# CCTVCV: Computer Vision model/dataset supporting CCTV forensics and privacy applications

Hannu Turtiainen, Andrei Costin, Timo Hämäläinen, Tuomo Lahtinen, Lauri Sintonen
*Faculty of Information Technology*
*University of Jyväskylä*
Jyväskylä, Finland
{turthzu,ancostin,timoh,tutalaht,lamijosi}@jyu.fi

***Abstract*—The increased, widespread, unwarranted, and unaccountable use of Closed-Circuit TeleVision (CCTV) cameras globally has raised concerns about privacy risks for the last several decades. Recent technological advances implemented in CCTV cameras, such as Artificial Intelligence (AI)-based facial recognition and Internet of Things (IoT) connectivity, fuel further concerns among privacy advocates. Machine learning and computer vision automated solutions may prove necessary and efficient to assist CCTV forensics of various types.**

**In this paper, we introduce and release the first and only computer vision models are compatible with Microsoft common object in context (MS COCO) and capable of accurately detecting CCTV and video surveillance cameras in street view, generic images, and video frames.**

**Our best detectors were built using 8,387 images, which were manually reviewed and annotated to contain 10,419 CCTV camera instances, and achieved an accuracy rate of up to 98.7%. This work proves fundamental to a handful of present and future applications that we discuss, such as CCTV forensics, pro-active detection of CCTV cameras, providing CCTV-aware routing, navigation, and geolocation services, and estimating their prevalence and density globally and on geographic boundaries.**

***Index Terms*—CCTV, cameras, computer vision, datasets, machine learning, mapping, object detection, privacy, video surveillance**

## I. Introduction

CCTV and video surveillance cameras are currently some of the most ubiquitous technology, and it is almost impossible to live through one day without entering the field of view of at least one, if not dozens, of closed-circuit television (CCTV) cameras [1], [2]. At present, CCTV cameras are an integral part of any infrastructure (e.g., cities, buildings, streets, and businesses), and it is expected that more than one billion closed-circuit television (CCTV) cameras will be operating globally by the end of 2021 [3]. In terms of cybersecurity, CCTV cameras, digital video recorders (DVRs), and video surveillance systems have already been the target of numerous cyberattacks [4], and they were the main culprit behind the legendary and massive attack by the Mirai Internet of Things (IoT) botnet [5]. Both the CCTV forensics challenges [6], and the privacy risks [7]–[11], associated with CCTVs are also well known. There are several ways to overcome CCTV forensics challenges and mitigate the privacy risks of CCTV cameras, including their additional features, such as face recognition. One way is to develop, provide and use appropriate *high-tech tools* both for *CCTV forensics applications* [6] and against the invasion of privacy by CCTV cameras [12]. Examples of such tools include CCTV-aware route planning and navigation as well as real-time early warning systems that alert users when their video input-equipped mobile and embedded devices (e.g., wearables, smartphones, and drones) enter the potential field of view of CCTV cameras. Some recently presented tools can perform city-wide CCTV-aware privacy-oriented route planning [12]. However, the authors primarily relied on manual camera mapping, which is not a scalable approach given the total number of cameras worldwide. Such tools require accurate object detection and mapping, and localization and computer vision are proven methods for performing these tasks [13]–[15]. However, various foundational blocks are currently unavailable to implement such CCTV-aware technology. We argue that one of the most crucial missing components is an object detector for quick, accurate, and automated computer vision detection of CCTV cameras.

*Contributions.:* In this paper, we develop a novel dataset with high practical applicability [12], [16], [17]. From a scientific method point of view, we use state-of-the-art CV methods and toolsets to build, train and evaluate our models. We also attempt to close the existing fundamental research and technology gaps, as well as, to address the strong and imminent need for such tools. During the experiments on real-world data, our system achieved an accuracy rate of up to 98.7%, which is comparable to Google's original automatic system for the large-scale privacy protection of human faces and car license plates in Google Street View [18].

1) To the best of our knowledge, since 2020, we have been the first to research, implement, and evaluate Computer Vision (CV) models to detect *CCTV cameras* [1] In street view, images and video frames, with a *emphasis on privacy, anonymity, and anti-surveillance applications*.
2) In the context of CCTV forensics and Privacy Enhancing Technologies (PET), we identify and discuss a handful of core challenges and applicative scenarios associated with this research direction
3) We release, as *open data*, the models and datasets necessary to validate our results as well as to further expand the

[1]At this stage, our core aim is the accurate, large-scale detection *visible* CCTV cameras. Although our object detectors work well on certain edge-cases (Figs. 4, 5), the camouflage attacks on object detectors (e.g., careful placement, and decoration) is a separate emerging topic [19], [20].

datasets and the research field. To our knowledge, these are the first, the largest, and the best-performing datasets and models publicly released for solving the stated problems. The relevant artefacts (e.g., code, datasets, trained models, and documentation) will be available at the acceptance stage at: *https://github.com/Fuziih*.

## II. Related works

Lin et al. [21] proposed a novel dataset for general object detection called Microsoft's Common Objects in COntext (COCO, or MS COCO). The most recent (2017) update of MS COCO has a fully annotated training dataset containing 118000 images and a 5000 image validation dataset. In addition, a 41000 image testing dataset is also available. There are more than 80 different classes for state-of-the-art object detector testing with the MS COCO [22], including pedestrians, traffic lights, cars, and even teddy bears. However, it does not contain object detection models for CCTV cameras, though they are currently an indispensable part of any street and city infrastructure, such as traffic lights and road signs.

