation lies in its use of an LLM (GPT-3.5-Turbo) to process and integrate data streams, enabling contextual understanding and real-time user assistance. This approach is particularly aimed at task quantification and error correction in AR environments.

- **Frame Sampling:** To align with real-time requirements, captions are generated for every 8 frames, based on a 30 fps video stream, balancing computational efficiency with output quality.
- **Frame-by-frame Captioning:** Video frames are processed using LaViLa, a pre-trained video-to-text model that generates descriptive captions for each frame.
- **External Content Integration:** MISAR incorporates additional content by combining recipes documentation and GPT-3.5-Turbo's inferential capabilities. This step ensures the outputs are adapted to the user's current state and task.

3.1.2 VidIL (2022)

VidIL [49] is a system that uses large-scale pre-trained language models alongside image-language models to perform video-to-text tasks with minimal supervision. The approach is particularly suited for few-shot learning scenarios, requiring no pre-training or fine-tuning on video datasets. VidIL excels in tasks such as video captioning and question answering, by utilizing a representation of videos that integrates spatial and temporal information.

- **Frame Sampling:** To reduce computational complexity, frames are sparsely sampled, typically selecting four for frame-level analysis and eight for visual tokenization. This ensures that essential visual features are captured while maintaining efficiency.
- **Frame-by-frame Captioning:** Frame captions are generated using BLIP, a pre-trained image-language model. BLIP provides both descriptive and filtered captions for each sampled frame.
- **Object Detection:** Visual tokens, including objects, are extracted using CLIP, a multimodal contrastive model. CLIP retrieves the most relevant textual descriptors for each frame.
- **Temporal Awareness:** Temporal markers such as "First", "Then", and "Finally" are injected into prompts to structure frame-level captions and visual tokens in a coherent sequence. This enables the system to interpret temporal dynamics effectively.

3.1.3 M3SUM (2024)

M3SUM [48] introduces an unsupervised approach to language-guided video summarization, which avoids the need for training data and complex feature alignment models. The system utilizes off-the-shelf models to convert multimodal video data (frames and audio) into textual descriptions, which are then processed by an LLM for generating video summaries based on the highest score frames.

- **Frame Sampling:** The authors adopted a sampling strategy of captioning one frame every 2 seconds to capture key visual information throughout the video.

- **Frame-by-frame Captioning:** The visual content of each video frame is processed using ViT-GPT2, an image-to-caption model. It generates detailed captions for each frame, capturing key visual elements.
- **Temporal Awareness:** Each frame is associated with its corresponding transcript segment. This is done in the Algorithm 1 presented in the paper.

3.1.4 AVIWT (2023)

This work [3] introduces a novel framework that verbalizes videos into coherent textual stories using multimodal input (visual, audio, and metadata) to enable downstream video understanding tasks in a zero-shot setting. All textual elements (transcripts, captions, Optical Character Recognition (OCR) data, and metadata) are combined into a structured prompt for an LLM, GPT-3.5.

- **Frame Sampling:** Keyframes are extracted using GMFlow, an optical flow-based algorithm, or uniform sampling for longer videos. This ensures efficient processing by selecting frames that represent the video's major events or scenes.
- **Frame-by-frame Captioning:** Each selected keyframe is processed with the BLIP-2 model to generate captions describing the scene.
- **Object Detection:** BLIP-2 additionally performs object recognition to identify and label distinct elements within each frame, such as characters, items, or environmental features using prompts like "Describe the objects in this image".
- **External Content Integration:** Metadata, including video titles, channel names, and related information, is retrieved via Application Programming Interfaces (APIs) and Wikidata.

3.1.5 XaiR (2024)

XaiR [42] is a platform that integrates MLLMs with Extended Reality (XR) to enable intelligent task guidance. XaiR facilitates step-by-step task instruction, object detection, and AR content placement. The system is built on a split-architecture design, where heavy computational tasks are offloaded to a server while XR headsets handle real-time spatial interactions. This balance enables resource-efficient operation for complex tasks such as object manipulation and environment understanding.

- **Frame Sampling:** Egocentric video frames are sampled at a rate of one frame per second to reduce data transmission and computational load.
- **Frame-by-frame Captioning:** Selected frames are processed by GPT-4V or Video-LLaVA to generate descriptive summaries. These captions highlight key actions, objects, and interactions occurring in each frame.
- **Object Detection:** Ferret, a vision-language model, identifies and localizes objects in the sampled frames. The model returns 2D coordinates of detected objects, which are later projected into a 3D environment using headset-generated world meshes.
- **External Content Integration:** The system enhances its multimodal input by incorporating 3D world mesh data generated locally by the XR headset. Camera pose information and 3D projections of object coordinates ensure spatially accurate AR overlays.

- **Temporal Awareness:** Frames are tagged with timestamps and camera pose information for accurate spatial mapping during subsequent stages.

Through the previous analysis, the key steps were identified. These are summarized in Table 3.2.

Table 3.2: Comparison of steps across different approaches.

Feature	MISAR	VidIL	M3SUM	AVIWT	XaiR
Frame Sampling	✓	✓	✓	✓	✓
Frame-by-frame Captioning	✓	✓	✓	✓	✓
Object Detection	✗	✓	✗	✓	✓
External Content Integration	✓	✗	✗	✓	✓
Temporal Awareness	✗	✓	✓	✗	✓

3.2 Steps in Video Summarization Frameworks

Informed by the feature analysis and the specific needs of real-world scenarios, a draft pipeline for video summarization is shown in Figure 3.1. This pipeline combines the most relevant steps observed across the studied systems.

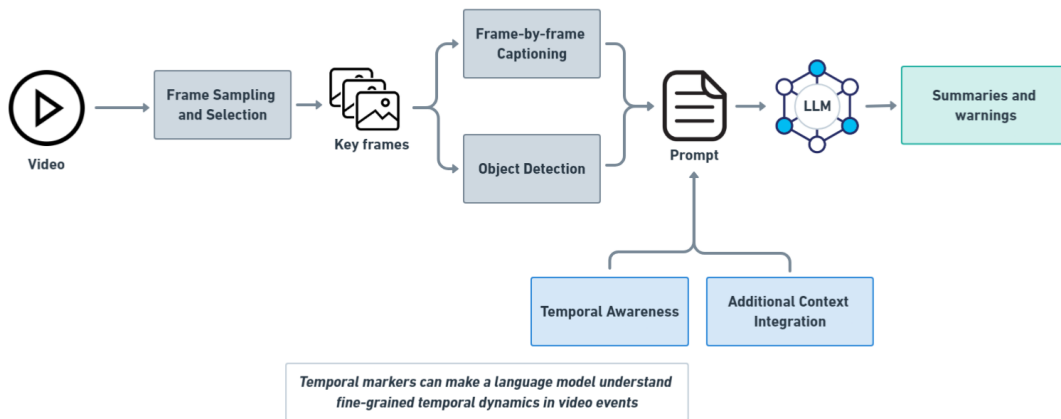


Figure 3.1: Draft pipeline with most common steps in video summarization frameworks.

The video analysis pipeline begins with Frame Sampling, which selects representative frames from the video while maintaining computational efficiency. These sampled frames are then passed through a Captioning process, where detailed captions are generated for each frame. This step is supported by Object Detection, which enhances frame-level understanding by identifying and labeling significant elements within the scene. To capture the dynamics of the video over time, Temporal Awareness is introduced, adding temporal markers that track changes and motion. Context Integration further enriches the pipeline by incorporating metadata or other relevant external inputs, thereby improving overall video comprehension.

The following subsections present the state-of-the-art approaches used in each of these key components across the systems studied.

3.2.1 Frame Sampling

A video consists of a sequence of frames, making both spatial details and temporal relationships essential for accurate recognition. Unlike neatly trimmed clips, untrimmed videos, such as those of people operating machinery, present a greater challenge, as not all frames consistently align with the context. Also, feeding a model with all the frames of a video is a sort of brute force and introduces a much unnecessary computation burden [50]. Therefore, the challenge of identifying relevant frames, clips, or segments within a video has been explored for various applications. For instance, video summarization involves producing a significantly shorter version of the original video by combining a select set of snippets that capture the most informative or thrilling moments [24]. That's where sampling (or frame selection [13]) comes in. Sampling involves selecting a subset of the more "important" frames from a video. This process is crucial because video data volumes are typically too large to process in full, making it necessary to work with a smaller, representative subset of data [18]. Various sampling strategies are discussed in the literature [18].

AdaFrame [20] combines Reinforcement Learning (RL) with a Long Short-Term Memory (LSTM), enabling context-aware frame selection. For each frame, it decides if the frame helps in identifying what's happening. It uses the LSTM to keep track of the context so it can make smarter decisions about what frames to view next. The agent calculates if it's worth looking at more frames or if it has enough information already.

FastForward [11] adopts an end-to-end RL setup with two sub-networks: an adaptive stop network, which decides whether to continue or halt frame sampling, and a fast-forward network, which includes several actions for moving backward or forward by various time intervals. For example, it may start skipping parts with little action or go back to replay important moments. The RL agent learns to skim through the video efficiently.

FrameGlimpse [53] operates on the idea that action detection relies on iterative observation and refinement. Their method uses an RNN-based agent to select where to look next. Based on its confidence level in the current frame, the agent may emit a prediction or choose to observe additional frames.

Multi-agent RL [50] treats frame sampling as multiple, parallel Markov decision processes (a fancy way of saying each action depends only on the current state, not the whole history), where agents work together, refining frame selection over time. A context-aware observation network models context between agents and historical agent states, while a policy network provides a probability distribution over possible actions.

SCSampler [24] offers a lightweight clip-sampling approach, selecting the most salient temporal clips directly from compressed video features and incorporating audio data for context. This approach is well-suited for untrimmed videos but has also shown results with the Kinetics dataset, where audio is available as an extra input.

Attention Aware Sampling [10] employed an RL agent with attention mechanisms to discard irrelevant frames, treating the frame selection as a Markov decision process and training the agent without additional labels.

Although these methods have shown strong performance, they primarily address untrimmed videos, which contain substantial unnecessary data that is easier to filter out. However, SCSampler has also been tested on trimmed datasets such as Kinetics.

3.2.2 Frame-by-frame Captioning

Translating visual content into natural language descriptions is a fundamental component in computer vision, with diverse practical consequences [51]. Recent advancements in MLLMs have introduced innovative approaches to frame-by-frame captioning. These methods are not only capable of generating detailed captions but also incorporate temporal and multi-modal awareness, making them highly relevant to scenarios where understanding sequential events is needed.

MISAR [4] approaches the frame captioning step through a two-stage process that combines a pre-trained video-to-text model with an LLM for refinement. Initially, they use the LaViLa model, which has been pre-trained on the Ego4D dataset, to convert egocentric video footage into textual descriptions. This model generates a sentence approximately every eight frames of video, aligning with human perceptual latency to ensure the system can operate in real-time, particularly within the constraints of physical augmented reality environments. However, the authors recognize that captions generated by LaViLa can be noisy or lack sufficient contextual grounding. To address this, they incorporate GPT-3.5-turbo to refine the raw captions. By integrating contextual information such as recipe instructions and the specific task environment, GPT-3.5-turbo enhances the informativeness of the generated text. This dual-step captioning enables the system to deliver context-aware descriptions without requiring additional domain-specific training.

In VidIL [49], the authors begin by performing sparse frame sampling. Each sampled frame is then passed through a pre-trained image-language model, namely BLIP, which generates descriptive frame-level captions. The BLIP model is not just used for captioning but also incorporates a filtering mechanism to ensure that only relevant or high-confidence captions are retained. This process captures static visual semantics but does not yet model the temporal relationships between events across frames. To address this, the authors further enhance the frame captions by integrating them into a temporal-aware prompt ("First," "Then," and "Finally,") so that a downstream model can interpret and reason over the sequence of frame-level semantics. Thus, the frame captioning in VidIL serves as a crucial mid-level step: it transforms raw visual input into coherent, temporally ordered text that can be composed - along with object, event, and attribute descriptors - into prompts for higher-level video understanding tasks such as captioning and question answering.

In the XaiR system [42], the frame captioning process serves as a foundational step in the automatic generation of task instructions from egocentric video data. To initiate this process, a user wears a head-mounted device and performs a given task while the system records a continuous video stream. This video is then segmented into discrete 10-second intervals, and one image frame per second is sampled from each segment to represent the temporal dynamics of the activity. These frames are subsequently input into an MLLM, such as GPT-4V or Video-LLaVA, along with a structured prompt designed to elicit a concise summary of the depicted actions. The prompt contextualizes the frames as egocentric visual input and explicitly requests a two-sentence description of the activity occurring within the segment. The MLLM processes this input and generates a textual summary that encapsulates the key actions observed during that portion of the video.

The AVIWT framework [3] begins by sampling representative frames the video. The selected frames are then passed through a pre-trained BLIP-2 model to generate captions. BLIP-2 is prompted in two distinct ways: one prompt asks for a general caption describing what is happening in the image, while the other focuses on identifying the objects present in it. These dual prompts ensure that both the overall scene and specific visual elements are captured

in the frame-level textual descriptions. The resulting captions contribute to a multimodal textual representation of the video, which also includes audio transcriptions (from automatic speech recognition) and on-screen text (via OCR). These elements are concatenated and used to prompt an LLM, such as GPT-3.5, Flan-T5, or Vicuna, which generates a coherent, narrative-style story that verbalizes the entire video.

Captioning Models

The techniques previously mentioned use various MLLMs for frame captioning. Here, we detail some of the most recent and powerful.

GPT-4o [32] by OpenAI, introduced as an "omni model", is designed to handle a combination of text, audio and image inputs, and generate text, audio, and image outputs within a single, unified neural network. Its end-to-end training across different modalities allows for more natural human-computer interaction. For video understanding, its ability to process sequences of frames enables it to perform video captioning with a deeper understanding of the temporal context.

*GPT-4.1*¹ by OpenAI focuses on enhancing performance for real-world developer applications, with significant gains in instruction following. While building on the multimodal capabilities of its predecessors, GPT-4.1 offers a much larger context window of up to 1 million tokens, which is particularly beneficial for analyzing long video sequences where extended context is crucial for accurate captioning and summarization. It also includes improved long-context comprehension, allowing it to more effectively utilize the information within this expanded window.

*Gemini 2.0 Flash*² by Google is optimized for high-volume, cost-efficient tasks that require fast performance. It is an MLLM that can process text, code, images, audio, and video to generate text-based outputs. This makes it well-suited for applications that need to quickly generate captions or descriptions for a large number of video frames. While it is a more lightweight model, it still benefits from the advanced reasoning and multimodal understanding capabilities of the broader Gemini family.

Gemma 3 [44] family of models by Google introduces vision understanding capabilities to the lightweight, open-model Gemma series. Ranging from 1 to 27 billion parameters, these models are designed to run on a variety of hardware, from mobile devices to high-end Graphics Processing Units (GPUs). The multimodal versions of Gemma 3 integrate a SigLIP [56] vision encoder, enabling them to process image inputs and generate descriptive text. This makes them a strong candidate for frame-by-frame captioning tasks.

*LLaVA*³ is an open-source MLLM built by fine-tuning LLaMA with vision encoders. It is capable of producing high-quality image-based captions and answering image-related questions. Though not the most advanced in vision fidelity, LLaVA provides good performance in controlled environments, such as industrial simulations, making it ideal for academic or low-resource use.

¹<https://openai.com/index/gpt-4-1/> (accessed 2025-06-29)

²<https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-0-flash> (accessed 2025-06-29)

³<https://llava-v1.github.io/> (accessed 2025-06-29)

3.2.3 Object Detection

While certain state-of-the-art models offer end-to-end frame captioning, integrating object detection as a distinct step in the pipeline may bring several advantages. Object detection stands as one of the most essential and challenging tasks in computer vision [23]. Its main goal is to identify visual objects from specific classes, such as machines, tools, and humans, locate them with bounding boxes, and categorize them according to their respective classes [60]. The evolution of object detection can generally be divided into two phases: before 2014, when traditional methods were predominant, and post-2014, when deep learning-based approaches became the standard [60]. The architectures from each phase differ significantly in terms of accuracy, speed, and hardware requirements. Compared to traditional techniques, Convolutional Neural Networks (CNNs) offer a more advanced architecture and are considerably more expressive in capturing complex patterns [39].

You Only Look Once (YOLO) YOLO is among one-stage object detection methods. The algorithm is recognized as one of the leading due to its remarkable balance of speed and accuracy. Its wide-ranging applications across various fields and industries have made it an essential tool in modern object detection tasks [19]. Each new version has architectural improvements and optimizations that have established YOLO as a leading model in one-stage object detection. The latest iterations of YOLO, including versions YOLOv8 through YOLO11, are the vanguard of the model's advancements.

YOLOv8 [40] bring in semantic segmentation, enabling pixel-level classification, alongside scalable model versions that adapt to diverse environments, from resource-limited devices to high-performance systems. It also expanded functionality to tasks like pose estimation, image classification, and oriented object detection.

Building on these enhancements, YOLOv9 [47] added Generalized Efficient Layer Aggregation Network (GELAN) to enhance layer connectivity, further boosting model efficiency and accuracy, and Programmable Gradient Information (PGI) to optimize gradient flow during training.

