DOI 10.1590/s1982-21702021000200013

ORIGINAL ARTICLE

FINE-TUNING DEEP LEARNING MODELS FOR PEDESTRIAN DETECTION

Fine-tuning de modelos de aprendizado profundo para detecção de pedestres

Caisse Amisse[1,2] - ORCID: 0000-0001-9458-5510

Mario Ernesto Jijón-Palma[1] - ORCID: 0000-0003-4890-2997

Jorge Antonio Silva Centeno[1] - ORCID: 0000-0002-2669-7147

[1] Universidade Federal do Paraná, Programa de Pós-graduação em Ciências Geodésicas, Curitiba - Paraná, Brasil.
E-mails: caamisse@gmail.com; majijpa@hotmail.com; centeno@ufpr.br

[2] Universidade Rovuma, Departamento de Ciências Naturais, Nampula, Moçambique.

Received in 28th Mars 2020.
Accepted in 26th Mars 2021.

Abstract:

Object detection in high resolution images is a new challenge that the remote sensing community is facing thanks to introduction of unmanned aerial vehicles and monitoring cameras. One of the interests is to detect and trace persons in the images. Different from general objects, pedestrians can have different poses and are undergoing constant morphological changes while moving, this task needs an intelligent solution. Fine-tuning has woken up great interest among researchers due to its relevance for retraining convolutional networks for many and interesting applications. For object classification, detection, and segmentation fine-tuned models have shown state-of-the-art performance. In the present work, we evaluate the performance of fine-tuned models with a variation of training data by comparing Faster Region-based Convolutional Neural Network (Faster R-CNN) Inception v2, Single Shot MultiBox Detector (SSD) Inception v2, and SSD Mobilenet v2. To achieve the goal, the effect of varying training data on performance metrics such as accuracy, precision, F1-score, and recall are taken into account. After testing the detectors, it was identified that the precision and recall are more sensitive on the variation of the amount of training data. Under five variation of the amount of training data, we observe that the proportion of 60%-80% consistently achieve highly comparable performance, whereas in all variation of training data Faster R-CNN Inception v2 outperforms SSD Inception v2 and SSD Mobilenet v2 in evaluated metrics, but the SSD converges relatively quickly during the training phase. Overall, partitioning 80% of total data for fine-tuning trained models produces efficient detectors even with only 700 data samples.

Keywords: fine-tuning; pedestrian detection; training data; deep learning models.

How to cite this article: AMISSE, C.; JIJÓN-PALMA, M.E.; CENTENO, J.A.S. Fine-tuning deep learning models for pedestrian detection. *Bulletin of Geodetic Sciences*. 27(2): e2021013, 2021.

1. Introduction

The availability of a large amount of image sequences obtained using security cameras of video cameras installed on unmanned aerial vehicles (drones), or low cost imaging sensors such as smartphones opened a large series of new application to the remote sensing community. Among them, there is the possibility to detect and track persons in urban scenes for security purposes. Pedestrian detection in video sequences is a challenging problem because the appearance of the pedestrian changes from image to image along the scene. A flexible model and high computation effort are necessary to perform this task with accuracy. One solution to this problem is the use of artificial intelligence techniques, like convolutional neural networks (CNN) that have the disadvantage of requiring a large amount of training samples and computational effort, generally using a Graphical Processing Units (GPUs) to speed up the process, to achieve the desired performance. Gathering and labelling large datasets and training a network for specific tasks is therefore impractical and time-consuming.

Due to the generalization capability of CNN features across different datasets, the transfer learning approach has shown state-of-the-art performance (Huh et al., 2016) for tasks such as object classification, object detection, and segmentation. For instance, the work of Zeng et al. (2014) proposes a transfer learning method for pedestrian detection that combines the ability to extract features from CNN and the similar ability to transfer features from the auto-encoder network. Masita et al. (2018) use R-CNN via transfer learning to detect pedestrians. Initially, they fine-tune a pre-trained Alexnet CNN in the Penn-Fudan (170 images) and KTH Multiview Football (771 images) datasets. Experimental results show 52% accuracy in the Penn-Fudan dataset and 84% in the KTH Multiview Football dataset. Wei and Kehtarnavaz (2019) used a deep learning framework of the semi-supervised convolutional neural network (SF-RCNN) with ResNet50 and GoogleNet networks to detect people and classify their load in video data through transfer learning. In the transfer learning technique used, a pre-trained ResNet50 model was retrained to detect people. In the classification phase, transfer learning CNN GoogleNet is used to distinguish between situations involving a detected person carrying a package or carrying a long arm.

In recent years, there has been considerable progress in boosting the performance of deep convolutional neural networks (CNN) with the growing availability of better performing model architectures. Commonly, researchers release their final model checkpoints to aid other researchers who can adapt (fine-tune) these results to new problems. Instead of training a CNN from scratch on a small dataset, it is possible to do a special kind of training called transfer learning (Torrey and Shavlik, 2010). Transfer learning takes advantage of pre-trained networks on a huge dataset, which is then optimized and reused on a small, but correlated, dataset from a different domain (Sharif Razavian et al., 2014). A large volume of annotated datasets for training such as the ImageNet database (Deng et al., 2009), Pascal VOC (Everingham et al., 2010), the Common Objects in Context (COCO) dataset (Lin et al., 2014) and the Open Images dataset are readily available for public use. Since all layers (except the output layer) can use pre-trained parameters during transfer learning, training time drops drastically while the algorithm's generalizability is improved (Guo et al., 2016).