The Mappilary Vistas work [23] presented a novel large-scale dataset built from about 25,000 street-level images. The images in the dataset were annotated into 66 object categories, while 37 object classes also have additional instance-specific annotations. However, this dataset contained less than 20 image instances of CCTV cameras, which is not enough to build a reliable and representative CV model out of it. In comparison, our CCTVCV dataset offers manually reviewed and annotated 10,419 CCTV camera instances.

Recently, Sheng et al. [24] attempted to estimate the prevalence of surveillance cameras in 16 major cities. For this, they built their CV models, which provided an accuracy rate of 0.93, which is almost 6% less accurate than our CCTVCV model we developed one year before their work.

In a similar yet distinct direction, Buzzon [25] collected images of CCTV surveillance warning signs. The author did not build a CV model out of it; its main scope was an artistic-perspective exploration of surveillance in modern society.

The ResNeSt split-attention networks was introduced in April 2019 by H. Zhang et al. [26]. It features a novel building block to improve the standard ResNet backbone structure, originally developed for image classification rather than object detection. The authors introduced the modular split-attention block to enhance the performance of the ResNet architecture and gear it toward more downstream object detection and segmentation tasks. Their ResNeSt network consists of multiple split-attention blocks stacked on the ResNet structure. Each split-attention block generally divides the feature maps into subgroups. The feature representation of the group is derived from a weighted combination of even smaller feature divides called splits. Zhang et al. [26] showcased their state-of-the-art results using the ResNeSt backbone on cascade region-based convolutional neural networks (R-CNN) model [27].

In June 2020, Qiao et al. [28] published a paper on DetectoRS, which features two distinct mechanisms to improve object detection tasks. They propose inserting a Recursive Feature Pyramid (RFP) on top of regular Feature Pyramid Networks (FPN) [29] structure. The RFP features an increased number of feedback connections from the FPN [29] to the bottom-up backbone layers, which is a "looking and thinking twice "design. The authors explained that this approach creates more powerful feature representations [28]. Another distinct feature of the DetectoRS is switchable atrous convolution (SAC). It enables the backbone to adapt to different scales of objects conditionally. The gist of SAC is that it does not require previous retraining weights, as it can effectively adapt standard convolutions to conditional convolutions [28]. Using DetectoRS with the ResNet-50 and ResNeXt-101-32x4d backbones achieve state-of-the-art results in Microsoft Common Objects in COntext (MS COCO) test-dev tasks [28].

## III. Applications

We believe our present work, CCTVCV, can enable more future CCTV forensics applications and address some of its imminent challenges [6]. For example, OSRM-CCTV [30] and CCTV-Exposure [30] systems, building over CCTVCV can be used to analyze both future predictions and past retrospects, if a particular navigation route went nearby any CCTV cameras. This can help, in turn, investigate criminal, humanitarian, and insurance cases, e.g., by quickly identifying the subset of CCTV cameras that would be most interesting to capture evidence for the case. Overall, an ML/CV-based system (such as CCTVCV) that can automatically detect and mark CCTV cameras in street view imagery and video footage, will be required, or at least be highly beneficial, in any CCTV forensics applications, given that the number of CCTV cameras is growing fast beyond 1 billion and manual efforts are unlikely to scale with that.

## IV. Methodology and experimental setup

As with any CV object detector, we followed a two-phase approach. First, we trained multiple models for object detection using training sets and additional validation sets for internal self-validation during model training. To train the CV object detectors, we split the phase into four parts: dataset gathering (Section IV-A), image annotation (Section IV-C), and model training (Section IV-E). Once the model training is completed, we evaluate each trained model against a testing set.

### A. Dataset compiling and definitions

When we began compiling the dataset, we made a practical decision to classify cameras into at least two distinct subclasses based on their shape: *directed cameras* and *round cameras*. This information enables us to model the cameras' 3D field of view coverage more accurately in future work. The algorithm can decide whether a particular point in space (e.g., sidewalk, street, or street corner) provides adequate privacy for an individual. However, it is essential to note that, from the viewpoint of MS COCO annotations [21] and CV/ML training, we treat all the CCTV cameras as a single *camera* category, regardless of whether they are annotated as directed or round.

This is in line with the MS COCO [21] approach, where, for example, the *cars* category is represented as a single category regardless of the actual car's properties (e.g., shape, make, or color). In Table I, we compare our dataset to MS COCO per class basis.

*Directed cameras* include box- and bullet-shaped cameras, and we assume such cameras record a limited field of view in the direction they are pointed. Some of them are motorized (e.g., via pan-tilt-zoom [PTZ] hardware and protocols) and therefore have a mobile field of view (in theory, up to 360°). However, it is challenging, if not impossible, to detect PTZ elements with computer vision on static and low-resolution images.

*Round cameras* include dome- and sphere-shaped cameras, and we assume that such cameras potentially record a 360° field of view. Although some dome- and sphere-shaped cameras have an internal static and directed camera sensor, it is usually difficult to determine because of the reflective glass. Therefore, to simplify our experiments and future geomapping modeling, we assume for now that such cameras record a 360° field of view.

### B. Datasets overview

*a) DatasetAll.:* Our latest and most complete dataset is `DatasetAll`, which is the core of this paper. It was obtained by merging `Dataset0`, `Dataset1`, and `Dataset2` and applying de-duplication, cleanup, and quality checks after the merging process. It is important to note that due to the complex nature of the merging process, the counts in `DatasetAll` are not just a plain sum-up of all the individual counts from `Dataset0`, `Dataset1`, and `Dataset2`. After the merging process was completed, the `DatasetAll` training set contained $8,387$ images with $10,419$ camera instances ($6,137$ directed and $4,282$ round), while its validation set had $533$ images with $647$ camera instances ($379$ directed and $268$ round). We used `DatasetAll` to train our most recent models, namely ResNeSt-200 and DetectoRS: Cascade + ResNet-50.

