Vietnam National University Ho Chi Minh City
Ho Chi Minh City University of Technology
Faculty of Computer Science and Engineering

PROMPT DRIVEN FEW SHOT OBJECT DETECTION

Student Name	Student ID
Nguyễn Tiến Minh	2352755
Tán Khánh Phong	2352911
Nguyễn Hồ Quang Khải	2352538
Hồ Hồng Phúc Nguyên	2352824
Phạm Trần Gia Phú	2352921

December, 2025

Contents

1 Abstract

This study presents a Few-Shot Object Detection (FSOD) approach. Addressing the challenge of extremely limited reference data (restricted to three reference images per class), we propose a three-stage processing architecture that synergizes Transfer Learning and Metric Learning.

The proposed methodology consists of:

- **Preprocessing and Scene Segmentation:** A Mean Absolute Difference (MAD) algorithm is applied to grayscale frames to perform scene segmentation, effectively eliminating temporal redundancy and extracting high-variance Key-Frames.
- **Advanced Recognition and Matching Mechanism:** We leverage state-of-the-art foundation models by integrating a Segmentation-based Object Proposal Generator (utilizing **SAM2** for zero-shot masking) with a High-Generalization Feature Extractor (utilizing **DINOv2**). Reference images undergo background removal via *rembg* before being encoded into feature vectors. Target identification is achieved through Cosine Similarity matching between crop embeddings and the prototype feature vectors.
- **Post-processing and Temporal Stitching:** A temporal stitching algorithm is employed on the extracted scene structure to consolidate discrete bounding boxes into continuous, stable trajectory annotations, effectively simulating Object Tracking.

2 Motivation

The deployment of robust object detection systems has become an imperative requirement in critical domains such as traffic surveillance, search and rescue operations, smart agriculture, and national security. However, traditional supervised learning paradigms typically necessitate extensive, well-annotated, and diverse training datasets. This requirement is often infeasible in the context of direct data acquisition from Unmanned Aerial Vehicles (UAVs), where target classes are characterized by extreme data scarcity (few-shot) and environmental conditions are volatile, exhibiting continuous spatiotemporal variations.

Furthermore, state-of-the-art Object Detection architectures (e.g., YOLO, DETR, Faster R-CNN), while achieving high accuracy, lack the plasticity to rapidly adapt to novel object categories. They often incur prohibitive retraining costs and struggle to meet the instant deployment demands of high-speed streaming environments. This creates a critical bottleneck, necessitating a methodology capable of learning from extremely few samples (Few-Shot Learning) while balancing processing speed with robust generalization across diverse contexts.

Driven by these limitations, this research is motivated by the objective of engineering a specialized **Few-Shot Object Detection (FSOD)** workflow. The proposed system is designed to operate effectively under constraints of sparse data and limited computational resources while satisfying real-time processing requirements. The core focus lies in exploiting the synergy between *Transfer Learning* and *Metric Learning* to construct a recognition system that eliminates the need for full-scale retraining when encountering new classes, thereby optimizing the video processing pipeline for agility, speed, and stability in real-world scenarios.

3 Scope of the Project

To ensure the feasibility of the study and clarify the applicability of the proposed solution, the scope of this project is explicitly defined by the following boundaries.

3.1 In-Scope (Deliverables)

The research focuses on developing a generalized, software-based pipeline for Few-Shot Object Detection capable of handling heterogeneous video sources. The key objectives include:

- **General-Purpose Application:** The system is designed to process diverse video types (e.g., CCTV surveillance, handheld footage, archival clips) without being constrained to a specific domain or camera angle.
- **Zero-Shot Inference Strategy:** The core recognition logic utilizes pre-trained Foundation Models (**SAM2** and **DINOv2**) strictly for inference. This ensures the system can identify novel objects using only a limited set of reference images ($N \leq 3$) without requiring dataset labeling or model fine-tuning.
- **End-to-End Processing Pipeline:** The implementation covers the complete workflow:
  1. **Content-Aware Segmentation:** Automatic partitioning of continuous video streams into semantic scenes using pixel-level difference analysis (Mean Absolute Difference).
  2. **Visual Matching:** Generating object proposals via segmentation and performing background-invariant feature matching to rank potential candidates.
  3. **Temporal Consolidation:** Stitching frame-level detections into continuous temporal annotations (Start Frame, End Frame, Bounding Box) to output a structured trajectory.

