Open Set Image to Video Entity Matching

Ben Michael Taylor
BSc (Hons) Computer Science

A dissertation submitted for the degree of
Master of Science in Data Science

Supervised by *Dr Bryan Williams*

School of Computing and Communications
Lancaster University

September, 2025

Declaration

I declare that the work presented in this dissertation is, to the best of my knowledge and belief, original and my own work. The material has not been submitted, either in whole or in part, for a degree at this, or any other university.

Name: **Ben Michael Taylor**
Date: **September, 2025**

Open Set Image to Video Entity Matching
Ben Michael Taylor, BSc (Hons) Computer Science.
School of Computing and Communications, Lancaster University
A dissertation submitted for the degree of *Master of Science* in Data Science.
September, 2025

Abstract

Open set image to video entity matching describes the task of determining whether a reference image contains the same entity as one appearing in a video and retrieving the best frames from the video that contain the entity. This is done without relying on fixed classes meaning that the entity can be atypical such as badges or logos. This problem is of growing importance in the area of digital forensics where there is often large amounts of video data under various conditions to search through.

This project develops and evaluates two key methodologies for open set image video entity matching and systematically compares them to a set of baseline methods (SIFT, ORB, AKAZE, Faster R-CNN, YOLO, DETR, and a simple Siamese Network) with the use of a bespoke dataset created specifically with open set video forensics in mind. The two approaches are: (1) a Feature Pyramid Network (FPN) Siamese model that utilises multi-scale feature representations to improve performance on small objects, and (2) a proposal-verification pipeline that integrates CLIP and GroundingDINO for open-vocabulary proposal generation as well as a Siamese verification head for instance level matching.

Experimental results show that the two proposed methods improve upon baseline approaches, with the FPN Siamese achieving significant gains on small object retrieval and the proposal-verification pipeline promising strong open set generalisation at the cost of higher computational requirements. Analysis highlights the trade-offs between accuracy and efficiency while providing practical guidance for deployment in real world scenarios.

In summary, the dissertation contributes a new dataset (with 50 videos, 160 references, and frame level ground truths), two novel open set matching methods, and a comparative study of baselines. These contributions help to advance the wider research in image-video entity matching and retrieval and provide a foundation for practical security applications.

Acknowledgements

I would like to thank my academic supervisor Dr. Bryan Williams as well as my company supervisors Alan Crameri and Dr. Scott Stainton for their invaluable insights and guidance throughout this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Visual entity matching (VEM), the task of determining if two visual inputs are the same entity instance, has become increasingly important in many domains such as law enforcement, security, and digital forensics. The application of artificial intelligence (AI) driven approaches in the area of VEM can help automate laborious video search processes, improve detection consistency, and enable rapid search and retrieval of critical evidence. Entity matching can be applied to a plethora of areas such as facial recognition, vehicle re-identification, or landmark recognition. This dissertation develops two novel processes for open set image to video entity matching. The novel methods are implemented and compared to multiple baseline models. Model effectiveness is assessed on a custom dataset, designed with suggestions from the Collaboraite team, specifically to reflect real world forensic use.

1.1 Motivation and Importance of Entity Matching

Law enforcement, particularly forensic analysis, often requires the extremely time consuming and laborious process of collecting, analysing, and interpreting large amounts of digital data relating to a security incident. In an interview in 2014 former FBI Executive Assistant Director Stephanie Douglas stated that after the 2011 Boston Marathon attack "almost 13,000 different videos were obtained, (and) more than 120,000 still photographs" were collected with over 120 analysts assigned to search the footage[1]. Examples such as this demonstrate the large amount of data and manpower required for manual digital forensics.

Current object detection methods allow this process to be sped up. However, these methods often come with several limitations. Many models may be able to identify specific object categories, such as a bike, but may not be able to distinguish specific instances such as different bike models. Furthermore, models are often limited by the class types they are trained on and may struggle to recognise objects that lie outside of this scope. This highlights a need for open set, instance level entity matching techniques that can be applied to realistic video forensic settings.

1.2 Problem Statement

This project aims to develop and evaluate methodologies for open set image to video entity matching, where given a reference image, the system must retrieve and timestamp the frames within a video that contain the same entity. The model should be able to perform on unseen videos and references (open set recognition) as well as being able to identify small objects within the video's frames.

1.3 Aims and Objectives

Through the use of image and video entity matching techniques this project aims to tackle the limitations that many contemporary object detection models face. Specifically this project aims to:

- Use image based querying for locating specific entities instead of a text based approach allowing for flexibility beyond fixed class labels.
- Develop methods that can match object instances, rather than broad object categories.
- Extend recognition to entities that typically fall outside of the scope of standard object detection classes by using open set and open vocabulary approaches.
- Propose and implement novel methodologies for dealing with open set image video entity matching (FPN Siamese and proposal-verify pipeline).
- Construct a dataset of videos and reference images that span a wide range of categories and object sizes including non-standard entities (e.g. badges) representative of the real world image video entity matching application.
- Detect entities within a video in less time than the video's total length to ensure practical use.

1.3.1 Constraints

Due to Collaboraite's security requirements, any solutions must not rely on any public internet access.

1.4 Research Questions

In order to achieve the aims and objectives of this project the following research questions must be addressed.

1. How well do classical keypoint methods, modern object detectors, and global Siamese embeddings perform for image to video matching on an open set dataset?
2. Does a Siamese metric-learning approach generalize across videos and references?
3. Does incorporating a Feature Pyramid Network (FPN) with multi scale features into a Siamese network improve performance, and more specifically on small/zoomed objects when compared to global embeddings?
4. Can an open-vocabulary proposal–verification pipeline improve retrieval performance in open set video entity matching?
5. What are the computational efficiency–accuracy trade-offs for the evaluated methods and how do they effect practical deployment for real world usage?
6. How well do the proposed methods generalise across object sizes and unseen videos compared to the baseline approaches?

1.5 Contributions

This project makes several contributions.

A systematic comparison of baseline methods adapted to tackle the open set image video entity matching problem. These methods include SIFT, AKAZE, ORB, Faster R-CNN, YOLO, DETR, and a global Siamese Network.

A custom bespoke dataset of 50 videos and 160 unique reference images with frame-level ground truths.

Two suggested novel approaches to the open set image video entity matching problem. A secure end-to-end pipeline for image to video entity matching utilising CLIP and GroundingDINO to generate proposals and ensure suitability in an open set domain and a Siamese network verification head to match proposals to reference objects. To our knowledge, no prior work has combined CLIP and GroundingDINO with Siamese verification for open set video retrieval in a forensic context. The second approach is a Siamese Feature Pyramid Network model which improves performance over the global Siamese model especially on entities that take up a small fraction of a video frame. The FPN Siamese extends prior global embedding approaches by incorporating multi-scale features to handle small objects.

Chapter 2

Background and Related Work

This chapter reviews the background and related work that underpin the image-to-video entity matching problem.

We first introduce classical approaches based on local key-point features, which historically formed the foundation of visual matching. We then discuss the evolution of modern object detection architectures, along with the shift from two-stage to one-stage detectors and the recent adoption of transformer-based methods. Additionally, we examine research on visual similarity and metric learning, with a particular emphasis on Siamese networks and their ability to learn discriminative embeddings. We also consider multi-scale representation learning strategies such as Feature Pyramid Networks (FPN). These methods are especially useful at detecting small objects. Open-vocabulary detection and vision-language models are also explored as they offer proposal generation for open-class data.

This review will position our approach within the context of the broader literature and clarify the technical foundations that we add to throughout this paper.

2.1 The Company and the Project

This project is in collaboration with the software development company Collaboraite whose mission statement is to develop bespoke data-related products and applications for their secure partners [2]. Collaboraite proposed a project which aims to establish a methodology and develop a working prototype to enable a user to rapidly identify a specific object within a defined set of images or videos. The user will provide a reference image containing an entity and a search video. The user expects to receive a frame from the search video where the reference entity is visible, as well as the timestamp of the given frame. Additionally, Collaboraite wants an open-set model that can be applied to any general image or video this presents a large challenge as typical object detection methods are often limited to their set of training classes.

2.2 Classical Local Features

A classic method of matching two entities is through the use of key-point detection and descriptors. These features are image regions that are distinctive and repeatable under changes in viewpoint, illumination and scale. These invariances allow entities under different conditions to be matched to one another based on their local features. SIFT [3], ORB [4], and AKAZE [5] are common classical feature descriptors that are often applied to entity matching problems.

2.2.1 SIFT

Scale Invariant Feature Transform (SIFT) is often used as a gold standard in earlier image retrieval work. It is robust to scale and rotation. SIFT detects local extrema in a scale-space pyramid by applying Difference-of-Gaussians (DoG) and each keypoint is described using a 128-D histogram of gradient orientations.

Difference-of-Gaussians is described as:

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma), \tag{2.1}$$

where

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{2.2}$$

is the Gaussian-blurred image at scale σ, G denotes the Gaussian kernel, and $I(x, y)$ is the input image. This causes SIFT to be computationally expensive which limits scalability for long video streams. SIFT may not be appropriate for video frame data as it struggles on low-texture or blurred regions [6].

2.2.2 ORB

Oriented FAST and Rotated BRIEF was originally designed as a fast, lightweight alternative to SIFT. ORB combines FAST corner detector and BRIEF binary descriptor. A fast corner detector is used to pick high contrast corner like pixels while computing orientation via an intensity centroid which enforces rotation invariance with:

$$\theta = \arctan\left(\frac{m_{01}}{m_{10}}\right), \quad m_{pq} = \sum_x \sum_y x^p y^q I(x, y), \tag{2.3}$$

where m_{pq} are spatial image moments. The descriptor of local neighbourhoods is then formed as a binary string from pairwise intensity comparison. This allows fast matching comparisons between binary descriptors through the use of Hamming distance [7].

2.2.3 AKAZE

Accelerate-KAZE improves upon KAZE [8] by using fast explicit diffusion to approximate non-linear diffusion more efficiently. This allows AKAZE to build features in a non-linear diffusion scale space instead of the previously used Gaussian scale space. Non-linear diffusion allows for the preservation of strong edges while smoothing out low-contrast regions. Practically this means AKAZE can be used in real-time applications where KAZE was too heavy. AKAZE is slower than ORB but offers a compromise between speed and distinctiveness (less collisions between different regions).

2.3 Modern Object Detectors

Classical key point detectors can be unreliable in areas of low-texture or motion blur and only provide sparse matches as opposed to a complete understanding of an image. This motivated a shift towards deep convolutional detectors that produce dense bounding boxes rather than sparse point matches allowing for object level reasoning. Object detection receives an image or video frame as input and attempts to identify key class labels for all of the different objects within the image. This differs from classification in that images for classification typically only contain one object to be classified/identified. Localisation is an extended case of object detection in which the localiser predicts bounding boxes around each object. Object detection predictions are typically made based on a fixed vocabulary often restricted to the classes of which ever dataset the model is trained on.

Several methods have been developed for object detection, the most relevant of which for this paper are: Faster R-CNN (a two-stage detector), YOLO (one-stage detector), and DETR (which utilises transformers).

2.3.1 Two Stage Detection

an example of a two stage detector is Faster R-CNN. Faster R-CNN is an evolution of R-CNN[9] and Fast R-CNN[10] by replacing the slow, hand-crafted region proposal step with a learnable Region Proposal Network (RPN), improving detection speed and accuracy. The RPN is a fully convolutional network that slides a small set of convolutional filters over the backbone feature map, generating candidate Regions of Interest (ROIs) with object scores and bounding boxes associated to each region. Each ROI is then cropped and a classifier head is used to predict object class, with a regressor being used to refine the bounding box coordinates.

