

People Analytics for Linear Accelerator Rooms

https://github.com/anrobr/MSDS_462-DL-Final_Project

1 Abstract

This report presents a research project to automate the counting of people in radiation treatment rooms using computer vision and edge-computing to further reduce the risk of accidental irradiation of medical staff. Three deep learning-based object detectors namely YOLO v4 tiny, MobileNetV1-OpenPose, and MobileNetV2-SSDLite are evaluated based on the frame-by-frame object detection performance, the inference speed in frames per second, and 16 different entry and exit scenarios focused on the maze entry, which is the single point of entry to the radiation treatment room.

2 Introduction

According to (Sung, Ferlay, Siegel, & Laversanne, 2021) global cancer cases are projected to be at around 28 million in 2040, which would represent a rise of around 47%. As stated by (Hossein-zadeh, Banaee, & Nedaie, 2017) radiotherapy accounts for around 50% of the treatment cases and thus presents a core treatment modality. Radiation, however, presents a severe health risk to medical staff that constantly operate in the treatment rooms. Layered security measures are in place to protect medical staff from accidental irradiation. One such measure is the comprehensive video surveillance of the treatment room, which is manually checked before irradiation is started. Enhancing the manual check by automating the counting of people in the treatment room should help to further enhance the safety of medical professionals.

3 Literature Review

Several deep learning-based object and people detectors exist that are suitable for real-time people detection needed for the project. Among those models are YOLO v4 (Bochkovskiy, Wang, & Liao, 2020) and variations as proposed by (Jiang, Zhao, & Jia, 2020), (Guo, Lv, Zhang, & Zhang, 2021), and (Yao, Cai, Fan, & Li, 2022). A modified version is YOLO v4 tiny (Bochkovskiy, 2020) is especially suited for edge devices. Another such object detector is MobileNetV2-SSDLite as proposed by (Sandler, Howard, Zhu, Zhmoginov, & Chen, 2018), which also strikes a good balance between inference performance and detection accuracy. Both object detectors use bounding boxes for object localization. Another approach to detecting people in videos is based on human pose estimation as proposed by (Cao, Hidalgo, Simon, Wei, & Sheikh, 2021). A common backend for human pose-based detectors is MobileNetV1 or newer variants as presented in (Howard, et al., 2017). The aforementioned object detectors can be combined with centroid tracking or SORT (Wojke, Bewley, & Paulus, 2017) to use object detectors for object tracking.

4 Objectives

Clinics need to operate in a safe and cost-efficient way to ensure the sustainability of treatments. Thus, the focus of the research is to evaluate deep-learning-based, real-time capable, people detectors deployed on low-cost edge devices to count the number of people in a radiation treatment room.

5 Methodology and methods

Two edge devices were used for this research, the Intel Neural Compute Stick 2 (NCS2) and the NVIDIA Jetson Nano (Nano). Implementation and evaluation were mainly done using OpenVINO and NCS2. The Nano was used for inference performance comparison and video acquisition. The models were compiled to CPU using FP32 precision, NCS2 using FP16 precision, and GPU for the Nano. Quantization was not evaluated as part of this project.

The video frames with a size of 1280x1080 pixels were cropped to 600x1080 with the center on the maze entry. Three pre-trained models namely YOLO v4 tiny, MobileNetV1-OpenPose, and MobileNetV2-SSDLite were used. YOLO and MobileNetV2 were trained on the MS COCO dataset with 80 and 92 classes respectively. Classes other than “person” were discarded. An empirical tuning of detector-specific hyperparameters was carried out with a focus on class confidence and IOU to limit the number of redundant bounding boxes. A rudimentary centroid-based tracking implementation from (Rosebrock, 2018) was used as part of the counting mechanism implementation. The following metrics were used for evaluation:

- The avg. inference speed in FPS on random input over one minute.
- The ratio of correctly detected people on a frame-by-frame basis (object detection performance).
- The number of correctly counted people using 16 different entry and exit scenarios (combination of object detection performance and object tracker). A description of the scenarios is listed in the appendix.