*b) Dataset2.:* `Dataset2` was collected in late September 2020 using a crowd-sourcing effort by eight contributors. We used our own annotation tool (see Section IV-C). This dataset contains $4,167$ images with $5,380$ camera instances ($3,325$ directed and $2,055$ round). The images were captured mostly from Google Street View [31], but Flickr [32] and Baidu Maps [33] were also used. This dataset was never used *as is* for model training and evaluation. However, all of the images captured in this dataset were used to create the training, validation and testing sets of our most complete dataset, `DatasetAll`.

*c) Dataset0, Dataset1.:* Our first dataset, which we call `Dataset0`, was originally collected in January-February 2020 by contributor *Person0 HT*. The images were collected from Flickr [32], 500px [34], Unsplash [35] and Google Maps Street View [31]. All the images in this dataset were annotated using Wada's Labelme standalone annotation tool [36] by contributor *Person0 HT*. The `Dataset0` training set contained $3,401$ images with $3,986$ instances ($2,244$ directed and $1,742$ round) and the validation set had $528$ images with $617$ instances ($348$ directed and $269$ round) respectively. The `Dataset0` also featured a small testing set of $186$ images with $231$ instances ($126$ directed and $105$ round) [2].

As our most immediate goal was to detect cameras from street-level imagery, our next effort to improve the dataset was to focus on street-view imagery in the dataset's content. Therefore, `Dataset1` only included street view-images from `Dataset0` and some new images contributor *Person6 TL* captured during May 2020 from Google Street View [31]. The `Dataset1` training set contained $2,457$ images with $3,101$ instances ($1,554$ directed and $1,547$ round), and the validation set had $270$ images containing $354$ instances ($178$ directed and $176$ round). The testing set was the same as with `Dataset0`.

### C. Image annotation

Since we used supervised training at this stage, we had to label and annotate the images in our datasets.

*a) Dataset0, Dataset1.:* To annotate the images in these datasets, we used the standalone Wada's Labelme tool [36]. Wada's Labelme [36] allows annotation with polygon segments that can be used in object segmentation architectures, such as Mask-RCNN [37] and CenterMask [38]. When using object segmentation, the outlines of the objects can be identified more precisely, instead of a mere bounding box placed around the object of interest.

*b) Dataset2.:* To enable rapid and straightforward crowd-sourced contribution, especially for completing the work in this article, we designed and developed a novel annotation tool implemented as a highly flexible and MS COCO-compatible *browser-only extension* [16]. It requires minimal setup and configuration effort and is ready to use out of the box, even by novice contributors. The browser extension was written in JavaScript and came with a set of more than ten distinctive features.

### D. Datasets details

*1) Training and validation datasets:* As for the dataset, we settled on a training dataset with 3,401 images containing 2,244 directed class instances and 1,742 round class instances. The validation dataset contained 528 images with 348 directed class instances and 269 round class instances marking about 15.5% split between training and validation datasets. MS COCO format dataset made it easy to use several architectures out of the box and allowed instance segmentation detection due to polygon annotation possibility.

Our training dataset contains $3,401$ images for a total of $2,244$ directed class instances and $1,742$ round class instances. Our validation dataset contains $528$ images for a total of $348$ directed class instances and $269$ round class instances, marking about $15.5\%$ split between training and validation datasets.

For the datasets, we used the MS COCO format as this made it easy to use several object detection architectures out

[2]Later on, this testing set was replaced with improved alternatives.

of the box (see Section IV-E), and it also allowed instance segmentation detection thanks to the polygon annotation possibility.

*2) Testing dataset:* For evaluation purposes (see Section V), we gathered a separate test dataset. The test dataset contained only street-level imagery sources such as OpenStreetCam [39] and Google Street View [31]. The test dataset is intended to serve as an indicator of how well our trained models will work as CCTV camera detectors specifically for street-level imagery sources since such sources are the primary use case proposed for future work. Our test dataset contains 186 images for a total of 126 instances of *directed cameras* class and 105 instances of the *round cameras* class. In Table II, we disclose the statistics of our dataset iterations in detail.

TABLE I: Comparison: our datasets vs. MS COCO 2017.

| Dataset Name | Total Categories | Total Instances | Median Instances per categ. | Increase (vs. Median MS COCO) |
|---|---|---|---|---|
| MS COCO (train) | 80 | 860001 | 6097 | 1x |
| `DatasetAll` (train) | 1 | 10,419 | **10,419** | **1.70x** |
| `Dataset0` (train) | 1 | 3,986 | 3,986 | 0.65x |
| MS COCO (val) | 80 | 36781 | 265 | 1x |
| `DatasetAll` (val) | 1 | 647 | **647** | **2.44x** |
| `Dataset0` (val) | 1 | 617 | 617 | 2.32x |

TABLE II: High-level statistics for the datasets (training set).

