# CAMERA CLUSTERING FOR SCALABLE STREAM-BASED ACTIVE DISTILLATION

#### A PREPRINT

Dani Manjah\*
dani.manjah@uclouvain.be

Davide Cacciarelli d.cacciarelli@imperial.ac.uk

© Christophe De Vleeschouwer christophe.devleeschouwer@uclouvain.be Benoît Macq benoit.macq@uclouvain.be

# ABSTRACT

We present a scalable framework designed to craft efficient lightweight models for video object detection utilizing self-training and knowledge distillation techniques. We scrutinize methodologies for the ideal selection of training images from video streams and the efficacy of model sharing across numerous cameras. By advocating for a camera clustering methodology, we aim to diminish the requisite number of models for training while augmenting the distillation dataset. The findings affirm that proper camera clustering notably amplifies the accuracy of distilled models, eclipsing the methodologies that employ distinct models for each camera or a universal model trained on the aggregate camera data.

Keywords Object Detection, Knowledge Distillation, Active Learning, Neural Network Deployment.

#### **1** Introduction

Traditional training methodologies for Deep Neural Networks (DNNs) face substantial barriers, including significant time, financial investment, and labor for dataset preparation and training Beyer et al. [2022]. These challenges create limitations in operating and scaling ever-evolving video-analytics systems such as city-scale CCTV because 1) DNNs need to be regularly fine-tuned to maintain performance and 2) update costs increase with the number of cameras.

To address these issues, we propose a large general-purpose model fine-tuning compact DNNs on subsets of cameras. Our method facilitates localized consistent updates, which are crucial for maintaining local DNN performance in individual cameras Manjah et al. [2023] and promoting cost-efficient training and scalability Hill [1990], Hossfeld et al. [2023], Fox et al. [1997]. This paper builds on the Stream-Based Active Distillation (SBAD) mechanism, initially presented in Manjah et al. [2023], which selects images from a single stream on-the-fly for training a low-complexity DNN optimized for local conditions. We propose a novel multi-stream SBAD approach based on clustering camera nodes in a multi-camera environment. This method rests on the premise that models fine-tuned on a specific camera domain are likely to transfer effectively to other domains with similar characteristics.

Our cluster-based fine-tuning (Fig. 1) reduces the marginal cost of training additional units and enhances DNN accuracy by striking a balance between specificity (allocating a model for each camera) and universality (utilizing a larger, diverse training set). In other words, stream aggregation not only reduces the requirement for numerous student models but also improves their prediction accuracy compared with training a separate model for each stream with only its specific images. Moreover, this method achieves higher accuracy than training a universal student model with images from all streams, indicating that employing multiple clusters leads to superior accuracy.

<sup>\*</sup>Dani Manjah and Benoit Macq are affiliated with Université catholique de Louvain, Belgium. Christophe De Vleeschouwer is funded by the National Fund for Scientific Research (FNRS). Davide Cacciarelli is affiliated with Imperial College London. 16/04/2024 - This manuscript is currently under review at IEEE Transactions on Circuits and Systems for Video Technology.



Figure 1: Updates of video analytics models: Cameras (C1 to C7) send selected image data to a central server, which pseudo-labels, trains and updates specialized *Students* models for groups of similar cameras. These updated models are then sent back to their associated cameras.

Our key contributions include:

- 1. The validation of active distillation in a multi-camera setup. We showed that selecting the frames obtaining the highest confidence score by local models results in the most accurate DNN. This includes an analysis of the potential biases of model-based pseudo-annotations.
- 2. Introduction of a novel camera-clustering based on model cross-performance, coupled with an in-depth analysis of its impact on model accuracy. This dual contribution showcases model compactness improvements without sacrificing accuracy and explores how various clustering strategies affect training complexity and model performance.
- 3. Novel data and codebase available at https://github.com/manjahdani/CSBAD/ to support further research.

# 2 Related Work

We previously introduced Stream-Based Active Distillation (SBAD), a technique combining Knowledge Distillation (KD) with Active Learning (AL) to train compact DNN models tailored for their video-stream Manjah et al. [2023]. We had developed sampling techniques for unlabeled streaming data, mitigating labeling inaccuracies and aiming to assemble the smallest training set that achieves the best local performance. This paper expands upon this by proposing Clustered Stream-Based Active Distillation (CSBAD), which organizes cameras' data into clusters; balancing training complexity and improving model accuracy. This section delves into foundational principles of SBAD — specifically, KD and AL — and reviews the existing literature on scene clustering.

# 2.1 Knowledge Distillation

KD is a process where a compact *Student* model learns from a more complex *Teacher* model Hinton et al. [2015]. It is used to create lightweight models that are suited to limited storage and computational capacities Cheng et al. [2018], as encountered on edge devices Mishra and Gupta [2023], Tanghatari et al. [2023]. Through regular updates, it can ensures that the *Student* model continues to perform effectively even when operating on changing data distributions Saito et al. [2019].

Recently, KD has supported innovative applications in video analytics, particularly through *Online Distillation* Cioppa et al. [2019], Mullapudi et al. [2019], where a compact model's weights are updated in real time to mimic the output of a larger, pre-existing model. Other notable applications include *Context- and Group-Aware Distillation* Rivas et al. [2022], Habibian et al. [2022], Vilouras et al. [2023], which accounts for context and/or sub-population shifts when

delivering student models. Finally, *Active Distillation* Baykal et al. [2022] combines AL with KD to train students with a reduced number of appropriately selected samples, while *Robust Distillation* Goldblum et al. [2020] explores the transfer of adversarial robustness from *Teacher* to *Student*.

In our work, we studied the impact of using KD instead of a human annotator on Students' accuracy.

# 2.2 Active Learning

AL aims to select and label the most informative data points to reduce the number of samples required to train a model Settles [2009]. AL can be implemented offline (Pool-Based AL) or on-the-fly (Stream-Based AL), from continuous streams.

Stream-Based AL is the most relevant for video-based systems, where vast amounts of unlabeled data arrive continuously, making data storage impractical. However, most of the AL methods developed for Deep Neural Networks (DNNs) have been focused on the pool-based scenario Sener and Savarese [2017], Yoo and Kweon [2019], Ash et al. [2019], Elhamifar et al. [2013], Agarwal et al. [2020], Prabhu et al. [2020], Sinha et al. [2019], Yuan et al. [2021]. Stream-based AL has mostly been studied in conjunction with classification or regression models Cacciarelli et al. [2022a, 2023, 2022b], with traditional statistical models Rožanec et al. [2022], Narr et al. [2016]. Those methods do not generalize to DNNs. In contrast, our work considers AL to train a DNN with images labeled by a *Teacher* model (and not a human). As an original contribution, we show that the confidence associated with automatic labeling of training samples plays a crucial role in the AL selection process.