Faster R-CNN optimises a multi-task loss:

$$L = L_{\text{cls}}(p, p^*) + \lambda L_{\text{reg}}(t, t^*), \tag{2.4}$$

where L_{cls} is the cross-entropy classification loss between predicted class p and ground truth p^*. L_{reg} is a smooth L_1 loss between predicted bounding box coordinates t and ground truth t^*. Faster R-CNN although powerful can be slow due to this two stage approach.

2.3.2 One Stage Detection

One stage detectors such as YOLO (You Only Look Once) [11], reduces the number of steps from two to one by treating detection as a single step regression problem. This means that YOLO predicts bounding boxes and class labels directly from the image in a single pass. It does this by dividing the image into a grid where each cell in the grid is responsible for detecting and localising any objects that fall into it. Several overlapping object boxes are predicted and non-maximum suppression (NMS) [12] is used to retain only the highest confidence boxes by discarding boxes with high intersection of union (IoU) overlap. Which is calculated using

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

Area of Overlap: pixels covered by both boxes. Area of Union: total pixels covered by either box. YOLO's coarse grid division can cause the model to often struggle when detecting small objects as found in a 2025 technical review [13]. This may mean that YOLO may fail in the context of our open-set image and video entity matching problem.

2.3.3 Transformer Based Detectors

Transformer based detectors, such as DETR [14], use self attention mechanisms in transformers to output box predictions based on an image's encoded features. This allows models to capture global context via self attention which can improve their performance on cluttered scenes with many objects [15]. Additionally, DETR implements bipartite matching loss with the Hungarian algorithm in order to avoid the use of NMS.

Object detectors require class labels for training and detection. This is problematic for the open set setting that this project takes place in.

2.4 Object Verification and Metric Learning

Object verification is the process of confirming that two entities are the same typically by comparing two images to one another. This differs from object detection in that object detection only labels an object as being part of a given class and does not have the ability to verify a specific instance of a class. A common method for accomplishing this is through the use of Siamese models and contrastive loss [16] or triplet loss [17] learning. Siamese networks [18] utilise two identical twin networks and map inputs to an embedding space using distance metrics to determine similarity.

2.4.1 Contrastive Loss

Given a pair of samples (x_i, x_j) with label $y \in \{0, 1\}$, where $y = 1$ denotes a positive pair (same entity) and $y = 0$ a negative pair (different entities), the contrastive loss is defined as:

$$L_{\text{contrastive}} = y \cdot d^2 + (1 - y) \cdot \max\big(0,\ m - d\big)^2,$$

where $d = \|f(x_i) - f(x_j)\|_2$ is the Euclidean distance between embeddings produced by the network $f(\cdot)$, and $m > 0$ is a margin hyperparameter. The loss encourages positive pairs to be close in embedding space (d small) and negatives to be separated by at least a margin of m.

2.4.2 Triplet Loss

Given an *anchor* a, a positive sample p, and a negative sample n, the triplet loss is:

$$L_{\text{triplet}} = \max\big(0,\ d(a, p) - d(a, n) + m\big),$$

where $d(u, v) = \|f(u) - f(v)\|_2$ is the embedding distance and m is the margin. Triplet loss enforces that the distance between the anchor and positive is smaller than the distance between the anchor and negative by at least a margin of m. This directly models relative similarity due to the anchor being closer to the positive (matching entity) than the negative (different entity).

2.4.3 Feature Pyramid Networks

Course high level features like those produced by CNN backbones can cause smaller objects within the image to disappear or lose prominence. Feature Pyramid Networks (FPNs) [19] incorporate connections from later layers in the CNN backbone to earlier layers. This allows FPNs to produce embeddings that combine high level semantic features with low level fine details improving the detection of objects at varying scales. FPNs have been shown to significantly boost the detection of small and medium objects. This quality is extremely beneficial for an open-set problem where scale varies greatly from entity to entity such as the image-video entity matching problem that this paper tackles.

2.5 Image-video Entity Matching and Retrieval

This project's main focus is image-video entity matching and retrieval. Unlike traditional object detection, which predicts objects from a fixed vocabulary, this problem is an open-set retrieval setting. The model must be able to identify frames from the search video that contain the reference image entity without relying on predefined classes.

Image and video matching and retrieval incorporates aspects of both object detection -detecting all instances of pre-defined classes- and verification -matching two entities. However, differs from both tasks in that it aims to identify and return the best possible matching frames and timestamps from a video using open-set image queries.

2.5.1 Challenges

There are several challenges specific to Image-video entity matching and retrieval.

Many consecutive frames are near-duplicates and when an entity appears within a video there will be a group of correct frames preceded and followed by groups of incorrect frames. This temporal redundancy can be overcome with efficient retrieval techniques such as setting a step value to only take 1 every k frames. However, this could miss very brief appearances. Another method of dealing with this challenge is by detecting when significant changes occur within the video and sampling a frame for each significant scene change.

Appearance variation is one of the biggest challenges when dealing with image and video entity matching. This occurs when the reference image appears with different properties in the video such as scale, occlusion, orientation, blur, etc. Similar issues are faced in person re-identification [20].

This project focuses on an open-set general domain, this faces the issue of domain shift which is where the training and testing set differ too greatly from one another. A robust training set must be used in order to mitigate this problem as much as possible. Unfortunately, training cannot cover all possible entity categories in open-set retrieval.

2.6 Open-vocabulary & Vision-language Models

Open-vocabulary models counter the close-vocabulary problem that many traditional models face. This problem is due to traditional object detectors being trained on fixed class sets and thus not being able to recognise objects outside of the predetermined class set. The closed-vocabulary problem, and by extension many traditional object detection models, is too restrictive for an open-world entity matching that aims to be applied to a plethora of real world general scenarios.

2.6.1 CLIP

Contrastive Language Image Pre-training (CLIP) [21] is a foundational open-vocab recognition model. It learns to associate images with corresponding textual descriptions through being trained on image-text pairs. At test time, this allows zero-shot recognition meaning the model can handle classes that it was never explicitly trained on as it learnt the associations between pairs and was never presented explicit class labels. CLIP gives one embedding for a whole image, this means that if CLIP is to be used to detect objects within an image CLIP needs to be used in tandem with other models/frameworks.

2.6.2 GroundingDINO

GroundingDINO [22] builds upon the family of transformer based object detectors, like DINO [23] and DETR, by guiding detection queries with free form language prompts. Image and text encoders are used to extract tokens and embeddings from the queries. Cross-attention layers in the transformer decoder then allow object queries to utilise together both visual and textual tokens. GroundingDINO then produces a set of bounding boxes with confidence scores with each box being explicitly linked to the initial text prompt.

GroundingDINO has many desirable characteristics in the context of open-set image and video entity matching such as open-vocabulary detection and zero shot generalisation. Furthermore GroundingDINO is better suited to text-to-box proposals from direct prompts than other state of the alternatives such as OWL-ViT as it has higher recall better prompt alignment particularly for small/multi object scenes [24].

2.7 Evaluation Metrics

Evaluation metrics are needed to help establish how a performance is measured. Furthermore, multiple complementary metrics can paint a more detailed picture as to where a model excels and in what areas it needs improvements.

2.7.1 Classification Metrics

Accuracy represents the proportion of correctly classified frames from a video. This can be misleading under a class imbalance or when the model only classifies and returns the top k-frames from a video.

Precision is the proportion of predicted positives that are correct, while recall is the proportion of true positives that are retrieved.

F-1 score is an average of precision and recall that greatly penalises if either score is very low.

2.7.2 Ranking and Retrieval Metrics

Mean Average Precision (mAP) summarizes ranking quality by averaging precision across queries.

Recall@K is the proportion of queries where a correct frame appears within the top-K retrieved results. This means that as K increases Recall@K increases.

Mean Reciprocal Rank (MRR) evaluates how early the first correct frame is retrieved in the ranking. This is useful in the intended use of using a model to search a video for an entity in a given image.

2.7.3 Evaluation Curves

ROC-AUC (Receiver Operating Characteristic - Area Under the Curve) plots the true positive rate against the false positive rate across different threshold values. The threshold value is the value at which a frame is deemed to contain the reference entity (e.g. score ¿= T, predict entity present).

PR-AUC (Precision Recall - AUC) is an alternative to ROC-AUC specifically used when a strong class imbalance is present as it reflects the trade-off between precision and recall.

2.7.4 Efficiency Metrics

Runtime, memory usage, and CPU/GPU utilisation can be used to measure various aspects of a model's computational performance.

Chapter 3

Methodology

3.1 Problem Formalisation

Given a reference image I_r containing a single entity and a video of frames $\{F_t\}_{t=1}^{T}$, the goal is to identify the frame which best displays the reference entity from the subset of frames containing the entity. The 'best' frame (most likely match) and timestamp of said frame are to be output to the user. The problem is open set meaning that the reference objects that can be used are not limited to any specific classes. This can include arbitrary entities such as accessories, clothing, or logos. Because of this, the problem is more challenging than conventional object-detection as models must be able to generalise to unseen novel classes. For this a custom dataset was created and two main approaches were primarily explored. These approaches are:

- **Feature Pyramid Network (FPN) Siamese Matching**, where a deep network embeds both I_r and F_t into multi-scale feature embeddings using a FPN and computes a similarity score $s_t = \mathrm{sim}(f(F_t), f(I_r))$. A match is predicted if $s_t \geq \tau$. This approach is an extension of a standard global Siamese network. The FPN extension was added as global embeddings may fail when the reference object occupies only a small portion of the frame.

- **Proposal-Verify Pipeline**, where an open-vocabulary detector generates candidate crops $P_t = \{b_{t,k}\}$ from F_t. Each crop is compared against the reference embedding and the crops whose similarity pass the threshold are taken: $\mathrm{sim}(f(\mathrm{crop}(F_t, b_{t,k})), f(I_r)) \geq \tau$. This method is more robust to scale and occlusion but depends heavily on the quality of localisation and prompts.

By comparing these two paradigms, the study evaluates the trade-off between efficiency and robustness in whole-frame generalisation vs localised proposals for open set instance-level video matching.

3.2 The Dataset

A custom dataset was designed and created to capture the challenges of instance-level entity matching within temporally continuous videos.

3.2.1 Pre-made Dataset Review

Before creating a bespoke dataset, prominent image and video datasets such as COCO, LVIS, and Visual Genome, were reviewed as they provide large collections of annotated objects. However, they presented several constraints that prevented them from being directly suitable for the task of image to video entity matching. Firstly, many traditional datasets contain independent still images, not video data. This excludes the comparison and ranking context (locating the reference object across a set of frames and identifying the most suitable frames) that the image and video entity matching problem requires. Furthermore, annotations in these datasets focus on object categories, not instance-level matching/identification. For example, COCO indicates if a "dog" is present, but would not be able to answer if the dog was the particular reference dog or another dog as it does not have precise enough instance-level labels. Finally, even video datasets (e.g. YouTube-VIS) label object categories over time but still do not provide paired reference images with explicit frame-level correspondence annotations.

3.2.1.1 Partial Reuse

Although these dataset were not suitable for direct evaluation or training, they were still partially applied specifically for the open-vocabulary pipeline. The COCO and LVIS category lists served as the foundation for constructing the prompt vocabulary. Visual Genome was employed to extend the vocabulary with additional object synonyms and variations.

3.2.2 Dataset Creation

Due to the previously stated incompatibilities that image and video entity matching faced with many common computer vision datasets, the decision to make a custom dataset was made. The dataset consists of 50 diverse videos with 1-5 unique references for each video (totalling 160 unique reference images). The videos were sourced using YouTube's Creative Commons search filter. The references images contain objects that have been cropped from the video. The reference images have a probability of a random transformation applied to them. These transformations and the probability of occurrence are: changes in horizontal orientation (50%), brightness (30%), contrast(30%), saturation(30%), and rotation of up to $\pm 15°$ (10%). The purpose of applying transformations to the reference image is to simulate realistic variations in user supplied images. For this reason transformations are not applied in the same way to the frames of the video.

