AN EFFICIENT SOLUTION FOR PEOPLE DETECTION, TRACKING AND COUNTING USING CONVOLUTIONAL NEURAL NETWORKS

Eduard COJOCEA[2]
Traian REBEDEA[3]

Abstract: *The number of unique persons walking near a shop or inside a mall is relevant since it can indicate the possible extension margin of a certain business. Also, being able to extract statistics regarding gender, age group and so on, can offer key insights regarding how to better manage and stock a business. In this paper we present a system which detects, tracks and counts the number of people in a video stream. The results obtained can be visualised in a GUI interface which allows for customizing multiple visualization tools. We use YOLOv3, a Convolutional Neural Network model, for object detection and Deep SORT for tracking. We describe how the system works on different hardware architectures: on a server with two high-end GPUs and on various edge devices, such as Raspberry Pi 3, Raspberry Pi 4 and NVidia Jetson TX2.*

Keywords: *People flow analysis, CNN, Object Detection, Object Tracking, Computer Vision, Deep Learning*

1. Introduction

In the recent years, the rapid growth of the Computer Vision field made it possible for the video camera surveillance process to be semi-automated or fully automated, instead of a person manually reviewing the stream. This is possible thanks to great leaps forward in solutions for visual problems, especially Object Detection and Object Tracking. The models used for such problems were initially heuristic models or simple Artificial Neural Networks using manually extracted features from images as input. In the last decade, the approach has shifted, being centred around Convolutional Neural Networks (CNNs). The new architectures coupled with significant improvements in computational power of the hardware, boasting high-end GPUs, has enabled models to achieve great speed and performance. Thus, they can be used in the real world with success, not only in heavily controlled environments.

[2] University Politehnica of Bucharest, 313 Splaiul Independentei, Bucharest, Romania, iedi.cojocea@gmail.com
[1] Open Gov SRL, 95 Blvd. Alexandru Ioan Cuza, Bucharest, Romania
[3] University Politehnica of Bucharest, 313 Splaiul Independentei, Bucharest, Romania, traian.rebedea@cs.pub.ro
[2] Open Gov SRL, 95 Blvd. Alexandru Ioan Cuza, Bucharest, Romania

There are many scenarios where the ability to extract relevant information from video streams can be very useful for increasing profit, increasing security, better knowledge of a business and so on. Shopping malls can benefit from such a system, which can offer information regarding the total number of unique customers and some statistics such as gender, age group, height and weight distributions. All these data can be used by the shops to have relevant stocks and increase their profit. Also, such a system can be extended for security reasons. It could detect visual anomalies in crowded place, such as a person falling on the subway lines, violent behaviour in crowds, luggage left unattended and so on.

Thus, the advancements in Computer Vision

2. Related work

2.1. Convolutional Neural Networks

CNNs are a class of Neural Networks which have the distinct features where the neurons are displayed in a multidimensional array fashion, each neuron receiving input from a certain window of neurons from the previous layer, in contrast with the fully connected layers where each neuron in a layer is connected with each neuron from the previous layer. Also, the weights for the connections between a neuron and the neurons in the previous layer are shared, acting as a filter. Usually, between convolutional layers there are pooling layers which reduce the dimensions of the data.

Despite human performance in Object Detection and Tracking, it is significantly harder to explain why an object belongs to a certain class or to generate a mathematical description of an object. Thus, instead of manually extracting features from images, the automatic feature extraction capability of CNNs enables it to learn representations of objects, simply by analysing a big number of labelled images. This, correlated with the capability of processing whole images as they are and the shared weights, make CNNs the go to solution for image processing.

Even though the concept of CNN existed before, it started to dominate the Computer Vision field since 2012, when AlexNet [1] won the ImageNet Large Scale Visual Recognition Competition by a large margin, using a CNN architecture. Since then, there has been exponential growth of CNN models, offering increasingly faster and more performant solutions for processing images and video streams.

2.2 Object detection

Object Detection is a Computer Vision problem which requires identifying multiple objects in an image, localizing and classifying them. Also, it is important to be able to distinguishing between multiple objects belonging to the same class. Thus, it is a central problem when processing images and video streams.