Developed by Tsinghua University, YOLOv10 [46] took a unique approach by eliminating Non-Maximum Suppression (NMS) in favor of a dual-assignment strategy, improving object localization without redundant bounding boxes. It also introduced innovations like spatial-channel decoupled downsampling, and a rank-guided block design, making it exceptionally efficient.

The latest version, YOLO11 by Ultralytics [45], retains the versatility of YOLOv8 in pose estimation, instance segmentation, and oriented object detection. YOLO11 replaced the C2f block with the C3k2 block to enhance performance while maintaining speed. Additionally, the new Cross Stage Partial with Spatial Attention (C2PSA) module improves spatial attention, especially beneficial for detecting small and overlapping objects, thus boosting accuracy.

Each of these improvements underscores the rapid evolution of the YOLO architecture, cementing its role across various applications.

3.2.4 External Content Integration

The creation of effective summarization models for all scenarios is blocked by the diversity in user expectations and what they consider a good summary. Overcoming these issues calls for persistent research and innovative approaches, prioritizing advancements in model design, training strategies, and evaluation [26]. Considering this, it is important to explore

methods of incorporating domain-specific context into downstream models to increase the quality of summaries. Some approaches are presented below.

Pre-training

Pre-training serves as a foundational approach that typically works alongside fine-tuning for downstream applications. In this process, model parameters are trained on predefined tasks to learn key characteristics, structural patterns, and contextual information. These pre-trained features support downstream tasks by offering rich information and accelerating the model's convergence [59]. There are five types of pre-training methods [22].

Pre-training from Scratch (PTS) involves training transformer-based language models like BERT [9], ELECTRA [6], and T5 [33] from the ground up on large volumes of unlabeled text. These models typically include embedding layers, transformer encoder and/or decoder layers, all of which are initialized randomly and trained during pre-training to minimize losses from tasks like Masked Language Modeling and Next Sentence Prediction. PTS is computationally intensive and demands significant resources, such as multiple GPU.

Continual Pre-training (CPT) initializes a model from existing pre-trained models and adapts it to specific domains or tasks by further pre-training. It avoids the computational cost of training from scratch, making it more efficient. However, it struggles with domain-specific vocabulary, leading to suboptimal performance when the domain contains many unique terms. A solution is incorporating new vocabulary during pre-training, as seen in models like RuBERT [25] and PortugueseBERT [41].

Simultaneous Pre-training (SPT) is designed for domains with limited domain-specific text. It trains models using a mix of general and domain-specific text, ensuring a balanced vocabulary through upsampling of domain-specific data. This method prevents overfitting while adapting models to specialized domains. SPT is particularly useful for resource-limited domains and languages where large datasets are unavailable.

Task Adaptive Pre-training (TAPT) focuses on task-specific adaptation by pre-training models on small amounts of task-related unlabeled text. This approach is less expensive and complements other methods like PTS or CPT, enhancing performance further when combined. It uses lightweight techniques like VAMPIRE [14] to extract additional data.

Knowledge Inherited Pre-training (KIPT) combines self-supervised learning and knowledge distillation to train models, reusing knowledge from existing pre-trained models (teacher models). Unlike traditional distillation, KIPT often involves a larger student model that learns both from the teacher and the pre-training corpus. This dual learning process accelerates convergence and improves performance while reducing training costs compared to self-supervised learning alone. KIPT supports lifelong learning and knowledge transfer, with CPM-2 [57] as a notable example of its application.

While pre-training methods provide a robust foundation for downstream tasks, they are inherently resource-intensive, requiring substantial computational power, extensive data, and significant time investments. Fortunately, there are more alternatives.

Fine-Tuning

Improving the performance of LLMs on unfamiliar tasks and datasets often relies on fine-tuning. However, as models grow in size (1.5 billion parameters in GPT-2 to 175 billion in GPT-3) traditional fine-tuning approaches become increasingly impractical [15]. These

methods demand enormous computational resources, such as thousands of GPUs running in parallel, making them inefficient and unsustainable. To overcome this challenge, researchers have embraced a smarter alternative: tweaking only a small fraction of the model's parameters while leaving the rest untouched. This approach is known as Parameter-Efficient Fine-Tuning (PEFT). It minimizes computational demands without sacrificing performance by focusing on the parts of the model that matter most for adaptation [15]. Similar breakthroughs have been seen in the world of vision and multimodal models. These large-scale, pre-trained models excel at transferring their learned knowledge from massive datasets to smaller or specialized ones. This cross-domain flexibility has made PEFT techniques increasingly appealing to researchers across disciplines [15]. PEFT methods generally fall into four categories: additive, selective, reparameterized, and hybrid [15].

Additive Fine-Tuning adds tunable components to the existing model structure, such as adapters or soft prompts. These additions modify the model's behavior without altering its core parameters.

Selective Fine-Tuning, instead of adding new components, carefully selects a subset of the model's parameters to adjust while leaving the majority unchanged. It can be further divided into two types. Unstructured Masking: chooses parameters individually; and Structured Masking: selects groups of parameters in a more organized way.

Reparameterized Fine-Tuning reimagines the model's parameters by introducing low-rank, trainable components during training. Once the model is fine-tuned, these components integrate seamlessly into the original structure. Two popular strategies in this category are: Low-rank Decomposition and Low-Rank Adaptation (LoRA) [17].

Hybrid Fine-Tuning combines elements from different PEFT approaches, using their unique strengths to optimize results.

Retrieval-Augmented Generation (RAG)

This method improves machine learning models by retrieving relevant information from external knowledge bases using semantic similarity. This approach helps reduce factual inaccuracies in generated content [12]. RAG's integration into machine learning models has led to its widespread use, making it a pivotal technology for chatbots and other real-world applications. Research on RAG focuses on enabling LLMs to handle more complex, knowledge-intensive tasks during inference, driving rapid advancements in the field [12]. The RAG's research progression is categorized into three stages: Naive RAG, Advanced RAG, and Modular RAG [12].

Naive RAG is an early methodology that gained attention with the rise of ChatGPT. Its process - indexing, retrieval, and generation - is often described as "Retrieve-Read". In indexing, raw data in various formats (PDF, HTML, Word) is cleaned, converted to text, and segmented into smaller chunks. The chunks are encoded into vector representations and stored in a vector database. Retrieval involves converting user queries into vector representations, comparing them to stored vectors, and retrieving the most relevant chunks. During generation, these chunks are used to create prompts, enabling the model to generate responses that may also include multi-turn dialogue history. Despite its simplicity, Naive RAG has significant challenges. Retrieval can lack precision, resulting in irrelevant or incomplete information. Generation may lead to hallucinations or bias while integrating retrieved information often results in disjointed outputs. Redundancy and inconsistent tone further reduce response quality.

Advanced RAG improves upon its predecessor with enhancements in indexing, retrieval, and context integration. Pre-retrieval optimization focuses on refining the indexing structure and user queries using methods like query rewriting and metadata incorporation. This ensures better data granularity and alignment. Post-retrieval techniques like re-ranking and context compression streamline the relevance of retrieved content, ensuring that only critical information is used in generation. Tools like *LlamaIndex*⁴ and *LangChain*⁵ exemplify these innovations by addressing information overload and emphasizing key sections.

Modular RAG represents a leap in adaptability and modularity. It introduces specialized components, such as a Search module for advanced searches across diverse data sources, a Memory module for iterative context enhancement, and a Task Adapter module for task-specific retriever customization. Additionally, Modular RAG introduces new patterns like Rewrite-Retrieve-Read, which refines query generation, and Generate-Read, where LLMs produce content directly. Flexible module configurations (e.g., Demonstrate-Search-Predict) enhance interaction between components. Its ability to incorporate fine-tuning and RL makes it highly versatile, combining retrieval and adaptation for diverse scenarios.

Prompting

Prompting has been a key technique for leveraging the capabilities of LLMs in various tasks. Prompting directly exploits pre-trained models by framing tasks as natural language instructions, often referred to as prompts, which guide the model to produce desired outputs [27]. This paradigm is particularly powerful because it enables zero-shot and few-shot learning, allowing models to adapt to new tasks with minimal labeled data [27]. The shift to the "pre-train, prompt, and predict" paradigm marks a significant departure from the earlier "pre-train and fine-tune" framework. [27] In this newer paradigm, tasks are reformulated using textual prompts to align with the pre-training objectives of the LLMs. This allows models to generate outputs directly relevant to downstream tasks without additional supervised training [27]. For example, in NLP, prompts can take the form of masked tasks (e.g., "The capital of Portugal is [MASK]") or open-ended questions (e.g., "What is the capital of Portugal?"). Studies have shown that the effectiveness of prompting depends heavily on the design of the prompts themselves. Specifically, open-ended prompts often outperform restrictive prompts in diverse tasks, as they align more closely with the probabilistic text-generation capabilities of LLMs [1].

Despite its advantages, prompting is not without challenges. Small variations in prompt wording can lead to significant differences in model performance, a phenomenon known as "prompt brittleness" [1]. Additionally, creating effective prompts often requires substantial manual effort and domain expertise. To address this, techniques such as automated prompt generation [21] and aggregation of multiple imperfect prompts [29] have been explored. These methods aim to reduce the reliance on human intervention while improving model reliability [1].

Zero-Shot, One-Shot, and Few-Shot Learning Prompting is closely linked to the concepts of zero-shot, one-shot, and few-shot learning, which describe how effectively a model can generalize to new tasks with varying amounts of task-specific examples.

⁴<https://www.llamaindex.ai/> (accessed 2025-06-29)

⁵<https://www.langchain.com/> (accessed 2025-06-29)

Zero-shot learning refers to the ability of a model to perform a task it has never seen before using only a natural language description of the task or a relevant prompt. This capability arises from the extensive pre-training of machine learning models on diverse and massive datasets, enabling them to transfer knowledge from related contexts to unseen tasks. For instance, an LLM can answer questions about an unseen topic simply by being prompted with a well-crafted question template [27, 1].

One-shot learning extends this paradigm by providing the model with a single example of the task. The model uses this example, often presented within the prompt, as a guide to adapt its behavior. For example, to classify text, the prompt might include a single labeled example along with the query text to signal the desired format and reasoning process [27].

Few-shot learning builds on this idea by including a small number of labeled examples (typically fewer than 10) within the prompt. These examples serve to further clarify the task and improve the model's performance by providing more context. Research has shown that as the number of examples increases, the accuracy of predictions improves, although the gains diminish after a certain point [1]. Few-shot learning strikes a balance between the zero-shot scenario, which requires no task-specific data, and fully supervised learning, which demands extensive labeled datasets.

3.2.5 Temporal Awareness

Unlike static images, which capture objects, attributes, and events in a single moment, videos add a temporal dimension that reveals changes in the state of objects, actions, and events over time. Therefore, to accurately support video-level descriptions and queries, it's essential to capture both the content and the sequential flow of information over time [49].

Wang et al. [49] introduced a technique that uses a temporal-aware hierarchical representation and innovative prompting strategies. The videos are decomposed into three semantic levels to handle temporal aspects effectively. They perform a structure-aware visual tokenization, extracting fine-grained details such as objects, events, and attributes from sampled frames. Each frame undergoes caption generation. The captions provide a snapshot of individual moments in the video. While static in nature, this level captures granular temporal points, forming a foundation for temporal reasoning when aggregated. This level aggregates the visual tokens and frame captions into a single representation, incorporating temporal markers to encode the sequence and progression of events. These markers introduce explicit temporal ordering, enabling the language model to reason over the flow of changes in the video. At the core of this method is a temporal-aware few-shot prompt that instructs a pre-trained language model to understand and describe the video. The prompt explicitly encodes the order of events and states with natural language phrases like "First" "Then" and "Finally". By doing this, the model is guided to recognize the sequential dependencies among frames, events, and actions, avoiding the pitfalls of treating videos as static sequences. Few-shot examples are selected and structured to emphasize the temporal changes captured in the hierarchical representation. The temporal-aware prompt distinguishes scenarios with different temporal orders (e.g., sunset vs. sunrise) by correctly interpreting the sequence of visual elements.

Wang et al. [48] introduced a novel approach to video summarization by using textual transformations and the reasoning abilities of LLMs to address temporal challenges in unsupervised video summarization. The key of their method is converting multimodal video content into textual representations to unify different modalities (audio, visual frames, etc.). Each frame is processed through an off-the-shelf image-to-caption model, converting visual information

into descriptive text captions. Audio is transcribed using a speech-to-text model, resulting in text transcripts with associated timestamps for future alignment. Unifying all modalities simplifies multimodal fusion while retaining key temporal and semantic information. The framework aligns information from textual descriptions of frames and audio using an algorithm that ensures temporal coherence. Then, framework selects the most relevant frames using a knapsack algorithm to generate a concise summary. Because the start and end times are passed to the algorithm, it preserves the temporal order of the video.

Srinidhi, Lu, and Rowe [42] addressed temporal awareness in several ways, focusing on capturing and understanding changes over time within tasks and environments. They use egocentric video streams, breaking these into 10-second segments to analyze and summarize each segment using multimodal models like GPT-4V. This process guarantees that sequences of actions are captured in their correct order. When guiding users in real-time, the system considers two consecutive image frames to provide context about the user's current state and recent actions. This dual-frame approach helps the system determine if an instruction has been completed and whether the user is on track. The prompts sent to the system's language models include the ongoing instruction, the sequence of previous steps, and contextual information from the video frames. The temporal continuity let the models adapt their responses dynamically. To maintain synchronization between physical space and temporal awareness, all data is timestamped. This guarantees that AR content, such as virtual arrows or text instructions, is aligned accurately with the user's position and actions over time.

3.2.6 Summarization

LLMs have been central to the video summarization process, using their NLP capabilities to transform raw data into structured and meaningful outputs. In this project, their primary role is to integrate information captured across all preceding steps, synthesizing it into a cohesive format that facilitates anomaly detection. Several recent approaches demonstrate the complexity of this task.

For instance, Wang et al. [49] utilize a large-scale pre-trained language model, specifically GPT-3, to generate a video-level text output. This is accomplished through a temporal-aware few-shot prompting strategy. The prompt is constructed with three main components: an instruction that defines the task, a few-shot context containing examples, and a task query. To ensure the language model accounts for the progression of events, textual representations from the frame and visual token levels are aggregated and ordered chronologically, prefixed with temporal markers. This structured prompt guides the model to generate a coherent summary without requiring any fine-tuning.

Similarly, Bhattacharya et al. [3] combine information from multiple modalities, including video metadata, OCR text, scene captions, and audio transcripts. This aggregated textual data is used to prompt a generative LLM (e.g., GPT-3.5, Flan-t5, or Vicuna) with instructions to write a continuous narrative based only on the provided information. To manage the input size, especially for longer videos, the authors employ efficiency techniques such as removing redundant frame captions and OCR outputs.

In a more specialized application, Srinidhi, Lu, and Rowe [42] use an MLLM like GPT-4V to first generate summaries of an expert's actions in 10-second video batches. These individual text descriptions are then collected and passed to GPT-4, which is prompted to synthesize them into a holistic, step-by-step instructional guide for non-experts.

Reasoning Models

These methodologies highlight an important point: modern video summarization is not merely a task of text aggregation but one of sophisticated reasoning. The model must infer temporal relationships, synthesize information from disparate sources, discern and filter redundancy, and abstract specific events into a general narrative or a structured guide. The quality of the final output - be it a caption, a story, or a set of instructions - is therefore directly proportional to the reasoning capability of the language model employed. Consequently, selecting a summarization model for this project cannot be based on its linguistic fluency alone; it must be based on its proven ability to perform complex reasoning. To identify the most suitable models, we turn to benchmarks designed specifically to evaluate these advanced cognitive skills. The Graduate-Level Google-Proof Q&A Benchmark (GGPB), for example, assesses models on questions that require deep reasoning rather than simple information retrieval. The leaderboard from this benchmark, shown in Figure 3.2, provides an empirical ranking of state-of-the-art models on such tasks.

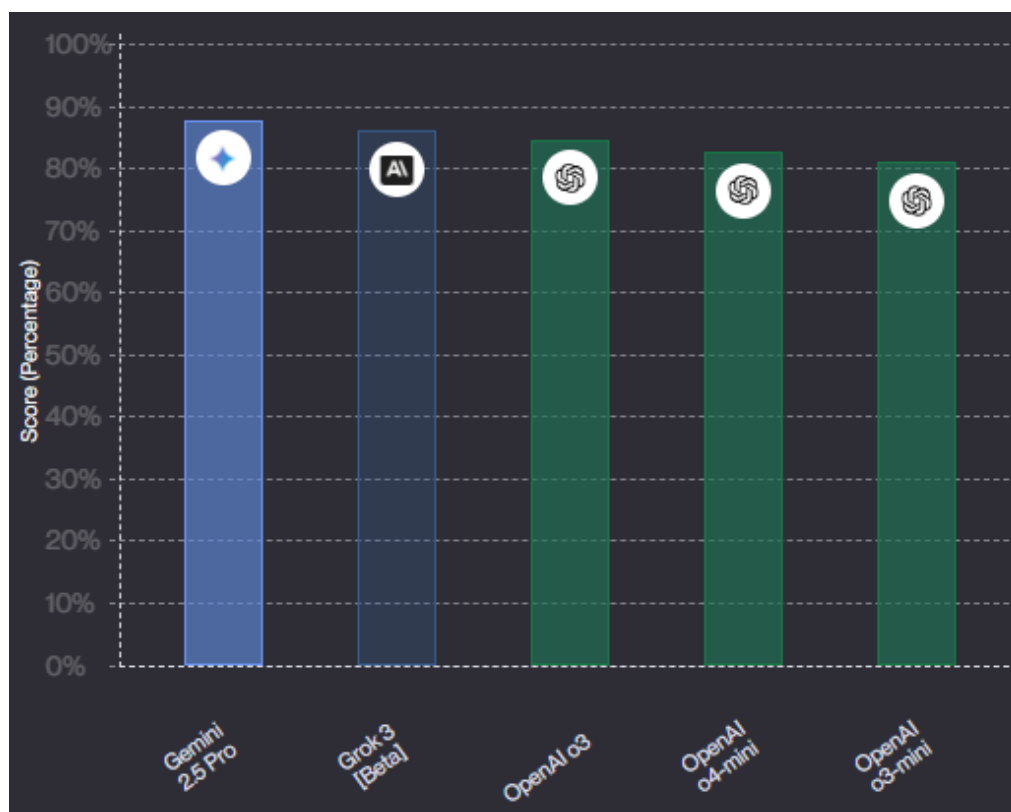


Figure 3.2: Best reasoning models chart, according to the Graduate-Level Google-Proof Q&A Benchmark (GGPB), by vellum.ai.