| | `Dataset0` | `Dataset1` | `Dataset2` | `DatasetAll` |
|---|---|---|---|---|
| Total counts | | | | |
| Total collected images | 3,401 | 2,457 | 4,167 | 8,387 |
| Total annotated camera instances | 3,986 | 3,101 | 5,380 | 10,419 |
| Images grouped by source | | | | |
| Google (Street View, Images Search) | 1,906 | 2,457 | 3873 | 6,598 |
| Baidu street view | - | - | 269 | 269 |
| Flickr | 935 | - | 25 | 960 |
| 500px | 482 | - | - | 482 |
| Unsplash | 78 | - | - | 78 |
| Instances grouped by sub-type | | | | |
| Directed camera instances | 2,244 | 1,554 | 3,325 | 6,137 |
| Round camera instances | 1,742 | 1,547 | 2,055 | 4,282 |
| Instances grouped by pixel area | | | | |
| Small (¡ 32x32 px) | 685 | 762 | 1455 | 2,331 |
| Medium (32x32 – 96x96 px) | 2,247 | 2,147 | 3345 | 6,397 |
| Large (¿ 96x96 px) | 1,054 | 193 | 580 | 1,691 |

## E. Model training

To date, it has taken the equivalent of at least *three person-months of effort* to perform all the model training experiments with trial and error. It also took the equivalent of at least *600 GPU hours* of computing effort using the described hardware configurations.

*1) Initial models:* Our first efforts with `Dataset0` included three detectors with with total of six models. We trained CenterMask2 [38], [40] with VoVNet-V2 [41] as the backbone using the *V-57-eSE*, *V-99-eSE*, *V-39-eSe (lightweight)* variants. For ATSS [42], we used ResNet-50 [43] and ResNeXt-101 [44] backbones with multi-scale training and deformable convolutions and for TridentNet [45], we used ResNet-101 [43] C4 backbone. As previously mentioned, we achieved excellent results with `Dataset0`, with 95.6% and 91.1% accuracy rates for the validation and testing sets, respectively. Although the results were satisfactory, as our model use-cases began to realize and given the peer-reviewed received, we decided to create a larger dataset with more variance in the images. Object detection is evolving quickly, so we substituted our detectors for more promising ones.

*2) Latest models:* Therefore, for `DatasetAll` we used two state-of-the-art object detection frameworks to train our models, FAIRs detectron2 [46] and MMdetection [47] by Multimedia Laboratory, CUHK. Our ResNeSt [26] model used detectron2 [48], and the DetectoRS [28] model was trained on MMDet [47]. Both of the frameworks are equipped to handle our COCO-style dataset by default and feature dataset evaluators using the pycocotools library.

*a) ResNeSt.:* For ResNeSt [26], we chose a huge 200-layer-deep backbone with the Cascade R-CNN [27] method. Standard settings were primarily used therefore, the models feature FPN [29], SyncBN [49] and image scale variation that randomizes the short side of the input image between 640 and 800 pixels. We changed the class number to two and used the same training schedule as our previous experiments. We first set the maximum iterations to 45,000 (i.e., 0.5x of the 1x learning rate schedule, which translates to 90,000 iterations by default). However, we achieved even better results with 20,000–40,000 iterations. Therefore, we set our schedule to 36,000 maximum iterations, lowering steps to 30,000 and 34,000. Our base learning rate was 0.02.

*b) DetectoRS.:* For DetectoRS [28], we chose the ResNet-50-based backbone. The images were input at 800 pixels short-size, and the learning rate was set at 0.01. We set the number of classes to two, and the scheduler remained at 12 epochs. Our dataset and the setting at two samples per worker resulted in 4,195 iterations per epoch, totaling 50,340 iterations. The learning rate automatically stepped to one-tenth of the previous at the beginning of epochs nine and twelve.

Table III show the training iteration count for each model that provided the best performance for the whole training session. Additionally, these tables present the timings for training and inference under each detection model. For detailed structure and workflow of the standard models, we reference the interested readers to the corresponding papers [26], [28].

# V. Analysis of Results

We tested all the models separately with the validation dataset and a separate testing dataset (see Section IV-A). As mentioned earlier, the testing dataset mainly features images from street-level maps since this corresponds to the intended use for the detector model.

TABLE III: Final configuration, iterations count, training and inference times with training detectors on `DatasetAll`.

| Detector | Best-result iterations (count [set]) | Weights file size (MB) | Avg. train time / iter. (seconds [batch size]) | Avg. inference time / 800px image (seconds) |
|---|---|---|---|---|
| ResNeSt | 24,999 [test] / 19,999 [val] | 1,009 | 1.6 [batch 8] | 0.171 |
| DetectoRS | 37,750 (epoch 9) [test, val] | 989 | 0.66 [batch 2] | 0.13 |

### A. Evaluation metrics

To evaluate our models, we used *pycocotools* and the MS COCO evaluator built into the frameworks. For all the models trained, we used the same evaluator. The metrics we used and will present in this paper are modified from the standard MS COCO [21] evaluation metrics to suit our goals. We offer the modifications and the arguments for them. We also present an F1 score for our results, a metric derived from the average precision and recall. For the F1-score, we measure with 0.5 IoU and 100 detections per image threshold. We also list the average precision rate per category (directed and round).

To evaluate the detector for the *detection performance*, MS COCO employs 12 characterizing metrics. With MS COCO, *average precision (AP)* essentially represents the *mean average precision (mAP)*, which is the average precision rate across all classes with *localization accuracy* built in. MS COCO's standard measurements are primarily *average precision (AP)* and *average recall (AR)*, where the average is taken on 10 intersection-over-union (IoU) threshold from 0.5 to 0.95 with a 0.05 interval. The AP across scales includes the area of pixels within the segmentation mask or the bounding box. In this context, based on the pixel size of the detected segmentation mask or the bounding box, small objects are up to `32x32` pixels, medium objects are between `32x32`, and large objects are greater than `96x96` pixels in size.