The pre-trained models can be used as a feature extractor or as a basis for fine-tuning (Sharif Razavian et al., 2014). To use the pre-trained model as a feature extractor one should remove the fully connected layer of a pre-trained network and preserve the rest of the network. The removed layer is replaced by a machine learning classifier such as random forests, support vector machines or a new CNN, thus enabling to train the new classifier to a specific dataset.

The strategy for fine-tuning from a pre-trained model involves updating the weights of the pre-trained model by continuing the backpropagation algorithm. Fine-tuning can be applied to all the layers of the network or it is possible to keep some of the earlier layers (also called frozen layers) and only fine-tune some higher-level layers. This can be done because the features of the initial layers are generic and therefore transferable to a variety of specific datasets and tasks, while later features become progressively more specific to a particular problem (Yosinski et al., 2014).

The performance of transfer learning within deep learning architectures varies due to the difference in the used datasets, the amount of data, and the computational resources. A typical procedure for the practical application of CNN pre-trained models for object detection via transfer learning is to adapt them using a new small dataset. Setting the amount of data needed to build a robust object detection model is an important issue. Huh et al. (2016) explore relative importance of training samples in transfer learning starting from the following question: how does the amount of pre-training data affect transfer performance? Shin et al. (2016) compared various deep network architectures, dataset characteristics and training procedures for computer tomography-based abnormality detection to explore how the performance of a CNN changes according to architecture, dataset characteristics, and transfer learning. They considered five deep CNNs, namely AlexNet, CifarNet, GoogLeNet, OverFeat, and VGG-16, which achieved state-of-the-art performance in various computer vision applications. They observed that choice of architecture, parameter setting, and model fine-tuning needed is a dataset-specific problem. So, the relationship between the amounts of training samples required to train the object detection model via transfer learning to obtain a robust detector can still be explored. In this paper, it is presented the result of a study aimed at evaluating the use of fine-tuning convolutional networks to detect pedestrians in image frames of video sequences. We discuss this issue using a limited dataset and test the data on the pre-trained network models - Faster R-CNN Inception v2, SSD Inception v2, and SSD Mobilenet v2. Among the questions that are discussed based on the experiments using the three different network models are: the size of the necessary training set to fine-tune the models; the necessary iterations to achieve a stable quality in the training process and the quality of the model when applied to a test set.

2. Related Works

Ever since deep learning emerged, it has been successfully applied to various real-world problems such as, image classification (Wei and Kehtarnavaz, 2019), image segmentation (Luo et al., 2017), object detection (Hu et al., 2017; Zhao et al., 2019), and tracking (Jiang and Huynh, 2017). For the pedestrian detection task, deep learning-based methods have achieved great success (Tian et al., 2015; Hu et al., 2017; Zhao et al., 2019; Xie et al., 2020).

Deep learning-based pedestrian methods either employ the single-stage (Liu et al., 2016; Redmon et al., 2016) or two-stage (Ren et al., 2015) strategy as their backbone architecture. Two-stage detectors, such as Fast-RCNN (Girshick, 2015) and Faster R-CNN (Ren et al., 2015), first generates a set of object candidate locations through a Region Proposal Network (RPN) on the feature map, and then classifies each candidate as foreground or background while jointly performing localization regression. On the other hand, the one-stage detectors such as the SSD (Liu et al., 2016) or You Only Look Once (YOLO) family (Redmon et al., 2016) directly predict the output without region proposal stage. In Huang et al. (2017) is presented a comprehensive review that compares the trade-off between accuracy and speed among different deep learning-based object detectors, which demonstrates an overall conclusion that two-stage detectors achieve higher accuracy while one-stage detectors perform better in speed.

The two-stage detectors proposed by Zhao et al. (2017) detects pedestrian using Fast R-CNN. Recent trends on two-stage detectors focus on developing end-to-end approaches by using customized architectures or adapting Fast-RCNN or Faster-RCNN to detect pedestrians. For instance, Scale Aware Fast R-CNN (SA-Fast-RCNN) (Li et al., 2017) extends Fast-RCNN with multiple built-in sub-networks to adaptively detect pedestrians of different scales. Zhang et al. (2016) presented RPN (Region Proposal Network) + BF (Boosted Forest) framework that questioned the effect of the Faster R-CNN algorithm (Ren et al., 2017) for detecting pedestrians. Since then, much effort has been made on improving the performance of pedestrian detection (Zhang et al., 2017; Pang et al., 2019; Zhai et al., 2020), and some others (Tian et al., 2015; Zhou and Yuan, 2018; Pang et al., 2019; Xie et al., 2020) target at handling occlusion. The work of Tian et al. (2015), Masita et al. (2018), Wei and Kehtarnavaz (2019), and Zhang et al. (2020) proposed to detect pedestrians by addressing the fine-tuning problem, while Jiang et al. (2016) and Zhai et al. (2020) focus on speed up two-stage pedestrian detection algorithm. Since Fast R-CNN and Faster R-CNN have

limited success for detecting small pedestrians due to the low resolution of their convolutional features (Zhang et al., 2016), Hu et al. (2017) introduced semantic labels to improve pedestrians detections.

For the speed issues, researchers came up with one-stage detectors that sped up the detection process for pedestrian detection. It is the case of works of Du et al. (2017), Liu et al. (2018) or Bai et al. (2019). Du et al. (2017) exploit multiple neural networks in parallel for further refinement of pedestrian candidates obtained by the SSD. Liu et al. (2018) propose ALF (Asymptotic Localization Fitting) based on SSD to detect pedestrians. It stacks together multiple predictors to learn a better detection from default anchor boxes. Bai et al. (2019) propose a single shot proposal relation based approach for pedestrian detection. To boost the performance of one-stage detectors for pedestrian detection, Jin et al. (2020) implemented auxiliary detection head (ADH) module. Before ADH module, Du et al. (2017) and Mao et al. (2017) came up with idea of improving the pedestrian detection performance by combining semantic information. More current state-of-the-art pedestrian detectors are fine-tuned from pretrained classification networks (Shen et al., 2017; Zhu et al., 2019) focused on training a one-stage object detector from scratch.