As the figure illustrates, models like Gemini 2.5 Pro and OpenAI's o4-mini lead the rankings, demonstrating superior performance on tasks that demand advanced reasoning. "This benchmark data provides a strong rationale for selecting a top-tier reasoning model, whether an LLM or an MLLM."

3.3 Datasets

For production lines to effectively implement automatic quality control, the AI models must use datasets that accurately reflect real-world conditions and operations [37]. While deep learning has made significant advances in certain specialized areas, such as medical imaging for cancer detection, many industrial and technical applications remain underserved [37]. The challenge lies in the limited availability of datasets across various technical domains - from life sciences to manufacturing and industrial processes. This data scarcity has created a notable gap in the practical implementation of AI solutions for many specialized technical applications [37].

3.3.1 Candidate Datasets

While vision datasets for industrial applications are scarce, the following options, gathered in Section 2.2, offer valuable features and can serve as useful complements.

The InHARD dataset [7] is designed for industrial human action recognition, particularly within the domain of human-robot collaboration. It is notable for its emphasis on real-world industrial environments. The dataset provides RGB video files with a resolution of 1280x720 at 30 frames per second, as well as skeletal data stored in the BioVision Hierarchy (BVH) format. The dataset includes two main modalities. The video modality comprises recordings from three different camera angles - top, left-side, and right-side views - which are combined into a single RGB video stream. The skeletal modality captures data at 120 Hz using a motion sensor that tracks 17 major body joints, offering 3D positional and rotational information.

The HA4M dataset [5] is a multimodal dataset created to support research in human action recognition within industrial assembly contexts. It focuses on monitoring the assembly of an Epicyclic Gear Train by 41 subjects using a Microsoft Azure Kinect camera. The dataset provides six synchronized data modalities: RGB images (2048x1536), aligned RGB-depth images, depth maps, infrared frames, 3D point clouds, and full-body skeleton data comprising 32 joints with position and orientation information. The camera records at 30 FPS from a fixed perspective in a controlled laboratory environment simulating industrial assembly tasks. Participants performed 217 assembly trials following flexible action sequences comprising 12 distinct labeled actions. The dataset includes frame-by-frame annotations and supports fine-grained temporal action segmentation.

Assembly101 [38] is a large-scale multi-view video dataset curated for procedural activity understanding, specifically in the domain of toy vehicle assembly and disassembly. It features 4321 videos totaling 513 hours of footage, capturing 362 unique assembly sequences performed by 53 participants. The dataset includes recordings from 8 static and 4 egocentric RGB cameras, enabling simultaneous multi-view analysis. Assembly101 offers more than 1 million fine-grained and 100,000 coarse action segments, annotated with start and end times, as well as over 18 million 3D hand poses derived from egocentric views. The recording setup simulates realistic, goal-oriented assembly tasks without fixed instructions, introducing natural variation, mistakes, and corrections in procedural actions.

MECCANO [34] is a first-person, multimodal dataset focused on human behavior understanding in industrial-like scenarios. Participants were recorded building a toy motorbike using real tools and components while wearing a custom headset equipped with an RGB camera, depth sensor, and gaze tracker. The dataset includes 20 RGB videos (1920x1080 at 12 fps), corresponding depth maps (640x480 at 12 fps), and gaze data sampled at 200

Hz. Rich annotations are provided for multiple vision tasks including action recognition, egocentric human-object interaction detection, action anticipation, and next-active object prediction.

The HA-ViD dataset [58] is a comprehensive video dataset curated to advance the understanding of procedural knowledge in industrial assembly contexts. It focuses on capturing realistic human assembly processes with fine-grained annotations to support both human-robot collaboration research and vision-language understanding. The dataset records the assembly of a Generic Assembly Box by 30 participants. The dataset provides 3222 multi-view videos (side, front, top views) recorded at 1280×720 resolution with synchronized depth data. Annotations include 96K temporal labels and nearly 2M spatial labels detailing subject, action verb, manipulated object, target object, and tool. Additionally, it captures two-handed collaboration status, pauses, and errors.

HUMM [52] is a video dataset designed to facilitate research on human movement and trajectory prediction within smart manufacturing environments. It addresses the gap in available datasets for analyzing human motion dynamics in factory contexts, with a particular focus on supporting human-robot collaboration. The dataset comprises 651 video clips captured under two different detection perspectives: Fixed Detection Perspective (FDP) and First-Person Perspective (FPP). The videos were recorded in a controlled experimental setup simulating a small manufacturing workshop, with participants performing both collaborative and independent workflows. The FDP videos were recorded using static IP cameras at 1920×1080 resolution at 30 fps, while FPP videos were captured using a wearable PTZ camera at 1920×1080 resolution at 60 fps.

Dataset Analysis for Captioning Assessment

To ensure datasets were suitable for our captioning models assessment (Section 4.3), we established specific criteria. These criteria guided our review through the datasets gathered, helping us select the one that could robustly support our assessment. They are detailed as follows:

- Industrial contexts: The setting should be industrial or manufacturing environments.
- Human-machine interactions: Data should capture interactions between people and machines.
- Not egocentric: The data should be from a fixed perspective, not a first-person point of view.
- Open and free: The dataset must be publicly available and free to use.
- Image-question-answer pairs: The dataset needs to include images with corresponding questions and answers (IQA triplets), going beyond simple image-class pairs.

The last criterion appears from the nature of our pipeline's downstream task, which involves interpreting visual scenes through the lens of task-specific, language-based queries typical of industrial contexts. Rather than relying on generic descriptions (e.g., "What is in the image?"), which risk eliciting broad or biased responses, IQA triplets allow us to probe model performance on targeted relevant tasks.

With the establishment of these criteria, the candidate datasets were reviewed, and their attributes were benchmarked. Table 3.3 summarizes this analysis.

Table 3.3: Evaluation of datasets against project-specific criteria.

Dataset	Industrial Contexts	Human-Machine Interactions	Not Egocentric	Open and Free	IQA Pairs
InHARD [7]	✓	✓	✓	✓	✗
HA4M [5]	✓	✓	✓	✓	✗
Assembly101 [38]	✓	✓	✓	✓	✗
MECCANO [34]	✓	✓	✗	✓	✗
HA-ViD [58]	✓	✓	✓	✓	✗
HUMM [52]	✓	✓	✗	✓	✗

Unfortunately, the comprehensive analysis confirmed that no publicly available dataset fully met our specific criteria, particularly regarding the IQA criterion. As a result, we constructed our own dataset for evaluating captioning models. It is present on Subsection 4.3.1.

3.3.2 Video Games

Industrial simulation games can provide another valuable source of data, as they often incorporate realistic machinery operations and professional workflows. The *Milling Machine Simulator*⁶, available on Steam, offers detailed milling operations, including tool selection, and workpiece setup. A print screen of the game is shown on Figure 3.3.



Figure 3.3: Print screen of the game "Milling Machine Simulator".

Similarly, the *Lathe Machine Simulator*⁷ (Figure 3.4) provides hands-on experience with lathe operations, featuring realistic turning processes and cutting parameter selection.

⁶<https://ui-games.com/milling-machine-simulation/> (accessed 2025-06-29)

⁷<https://ui-games.com/lathe-machine-simulation/> (accessed 2025-06-29)

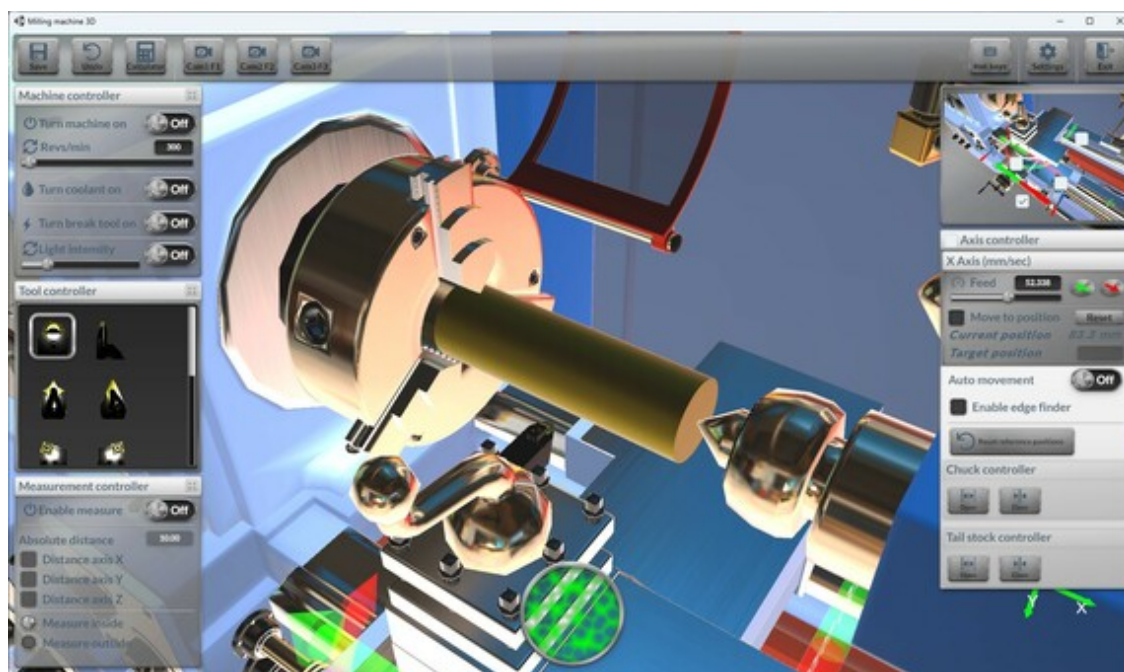


Figure 3.4: Print screen of the game "Lathe Machine Simulator".

These games incorporate authentic machine interfaces and physics-based material behavior. They can provide accessible, controllable, and repeatable environments that emulate real-world industrial operations.

3.4 Evaluation Methods and Metrics

Evaluating the performance of text-generating systems - such as captioners and summarizers - requires reliable and scalable methods. While human evaluation offers high-quality insights, it is limited by cost and scalability. Therefore, automated approaches are commonly used to approximate human judgment along dimensions like fluency, coherence, relevance, and factual accuracy [36].

We categorize evaluation approaches into four main types: human evaluation, lexical-based metrics, embedding-based metrics, and model-based evaluation. Each has its own set of strengths, limitations, and typical use cases.

3.4.1 Human Evaluation

Human judgment remains the gold standard. Domain experts assess clarity, accuracy, and relevance directly. However, manual review is labor-intensive and difficult to scale for large datasets or rapid iterations. In the case studies, human evaluation was primarily used to validate automated metrics.

3.4.2 Lexical-Based Metrics

Lexical metrics compare generated texts to reference outputs based on surface-level overlap. These metrics are fast and widely adopted but often penalize valid paraphrases, missing semantic equivalence.

- BLEU measures n-gram precision by computing the proportion of n-grams in the generated text that appear in one or more reference texts. It penalizes overly short outputs using a brevity penalty. BLEU is sensitive to exact word matches, making it effective for structured tasks but less tolerant of valid paraphrasing. It was used in VidIL [49], M3SUM [48], and AVIWT [3] to assess linguistic overlap between generated and reference captions or summaries.
- ROUGE-L focuses on the longest common subsequence between a candidate and a reference text. Unlike BLEU, which measures precision, ROUGE-L places more emphasis on recall and the preservation of word order and sentence structure. This makes it suitable for evaluating content coverage in summarization tasks. It was applied in both VidIL [49] and AVIWT [3] to capture structural similarity.
- METEOR improves upon BLEU by aligning words based on exact matches, stemmed forms, synonyms, and paraphrases using WordNet. It combines both precision and recall with a harmonic mean and includes a fragmentation penalty to reward coherent word sequences. It was used in VidIL [49] and AVIWT [3] to better reflect human-like similarity judgments.
- CIDEr evaluates the informativeness of a generated caption by computing a cosine similarity between the n-grams in the candidate and a set of reference captions. It emphasizes n-grams that are common in the references but rare overall, thus rewarding salient, relevant content. This metric was used in both VidIL [49] and AVIWT [3] to assess how well generated captions aligned with human descriptions.
- F1 Score is the harmonic mean of precision and recall. In M3SUM [48], it was used to evaluate keyframe selection accuracy by comparing predicted keyframes to ground-truth annotations in the TVSum and SumMe datasets.

3.4.3 Embedding-Based Metrics

Embedding-based metrics evaluate semantic similarity via vector representations. They capture meaning beyond surface form.

- BERTScore computes contextual similarity using pre-trained embeddings. It was used in M3SUM [48] to assess how well captions and frame descriptions aligned with user queries.
- Perplexity, an intrinsic language modeling metric, measured textual coherence in M3SUM's multimodal summaries [48].

3.4.4 Model-as-a-Judge

An emerging trend uses reasoning models to evaluate generated content. Instead of comparing to reference outputs, models are prompted to assess quality along axes like fluency, relevance, or factuality [55, 28]. This approach addresses key limitations of traditional evaluation methods, such as dependency on scarce, expensive human annotations or brittle automatic metrics like BLEU and ROUGE that fail to capture semantic adequacy or nuance. Since reasoning models can be reused across tasks and domains with minimal reconfiguration, this method reduces the need for task-specific evaluation pipelines and large sets of reference outputs, enabling more scalable and consistent benchmarking across diverse NLP applications.

Chapter 4

Design

This chapter transitions from the theoretical groundwork and literature review to the design of our empirical framework. The following sections detail the systematic approach adopted to develop the pipeline architecture and the candidate captioning models assessment strategy.

4.1 Methodology

The design methodology for this project follows a structured, two-pronged approach, as illustrated in Figure 4.1.

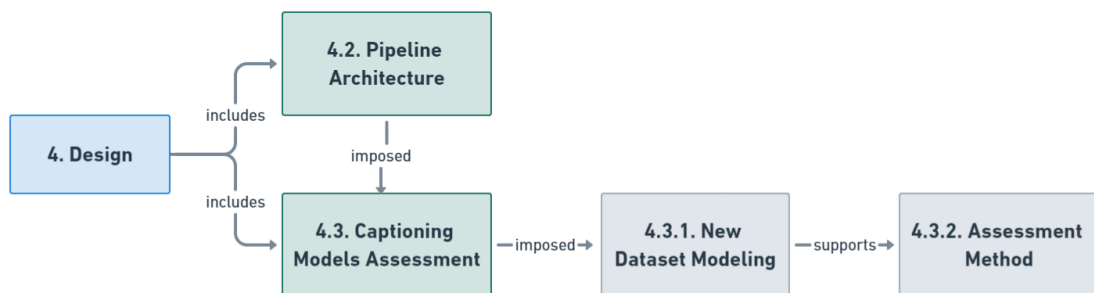


Figure 4.1: The systematic methodology adopted for the design phase of the project. The process includes the parallel development of the pipeline architecture and a required model assessment workflow, which involves a new dataset modeling and the creation of an assessment method.

The primary focus is on the development of the overall pipeline architecture which led to a rigorous captioning models assessment. The assessment of captioning models was identified as a critical prerequisite, as the quality of generated frame descriptions directly impacts the entire system's performance. This generated a systematic sub-process that began with establishing a list of criteria to identify suitable evaluation datasets.

A deeper dataset analysis, made in Subsection 3.3.1, revealed that no existing datasets met the specific needs of this project, particularly the requirement for IQA triplets in an industrial context. Consequently, it was necessary to undertake a new dataset modeling strategy, creating a custom dataset structured around specific categories and capabilities relevant to industrial tasks. This custom dataset forms the backbone of the assessment method to evaluate the performance of different captioning models.

4.2 Pipeline Architecture

The proposed pipeline architecture is a modular framework designed to process video data in a manner that is both computationally efficient and effective for structured summarization. Drawing inspiration from the case studies analyzed in Section 3.1, the system is engineered to handle video in chunks, simulating a real-time scenario. This is achieved through parallel processing of video segments, ensuring that the analysis can keep pace with ongoing operations. The architecture follows the logical flow presented in Figure 4.2 and is composed of several specialized modules, each responsible for a distinct phase.