We argue that the small objects category is not representative or required for our model use cases (i.e., detection in street-view images). We discuss that considering the varying quality of street-view images, the resolution we want to run the images at and with computer-collected images, objects under the `32x32` pixel threshold are subject to unnecessary false detections. Therefore, we omit the small category from our results and further use it with our models. We prefer to create a more intelligent image-capturing algorithm for better coverage of the captured streets. Our datasets still contain instances of the small category, their detection can be enabled or disabled at will.

With our main and immediate use-cases, pixel-precise localization is unnecessary; hence, the average precision with 0.5 IoU (i.e., AP@0.5 metric) is most relevant in our case. Therefore, we scaled all of our results to the 0.5 IoU threshold.

### B. Numerical results

In Table IV, we present the metrics (bounding box detection) for the testing dataset with 800-pixel short-side images. The results are outstanding. Cutting the tiny samples from the detections increased our percentages substantially. Our F1 scores with both models are quite high, suggesting the lack of false detections in the larger detection categories. Directed samples seem to be more difficult to detect in our testing dataset. We are also pleased with our localization accuracy, as the difficult 0.5:0.95 IoU category is in the 70th-percentile range. Large samples are no issue for our models, and the large sample size seems adequate.

In Table V, we show the detection metrics (bounding box detection) for the validation dataset. The results are excellent. The scores are high for all categories. Cross-referencing the testing set and validation set scores, we could argue a slight overfitting issue exists, especially with the DetectoRS model. In comparison, the directed subcategory also takes a hit in detection precision. Therefore, we will argue that our validation dataset needs another revision to improve detection generalization. We are pleased with the results, including the tougher 0.5:0.95 IoU localization tests.

In Table VI, we present the segmentation detection results for the ResNeSt-model. In comparison, we can see that the segmentation results are not far off from the bounding box results. In a few categories, the results are even slightly better. This suggests great localization precision and a great beginning for the more accurate detection of camera types, as the model can detect shapes more accurately.

Input image resolution is a debatable topic. Should we input images in their native resolution or training image size? The ideal case would be if there were no discrepancies between the two. As our testing dataset contains slightly higher resolution images than our testing set, we also tested the testing set with 1,200-pixel short-side input images. We present the results in Table VII.

The results show that our DetectoRS-model benefits from the increased input resolution and achieves the highest scores with the testing set. However, our ResNeSt-model suffers from increased input resolution and lower scores than smaller images. ResNeSt benefits from images that are close to the training resolution. The results imply that testing the models with varying resolutions is beneficial, and keeping the resolution constant will yield stable results. As transfer-training of these models is not excessively time-consuming, creating specialized models for each task with different image properties is most likely beneficial.

### C. Visual result samples

In this section, to understand our detectors' successes, failures, and challenges, we present a selection of relevant samples and some comments.

Figs. 1, 2, 3 are perfect examples of our excellent results – all TPs are found, and nothing else is detected. Furthermore, the confidence levels on these samples are high (i.e., the lowest is 80%).

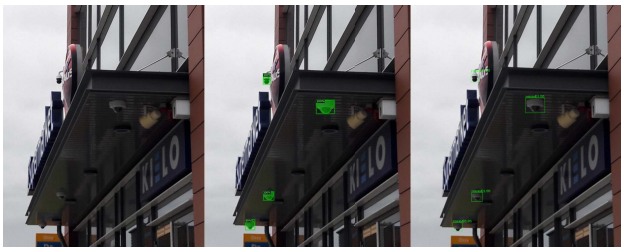

Fig. 1: Visual results (Ground Truth - 4 TP) (left to right): ResNeSt - 4 TP (3x100% and 80%); DetectoRS - 4 TP (3x100% and 95%)

Figs. 4 and 5 are examples of how DetectoRS edges out on some of the samples. In Fig. 4, the camera blends into

TABLE IV: Results for bounding box detection with the `DatasetAll` *testing set*, 800px short side images, **bold**=best

| Detector | AP@0.5 | AP@0.5:0.95 | APm | APl | AR 100 | ARm | ARl | F1 | AP@0.5 (*directed* type) | AP@0.5 (*round* type) |
|---|---|---|---|---|---|---|---|---|---|---|
| ResNeSt | **92.0%** | **71.4%** | **91.5%** | **96.0%** | **94.5%** | **93.7%** | **100%** | **93.2%** | **89.2%** | **94.8%** |
| DetectoRS | 91.5% | 68.9% | 91.1% | 93.9% | 93.1% | 92.6% | 95.5% | 92.3% | 89.1% | 93.9% |

TABLE V: Results for bounding box detection with the `DatasetAll` *validation set*, 800px short side images, **bold**=best

| Detector | AP@0.5 | AP@0.5:0.95 | APm | APl | AR 100 | ARm | ARl | F1 | AP@0.5 (*directed* type) | AP@0.5 (*round* type) |
|---|---|---|---|---|---|---|---|---|---|---|
| ResNeSt | 97.3% | 80.1% | 96.8% | 98.7% | 98.4% | 98.0% | **99.6%** | 97.8% | 98.0% | 96.5% |
| DetectoRS | **98.7%** | **83.4%** | **99.2%** | **98.8%** | **99.5%** | **99.7%** | 99.1% | **98.5%** | **98.6%** | **98.7%** |

TABLE VI: Results for ResNeSt *segmentation* detection with the `DatasetAll` *testing and validation sets*, 800px short-side images

| Set | AP@0.5 | AP@0.5:0.95 | APm | APl | AR 100 | ARm | ARl | F1 | AP@0.5 (*directed* type) | AP@0.5 (*round* type) |
|---|---|---|---|---|---|---|---|---|---|---|
| Test | 91.5% | 70.2% | 90.9% | 96.2% | 94.0% | 93.2% | 100% | 92.7% | 89.3% | 93.8% |
| Val | 97.0% | 80.8% | 96.5% | 98.8% | 98.1% | 97.5% | 99.6% | 97.5% | 97.3% | 96.8% |