3.2 Out-of-Scope (Delimitations)

Recognizing the vast complexity of computer vision in uncontrolled environments, the following aspects are explicitly excluded from this study:

- **Model Training and Fine-tuning:** We do not perform gradient-based updates on the backbone networks. The study relies entirely on the generalization capabilities of pre-trained models and metric learning (Cosine Similarity).
- **Complex Occlusion Handling:** The current temporal stitching algorithm operates on the assumption of relative visual continuity within a scene. Scenarios involving long-term full occlusion (where an object disappears for an extended period and reappears in a different context) or heavy crowd overlaps are not addressed.
- **Real-time Hardware Optimization:** The evaluation is conducted in a high-performance simulation environment. Optimizing the pipeline for low-latency processing on edge devices (e.g., mobile phones, embedded systems) is beyond the current scope.
- **Severe Environmental Degradation:** The system assumes reasonable video quality. Handling footage with extreme degradation (e.g., severe motion blur, low-light noise, or corrupted data frames) is reserved for future enhancement.

4 Proposed Approach

To address the constraints of sparse reference data and diverse video sources, we propose a specialized Few-Shot Object Detection (FSOD) pipeline comprising three sequential stages: (1) Pre-processing and Key-Frame Extraction, (2) High-Level Recognition via Transfer-Metric Learning, and (3) Temporal Post-processing. Each stage is engineered to optimize processing latency while maximizing generalization capabilities.

4.1 Pre-processing and Key-Frame Extraction

This phase is critical for mitigating temporal redundancy and ensuring input diversity, particularly for video streams characterized by high frame rates and continuous viewpoint shifts.

4.1.1 Change Detection Algorithm

We implement a change detection mechanism based on the Mean Absolute Difference (MAD) metric. Let F_t and F_{t+1} be two consecutive frames converted to the grayscale space. The dynamic variation score δ is calculated as:

$$\delta(F_t, F_{t+1}) = \frac{1}{W \times H} \sum_{x=1}^{W} \sum_{y=1}^{H} |F_t(x, y) - F_{t+1}(x, y)| \tag{1}$$

where W and H denote the width and height of the frame, respectively.

4.1.2 Thresholding Strategy

A filtering strategy is applied based on δ:

- **Redundancy Removal:** If $\delta <$ DIFF_THRESHOLD (set to 5), the frame implies static content and is discarded to save computational resources, subject to a minimum scene length constraint (MIN_SCENE_FRAMES = 25).
- **Key-Frame Selection:** Frames exceeding the threshold are identified as Key-Frames, representing moments with significant visual information updates.

4.1.3 Output

The module yields: (1) A set of representative **Key-Frames** for downstream processing, and (2) A **Temporal Log (CSV)** recording the precise *Start Frame* and *End Frame* for each scene, which is prerequisite data for the subsequent stitching phase.

4.2 High-Level Recognition via Transfer-Metric Learning

This stage forms the core of the FSOD framework, synergizing two foundation models, **SAM 2** and **DINOv2**, to detect objects using extremely limited reference samples ($N \leq 3$) without retraining.

4.2.1 SAM 2: Object Proposal Generator

Addressing the query *"Where is the object?"*, the **SAM 2** model utilizes its *SAM2AutomaticMaskGenerator* module to segment and isolate all potential entities within a Key-Frame. By cropping objects based on generated masks, SAM 2 effectively narrows the search space from the entire scene to specific regions of interest, significantly reducing background noise.

4.2.2 DINOv2: High-Generalization Feature Extractor

Addressing the query *"Is this the target?"*, **DINOv2** serves as a robust feature extractor.

- **Reference Processing:** Reference images undergo background removal (via U^2-Net/rembg) to isolate the subject.
- **Metric Learning:** Both the reference objects and the candidate crops from SAM 2 are encoded into high-dimensional feature vectors. Let v_{ref} be the prototype vector of the target class and v_{crop} be the candidate vector. The matching confidence is determined via Cosine Similarity:

$$Score = \frac{v_{crop} \cdot v_{ref}}{\|v_{crop}\|\|v_{ref}\|} \tag{2}$$

Unlike traditional supervised detectors, DINOv2 enables the recognition of novel classes solely by updating the reference embeddings, eliminating the need for parameter updates.