3. Methodology

Figure 1 illustrates the key steps of the proposed object detection method. Three main elements compose the process, these are: image dataset splitting, deep model transfer learning, and detector performance evaluation. In this paper, we focus on the detection of pedestrians in image frames from video sequences of urban scenes. To perform the experiments, natural scenes were recorded using a mobile phone with a 4MP camera (16:9) from a window located on the fifth floor to obtain oblique views and others were obtained at the ground level. To simulate a more general situation, the camera was not fixed, so that the exterior orientation of the device changed along the recording time. The videos were then splitted into image frames using OpenCV to compose the dataset. To reduce the number of samples used for training, not all frames in the video are used but only one frame from every ten frames was taken for training. The image database comprises 700 1080w x 1920h RGB images. The images show objects such as vehicles and pedestrians in multiple locations in complex situations, including reduced illumination. The size of the region occupied by pedestrian in the images varies around 48x129 pixels, depending on the pose of the pedestrian and the distance to the camera. Pedestrian were labelled in the images, including those who were walking or standing, and if they were 75% visible in the image.

To assess the influence of the size of the training set and select the minimum number of images that should be used as training set, the dataset was splitted into training and testing subsets. The size of the training set varied from 50%, 60%, 70%, 80% and 90% of the total dataset. Three deep models were evaluated: The Faster R-CNN Inception v2, the SSD Inception v2 and the SSD Mobilenet v2. The detectors were retrained and tested in our dataset and using a public domain dataset provided by the Cityscapes project (Cordts et al., 2015).

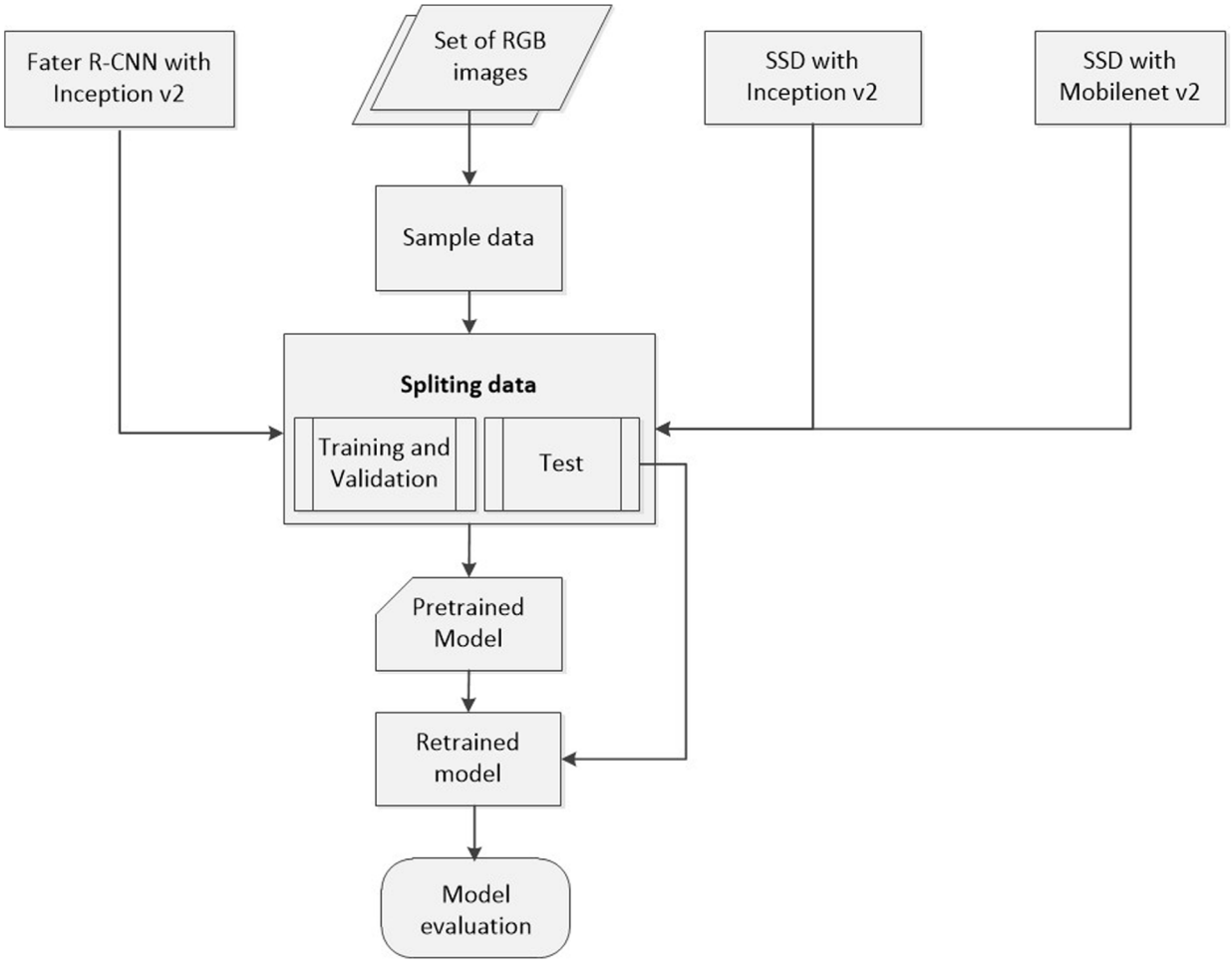

Figure 1. Flowchart of the experiment.