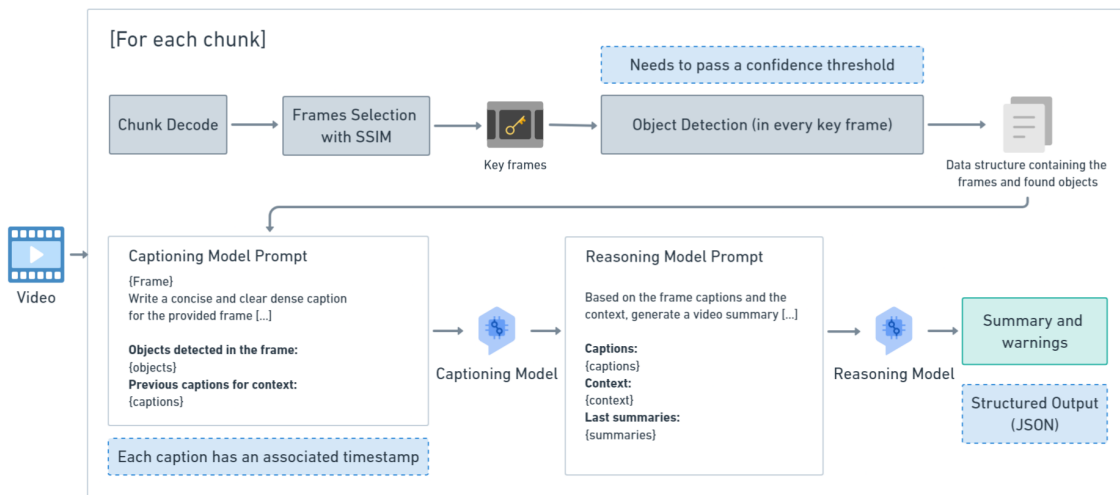


Figure 4.2: Overview of the designed data processing pipeline, illustrating the sequential flow from video ingestion to summarization.

4.2.1 Pipeline Breakdown

The pipeline is deconstructed into its constituent modules, each addressing a distinct stage of the data processing workflow. The following sections detail each component's function in sequential order, from initial video input to the final structured output.

Chunks Decode and Frames Selection

The pipeline begins by ingesting a video file and decoding it into manageable segments. Instead of processing the video as a single monolithic entity, it is divided into chunks. This segmentation is crucial for managing memory and enabling parallel processing. Within each chunk, a content-aware frame sampling strategy is employed to select only the most informative frames: we call them "keyframes". This addresses the challenge of computational burden associated with processing every single frame of a video. The selection is not based on a fixed interval; instead, it utilizes the Structural Similarity Index Measure (SSIM) [31]. A frame is selected as a keyframe only if its visual content differs significantly from the previously selected keyframe, determined by an SSIM score falling below a threshold. This method ensures that redundant frames are discarded while frames capturing significant visual changes - such as a worker performing a new action or a change in machine state - are retained for analysis. Each selected keyframe is stored along with its corresponding timestamp to preserve the temporal order of events.

Chunks Duration Discussion When designing the pipeline, a critical parameter is the duration of the video chunks. The choice of chunk size involves a trade-off between real-time responsiveness and contextual understanding. Shorter chunks, such as five to 10 seconds, offer lower latency since less data needs to be buffered and processed at once. This makes them ideal for monitoring rapid actions where immediate feedback is crucial. Additionally, memory requirements per chunk are lower. However, the downside of shorter chunks is that they may cut off longer, more complex actions or processes, leading to a loss of temporal context. Models might struggle to understand the overall goal of a sequence of actions if it only sees small fragments. Longer chunks, ranging from 60 to 120 seconds, provide the summarization model with a more complete view of an event or task. This allows for a better understanding of temporal relationships between actions, which is essential for summarizing processes that unfold over a longer duration, such as a multi-step machining operation. The trade-offs, however, include increased latency and higher memory and processing requirements. In real-time monitoring scenarios, a 120-second delay before a summary or alert is generated might be unacceptable. For this project, a chunk duration of 30 to 60 seconds is proposed as a balanced starting point. This duration is short enough to enable near-real-time processing suitable for many industrial safety and operational monitoring tasks. At the same time, it is long enough to capture most atomic actions (e.g., picking up a tool, checking inside a machine). The optimal duration is ultimately task-dependent and can be tuned as a hyperparameter based on the specific industrial process being analyzed.

Object Detection

Once keyframes are extracted for a chunk, they are passed to a visual analysis module where object detection occurs. A dedicated object detection model is used to identify and classify objects within each keyframe. The decision to use a purpose-built model is supported by advancements in model architectures. As discussed in Subsection 3.2.3, recent object detection models enhance spatial attention, making them especially effective at identifying the small, frequently overlapping objects typical of industrial environments. To ensure high-quality detections, a confidence threshold was applied, filtering out low-confidence predictions. The output is a list of detected object names for each keyframe.

Frame-by-Frame Captioning

After the detection of the objects, each keyframe is processed by an MLLM to generate a descriptive caption. We tested the models talked in 3.2.2 and the results are presented in Chapter 6.

Information Enrichment

To generate contextually rich and temporally coherent captions, the pipeline incorporates an information enrichment "hard coded" module before invoking the MLLM.

Exclusion of RAG While RAG is a powerful technique for enriching model outputs with external knowledge, it was deliberately excluded from the core scope of this project for several reasons. Integrating RAG would introduce significant architectural complexity beyond the current scope. A robust RAG implementation requires curating a domain-specific knowledge base, such as machine operation manuals and safety protocols, which involves sourcing, cleaning, and structuring the data. It also demands advanced retrieval systems, including

setting up and optimizing a vector database capable of efficiently locating relevant document snippets based on the visual context - a task that constitutes a research domain in its own right. Integrating RAG may be a logical next step for enhancing the system, as acknowledged in Chapter 7.

Temporal Awareness

The system maintains a rolling history of the five most recent captions in a data structure. For each new frame to be captioned, this history is included in the prompt, providing the model with immediate temporal context. This allows the MLLM to understand the current frame in the context of preceding events, leading to a more logical and narrative-driven sequence of captions.

Data Integration

The prompt for the captioning model is dynamically constructed. It includes the image itself, the broader context of the video (e.g., "CNC machine operation"), and the recent caption history. The list of objects detected is also prepared for integration into this prompt, allowing the MLLM to ground its description in pre-identified objects.

Final Summarization

The timestamped, descriptive captions generated for a video chunk are aggregated and passed to a final reasoning model for the synthesis phase. Reasoning models are recognized for its advanced ability to process complex information to produce structured textual outputs. The model receives a prompt containing all chronologically ordered captions for the chunk, along with additional context for its analysis. To expand the context window while reducing the number of input tokens, a list of recent summaries is also provided. The model's task is to synthesize this information into a concise summary and identify any deviations from normal operation, which are flagged as "alerts".

Use of MLLM over LLM The selection of a model for the final summarization stage warrants a brief clarification. This step of the pipeline processes purely textual data - the chronologically ordered captions from the preceding stages. While a text-only LLM would be a conventional choice, the implementation may use a state-of-the-art MLLM for this reasoning task. This decision is a pragmatic response to the current landscape of AI models and the project's resource constraints. Some of the market's most advanced and cost-effective reasoning models, such as Google's gemini-2.5-pro and OpenAI's o4-mini, are indeed multi-modal. These models offer sophisticated reasoning at a significantly lower operational cost than many previous-generation, text-focused alternatives. Therefore, employing a modern MLLM was the most logical and economical approach, securing access to a powerful reasoning engine without being constrained by a model's modality.

Structured Output Generation

A core objective of this dissertation is to produce structured video summaries. To enforce this, the final model is constrained to generate its response in a predefined JSON format separated by summary and alerts.

4.3 Captioning Models Assessment

As noted earlier, a critical component of the proposed pipeline is the accurate captioning of video frames, as these captions constitute the primary descriptive input for the downstream summarizer. Consequently, a robust assessment of potential frame captioning models is paramount. To assess their capabilities, particularly in interpreting human-machine interactions within predefined tasks, we adopted an evaluation strategy centered on IQA triplets. This methodology involves providing a model with an image and a pertinent question, then comparing its generated response against a ground-truth answer.

4.3.1 Dataset Modeling

Following the comprehensive analysis made in Subsection 3.3.1, which confirmed that no publicly available dataset met our specific criteria, we constructed a custom dataset based on IQA triplets. The foundation of this dataset is its organization into five principal categories, each chosen to systematically evaluate the nuanced capabilities of models in complex industrial environments.

Categories

The chosen categories were Human-Machine Interaction, Machine State and Identification, Visual Quantification and Estimation, Text and Display Interpretation, and Functional Reasoning. This structure is not arbitrary; it represents a deliberate methodology for breaking down the complex task of industrial video understanding. The hierarchical organization of the dataset, detailing these categories and their associated tasks, is presented in Figure 4.3.

Human-Machine Interaction focuses on how people subtly interact with machines - an area where misunderstandings and inefficiencies often arise [16]. To evaluate a model's understanding of these interactions, it's not enough to detect objects; it must also interpret intent and assess procedural correctness. For instance, distinguishing between a person actively pressing a button versus merely hovering a hand over a control panel is subtle, yet critical. The tasks in this category assess fine-grained recognition and human pose estimation, pushing the model to capture the intricacies of body posture and intent.

Machine State and Identification tests a model's ability to go beyond surface-level observation to determine a machine's operational state. It challenges generalization, as different types of equipment may feature unique controls, indicators, and displays. A robust model must synthesize multiple cues - such as indicator lights, screen text, and component positions - to accurately determine states like "idle" or "error". It also evaluates whether the model can identify machine types and recognize universal symbols, such as warning lights, linking visual recognition with domain knowledge.

Visual Quantification and Estimation explores the model's ability to perform items counting, distances measuring, or process duration estimating, directly from 2D imagery, without depth data or explicit time information. Tasks here test spatial reasoning and numerical inference, such as estimating how far an object is from another or reading a time display.

Text and Display Interpretation follows the fact that modern industrial systems frequently incorporate textual information via digital displays, printed labels, and safety signage. Being able to read and interpret this text is essential for any model operating in such environments. This category assesses the robustness of OCR capabilities in real-world conditions.

Functional Reasoning pushes into higher-order cognitive territory. It evaluates whether a model can interpret not just what is visible, but also why it matters, connecting perception to reasoning. For example, inferring the function of a button being pressed requires visual detection, contextual awareness, and an understanding of standard machinery functions. This category is particularly important for assessing hallucination risk and logical consistency, as it often requires reasoning about information not explicitly shown in the image.

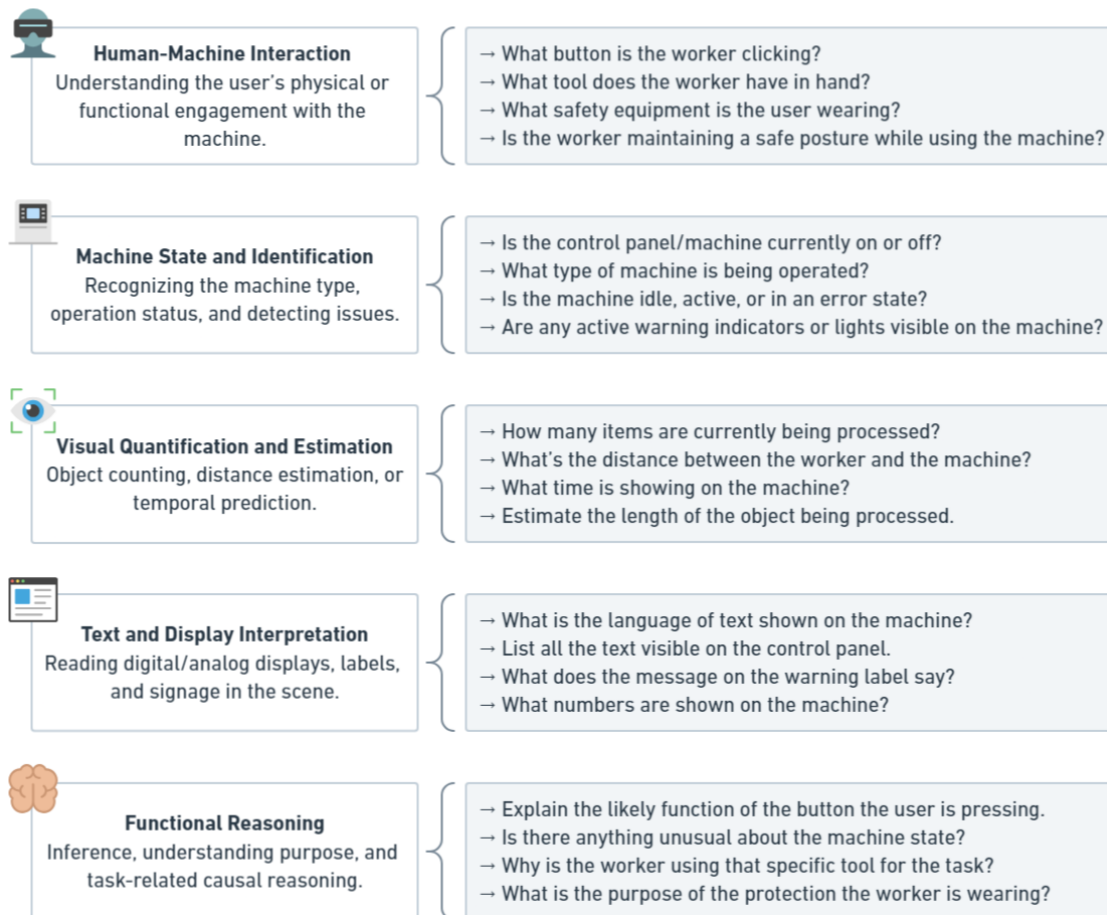


Figure 4.3: Illustration of the various categories and their associated tasks within the proposed dataset.

Capabilities

The purpose behind defining these categories and associated tasks is to establish a benchmark that reflects the complexity and realism of practical scenarios a model is likely to face. This framework enables an evaluation across six core vision-language capabilities, as identified by [55]:

- **Recognition:** The ability to identify and comprehend elements within an image, such as scenes, objects, their attributes (color, shape, material), and quantities.
- **Knowledge:** The ability to apply general or common-sense knowledge to answer questions or solve problems.
- **OCR:** The skill to read text in images and use that text to answer questions or complete tasks.

- Spatial awareness: Understanding where things are in a picture and how they relate to each other in space.
- Language generation: The ability to give clear, detailed, and well-written answers, especially for questions that need longer explanations.
- Human pose: The ability to detect and understand how people are standing, moving, or positioned in an image.

4.3.2 Assessment Method

To quantitatively assess the performance of the captioning models using our dataset, a rigorous evaluation method was required. Several approaches were considered in Section 3.4, each with distinct advantages and disadvantages. This section details the implemented approach. Among the alternatives considered, the Model-as-a-Judge approach offers a well-balanced solution by aligning three key objectives of a good assessment: scalability, semantic fidelity, and automation.

Model-as-a-Judge allows for consistent and scalable assessment across large datasets without the prohibitive costs of human evaluation, while still approximating nuanced human judgments. This approach ensures that assessment remains robust in capturing semantic quality.

The assessment method via Model-as-a-Judge is illustrated in Figure 4.4.

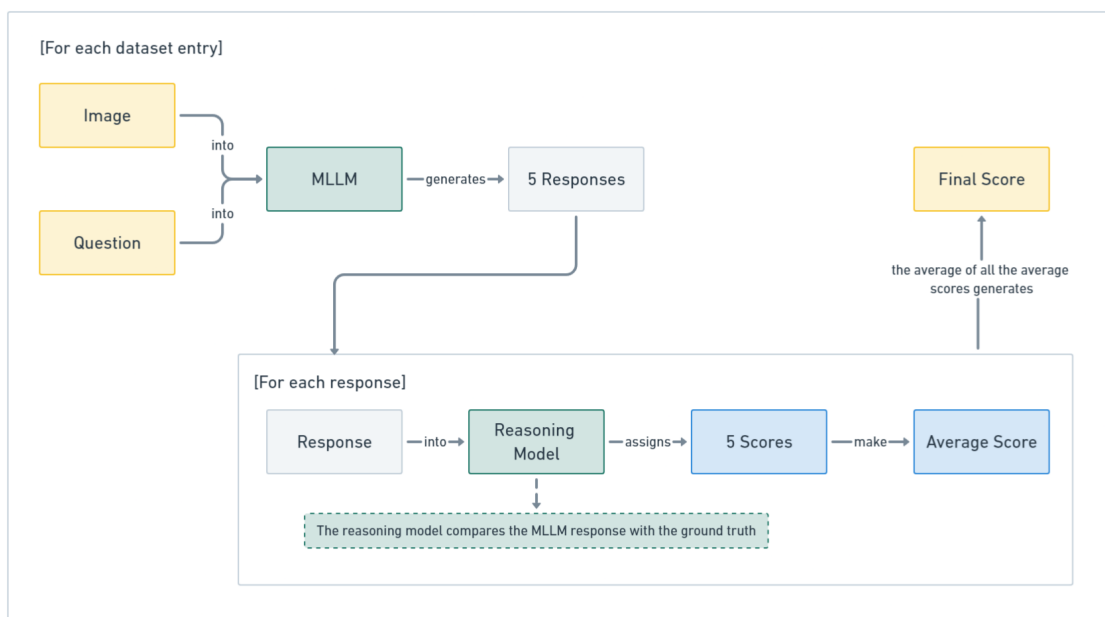


Figure 4.4: End-to-end assessment method for MLLM performance using Model-as-a-Judge. Each image-question pair is passed to the MLLM to generate five responses, which are individually scored by a reasoning model five times. The resulting scores are averaged per response and then across all responses to compute the final model score.