# 2.3 Scene Clustering

In multi-camera systems, the technique of scene clustering plays a crucial role in aggregating and synthesizing information from diverse sources effectively. This technique supports video summarization Aner and Kender [2002], anomaly detection Sun and Gong [2023], and content recommendation Wang et al. [2013], with hierarchical methods addressing redundancy in tracking and overlapping scenes Li et al. [2023], Szűcs et al. [2023], Wang et al. [2008], Specker et al. [2021], Patel et al. [2021], Simon et al. [2010]. Despite advances in techniques such as multiview clustering Peng et al. [2023], Huang et al. [2023], a significant challenge persists in **selecting representative frames for each video** and, to the best of our knowledge, no previous work has studied the aggregation of views to reduce the number and improve the accuracy of student models serving a multi-camera system. Our research introduces a novel scene-clustering approach that groups videos based on the cross-dataset performance of models with the aim of improving model training and system efficiency in multi-camera environments.

# **3** Problem Statement

We focus on large-scale networks where nodes, equipped with compact DNNs, analyze video-streams such as city-scale CCTVs. All nodes are connected to a central training server. In our system, the central unit collects the images from the nodes, annotates them, and trains the DNNs to be deployed on the nodes. In practice, DNNs may require periodic updates or adjustments when underperforming due to changes in the operating conditions. Furthermore, the system can grow with the deployment of additional sources of video. Therefore, we seek a **cost-effective scalable** Hill [1990], Hossfeld et al. [2023] training methodology for DNNs suited to the visual specificity of their video streams.

Since automatic management of the system is desired Fox et al. [1997], annotation of the collected data is implemented using a large, general purpose *Teacher* model instead of human annotators. In practice, the *Teacher* may be inaccurate, potentially causing *Students* DNNs to overfit to incorrect pseudo-labels. This phenomenon is generally referred to as *confirmation bias* Arazo et al. [2020], and is further studied in Section 6.2.

Obviously, the amount of data collected per stream and the number of trained DNNs directly affect the maintenance (DNNs update) costs and system scalability. To mitigate training complexity in large-scale systems, we propose to cluster streams based on their similarities and to train a single model for each cluster. Choosing the right number of clusters is not trivial. Indeed, fewer but larger clusters allow models to train on a broader diversity, enhancing adaptability Chen et al. [2023]. Conversely, numerous but smaller clusters fine-tune models to the nuances of individual streams but are trained with less data Manjah et al. [2023].

Given a fixed number of training images per stream, the partitioning in clusters also calls for reevaluation of the training management. Indeed, in DNN training, the number of times a model updates its weights in one epoch depends on the amount of training images. Hence, with a fixed number of training epochs, models trained over smaller clusters update their weights less frequently. This disadvantages those models compared with those trained on larger clusters. Therefore, in scenarios where computational power is abundant, we could investigate increasing the number of epochs

to reach the model's peak accuracy even when the training set gets smaller. In this context, our work addresses four research questions:

- 1. How can we select images from each stream to maximize the resulting model's accuracy?
- 2. To what extent does the quality of pseudo-labels affect downstream models' performance? How can we mitigate the impact of inaccurate pseudo-labels and reduce the risk of confirmation bias?
- 3. Given a constant number of epochs per model, thereby keeping the overall training complexity constant, which partitioning results in the highest model performance?
- 4. Given a constant number of iterations (gradient descents) per model, which partitioning results in the highest model performance?

# 4 Methodology

Consider a set S of N video streams  $\{X_1, \dots, X_N\}$ , with each  $X_i$  linked to **a compact** Student model,  $\theta_i$  for object detection. To maintain Students up-to-date, models are fine-tuned on a central server using selected samples from the videos. The two-level system, encompassing Students and a Teaching Server, is detailed below.

#### 4.1 Student-level

Students operate on cost-effective hardware and use a SELECT (I) function to decide whether an image I is informative to train the model. SELECT functions must be *efficient*. Efficiency is crucial; if SELECT takes longer than the frame interval to make decisions, a buffer must be used to store incoming images. This increases the demand for data storage, thus conflicting with scalability goals. Additionally, to reduce training cost, the set of selected frames and their associated pseudo-labels, which constitutes the training set  $\mathcal{L}_i$ , must not exceed a frame budget B per student, hence,  $|\mathcal{L}_i| \leq B$ , for all  $i = 1, \dots, N$ .

#### 4.2 Teaching Server

To moderate annotation costs, the images are pseudo-labeled by a universal but imperfect model  $\Theta$  referred to as *Teacher* Buciluundefined et al. [2006]. Upon receiving an image I from a stream  $\mathcal{X}_i$ ,  $\Theta$  pseudo-labels it by generating a set of bounding boxes  $\tilde{P}$  and then adds the pair  $(I, \tilde{P})$  to the image stream's database  $\mathcal{L}_i$ , associated to the  $i^{th}$  student. This process results in a collection of sets  $\mathcal{L}_1, \dots, \mathcal{L}_N$ . Next, the server employs a CLUSTER method that partitions the databases  $\{\mathcal{L}_i\}_{i=1,\dots,N}$  into  $K \leq N$  training sets  $\{E_1, \dots, E_K\}$ . The CLUSTER approach builds on the premise that models are transferable between similar streams and is detailed in Section 4.4. We also define a mapping function,  $map(\cdot)$ , linking *Students* to their clusters, thus enabling the appropriate updating of their weights post-training.

K models are then trained on each  $E_k$ , with  $k = 1, \dots, K$ , and deployed on associated cameras. Algorithm 1 provides the pseudo-code of the method.

This research aims to explore the trade-offs involved in choosing K. On one hand, a small K implies models trained with a broad diversity and good applicability across various students, potentially with improved generalization. On the other hand, a K approaching N may tailor models to the nuances of individual cameras.