A description of the models used is presented as follows.

Faster R-CNN with Inception v2 backbone: mainly Faster R-CNN has two stages, first stage produces region anchors (regions having high probability about occurrence of the object (pedestrian)) via RPN. The next stage classifies object (pedestrian) using detected regions and extracts bounding box information. RPN takes a set of images (of any size) as input, represents them as multidimensional arrays (tensors) that, when passed through pre-trained CNN, create the convolutional feature map. In Faster R-CNN Inception v2, the Inception v2 (Szegedy et al., 2015) based model is used as backbone. Inception v2 is an enhanced version of the original version Inception v1 (Szegedy et al., 2015), where the 5x5 convolution layer is replaced by two 3x3 layers. Finally, the 3x3 convolutions are split into 1x3 and 3x1 and then made wider to remove the computational bottlenecks (Szegedy et al., 2015). There are other versions of inceptions: Inception v3 and v4 (Szegedy et al., 2016). Based on convolutional feature maps (exacted by Inception v2 backbone), the RPN generates candidate windows, i.e., region of interest (ROI). For each proposed region, ROI pooling is applied to convolutional feature maps to get a fixed-length feature vector used as input to the classification network. The fixed-length feature vector of each ROI generated from the pooling ROI layer is fed into two fully connected layers, referred to as fc6 and fc7 in Figure 2b. One of the output layers produces estimation softmax probability on object classes and background (classification). The other layer generates four outputs of four real-valued numbers (x, y, w, h) for each object class, encoding the bounding box positions refined for each class (regression). To estimate the loss for regression and classification, loss function L is computed as (Ren et al.,2015):

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (1)$$

Where $\{p_i\}$ and $\{t_i\}$ forms each, the output of the classification and regression layers; L_{cls} and L_{reg} are loss functions of the softmax classifier and regression of the bounding box, respectively. The output is obtained through the normalization of N_{cls} and N_{reg} via weighting by a balance parameter λ.

The regression loss $L_{reg}(t_i, t_i^*)$ is activated only if the anchor contains an object (i.e., the ground truth p_i^* is 1). The term t_i is the output prediction of the regression layer and consists of four variables $[t_x, t_y, t_w, t_h]$. The regression target t_i^* and learned regression output t_i are calculated as (Girshick et al., 2014):

$$t_x = \frac{(x-x_a)}{w_a} \quad t_y = \frac{(y-y_a)}{h_a} \quad t_w = \log(\frac{w}{w_a}) \quad t_h = \log(\frac{h}{h_a})$$
$$t_x^* = \frac{(x^*-x_a)}{w_a} \quad t_y^* = \frac{(y^*-y_a)}{h_a} \quad t_w^* = \log(\frac{w^*}{w_a}) \quad t_h^* = \log(\frac{h^*}{h_a}) \tag{2}$$

here (x, y, w, h), (x_a, y_a, w_a, h_a) and (x^*, y^*, w^*, h^*) denote the predicted box, proposal box and ground-truth box, respectively. To determine how the anchor regions agree with the ground truth bounding boxes, the intersection of the union (IoU) method is used. The detection d is considered as true positive whenever its IoU with the closest ground truth g is greater than 0.5 (Ren et al., 2015).

$$IoU(d,g) = \frac{|d \cap g|}{|d \cup g|} \begin{cases} \geq 0.5 = pedestrian \\ < 0.5 = background \end{cases} \tag{3}$$