Early approaches for Object Detection implied manually extracting features from images and then using simple classifiers, while modern approaches use CNNs.

One modern approach is extracting regions of interest from an image, which are the most likely candidates to contain a single dominant object. After these regions are identified, a classifier is run over each one of them. Region based Convolutional Neural Network (RCNN) [2], Fast RCNN [3], Faster RCNN [4] and Mask RCNN [5] represent successive iterations of this approach, each improving upon the performance and speed of the previous. The main disadvantage of this approach is the fact that a classifier is used for each region of interest, making inference thousands of times per image, which slows down significantly the speed of the model.

Another modern approach is to split an image into a grid of cells, each being responsible for a limited number of objects. Thus, the image is traversed a single time, which allows the models using this approach to be very fast. The YOLO (You Only Look Once) family is representative in this case, with the successive models YOLO [6], YOLO9000 [7], YOLOv3 [8], YOLOv4 [9]. These models are very fast, while keeping high performance at the same time. Also, there are a series of light versions of these models, generally known as Tiny YOLO [10], which can be used when high computational power is not available, such as in the case of edge devices.

Objects as Points

2.3 Object Tracking

Object Tracking represent a Computer Vision problem where it is necessary to identify if two objects in successive frames are one and the same object. There are two main classes of Object Tracking: Single Object Tracking and Multiple Objects Tracking. The first class requires tracking a single object through a video stream, representing the simple case. The second class has many more real world applications, but is significantly harder.

One approach to solve Object Tracking is to use the Euclidian distance between objects positions in successive frames and pairing the objects with the smallest distance, considering them to be the same object. This approach works well when tracking a single object, but when multiple objects are involved, it falls short, offering a high error rate, especially when the objects are clustered or occluded.

Simple Online and Realtime Tracking (SORT) [11] represents an improvement to this approach, by using a Kalman filter alongside the Euclidian distance, in order to predict the trajectory of objects, which will make it easier to track the objects.

A further improvement is presented in Deep SORT [12], where besides using the Euclidian distance and a Kalman filter, a deep metric is used for visual recognition. Thus, in order for an object to be considered the same in successive frames, it not only needs to be in the same vicinity and following roughly the projected trajectory, but it also must have similar appearances. This metric represents an

array with 128 values, learned by a CNN trained on a big dataset of pedestrians [13]. Thus, this array encodes some visual features of an object, and it will successfully identify the same object in successive frames if the detections are candidates (they are considered the same object by the Euclidian distance and the Kalman filter) and if the cosine distance between the projection of the two 128^{th} dimensional points (the two feature arrays) on the unit hypersphere is smaller than an empirically identified threshold.

3. System overview

An end to end system which acquires frames from a video stream and identifies, tracks and counts the people has many real world applications, but can differ based on the exact use, the required performance, available hardware and budget. Thus, we explore two different approaches: server processing and edge processing, with a third approach being possible as a combination of the first two.

3.1 Server processing

Server processing represents the approach where the data acquisition and the processing are separate. Usually the latter is done on a server with high computational power available, thus enabling us to use the models with the highest performance, while being able to run at realtime speed (30 frames per second).

This approach allows for the best performance and speed, but has some drawbacks as well: it is significantly more expensive, it will have an overhead caused by the data transmission and will depend on a secure and fast connection to the data acquisition module.

Thus, in this context, we used YOLOv3 for object detection, filtering the objects detected such as it will detect only persons and for tracking, we used Deep SORT.

The server has two GTX 1080 Ti GPUs and 64GB of RAM and the data acquisition is realised by an IP camera, sending the video stream to the server via a wireless or wired connection.

3.2 Edge processing

Edge processing represents the approach where the video stream is processed on a small embedded device directly connected to the data acquisition module. Thus, it is a self-contained system which can be easily used anywhere. Another advantage is that it is nor relying on a fast and secure connection, the video stream being directly transmitted via cable. Also, it has significantly lower prices than the server approach. The biggest disadvantage consists of its lower computational power available, meaning that lighter models have to be used, which in turn represent a drop in performance.

Thus, for Object Detection, we use Tiny YOLOv3 and for Object Tracking we use Deep SORT.