6 Results

6.1 YOLO v4 tiny (Bounding Box-Based Object Detection)

Inference Device	Avg. Inf. Performance (FPS)	Obj. Tracking Rate	Obj. Detection Rate
Intel NCS2	28,48	0,81	0,58
Intel CPU	65,66	0,81	0,66

The use of 3 slack frames to re-identify a lost object presumably accounts for the significantly higher obj. tracking rate as compared to the obj. detection rate.

6.2 MobileNetV1 (Human Pose Estimation-Based Object Detection)

Inference Device	Avg. Inf. Performance (FPS)	Tracking Rate	Obj. Detection Rate
Intel NCS2	8,74	0,44	0,81
Intel CPU	29,61	0,44	0,81
NVIDIA Jetson Nano	14,75	Equivalent	Equivalent

The low tracking rate of the pose-based model is most likely caused by only using the detected face “joint”, which is unstable e.g.; to head rotations. Incorporating a combination of several body joints should improve tracking rate significantly.

6.3 MobileNetV2-SSDLite (Bounding Box-Based Object Detection)

Inference Device	Avg. Inf. Performance (FPS)	Tracking Rate	Obj. Detection Rate
Intel NCS2	26,87	0,8125	0,77
Intel CPU	193,32	0,8125	0,77
NVIDIA Jetson Nano	22	Equivalent	Equivalent

6.4 Summary

MobileNetV2 achieves the highest inf. performance and tracking rate on the cost-efficient edge devices. Only the pose-based model on the NCS2 was too slow to be suitable for real-time detection. YOLO and MobileNetV2-SSDLite had the most difficulty with detecting people from a 90-degree angle. Re-training both models for this specific scenario or using a camera position from the ceiling should significantly boost detection and tracking accuracy. Differences in illumination did not significantly impact object detection and tracking performance. This research used the weakest form of object detection-based tracking through centroid-based tracking. Substantially more robust tracking algorithms like DeepSORT by (Wojke, Bewley, & Paulus, 2017) or its improved version StrongSORT as proposed by (Du, Song, Yang, & Zhao, 2022) exist and should be considered for future research.

7 References




- Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2017). Simple Online and Realtime Tracking. *IEEE International Conference on Image Processing (ICIP)* (pp. 1-5). Phoenix, AZ, USA: IEEE. doi:10.1109/ICIP.2016.7533003
- Bochkovskiy, A. (2020, May 3). *Darknet: Open Source Neural Networks in Python*. Retrieved May, 2022 from github: <https://github.com/AlexeyAB/darknet>
- Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. 1-17. From <https://arxiv.org/abs/2004.10934>
- Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., & Sheikh, Y. (2021). OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (pp. 172-186). IEEE. doi:10.1109/TPAMI.2019.2929257
- Du, Y., Song, Y., Yang, B., & Zhao, Y. (2022). *StrongSORT: Make DeepSORT Great Again*. doi:10.48550/arXiv.2202.13514
- Guo, C., Lv, X.-l., Zhang, Y., & Zhang, M.-l. (2021). Improved YOLOv4-tiny network for real-time electronic component detection. *Nature: Scientific Reports*, 1-13. doi:10.1038/s41598-021-02225-y
- Hosseinzadeh, E., Banaee, N., & Nedaie, H. A. (2017). Cancer and Treatment Modalities. *Current Cancer Therapy Reviews*, 13(1), 17-27. doi:10.2174/1573394713666170531081818
- Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., . . . Adam, H. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. From <https://arxiv.org/pdf/1704.04861.pdf>
- Jiang, Z., Zhao, L., & Jia, Y. (2020). Jiang, Zicong, Liquan Zhao, Shuaiyang Li and Yanfei Jia. "Real-time object detection method based on improved YOLOv4-tiny." *ArXiv abs/2011.04244 (2020): n. pag.* From <https://arxiv.org/ftp/arxiv/papers/2011/2011.04244.pdf>
- National Cancer Institute. (2019, January 8). *Radiation Therapy to Treat Cancer*. Retrieved May, 2022 from cancer.gov: <https://www.cancer.gov/about-cancer/treatment/types/radiation-therapy>
- roboflow. (2020, May 4). *YOLOv4-tiny*. From models.roboflow.com: <https://models.roboflow.com/object-detection/yolov4-tiny-darknet>
- Rosebrock, A. (2018, July 23). *Simple object tracking with OpenCV*. Retrieved May, 2022 from pyimagesearch.com: <https://pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/>
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE. doi:10.1109/CVPR.2018.00474
- Sung, H., Ferlay, J., Siegel, R. L., & Laversanne, M. (2021). Global Cancer Statistics 2020: GLOBOCAN Estimates of Incidence and Mortality Worldwide for 36 Cancers in 185 Countries. *CA: A Cancer Journal for Clinicians*, 71(3), 209-249. doi:10.3322/caac.21660
- Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. *IEEE International Conference on Image Processing (ICIP)*. Beijing, China: IEEE.