For each of the IQA triplets in the custom dataset, the image and its corresponding question are presented to the MLLM under evaluation. To account for the randomness nature of generative models and to test for response consistency, this process is repeated five times

for every triplet. This results in a total of five unique generated answers per image-question pair for each MLLM.

Each of the generated responses is then evaluated for its accuracy and relevance. We employ a powerful reasoning model to act as a judge. The judge model is tasked with scoring each generated answer on a continuous scale from 0.0 to 1.0 (in 0.1 increments), where 0.0 signifies a completely incorrect or irrelevant response and 1.0 represents a perfectly accurate and contextually appropriate answer. To ensure a robust assessment and mitigate potential biases from the judge model, each generated response is evaluated five separate times by the judge.

After all evaluations are complete, the scores are aggregated to produce a final performance metric for each MLLM. For each of the generated answers, the five scores from the judge model are averaged to create a single score for that specific response. Finally, all these averaged scores are themselves averaged to compute a single score that represents the overall capability of the MLLM on our custom industrial dataset. This multi-level averaging process is designed to smooth out variability and provide a reliable and comparable final score for each model.

This assessment framework is designed to be scalable, allowing for the straightforward addition of new triplets in the future with minimal effort.

Chapter 5

Implementation

This chapter details the practical implementation of the components designed in Chapter 4. It begins by describing the execution of the captioning models assessment, including the construction of the custom dataset and the workflow for evaluating and scoring the models. Subsequently, it covers the implementation of the end-to-end video summarization pipeline.

5.1 Models Assessment

The primary objective of the models assessment was to empirically determine the most suitable MLLM for the frame captioning step of our pipeline. As established in the design phase, the quality of these captions is fundamental to the overall system's performance. The implementation followed the assessment method detailed in Section 4.3, using a custom-built dataset to test various models. The implementation was carried out within a Jupyter Notebook environment using Python 3.12.7.

5.1.1 Dataset Construction

As determined by the previous analysis, no existing public dataset fulfilled the specific criteria for our evaluation. Consequently, a new dataset was constructed to serve as the benchmark for this assessment.

Dataset Entry Fields

Each entry in the dataset was structured as a JSON object with the following key fields:

- **id**: A unique integer identifier for the data point.
- **image**: The filename of the corresponding image.
- **question**: A targeted question/task designed to probe a specific model capability regarding the image's content.
- **answer**: A ground-truth answer to the question/task, serving as the benchmark for accuracy.
- **category**: The category the IQA triplet belongs to, as defined in our dataset model (Figure 4.3).
- **capabilities**: A list of the core vision-language capabilities being tested, aligned with the framework design.

Listing 5.1 illustrates an example of an entry in our custom dataset, presented in JSON format.

```

{
  "id": 1,
  "image": "user_clicking_stop_button.jpg",
  "question": "What button is the worker clicking?",
  "answer": "Stop button",
  "category": "human_machine_interaction",
  "capabilities": [
    "recognition",
    "spatial_awareness",
    "human_pose"
  ]
}

```

Listing 5.1: The structure of an entry in our custom dataset.

Images

The dataset was compiled using real images sourced from Freepik and iStock, along with synthetic images generated using Imagen 3 by Google. In total, the dataset contains 26 images, of which 7 were synthetically generated. These images are reused in various entries. For example, an image of a worker clicking a red button of a milling machine can be used for two questions: "What button is the worker clicking?" and "What type of machine is being operated?".

Categories and Capabilities Frequency

To ensure the dataset composition is aligned with our intended evaluation criteria, we analyzed the distribution of both the categories and capabilities across the IQA triplets.

Figure 5.1 presents the frequency of each category as defined in our dataset model. This distribution highlights a similar focus on all the common scenarios in industrial collaboration.

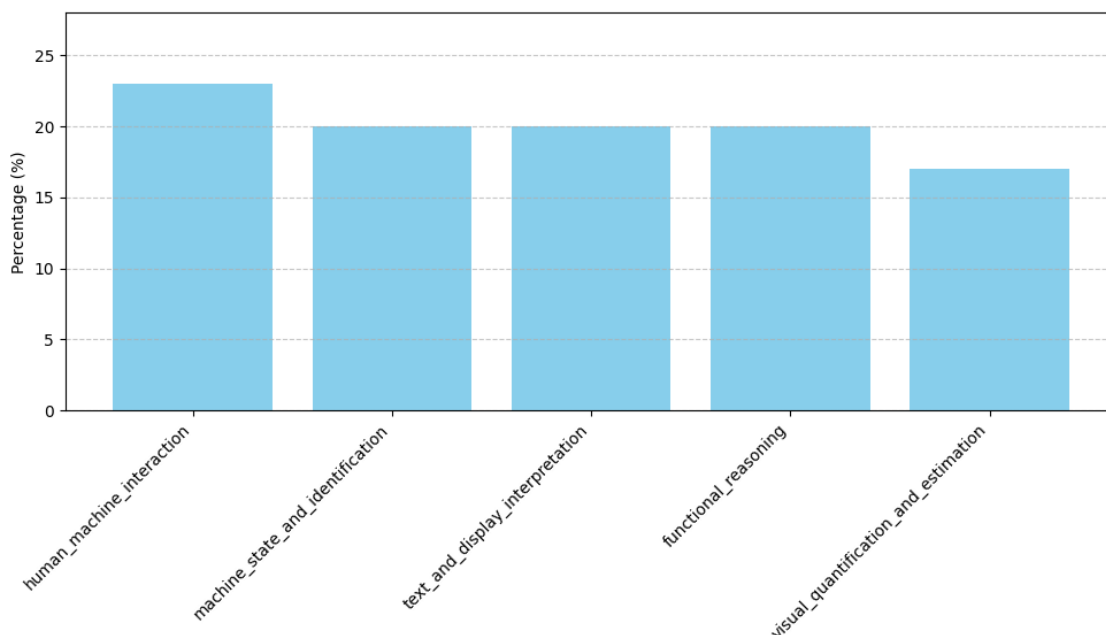


Figure 5.1: Category frequency in the custom dataset.

Figure 5.2 illustrates the frequency of the targeted capabilities in our dataset. The distribution reflects our aim to comprehensively test models across a spectrum of relevant skills, with recognition and knowledge being the most prominent.

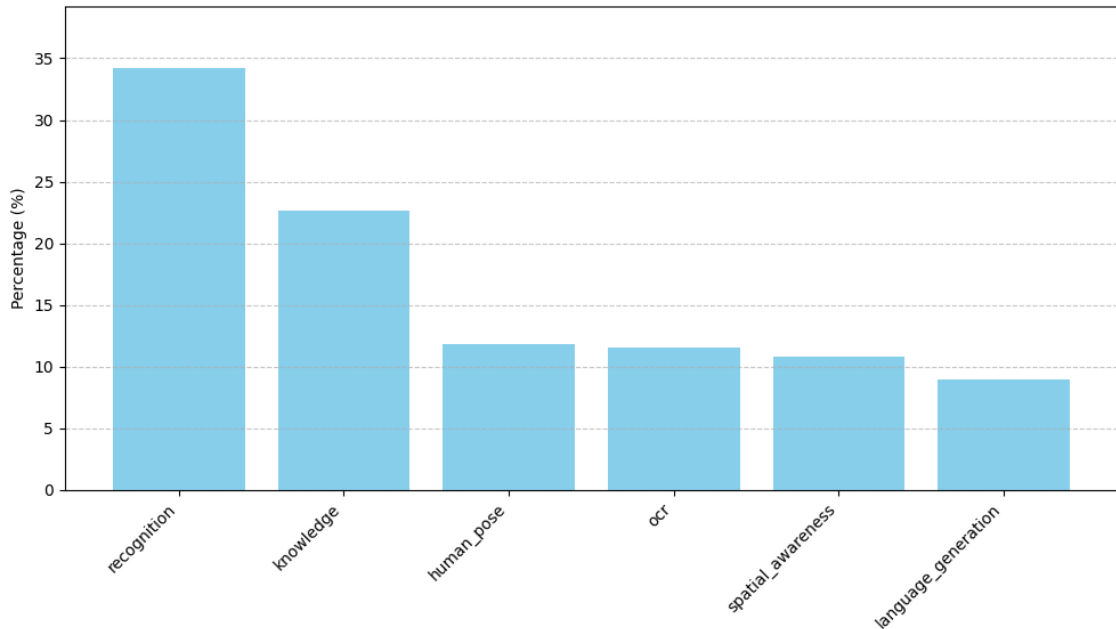


Figure 5.2: Capability frequency in the custom dataset.

The final dataset consists of **100** unique IQA triplets, meticulously crafted to cover the full spectrum of categories and capabilities outlined in the design.

5.1.2 Experimental Setup and Workflow

To ensure a reproducible and organized experimental process, the implementation was modularized into several key components.

Configuration Management

A configuration module was used to centralize all critical parameters for an evaluation run. This approach allows for easy switching of models and data paths without altering the core script. The key parameters include:

- `json_path`: The path to the dataset file.
- `images_dir_path`: The directory containing the image files.
- `judge_model`: The reasoning model designated to act as the evaluator (o4-mini).
- `eval_model`: The MLLM being assessed (e.g., gemini-2.0-flash, gemma3:12b, llava:13b).

API Abstraction

To streamline interactions with various MLLMs, a dedicated module was implemented. This module abstracts the specific API calls for different services (OpenAI and Google Gemini) and local instances (via Ollama). It provides a unified interface for sending a prompt and an image to a model and receiving a textual response. This design choice was crucial for easily extending the evaluation to new models.

Judge Model Prompt

A separate module was created to manage all prompt templates. The most critical component of this module is the judge template, which dynamically constructs the prompt for the "Model-as-a-Judge". This prompt employs a few-shot learning strategy, providing the judge model with several examples of questions, ground-truth answers, predictions, and their corresponding scores, as shown in Figure 5.3.

```

You are an evaluation assistant. Only respond with a single float score (0.0, 0.1, ..., 1.0), no
explanation.
Compare the ground-truth and prediction from AI models, to give a correctness score for the
prediction.
<AND> in the ground-truth means it is totally right only when all elements in the ground-truth
are present in the prediction, and <OR> means it is totally right when any one element in the
ground truth is present in the prediction. The correctness score is from 0.0 (totally wrong) to
1.0 (totally right).

Question | Ground truth | Prediction | Correctness
- | - | - | -
What safety equipment is the user wearing? | helmet <OR> gloves | glasses | 0.0
What safety equipment is the user wearing? | helmet <AND> gloves | helmet | 0.5
What safety equipment is the user wearing? | helmet <AND> gloves | helmet and gloves | 1.0
Why is this situation unsafe? | The electrical panel is left open with exposed wiring, and there
is water on the floor nearby. This combination creates a high risk of electrical shock or short
circuit, endangering anyone in the vicinity. | There is water on the floor, which can be
dangerous. People could slip or fall if they aren't careful. | 0.5
Why is this situation unsafe? | The electrical panel is left open with exposed wiring, and there
is water on the floor nearby. This combination creates a high risk of electrical shock or short
circuit, endangering anyone in the vicinity. | The electrical panel is open with exposed wiring,
and there is water on the ground close to it. This creates a serious risk of electrical shock and
makes the area extremely hazardous for anyone nearby. | 1.0
{question} | {ground_truth} | {prediction} |

```

Figure 5.3: Prompt given to Judge Model for MLLM response evaluation. Idea gathered from [55].

Evaluation Script

The main script orchestrates the end-to-end evaluation workflow. It begins by loading the parameters from the configuration module and the IQA dataset. We also defined constants for the evaluation, such as `N_OF_RESPONSES_PER_MODEL` and `N_OF_JUDGES_PER_RESPONSE`, to ensure robustness against model variability.

The script iterates through each IQA triplet in the dataset. For each triplet, it invokes the `eval_model` five times with the same image and question. This repetition accounts for the variability nature of generative models. An error-handling mechanism with a retry policy (`MAX_REQUEST_ATTEMPTS = 5`) was implemented to manage potential API timeouts or failures.

For each of the five responses generated by the `eval_model`, the script invokes the `judge_model`. The judge is provided with the original question, the ground-truth answer, and the model's generated response, using the judge prompt template (Figure 5.3). To obtain

a stable and reliable score, the judge model is prompted five times for the same response. This minimizes the impact of any randomness in the judge's own evaluation process.

The script collects these five raw scores (ranging from 0.0 to 1.0). The five scores for a single response are aggregated into a final score. The script stores the generated responses and their final aggregated scores in a list.

After processing all responses for a given IQA triplet, the average score across the five responses is calculated. The results for each triplet, including the ID, ground-truth, individual responses with their scores, the overall average score for the triplet, category, and capabilities, are appended to a list and periodically saved to a JSON file. This ensures that progress is not lost during long evaluation runs.

Once all IQA triplets have been evaluated, a final module processes the saved scores file. An example of the model's evaluation output file is shown in Listing 5.2

```
{
  "overall_average_score": 0.75,
  "average_score_by_category": {
    "human_machine_interaction": 0.77,
    "machine_state_and_identification": 0.89,
    "visual_quantification_and_estimation": 0.7,
    "text_and_display_interpretation": 0.7,
    "functional_reasoning": 0.67
  },
  "average_score_by_capability": {
    "recognition": 0.75,
    "spatial_awareness": 0.7,
    "human_pose": 0.78,
    "knowledge": 0.75,
    "ocr": 0.7,
    "language_generation": 0.62
  }
}
```

Listing 5.2: Structure of a model's evaluation output file.

To facilitate detailed performance analysis and ensure reproducibility of results, the script stores a comprehensive set of evaluation metrics at multiple levels:

- Overall average score for the `eval_model` across the entire dataset.
- Average score by category, revealing the model's strengths and weaknesses in different contexts.
- Average score by capability, providing insight into the model's performance on specific tasks.

The comprehensive outputs from this assessment workflow, which include detailed scores per model, category, and capability, are fundamental to the next stage of our research.

We utilize this data to perform a thorough analysis, gathering the final results and constructing the comparative charts and tables that are presented and discussed in detail in Chapter 6.

5.2 Summarization Pipeline

This section details the implementation of the end-to-end video summarization pipeline, translating the architecture designed in Chapter 4 into a functional system. The implementation was developed in Python 3.12.7, using a modular structure to ensure maintainability, and ease of experimentation.

The core of the pipeline is built upon several key open-source libraries and external APIs, chosen for their performance and capabilities in handling complex computer vision and natural language processing tasks. The orchestration of the pipeline is managed by a central script that simulates a real-time streaming scenario. It processes the source video in sequential chunks of 30 seconds, dispatching each chunk to a worker thread using Python's `ThreadPoolExecutor`. This concurrent approach allows for the parallel processing of video segments, which is essential for achieving the efficiency required in near-real-time industrial monitoring applications. The implementation is broken down into several distinct modules, each responsible for a specific stage of the workflow.

5.2.1 Video Processing Module

This initial module handles the ingestion and preparation of the video data. It utilizes the *OpenCV*¹ (`cv2`) library to decode the video file and segment it into chunks. Within each chunk, a keyframe selection strategy based on the SSIM is employed, using the `structural_similarity` function from `scikit-image`. The first frame of every chunk is automatically selected as a keyframe. Subsequent frames are then compared against the most recently selected keyframe. Every 1 second, a new frame is only added to the keyframe list if its SSIM score falls below 0.9, indicating a significant visual change, avoiding processing when nothing is happening (e.g., user is away from the machine). Figure 5.4 explains this approach visually.

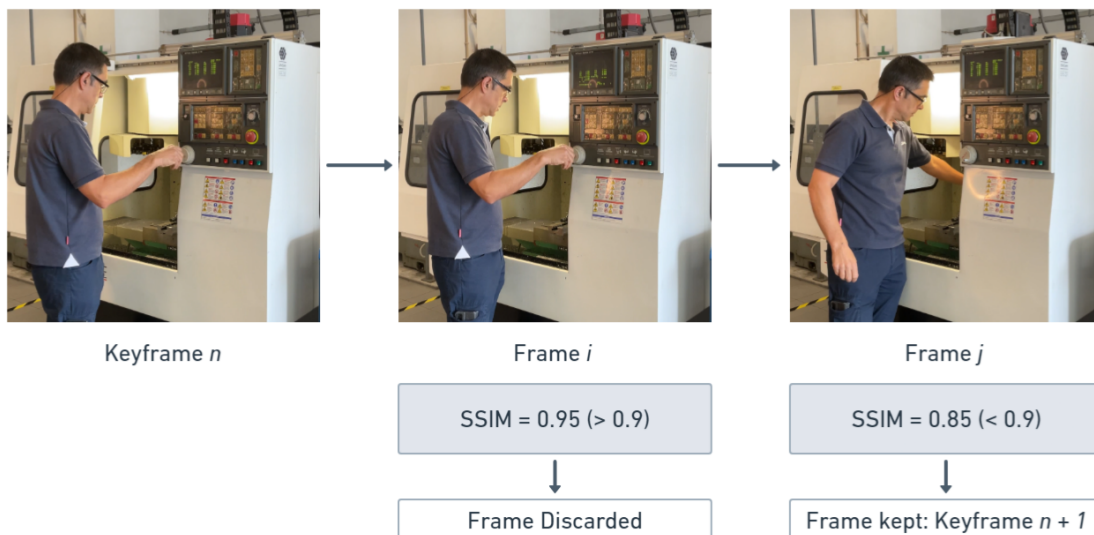


Figure 5.4: Pipeline SSIM approach explained.