As hardware, we used the following edge devices: Raspberry Pi 3, Raspberry Pi 4, Intel NCS, Intel NCS 2 and Nvidia Jetson TX2.

3.3 Central platform

After the data from the video streams is processed either by the server, edge devices or a hybrid approach, the results are sent to a web application offering a GUI interface. Thus, the results can be visualised and further processed by using multiple visualisation and formatting tools. In Figure 1 we present a chart showing the total number of people detected, which also has filters based on age or sex and the requested time period.

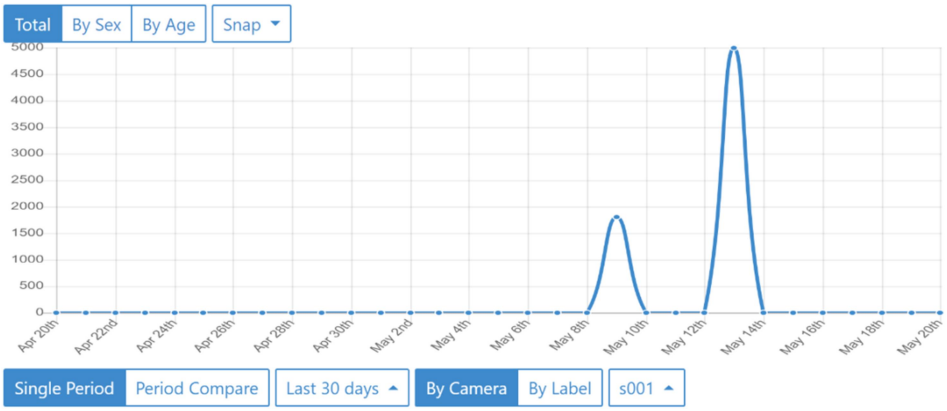

Figure 1 – Chart containing the total number of people detected. The bucket size is equal to one day.

In Figure 2, we present a pie chart which shows the distribution of people by age or sex. It is worth noting that the models that identify gender and age group are still work in progress, and the data regarding age and gender used in these charts is generated.

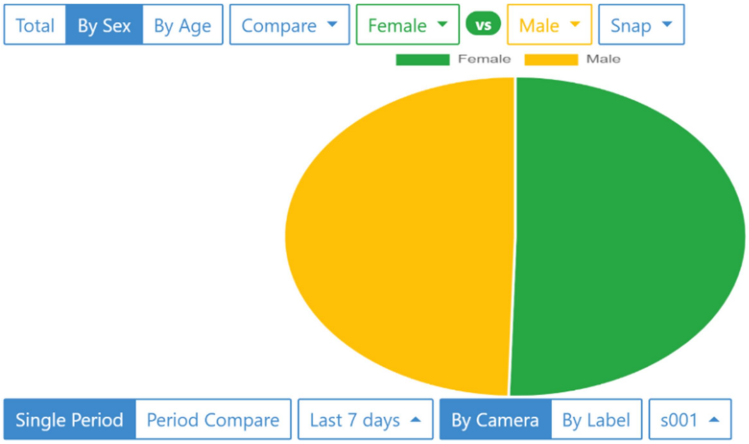

Figure 2 – Pie chart representing the distribution of people.

4. Results

In this section we present a comparison between the results obtained on the dedicated server and the edge devices available.

Cojocea, Hornea & Rebedea [14] make a comparison between some edge devices and we extend this by adding some new devices, thus enriching the general overview. For our experiments we used the dataset provided by the Multi Object Tracking Challenge 2017 [15], which contains 14 videos with crowds of people, with durations of tens of seconds.

4.1 Object Detection

In Table 1 we present a comparison regarding performance and speed between the available edge devices and the dedicated server. We can observe that most edge devices are still lacking in speed, even though we used light models. But, it is worth noting that Nvidia Jetson TX2 is capable of running both light and deep models, the first in real time and the second at a reasonable speed of more than 3 frames per second. Also, when comparing the performance of the light and deep models, we can observer that the latter is twice as performant than the first, which can have an impact in how well the system will work.