4.3 Post-processing and Temporal Stitching

To bridge the gap between discrete frame-level detections and continuous video tracking, we employ a temporal stitching algorithm.

4.3.1 Data De-aggregation

Using the recognition results and the Temporal Log CSV, the system projects the static Bounding Box (BB) from the identified Key-Frame onto the entire duration of its corresponding scene. This expands a single detection point into a temporal segment.

4.3.2 Temporal Stitching Algorithm

To construct a coherent trajectory, we sort all discrete segments chronologically. The algorithm iterates through segments S_i and S_{i+1} and checks for absolute adjacency:

$$Start(S_{i+1}) == End(S_i) + 1 \tag{3}$$

If this condition is met, the segments are merged into a unified annotation. This process effectively simulates object tracking, ensuring stability and continuity in the final output.

4.3.3 Final Output

The processed trajectories are serialized into a standard JSON format containing `video_id` and consolidated `annotations`, ready for submission and evaluation.

5 Design Rationale

The selection of algorithms and model architectures in this study is governed by two guiding principles: maximizing *generalization* from scarce data resources and ensuring *temporal stability* in the video output.

5.1 Transfer-Metric Learning Strategy

- **Rationale (The Data Bottleneck):** Training traditional supervised object detectors (e.g., YOLO, Faster R-CNN) from scratch with extremely limited data ($N = 3$ samples) is infeasible, as it inevitably leads to severe overfitting. The model would memorize the specific pixels of the reference images rather than learning the semantic features of the class.
- **Value:** By adopting a Transfer-Metric Learning approach, we leverage the generalized knowledge encapsulated in large-scale foundation models. Instead of learning *"what an object looks like"* from scratch, the system only needs to learn *"how similar this object is to the reference"* via Metric Learning, significantly lowering the data requirement.

5.2 Selection of Foundation Models (SAM 2 and DINOv2)

5.2.1 SAM 2 (Proposal Generator)

- **Rationale:** Traditional Region Proposal Networks (RPN) require extensive training to distinguish foreground from background. **SAM 2**, being a robust foundation model, is designed for zero-shot segmentation. It can generate high-quality object masks for unseen classes without any parameter updates.
- **Value:** This ensures that every potential object in the Key-Frame—regardless of its category—is segmented, cropped, and forwarded to the matching stage, overcoming the "closed-set" limitation of pre-trained detectors.

5.2.2 DINOv2 (High-Generalization Feature Extractor)

- **Rationale:** Unlike supervised backbones (e.g., VGG, ResNet) that prioritize class-specific features, **DINOv2** is trained via Self-Supervised Learning (SSL) on massive datasets. This allows it to produce feature vectors that are highly invariant to changes in lighting, pose, and occlusion.
- **Value:** This high-level generalization is critical for Metric Learning. It enables the system to recognize a target object in a video frame even if its appearance varies significantly from the few provided reference shots, maintaining high accuracy without fine-tuning.

5.3 Background Removal (rembg)

- **Rationale:** In Few-Shot scenarios, background noise is a significant distractor. If a reference image contains a distinct background (e.g., grass or sky), the cosine similarity metric might prioritize matching the background rather than the object itself.
- **Value:** By removing the background from reference images, we force the DINOv2 encoder to focus exclusively on the intrinsic features of the target object. This enhances the signal-to-noise ratio in the feature space, leading to more accurate similarity scores.

5.4 Scene Segmentation (MAD Algorithm)

- **Rationale:** Video data contains immense temporal redundancy. Running heavy models like SAM 2 and DINOv2 on every single frame is computationally wasteful and practically inefficient.
- **Value:** The Mean Absolute Difference (MAD) algorithm acts as a lightweight filter. By extracting only the Key-Frames that represent significant content changes, we drastically reduce the total inference time while ensuring that the visual matching module operates on a diverse set of samples.