#### 4.3 Our SELECT Strategy

Our investigation into video sampling strategies underscored the superiority of the TOP-CONFIDENCE approach. In this method, the local DNN *Student*  $\theta$  begins with a warm-up phase by inferring on w frames, without selecting any for training. For each frame  $I_t$ , where  $t \in \{1, \dots, w\}$ , it produces a set of L bounding boxes  $\tilde{P}_t = \{\tilde{p}_{1t}, \dots, \tilde{p}_{Lt}\}$  along with their associated confidence score  $C_t = \{c_{1t}, \dots, c_{Lt}\}$  predictions, computed as Redmon et al. [2016]. Each frame is then scored  $C(I_t)$  based on the highest confidence score, that is,  $C(I_t) = \max_l c_{lt}$ , where  $l \in \{1, \dots, L\}$ . This process establishes a threshold  $\Delta$  as the  $(1 - \alpha)$ -upper percentile of  $C(I_t)$ , aiming for an  $\alpha$  selection rate where  $\mathbb{P}(C(I_t) \ge \Delta) = \alpha$ . Subsequently, for t > w, a frame  $I_t$  is forwarded to the Server if  $C(I_t) \ge \Delta$ . The pseudo-code is presented in Algorithm 2.

Note that  $\alpha$  can be carefully increased to allow more permissive selection, meeting the image budget. This was applied in our experiments (see Section 5.4).

### Algorithm 1 CSBAD Framework

**Require:** pre-trained student model  $\theta$ , a general purpose teacher model  $\Theta$ , a training frame budget B, a SELECT strategy, image streams  $\{X_1, \dots, X_N\}$ , a mapping rule CLUSTER, K partitions. **Ensure:**  $B \ge 1, 1 \le K \le N$ .

▷ INITIALIZATION & SAMPLING 1: for  $\mathcal{X}_i \in \mathcal{S}$  do 2:  $\theta^{\mathcal{X}_i} \leftarrow \theta^{general}$  $\triangleright i = 1, \cdots, N$  $\mathcal{L}_i \leftarrow \emptyset$ 3:  $t \leftarrow 0$ 4: ▷ Timestamp 5: while  $|\mathcal{L}_i| \leq B$  do Observe current frame  $I_t$ 6: if  $SELECT(I_t) = TRUE$  then 7:  $\tilde{P}_t \leftarrow \Theta(I_t)$ ▷ Infer pseudo-labels 8:  $\mathcal{L}_i \leftarrow \mathcal{L}_i \cup (I_t, \tilde{P}_t)$ 9: 10: end if 11:  $t \leftarrow t + 1$ end while 12: 13: end for ▷ CLUSTERING 14:  $\{E_1, \cdots, E_k\} \leftarrow \mathsf{CLUSTER}(\mathcal{L}_1, \cdots, \mathcal{L}_N; K)$ ▷ STUDENTS FINE-TUNING 15: for  $k = 1, \dots, K$  do  $\theta^{E_k} \leftarrow train(\theta, E_k)$ 16: 17: end for ▷ UPDATE MODELS 18: for  $i \in 1, \dots, N$  do  $\theta_i \leftarrow \theta^{E_{map(i)}}$ 19: 20: end for 21: **return**  $\{\theta_i\}_{i=1,...,N}$ ▷ Return updated models

#### Algorithm 2 TOP-CONFIDENCE Thresholding for Object Detection

**Require:** model  $\theta$ , budget B, warm-up period w, selection rate  $\alpha$ .

1: Initialize confidence score array C with size w

2: **for** t = 1 to w **do**  $\tilde{P}_t, \tilde{C}_t \leftarrow \theta(I_t)$ 3:  $C[t] \leftarrow \max(\tilde{C}_t)$ 4: 5: end for 6:  $\Delta \leftarrow \text{percentile}(C, 1 - \alpha)$ 7:  $i \leftarrow 0$ 8: while i < B and more frames available **do** 9:  $C_t \leftarrow \max(\theta(I_t))$ if  $C_t \geq \Delta$  then 10: Forward  $I_t$  to server;  $i \leftarrow i+1$ 11: 12: end if 13:  $t \leftarrow t + 1$ 14: end while

 $\triangleright \text{ Collect Confidence Scores}$  $\triangleright \text{ Infer boxes and scores}$  $\triangleright \text{ Max score for } I_t$  $\triangleright \text{ Compute selection threshold } \Delta$  $\triangleright \text{ Forward frames meeting } \Delta \text{ until budget is met}$  $\triangleright \text{ Selected frames counter}$ 

 $\triangleright \text{ Get max score for new } I_t$  $\triangleright \text{ If meets } \Delta, \text{ forward}$ 

▷ Next frame

#### 4.4 Clustering

We propose a node clustering method for DNN training. Our method is based on the premise that once a model is fine-tuned on a particular camera domain, it will likely transfer effectively to other camera domains with similar features. Conversely, significant performance drops are expected across dissimilar domains.

CLUSTER uses a *Hierarchical Clustering Algorithm* after the initial training of N stream-specific *Student* models. This algorithm does not need a pre-specified number of clusters. Instead, the resulting dendrogram can be cut at the appropriate level to obtain an appropriate number of clusters.

Before discussing our method, we define the validation set  $\mathcal{V}_i$  as the set of image and pseudo-label pairs  $(I_{it}^{val}, \tilde{P}_{it})$  from a stream  $\mathcal{X}_i$ . We also define the function  $f(\theta, \mathcal{V})$  measuring the quality of a model  $\theta$  on a set  $\mathcal{V}$ .

**STEP 1 – Train** N **Stream-specific Models:** We fine-tune  $N^2$  stream-specific models, denoted by  $\theta^{\mathcal{L}_i}$ . They will subsequently be grouped according to their validation performance on the sets  $\{\mathcal{V}_1, \ldots, \mathcal{V}_N\}$ . From an operational standpoint, the choice of B and number of epochs for this step should align with your specific constraints and use case. Generally, a higher training budget B allows for better capture of the stream's distribution, thus the transferability (or not) of models will be more pronounced, hence leading to more effective clustering.