The object classes within the video vary significantly but maintain a focus on badges, logos, and personal items like clothing, bags, and bikes.

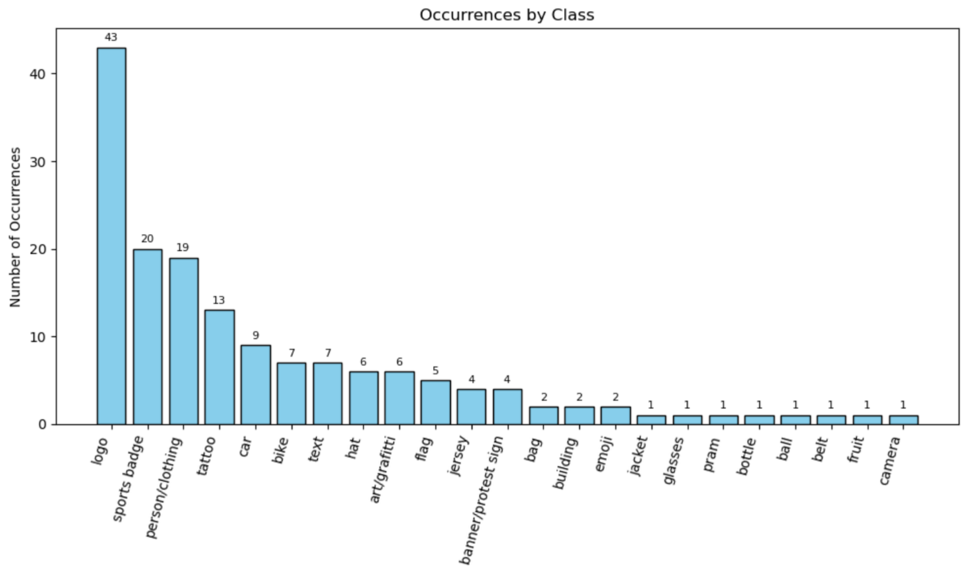

Figure 3.1: Class counts of the dataset.

This focus was in order to align with Collaboraite's eventual use case for the project. Additionally, by including non standard classes like logos and badges, we can better ascertain the model's true ability to match entities regardless of their shape or class.

After the videos and references were collected, timestamps were manually gathered pertaining to when each reference entity occurred in the video. These timestamps were then used to create binary frame-level ground truth labels for each reference. These ground truths were stored as JSON files unique to each reference and listed which frames visually contained the reference and which frames did not.

3.2.3 Dataset Statistics

The mean duration of the videos in the dataset is 3 minutes and 8 seconds with the longest video being 14 minutes and 46 seconds (long-tail). The majority of videos being short in length allows for better training and testing speed. Having only the occasional video being longer in length allows the model to still be challenged to work with higher frame counts without drastically increasing the overall runtime for the dataset.

The dataset contains videos both in 1280x720 (landscape) resolution and 360x640 (portrait). Varying resolutions forces the model to recognise objects in varying levels of detail thus making it more robust to scale and quality. Additionally, multiple resolution types is more reflective of a real world setting.

Reference object size varies throughout the dataset with a roughly even distribution of 'small' (taking up 10% or less space of the frames that appears in) and 'large' (¿10% of the frame) objects. Many reference objects appear with varying sizes in the video, such as when a bike approaches the camera). Seeing the object in varying scales teaches the model that an entity's identity does not depend on absolute size. This encourages the model to develop scale invariance.

3.2.3.1 Data Split

The dataset was split by video with 80% of the videos used for training and 20% for testing. A fixed random seed was applied to ensure reproducibility. Video-level splitting is essential in order to prevent data leakage by ensuring that references and frames from the same video never appeared in both the train and test splits. In a closed-domain setting, where the videos share very similar environments (e.g. CCTV), data leakage would cause less issues. However, when addressing open set matching, evaluation metrics must reflect the model's true generalisation ability for completely unseen videos.

3.3 Preprocessing & Data Handling

3.3.1 Frame Extraction

Frame extraction for both the training and testing worked via extracting every N frames. For example if N=1 then every frame from a video will be extracted. If N=2, 1 frame every 2 frames will be extracted. This would reduce the computational load by half. As N increases the computational load also decreases significantly each step. However, the chance of skipping a strong matching frame also increases. Due to computational constraints a value of N=10 was typically used to extract frames for training. And a value of N=30 was often used for testing, this meant that roughly 1 frame per second was taken from a video, drastically reducing the runtime of testing.

3.3.2 Data Normalisation

Extracted video frames and references were resized to 224x224 as the CNN backbones used for the image matching models were pre-trained on ImageNet which uses standardised input images of the same dimensions. Resizing ensures compatibility with the network architecture as well as reducing computational cost for larger frames. Another benefit of resizing the inputs to a standard resolution is that it provides consistency across all videos and reference images, ensuring that object scale variation is relative to the same frame size.

The input images were normalised to mean=[0.485, 0.456, 0.406] and standard deviation=[0.229, 0.224, 0.225]. ImageNet assumes pixel values are normalised to the dataset's channel mean and standard deviation, thus by normalising to these values the input distribution matches the pre-trained backbones expectations. This acts to stabilise training and aid in convergence by guaranteeing that the pre-trained weights are suitable for the dataset.

3.3.3 Data Augmentation

As previously mentioned, data augmentation was applied to the reference images in order to improve generalisation and account for real-world discrepancies in input images. These augmentations included horizontal flips, colour jitter, and rotations. The augmentations were each applied based on set probabilities in order to have varied distribution of augmented data and avoid the model overtraining on specific augmentations.

3.4 Baseline Methods

Before implementing a full model, several potential baseline methods were prototyped and explored in small scale tests to understand their suitability for the entity matching problem. These baseline models were based on: SIFT, ORB, AKAZE, Faster R-CNN, YOLO, DETR, and Siamese networks. The purpose of testing multiple baseline models is to ascertain which common models in existing literature would be suitable to focus on for the image and video entity matching problem. The methods were assessed on a small, simple test set without prior training in order to judge their potential for the project.

3.4.1 Classical Keypoint Matching

Classical feature matching has a long history in object detection and tracking and was a foundational approach to object matching prior to deep learning. In this project SIFT, ORB, and AKAZE were implemented as baselines using the OpenCV library, which provides optimized reference implementations of these algorithms [25]. By including these methods we provide a classical baseline to benchmark the performance of more modern embedding-based methods. The implementation of these three different feature matching methods covers accuracy and speed trade-offs across the most common types of handcrafted descriptors.

For each method, local descriptors were extracted from the reference image and video frames, then matched using a BruteForce matcher. Matches were filtered by distance, discarding weak matches with distances more than $0.7\times$ the worst (highest distance) match in the sorted set. This discards high-distance and weak matches, and reduced the overall necessary computation whilst maintaining the most distinctive matches. The video frames were then ranked by the total number of 'strong' matches. For the top frame candidates, RANSAC (Random Sample Consensus) homography was estimated in order to verify geometric consistency. Counting matches in this way is much more reliable than counting raw matches as it disregards random and inconsistent matches as well as confirming that matches follow a valid spatial mapping.

3.4.1.1 Scale-Invariant Feature Transform

Scale-Invariant Feature Transform (SIFT) creates accurate and robust descriptors (invariant to scale and rotation) but it is computationally expensive. This makes it less suitable for large-scale video processing especially in regions where descriptors are sparse such as areas with blur or low-texture.

3.4.1.2 Oriented FAST and Rotated BRIEF

ORB is significantly more efficient than SIFT and can be used for real-time matching. This property makes it more suitable for working with video data. However, in comparison to SIFT, ORB produces less distinct descriptors due to binary descriptors being more compact and less discriminative than higher-dimensional continuous descriptors. This leads to weaker performance on challenging entities.

3.4.1.3 Accelerated-KAZE

AKAZE offers better edge localisation than ORB due to the non-linear diffusion space causing keypoints to often fall on edges and texture boundaries. However, AKAZE is slower than both SIFT and ORB due to the computationally expensive non-linear diffusion filtering, thus making it less practical for video processing.

3.4.2 Faster R-CNN

Faster R-CNN is a two-stage detector that combines a Region Proposal Network (RPN) with a classifier and regressor head. Faster R-CNN was modified and used as a proposal generator in order to fit the image and video entity matching problem. For each frame F_t, Faster R-CNN produced a set of candidate bounding boxes $\{b_{t,k}\}$ with associated category scores. In this project, these detections were not used directly for classification but instead as candidate crops for entity instance verification. Each candidate crop was resized to 224×224 and passed through a pretrained ResNet-50 to obtain an embedding $f(crop_{t,k})$. A reference embedding $f(I_r)$ was computed once for the input reference image. Cosine similarity was then used to score each candidate against the reference image:

$$s_{t,k} = \frac{f(I_r) \cdot f(crop_{t,k})}{\|f(I_r)\| \|f(crop_{t,k})\|}. \tag{3.1}$$

A frame was marked as a match if the cosine similarity exceeded a set value of $\tau = 0.7$. This value was adjusted and settled on in order to trade off between the number of predictions and the quality of matches. Reducing τ will lead to more matches being displayed but the quality of matches reducing and vice versa.

3.4.3 You Only Look Once

You Only Look Once (YOLO), attempts to massively reduce the runtime of approaches like Faster R-CNN by reducing the approach from two stages to one. This is accomplished partially by reformulating the detection into a single regression problem. Similar to Faster R-CNN, YOLO was used solely to propose bounding boxes per frame. These cropped boxes were then verified via the same ResNet-50 embedding and cosine similarity check against the reference embedding. This adjusted pipeline again enabled YOLO to serve as a candidate generator in an open set, instance-level context rather than as a closed-set category detector as it is typically used.

3.4.4 DEtection TRansformer

DEtection TRansformer (DETR) uses a transformer encoder–decoder architecture. For each frame DETR gives a set of predictions $\{(b_i, c_i)\}_{i=1}^{N}$, where b_i is a bounding box and c_i is a probability vector over all of DETR's classes including a special 'no-object' class. For the purpose of this project, the box outputs were extracted as candidate crops and subsequently compared to the reference embedding using cosine similarity (as shown in equation 3.1). By adjusting DETR in this way it works within an entity matching setting without being limited to the original category labels that DETR was trained on.

3.4.5 Siamese Networks

The final baseline method that was implemented was a Siamese model. Siamese networks consist of two identical neural networks, these networks are each fed a different image and compute an embedding for the image. Then the embeddings of each image are compared in order to ascertain the similarity between the images. Normally, Siamese networks are trained with triplet loss or contrastive loss, however, a pre-trained ResNet backbone was sufficient to provide as a baseline. A ResNet-50 backbone, pre-trained on ImageNet, was used with the final classification layer being removed. The penultimate feature map was then passed through a fully connected head that reduced the dimensionality in order to produce more compact embeddings that are cheaper to store and compare. This results in a 256-D embedding vector for each input image. Both the reference and video frames were resized and normalised to fit ImageNet's prerequisites. Cosine similarity was then used on the normalised embeddings, allowing the similarity to be based on orientation in the feature space.

A frame F_t is classified as a match with reference image I_r if the cosine similarity between their embeddings exceeds a fixed threshold τ:

$$\hat{y}_t = \begin{cases} 1 & \text{if } s(f(I_r), f(F_t)) \geq \tau, \\ 0 & \text{otherwise,} \end{cases} \tag{3.2}$$

where $f(\cdot)$ is the embedding function (ResNet-50 + projection head), $s(\cdot, \cdot)$ is cosine similarity, and τ is the similarity threshold.

3.4.6 Baseline Limitations

Although the baseline methods are good starting points for addressing the problem of open set image to video entity matching, each baseline poses its own pitfalls and limitations.