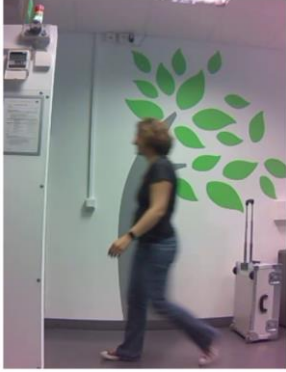
Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. *IEEE International Conference on Image Processing (ICIP)*. Beijing, China: IEEE.
doi:10.1109/ICIP.2017.8296962

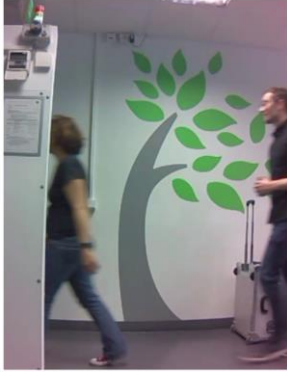

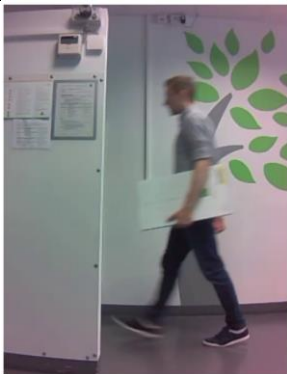


Yao, J., Cai, D., Fan, X., & Li, B. (2022). Improving YOLOv4-Tiny's Construction Machinery and Material Identification Method by Incorporating Attention Mechanism. *Mathematics*, 1-20.
doi:10.3390/math10091453


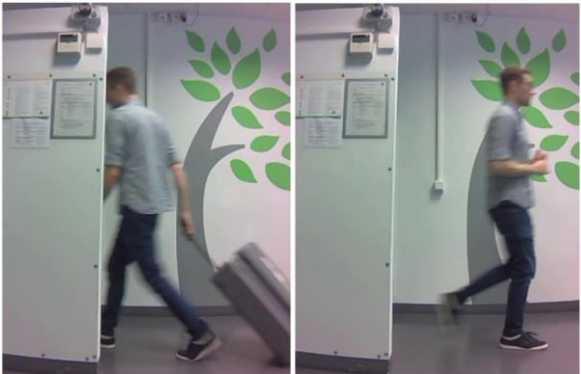
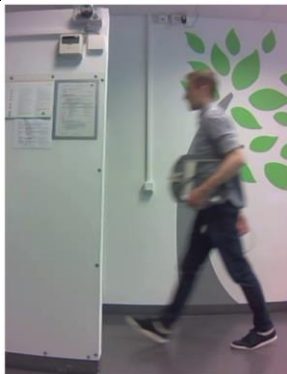
8 Appendix