**STEP 2 – Compute Cross-Domain Performance:** We compute the cross-domain performance matrix  $M \in \mathbb{R}^{N \times N}$ , with each element  $M_{ij}$  representing the performance metric  $f(\theta^{\mathcal{L}_i}; \mathcal{V}_j)$ , which indicates how a model trained on domain  $\mathcal{X}_i$  performs on the validation set of  $\mathcal{X}_j$ . The matrix is defined in Eq.1.

$$M := \begin{bmatrix} f\left(\theta^{\mathcal{L}_{1}}; \mathcal{V}_{1}\right) & \cdots & f\left(\theta^{\mathcal{L}_{1}}; \mathcal{V}_{N}\right) \\ \vdots & \ddots & \vdots \\ f\left(\theta^{\mathcal{L}_{N}}; \mathcal{V}_{1}\right) & \cdots & f\left(\theta^{\mathcal{L}_{N}}; \mathcal{V}_{N}\right) \end{bmatrix}$$
(1)

**STEP 3 – Compute Distance Matrix:** To quantify the dissimilarities between the models trained on different domains, we calculate the distance matrix D. This matrix is computed using the pairwise Euclidean distances between the models' performance vectors, derived from the cross-domain performance matrix M.

The distance matrix D is defined as:

$$D_{ij} = \sqrt{\sum_{k=1}^{N} (M_{ik} - M_{jk})^2} \quad \text{for } i, j = 1, \dots, N$$
(2)

**STEP 4 – Apply Single Linkage Clustering:** Using Sibson [1973], we construct a dendrogram, i.e., a tree diagram depicting how clusters merge based on minimum distance. Formally, for two clusters A and B, the single linkage distance L(A, B) is given by Eq. 3

$$L(A,B) = \min\{D_{ij} : i \in A, j \in B\}$$
(3)

where  $D_{ij}$  is defined as in Eq. 2. This linkage criterion tends to merge clusters with their nearest elements.

**STEP 5 – Define Clusters:** We define the clusters by selecting a cut-off distance t on the dendrogram (see Fig. 3b), guided by expertise, statistical methods, or visual analysis.

#### **5** Materials And Methods

This section elaborates on the datasets, models, and evaluation metrics employed in our study.

#### 5.1 Dataset

The Watch and Learn Time-lapse (WALT) dataset Reddy et al. [2022] features footage from nine cameras over several weeks, offering a variety of viewpoints, lighting and weather conditions, including snow and rain. Data collection spans periods ranging from one to four weeks per camera, resulting in a weekly range of 5,000 to 40,000 samples due to asynchronous recordings and differing sampling rates (details can be found in the GitHub repository). Notwithstanding, some instances of burst phenomena are observed, presenting cases of strong temporal redundancy.

<sup>&</sup>lt;sup>2</sup>When N is too large, we can randomly select a small subset of the streams to conduct this step. This bounds the number of stream-specific models to train, while providing sufficient diversity. The resulting cluster models are then deployed based on their performance on the node's validation set.

#### 5.2 Evaluation

System performance was evaluated by the accuracy of *Student* models on their respective camera.

#### 5.2.1 Test sets

We had nine test sets, two from the original paper and seven that we annotated by selecting a week of footage from each camera, which we excluded from training. These sets, now publicly available, facilitate further research (details and links can be found in Supplementary Materials and Github repository).

#### 5.2.2 Detection accuracy

In object detection model evaluation, the mAP50-95 metric signifies the mean Average Precision (mAP) across various Intersection over Union (IoU) thresholds, spanning from 0.50 to 0.95 in increments of 0.05.

#### 5.2.3 Adjusted Training cost

We define the training cost T as the total number of training iterations, which equates to the number of weight updates within a model. To facilitate fair comparisons among models trained on clusters of varying sizes, we introduce an adjusted training cost  $T^N$  for a specified number of streams N. Considering the number of samples per stream B, we adjust the epoch count to ensure that the iteration count remains consistent regardless of the cluster count K. This calibration for models trained on any given cluster k, with  $k \in \{1, \ldots, K\}$ , is encapsulated in Eq. 4:

$$T_k^N = \frac{B}{\text{batch size}} \times \frac{\text{epochs}_{|K=1} \times N}{n_k}$$
(4)

Here,  $n_k$  represents the stream count in cluster k, B is the number of images collected per stream, batch size is the number of samples processed in one forward and backward pass, and epochs<sub>|K=1</sub> denotes the number of epochs for the universal model, i.e., when K = 1.

#### 5.3 Models, Training and Distillation Implementation

We utilize models pretrained on the COCO dataset Lin et al. [2014] to ensure a broad foundational understanding of the object detection task. We employ **YOLOv8x6**<sup>COCO</sup> as the *Teacher*  $\Theta^{COCO}$  and **YOLOv8n**<sup>COCO</sup> as the *Student*:  $\theta^{COCO}$ . The architectures have 68.2 and 3.2 million parameters, respectively. Both models serve as the evaluation benchmark. No preprocessing was done, but for post-processing, similar COCO object classes (bikes, cars, motorcycles, buses, and trucks) were grouped into a "vehicle" category. The *Student* is then fine-tuned by default on 100 epochs, which can vary according to the experiment. The batch size remains constant at 16. All other training parameters are defaults as configured in Jocher et al. [2023].

#### 5.4 SELECT an Parameters

Our investigation evaluates, alongside Top-Confidence, three other SELECT strategies:

- Uniform-Random: a frame is chosen if  $s \sim U(0,1) \ge 1 \alpha$ . To ensure statistical reliability, we conducted six iterations with six different seeds. The variability in performance is quantified by the Margin of Error (MoE), calculated as MoE =  $z \times \frac{\sigma}{\sqrt{n}}$ , where z represents the 1.96 z-score correlating with a 95% confidence level,  $\sigma$  is the standard deviation of the scores, and n is the effective sample size.
- Least-Confidence: targets frames with minimal prediction confidence to present the model with challenging instances. This method mirrors the Top-Confidence algorithm (2), albeit with a reversed selection criterion, shifting from  $C(I_t) \ge \Delta$  to  $C(I_t) \le \Delta$ .
- N-First: opts for the initial *B* frames from the stream, establishing an unbiased baseline.

Standard parameters are set to  $\alpha = 0.1$  and w = 720. Relaxation was done in high-budget scenarios for cameras 3 and 9 due to data constraints, increasing  $\alpha$  to 0.55 for camera 9 and 0.25 for camera 3, thus facilitating expedited frame selection to fulfill our image budget.

# **6** Results

Section 6.1 assesses the efficacy of various SELECT sampling strategies and examines how the number of images per stream (B) affects the performance of individual *Student* models.

Next, we analyzed the correlation between the complexity of *Teacher* models and their tendency to induce confirmation bias, as detailed in Section 6.2.

Section 6.3 considers the clustering of cameras. Sections 6.4 and 6.5 investigate the influence of cluster count and training complexity on model performance.