Classical feature matching methods typically perform poorly on small or blurred objects and fail to generalise to unconstrained video conditions. This does not align with the open set goal of the project where we wish to develop a model that works well on all types of videos and references.

The adjusted object detection methods (R-CNN, YOLO, DETR) are unfortunately limited to only the objects they have been trained on. This would mean that they likely struggle to propose confident regions for atypical entities such as logos or signs.

The Siamese baseline is able to be used for an open set problem due to no limitations on training classes. However, currently the Siamese network is comparing each global embedding of the video frames to the reference image. This means that unless the reference entity takes up a large portion of the video frame it is unlikely to contribute much to the final embedding.

Due to these shortcomings two new methods were proposed to address the open set image-video entity matching problem.

3.5 Proposed Models

This section introduces the two main proposed approaches developed for the Image and Video entity matching task. Both approaches address the observed limitations of global frame-level matching and category based detection.

3.5.1 Feature Pyramid Network Siamese Model

The baseline implementation of ResNet Siamese networks matches at the frame-level at a single resolution. This causes reference objects that only take up a small portion of a frame to be overlooked. Feature Pyramid Networks (FPNs) provide a multi-scale representation, this is utilised along with the Siamese matching to improve robustness to scale variation and improve small object detection. This is especially important as the dataset includes logos, badges, and small accessories that often occupy ¡10% of a frame.

3.5.1.1 Backbone Design

The Siamese implementation for this method is based on ResNet-18 pretrained on ImageNet. Both a ResNet-18 and ResNet-50 model were initially trained and tested, revealing minimal differences in accuracy. Therefore, the ResNet backbone was downgraded in order to reduce computational cost and improve runtime by reducing the complexity of the model. Intermediate feature maps are extracted from four levels of the backbone (C2-C5). These levels correspond to output strides 4, 8, 16, and 32 respectively. With the lower levels maps e.g. C2 containing finer detail but less semantic meaning and the higher level maps containing more context but losing small object details. In order to combine the information across scales the feature maps are reduced to 256 channels by a 1x1 convolution and then added to the upsampled version of the next, coarser, level.

$$P_l = W_l * C_l + \text{Upsample}(P_{l+1}), \quad l \in \{2, 3, 4\}, \tag{3.3}$$

This results in several P-maps:

- P5 = C5 (deep semantics).
- P4 = C4 (mid-level detail) + upsampled P5 (semantics).
- P3 = C3 + upsampled P4.
- P2 = C2 + upsampled P3.

The P-maps are then passed through a 3x3 convolution to clean up any artifacts caused by the upsampling. Together these P-maps form a feature pyramid that combines features from all scales where the layers provide complementary information of both high-level context and fine grained details.

3.5.1.2 Multi-Scale Representation and Embedding

Each level of the pyramid specialises in different visual levels e.g. P_2 detects fine details at a high resolution and P_5 detects coarse resolution, semantic context. Global average pooling is applied to each pyramid level to collapse each into a single 256-D vector. This makes the layers of the pyramid directly comparable with one another. Each pooled vector represents the summary of the object's appearance at their specialised scales. The vectors are then concatenated to create a 1024-D vector that contains the combination of this information.

The concatenated features are then passed through fully connected layers resulting in a 128-D embedding that is small enough for efficient similarity scoring but still rich enough to encode the multi-scale information.

3.5.1.3 Normalisation

L2 normalisation is applied to the embeddings which causes cosine similarity and Euclidean distance to be directly related. This trick was used in earlier metric learning work [17]. Not only does this have the benefit of stabilising training, it also aligns with the cosine similarity evaluation used in the baseline methods.

3.5.2 Proposal and Verification Pipeline

A secondary solution to the struggles that global frame-level matching faces when dealing with small objects is proposed in the form of a pipeline that introduces an open-vocabulary localisation stage and verifies each candidate crop with Siamese matching. This ensures that the reference images are being compared to potential entity matches from small regions as opposed to entire frames. Isolating objects from the frame into proposals makes the system more robust to scale, occlusion and scene clutter, however, it comes at a greater computational cost.

3.5.2.1 Prompt Vocabulary & Selection

Initially, a safe vocabulary is constructed that combines object categories from COCO, LVIS, and Visual Genome into a single deduplicated vocabulary. Colour modifiers are added (e.g. ”blue coat”, ”yellow boots”) to capture additional visual attributes. The combination of categories from various datasets with added colour modifiers increases the flexibility of prompts beyond a closed-set setting.

OpenCLIP is used to encode an image embedding for each reference image I_r and encode all candidate prompts from the safe vocabulary. The text prompts are then scored by cosine similarity and ranked. Top k prompts are selected with a diversity penalty to avoid near-duplicate phrases. Taking multiple prompts accounts for any inaccuracies that CLIP might produce when embedding the reference image as well as accounting for the fact that the exact object may not exist within the safe vocabulary. However, as k number of prompts increases the computational requirements increases as the pipelines subsequent steps need to be carried out for each prompt.

3.5.2.2 Proposal Generation (GroundingDINO)

GroundingDINO is run once every x frames with the combined top-k prompts. Increasing x reduces the number of times GroundingDINO is run thus reducing the overall runtime, however it means that some frames that could potentially provide good matches are skipped over. Additionally, using a single combined caption of the top-k prompts reduces the number of times GroundingDINO has to be run, improving speed. GroundingDINO outputs bounding boxes along with confidence scores and Non-Maximum Suppression (NMS) is applied to remove duplicate or greatly overlapping proposals. This ensures diverse and high quality boxes are kept and redundant boxes are not verified. Padding is added around bounding boxes before cropping (4 pixels) to include some additional context around the object which can help with partial occlusion.

3.5.2.3 Siamese Matching Verification

A Siamese network is used as the final step to match the cropped proposal objects to the original reference image. Each crop is resized and embedded using the Siamese network. Cosine similarity is then computed between the crop embedding and the reference embedding as in previous methods that computed the frame and reference. Each frame is ranked by the maximum similarity scored by any one of its proposals. This means that even if multiple crops are proposed, only the most likely match fuels the decision instead of favouring frames that have many proposed crops.

3.6 Training

OpenCLIP and GroundingDINO were not trained within this project. Instead they were utilised as frozen pretrained components solely for the purpose of proposal generation and prompt ranking. Novel training was entirely in the custom verification stage utilising Siamese matching networks. Both a standard Siamese model and an FPN Siamese model (with ResNet-18 backbones) were trained using 80% of the videos from the custom dataset. The training was implemented using PyTorch with supporting libraries from Torchvision.

3.6.1 Triplet Loss Function

Siamese models are commonly trained with triplet loss or contrastive loss functions. For this project triplet loss was selected over contrastive loss. Triplet loss was implemented on L2 normalised embeddings, this means that Euclidean distance is equivalent to cosine similarity. This ensures consistency between training (learning with cosine-geometry via normalised Euclidean distance) and verification (deciding if a reference matches a frame using cosine similarity).

Cosine Similarity Triplet Loss:

$$L = \frac{1}{N}\sum_{i=1}^{N} \max\Big(0,\ \|\hat{f}(a_i) - \hat{f}(p_i)\|_2\ -\ \|\hat{f}(a_i) - \hat{f}(n_i)\|_2\ +\ m\Big) \tag{3.4}$$

where:

- $\hat{f}(x) = \frac{f(x)}{\|f(x)\|_2}$ is the L2-normalised embedding of image x,

- a_i, p_i, n_i are the anchor, positive, and negative images in the i-th triplet,
- m is the margin hyperparameter,
- N is the number of triplets in the batch.

Training on triplets lends itself to rank ordering where correct video frames are closer than incorrect frames. Additionally, triplet loss has better generalisation to unseen videos than contrastive loss as contrastive loss can over fit by collapsing embeddings too tightly around training positives, where triplet loss forces the model to learn discriminative boundaries that generalise better across new videos. Triplets were created using a reference image, positive frames, and negative frames identified via the datasets ground truth labelling.

Triplets were regenerated every 8 epochs. This was implemented as earlier triplet-based papers[17] found that regeneration consistently outperformed fixed triplets especially on small or moderately sized dataset.

3.6.2 Optimisations

Some operations were cast to Mixed Precision (FP16) where safe for the benefit that mixed precision halves the memory footprint on the GPU and accelerates training. This comes with a small risk of numerical instability in some operations, however if that is detected then the model will fall back to full precision (FP32).

AdamW ($LR = 1 \times 10^{-4}$) with weight decay was used as the optimizer as triplet loss often produces sparse gradient signals which Adam's adaptive updates can handle better than plain SGD. Optimizer resets were used in order to avoid unnecessary memory writes and to improve training times. This is done by setting the gradient tensor to None instead of filling it with zeros, which allows PyTorch to skip some gradient tracking work.

3.6.3 Checkpointing

Throughout training, the best models were saved when the model achieved a new lowest validation loss. Loss was used instead of accuracy as it reflects both the confidence and margin violations in the triplet loss. Metrics including training loss, validation loss, ROC-AUC, and mean average precision (mAP) were recorded at each epoch. Fixed random seeds and deterministic dataloaders were used to ensure that experiments could be replicated and results compared fairly across model variants.

3.6.4 Hyperparameters

Hyperparameter	RN-18 Siamese	FPN-Siamese
Backbone	ResNet-18	ResNet-18 + FPN
Embedding dimension	128	128
Batch size	32	32
Epochs	32	20
Learning rate	1×10^{-4}	1×10^{-4}
Optimizer	AdamW	AdamW
Weight decay	1×10^{-2} (default)	1×10^{-2} (default)
Margin (triplet loss)	1.0	1.0
Triplet regeneration	every 8 epochs	every 5 epochs
Max triplets / ref	30	30
Max triplets / video	300	300
Input size	224×224	224×224
Normalisation	ImageNet mean/std	ImageNet mean/std
Augmentations	Flip, color jitter, rotation	Flip, color jitter, rotation
Precision	Mixed precision (FP16 + GradScaler)	Mixed precision (FP16 + GradScaler)
Seed	42	42

Table 3.1: Training hyperparameters for the Siamese verification models.

The FPN model, potentially due to its multi-scale features, converged faster, making fewer epochs sufficient. Early stopping by monitoring validation loss showed that the FPN model began to overfit if pushed much beyond 20 epochs, whereas RN-18 remained stable up to 24.

3.7 Evaluation Protocol

The main goal for the model is to have the ability to find the best frame of a video given k-predictions to do so. The evaluation protocol is set in order to reflect that.

3.7.1 Setup

Each evaluation run is on the same 20% test split across all methods for comparability. This test split contains 10 videos, with 35 total reference images. The videos in the test set were hand selected from the main dataset in order to present a diverse range of search video and entity types. The ground truths for the reference images are matched across all frames of the test videos. This allows for scores to be computed at the frame level and aggregated across videos for an overall evaluation result.

Both the proposal generation stage and the Siamese verification stage produce real type scores. A threshold is applied to these scores in order to ascertain whether a frame or proposal is classified as containing the reference entity or not. For the trained triplet-based Siamese and FPN-Siamese models most evaluation was conducted using threshold-independent metrics (ROC-AUC and mAP), which assess the full distribution of similarities rather than a single cut-off. Additionally, a fixed threshold of $\tau = 0.75$ was used when necessary. This value was based on preliminary experiments on the baseline Siamese model that balanced false positives and false negatives whilst maintaining runtime efficiency.

For GroundingDINO box and text threshold were set to 0.25 with non-maximum suppression at IoU of 0.6. Furthermore, the Top-K proposals per frame were capped at 100. These choices were made as without filtering, the number of proposals per frame can be very large, which creates both computational inefficiency and an inflated false-positive risk.

3.7.2 Metrics

Model performance was assessed using a combination of different metrics. A multi-faceted approach ensures that models are not only accurate in discriminating positives from negatives, but also effective in retrieving the correct frame quickly and in a manner consistent with deployment requirements.