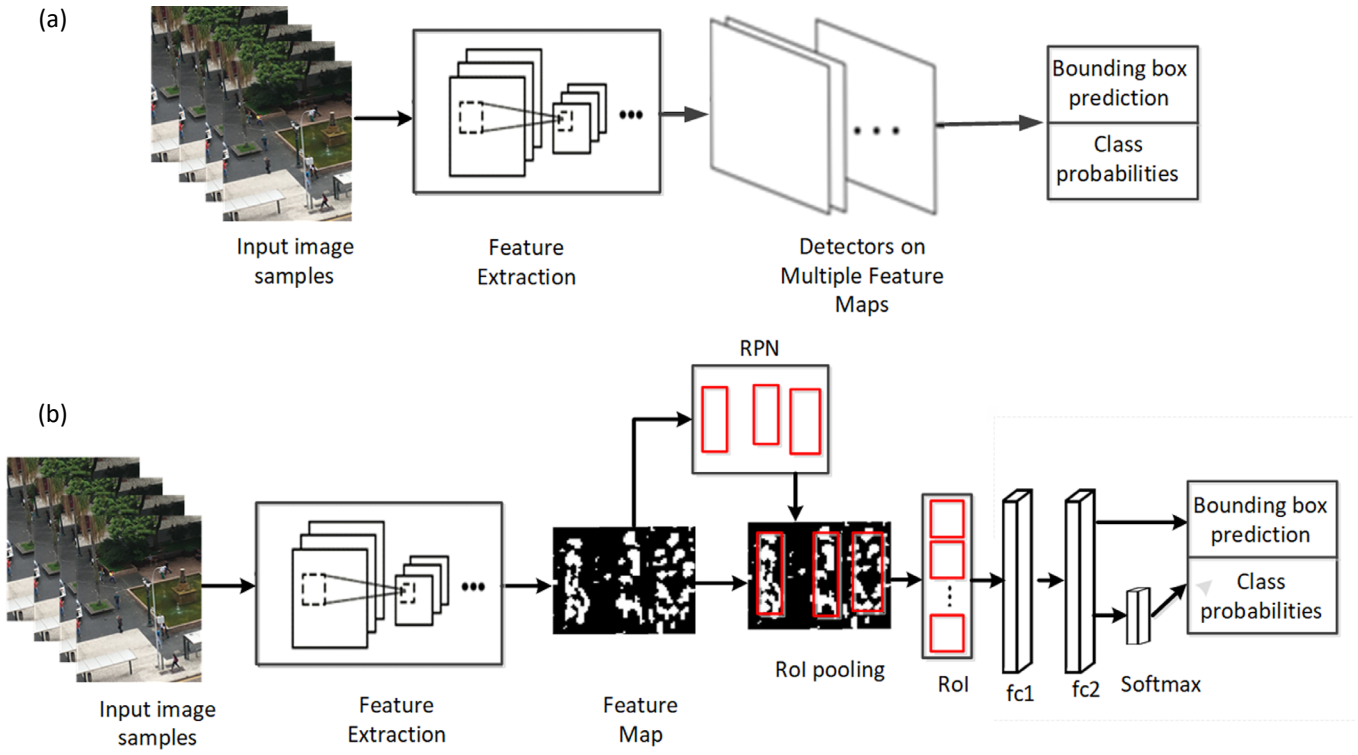

Figure 2: Schematic representation of the architecture of (a) SSD and (b) Faster R-CNN detectors.

SSD with Inception v2 backbone: the SSD detector is an object detection algorithm proposed by Liu et al. (2016) that solves the problem through a single pass in the image of a feedforward convolutional network. Here, an input image is passed through a series of convolutional layers and downsampled via the SSD base network layer, as displayed in Figure 2a. The convolutional layer produces a set of feature maps at different scales and various bounding boxes with object class scores in boxes. The variable size of the feature maps produces boxes with different dimensions, which in turn, allows detecting objects of different sizes. Ning et al. (2017) proposed the detection of objects using Inception Single Shot MultiBox Detector (I-SSD), which improved the SSD algorithm, i.e., increasing

its classification accuracy without affecting its speed. The purpose in I-SSD is to reduce the data representation or bottleneck, i.e., the use of intelligent factorization methods and convolution operations can make it more efficient in terms of computational complexity (Rosebrock, 2017). Similar to Faster R-CNN model, the SSD estimate the loss for regression and classification. Its loss function L is computed as a combination of two losses namely: the confidence loss L_{conf} which is dependent on the confidence rate and the localization L_{loc} representing the network's performance on estimating the bounding boxes. Both losses are weighted and added, such in the (Liu, 2016):

$$L(x, c, l, g) = \frac{1}{N}\left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right) \quad (4)$$

here, N is number of default boxes, L is localization loss, α is a weight to balance two losses and set to 1 in all experiments, c is offset for center, l is predicted box and g is the ground truth box parameters.

SSD with MobileNet v2 backbone: similar to SSD Inception v2, SSD-MobileNet v2 is a one-stage object detector (Figure 2a) but is employed for reducing the size and complexity of the model. The model uses a multiple feature map along a single network. In order to increase speed and eliminate proposed regions, the network can use this information to predict both, very large objects through deeper layers, as well as to predict very small targets by means of shallow layers. The MobileNet backbone has comparatively simple architecture consisting of a 3×3 depthwise convolution followed by a 1×1 pointwise convolution. This, make SSD-MobileNet valuable for applications on mobile and embedded devices (Liu et al., 2020).

3.1 Experimental Setup

In the training step, we applied transfer learning using the pre-trained models provided by the TensorFlow API (Huang et al., 2016). TensorFlow API, is a toolbox that was designed for simple, flexible, and easy use of CNN building blocks. It provides premade models (model zoo) combining some of the state-of-the-art detection algorithms with many of the main CNN backbone models. The provided models are trained on the MS COCO dataset with 90 classes. TensorFlow API also provides detailed tutorials to retrain models for new categories using transfer learning. To use the detectors for our task we set a number of classes to one and the number of training steps to 5000. All images were randomly divided into different proportions: 50%, 60%, 70%, 80% and 90% for a total of sample data, referred respectively as experiment 1 (e1), experiment 2 (e2), experiment 3 (e3), experiment 4 (e4) and experiment 5 (e5). The performance of detectors models is reported by accuracy, precision, mean average precision (mAP), F1-score and recall. The recall (r), precision (p) and F1-score metrics (Goutte and Gaussier, 2005) are computed as:

$$r = \frac{TP}{TP + FN} \quad (5)$$

$$p = \frac{TP}{TP + FP} \quad (6)$$

$$F1 - score = \frac{2 * r * p}{r + p} \quad (7)$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (8)$$

TP (true positive) indicates a detection where the IoU is greater than 0.5;

FN (false negative) is a pedestrian that is not detected at all (omission errors);

FP (false positive) when a pedestrian is completely incorrectly labeled (commission errors).

The mean average precision metric is calculated as the mean of average precision of the object class.

The experiments were implemented on a computational server with the following hardware specifications: Intel (R) Core (TM) Processor i7-7500U CPU @ 2.70GHz 2.90GHz, Installed Memory (RAM) 8.00 GB 64-bit operating system type, x64-based processor and NVIDIA GeoForce 940MX graphics processing unit.