8.1 Description of the 16 entry and exit scenarios

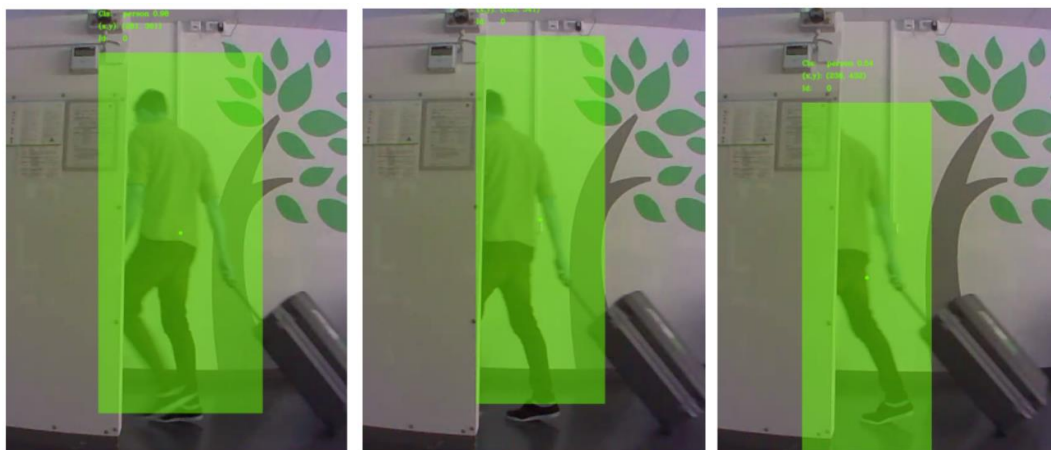
Video #	Description	Example
1	Low illumination, single person, no equipment, entry and exit.	
2	Low illumination, single person, detector with diagonal upward orientation, entry and exit.	
3	Low illumination, single person, opaque detector with horizontal orientation, entry and exit.	

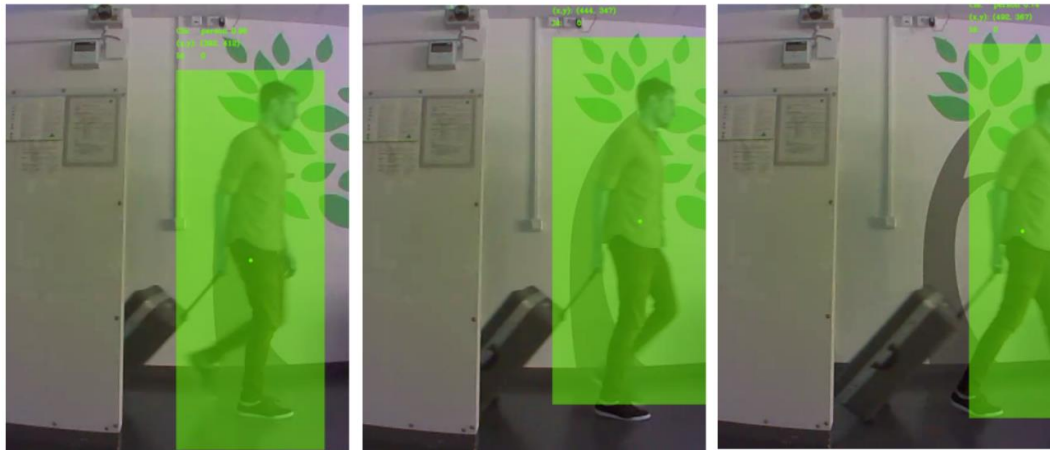
4	Low illumination, single person, opaque detector with vertical orientation, entry and exit.	
5	Low illumination, single person, rollable equipment, entry and exit.	
6	Low illumination, single person, transparent detector, reversal before entering facing the camera.	
7	Low illumination, single person, transparent detector, reversal before entering facing away from the camera.	
8	High illumination, single person, no equipment, entry and exit.	