3.7.2.1 Frame Level Classification

For each reference-video pair frames are scored by their cosine similarity with the reference image. The threshold is then applied to get a binary result indicating if the object is predicted as present within the frame or not. These results are then used to calculate further metrics.

- **Receiver Operating Characteristic - Area Under Curve (ROC-AUC)** - measures the probability that a randomly chosen positive frame is assigned a higher similarity score than a randomly chosen negative frame.
- **Precision Recall - AUC (PR-AUC)** - an alternative to ROC-AUC that is preferred when positive frames are scarce.
- **Simple Accuracy** - measures the overall proportion of correctly classified frames (sim(anchor,pos) ¿ sim(anchor,neg)).
- **Size Stratified Analysis** - The ROC-AUC and PR-AUC average values calculated on positive frames grouped by size (small, medium, and large)

3.7.2.2 Retrieval/Ranking

Ranking based metrics were also computed in order to align with the ultimate task of presenting the best matching frame from a video.

- **Recall@k** - measures if any true positive appears in the top-K frame predictions.

3.7.2.3 Operational Metrics

- **Runtime** - total time taken to make predictions for all videos in the test set.

Chapter 4

Results

This chapter presents a comprehensive evaluation of the proposed image to video entity matching methodologies and a comparison to the range of baseline variants. We first assess classical feature matching methods, object detector–based approaches, and global Siamese embeddings to establish baseline performance. Next, we evaluate the proposed FPN Siamese network and the Proposal-verification pipeline. The results are reported using both quantitative metrics (ROC-AUC, PR-AUC, Recall@K, runtime) and qualitative analysis in order to form a robust evaluation of the method's strengths and weaknesses.

4.1 Experimental Setup

Before presenting the results we will establish the experimental setup in order to establish reproducibility and frame the context of the results.

4.1.1 Hardware and Software

Experiments were conducted on an NVIDIA GeForce RTX 3050 Laptop GPU with system CUDA 12.9 installed. This led to certain implementation decisions such as using mixed precision or evaluating every k-th video frame instead of every single frame.

Key software framework versions used were: PyTorch: 2.5.1+cu121, Torchvision: 0.20.1+cu121, OpenCV: 4.10.0, scikit-learn: 1.3.0, Ultralytics YOLOv8: 8.3.156.

4.1.2 Sampling and Testing Background

The test split contained 10 video's frames and references which were unseen during training to prevent data leakage. All videos were sampled at 30fps and frames were extracted at a stride of 30 for testing (so only 1 frame per second was evaluated). This high stride could potentially lead to missing strong matching frames, however, it was viewed as a necessity due to hardware and time limitations. Classical baselines (SIFT, ORB, AKAZE), detectors (YOLO, faster R-CNN, DETR), global Siamese networks (both trained and untrained on the dataset), and novel contributions (FPN-Siamese, Proposal-Verify pipeline) were all tested on the same video set for consistency.

4.1.3 Metrics Protocol

In order to properly assess a model's performance, the exact method of computing key metrics needs to be outlined. Video ground truths used to calculate metrics are binary indicating if a frame either does or does not contain the reference entity.

For each reference–video pair, methods produce a score per frame (or per proposal).

- **Best Accuracy** - the fraction of correctly classified frames at the decision threshold τ that yields the highest accuracy for the test set. This is optimistic and mainly used to illustrate the best case.
- **ROC-AUC** - the area under the Receiver Operating Characteristic curve is obtained by altering τ over all scores.
- **PR-AUC** - the area under the Precision–Recall curve is more informative than ROC_AUC when dealing with class imbalance.
- **Recall@K** - the proportion of reference-video pairs where at least one true positive frame is ranked within the top-K candidates. This is reported for K = 1,5,10,25,50.
- **Size-stratified analysis** - positive frames are grouped into *small/medium/large* based on the percentage of the total frame size that the search object takes up. For each bin PR-AUC and ROC-AUC are recomputed.
- **Runtime** - per frame runtimes are logged and summed in order to compute the total runtime.

4.2 Results of Methods on the Fixed Test Set

95% confidence intervals (CIs) were computed using nonparametric bootstrap resampling. The evaluation was repeated 1,000 times, each time drawing with replacement from the set of query, video pairs in the ground-truth annotations. The scores were then aggregated based on reference, video pairs in order to prevent imbalances from per-frame correlations as the videos have varying frame counts. Table 4.1 reports the point estimates of each evaluation metric on the full test set.

The accuracy metric represents the best accuracy found over a range of metrics. It is an optimistic metric meant to display the best outcome when the threshold is ideal. It is not meant to be used as a descriptor of the model's overall performance.

Method	**Accuracy**	**ROC-AUC**	**PR-AUC**	**R@1**	**R@5**	**R@10**	**R@25**	**R@50**
AKAZE	0.933	0.607	0.240	0.600	0.714	0.750	0.800	0.829
SIFT	0.945	0.657	0.343	0.714	0.821	0.829	0.829	0.846
ORB	0.933	0.435	0.068	0.400	0.486	0.629	0.657	0.714
Faster RCNN	0.935	0.537	0.144	0.600	0.714	0.800	0.857	0.857
YOLO	0.934	0.536	0.123	0.543	0.743	0.800	0.864	0.886
DETR	0.935	0.526	0.128	0.571	0.714	0.771	0.829	0.886
Base Siamese	0.933	0.558	0.087	0.229	0.343	0.486	0.571	0.629
Trained Siamese	0.933	0.602	0.129	0.257	0.429	0.571	0.714	0.743
FPN Siamese	0.933	0.633	0.132	0.371	0.629	0.657	0.714	0.800
Proposal-verify Pipeline	0.933	0.470	0.061	0.318	0.409	0.545	0.773	0.864

Table 4.1: Main comparison of methods on unseen videos.

4.3 Baseline Performance

In order to establish a reference point for subsequent experiments, a series of baseline methods were first evaluated. These methods covered classical keypoint matching approaches, adapted object detection models, and a global Siamese network. These methods give complementary perspectives with each differing significantly in their approach. Keypoints rely on local texture cues; object detectors utilise pretrained region proposal methods; and Siamese embeddings capture global similarity. Their performance on the unseen video test set is summarised in Table 4.1.

4.3.1 Classical Keypoint Matching

The classical computer vision methods implemented were SIFT, ORB, and AKAZE. These methods work by detecting and matching keypoints across images. ORB achieved moderate retrieval success (R@10 = 0.63), but its overall discriminative ability was low, based on poor ROC-AUC (0.435) and PR-AUC (0.068) values. These results indicate that although ORB can occasionally output correct frames, its confidence calibration and ranking quality remain limited.

4.3.2 Adapted Object Detectors

Object detection models (Faster R-CNN, YOLO, and DETR) were adapted to work as region proposal generators without dedicated verification. Faster R-CNN was slightly stronger than

YOLO and DETR but all three performed competitively overall. All three models achieved relatively high top-k recall, however produce poor PR-AUC values. This is potentially due to the models producing many bounding boxes per frame so that even if their precision is low (lots of false positives) the correct object is still often found. This suggests that while the pretrained detectors offer strong localisation, they likely require an additional verification step in order to reduce false positives and be able to reliably identify the reference image.

4.3.3 Global Siamese Networks

The baseline Siamese network (untrained network with ResNet-18 backbone) was likely the simplest approach in that it compared the global embedding of each frame to the embedding of the reference image. This approach (base Siamese) produced low ranking quality metrics (ROC-AUC = 0.558, PR-AUC = 0.087), with Recall@10 = 0.486, consistent with poor generalisation.

Additionally, a standard Siamese network was trained on the dataset to see if it could learn more descriptive embeddings over a variety of unique references despite being applied globally (without cropping). Fine tuning on the training data improved the ROC-AUC to 0.599, PR-AUC to 0.126, and Recall@10 to 0.571. These results highlight the potential of discriminative embeddings despite the limited dataset. However, performance still remained below detector-based baselines on unseen videos.

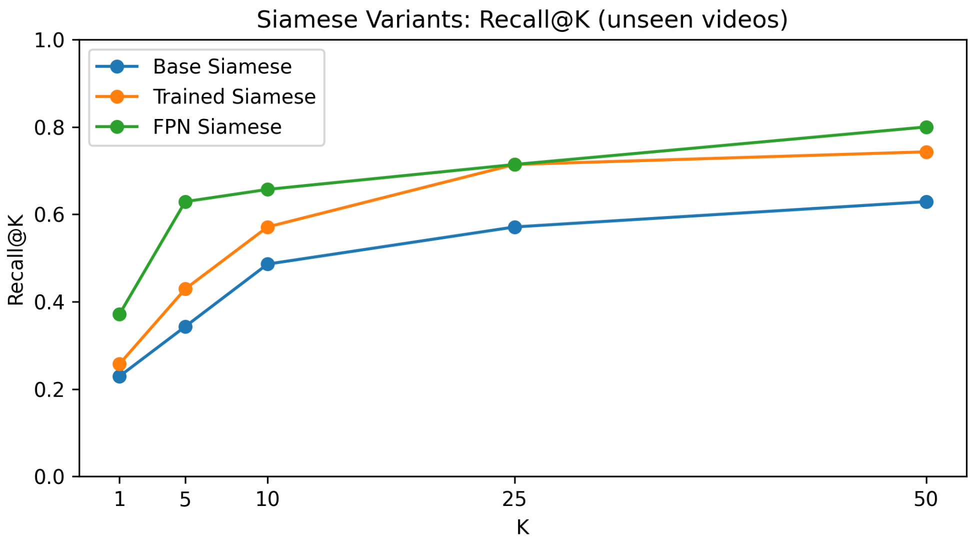

Figure 4.1: Recall@K performance of the Siamese models.

4.3.4 Summary of Baseline Results

Overall, many of the baselines demonstrated the ability to retrieve the correct frame within a given top-K predictions. The classical methods, however, struggled with robustness, while detector-based models offered better ranking ability. Global Siamese embeddings benefitted from training but still underperformed. This further motivates the need for architectures with explicit multi-scale reasoning and proposal-based verification.

4.4 Siamese Model Variant Comparison

In this section we compare the results of three iterative Siamese configurations. These networks are: an *untrained* ResNet-18 ("Base"), a *trained* ResNet-18 ("Trained"), and an *FPN* incorporated Resnet-18 Siamese.

Variant	**ROC-AUC**		**PR-AUC**		**R@10**		**Runtime**
Base Siamese	0.555		0.081		0.486		0h 9m 12s
→ Trained Siamese	0.599	(+0.044)	0.126	(+0.045)	0.571	(+0.085)	0h 9m 11s
→ FPN Siamese	**0.633**	(+0.034)	**0.128**	(+0.002)	**0.657**	(+0.086)	0h 12m 39s

Table 4.2: Performance of Siamese variants on unseen videos. Deltas indicate improvements on the row above.

Training the ResNet-18 Siamese yields a consistent gain over the untrained model with effectively the same runtime. Upgrading the backbone with a multi-scale FPN brings additional improvements in ROC-AUC, PR-AUC, and Recall@K with a minor additional runtime cost. PR-AUC only improves marginally with the FPN implementation, this suggests that the FPN model's main benefit is recall@K.

In summary of the incremental Siamese implementations, architectural changes that add *multi-scale features* (FPN) are more effective than simply training a global backbone when the target includes small instances. The cost of the FPN addition is an increase in compute time. For high-recall retrieval, FPN Siamese is a better verifier, while the trained ResNet-18 remains a strong speed–accuracy baseline.

4.5 Proposed Methods

We now evaluate the results of the proposed methods: Feature pyramid network Siamese model, and the proposal and verification pipeline. These two methods were designed in an attempt to address two key limitations: (1) the struggle of retrieving small or occluded objects, and (2) the poor generalisation when dealing with open-set data containing unseen references and videos.