4. Experimental Results and Analyses

On the basis of the dataset redistribution, all models are tested, evaluated, and their performance assessed. Table 1 depicts the comparison of detection metrics from tested models. It can be seen from this table that the Faster R-CNN Inception v2 model offers the maximum performance in pedestrian detection, namely high precision, recall, accuracy and F1-score in almost all experiments. Although the performance of the detector improves with increasing number of training samples (e1-e4), adding much more training samples for training (e5) decrease the performance of almost all tested models. For example, the recall metric increases from e1 to e4, and then decrease in e5. Yet when taking a look at the F1-score metric, the scenario e4 outperforms the rest of scenarios. In this case, the SSD Inception v2 substantially outperforms SSD Mobilenet v2 and achieves a difference of 6.6%. It can also be observed that this difference increases progressively from e1 to e3. The trend of the accuracy, precision, recall, and F1-score values are similar for tested models.

Table 1: Comparison of detection metrics from Faster R-CNN Inception v2, SSD Inception v2 and Mobilenet V2.

Experiment	Model	Prec (%)	Recall (%)	Acc (%)	F1-score
e1 (50%)	Faster R-CNN Inception v2	82.5	76.4	54	78.2
	SSD Inception v2	70.7	64	44	68.2
	SSD Mobilenet v2	62.1	59.1	40.1	63.1
e2 (60%)	Faster R-CNN Inception v2	86.2	82.5	66	84.8
	SSD Inception v2	81.5	78.5	61	80.8
	SSD Mobilenet v2	76	63.0	58.7	71.4
e3 (70%)	Faster R-CNN Inception v2	86.0	82.3	77	84.6
	SSD Inception v2	78.03	76.3	67.7	78.26
	SSD Mobilenet v2	85.4	57.8	65.3	66.4
e4 (80%)	Faster R-CNN Inception v2	96.6	92.6	84	94.6
	SSD Inception v2	92	85.6	67.7	86
	SSD Mobilenet v2	84.5	58.7	78.5	79.4
e5 (90%)	Faster R-CNN Inception v2	88.8	84.3	81	86.2
	SSD Inception v2	90.1	83.2	78.1	84.2
	SSD Mobilenet v2	73.5	56.9	64.3	74.8

For visualization purposes, we plot the accuracy metric computed for the different number of training samples (Figure 3). The accuracy tended consistently to increase when the training data was increased. For example, increasing the number from 350 to 420 samples increased the accuracy, in average, 21.5% for Faster R-CNN inception v2, 30% for SSD inception v2, and 28.4% for SSD Mobilenet v2. The results agree with the studies of Sa et al. (2016) and Huh et al. (2016). Sa et al. (2016) found out that the performance (precision, recall, accuracy, and F1-score) can be increased varying the number of training samples (i.e., 43, 45, 51, 100, 109, 122, and 136) for retraining Faster R-CNN with VGGNet via transfer learning. They achieved, respectively, F1-scores of 0.848, 0.948, 0.938, 0.932, 0.942, 0.915 and 0.828. Like reported by Huh et al. (2016), for classification tasks, the increase of the amount of training

data conduce to increase the performance on transfer learning. This tendency maintains until 80% of the samples are used. In all experiments, SSD Mobilenet v2 had the lowest accuracy. When 90% of the samples were used, the accuracy decreases for all tested network models. The drop is almost equal in Faster R-CNN Inception v2 and SSD Inception v2, and lower than the one registered when SSD Mobilinet v2 is applied.

The comparative study suggest that the best results can be obtained using 80% of the samples, i.e., 560 samples, for training. In this case, the mAP is 96.6%, 90.1%, and 78% for Faster R-CNN Inception v2 SSD Inception v2, and SSD Mobilenet v2 respectively. Similarly, Sa et al. (2016) found that fine-tuning model using a small dataset generalizes well when allocating 80% of total data for training.

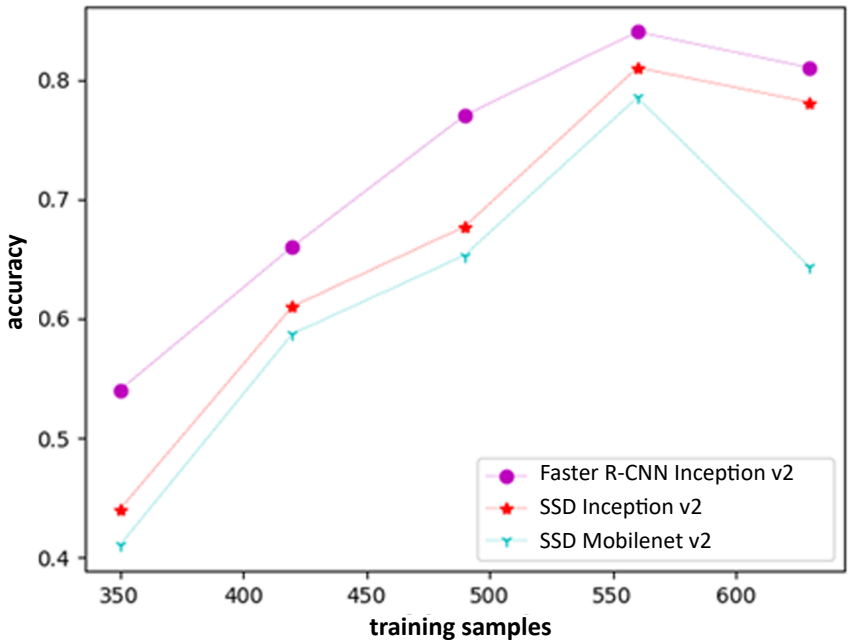

Figure 3: Accuracy versus number of training instances on Faster R-CNN Inception v2, SSD Inception v2 and SSD Mobilenet v2.