TABLE VII: Results for bounding box detection with the `DatasetAll` *testing set*, 1200px short-side images, **bold**=best

| Detector | AP@0.5 | AP@0.5:0.95 | APm | APl | AR 100 | ARm | ARl | F1 | AP@0.5 (*directed* type) | AP@0.5 (*round* type) |
|---|---|---|---|---|---|---|---|---|---|---|
| ResNeSt | 90.1% | 71.9% | 90.8% | 87.0% | 94.0% | 93.7% | **95.5%** | 92.0% | 88.3% | 92.0% |
| DetectoRS | **92.6%** | **72.8%** | **92.4%** | **93.3%** | **95.4%** | **95.3%** | **95.5%** | **94.0%** | **90.0%** | **95.2%** |

Fig. 2: Visual results (Ground Truth - 4 TP) (left to right): ResNeSt - 4 TP (4x100%); DetectoRS - 4 TP (2x100%, 99% and 96%)

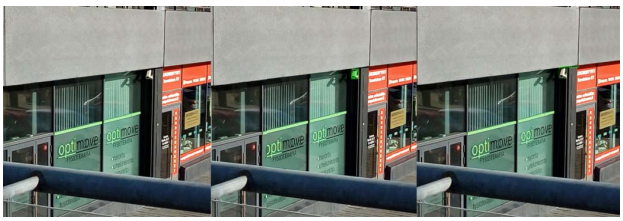

Fig. 3: Visual results (Ground Truth - 1 TP) (left to right): ResNeSt - 1 TP 99%; DetectoRS - 1 TP 100%

Fig. 4: Visual results (Ground Truth - 1 TP) (left to right): ResNeSt - 1 FN; DetectoRS - 1 TP 95%

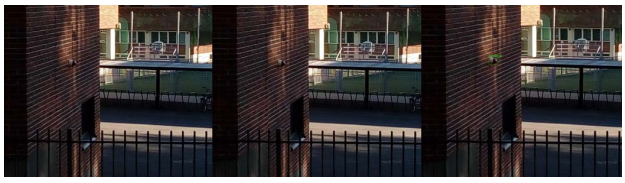

Fig. 5: Visual results (Ground Truth - 1 TP) (left to right): ResNeSt - 1 FN; DetectoRS - 1 TP 81%

the white wall in the sunlight, and Fig. 5 is quite dark. The ResNeSt-model cannot detect the cameras, but DetectoRS finds them and achieves good confidence in the TPs.

Fig. 6 is the worst sample on our testing set. ResNeSt finds two FPs on the two lamps, but it also finds the camera. DetectoRS only finds a single FP. Confidence levels on the FPs are worryingly high.

Fig. 7 showcases an FN on both models. The directed camera in the upper corner of the image is undetected. However, the lower round-type camera is found in high confidence (99%).

Figs. 8 and 9 showcases how the angle of the image can affect the results. Regardless of the angle, both models find the TPs; however, at another angle, both models find a single FP. With ResNeSt, the confidence of the FP is quite high (93%), compared to 74% in the DetectoRS case, which could be rooted out with confidence limiting. We could also use

sensor fusion here to limit the possibility of FPs.

Fig. 6: Visual results (Ground Truth - 1 TP) (left to right): ResNeSt - 1 TP 88%, 2 FP (99% and 96%); DetectoRS - 1 FN, 1 FP 95%

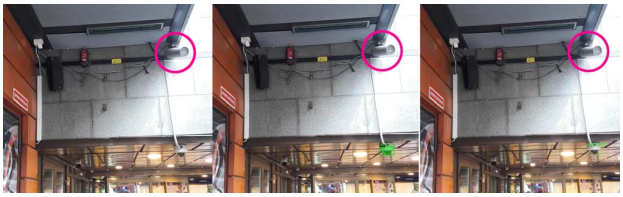

Fig. 7: Visual results (Ground Truth - 2 TP) (left to right): ResNeSt - 1 TP 99%, 1 FN; DetectoRS - 1 TP 99%, 1 FN

Fig. 8: Visual results (Ground Truth - 2 TP) (left to right): ResNeSt - 2 TP (2x100%); DetectoRS - 2 TP (2x100%), 1 FP 74%

Fig. 9: Visual results (Ground Truth - 2 TP) (left to right): ResNeSt - 2 TP (2x100%), 1 FP 93%; DetectoRS - 2 TP (100% and 96%)

In Fig. 10, both models find the TPs easily with high confidence. However, both models also find a single FP with the lamp. The confidence of these FPs is still lower than usual (78% and 73%).

Next, we present some examples and results where we applied image alteration techniques to improve confidence levels and achieve TP/TN/FP values as close as possible to the Ground Truth (GT). Figs. 11 and 12 showcase the result of image alterations with the ResNeSt model. In Fig. 11, we find improvements as an FN is turned into a TP in exposure and equalizer tests. In Fig. 12, the results are mixed. On the exposure test, we remove a single FP, but the confidence level of the TP camera is hindered, and a single FP is still present.