4.5.1 Feature Pyramid Network Siamese Model

The Feature Pyramid Network (FPN) Siamese model implementation was motivated by the desire to better deal with small objects in a frame. As shown in Table 4.1, the FPN variant achieved the second highest ROC-AUC among all methods (0.633) and improved upon the Recall scores of the other Siamese architectures (both untrained and trained global models).

4.5.2 Proposal and Verification Pipeline

The second proposed method was combining an open vocabulary detector with a Siamese verification step. GroundingDINO was used to generate region proposals based on vocabulary prompts generated by CLIP. This design restricted the search space to realistic regions while allowing the embedding model to work as a fine-grained verifier. The number of proposals per frame was capped at 100 due to computational restrictions. Results show that the proposal-verify pipeline achieved the strongest retrieval at higher recall thresholds, reaching Recall@25 = 0.773 and Recall@50 = 0.864 (Table 4.1). This shows its ability to consistently return correct frames within the top ranks. On the other hand, the proposal-verification pipeline struggled at smaller K values for Recall@K. This indicates that the pipeline is prone to proposal noise, with false positives taking up the top ranks. As the earlier ranks are sensitive, a few confident false positives can replace the true entity despite it being detected.

4.5.3 Size Stratified Results

One of the key objectives with introducing the two proposed methods was to overcome difficulty when faced with small objects. In order to assess this, reference objects were grouped into three size bins (small/medium/large) based on their appearance in a frame. Small - the reference takes up less than 2.5% of the frame (this made up roughly 40% of references). Medium - the reference takes up 2.5-10% of the frame (roughly 45% of total references). Large - references that take up 10% of the frame.

Table 4.3 shows several interesting insights.

For small objects performance is generally weak across models. SIFT shows the highest ROC-AUC (0.708), followed by FPN Siamese (0.620). ORB displayed the worst ROC-AUC with 0.379. Most models score below 0.10 in terms of PR-AUC with FPN Siamese (0.114) providing the best relative performance.

Most methods performed better on medium sized objects. SIFT again achieves the strongest ROC-AUC (0.876), followed by AKAZE (0.845) and the base Siamese model (0.844). As for PR-AUC, DETR (0.216) and Faster R-CNN (0.209) perform the best. This suggests that detection-based methods benefit more from medium-scale targets.

For large objects classical feature matching methods again perform well with AKAZE achieving the highest ROC-AUC (0.949). SIFT achieves the best PR-AUC (0.258) again

Model	PR AUC			ROC AUC		
	Small	Medium	Large	Small	Medium	Large
AKAZE	0.0845	0.1539	0.2029	0.5798	0.8448	0.9490
SIFT	0.0384	0.1473	0.2580	0.7076	0.8597	0.8763
ORB	0.0360	0.0427	0.1008	0.3792	0.7919	0.8590
Faster R-CNN	0.0800	0.2091	0.0126	0.5164	0.7007	0.5323
YOLO	0.0694	0.1866	0.0141	0.5186	0.6700	0.5092
DETR	0.0681	0.2161	0.0105	0.5040	0.6846	0.4728
Proposal-verify	0.0586	0.0176	0.0113	0.4399	0.5223	0.3707
Base Siamese	0.0629	0.0109	0.1041	0.5628	0.8442	0.7142
Trained Siamese	0.0746	0.1272	0.0179	0.5328	0.7165	0.7699
FPN Siamese	0.1143	0.0188	0.0177	0.6399	0.6771	0.7803

Table 4.3: Size-stratified performance across models: PR AUC and ROC AUC (Small, Medium, Large object bins).

followed by AKAZE (0.203) and the base Siamese model (0.104). Detection based methods perform the worst on large objects.

In terms of the proposed methods, FPN Siamese performs well on small objects but is out performed by the base Siamese on large objects. The proposal-verify GroundingDINO pipeline, similarly to other object detectors, performed poorly on large objects. The pipeline produced its best scores on small objects, however, it produced below the average results across all categories.

4.5.4 Proposed Models Results Summary

In summary, FPN Siamese generally improves upon both the untrained and trained global Siamese networks while performing especially well on small objects.

The proposal-verify pipeline, had the highest eventual recall@50 indicating that it is the nest at eventually retrieving the object. However, the pipeline matched up poorly against many of the other methods in terms of ROC-AUC and PR-AUC, reflecting that it produces many false positives early in the ranking.

4.6 Efficiency & Resource Use

Table 4.4 displays the total runtime of each method on the 10 video test set (totalling approximately 40 minutes of footage). The evaluation of all models were executed overnight to guarantee exclusive access to computational resources. This helps to provide a fair and consistent runtime measurement across all methods.

Method	**Total Runtime**
Feature Matching (SIFT)	2h 45m 46s 888ms
Feature Matching (AKAZE)	1h 40m 14s 701ms
Feature Matching (ORB)	0h 16m 18s 969ms
Faster R-CNN	3h 26m 51s 638ms
YOLO	0h 52m 6s 132ms
DETR	4h 24m 54s 219ms
Proposal-Verify	1h 12m 2s 110ms
Siamese-Global	0h 9m 12s 688ms
Siamese-Trained	0h 9m 11s 130ms
Siamese-FPN	0h 12m 39s 314ms

Table 4.4: Runtime comparison across methods for processing the full test set.

4.6.1 DETR and Faster R-CNN, Causes of Slowness

DETR had by far the longest run time, this may be due to the fact that it uses a full transformer encoder–decoder for every frame. This means that each frame is tokenized into hundreds of patches and passed through the multi-head attention mechanism and eventually decoded into bounding boxes. This quadratic attention makes it computationally expensive especially on long videos.

Faster R-CNN runs a convolutional backbone, a region proposal network, and a classification head. Each frame generates many region proposals, each requiring feature extraction. This cascaded pipeline is much slower than single-pass embeddings. In order to improve the runtime of faster R-CNN when used in this context the number of region proposals should be limited.

Furthermore, unlike the Siamese implementations, the baseline methods did not implement batching and as such the GPU usage was suboptimal.

4.6.2 Siamese and YOLO efficiency

For Siamese models a single forward pass gives a compact embedding for each frame (ResNet18/50), making them extremely fast (minutes instead of hours). Furthermore, each of the Siamese implementations had batch processing built in in order to use the GPU more optimally.

YOLO is designed for real-time inference, so even with region crops, it was an order of magnitude faster than DETR/faster R-CNN.

4.6.3 Proposal Verify Pipeline

GroundingDINO is close in complexity to DETR, which is why this method is much slower than the raw Siamese. The proposal verify pipeline is still faster than DETR and faster R-CNN because of the limited proposals leading to less redundancy and unnecessary matching.

A break down analysis of the runtime for each step in the proposal-verify pipeline was performed. This can help identify any potential bottlenecks.

Step	**Runtime**	**Share**
Prompts	0h 7m 14s	10.05%
Load model	0h 0m 23s	0.54%
Prefetch (load images)	0h 2m 23s	3.30%
Proposals	0h 57m 44s	80.15%
Verification	0h 4m 18s	5.96%
Total	**1h 12m 2s**	**100%**

Table 4.5: Proposal-Verify Pipeline runtime by step (summed across all runs).

Table 4.5 shows that the proposals step of the pipeline (the GroundingDINO implementation) took by far the longest amount of time. However, when this method is compared to the runtime of the other object detector methods tested (YOLO, DETR, Faster R-CNN) its runtime is the shortest. This is likely due to the limit applied to the total number of possible proposals per frame (capped at 100).

4.7 Threats to Validity

The results calculated come with some nuance and threats to validity.

Temporal sampling: Frames were sampled at stride size of 30 meaning 1 frame was sampled every second. This was implemented to reduce total runtime but comes at the cost of missing frames containing the object that my be short lived or stronger than the frame selected at 1fps. This can cause an underestimate in Accuracy, Recall@K, ROC-AUC, and PR-AUC.

Open set problem: Due to the open-set nature of the image video entity matching problem and the limited test set, the metrics only reflect the model's ability on reference objects in the test set.

Hardware limitations: Several decisions were made to alleviate the computational load when testing and training the models. Mixed precision was used when possible which reduces the accuracy of certain calculations. The number of proposals for the GroundingDINO proposal-verify pipeline were limited to 100. This could prevent potential hits being found as they are not in the top 100 proposals.

Chapter 5

Discussion

This chapter further discusses and interprets the findings presented in Chapter 4 and relates them back to the broader literature of open set image to video entity matching.

5.1 Answering the Research Questions

This section revisits the original research questions (RQs) from the Introduction and interprets the qualitative observations and findings from the results chapter to answer them. The original research questions were:

RQ1: How well do classical keypoint methods, modern object detectors, and global Siamese embeddings perform for image to video matching on an open set dataset? Of the classical keypoint methods (SIFT, AKAZE and ORB) SIFT performed well across the detection metrics, scoring the highest of all methods in overall ROC-AUC. ORB on the other hand performed poorly in comparison with an especially low PR-AUC value. All three keypoint matching methods performed their best on large entities, this is likely due to larger objects having more detail and therefore more keypoints to match. These methods however, came at the cost of long runtimes with SIFT taking almost three hours to complete the 10 video test set.

The object detection methods (Faster R-CNN, YOLO, DETR) showed strong Recall@K but lower ROC-AUC and PR-AUC when compared to other methods. This indicates that although the models eventually find a positive frame they struggle to separate positives from negatives across thresholds (noisy rankings).

Global Siamese embeddings produce high ROC-AUC but low PR-AUC indicating that although the model can discriminate in a global sense (scoring positives higher than negatives). In practice the precision collapses due to the small number of true positives being overrun by a large number of false positives. This suggests that a global Siamese implementation may be appropriate for a setting where each frame is scored and then filtered using additional oversight but is less suited for automatic top-k retrieval as the first results are likely to be wrong, which is shown by poor recall@K scores especially when k is low.

RQ2: Does a Siamese metric-learning approach generalize across videos and references? The trained Siamese model improved upon the base global model (+0.045 PR-AUC, +0.085 Recall@10) indicating that metric learning adds discriminative power and the ability to generalise moderately well. A larger more comprehensive dataset may increase these improvements further. However, global embeddings still struggle on small entities likely due to the embedding being diluted by background context and not solely representing the desired object within a frame.

RQ3: Does incorporating a Feature Pyramid Network (FPN) with multi scale features into a Siamese network improve performance, and more specifically on small/zoomed objects when compared to global embeddings? Based on the findings in the previous chapter, incorporating a FPN into a Siamese network does improve performance (+0.171 R@10, +0.078 ROC-AUC, +0.047 PR-AUC when compared to the base Siamese). This performance is seen especially on small entities as shown in Table 4.3. These findings confirm our hypothesis that multi-scale embeddings are better for smaller entities.

RQ4: Can an open-vocabulary proposal–verification pipeline improve retrieval performance in open set video entity matching? The proposed proposal-verify pipeline that utilised CLIP, GroundingDINO and Siamese models produces the highest recall@50 showing that it has strong eventual retrieval. However, the poor PR-AUC and early k recall scores showed that the pipeline had weak early precision and struggled to distinguish positives cleanly from negatives. This pipeline may be suitable for exhaustive forensic search but is suboptimal for user-facing top-K retrieval.

RQ5: What are the computational efficiency–accuracy trade-offs for the evaluated methods and how do they effect practical deployment for real world usage? Based on the Table 4.4, Siamese models were the fastest followed by ORB, all other methods took a significant amount of time in comparison to the test set's total length (40 minutes). The object detector methods are likely bottlenecked by their proposal generation stage as shown in Table 4.5 of the proposal-verify pipeline's runtime per stage. The proposal-verify pipeline itself was faster than other object detector methods likely due to the number of proposals per frame being capped at 100. Therefore, selecting a model depends on whether real-time or exhaustive search is needed.