¹<https://opencv.org/>(accessed 2025-06-29)

This content-aware sampling, performed on grayscale versions of the frames to focus on structural differences, drastically reduces computational redundancy. Frames are managed as NumPy arrays, a standard for numerical and image data manipulation in Python.

5.2.2 Object Detection Module

For each selected keyframe, this module identifies and classifies relevant objects. The implementation uses a state-of-the-art YOLOv11 model from the Ultralytics framework, approached on Subsection 3.2.3. The model is executed using PyTorch, with automatic offloading to a CUDA-enabled GPU (NVIDIA GeForce RTX 3060 Ti), to accelerate inference. A confidence threshold of 0.8 is applied to the model's output to filter out low-probability detections, ensuring that only high-confidence object data is passed to the subsequent stages, as illustrated in Figure 5.5.



Figure 5.5: Visual workflow of the object detection module. An input frame is processed by the YOLOv11 model, which produces bounding boxes with class predictions and confidence scores. A confidence threshold of 0.8 is then applied to filter out low-probability detections (e.g., "backpack: 0.62"), resulting in a high-confidence list of objects.

5.2.3 Frame Captioning Module

This module is responsible for generating descriptive text for each keyframe. It interfaces with the OpenAI API to utilize the gpt-4.1-mini model (chosen based on the results presented in Chapter 6). Google's gemini-2.0-flash model was the best model; however, due to API reliability issues - despite staying within the documented rate limits - the pipeline experienced intermittent failures. As stated in Gemini's API documentation, "Specified rate limits are not guaranteed and actual capacity may vary". Owing to this instability, we opted for the more reliable gpt-4.1-mini model as the second-best rated alternative.

A critical feature of this module is the implementation of temporal context. A thread-safe rolling queue (`collections.deque`) maintains a history of the five most recent captions. This history is dynamically inserted into the prompt for each new frame, providing the model with immediate context of preceding events and enabling it to generate a more coherent narrative. Image frames are encoded into base64 format before being sent to the API, a standard practice for vision model endpoints. Figure 5.6 shows the dynamic prompt construction for captioning.

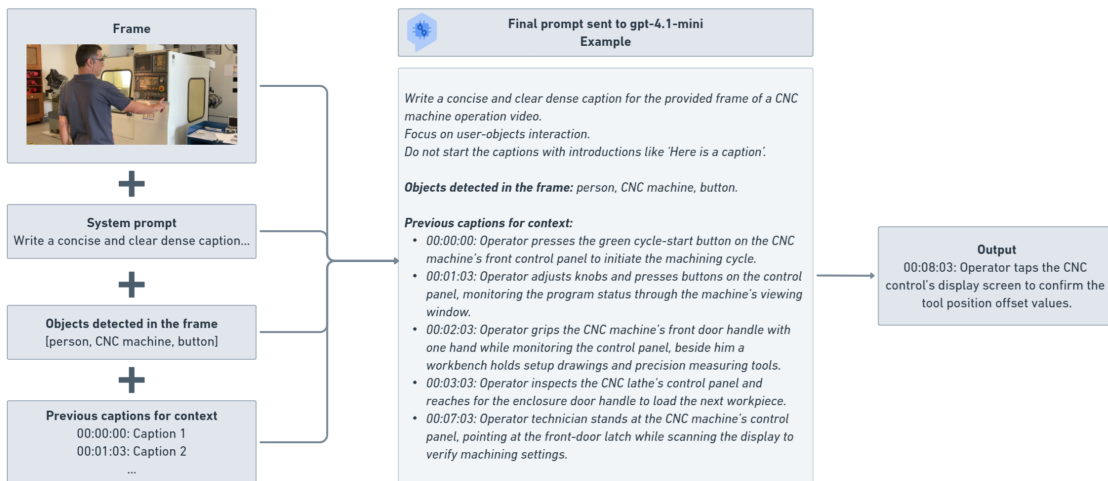


Figure 5.6: Dynamic prompt construction for captioning. It combines the frame with system instructions, a list of detected objects, and context from previous captions. This complete "package" is then sent as a single prompt to the MLLM, which generates the final, context-aware caption of the frame.

5.2.4 Final Summarization Module

In the final stage, the chronologically ordered captions from a chunk are synthesized into a structured summary. This is accomplished using the Google Generative AI API to access the powerful, as noted in Section 3.3, gemini-2.5-pro model. This model was chosen for its advanced reasoning capabilities and, crucially, its native support for generating structured JSON output. By providing the API with a Pydantic schema, the model is constrained to return a response containing a descriptive summary and a distinct list of detected alerts. This eliminates the need for complex and brittle output parsing. The final output is saved in two formats: a raw JSON file for downstream programmatic use and a human-readable Markdown file. Figure 5.7 shows the dynamic prompt construction for summarization.

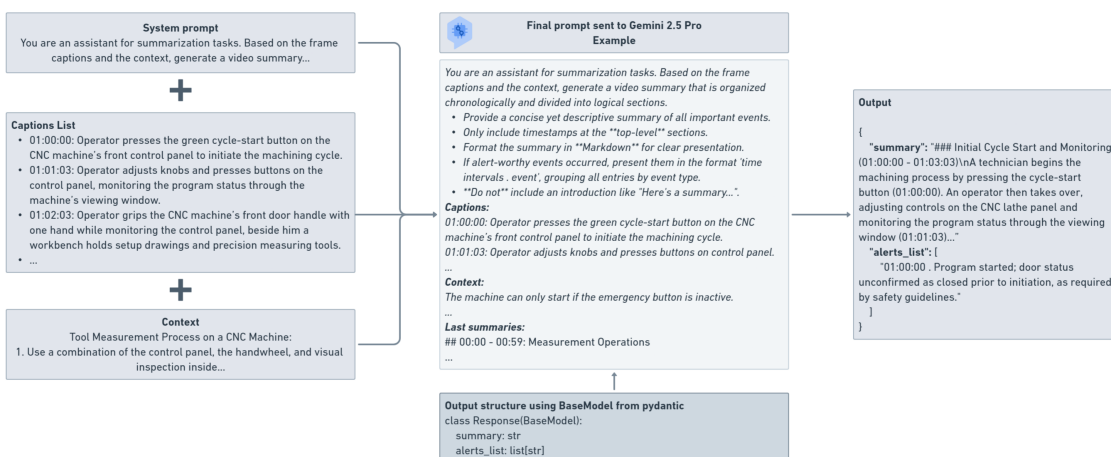


Figure 5.7: Dynamic prompt construction for summarization. The workflow combines a system prompt, frame captions, context, and the last 10 summaries to guide the model in producing a specific output format (summary and alerts).

System Prompt plays an important role in guiding the model into the appropriate operational mode. By defining the role as a summarization assistant, it immediately focuses the model's capabilities on the task at hand, steering it away from general conversation. It also establishes a clear objective by specifying that the expected output is a video summary, which helps prevent the model from drifting into unrelated activities such as rephrasing captions or generating creative text.

Captions List serves as the ground-truth - the factual foundation on which the entire summary is built. Each entry in the list is tied to a timestamp (t_1, t_2, \dots, t_n), enabling the model to maintain the chronological order of events. These timestamps help construct a narrative that reflects the actual sequence in the video. Furthermore, the descriptions within the captions provide essential details that the model uses to compose an accurate and meaningful summary.

Context introduces domain-specific knowledge that goes beyond what is directly visible in the video frames. This information allows the model to understand not just what happened, but why it matters. For instance, if the machine starts while the door status is unconfirmed, the model can recognize this as a potential anomaly - something it would not infer from the captions alone. Context also helps reduce ambiguity by clarifying user actions that might otherwise seem out of place.

Last Summaries maintain narrative coherence across multiple video chunks. This component provides the model with long-term temporal context. Implemented as a rolling queue that stores the 10 most recent summaries, this mechanism acts as the system's memory. Before processing a new chunk, the summaries from previous, chronologically ordered chunks are appended to the prompt. This allows the model to understand the broader context of events that have already transpired, preventing redundant descriptions and enabling it to recognize sequences that span across chunk boundaries. With a 30-second chunk duration and a maximum of 10 summaries in the queue, the model has a context of the last 5 minutes. This context window scales directly with the chunk duration; for example, 1-minute chunks provide a 10-minute context.

Final Assembled Prompt is where all the components (system prompt, captions, and context) come together within a single, unified instruction. In addition to guiding content, this final prompt enforces structure, encouraging the model to organize events chronologically and logically. Formatting constraints, such as using Markdown for readability and structuring alert-worthy events as key-value pairs, make the output not only more presentable but also machine-readable. Additionally, by explicitly instructing the model to avoid conversational filler (e.g., "Here is the summary requested:"), the prompt ensures a direct and clean result.

Structured Output finally enforces a strict schema using a `BaseModel` from Pydantic, which defines the expected format: a string for the summary and a list of strings for the alerts. It removes the need for fragile parsing logic.

This modular and concurrent implementation provides a scalable foundation for processing industrial video data. It effectively translates the theoretical design into a practical system capable of generating the structured summaries that are analyzed in Chapter 6.

Chapter 6

Results

This chapter presents the empirical findings from the experiments detailed in Chapter 5. The analyses provide a quantitative evaluation of the project's primary objectives, from model selection to final output generation.

6.1 Models Assessment Results

This section presents the performance results of the different MLLMs evaluated using the assessment framework detailed in Section 4.3. The goal was to empirically determine the most suitable model for the frame captioning step of the pipeline. The following subsections provide a quantitative breakdown of each model's performance, starting with an overall comparison and then delving into specific categories and capabilities.

6.1.1 Overall Model Performance

Figure 6.1 shows a comparison of the models, illustrating their final average scores across our custom industrial dataset.

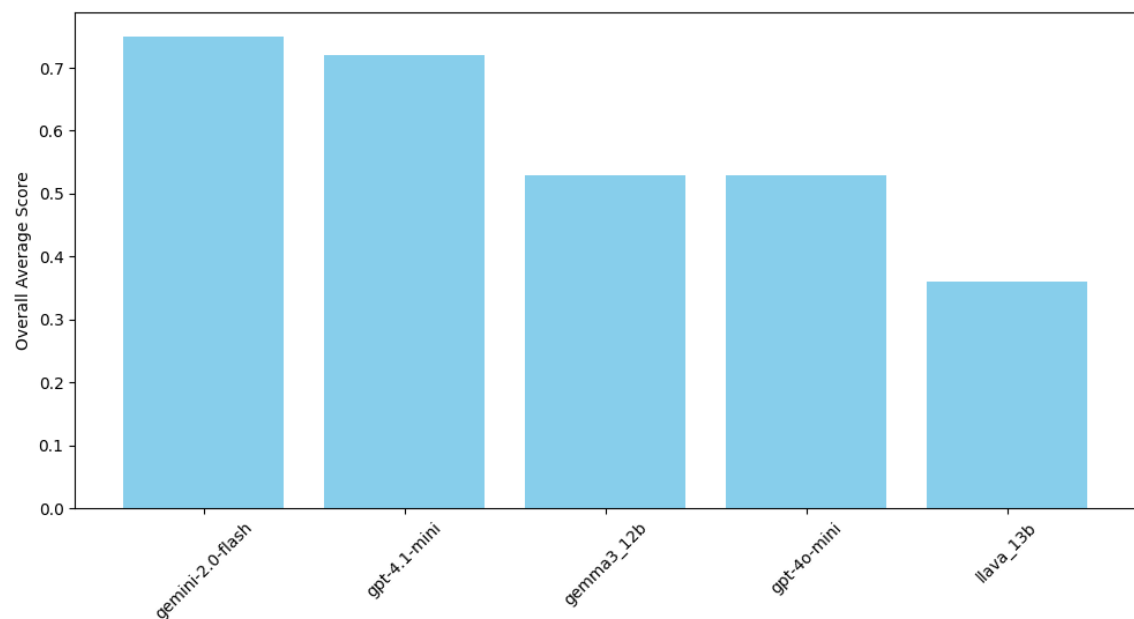


Figure 6.1: Overall MLLM Performance on our custom industrial dataset, with the models on the x-axis and the average score (from 0.0 to 1.0) on the y-axis.

This overall score, derived from the "Model-as-a-Judge" methodology, serves as a primary indicator of each MLLM's general proficiency in understanding and responding to the diverse tasks. The results clearly show a performance hierarchy among the evaluated models, with gemini-2.0-flash achieving the highest overall score.

6.1.2 Performance by Category

To understand the specific strengths and weaknesses of each model, their performance was disaggregated across the five distinct categories defined in the dataset: Functional Reasoning (FR), Human-Machine Interaction (HMI), Machine State and Identification (MSI), Text and Display Interpretation (TDI), and Visual Quantification and Estimation (VQE). Table 6.1 presents the average scores for each model within these categories.

Table 6.1: Average MLLM scores by category.

Model	FR	HMI	MSI	TDI	VQE	Total
llava_13b	0.62	0.42	0.40	0.11	0.22	0.36
gpt-4o-mini	0.63	0.56	0.62	0.57	0.20	0.53
gemma3_12b	0.45	0.69	0.54	0.43	0.49	0.53
gpt-4.1-mini	0.61	0.86	0.80	0.68	0.63	0.72
gemini-2.0-flash	0.67	0.77	0.89	0.70	0.70	0.75

For a more intuitive comparison, Figure 6.2 illustrates this data in a radar chart. Each axis represents a category, and the area covered by a model's polygon indicates its balanced performance across the different contexts.

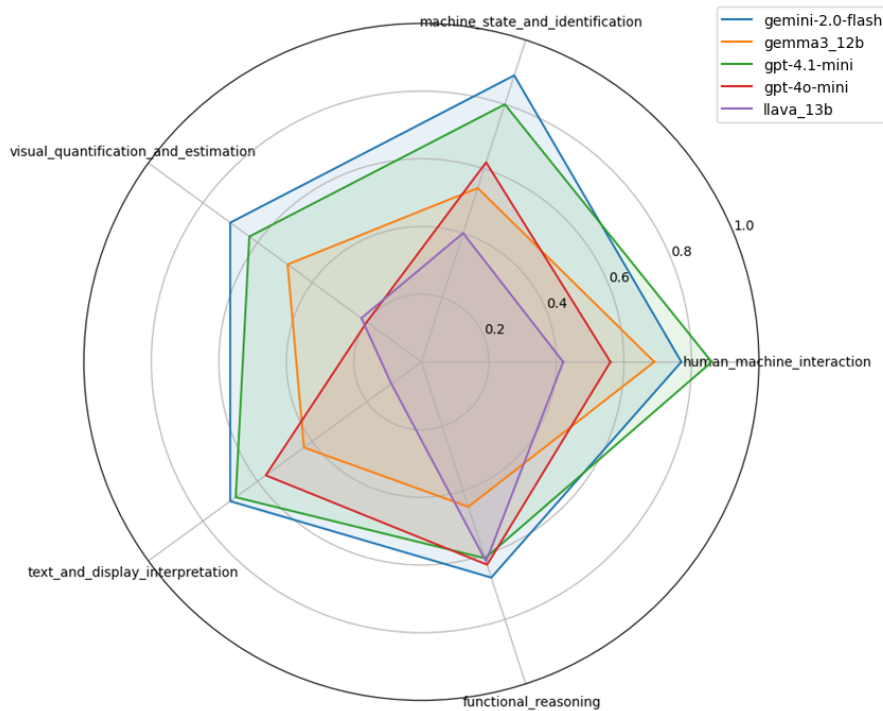


Figure 6.2: Average MLLM scores by category (radar chart).

The dominant performance of gemini-2.0-flash is immediately apparent, as it achieves the highest scores in four out of five categories. However, a key insight from the chart is the

specialized strength of gpt-4.1-mini, which surpasses all other models in the HMI category, making it a strong candidate for operator-focused tasks. In contrast, categories such as VQE proved challenging for most models, with gpt-4o-mini and llava_13b showing significant performance deficits in this area.

6.1.3 Performance by Capability

To gain a more granular insight into the underlying skills of each model, this section analyzes performance across the six core vision-language capabilities: Recognition (Rec), OCR, Knowledge (Know), Language Generation (Gen), Spatial Awareness (Spat), and Human Pose (Pose). Table 6.2 shows the average scores for these abilities.

Table 6.2: Average MLLM scores by capability.