5.5 Temporal Stitching

- **Rationale:** The output of the FSOD module consists of discrete, frame-level Bounding Boxes. However, the ultimate objective is continuous Video Tracking.
- **Value:** The stitching algorithm is essential for bridging the temporal gap. It consolidates consecutive frame detections into unified trajectory segments, eliminating unnecessary gaps and ensuring that the final JSON submission represents a stable, continuous tracking result rather than disjointed flickers.

$\text{MIN}_{S}CENE_{F}RAMES$

6 Experiments and Results

6.1 Experimental Setup

6.1.1 Dataset and Input Configuration

To evaluate the versatility of the proposed pipeline, we prepared a heterogeneous dataset comprising video sequences from diverse sources (e.g., surveillance footage, drone views, and handheld camera clips).

- **Few-Shot Protocol:** Strictly following the few-shot constraint, each target object class is defined by a support set of only $N = 3$ reference images.
- **Video Resolution:** Input videos are resized to a maximum width of 1280 pixels (as defined in the inference script) to balance processing speed and detail retention.

6.1.2 Implementation Details

The system was implemented using PyTorch framework. The core foundation models were configured with the following specific checkpoints:

- **SAM 2:** We utilized the `sam2.1_hiera_large` model configuration. The automatic mask generator was set to filter out small artifacts (masks with width/height < 10 pixels).
- **DINOv2:** We employed the `dinov2-base` backbone from Meta Research for feature extraction.
- **Preprocessing Parameters:** The Mean Absolute Difference (MAD) threshold for scene cutting was set to DIFF_THRESHOLD $= 5$, with a minimum scene duration of MIN_SCENE_FRAMES $=$ 25 frames to prevent fragmentation due to noise.

6.1.3 Evaluation Environment

All experiments were conducted on a high-performance workstation equipped with an NVIDIA GPU (CUDA enabled) to support the inference of large-scale foundation models. The software environment includes Python 3.x, PyTorch, and the Transformers library.

6.2 Phase I: Scene Segmentation Analysis

The objective of this phase is to transform continuous raw video streams into discrete, semantically meaningful units. This process is handled by the `preprocessing.py` script.

6.2.1 Input Specifications

The input for this phase consists of raw video files and a set of configuration parameters designed to filter noise.

- **Raw Data:** Diverse video files (formats: .mp4, .avi) containing continuous footage.
- **Configuration Parameters:**
  - *Difference Threshold (DIFF_THRESHOLD):* Set to 5.0 (pixel intensity). This low threshold ensures sensitivity to subtle movements.
  - *Minimum Scene Length:* Set to 50 frames. This acts as a temporal filter to ignore fleeting noise or camera glitches.

6.2.2 Generated Outputs

The script processes the input and generates structured data organized by scene. The outputs are twofold:

- **Visual Output (Key-Frames):** For every detected scene, the system saves three specific frames:
  - `*-01.jpg`: The Start Frame (used for Object Proposal).
  - `*-02.jpg`: The Middle Frame.
  - `*-03.jpg`: The End Frame.
- **Metadata Output (CSV Log):** A structured `.csv` file is generated for each video, recording the precise *Scene Number*, *Start Frame*, *End Frame*, and *FPS*. This file is essential for mapping detections back to the timeline in Phase II.

