Robust People Counting in Public Spaces Using Overhead Cameras: Addressing Challenges of Rapid Movement, Headwear, and Bidirectional Flow

Anderson Tavares[1], Maycel Isaac[2], Jens Lundström[3] and Stefan Byttner[4]

Abstract

This work describes how datasets with specific people behavior, for example, movement speed and grouping, affect training and testing performances, which influences the overall model accuracy. By splitting the dataset based on how fast people move or how dense they are, we realize that datasets with fast-paced people do not generalize well. However, surprisingly, generalization from training in clustered people is even worse. This shows that, when lacking data, focusing on slow movement and isolated objects gives more information for the network to generalize.

Keywords

People counting, Object detection, Deep learning, Object tracking

1. Introduction

Accurate counting of people in public spaces using overhead cameras is crucial for applications such as crowd management, resource allocation, and security. However, various challenges, such as rapid movement, headwear occlusions, and bidirectional flow, can affect the precision of counting systems. Recent advances have focused on addressing these issues through innovative hardware and sophisticated algorithms.

Tracking and counting people has been done historically both with traditional image processing methods and more recent deep learning methods, or hybrid methods. As an example, in [1], a system is presented that tracks individuals by monitoring their heads and shoulders in complex environments. It addresses challenges such as occlusions and varying movement patterns by employing a combination of Hough Circular Gradient Transform for head detection and Histogram of Oriented Gradients (HOG) [2] based symmetry methods for shoulder detection. People tracking is done through deep learning, and counting of people is done through cross-line judgement as they are tracked.

This work analyzes how training on specific behavior affects training and testing, which affects tracking and counting.

SAIS2025: Swedish AI Society Workshop 2025, 16-17 June 2025, Halmstad, Sweden.

*Corresponding author.

†These authors contributed equally.

anderson@synteda.com (A. Tavares); maycel@synteda.com (M. Isaac); jens.lundstrom@hh.se (J. Lundström); stefan.byttner@hh.se (S. Byttner)

https://www.synteda.com/ (A. Tavares); https://www.synteda.com/ (M. Isaac); https://www.hh.se/ (J. Lundström); https://www.hh.se/ (S. Byttner)

0000-0002-8739-6479 (A. Tavares); 0000-0001-8804-5884 (J. Lundström); 0000-0002-0293-040X (S. Byttner)

2. Literature review

There is an extensive list of works on detecting and tracking people from overhead camera images. Ahamed et al. [3] develop a people counter. The software was executed on an Intel NUC 12 Pro Mini PC, without GPU. Even though they replaced a line by a region when counting, the uncertainty generated by the process and the measurement system (image grabbing + detection) is not taken into account.

Konrad et al. [4] introduce a people counter which uses the RAPID model for fish eye cameras. However, there is no work that analyzes the performance of RAPID with embedded devices.

Serrano-Cuerda et al. [5] shows a handmade people counter using image processing and morphological operations. It suffers from the limitations of image processing solutions: morphological operations will not account for scene variations like lighting. Also, the blob detector does not work with people with an appearance (e.g. clothing) similar to the ground. Since it is not a learnable model, it will not work with different camera setups (e.g. different heights and orientations) without manual reparametrization.

3. Methodology

The system can be divided into four main parts, illustrated in Figure 1.

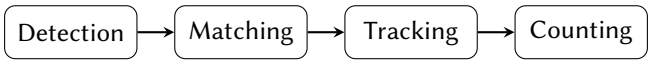

Figure 1: Main steps of the people counter

For custom model training, dataset collection and annotation are important steps. Annotations were made in our in-house annotation tool. Figure 2 shows a screenshot of it.

3.1. Detection

Detection means finding the location of instances of a specific class within an input source, for example, a camera video frame [6, 7]. This location may also be accompanied by additional parameters, such as the region of interest, also called the *bounding box*. Deep learning-based works usually make a distinction between object detection (finding bounding boxes) and *instance segmentation* (finding the actual subset of pixels that belongs to the instance).

Solutions for the object detection problem make extensive use of methods from areas such as Image and Signal processing, Computer Vision, and Machine Learning, with special attention to the increasing demand for deep learning. Our work gives special attention to models that are designed for accuracy, small size, speed, and low power consumption. A popular model that we choose for this analysis is MobileNet V3 [8] with SSD Lite [9].

In this work, the detector receives an image $I : \mathbb{Z}^d \rightarrow \mathbb{Z}^c$, where d is the coordinate space dimension and c is the number of channels, and outputs a list of bounding boxes $\{\mathbf{d}_i\}$ called *detections*. There are different representations for a box, like two opposing corners or a center/size. Some tasks may also require box orientation. Usually multiple detections may occur for a single target, leading to post-processing tasks such as *non-max suppression*.