The graphs in Figure 4 show the evolution of the mAP over the 5000 steps of the iterative training process. The Precision of the Faster R-CNN Inception v2 model has near logarithmic growth and reaches a plateau after 2000 steps. At this point, the performance of the model slightly increases. The mAP of SSD Mobilenet v2 and SSD Inception v2 reaches respectively, a plateau after 1700 and 1500 steps confirming that comparing to Faster R-CNN, SSD models are faster (Xu, 2017). The mAP performance of all models increased fast initially (0.0 to 0.7), to later stagnate around 2000 to 1500 steps, as seen in Figures 4a, 4b, and 4c.

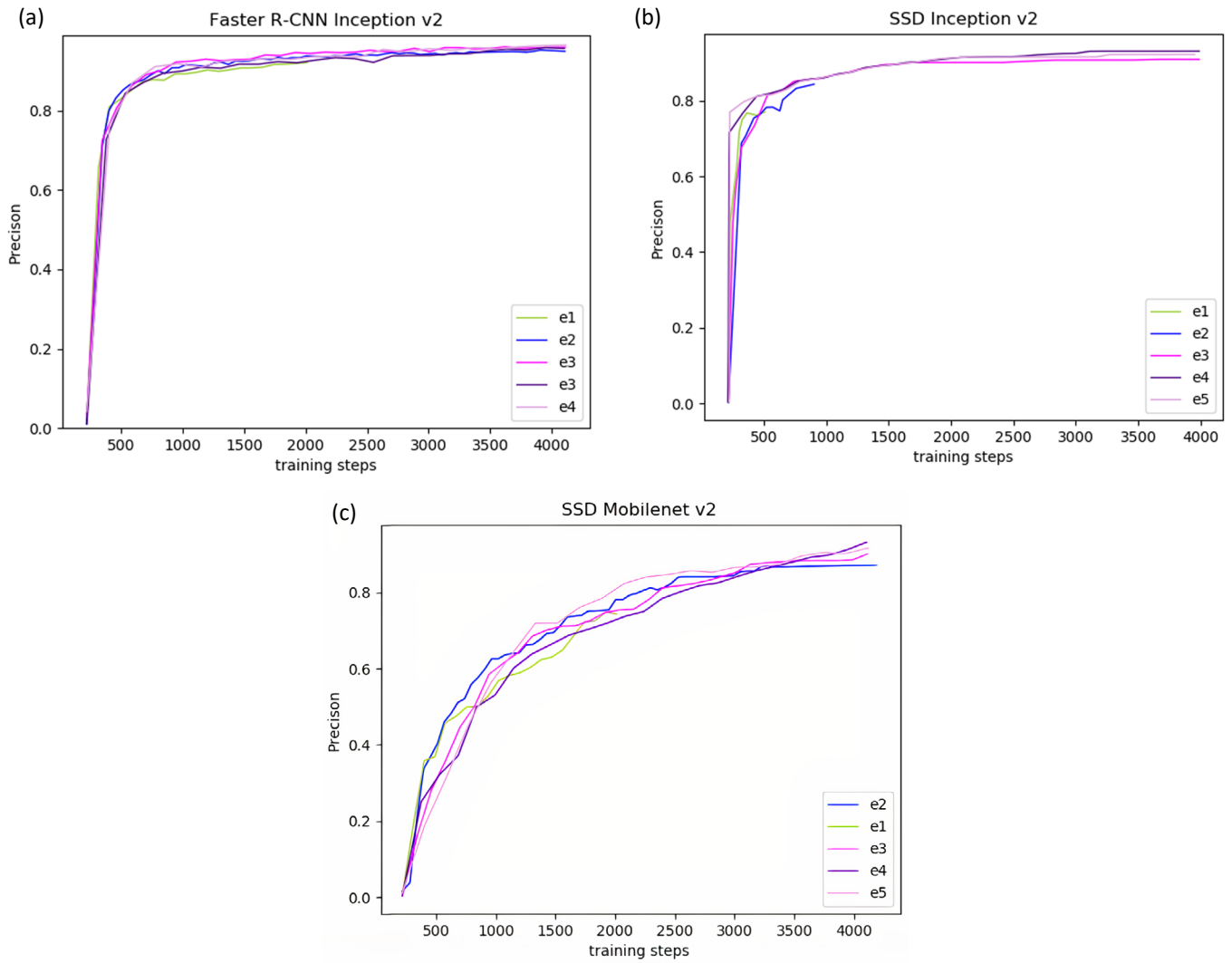

Figure 4: Development of overall mAP when fine-tuning (a) Faster R-CNN Inception-v2, (b) SSD Inception-v2, and (c) SSD Mobilenet v2.

As observed in Figure 4, throughout the training, the precision values for the SSD detector are outperformed by the corresponding values of the Faster R-CNN. Moreover, when split training data to a proportion of 80% (e4) and 90% (e5), the mAP surpasses the other configuration variations of training data. SSD Inception v2 model is very sensitive at a proportion of 50% as it has the worst precision (0.725) compared to the same scenario for the SSD Mobilenet v2 and Faster R-CNN Inception v2 models which reaches approximately and 0.80 and 0.85 respectively. For Faster R-CNN Inception v2 the mAP does not vary significantly within a different variation of training data.

Although Faster R-CNN Inception and SSD Inception detectors use the same base extraction network (Inception v2), in all cases of our experiment, a two-stage detector Faster R-CNN Inception v2 model reaches near-optimal performance, whereas the one stage detector SSD Inception v2 performs comparatively worse. As Zhao et al. (2019) has pointed out, two-stage detectors are more accurate than single-stage detectors.

After the initial comparative test, the three models were applied to the same test set. Figure 5 displays some detection results of experiments 1 and 4 that represents, respectively, the worst and best scenarios. The detectors successfully detect pedestrians with various scale instances and background variation.