Fig. 10: Visual results (Ground Truth - 2 TP) (left to right): ResNeSt - 2 TP (2x100%), 1 FP 78%; DetectoRS - 2 TP (100% and 99%), 1 FP 73%

Fig. 11: Visual results (Ground Truth - 1 TP) (left to right): Original - 1 FN; Contrast - 1 FN; Equalizer - 1 TP 73%; Exposure - 1 TP 79%; Saturation - 1 FN

Fig. 12: Visual results (Ground Truth - 1 TP) (left to right): Original - 1 TP 88%, 2 FP (99% and 96%); Contrast - 1 TP 97%, 2 FP (99% and 96%); Equalizer - 1 TP 93%, 2 FP (99% and 80%); Exposure - 1 TP 65%, 1 FP 99%; Saturation - 1 TP 52%, 2 FP (96% and 92%)

## VI. Conclusion

In this paper, we presented the first computer vision-based object detectors to accurately identify CCTV cameras in images and video frames. To build our system, we used and evaluated several state-of-the-art computer vision frameworks and backbones in parallel. Our best detectors were built using 8,387 images that were manually reviewed and annotated to contain 10,419 CCTV camera instances, and achieve an accuracy of up to 98.7%. We believe our present work, CCTVCV, can enable more future CCTV forensics applications and address some of its imminent challenges [6]. For example, OSRM-CCTV [30] and CCTV-Exposure [30] systems, building over CCTVCV can be used to analyze both future predictions and past retrospects, if a particular navigation route went nearby any CCTV cameras. This can help, in turn, investigate criminal, humanitarian, and insurance cases, e.g., by quickly identifying the subset of CCTV cameras that would be most interesting to capture evidence for the case. Overall, an ML/CV-based system (such as CCTVCV) that can automatically detect and mark CCTV cameras in street view, imagery and video footage will be required, or at least be highly beneficial, in any CCTV forensics applications, given the number of CCTV cameras, are growing fast beyond 1 billion and manual efforts are unlikely to scale with that.

## Acknowledgments

The authors acknowledge the grants of computer capacity from the Finnish Grid and Cloud Infrastructure (persistent identifier *urn:nbn:fi:research-infras-2016072533*).

Part of this research was supported by a grant from the *Decision of the Research Dean on research funding within the Faculty (17.06.2020)*, *Decision of the Research Dean on research funding within the Faculty (07.04.2021)*, and *Decision of the Research Dean on research funding within the Faculty (20.04.2022)* of the Faculty of Information Technology of University of Jyväskylä (The authors thank Dr. Andrei Costin for facilitating and managing the grant).

Hannu Turtiainen thanks the Finnish Cultural Foundation / Suomen Kulttuurirahasto (https://skr.fi/en) for supporting his Ph.D. dissertation work and research (under grant decision no.00221059) and the Faculty of Information Technology of the University of Jyvaskyla (JYU), in particular, Prof. Timo Hämäläinen, for partly supporting and supervising his Ph.D. work at JYU in 2021–2023.

The authors would also like to thank the following persons for their dedicated efforts during the crowdsource data collection phase: Arttu Takala, Syed Khandker, Pyry Kotilainen, Janne Uusitupa, Anna Arikainen.

Last but not least, the authors also thank Chairs of 2021 AAAI/ACM Conference on Artificial Intelligence, Ethics and Society (AIES) for their support and assistance solving a research ethics and integrity case related to our CCTVCV work, as detailed in [24].

## References

[1] J. Pasley, "I documented every surveillance camera on my way to work in New York City, and it revealed a dystopian reality," https://businessinsider.com/how-many-security-cameras-in-new-york-city-2019-12, 2019. 1

[2] D. Barrett, "One surveillance camera for every 11 people in Britain, says CCTV survey," https://telegraph.co.uk/technology/10172298/One-surveillance-camera-for-every-11-people-in-Britain-says-CCTV-survey.html, 2013. 1

[3] E. Cosgrove, "One billion surveillance cameras will be watching around the world in 2021," https://cnbc.com/2019/12/06/one-billion-surveillance-cameras-will-be-watching-globally-in-2021.html. 1

[4] A. Costin, "Security of CCTV and video surveillance systems: Threats, vulnerabilities, attacks, and mitigations," in *6th International Workshop on Trustworthy Embedded Devices (TrustED)*, 2016. 1

[5] M. Antonakakis et al., "Understanding the mirai botnet," in *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, 2017. 1

[6] R. Gomm, R. Brooks, K.-K. R. Choo, N.-A. Le-Khac, and K. W. Hew, "CCTV Forensics in the Big Data Era: Challenges and Approaches," *Cyber and Digital Forensic Investigations*, pp. 109–139, 2020. 1, 2, 7

[7] Electronic Frontier Foundation, "Street-Level Surveillance – Surveillance Cameras," https://eff.org/pages/surveillance-cameras. 1

[8] C. Slobogin, "Public privacy: camera surveillance of public places and the right to anonymity," *Miss. lJ*, vol. 72, p. 213, 2002. 1

[9] B. v. S.-T. Larsen, *Setting the watch: privacy and the ethics of CCTV surveillance*. Bloomsbury Publishing, 2011. 1

[10] J. Ryberg, "Privacy rights, crime prevention, cctv, and the life of mrs aremac," *Res Publica*, vol. 13, no. 2, pp. 127–143, 2007. 1

[11] B. J. Goold, "Privacy rights and public spaces: Cctv and the problem of the "unobservable observer"," *Criminal Justice Ethics*, vol. 21, no. 1, pp. 21–27, 2002. 1

[12] T. Lahtinen, L. Sintonen, H. Turtiainen, and A. Costin, "Towards CCTV-aware Routing and Navigation for Privacy, Anonymity, and Safety-Feasibility Study in Jyväskylä," in *2021 28th Conference of Open Innovations Association (FRUCT)*. IEEE, 2021. 1