RQ6: How well do the proposed methods generalise across object sizes and unseen videos compared to the baseline approaches? The size stratified results Table 4.3 shows that the FPN Siamese was the best relative performer on small objects. Interestingly, the Base Siamese produces higher PR-AUC scores on large objects (0.104 vs

0.018) than the FPN Siamese while producing a worse ROC-AUC score (0.714 vs 0.780). This indicates that the FPN Siamese has greater discriminative power than the base Siamese, but its focus on small/medium objects means that large objects become less precise, the noisy false positive predictions cause a lower PR-AUC.

The proposal-verify pipeline proposed using CLIP, GroundingDINO and a Siamese network did not drastically improve performance in comparison to many of the baseline methods. However, its high recall@50 indicates that it may be a useful method to apply in forensic workflows where exhaustive coverage is prioritised over real-time and low k settings.

5.2 Why Methods Succeeded or Failed

5.2.1 Baseline methods

In order to present a contextual comparison we start by outlining why the classical keypoint detectors (SIFT, ORB, AKAZE) and the object detector based methods succeeded or failed.

5.2.1.1 Classical Keypoint Methods

The keypoint matching methods performed relatively well across the board with especially strong performances on large objects. SIFT had the highest large object PR-AUC (0.258) and AKAZE had the highest large object ROC-AUC (0.949 which means a 94.9% chance of a positive frame being given a higher similarity to the reference than a negative frame) indicating a very strong discriminative ability between positive and negative frames. This is likely due to large unobscured entities containing many descriptive keypoints for SIFT, AKAZE, and ORB to utilise and match. However, SIFT and AKAZE both came with a significant total runtime cost taking roughly 2 and 1 hours longer than the test sets total video runtime respectively. ORB on the other hand had a relatively short runtime (16 minutes). This is due to ORB using FAST (Features from Accelerated Segment Test) for keypoint detection and BRIEF (Binary Robust Independent Elementary Features) with an orientation step for its final descriptors. Therefore, ORB uses a very simple feature detector and its descriptors are binary strings which allow for fast matching using Hamming Distance. This drastically reduces ORBs runtime but also reduces its detail which explains why its performance in Table 4.1 was worse than SIFT and AKAZE.

These findings align with the literature as ORB was explicitly designed as a real-time alternative to SIFT/SURF.

5.2.2 Object Detector Methods

Object detector methods Faster R-CNN, YOLO, and DETR produce high top-k recall displaying their ability to retrieve a positive prediction within a given k chances. However, these methods often showed a poor PR-AUC score indicating that they also produce many false positives. This suggests that the baseline object detection methods are strong methods for proposal generation but require additional guidance to address the specific problem of open set matching. An attempt to address this was with the introduction of the proposal-verify pipeline which will be analysed later.

5.2.3 Global Embeddings

The base Siamese method as well as the trained Siamese network both calculated and compared global frame embeddings from the search video to the reference image. These methods showed comparatively weak PR-AUC and recall@K. The base model displayed a stronger PR-AUC on large objects while the trained model displayed a stronger PR-AUC on small/medium objects. The cause of poor performance is likely due to the global frame embedding being dominated by the background over the desired search object. The trained network performing better on small and medium objects (albeit still relatively poorly) than the base Siamese is plausibly due to the training data containing more small/medium objects thus training the model to focus more on those object sizes.

These findings are consistent with prior work in re-identification and retrieval where global descriptors underperform on cluttered or obscured scenes[26].

5.2.4 FPN Siamese

The FPN Siamese improved across the board upon the base and trained Siamese methods although taking a marginally longer runtime. Most notably, it produces the best score of all models in PR-AUC (0.114) and the second highest in ROC-AUC (0.620) on small objects (<2.5% of the frame) shown in Table 4.3. On Recall@10, the FPN model delivered a gain of +0.086 over the base Siamese (Table 4.2), coming at the cost of a minimal runtime increase (12 minutes per test set vs 9 minutes for the global model). The performance gains can be attributed to the implementation of multi-scale features provided by the FPN. By aggregating features from multiple levels of the backbone (P2–P5), the network combines both fine grained detail (preserving edges and local textures) and coarse semantic context (object shapes and background separation). This directly benefits entities such as badges, logos, and accessories that take up a small fraction of the frame and would otherwise disappear in global feature pooling.

These findings match the literature around Feature Pyramid Networks in object detection, where multi-scale feature fusion was shown to substantially improve recall for small targets[19]. In the context of video frame entity matching, the advantage is even more pronounced, since the discriminative signal from a small object must survive comparison against a much larger and often cluttered background. Therefore, the results validate the hypothesis that explicitly incorporating multi-scale features into embedding models improves open set retrieval on small entities.

5.2.5 Proposal-Verify Pipeline

The proposal-verification pipeline was designed in order to overcome the weakness of global frame embeddings without limiting the pipeline to a closed set limited number of object classes. This was accomplished by first generating a large dictionary of various nouns and adjectives that can be combined to use as prompts which were then embedded and ranked by cosine similarity to the reference image embedding using CLIP. The top k prompts were then selected (with a diversity penalty). This means that although prompts are limited to a combination of words in the dictionary, the dictionary is easily extendable with more nouns and adjectives as well as the combination word limit in the prompt also being extendable. An open vocabulary detector (GroundingDINO) was then used to generate candidate regions using the CLIP selected prompt. Finally, a Siamese network used to match and rank proposals based on the reference object. By creating the pipeline in this way, open set detection becomes conceptually possible as there is no reliance on a fixed training set and the system can be adapted to new unseen categories without retraining. However, in practice the vocabulary is still bound to the prompt set (COCO + LVIS + Visual Genome + colour modifiers) meaning that if an entity is out of the ordinary CLIP may map it to a weak or mismatched prompt causing GroundingDINO's recall to suffer. The proposal-verification pipeline therefore relies heavily on prompt alignment and proposal quality to produce the best results.

As shown in Table 4.1 the pipeline achieves high recall@25/50 (0.773 and 0.864) but poor recall@1 (0.318). Which means that the top ranked frame only has a 31% chance of being correct. This high eventual recall but low initial recall shows that the model is effective at casting a wide net ensuring the entity's eventual detection but its ability to rank frames based on the proposals is poor. This fault therefore lies with the verification head of the pipeline: the global Siamese network. The ability to cast a wide net and produce high eventual recall values is promising for open set detection but the poor PR-AUC, ROC-AUC, and early recall values demonstrates that the pipeline's ranking ability is a major weakness.

The runtime breakdown shown in Table 4.5 highlights that 80% of the pipeline's total runtime is spent on proposal generation. GroundingDINO, like DETR, runs a transformer encoder–decoder over each frame, which is computationally expensive and explains the long runtime. Limiting the number of proposals to 100 per frame clearly helped as shown by DETR taking almost four times longer.

Overall, the pipeline demonstrates a strong potential for real world use where the goal is to ensure that no potential match is missed. However, for cases that require real-time high-precision retrieval the current design is not yet reliable.

5.3 Trade-offs and Practical Deployment

This section focuses primarily on the various trade-offs and practical deployment recommendations of the two proposed methods (FPN Siamese and proposal-verify pipeline).

5.3.1 Accuracy vs Efficiency

Table 4.1 and Table 4.4 show the task performance metrics and the runtime comparisons respectively. Out of the models the FPN Siamese model provides the best balance in speed and accuracy with Siamese based models in general having the shortest runtime. The proposal-verify pipeline, along with the detector based methods like YOLO, Faster R-CNN, DETR, are accurate but are bottlenecked by the proposal generation step (Table 4.5). Of the detectors YOLO is the fastest while being competitive across the board in other performance metrics. Capping the number of proposals per frame helped the proposal-verification pipeline reduce its runtime when compared to the uncapped detector based methods.

5.3.2 Operational Scenarios

Not all real world uses will have the same goals and time constraints, therefore, we can separate the models into distinct scenarios in which they are best suited.

Real-time resource limited scenarios: Lightweight methods like Siamese and ORB are fast and provide reasonable recall albeit not the highest of the models tested. The FPN Siamese model provides a clear improvement on both the base and trained Siamese implementations with minimal additional runtime cost. Additionally, it outperformed ORB in ROC-AUC, PR-AUC and recall at every k except 1. This makes it a prime candidate for use in time limited scenarios where computational resources may also be limited due to its fast and lightweight nature.

High recall search scenarios: The proposal-verification pipeline scores highly for recall at high k meaning that it does a good job of eventually producing a correct prediction. This implies that it would be a good method to use when exhaustive coverage is required and when search items are atypical. The total runtime of the proposal-verification pipeline is longer than the total length of the videos in the test set but when you consider that the search is done automatically requiring little to no manpower it still presents itself as a feasible solution when time and computational requirements are of less importance.

5.3.3 Threshold Selection

The distribution for positive and negative frame scores created from cosine similarity often overlap. Consequently, careful calibration is required to find the correct decision boundary (threshold) that determines if a frame is classified as positive or negative. A lower threshold will lead to fewer false negatives but more false positives and vice versa. As a result, lower thresholds increase recall but risk false positives, while higher thresholds improve precision but risk missing true matches. In testing a fixed $\tau = 0.75$ was used for Siamese verification. Further threshold calibration is therefore a key factor in balancing precision and recall and the overlap in positive and negative score distributions helps explain the low PR-AUC observed across several methods.

5.3.4 Practical Guidance

Method	**Strengths**	**Recommended Use Case**
FPN-Siamese	Improved recall and ROC-AUC on small/medium objects due to multi-scale features. Small runtime cost.	Balanced deployment: settings where both efficiency and robustness to small entities are required.
Proposal-Verify Pipeline	High eventual recall (R@25/50). Adaptable to unseen categories. Strong coverage of difficult cases.	Exhaustive difficult search: settings where completeness is prioritised and longer runtimes are acceptable, settings where objects are atypical (not found in common class labels).

Table 5.1: Practical guidance for selecting methods based on strengths and use cases.

5.4 Threats to Validity

While the results presented provide strong evidence for the relative strengths and weaknesses of each evaluated method, several threats to validity should be acknowledged.

5.4.1 Internal Validity

Test videos were sampled at a stride of 30, this meant that approximately 1 frame per second was taken. This choice was made in order to drastically reduce computational complexity and runtime. However, the repercussion of this decision is that short lived or strong entity appearances may be missed. This means that recall and precision may be underestimated.

Ground truths for the labels were created by using a Python script that converts timestamps indicating where in the video a reference appears into JSON files. These timestamps were recorded manually meaning that there is potential for bias or omission errors.

Hardware constraints led to the decision to use mixed precision and capped proposals per frame. These choices may have affected the metrics negatively and the models may perform differently in higher capacity environments where these limitations do not need to be placed.

5.5 Ethical and Security Considerations

There are several ethical and security implications that need to be considered if the proposed models are to be deployed for legal or forensic use.

Due to the high number of false positives (poor PR-AUC scores) there is a risk of detecting or storing non-target entities such as bystanders. Anonymisation can be applied to videos prior to use via facial blurring or selective video segmentation to reduce personal privacy concerns.

As the models do not have a 100% accuracy, they are prone to incorrect predictions, in high stakes deployment, such as ones were legal action may be pursued a certain level of human interference should take place in order to validate that the model outputs are trustworthy.

Collaboraite's secure deployment environment explicitly prohibits reliance on public internet access. So as to fulfil this requirement all components of the proposed methods, including CLIP, GroundingDINO, and the Siamese verification networks, were implemented using locally stored models and vocabularies to avoid risk of data leakage to external servers.

5.6 Comparisons to Alternatives

This section situates the proposed methods against other possible approaches from the wider literature while accounting for the constraints applied to our task. These constraints are: an open set setting, offline deployment, hardware limitations, and a final use within a forensic context.