Device	Model	Frames per second	mAP
Raspberry Pi 3	Tiny YOLOv3	0.33	29.8
Raspberry Pi 3 + Intel NCS	Tiny YOLOv3	2.90	
Raspberry Pi 4	Tiny YOLOv3	0.87	
Raspberry Pi 4 + Intel NCS2	Tiny YOLOv3	7.86	
Nvidia Jetson TX2	Tiny YOLOv3	37.8	29.8
	YOLOv3	3.1	53.2
Dedicated server	Tiny YOLOv3	343	29.8
	YOLOv3	47.44	53.2

Table 1 – Results for Object detection

4.2 Object Tracking

In Table 2 we present the results obtained for Object Tracking (we track only people), using multiple object tracking accuracy (MOTA) and multiple object tracking precision (MOTP) [16] as metrics. We can see a twofold difference in speed between SORT and Deep SORT algorithms across all three devices we used. It is worth noting that while Raspberry Pi 4 is still farm from realtime speed and the dedicated server surpasses this speed for both algorithms, Nvidia Jetson TX2 is on the verge of realtime performance. When talking about MOTA we see a twofold

difference in performance, which is not visible regarding the MOTP metric, where the results are tighter.

Device	Algorithm	FPS	MOTA	MOTP
Raspberry Pi 4	SORT	2.89	29.8	65.4
	Deep SORT	1.23	57.4	77.5
Nvidia Jetson TX2	SORT	24.78	29.8	65.4
	Deep SORT	11.69	57.4	77.5
Dedicated server	SORT	67.45	29.8	65.4
	Deep SORT	35.48	57.4	77.5

Table 2 – Results for object tracking

5. Conclusions

The system that we described in this paper is meant to be used in the real world in shopping malls, concerts and other crowded events, in order to extract relevant information regarding people. It is an ongoing discussion regarding the pros and cons of using server processing, edge processing or hybrid processing. The results presented in the previous section show that all approaches are possible from a technical point of view, with the afferent trade offs regarding performance and speed, flexibility, size and price. Also, we can see a great evolution of edge devices regarding computational power, which encourages us to consider the edge processing approach in the future, when such devices will be able to run deep models in realtime or near realtime.

Acknowledgements

This research was funded by the MARKSENSE project "Real-time Analysis Platform For Persons Flows Based on Artificial Intelligence Algorithms and Intelligent Information Processing for Business and Government Environment", contract no. 124/13.10.2017, MySMIS 2014 code 119261.

References

[1] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[2] Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[3] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.

[4] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.

[5] He, Kaiming, et al. "Mask r-cnn." Proceedings of the IEEE international conference on computer vision. 2017.
[6] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[7] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
[8] Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement." arXiv preprint arXiv:1804.02767 (2018).
[9] Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao. "YOLOv4: Optimal Speed and Accuracy of Object Detection." arXiv preprint arXiv:2004.10934 (2020).
[10] Huang, Rachel, Jonathan Pedoeem, and Cuixian Chen. "YOLO-LITE: a real-time object detection algorithm optimized for non-GPU computers." 2018 IEEE International Conference on Big Data (Big Data). IEEE, 2018.
[11] Bewley, Alex, et al. "Simple online and realtime tracking." 2016 IEEE International Conference on Image Processing (ICIP). IEEE, 2016.
[12] Wojke, Nicolai, Alex Bewley, and Dietrich Paulus. "Simple online and realtime tracking with a deep association metric." 2017 IEEE international conference on image processing (ICIP). IEEE, 2017.
[13] Zheng, Liang, et al. "Mars: A video benchmark for large-scale person re-identification." European Conference on Computer Vision. Springer, Cham, 2016.
[14] Cojocea, Eduard, Stefan Hornea, and Traian Rebedea. "Balancing between centralized vs. edge processing in IoT platforms with applicability in advanced people flow analysis." 2019 18th RoEduNet Conference: Networking in Education and Research (RoEduNet). IEEE, 2019.
[15] Multiple Object Tracking Benchmark MOT17 homepage, https:// motchallenge.net/data/MOT17/ , last accessed 2020/05/30
[16] Bernardin, Keni, and Rainer Stiefelhagen. "Evaluating multiple object tracking performance: the CLEAR MOT metrics." EURASIP Journal on Image and Video Processing 2008 (2008): 1-10.