[13] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial intelligence review*, vol. 43, no. 1, pp. 55–81, 2015. 1

[14] D. Onoro-Rubio and R. J. López-Sastre, "Towards perspective-free object counting with deep learning," in *European Conference on Computer Vision*. Springer, 2016. 1

[15] G. Verhoeven, M. Doneus, C. Briese, and F. Vermeulen, "Mapping by matching: a computer vision-based approach to fast and accurate georeferencing of archaeological aerial photographs," *Journal of Archaeological Science*, vol. 39, no. 7, pp. 2060–2070, 2012. 1

[16] T. Lahtinen, H. Turtiainen, and A. Costin, "BRIMA: low-overhead BRowser-only IMage Annotation tool," in *IEEE International Conference on Image Processing*, 2021. 1, 3

[17] L. Sintonen, H. Turtiainen, A. Costin, T. Hamalainen, and T. Lahtinen, "OSRM-CCTV: Open-source CCTV-aware routing and navigation system for privacy, anonymity and safety (Preprint)," *arXiv preprint arXiv:2108.09369*, 2021. 1

[18] A. Frome, G. Cheung, A. Abdulkader, M. Zennaro, B. Wu, A. Bissacco, H. Adam, H. Neven, and L. Vincent, "Large-scale privacy protection in Google Street View," in *12th international conference on computer vision*. IEEE, 2009. 1

[19] L. Huang, C. Gao, Y. Zhou, C. Xie, A. L. Yuille, C. Zou, and N. Liu, "Universal Physical Camouflage Attacks on Object Detectors," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1

[20] R. A. Sharma, E. Soltanaghaei, A. Rowe, and V. Sekar, "Lumos: Identifying and Localizing Diverse Hidden IoT Devices in an Unfamiliar Environment," in *USENIX Security Symposium*, 2022. 1

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision*. Springer, 2014. 2, 3, 5

[22] "COCO - Common Objects in Context," http://cocodataset.org/#home. 2

[23] G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *IEEE international conference on computer vision*, 2017. 2

[24] H. Sheng, K. Yao, and S. Goel, "CORRIGENDUM to Surveilling Surveillance: Estimating the Prevalence of Surveillance Cameras with Street View Data," in *AAAI/ACM Conference on AI, Ethics, and Society*, ser. AIES '21. ACM, 2021. [Online]. Available: https://dl.acm.org/action/downloadSupplement?doi=10.1145%2F3461702.3462525&file=3462525-corrigendum.pdf 2, 8

[25] D. Buzzo, "Signs of Surveillance," in *Technology, Design and the Arts-Opportunities and Challenges*. Springer, Cham, 2020. 2

[26] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, Z. Zhang, H. Lin, Y. Sun, T. He, J. Mueller, R. Manmatha, M. Li, and A. Smola, "Resnest: Split-attention networks," 2020. 2, 4

[27] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," 2017. 2, 4

[28] S. Qiao, L.-C. Chen, and A. Yuille, "Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution," 2020. 2, 4

[29] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," in *IEEE conference on computer vision and pattern recognition*, 2017. 2, 4

[30] L. Sintonen, H. Turtiainen, A. Costin, T. Hämäläinen, and T. Lahtinen, "OSRM-CCTV: CCTV-aware routing and navigation system for privacy and safety," in *12th International Conference on Business Modeling and Software Design (BMSD'22)*, 2022. 2, 7

[31] "Google Street View," https://google.com/streetview/explore/. 3, 4

[32] "Flickr," https://flickr.com/. 3

[33] "Baidu Total View," https://map.baidu.com/. 3

[34] "500px.com," https://500px.com/. 3

[35] "Unsplash.com," https://unsplash.com/. 3

[36] K. Wada, "labelme: Image Polygonal Annotation with Python," https://github.com/wkentaro/labelme, 2016. 3

[37] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *IEEE international conference on computer vision*, 2017. 3

[38] Y. Lee and J. Park, "CenterMask: Real-Time Anchor-Free Instance Segmentation," in *IEEE/CVF conference on computer vision and pattern recognition*, 2020. 3, 4

[39] "OpenStreetCam," https://openstreetcam.org/. 4

[40] Y. Lee and J. Park, "CenterMask2: Real-Time Anchor-Free Instance Segmentation," Apr. 2020. 4

[41] Y. Lee, J.-w. Hwang, S. Lee, Y. Bae, and J. Park, "An Energy and GPU-Computation Efficient Backbone Network for Real-Time Object Detection," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019. 4

[42] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the Gap Between Anchor-based and Anchor-free Detection via Adaptive Training Sample Selection," in *IEEE/CVF conference on computer vision and pattern recognition*, 2020. 4

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE conference on computer vision and pattern recognition*, 2016. 4

[44] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated Residual Transformations for Deep Neural Networks," in *Conference on Computer Vision and Pattern Recognition*. Honolulu, HI: IEEE, 2017. 4

[45] Y. Li, Y. Chen, N. Wang, and Z. Zhang, "Scale-Aware Trident Networks for Object Detection," in *IEEE/CVF International Conference on Computer Vision*, 2019. 4

[46] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2: A PyTorch-based modular object detection library," https://ai.facebook.com/blog/-detectron2-a-pytorch-based-modular-object-detection-library-/. 4

[47] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "Mmdetection: Open mmlab detection toolbox and benchmark," 2019. 4

[48] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," Facebook Research, 2019. 4

[49] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *International Conference on Machine Learning*. PMLR, 2015. 4