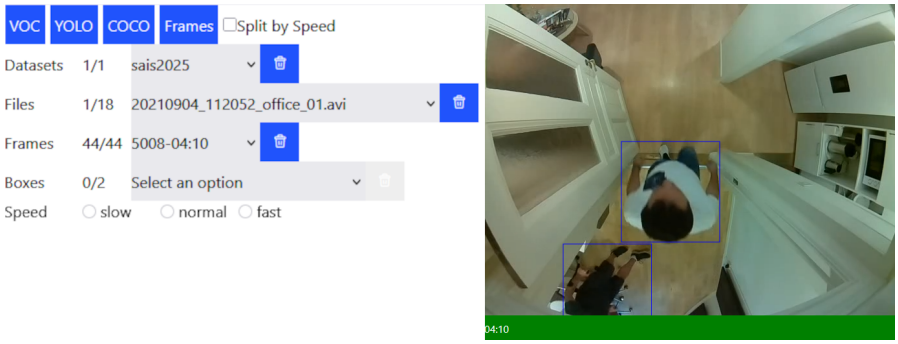

Figure 2: In-house web-based video annotation tool used during dataset annotation.

3.2. Matching

We apply Kuhn–Munkres (or Hungarian matching) algorithm [10] to match the detections $\{\mathbf{d}_i\}$ with previously detected people, or *tracks* $\mathbf{t}_j$, which outputs:

- The pairs $(\mathbf{d}_i, \mathbf{t}_j)$ where detection $\mathbf{d}_i$ matches track $\mathbf{t}_j$;
- The list $\{\mathbf{d}_i\}$ of detections without assigned track, i.e., potentially new people.
- The list $\{\mathbf{t}_j\}$ of tracks without detections, e.g., lost people.

3.3. Tracking

Tracking, in the context of video, is the process of associating target objects, a.k.a *tracks* $\mathbf{t}_j$, across different frames. The real states of those tracks are unknown. The goal is to estimate the state based on the input source and the intermediate results, for example the detections $\mathbf{d}_i$.

Kalman Filter [11] is a popular method which uses a series of noisy *measurements* (detections $\mathbf{d}_i$) observed over time to estimate the unknown state of the target object. Its basic form works with linear systems, where variations like *Extended Kalman Filter* (EKF) and *Unscented Kalman Filter* (UKF) are designed for non-linear systems [11].

3.4. Counting

We define a line to count how many tracks traverse it, in Hessian normal form: $\mathbf{l} = [l_1, l_2, -\delta]$, where $\langle \mathbf{l}, \mathbf{p} \rangle = 0, \forall \mathbf{p} \in \mathcal{L}$ ($\mathcal{L}$ is the *trace* of the line), $[l_1, l_2]$ is the normal vector of the line, and δ is the orthogonal distance from the line to the origin. Since $\mathbf{l}$ is homogeneous, $\mathbf{l}$ and $\alpha\mathbf{l}$ ($\alpha \neq 0$) represent the same line. The *line normalization*[12] of $\alpha\mathbf{l} = \mathbf{v} = [v_1, v_2, v_3]$ recovers $\mathbf{l}$:

$$\mathbf{l} = \mathbf{v}\frac{-\text{sign}(v_3)}{\|\mathbf{v}_{:2}\|}, \text{ where } \mathbf{v}_{i:j} = [v_i, v_{i+1}, \ldots, v_j] \text{ and } \mathbf{v}_{:j} = \mathbf{v}_{1:j}.$$

An easy way to find the line that passes over two homogeneous points $(\mathbf{p}_1, \mathbf{p}_2)$ is $\mathbf{l} = \mathbf{p}_1 \times \mathbf{p}_2$, that is, the cross product between them. For any homogeneous point $\mathbf{p}$, the *signed distance* of the point to the line $\mathbf{l}$ is their dot product: $\delta_\mathbf{p} = \langle \mathbf{p}, \mathbf{l} \rangle$.

Assuming that we choose the center $\mathbf{c_t}$ of the track $\mathbf{t}$, for two subsequent frames, we can check whether $\mathbf{c_t}$ crosses $\mathbf{l}$ by verifying if its distance $\text{sign}\langle \mathbf{c_t}, \mathbf{l} \rangle$ changes sign.

4. Results

4.1. Based on people speed

We compared training sessions based on speed (fast and normal). Figure 3 (left) shows two training sessions when training and testing in the same speed category (but different data set splits). Figure 3 (right) shows the training sessions when choosing different speeds for training and validation. We see that a dataset with walking speed conveys more information than one with running speed.