#### 6.1 Impact of SELECT and B



Figure 2: Average mAP50-95 scores for different training sample sizes, using four SELECT models. The number of epochs is 100. Key observation are: 1) Top-Confidence is the best sampling strategy and 2) fine-tuned compact models can outperform a *Teacher* YOLOv8x6<sup>COCO</sup>.

Student  $\theta^{\text{COCO}}$  is fine-tuned using B samples from a camera, selected on-the-fly according to the SELECT strategies (detailed in Section 5.4). The resulting model is then evaluated on the camera's test set. The low budget  $B \leq 256$  experiments were conducted across all cameras for each week, resulting in eighteen pairs per experiment. For the high budget B > 500 scenario, we combined weekly data from each camera to reach the budget count, forming nine pairs per experiment.

Figure 2 presents the average mAP50-95 scores as a function of *B* across different SELECT strategies, maintaining a consistent 100 epochs for all tests. Key observations include the following:

- 1. Top-Confidence is the best sampling strategy. Conversely, N-First was catastrophic and performed worse than the student when a small B was considered.
- 2. Using Top-Confidence, we outperformed the unfine-tuned models  $\theta^{\text{COCO}}$  and  $\Theta^{\text{COCO}}$  with sample budgets exceeding 48 and 750, respectively.
- 3. As B increases, the model quality improves but at a slower rate, suggesting the achievement of peak accuracy.

The findings emphasize the need for careful design of SELECT, particularly when the budget is limited. We observe that when *B* is small, a bad selection can severely disrupt learning. Methods such as N-First and Random are **ill-suited for** the redundant nature of video streams and tend to select frames with little or no relevant activity, penalizing the learning process. Conversely, the Top-Confidence approach tends to select frames containing objects. Furthermore, opting for higher-confidence frames can enhance the accuracy of pseudo-labels in a distillation scheme, while selecting the least confident frames tends to amplify the teacher's inaccuracies. The section 6.2 delves into this phenomenon, known as confirmation bias.

Although DNNs generally benefit from larger training sets, our observations reveal a plateau, indicating that beyond B = 1500, adding more images from the same stream and acquisition period does not increase model performance further.

#### 6.2 Confirmation bias

Confirmation bias in semi-supervised or unsupervised learning can lead to significant errors by reinforcing incorrect predictions and thereby misleading learning Arazo et al. [2020]. To identify strategies mitigating it, we conducted two experiments exploring the interplay of Least/Top-Confidence with varying i) *Teacher* and ii) *Student* sizes. Model size (Params) is quantified by the number of parameters in millions. In both experiments, models were trained for 100 epochs.

#### 6.2.1 Impact of Teacher Size

In this experiment, we manually annotated 96 images sampled by a  $\theta^{COCO}$  Student utilizing the Least/Top-Confidence approach and subsequently compared the performance of these Students against scenarios in which a Teacher model pseudo-annotated the same set of images. This experiment was conducted across three camera streams<sup>3</sup>, employing various model sizes for the Teachers.

Table 1: mAP50-95 values, and percentage increase per annotator for Least/Top-Confidence. The *Student* is  $YOLOv8n^{COCO}$  with B = 96 samples and 100 epochs.

Annotator	Params (M)	Least	Тор	% Increase
yolov8n	3.2	0.28	0.45	56%
yolov8s	11.2	0.35	0.49	37%
yolov8m	25.9	0.37	0.52	40%
yolov8l	43.7	0.40	0.51	30%
yolov8x6	68.2	0.40	0.52	31%
human	N/A	0.40	0.52	27%

The results, detailed in Table 1, reveal that less complex *Teachers* tend to yield inferior *Students*, a situation that the Least-Confidence sampling strategy exacerbates. Intriguingly, human annotations did not offer improvements. This suggests the Least-Confidence's tendency to choose ambiguous or challenging samples, thereby reinforcing confirmation bias. Conversely, Top-Confidence significantly improves pseudo-label quality and, consequently, model performance, highlighting the critical need for selecting high-quality, clear images for training. Moreover, this finding reiterates the effectiveness of pseudo-labeling, indicating that it nearly matches the performance enhancements provided by human-generated labels.

# 6.2.2 Impact of Student Size

We explored the influence of *Student* model size on performance, utilizing B = 256 images from nine cameras, pseudo-labeled by  $\Theta^{\text{COCO}}$ .

Our observations, detailed in Table 2, yield two key insights, namely, 1) a general enhancement in performance with the increase in *Student* model size, and 2) a growing disparity in the effectiveness of Least/Top-Confidence strategies as the model size expands. These findings indicate that larger models are better equipped to regularize and generalize, a crucial aspect in correcting the inaccuracies inherent in pseudo-labels generated through the Least-Confidence strategy. Furthermore, the Top-Confidence strategy appears to leverage the advanced capabilities of bigger models more efficiently.

# 6.3 Cluster Definition

The cross-domain performance matrix M in Figure 3a, displays the mAP50-95 score of a model  $\theta^{\text{COCO}}$  fine-tuned on source  $cam_i$  and tested on target  $cam_j$ , where i and j are the nine cameras from the WALT dataset. The analysis utilized the first week's data from each camera, sampling B = 256 images using Top-Confidence.

Key observations from Figure 3a are:

<sup>&</sup>lt;sup>3</sup>Namely, Camera 1 from Week 1, Camera 2 from Week 1, and Camera 3 from Week 5. Links to the manual annotations can be found in the GitHub repository.

Table 2: Comparison of mAP50-95 Values and Percentage Increase per *Student* for Different Strategies. The *Teacher* is YOLOv8x6<sup>COCO</sup> and B = 256 samples and 100 epochs were used.

Student	Params (M)	Least	Top	% Increase
volov8n	3.2	0.52	$\frac{10p}{0.53}$	+2.8%
y010v811	11.2	0.52	0.55	+2.070
yolovas	11.2	0.50	0.58	+2.4%
yolov8m	25.9	0.56	0.59	+4.6%
yolov8l	43.7	0.57	0.60	+4.3%
yolov8x	68.2	0.57	0.60	+5.0%

- 1. Models perform best within their own domain.
- 2. The transfer of models results in performance degradation. The severity is variable.