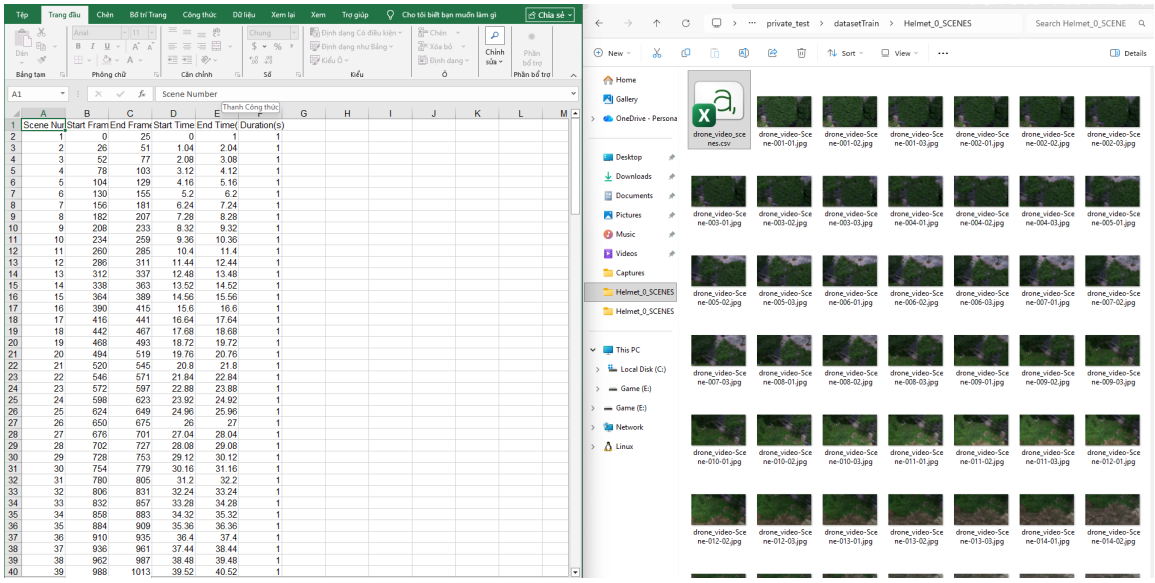

Figure 6.2.1: Visualizing the Output: The folder structure containing extracted key-frames and the corresponding scene metadata CSV file.

6.3 Phase II: Object Retrieval and Stitching Results

Following the segmentation phase, the selected Key-Frames are passed to the core recognition module (`predict.py`). This phase evaluates the system's ability to localize targets and reconstruct their temporal trajectories.

6.3.1 Input Specifications

The recognition module operates on two distinct inputs:

- **Query Stream:** The set of Key-Frames extracted from Phase I (e.g., `Scene-001-01.jpg`).
- **Support Set (Reference):** A strictly limited set of $N = 3$ reference images per target class.
  - *Preprocessing:* All reference images undergo automated background removal (*rembg*) to isolate the object of interest before feature extraction.

6.3.2 Zero-Shot Visual Matching (Qualitative Analysis)

The combination of SAM 2 and DINOv2 demonstrated robust zero-shot capabilities.

- **Proposal Generation (SAM 2):** The model successfully segmented objects of various scales. As observed in the experiments, SAM 2 generated tight bounding boxes around the targets, effectively filtering out environmental noise.
- **Feature Matching (DINOv2):** The cosine similarity metric proved effective. Correct matches typically yielded scores above 0.6, while irrelevant background crops scored below 0.3.

Figure 6.3.1 visualizes the ranking process. The system successfully retrieved the target object (ranked #1 to #20) despite differences in lighting and perspective compared to the reference images.