CLIP-only retrieval is a possible approach which has been used for zero-shot image search due to its open-vocabulary capabilities. However, CLIP has been shown to have poor localisation and struggle on small objects which is consistent with global embedding methods observed in Table 4.1 (see the base Siamese approach). Global embedding approaches can achieve strong ROC-AUC but collapse in PR-AUC when there is severe class imbalance. This means that CLIP will likely be unsuitable for top-k retrieval.

When researching the area of image video entity matching, re-identification models showed promise. These models have been proven to work well in person/vehicle tracking even with occlusion. The main downside to these models is that they are often limited to domain-specific training (faces, people, cars) but are brittle when used to identify arbitrary entities like logos or badges. Thus making them unsuitable for our open set dataset and by extension the open set problem as a whole.

Finally, recent video transformer architectures (TimeSformer, ViViT, etc) have shown great promise in extending image transformers across frames allowing them to capture motion cues and appearance changes. This ability to handle motion blur and partial occlusion could be very beneficial when matching a static reference image with a moving video object. These models however are very computationally expensive and require large scale video datasets for training, both of which were beyond the limitations of this project.

In contrast to these alternative approaches, the FPN Siamese and proposal-verify pipeline directly address the project's aims and constraints. The FPN Siamese maintains accuracy across a range of object sizes and types while remaining lightweight and fast enough for real world deployment. Its ability on small objects is especially beneficial as these are common in forensic analysis (badges, flags, etc). The proposal-verify pipeline, despite being slower, provides open-vocabulary flexibility and exhaustive coverage without relying on external APIs or internet access, cementing its suitability for use on secure projects. In conclusion, the two proposed methods offer a balance between discriminative ability, open set use and operational feasibility within Collaboraite's requirements.

5.7 Future Work

This project was limited to three months for research, implementation, and writing as such there are still many improvements that can be made and areas to test or address in future work.

5.7.1 Dataset Expansion

The bespoke dataset contained 50 different videos: this could be expanded further. Including a greater variety of videos with different levels of occlusion, lighting, and artifacts could teach the trained models to become even more robust to unseen domains.

The dataset has frame level ground truths which are necessary for accurate training and evaluation. These ground truths could be improved upon even further through the incorporation of bounding boxes which were omitted due to time constraints. This inclusion would enable localisation studies and further size stratified analysis.

5.7.2 Hard Triplet Mining

Triplet loss was used for training the Siamese networks due to its ability to aid in learning discriminative embeddings that have the ability to generalise to unseen objects in an open set matching environment. The triplets were randomly generated using positive and negative frames with a set number of triplets being generated for each reference image video pair. However, wider literature has shown that hard triplet mining outperforms random triplets. Hard triplet mining is the act of purposefully populating triplets with positive and negative frames that are difficult to distinguish by using deliberately challenging negatives and weak positives. It has been shown that hard triplet mining produces stronger training signal and encourages embeddings that better separate close negatives[17]. Hard triplets were not explored in this project due to the difficulty in selecting trustworthy hard triplets, therefore leaving this area open to future work.

5.7.3 Similarity Metric Testing

All methods utilise a form of cosine similarity as the final verification step in order to rank frames by their similarity to the reference object. Cosine similarity was a reasonable choice given the use of embedding normalisation (as it compares relative orientation of embeddings rather than their raw magnitudes). However, its weaknesses (especially threshold calibration and background dominance) directly explain some of the models' low PR-AUC and poor early recall shown in the Results chapter. It also explains why your ROC-AUC was stronger than PR-AUC as positives and negatives were somewhat separable by angle, but overlapping magnitude differences weren't exploited. These shortcoming could be addressed by utilising a different distance/similarity metric.

5.7.4 Proposal-Verification Pipeline Extensions

The testing of the proposed proposal-verification pipeline produced middling results. Although the pipeline shone in certain areas (late k recall) its general performance left much to be desired when compared to the baseline and other model implementations. Thankfully, the pipeline has plenty of areas that could be studied for improvement.

Prompts were generated from a combination set of other dataset classes and descriptors, this vocabulary could be expanded even further with more detailed descriptors and a greater range of class labels being used. Alternatively, a new approach could be implemented that generates sentence like prompts for the reference image using image captioning models. This could provide a more detailed and accurate prompt to be used in the open-vocabulary proposal stage. Future work could also attempt to implement a feedback loop in which after a first run of prompts and proposals are generated, the proposals can be evaluated for their similarity against the reference image. The high similarity proposals could then be used to further refine or expand the prompts and the process repeated until a condition is met (a set number of loops or similarity is reached). This creates an active learning feedback loop instead of relying on a fixed dictionary. Of course this suggestion would come with many challenges such as poor proposals being ranked highly and negatively contributing to the prompt feedback.

A final future work suggestion in regards to the proposal-verify pipeline would be to alter and test multiple verification heads. This project focused mainly on the use of a Siamese model as the verification head in order to align with the other experiments around Siamese models being conducted (incorporating an FPN and comparing trained and untrained Siamese) however, other verification heads could be used in the pipeline. This future work could be used to address the current proposal-verify pipeline's struggles pertaining to poor ranking metrics (its low early recall scores).

5.7.5 Temporal Modelling

As stated in the 'Comparisons to Alternatives' section of this chapter, recent advances in video transformer architectures have shown the ability to extend beyond per-frame matching. The implementation of sequence aware models to better capture video objects is an exciting solution to the image video entity matching problem and may serve as a new way to better verify video entities and overcome motion-blur and obscuration issues.

Chapter 6

Conclusions

This project set out to develop and evaluate methods that tackle the open set image to video entity matching problem. The motivation stemmed from the need for scalable forensic search tools capable of handling novel objects where manual search is slow and error-prone. The problem is particularly challenging due to the need to be robust to small or unusual objects.

This work contributed a bespoke dataset with 50 videos and 160 reference images with ground truth labels for open set video entity matching. Classical features (SIFT, ORB, AKAZE), object detectors (Faster R-CNN, YOLO, DETR), and Siamese embeddings were all systematically benchmarked. Additionally, two novel methods were proposed: a Feature Pyramid Network (FPN) Siamese that integrates multi-scale features, and a CLIP-GroundingDINO-Siamese proposal-verification pipeline with open-vocabulary detection. Results showed that the FPN-Siamese improved small-object retrieval when compared to the trained Siamese, raising overall ROC-AUC from 0.599 to 0.633 and Recall@10 from 0.571 to 0.657. While the proposal-verification pipeline excelled at high recall retrieval (recall@50=0.864), demonstrating its strong exhaustive retrieval capability, although its weaker early recall limits suitability for fast top-k recall. Together these findings highlight the trade of between precision, recall, and efficiency, showing how multi-scale and open-vocabulary pipelines improve both coverage and adaptability in the domain of forensic video search.

For operational use, the FPN-Siamese offers a fast screening tool that works especially well on small entities whereas the proposal-verification pipeline is more suitable for situations where exhaustive search and comprehensive recall is prioritised. In both cases, thresholds need to be tuned to find a balance between false positives and missed detections. Furthermore, human viewing should always be used as a final validation step in order to ensure the evidence's reliability.

Overall this study contributes novel methodologies and empirical benchmarks on an open set image to video entity matching dataset, extending the research in this area and providing practical insights for future developments.

References

[1] CBS, Mar. 2014. [Online]. Available: `https://www.cbsnews.com/boston/news/60-minutes-fbi-scanned-13000-videos-120000-photos-in-boston-marathon-bombings-probe/`.

[2] Collaboraite, Jul. 2024. [Online]. Available: `https://collaboraite.co.uk/capabilities/`.

[3] T. Lindeberg, "Scale invariant feature transform," in *Scholarpedia*. ResearchGate, May 2012, vol. 7. DOI: `10.4249/scholarpedia.10491`.

[4] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, "Orb: An efficient alternative to sift or surf," *2011 International Conference on Computer Vision*, pp. 2564–2571, 2011. [Online]. Available: `https://api.semanticscholar.org/CorpusID:206769866`.

[5] L. Kalms, K. Mohamed, and D. Göhringer, "Accelerated embedded akaze feature detection algorithm on fpga," in *Proceedings of the 8th International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies*, ser. HEART '17, Bochum, Germany: Association for Computing Machinery, 2017, ISBN: 9781450353168. DOI: `10.1145/3120895.3120898`. [Online]. Available: `https://doi.org/10.1145/3120895.3120898`.

[6] S. Bonilla, C. D. Vece, R. Daher, *et al.*, *Mismatched: Evaluating the limits of image matching approaches and benchmarks*, 2024. arXiv: `2408.16445 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2408.16445`.

[7] R. W. Hamming, "Error detecting and error correcting codes," *The Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950. DOI: `10.1002/j.1538-7305.1950.tb00463.x`.

[8] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, "Kaze features," in *Computer Vision – ECCV 2012*, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 214–227, ISBN: 978-3-642-33783-3.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, *Rich feature hierarchies for accurate object detection and semantic segmentation*, 2014. arXiv: `1311.2524 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/1311.2524`.

[10] R. Girshick, *Fast r-cnn*, 2015. arXiv: `1504.08083 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/1504.08083`.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, *You only look once: Unified, real-time object detection*, 2016. arXiv: `1506.02640 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/1506.02640`.

[12] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3, 2006, pp. 850–855. DOI: `10.1109/ICPR.2006.479`.

[13] M. F. Tariq and M. A. Javed, *Small object detection with yolo: A performance analysis across model versions and hardware*, 2025. arXiv: `2504.09900 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2504.09900`.

[14] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, *Deformable detr: Deformable transformers for end-to-end object detection*, 2021. arXiv: `2010.04159 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2010.04159`.

[15] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, *End-to-end object detection with transformers*, 2020. arXiv: `2005.12872 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2005.12872`.

[16] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, 539–546 vol. 1. DOI: `10.1109/CVPR.2005.202`.

[17] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2015, pp. 815–823. DOI: `10.1109/cvpr.2015.7298682`. [Online]. Available: `http://dx.doi.org/10.1109/CVPR.2015.7298682`.

[18] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," in *Proceedings of the 7th International Conference on Neural Information Processing Systems*, ser. NIPS'93, Denver, Colorado: Morgan Kaufmann Publishers Inc., 1993, pp. 737–744.

[19] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, *Feature pyramid networks for object detection*, 2017. arXiv: `1612.03144 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/1612.03144`.

[20] H.-X. Yu, W.-S. Zheng, A. Wu, X. Guo, S. Gong, and J.-H. Lai, *Unsupervised person re-identification by soft multilabel learning*, 2019. arXiv: `1903.06325 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/1903.06325`.

[21] A. Radford, J. W. Kim, C. Hallacy, *et al.*, *Learning transferable visual models from natural language supervision*, 2021. arXiv: `2103.00020 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2103.00020`.

[22] S. Liu, Z. Zeng, T. Ren, *et al.*, *Grounding dino: Marrying dino with grounded pre-training for open-set object detection*, 2024. arXiv: `2303.05499 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2303.05499`.

[23] H. Zhang, F. Li, S. Liu, *et al.*, *Dino: Detr with improved denoising anchor boxes for end-to-end object detection*, 2022. arXiv: `2203.03605 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2203.03605`.

[24] P. C. Chhipa, K. De, M. S. Chippa, R. Saini, and M. Liwicki, *Open-vocabulary object detectors: Robustness challenges under distribution shifts*, 2024. arXiv: `2405.14874 [cs.CV]`. [Online]. Available: `https://arxiv.org/abs/2405.14874`.

[25] G. Bradski, "The opencv library," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2000.

[26] G. Wang, Y. Yuan, X. Chen, J. Li, and X. Zhou, "Learning discriminative features with multiple granularities for person re-identification," in *Proceedings of the 26th ACM international conference on Multimedia*, ser. MM '18, ACM, Oct. 2018, pp. 274–282. DOI: `10.1145/3240508.3240552`. [Online]. Available: `http://dx.doi.org/10.1145/3240508.3240552`.