Model	Rec	OCR	Know	Gen	Spat	Pose	Total
llava_13b	0.38	0.27	0.46	0.50	0.29	0.47	0.36
gemma3_12b	0.54	0.41	0.53	0.41	0.53	0.65	0.53
gpt-4o-mini	0.53	0.58	0.60	0.51	0.24	0.60	0.53
gpt-4.1-mini	0.72	0.66	0.74	0.56	0.72	0.82	0.72
gemini-2.0-flash	0.75	0.70	0.75	0.62	0.70	0.78	0.75

Figure 6.3 translates these scores into a radar chart. Again, each axis represents a capability, and the area covered by a model's polygon indicates its balanced performance across the different contexts.

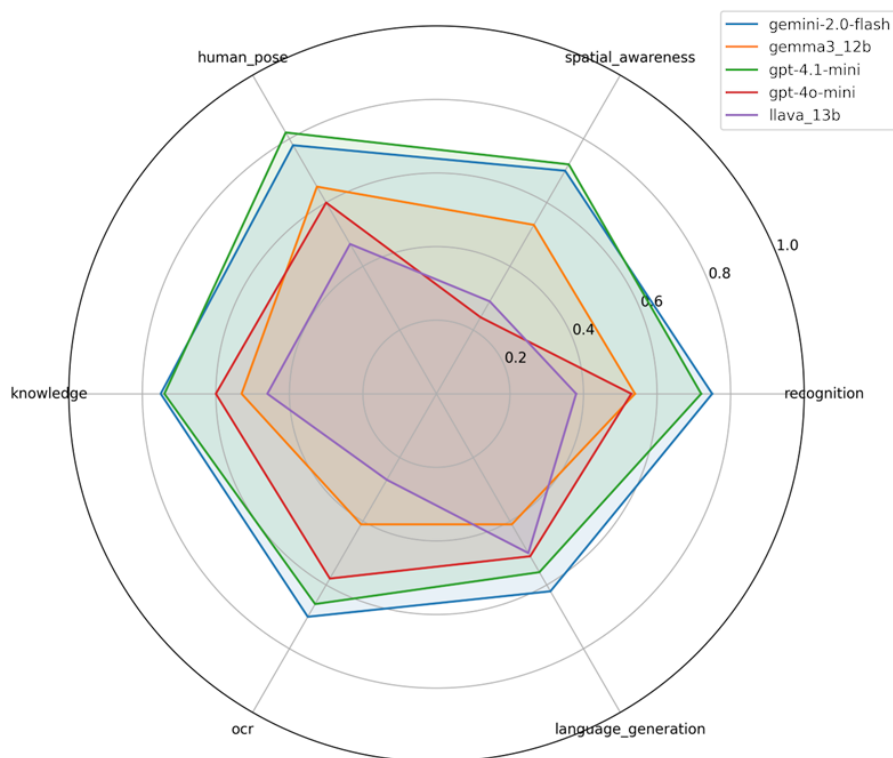


Figure 6.3: Average MLLM scores by capability (radar chart).

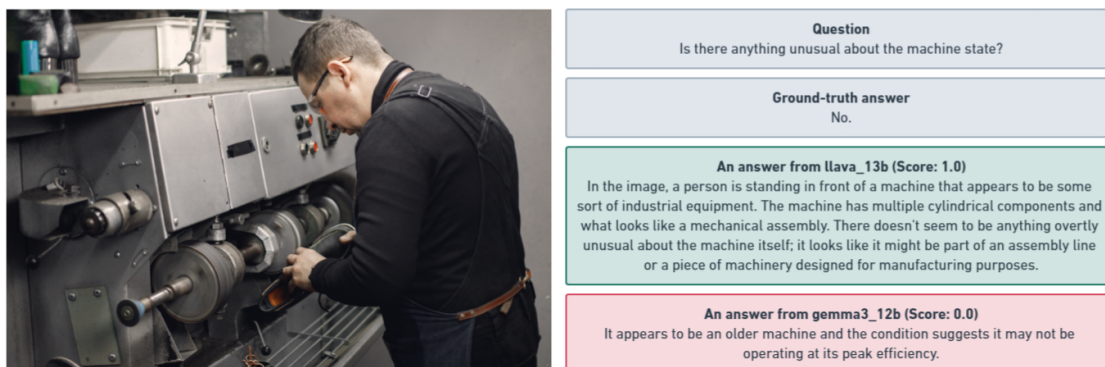
For instance, the superior performance of gpt-4.1-mini in the HMI category can be directly attributed to its leading scores in the Human Pose and Spatial Awareness capabilities. Conversely, the struggles of gpt-4o-mini with visual quantification are explained by its significant weakness in Spatial Awareness. The chart also highlights that capabilities like OCR and Spatial Awareness are crucial differentiators, with the top-tier models demonstrating robust performance while lower-performing models like llava_13b show critical deficiencies.

6.1.4 Qualitative Analysis

While the quantitative scores provide a high-level overview of model performance, a qualitative analysis of specific model responses is essential to understand the nuances behind the numbers. This section delves into two of the more unexpected findings from the assessment to illustrate the practical differences in model capabilities.

Gemma-3's Low Language Generation Score

One of the most counterintuitive results was that gemma3_12b scored lower on Language Generation (0.41) than the llava_13b model (0.50). The Language Generation capability is defined as the ability to provide clear, detailed, and well-written answers, which is especially critical for questions requiring longer explanations. This capability is heavily tested in the "Functional Reasoning" category. Figure 6.4 presents the qualitative analysis of the two models.



Question
Is there anything unusual about the machine state?

Ground-truth answer
No.

An answer from llava_13b (Score: 1.0)
In the image, a person is standing in front of a machine that appears to be some sort of industrial equipment. The machine has multiple cylindrical components and what looks like a mechanical assembly. There doesn't seem to be anything overtly unusual about the machine itself; it looks like it might be part of an assembly line or a piece of machinery designed for manufacturing purposes.

An answer from gemma3_12b (Score: 0.0)
It appears to be an older machine and the condition suggests it may not be operating at its peak efficiency.

Figure 6.4: A qualitative comparison of model responses to a functional reasoning question. The llava_13b model correctly identifies that there is nothing unusual, providing a descriptive and fact-based answer. In contrast, gemma3_12b makes an unsupported assumption about the machine's operational efficiency, which is penalized as a speculative response.

Llava_13b provided a detailed and descriptive answer grounded in the visual evidence. It described the scene accurately and correctly concluded that there is nothing "overtly unusual". This response directly answers the question based only on what can be seen, earning it a perfect score. Gemma3_12b, while starting with a reasonable observation ("It appears to be an older machine"), immediately pivots to speculation. It claims "the condition suggests it may not be operating at its peak efficiency". This statement is an inference that cannot be supported by the visual information in the image. By suggesting a potential inefficiency, the model implies there is something unusual, directly contradicting the ground-truth answer. This tendency to generate unverified and speculative information, rather than sticking to a clear description and direct answer, is why it scored zero.

GPT-4o-mini's Weakness in Spatial Awareness

Perhaps the most striking finding was the poor performance of gpt-4o-mini in Spatial Awareness, where it scored just 0.24, significantly lower than gemma3_12b's score of 0.53. Spatial Awareness is the ability to understand the location of objects and their relationships to each other in space. This is a fundamental skill for interpreting human-machine interactions and safety. Figure 6.5 shows a qualitative assessment of llava_13b and gpt-4o-mini, on a task requiring Spatial Awareness within a human-machine interaction scenario.

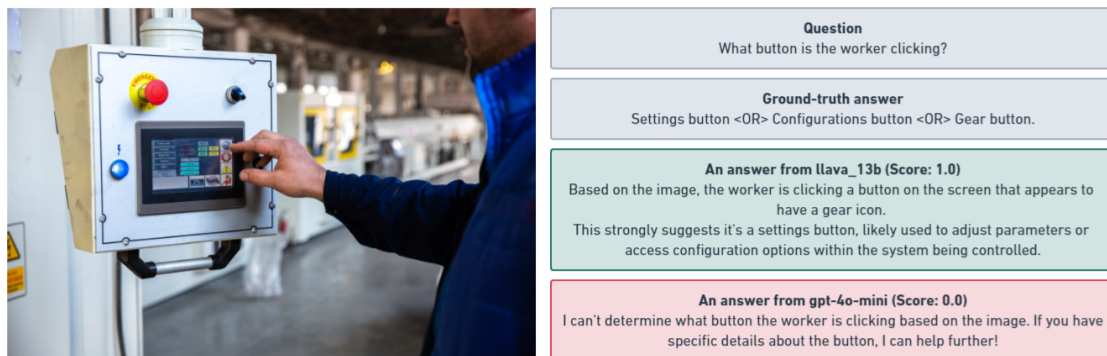


Figure 6.5: A qualitative example from the Spatial Awareness capability. While the llava_13b model correctly identifies the specific button being pressed by the worker based on its icon, gpt-4o-mini completely fails the task, claiming it cannot determine the action from the image. This demonstrates a fundamental deficit in spatial perception.

The models were asked the direct spatial question, "What button is the worker clicking?". The correct response is any variation of "Settings button", "Configurations button", or "Gear button", corresponding to the universally recognized gear icon being touched on the screen.

Llava_13b successfully completed the task. It correctly identified that the worker is pressing a button with a "gear icon" and accurately inferred its purpose as a "settings button". This answer demonstrates strong Spatial Awareness by correctly localizing the worker's finger and identifying the specific object of interaction. Gpt-4o-mini failed completely. Its response is factually incorrect. The worker's finger is clearly and unambiguously pointing to the gear icon on the screen. This is not a complex or ambiguous scene, and the model's inability to determine the button points to a critical failure in its visual perception and spatial reasoning capabilities. This example starkly illustrates why gpt-4o-mini performed so poorly in the Spatial Awareness capability.

Lexical-based metrics like BLEU and ROUGE, while useful for tasks with high word overlap such as translation or summarization, fall short in evaluating the quality of model-generated responses in more nuanced contexts. These metrics rely on n-gram comparisons with a reference text, which makes them incapable of detecting factual inaccuracies or hallucinations - such as a fluent yet incorrect response - and they also unjustly penalize accurate paraphrasing that uses different vocabulary. Because of these flaws, we use a "Model-as-a-Judge" method .

6.2 Pipeline Summarization Results

To validate the performance of the summarization pipeline, a series of real-world test videos were created. We recorded an operator using a CNC-TRIAC CENTRE machine at the ISEP's facilities. Footage was captured from two distinct angles, resulting in 30:50 minutes of video from the right angle and 30:26 minutes from the left angle. To test the pipeline's anomaly detection capabilities, the operator was instructed to intentionally commit a series of predefined faults during the recording sessions. Following the recordings, the videos were carefully reviewed and edited to create a curated set of evaluation scenarios. For each of these final video clips, a detailed ground-truth summary was manually generated to serve as a benchmark for accuracy.

Table 6.5 outlines the final, edited video clips used for the evaluation, detailing the specific context of each scenario and its duration.

Table 6.3: Contexts of worker interactions with CNC machine.

Video ID	Context	Duration
Vid1	The worker is measuring three different tools on the CNC machine.	05:05
Vid2	The worker is performing the machine zeroing procedure.	00:30
Vid3	The worker attempts to execute a program while the program protection feature is disabled.	00:58
Vid4	Standard execution process of a program on the CNC machine.	00:22
Vid5	The worker attempts to operate the machine while the emergency stop button is engaged.	00:24
Vid6	The worker tries to run a program while the machine door is open.	00:16

6.2.1 Summary Results

While traditional lexical-based metrics like BLEU offer a measure of textual overlap, they often fail to capture the semantic accuracy and contextual relevance of a generated summary. Recognizing this limitation, a significant effort was dedicated to employing a more sophisticated evaluation methodology. We adopted the "Model-as-a-Judge" approach, again, to use the advanced reasoning capabilities of a powerful model for a more semantic evaluation. For this task, we also utilized gemini-2.5-pro. The evaluation process involved providing the judge model with our manually created ground-truth summary alongside the summary generated by the pipeline for the same video. The model was then prompted to assign a score from 0.0 to 1.0 and to provide a qualitative assessment detailing the strengths and weaknesses of the generated summary.

The following prompt was used to guide the model's evaluation:

```
Give my pipeline summary a score from 0.0 to 1.0. Also specify the
strengths and weaknesses.
This is my ground-truth video summary:
{gt summary}
And this is the summary my summarization pipeline generated.
{pipeline summary}
```

Listing 6.1: Prompt used to guide the pipeline summaries evaluation.

Table 6.4 presents the results of this evaluation, including the scores and the judge model's qualitative feedback for each video. The pipeline achieved an overall average score of 0.72.

Table 6.4: Evaluation of video summaries: strengths and weaknesses of the pipeline, by the Model-as-a-Judge.

Video	Score	Strengths	Weaknesses
Vid1	0.7	Identifies repeated actions: button and dial use, handwheel operation, and note-taking. Recognizes general context as "tool measurement" on CNC.	Fragmented and overly detailed (e.g., hand usage), which disrupts high-level comprehension.
Vid2	0.7	Accurately captures the "zeroing procedure" and interaction with core components like control panel, buttons, and rotary handwheel.	Omits sequential zeroing of X, Y, and Z axes, which are critical to the procedure.
Vid3	0.5	Identifies CNC operation context and key actions: pressing buttons, referencing documents, pointing at the screen.	Too generic; lacks specific terms like "Program Selection", and omits key button names like "Library", "Reset", "Cycle Start".
Vid4	0.6	Notes operator interaction with CNC control panel and general screen-monitoring behavior.	Uses vague descriptions rather than concrete steps (e.g., "select program", "activate auto"). Possibly due to limited screen/button visibility.
Vid5	0.8	Recognizes topic of powering on CNC machine, including button pressing and screen observation. Provides general context of the activity.	Misses control panel state: machine is "Not Ready", a key issue in the task.
Vid6	1.0	Identifies components (operator, control panel, door), captures actions with detail ("pressing buttons", "reaching in"), and adds useful context beyond ground-truth.	–

6.2.2 Alert Detection Results

A primary objective of the pipeline is the accurate identification of operational anomalies and safety violations, which are flagged as "alerts". This capability was tested by evaluating the pipeline's ability to correctly identify the predefined faults in the test videos. Furthermore, scenarios with no faults were included to test for the generation of false positives, or "hallucinated" alerts. The nature of these alerts varied; some, like attempting to start the machine with the door open, should trigger an immediate alert, while others, such as a procedural deviation like measuring an incorrect number of tools, required a more extended temporal analysis to be detected.

Table 6.5 summarizes the pipeline's performance in alert detection for each video scenario.

Table 6.5: Validation of the alert detection across test scenarios.

Video ID	Context	Status
Vid1	We said in the context that the operator should only measure two tools. The operator measured three.	✓
Vid2	No alert should be produced. We used this to test possible hallucinations.	✓
Vid3	An alert saying that the operator tried to start a program with the program protection active should be produced. It was not.	✗
Vid4	A simple program execution. It should not produce any alert.	✓
Vid5	Operator tries to use the machine with the emergency button active. Pipeline found it.	✓
Vid6	Operator tries to start a program with the machine door open. Pipeline found it.	✓

The system successfully detected the majority of the predefined faults, correctly identifying procedural deviations (Vid1), critical safety issues like an engaged emergency stop (Vid5), and attempts to operate the machine with the door open (Vid6). The pipeline also proved robust against generating false positives in scenarios where no faults were present (Vid2 and Vid4), indicating a good level of precision. However, a notable failure occurred in Vid3, where the system did not detect the operator's attempt to run a program with the program protection feature disabled. This indicates a potential weakness in the pipeline's ability to recognize specific machine states from the available visual data.

The model's failure to detect a key safety and operational feature highlights an area for future improvement and refinement of its detection capabilities. Despite this, the overall success rate in the test set was high, with five out of six scenarios being correctly assessed.

Chapter 7

Conclusions and Future Work

This dissertation embarked on an investigation into the application of LLMs and MLLMs for structured video summarization in industrial environments. The primary goal was to design and evaluate a system capable of analyzing video footage of human-machine interactions to enhance operational safety and efficiency by automatically detecting anomalies. This final chapter synthesizes the key findings of the research, reflects on the achievements and limitations of the developed prototype, and outlines promising directions for future work.

7.1 Conclusions

The research successfully addressed its core objectives, culminating in a functional prototype that validates the feasibility of using modern AI models for this complex task.

First, a systematic literature review confirmed that while many frameworks exist for general video summarization, the specific application to structured, industrial contexts remains underexplored. Our analysis of state-of-the-art systems like MISAR [4], VidIL [49], M3SUM [48], AVIWT [3] and XaiR [42] allowed us to identify a common set of essential processing steps - including frame sampling, captioning, and object detection - which formed the architectural basis of our own pipeline.

A significant contribution of this work was the identification of a critical gap in publicly available data. The research method for RQ-3 revealed that no existing dataset provided the necessary IQA triplets required for evaluating model performance on nuanced, task-specific queries in an industrial setting. To overcome this, we constructed a custom dataset of 100 IQA triplets, categorized by specific industrial scenarios and designed to test core vision-language capabilities.

The empirical assessment of various MLLMs using this dataset yielded crucial insights. The results demonstrated a clear performance advantage for state-of-the-art, API-accessible models, with gemini-2.0-flash and gpt-4.1-mini achieving the highest overall scores of 0.75 and 0.72, respectively. The evaluation also uncovered specific model strengths and weaknesses; for instance, gpt-4.1-mini excelled in the Human-Machine Interaction category due to its superior grasp of human pose and spatial awareness, while gpt-4o-mini showed a surprising and significant deficit in spatial awareness tasks. These findings were instrumental in selecting the most appropriate models for our pipeline's captioning and summarization modules.