9	High illumination, two people, no equipment, entry and exit.	
10	High illumination, two people, no equipment, repeated entry and exit.	
11	High illumination, single person, opaque detector with horizontal orientation, entry and exit.	
12	High illumination, single person, opaque detector, carried into the maze, not out.	
13	High illumination, single person, no equipment.	

14	High illumination, single person, rollable equipment, entry and exit.	
15	High illumination, single person, rollable equipment, rolled into the maze, not out.	
16	High illumination, single person, transparent detector, entry and exit.	

8.2 Examples of bounding box-based object detections





8.3 Examples of human pose-based object detections



8.4 Detailed results

8.4.1 YOLO v4 tiny

Inference Device	Video #	Precision	Count Correct	FPS	Frames (1 obj)	Detected	Ratio	Frames (2 objs)	Detected
Intel NCS2	1	FP16	1		60	22	0,37	0	0
Intel NCS2	2	FP16	1		69	27	0,39	0	0
Intel NCS2	3	FP16	1		63	33	0,52	0	0
Intel NCS2	4	FP16	1		54	28	0,52	0	0
Intel NCS2	5	FP16	1		88	48	0,55	0	0
Intel NCS2	6	FP16	1		60	42	0,7	0	0
Intel NCS2	7	FP16	1		52	38	0,73	0	0
Intel NCS2	8	FP16	0		77	43	0,56	0	2
Intel NCS2	9	FP16	0		180	87	0,48	7	4
Intel NCS2	10	FP16	0		216	148	0,69	0	1
Intel NCS2	11	FP16	1		52	32	0,62	0	2
Intel NCS2	12	FP16	1		54	29	0,54	0	0
Intel NCS2	13	FP16	1		50	37	0,74	0	0
Intel NCS2	14	FP16	1		59	39	0,66	0	0
Intel NCS2	15	FP16	1		55	33	0,6	0	0
Intel NCS2	16	FP16	1		52	34	0,65	0	0
			0,81	28,48			0,58		
				FPS					
Intel CPU	1	FP32	1		60	25	0,42	0	0
Intel CPU	2	FP32	1		69	36	0,52	0	0
Intel CPU	3	FP32	1		63	32	0,51	0	0
Intel CPU	4	FP32	1		54	31	0,57	0	0
Intel CPU	5	FP32	1		88	50	0,57	0	0
Intel CPU	6	FP32	1		60	44	0,73	0	0
Intel CPU	7	FP32	1		52	42	0,81	0	0
Intel CPU	8	FP32	0		77	68	0,88	0	2
Intel CPU	9	FP32	0		180	104	0,58	7	4
Intel CPU	10	FP32	0		216	162	0,75	0	1
Intel CPU	11	FP32	1		52	33	0,63	0	2
Intel CPU	12	FP32	1		54	40	0,74	0	0
Intel CPU	13	FP32	1		50	35	0,7	0	0
Intel CPU	14	FP32	1		59	36	0,61	0	0
Intel CPU	15	FP32	1		55	40	0,73	0	0
Intel CPU	16	FP32	1		52	44	0,85	0	0
			0,81	65,66			0,66		