The variance in the model's transferability across the stream can be used to group the streams. Having verified our premises, we apply Hierarchical Clustering with *minimum linkage* to detect subtle similarities. Note that in this particular scenario the application of other linkage methods (i.e., *max, average* or *ward*) leads to the same clustering. Cluster definition involved experimenting with various threshold values and observing emerging groups in the dendogram (Figure 3b). The clustering obtained for the nine WALT cameras is presented in Table 3 in Appendix.



Figure 3: Clustering Definition. Fig. 3a depicts the cross-performance matrix M, where each element  $M_{ij}$ ,  $i, j = 1, \dots, 9$ , represents the mAP50-95 score of a model  $\theta_i$  retrained on source domain  $cam_i$  and evaluated on target domain  $cam_j$ . The setting is based on B = 256 images sampled using Top-Confidence. Fig 3b is the associated dendogram.

#### 6.4 Clustering vs Samples Budget

Figure 4 presents mAP50-95 scores as a function of the amount of images per stream B for different numbers of clusters K. The clusters' definition follows Fig. 3b. The observations are as follow:

- 1. On smaller budgets, training one model for all streams is superior. In fact, the smaller K is, the better are the models.
- 2. On larger budgets, K = 2 and K = 3 are dominating while the scenario K = N = 9 is underperforming.
- 3. For all  $K \in (1, \dots, 9)$ , the models reach their peak accuracy around B = 1500 samples per stream equivalent to total amount of iterations T = 84375, irrespective of K.

The amount of images per stream B dictates the best K. Training a single model for all streams proves most effective for a low sample budget B, reflecting the data-hungry nature of DNN. However, for a larger B, clustering increases

performance. In fact, there is a a "sweet spot" between stream-specific and universal models. This suggest that clustering offers robustness by leveraging more samples and a more specific distribution facilitating the learning, particularly in less complex architectures. Yet, there's a potential downside to excessive clustering, as it might overlook the need for a model of diversity.



Figure 4: Mean mAP50-95 scores per sample budget per stream (B) for varying numbers of clusters were observed. Models underwent training for 100 epochs with a batch size of 16. Key findings indicate that, at a constant complexity, a universal model (K = 1) is preferable for lower B values. However, for larger B values, segmenting the system into two or three clusters yields superior outcomes.

#### 6.5 Clustering vs Constant Iteration per Model

Since a larger K induces more models trained with smaller training sets, we also investigated the relationship between K and the number of training epochs.

Figure 5 illustrates the mAP50-95 scores as a function of the number of iterations  $T_1^{K=1}$  in a log-scale for the universal model. Models for clusters with K > 1 adjust their epoch counts, in accordance with Eq. 4, to maintain a consistent number of training iterations (i.e., model updates) across all models, regardless of K. The figure uses distinct markers to represent sample sizes of B = 16, 96 and 256 per stream, and vertical lines to mark the epoch counts when K = 1.

The key observations are:

- 1. Increasing B from 16 to 96, and further to 256, results in substantial improvements outpacing gains from extended epoch counts.
- 2. In comparison with Figure 4, smaller clusters (K = 5, 9) bridged the performance gap with larger-cluster scenarios, particularly as iteration counts increased.
- 3. Performance tends to plateau and even decline after reaching a base epoch (epochs<sub>K=1</sub>) of 80 epochs, suggesting the onset of overfitting beyond a base 100 epochs.

We conclude that careful escalation of the epochs enabled smaller K values to optimize model performance while minimizing the risk of overfitting.

# 7 Discussion

#### 7.1 Insights

The CSBAD framework performance is determined by the number of images per stream (B), active learning strategies (SELECT), number of clusters (K), number of epochs, and dimensions of the *Students* along with the complexity of the *Teacher*.



Figure 5: Mean mAP50-95 scores are presented over log-scaled iterations (T) for each model. Markers denote sample sizes per stream (B = 16, 96, 256). Vertical lines indicate the epochs for K = 1. For K > 1, the epoch count is adjusted to maintain constant iteration counts across models, following Equation 4. Our analysis reveals that an increase in epoch counts benefits smaller cluster configurations (K = 3, 5, 9), enabling them to achieve comparable or superior performance to more universal configurations (K = 1, 2).

Our analysis confirms the expected significance of B. We found that satisfactory results could be achieved with B > 64, with performance gains plateauing beyond  $B \ge 1500$ . The epoch count, while relevant, proved to be of secondary importance (see Fig. 5). Increasing B means more images are used for training, potentially boosting performance but requiring more resources. Naturally, the success of the Top-Confidence strategy highlights the importance of selecting clear, well-lit images with identifiable instances for training in order to minimize the budget.

As for the optimal number of clusters (K), there is no one-size-fits-all answer. The choice depends on the system size (N), available hardware, and budget B constraints. A smaller B or N requires a smaller K to create larger clusters, providing sufficient samples and training iterations for model training. Conversely, a larger N, B, or number of iterations per model T shifts the constraint to hardware capabilities, encouraging model tailoring. Essentially, K acts as an "adjusting variable" within the system to achieve both a specificity-diversity trade-off and sufficient training iterations per model.

Finally, the superior performance of compact fine-tuned *Students* over a large general-purpose *Teacher* demonstrates the distillation's effectiveness, simultaneously reducing complexity and boosting domain-specific performance without the need for a human annotator. Regarding model size, we observe that larger *Teacher* or *Student* models generally yield better outcomes.

#### 7.2 Limitations and perspectives

#### 7.2.1 Scalability

Inherently, large-scale systems exhibit irregularity and unpredictability. Therefore, the applicability of current K results is limited to comparable scales, and determining system behaviors for higher order systems (e.g., N > 100) requires experimentation at much higher scales.

#### 7.2.2 Continuous Deployment

Sustaining a budget b, where  $b \leq B$ , for each  $\mathcal{L}_i$  through successive deployment cycles (i.e., integration of a new device or regular updates), can have advantages. First, this strategy saves resources, as only B - b new images and annotations are required in the next deployment cycle. Second, this approach could mitigate *catastrophic forgetting* French [1999], a scenario where models perform well in new classes but decline in older ones Cheng et al. [2022],

French [1999]<sup>4</sup>. As this require further assumptions on the frequency of fine-tuning and the use case, we considered b = 0 in our experiments. The extension can be done by changing Line 3 in Algorithm 1 to  $\mathcal{L}_i \leftarrow \text{RECYCLE}(\mathcal{L}_i)$ , where RECYCLE manages the storage of  $\{\mathcal{L}_i\}_{i=1,...,N}$  for subsequent deployment.