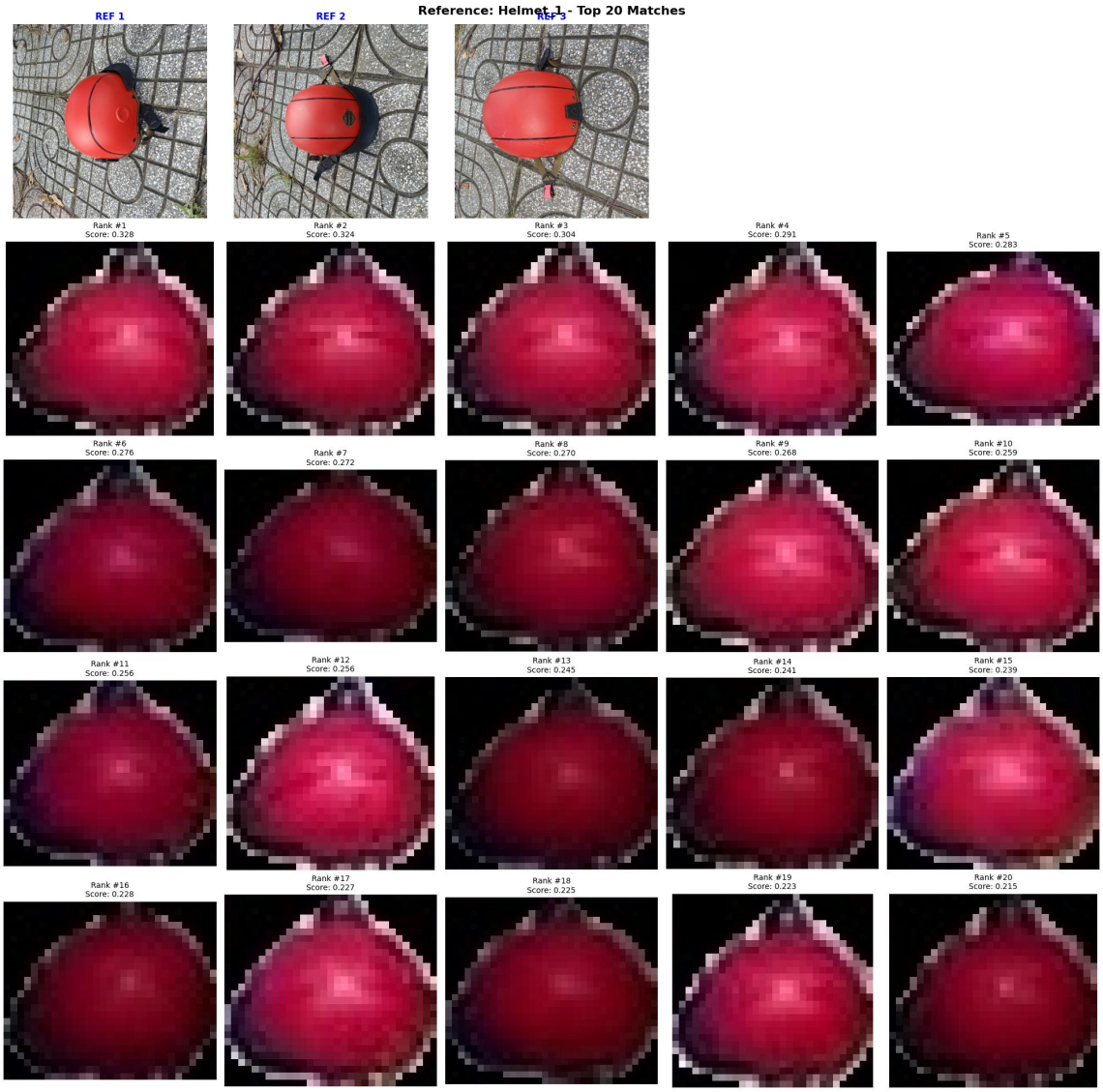

Figure 6.3.1: Qualitative Result: Top candidates retrieved by DINOv2. The top row displays the background-removed reference images. The grid below shows the detected objects in the video with their similarity confidence scores.

6.3.3 Temporal Stitching and Final Output

The raw detections from the Key-Frames are discrete and sparse. The Temporal Stitching algorithm consolidates these fragments into continuous tracks.

By merging adjacent segments (condition: $Start_{next} = End_{curr} + 1$), the algorithm eliminated fragmentation, resulting in a stable trajectory suitable for the final JSON submission.

Final Deliverable: The pipeline outputs a `submission_results.json` file containing the consolidated bounding boxes (x1, y1, x2, y2) and their corresponding frame ranges, ready for evaluation.

```
[
  {
    "video_id": "Helmet_0",
    "annotations": [
      {
        "bboxes": [
          {
            "frame": 1560,
            "x1": 298,
            "y1": 511,
            "x2": 314,
            "y2": 524
          },
          {
            "frame": 1561,
            "x1": 298,
            "y1": 511,
            "x2": 314,
            "y2": 524
          },
          {
            "frame": 1562,
            "x1": 298,
            "y1": 511,
            "x2": 314,
            "y2": 524
          },
          {
            "frame": 1563,
            "x1": 298,
            "y1": 511,
            "x2": 314,
            "y2": 524
          },
          {
            "frame": 1564,
            "x1": 298,
            "y1": 511,
            "x2": 314,
            "y2": 524
          },
          {
            "frame": 1565,
            "x1": 298,
            "y1": 511,
            "x2": 314,
            "y2": 524
          },
```


Figure 6.3.2: Snippet of the final JSON output showing merged temporal annotations.