8.4.2 MobileNetV1 OpenPose

Inference Device	Video #	Precision	Count Correct	FPS	Frames (1 obj)	Detected	Ratio	Frames (2 objs)	Detected
Intel NCS2	1	FP16	0		60	47	0,78	0	0
Intel NCS2	2	FP16	0		69	50	0,72	0	1
Intel NCS2	3	FP16	1		63	51	0,81	0	1
Intel NCS2	4	FP16	0		54	43	0,8	0	3
Intel NCS2	5	FP16	0		88	41	0,47	0	2
Intel NCS2	6	FP16	1		60	42	0,7	0	0
Intel NCS2	7	FP16	0		52	44	0,85	0	0
Intel NCS2	8	FP16	0		77	59	0,77	0	0
Intel NCS2	9	FP16	0		180	134	0,74	7	2
Intel NCS2	10	FP16	0		216	190	0,88	0	0
Intel NCS2	11	FP16	1		52	48	0,92	0	0
Intel NCS2	12	FP16	1		54	51	0,94	0	0
Intel NCS2	13	FP16	1		50	46	0,92	0	0
Intel NCS2	14	FP16	1		59	49	0,83	0	0
Intel NCS2	15	FP16	0		55	48	0,87	0	0
Intel NCS2	16	FP16	1		52	48	0,92	0	0
			0,44	8,74			0,81		
				FPS					
Intel CPU	1	FP32	0		60	49	0,82	0	0
Intel CPU	2	FP32	0		69	52	0,75	0	1
Intel CPU	3	FP32	1		63	50	0,79	0	1
Intel CPU	4	FP32	0		54	41	0,76	0	3
Intel CPU	5	FP32	0		88	41	0,47	0	2
Intel CPU	6	FP32	1		60	42	0,7	0	0
Intel CPU	7	FP32	0		52	46	0,88	0	0
Intel CPU	8	FP32	0		77	60	0,78	0	0
Intel CPU	9	FP32	0		180	133	0,74	7	2
Intel CPU	10	FP32	0		216	190	0,88	0	0
Intel CPU	11	FP32	1		52	46	0,88	0	0
Intel CPU	12	FP32	1		54	52	0,96	0	0
Intel CPU	13	FP32	1		50	46	0,92	0	0
Intel CPU	14	FP32	1		59	51	0,86	0	0
Intel CPU	15	FP32	0		55	48	0,87	0	0
Intel CPU	16	FP32	1		52	49	0,94	0	0
			0,44	29,61			0,81		

8.4.3 MobileNetV2-SDDLite

Inference Device	Video #	Precision	Count Correct	FPS	Frames (1 Detected	Ratio	Frames (2 objs)	Detected
Intel NCS2	1	FP16	1		60	42	0,7	0
Intel NCS2	2	FP16	1		69	46	0,67	0
Intel NCS2	3	FP16	1		63	41	0,65	0
Intel NCS2	4	FP16	1		54	38	0,7	0
Intel NCS2	5	FP16	1		88	57	0,65	0
Intel NCS2	6	FP16	1		60	40	0,67	0
Intel NCS2	7	FP16	0		52	38	0,73	0
Intel NCS2	8	FP16	1		77	72	0,94	0
Intel NCS2	9	FP16	0		180	127	0,71	0
Intel NCS2	10	FP16	1		216	174	0,81	0
Intel NCS2	11	FP16	0		52	37	0,71	0
Intel NCS2	12	FP16	1		54	47	0,87	0
Intel NCS2	13	FP16	1		50	49	0,98	0
Intel NCS2	14	FP16	1		59	47	0,8	0
Intel NCS2	15	FP16	1		55	47	0,85	0
Intel NCS2	16	FP16	1		52	45	0,87	0
			0,8125	26.87		0,77		
				FPS				
Intel CPU	1	FP32	1		60	42	0,7	0
Intel CPU	2	FP32	1		69	45	0,65	0
Intel CPU	3	FP32	1		63	42	0,67	0
Intel CPU	4	FP32	1		54	39	0,72	0
Intel CPU	5	FP32	1		88	56	0,64	0
Intel CPU	6	FP32	0		60	41	0,68	0
Intel CPU	7	FP32	1		52	38	0,73	0
Intel CPU	8	FP32	1		77	71	0,92	0
Intel CPU	9	FP32	0		180	132	0,73	0
Intel CPU	10	FP32	1		216	173	0,8	0
Intel CPU	11	FP32	0		52	39	0,75	3
Intel CPU	12	FP32	1		54	48	0,89	0
Intel CPU	13	FP32	1		50	48	0,96	0
Intel CPU	14	FP32	1		59	46	0,78	0
Intel CPU	15	FP32	1		55	45	0,82	0
Intel CPU	16	FP32	1		52	46	0,88	0
			0,8125	193.32		0,77		