#### 7.2.3 Non-optimal Resource Management

Our approach necessitates initiating model training from a general-purpose model with each iteration of the framework, leading to suboptimal resource utilization. This process neglects the potential advantages of leveraging previously trained models. For instance, in the CLUSTER approach, although N stream-specific models are initially developed, they are not utilized in subsequent training for cluster-specific models when K < N, resulting in their underutilization.

Future research should investigate **incremental deployment**, with the effective utilization of pretrained models, potentially through methods like weight aggregation Pillutla et al. [2022] and fine-tuning for fewer epochs, to boost learning efficiency and reduce training costs.

# 8 Conclusions

We developed a scalable framework aimed at streamlining model deployment across various video streams. This framework includes collecting video stream samples, utilizing general-purpose models for pseudo-labelling, and performing data clustering to enhance retraining efficiency. When it is applied to object detection in CCTV streams, CSBAD yields significant improvements. The Top-Confidence SELECT function, focusing on high-confidence samples from compact DNN models, significantly increases model accuracy with minimal labeling effort. This method addresses the challenges associated with active distillation, particularly in minimizing pseudo-label inaccuracies. The introduction of K, a variable representing the number of clusters, is crucial for adapting CSBAD to diverse system configurations, emphasizing the importance of balancing specificity, diversity, sample volume, and training epochs to boost the framework's flexibility. In this sense, CSBAD advocates for the clustering of camera domains to move beyond the constraints of specialized or universal networks, enhancing adaptability and performance.

# **Appendix - Clustering Table for WALT**

Thresh.	K	Clust. No.	Cams
1.2	2	1 2	1, 2, 3, 9 4, 5, 6, 7, 8
1.05	3	1 2 3	4, 5, 6, 7, 8 1, 3 2, 9
0.95	4	1 2 3 4	5, 6, 7, 8 1, 3 2, 9 4
< 1	5	1 2 3 4 5	6, 7, 8 4 2, 9 5 1, 3

Table 3: Clusters based on cutting the dendogram for different thresholds.

<sup>&</sup>lt;sup>4</sup>The causes behind catastrophic forgetting include (1) network drift, where the neural network's superior data fitting ability leads it to drift quickly from the feature space learned from the old class training data to that of the new class data and (2) inter-class confusion, where the boundaries between new and old classes are not well established because they have never been trained on together Wang et al. [2023].

# Acknowledgments

This work was partially funded by Win2WAL (#1910045) and Trusted AI Labs. We also thank *Openhub* for providing the equipment. We would also like to express our gratitude to Charlotte Nachtegael for her initial work on the relation between model size and accuracy. We also thanks Stéphane Galland for the discussion.

# References

- Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. Knowledge distillation: A good teacher is patient and consistent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10925–10934, June 2022.
- Dani Manjah, Davide Cacciarelli, Baptiste Standaert, Mohamed Benkedadra, Gauthier Rotsart de Hertaing, Benoît Macq, Stéphane Galland, and Christophe De Vleeschouwer. Stream-based active distillation for scalable model deployment. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pages 4998–5006, June 2023.
- Mark D. Hill. What is scalability? SIGARCH Comput. Archit. News, 18(4):18-21, dec 1990. ISSN 0163-5964.
- Tobias Hossfeld, Poul E. Heegaard, and Wolfgang Kellerer. Comparing the scalability of communication networks and systems. *IEEE Access*, pages 1–1, 2023. doi:10.1109/ACCESS.2023.3314201.
- Armando Fox, Steven D. Gribble, Yatin Chawathe, Eric A. Brewer, and Paul Gauthier. Cluster-based scalable network services. In *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles*, SOSP '97, page 78–91, New York, NY, USA, 1997. Association for Computing Machinery. ISBN 0897919165.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *NIPS 2014 Deep Learning Workshop*, 2015.
- Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, 2018.
- Rahul Mishra and Hari Prabhat Gupta. Designing and training of lightweight neural networks on edge devices using early halting in knowledge distillation. *IEEE Transactions on Mobile Computing*, 2023.
- Ehsan Tanghatari, Mehdi Kamal, Ali Afzali-Kusha, and Massoud Pedram. Federated learning by employing knowledge distillation on edge devices with limited hardware resources. *Neurocomputing*, 531:87–99, 2023.
- Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Semi-supervised learning for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 2019.
- Anthony Cioppa, Adrien Deliege, Maxime Istasse, Christophe De Vleeschouwer, and Marc Van Droogenbroeck. Arthus: Adaptive real-time human segmentation in sports through online distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.
- Ravi Teja Mullapudi, Steven Chen, Keyi Zhang, Deva Ramanan, and Kayvon Fatahalian. Online model distillation for efficient video inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3573–3582, 2019.
- Daniel Rivas, Francesc Guim, Jordà Polo, Pubudu M Silva, Josep Ll Berral, and David Carrera. Towards automatic model specialization for edge video analytics. *Future Generation Computer Systems*, 2022.
- Amirhossein Habibian, Haitam Ben Yahia, Davide Abati, Efstratios Gavves, and Fatih Porikli. Delta distillation for efficient video processing. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 213–229, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19833-5.
- Konstantinos Vilouras, Xiao Liu, Pedro Sanchez, Alison Q O'Neil, and Sotirios A Tsaftaris. Group distributionally robust knowledge distillation. In *International Workshop on Machine Learning in Medical Imaging*, pages 234–242. Springer, 2023.
- Cenk Baykal, Khoa Trinh, Fotis Iliopoulos, Gaurav Menghani, and Erik Vee. Robust active distillation. arXiv preprint arXiv:2210.01213, 2022.
- Micah Goldblum, Liam Fowl, Soheil Feizi, and Tom Goldstein. Adversarially robust distillation. In *Proceedings of the* AAAI Conference on Artificial Intelligence, volume 34, pages 3996–4003, 2020.
- Burr Settles. Active learning literature survey. Computer Sciences Technical Report, University of Wisconsin–Madison, 2009.

- Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR* 2018, 8 2017.
- Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In 2020 International Conference on Learning Representations, 6 2019.
- Ehsan Elhamifar, Guillermo Sapiro, Allen Yang, and S. Shankar Sasrty. A convex optimization framework for active learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 209–216. Institute of Electrical and Electronics Engineers Inc., 2013. ISBN 9781479928392. doi:10.1109/ICCV.2013.33.
- Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. Contextual diversity for active learning. In *European* Conference on Computer Vision (ECCV) 2020, 8 2020.
- Viraj Prabhu, Arjun Chandrasekaran, Kate Saenko, and Judy Hoffman. Active domain adaptation via clustering uncertainty-weighted embeddings. In *International Conference on Computer Vision (ICCV) 2021*, 2020.
- Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 5972–5981, 2019.
- Tianning Yuan, Fang Wan, Mengying Fu, Jianzhuang Liu, Songcen Xu, Xiangyang Ji, and Qixiang Ye. Multiple instance active learning for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 4 2021. URL http://arxiv.org/abs/2104.02324.
- Davide Cacciarelli, Murat Kulahci, and John Sølve Tyssedal. Stream-based active learning with linear models. *Knowledge-Based Systems*, 254:109664, 10 2022a. ISSN 09507051. doi:10.1016/j.knosys.2022.109664.
- Davide Cacciarelli, Murat Kulahci, and John Sølve Tyssedal. Robust online active learning. *Quality and Reliability Engineering International*, ENBIS Special Issue, 2023. doi:https://doi.org/10.1002/qre.3392. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/qre.3392.
- Davide Cacciarelli, Murat Kulahci, and John Sølve Tyssedal. Online active learning for soft sensor development using semi-supervised autoencoders. In *ICML 2022 Workshop on Adaptive Experimental Design and Active Learning in the Real World*, 2022b. doi:10.48550/arXiv.2212.13067. URL https://arxiv.org/abs/2212.13067.
- Jože M. Rožanec, Elena Trajkova, Paulien Dam, Blaž Fortuna, and Dunja Mladenić. Streaming machine learning and online active learning for automated visual inspection. *IFAC-PapersOnLine*, 55(2):277–282, 2022. ISSN 2405-8963. doi:https://doi.org/10.1016/j.ifacol.2022.04.206. URL https://www.sciencedirect.com/science/article/ pii/S2405896322002075. 14th IFAC Workshop on Intelligent Manufacturing Systems IMS 2022.
- Alexander Narr, Rudolph Triebel, and Daniel Cremers. Stream-based active learning for efficient and adaptive classification of 3d objects. In *Proceedings IEEE International Conference on Robotics and Automation*, volume 2016-June, pages 227–233. Institute of Electrical and Electronics Engineers Inc., 6 2016. ISBN 9781467380263. doi:10.1109/ICRA.2016.7487138.
- Aya Aner and John R Kender. Video summaries through mosaic-based shot and scene clustering. In Computer Vision—ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part IV 7, pages 388–402. Springer, 2002.
- Shengyang Sun and Xiaojin Gong. Hierarchical semantic contrast for scene-aware video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22846–22856, 2023.
- Yan Wang, Hengyu Wang, and Xirui Li. An intelligent recommendation system model based on style for virtual home furnishing in three-dimensional scene. In 2013 International Symposium on Computational and Business Intelligence, pages 213–216. IEEE, 2013.
- Zongyi Li, Runsheng Wang, He Li, Bohao Wei, Yuxuan Shi, Hefei Ling, Jiazhong Chen, Boyuan Liu, Zhongyang Li, and Hanqing Zheng. Hierarchical clustering and refinement for generalized multi-camera person tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5519–5528, 2023.
- Gábor Szűcs, Regő Borsodi, and Dávid Papp. Multi-camera trajectory matching based on hierarchical clustering and constraints. *Multimedia Tools and Applications*, pages 1–24, 2023.
- Xiaogang Wang, Kinh Tieu, and W Eric L Grimson. Correspondence-free multi-camera activity analysis and scene modeling. In 2008 IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8. IEEE, 2008.
- Andreas Specker, Daniel Stadler, Lucas Florin, and Jurgen Beyerer. An occlusion-aware multi-target multi-camera tracking system. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4173–4182, 2021.

- Toshal Patel, Alvin Yan Hong Yao, Yu Qiang, Wei Tsang Ooi, and Roger Zimmermann. Multi-camera video scene graphs for surveillance videos indexing and retrieval. In 2021 IEEE International Conference on Image Processing (ICIP), pages 2383–2387. IEEE, 2021.
- Cedric Simon, Jerome Meessen, and Christophe De Vleeschouwer. Visual event recognition using decision trees. *Multimedia Tools and Applications*, 50:95–121, 2010.
- Siyuan Peng, Jingxing Yin, Zhijing Yang, Badong Chen, and Zhiping Lin. Multiview clustering via hypergraph induced semi-supervised symmetric nonnegative matrix factorization. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(10):5510–5524, 2023. doi:10.1109/TCSVT.2023.3258926.
- Dong Huang, Chang-Dong Wang, and Jian-Huang Lai. Fast multi-view clustering via ensembles: Towards scalability, superiority, and simplicity. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11388–11402, 2023. doi:10.1109/TKDE.2023.3236698.
- Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In 2020 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2020.
- Jiyuan Chen, Changxin Gao, Li Sun, and Nong Sang. Ccsd: cross-camera self-distillation for unsupervised person re-identification. *Visual Intelligence*, 1(1):27, 2023.
- Cristian Buciluundefined, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the* 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06, page 535–541, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933395.
- Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
- R. Sibson. SLINK: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1): 30–34, 01 1973. ISSN 0010-4620.
- N. Dinesh Reddy, Robert Tamburo, and Srinivasa G. Narasimhan. Walt: Watch and learn 2d amodal representation from time-lapse imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), pages 9356–9366, June 2022.
- Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.
- Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLO, January 2023. URL https://github.com/ ultralytics/ultralytics.
- Robert M French. Catastrophic forgetting in connectionist networks. Trends in cognitive sciences, 3(4):128–135, 1999.
- Meng Cheng, Hanli Wang, and Yu Long. Meta-learning-based incremental few-shot object detection. *IEEE Transactions* on Circuits and Systems for Video Technology, 32(4):2158–2169, 2022. doi:10.1109/TCSVT.2021.3088545.
- Shaokun Wang, Weiwei Shi, Songlin Dong, Xinyuan Gao, Xiang Song, and Yihong Gong. Semantic knowledge guided class-incremental learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(10):5921–5931, 2023. doi:10.1109/TCSVT.2023.3262739.
- Krishna Pillutla, Sham M. Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *IEEE Transactions* on Signal Processing, 70:1142–1154, 2022. doi:10.1109/TSP.2022.3153135.