Finally, the end-to-end summarization pipeline was successfully implemented and tested on real-world video footage of a CNC machine in operation. The system achieved an overall performance score of 0.72 in a "Model-as-a-Judge" evaluation, demonstrating its ability to produce coherent, chronologically accurate summaries. More importantly, the pipeline proved

effective at its primary goal of anomaly detection, successfully identifying critical safety violations such as an attempt to operate the machine with the emergency stop engaged or the door open. While the system showed some limitations, such as occasionally generating overly generic descriptions or missing fine-grained procedural details, its success serves as a strong proof of concept.

7.2 Future Work

The developed prototype establishes a solid foundation, yet several exciting avenues exist for enhancing its capabilities.

7.2.1 Context-Aware RAG for Domain-Specific Knowledge

The current pipeline's context is provided through hard-coded instructions in the prompt. A major advancement would be to implement a Context-Aware RAG system. This would involve creating a vector database populated with domain-specific documentation, such as machine operation manuals, safety protocols, and standard operating procedures. When analyzing a video, the system could retrieve relevant passages from this knowledge base to provide the reasoning model with deep, explicit context. This would empower the model to detect subtle procedural deviations and generate alerts with much higher precision and detail, directly addressing the weakness of occasional generic summaries.

7.2.2 Fine-Tuning for Specialized Object Detection

The pipeline currently utilizes a general-purpose YOLOv11 model. While highly capable, it is not optimized to recognize custom tools or specific machine components. Future work should involve fine-tuning an object detection model on a custom-annotated dataset of objects specific to the target industrial environment. This would yield a more accurate and relevant list of detected objects for each frame.

7.2.3 Incorporating Audio as an Input Modality

Industrial environments are rich with auditory information that is currently ignored by our vision-only system. The next logical step is to make the system truly multimodal by adding an audio-processing module. This module could perform two key functions:

- **Speech-to-Text Transcription:** Transcribing any spoken words from the operator to capture verbal commands or comments.
- **Sound Event Classification:** Identifying and classifying non-speech sounds, such as the normal operating sound of a machine, a sudden clank indicating a dropped tool, or a grinding noise suggesting mechanical failure.

Bibliography

- [1] Simran Arora et al. *Ask Me Anything: A simple strategy for prompting language models*. 2022. arXiv: 2210.02441 [cs.CL]. url: <https://arxiv.org/abs/2210.02441>.
- [2] Zechen Bai et al. *Hallucination of Multimodal Large Language Models: A Survey*. 2025. arXiv: 2404.18930 [cs.CV]. url: <https://arxiv.org/abs/2404.18930>.
- [3] Aanisha Bhattacharya et al. *A Video Is Worth 4096 Tokens: Verbalize Videos To Understand Them In Zero Shot*. 2023. arXiv: 2305.09758 [cs.CV]. url: <https://arxiv.org/abs/2305.09758>.
- [4] Jing Bi et al. *MISAR: A Multimodal Instructional System with Augmented Reality*. 2023. arXiv: 2310.11699 [cs.CL]. url: <https://arxiv.org/abs/2310.11699>.
- [5] Grazia Cicirelli et al. "The HA4M dataset: Multi-Modal Monitoring of an assembly task for Human Action recognition in Manufacturing". In: *Scientific Data* 9.1 (2022), p. 745. issn: 2052-4463. doi: 10.1038/s41597-022-01843-z. url: <https://doi.org/10.1038/s41597-022-01843-z>.
- [6] Kevin Clark et al. *ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators*. 2020. arXiv: 2003.10555 [cs.CL]. url: <https://arxiv.org/abs/2003.10555>.
- [7] Mejdi DALLEL et al. "InHARD - Industrial Human Action Recognition Dataset in the Context of Industrial Collaborative Robotics". In: *2020 IEEE International Conference on Human-Machine Systems (ICHMS)*. 2020, pp. 1–6. doi: 10.1109/ICHMS49158.2020.9209531.
- [8] D. L. DeMott. "Human Reliability and the Cost of Doing Business". In: *Proceedings of the Annual Maintenance and Reliability Symposium*. Document ID: 20140008715, Report Number: JSC-CN-31348, Acquisition Source: Johnson Space Center. Society for Maintenance and Reliability Professionals. Galveston, TX, United States, Aug. 2014. url: <https://ntrs.nasa.gov/api/citations/20140008715/downloads/20140008715.pdf>.
- [9] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL]. url: <https://arxiv.org/abs/1810.04805>.
- [10] Wenkai Dong, Zhaoxiang Zhang, and Tieniu Tan. "Attention-Aware Sampling via Deep Reinforcement Learning for Action Recognition". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01 (July 2019), pp. 8247–8254. doi: 10.1609/aaai.v33i01.33018247. url: <https://ojs.aaai.org/index.php/AAAI/article/view/4836>.
- [11] Hehe Fan et al. "Watching a Small Portion could be as Good as Watching All: Towards Efficient Video Classification". In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, July 2018, pp. 705–711. doi: 10.24963/ijcai.2018/98. url: <https://doi.org/10.24963/ijcai.2018/98>.

- [12] Yunfan Gao et al. *Retrieval-Augmented Generation for Large Language Models: A Survey*. 2024. arXiv: 2312.10997 [cs.CL]. url: <https://arxiv.org/abs/2312.10997>.
- [13] Shreyank N Gowda, Marcus Rohrbach, and Laura Sevilla-Lara. "SMART Frame Selection for Action Recognition". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.2 (May 2021), pp. 1451–1459. doi: 10.1609/aaai.v35i2.16235. url: <https://ojs.aaai.org/index.php/AAAI/article/view/16235>.
- [14] Suchin Gururangan et al. *Variational Pretraining for Semi-supervised Text Classification*. 2019. arXiv: 1906.02242 [cs.CL]. url: <https://arxiv.org/abs/1906.02242>.
- [15] Zeyu Han et al. *Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey*. 2024. arXiv: 2403.14608 [cs.LG]. url: <https://arxiv.org/abs/2403.14608>.
- [16] Satoshi Hashimoto et al. *Anomaly Detection Based on Deep Learning Using Video for Prevention of Industrial Accidents*. 2020. arXiv: 2005.13734 [cs.CV]. url: <https://arxiv.org/abs/2005.13734>.
- [17] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL]. url: <https://arxiv.org/abs/2106.09685>.
- [18] Jingwen Hu et al. *Dense v.s. Sparse: A Comparative Study of Sampling Analysis in Scene Classification of High-Resolution Remote Sensing Imagery*. 2015. arXiv: 1502.01097 [cs.CV]. url: <https://arxiv.org/abs/1502.01097>.
- [19] Nidhal Jegham et al. *Evaluating the Evolution of YOLO (You Only Look Once) Models: A Comprehensive Benchmark Study of YOLO11 and Its Predecessors*. 2024. arXiv: 2411.00201 [cs.CV]. url: <https://arxiv.org/abs/2411.00201>.
- [20] Yu-Gang Jiang et al. "Exploiting Feature and Class Relationships in Video Categorization with Regularized Deep Neural Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.2 (Feb. 2018), pp. 352–364. issn: 2160-9292. doi: 10.1109/tpami.2017.2670560. url: <http://dx.doi.org/10.1109/TPAMI.2017.2670560>.
- [21] Zhengbao Jiang et al. "How Can We Know What Language Models Know?" In: *Transactions of the Association for Computational Linguistics* 8 (July 2020), pp. 423–438. issn: 2307-387X. doi: 10.1162/tac1_a_00324. eprint: https://direct.mit.edu/tac1/article-pdf/doi/10.1162/tac1_a_00324/1923867/tac1_a_00324.pdf. url: https://doi.org/10.1162/tac1%5C_a%5C_00324.
- [22] Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. *AMMUS : A Survey of Transformer-based Pretrained Models in Natural Language Processing*. 2021. arXiv: 2108.05542 [cs.CL]. url: <https://arxiv.org/abs/2108.05542>.
- [23] Ravpreet Kaur and Sarbjeet Singh. "A comprehensive review of object detection with deep learning". In: *Digital Signal Processing* 132 (2023), p. 103812. issn: 1051-2004. doi: <https://doi.org/10.1016/j.dsp.2022.103812>. url: <https://www.sciencedirect.com/science/article/pii/S1051200422004298>.
- [24] Bruno Korbar, Du Tran, and Lorenzo Torresani. "SCSampler: Sampling Salient Clips From Video for Efficient Action Recognition". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.
- [25] Yuri Kuratov and Mikhail Arkhipov. *Adaptation of Deep Bidirectional Multilingual Transformers for Russian Language*. 2019. arXiv: 1905.07213 [cs.CL]. url: <https://arxiv.org/abs/1905.07213>.

- [26] Mingyue Liu et al. "Deep-Learning-Based Pre-Training and Refined Tuning for Web Summarization Software". In: *IEEE Access* 12 (2024), pp. 92120–92129. doi: 10.1109/ACCESS.2024.3423662.
- [27] Pengfei Liu et al. *Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing*. 2021. arXiv: 2107.13586 [cs.CL]. url: <https://arxiv.org/abs/2107.13586>.
- [28] Yang Liu et al. *G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment*. 2023. arXiv: 2303.16634 [cs.CL]. url: <https://arxiv.org/abs/2303.16634>.
- [29] Swaroop Mishra et al. *Reframing Instructional Prompts to GPTk's Language*. 2022. arXiv: 2109.07830 [cs.CL]. url: <https://arxiv.org/abs/2109.07830>.
- [30] Humza Naveed et al. *A Comprehensive Overview of Large Language Models*. 2024. arXiv: 2307.06435 [cs.CL]. url: <https://arxiv.org/abs/2307.06435>.
- [31] Jim Nilsson and Tomas Akenine-Möller. *Understanding SSIM*. 2020. arXiv: 2006.13846 [eess.IV]. url: <https://arxiv.org/abs/2006.13846>.
- [32] OpenAI et al. *GPT-4o System Card*. 2024. arXiv: 2410.21276 [cs.CL]. url: <https://arxiv.org/abs/2410.21276>.
- [33] Colin Raffel et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67. url: <http://jmlr.org/papers/v21/20-074.html>.
- [34] Francesco Ragusa, Antonino Furnari, and Giovanni Maria Farinella. "MECCANO: A multimodal egocentric dataset for humans behavior understanding in the industrial-like domain". In: *Computer Vision and Image Understanding* 235 (2023), p. 103764. issn: 1077-3142. doi: <https://doi.org/10.1016/j.cviu.2023.103764>. url: <https://www.sciencedirect.com/science/article/pii/S1077314223001443>.
- [35] Ranjan Sapkota et al. *YOLOv10 to Its Genesis: A Decadal and Comprehensive Review of The You Only Look Once (YOLO) Series*. 2024. arXiv: 2406.19407 [cs.CV]. url: <https://arxiv.org/abs/2406.19407>.
- [36] Tempest A. van Schaik and Brittany Pugh. "A Field Guide to Automatic Evaluation of LLM-Generated Summaries". In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '24. Washington DC, USA: Association for Computing Machinery, 2024, pp. 2832–2836. isbn: 9798400704314. doi: 10.1145/3626772.3661346. url: <https://doi.org/10.1145/3626772.3661346>.
- [37] Tobias Schlagenhauf and Magnus Landwehr. "Industrial machine tool component surface defect dataset". In: *Data in Brief* 39 (Dec. 2021), p. 107643. issn: 2352-3409. doi: 10.1016/j.dib.2021.107643. url: <http://dx.doi.org/10.1016/j.dib.2021.107643>.
- [38] Fadime Sener et al. *Assembly101: A Large-Scale Multi-View Video Dataset for Understanding Procedural Activities*. 2022. arXiv: 2203.14712 [cs.CV]. url: <https://arxiv.org/abs/2203.14712>.
- [39] Aakash K. Shetty et al. "A Review: Object Detection Models". In: *2021 6th International Conference for Convergence in Technology (I2CT)*. 2021, pp. 1–8. doi: 10.1109/I2CT51068.2021.9417895.
- [40] Mupparaju Sohan, Thotakura Ram, and Venkata Ch. "A Review on YOLOv8 and Its Advancements". In: Jan. 2024, pp. 529–545. isbn: 978-981-99-7999-8. doi: 10.1007/978-981-99-7962-2_39.
- [41] Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. "BERTimbau: Pretrained BERT Models for Brazilian Portuguese". In: *Intelligent Systems: 9th Brazilian Conference, BRACIS 2020, Rio Grande, Brazil, October 20–23, 2020, Proceedings, Part I*. Rio

- Grande, Brazil: Springer-Verlag, 2020, pp. 403–417. isbn: 978-3-030-61376-1. doi: 10.1007/978-3-030-61377-8_28. url: https://doi.org/10.1007/978-3-030-61377-8_28.
- [42] Sruti Srinidhi, Edward Lu, and Anthony Rowe. “XaiR: An XR Platform that Integrates Large Language Models with the Physical World”. In: *2024 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2024, pp. 759–767. doi: 10.1109/ISMAR62088.2024.00091. url: <https://doi.ieeecomputersociety.org/10.1109/ISMAR62088.2024.00091>.
- [43] J. Suganthi and V. Abineshkumar. “Revolutionizing Anomaly Detection in Surveillance Footage with ARLSTM”. In: *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*. 2023, pp. 506–511. doi: 10.1109/ICAAIC56838.2023.10141056.
- [44] Gemma Team et al. *Gemma 3 Technical Report*. 2025. arXiv: 2503.19786 [cs.CL]. url: <https://arxiv.org/abs/2503.19786>.
- [45] Ultralytics. *YOLO11 NEW — docs.ultralytics.com*. <https://docs.ultralytics.com/models/yolo11/>. [Accessed 10-11-2024].
- [46] Ao Wang et al. *YOLOv10: Real-Time End-to-End Object Detection*. 2024. arXiv: 2405.14458 [cs.CV]. url: <https://arxiv.org/abs/2405.14458>.
- [47] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. *YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information*. 2024. arXiv: 2402.13616 [cs.CV]. url: <https://arxiv.org/abs/2402.13616>.
- [48] Hongru Wang et al. “M3sum: A Novel Unsupervised Language-Guided Video Summarization”. In: *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2024, pp. 4140–4144. doi: 10.1109/ICASSP48485.2024.10447504.
- [49] Zhenhailong Wang et al. *Language Models with Image Descriptors are Strong Few-Shot Video-Language Learners*. 2022. arXiv: 2205.10747 [cs.CV]. url: <https://arxiv.org/abs/2205.10747>.
- [50] Wenhao Wu et al. “Multi-Agent Reinforcement Learning Based Frame Sampling for Effective Untrimmed Video Recognition”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 6221–6230. doi: 10.1109/ICCV.2019.00632.
- [51] Zihui Xue et al. *Progress-Aware Video Frame Captioning*. 2024. arXiv: 2412.02071 [cs.CV]. url: <https://arxiv.org/abs/2412.02071>.
- [52] Jiarong Yao, Kaixu Li, and Rong Su. “HUMM: A New Dataset for Human Movement in Manufacturing”. In: *2023 8th International Conference on Robotics and Automation Engineering (ICRAE)*. 2023, pp. 287–292. doi: 10.1109/ICRAE59816.2023.10458665.
- [53] Serena Yeung et al. *End-to-end Learning of Action Detection from Frame Glimpses in Videos*. 2017. arXiv: 1511.06984 [cs.CV]. url: <https://arxiv.org/abs/1511.06984>.
- [54] Shukang Yin et al. “A survey on multimodal large language models”. In: *National Science Review* 11.12 (Nov. 2024). issn: 2053-714X. doi: 10.1093/nsr/nwae403. url: <http://dx.doi.org/10.1093/nsr/nwae403>.
- [55] Weihao Yu et al. “MM-Vet: evaluating large multimodal models for integrated capabilities”. In: *Proceedings of the 41st International Conference on Machine Learning. ICML'24*. Vienna, Austria: JMLR.org, 2024.
- [56] Xiaohua Zhai et al. *Sigmoid Loss for Language Image Pre-Training*. 2023. arXiv: 2303.15343 [cs.CV]. url: <https://arxiv.org/abs/2303.15343>.

-
- [57] Zhengyan Zhang et al. *CPM-2: Large-scale Cost-effective Pre-trained Language Models*. 2021. arXiv: 2106.10715 [cs.CL]. url: <https://arxiv.org/abs/2106.10715>.
 - [58] Hao Zheng, Regina Lee, and Yuqian Lu. "HA-ViD: A Human Assembly Video Dataset for Comprehensive Assembly Knowledge Understanding". In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 67069–67081. url: https://proceedings.neurips.cc/paper_files/paper/2023/file/d40e6e4b3ee6c24f2bf2cb72c2412f4b-Paper-Datasets_and_Benchmarks.pdf.
 - [59] Ce Zhou et al. *A Comprehensive Survey on Pretrained Foundation Models: A History from BERT to ChatGPT*. 2023. arXiv: 2302.09419 [cs.AI]. url: <https://arxiv.org/abs/2302.09419>.
 - [60] Zhengxia Zou et al. *Object Detection in 20 Years: A Survey*. 2023. arXiv: 1905.05055 [cs.CV]. url: <https://arxiv.org/abs/1905.05055>.