Table 1
Validation Loss (based on speed)

		training					
		normal			fast		
		classif.	regres.	total	clasif.	regres.	total
test	fast	2.57	2.327	4.897	**2.312**	1.559	3.871
	normal	2.479	2.344	4.823	2.367	**1.453**	**3.82**

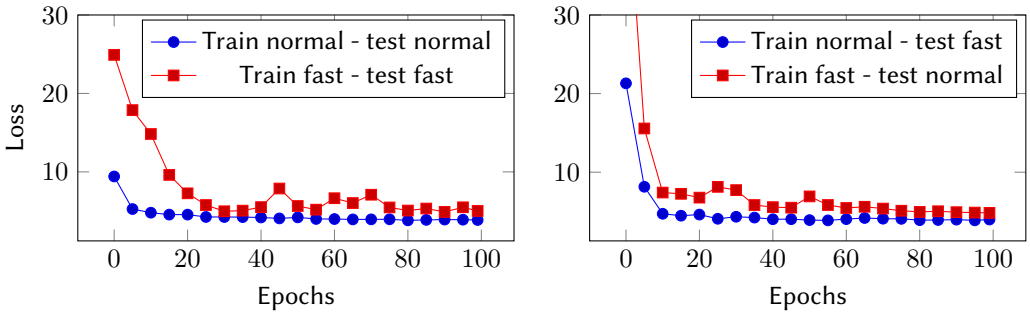

Figure 3: Validation loss: training and testing on same speed (left) or different ones (right).

4.2. Based on people clustering

We also investigated the performance of the model when we split the dataset based on how clustered the people are. We split the database based on the distance of the bounding box center to the closest neighbor. We set a threshold for this split. Table 2 shows the validation loss.

Table 2
Validation Loss (based on people clustering)

		training					
		alone			grouped		
		classif.	regres.	total	clasif.	regres.	total
test	alone	**2.299**	**1.498**	**3.797**	7.281	3.795	11.08
	grouped	2.472	2.139	4.611	2.362	1.764	4.186

Even though we have a low validation loss when training on clustered bounding boxes when testing also on clustered ones, the results show a poor generalization behavior when testing against other types of datasets.

5. Conclusion

In this work, we showed in this work that generalization of the model is affected by how the training dataset represents the characteristics of the people. We analyze how fast speed and grouped instances might affect the accuracy of inference in slow and individual instances.

References

[1] M. Gochoo, S. A. Rizwan, Y. Y. Ghadi, A. Jalal, K. Kim, A systematic deep learning based overhead tracking and counting system using rgb-d remote cameras, Applied Sciences 2021, Vol. 11, Page 5503 11 (2021) 5503. doi:10.3390/APP11125503.

[2] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005 I (2005) 886–893. doi:10.1109/CVPR.2005.177.

[3] I. Ahamed, C. D. Ranathunga, D. S. Udayantha, B. K. K. Ng, C. Yuen, Real-time ai-driven people tracking and counting using overhead cameras (2024). URL: http://arxiv.org/abs/2411.10072. doi:10.48550/arXiv.2411.10072.

[4] J. Konrad, M. Cokbas, P. Ishwar, T. D. Little, M. Gevelber, High-accuracy people counting in large spaces using overhead fisheye cameras, Energy and Buildings 307 (2024) 113936. doi:10.1016/j.enbuild.2024.113936.

[5] J. Serrano-Cuerda, J. C. Castillo, A. Fernández-Caballero, Indoor overhead video camera for efficient people counting, Jurnal Teknologi 63 (2013) 17–22. doi:10.11113/jt.v63.1948.

[6] R. Hartley, A. Zisserman, Multiple view geometry in computer vision (cited by: 11343), Cambridge University Press 2 (2004) 672. URL: http://www.robots.ox.ac.uk/~vgg/hzbook/.

[7] A. C. Heaton Jeff Ian Goodfellow, Yoshua Bengio, Deep learning, Genetic Programming and Evolvable Machines 19 (2018) 305–307.

[8] A. Howard, M. Sandler, B. Chen, W. Wang, L. C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, Q. Le, H. Adam, Searching for mobilenetv3, Proceedings of the IEEE International Conference on Computer Vision 2019-October (2019) 1314–1324. URL: https://arxiv.org/abs/1905.02244v5. doi:10.1109/ICCV.2019.00140.

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, Ssd: Single shot multibox detector, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9905 LNCS (2015) 21–37. URL: http://arxiv.org/abs/1512.02325http://dx.doi.org/10.1007/978-3-319-46448-0_2. doi:10.1007/978-3-319-46448-0_2.

[10] H. W. Kuhn, The hungarian method for the assignment problem, Naval Research Logistics Quarterly 2 (1955) 83–97. doi:10.1002/NAV.3800020109.

[11] K. P. Murphy, Probabilistic machine learning : advanced topics, The MIT Press, 2023.

[12] W. Förstner, B. P. Wrobel, Photogrammetric Computer Vision, volume 11, Springer International Publishing, 2016. URL: http://link.springer.com/10.1007/978-3-319-11550-4. doi:10.1007/978-3-319-11550-4.