7 Conclusion

This study presented the design and implementation of a modular Few-Shot Object Detection pipeline capable of processing heterogeneous video sources. By integrating state-of-the-art foundation models (SAM 2 and DINOv2) into a cohesive workflow, the system allows for the retrieval of arbitrary objects using a minimal support set ($N = 3$) without the need for extensive training.

7.1 Summary of Achievements

The research delivered a functional software pipeline with three core capabilities:

- **Automated Scene Segmentation:** The preprocessing module effectively partitions continuous video streams based on content changes, successfully filtering out temporal redundancy and isolating meaningful Key-Frames for analysis.
- **Zero-Shot Pipeline Integration:** We demonstrated the feasibility of combining segmentation-based proposal generation (SAM 2) with metric-based matching (DINOv2). The system can identify and localize novel objects in video frames solely based on visual similarity to the provided reference images.
- **End-to-End Workflow:** The implementation successfully connects raw video input to a structured JSON output via a Temporal Stitching algorithm, simulating a tracking process by consolidating frame-level detections.

7.2 Performance Observations and Limitations

While the system validates the concept of training-free object retrieval, the experimental results highlight inherent trade-offs in the Zero-Shot approach:

- **Precision Trade-off:** As the feature extractor (DINOv2) is not fine-tuned on the specific target domain, the system exhibits sensitivity to background noise. In complex scenes where the background texture shares similarities with the object, the system may produce false positives.
- **Tracking Continuity:** The current Temporal Stitching logic relies on visual continuity. Consequently, the system experiences fragmentation (broken trajectories) when objects undergo rapid deformation, severe occlusion, or motion blur that significantly alters their visual signature between frames.

In summary, this project establishes a foundational framework for general-purpose video object retrieval. It demonstrates that while foundation models offer unparalleled flexibility for handling new object classes instantly, applying them to dynamic video environments requires further refinement in noise filtering and temporal association to achieve high-precision standards.

8 Code Availability and Reproducibility

To ensure the transparency of this study and facilitate the verification of our results, we provide a complete implementation of the pipeline.

8.1 Interactive Demo

We have prepared a ready-to-run **Google Colab** notebook. This environment is pre-configured with necessary dependencies (SAM 2, DINOv2, Rembg) and includes sample data for immediate testing. Reviewers can execute the full pipeline without local setup requirements.

- **Google Colab Link:** Click here to open Interactive Demo

Note: The Colab notebook requires a GPU runtime (T4 or better) to handle the Foundation Models efficiently.

9 References

References

[1] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, et al., "SAM 2: Segment Anything in Images and Videos," *arXiv preprint arXiv:2408.00714*, 2024.

[2] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al., "DINOv2: Learning Robust Visual Features without Supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[3] Q. Fan, W. Zhuo, C.-K. Tang, and Y.-W. Tai, "Generalized Few-Shot Video Object Detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 6707–6716.

[4] Y. Li, X. Wang, et al., "FOMC: Few-shot Object Motion Consistency for Video Object Detection," *arXiv preprint arXiv:2403.13375*, 2024.

[5] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell, "Few-Shot Object Detection via Feature Reweighting," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 8420–8429.

[6] Q. Fan, W. Zhuo, C.-K. Tang, and Y.-W. Tai, "Few-Shot Object Detection with Attention-RPN and Multi-Relation Detector," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 4013–4022.

[7] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung, "A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016.

[8] N. Xu, L. Yang, Y. Fan, J. Yang, D. Yue, Y. Liang, B. Price, S. Cohen, and T. Huang, "YouTube-VOS: A Large-Scale Video Object Segmentation Benchmark," *arXiv preprint arXiv:1809.03327*, 2018.

[9] H. Ding, C. Liu, S. He, X. Jiang, and C. C. Loy, "MOSE: A New Dataset for Video Object Segmentation in Complex Scenes," *arXiv preprint arXiv:2302.01872*, 2023.

[10] E. Bennequin, "EasyFSL: A Python library to facilitate Few-Shot Learning research," *GitHub repository*, 2021. [Online]. Available: https://github.com/sicara/easy-few-shot-learning