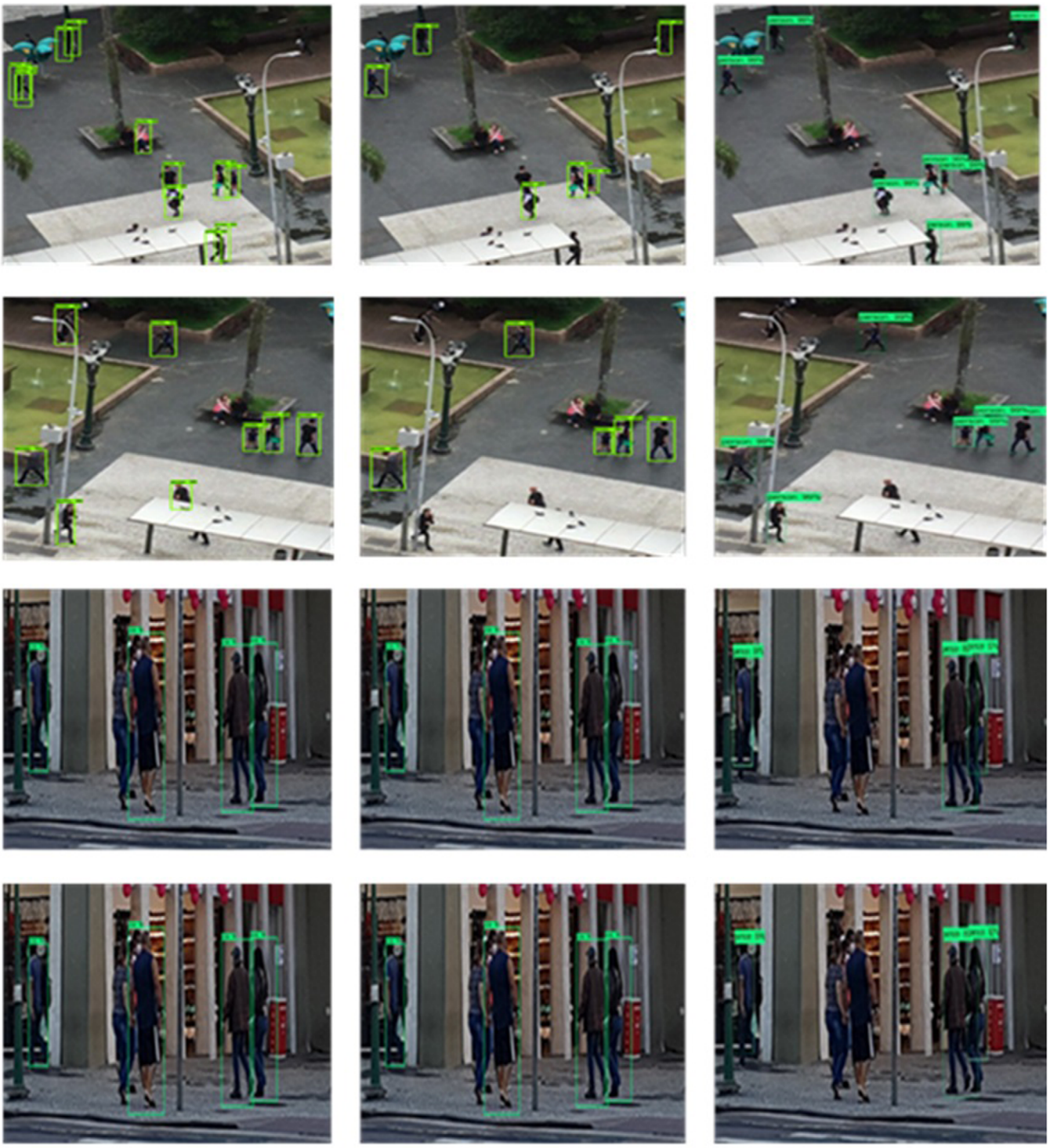

Figure 5: Examples of detections output in our dataset, for Faster R-CNN Inception v2 (column 1), SSD Inception v2 (column 2), and SSD Mobilenet v2 (column 3) systems.

The worst results are obtained with the reduced number of training samples, experiment e1. Here, the Faster RCNN Inception v2 model produced several false bounding boxes, while SSD with Inception v2 and Mobilenet v2 detected almost all pedestrians. In scenario e4, where the best accuracy was obtained for the training samples, the Faster R-CNN produces better results compared to the other models. In this scenario, SSD Inception v2 outperforms SSD Mobilenet v2.

The visual and qualitative results indicate a significant influence of the amount of training data on the detector's performance. Meanwhile the results also report the potential of transfer learning. It was also possible

to confirm that a good accuracy (over 85%) can be achieved with a small amount of training data. This information is interesting for the use of monitoring systems based on deep learning. Likewise, transfer learning can help avoid much expensive data-labeling efforts.

Figure 6: Comparisons on three challenging pedestrian detection scenarios, from the Cityscapes dataset, for Faster R-CNN Inception v2, SSD Inception v2, and SSD Mobilenet v2 models.

Finally, we applied the nets trained using 80% of the samples to detect pedestrians in three challenging scenarios, obtained from the Cityscapes dataset (Cordts et al., 2015). Examples of the images are displayed in Figure 6. Each row in Figure 6 displays images with different problems: varying scale, occlusion, and a crowed scene. The input images were all blurred to avoid face recognition. In all evaluated scenarios, SSD Mobilenet v2 output the worse detection results.

The experimental results illustrate that, although acceptable performance detection is achieved, errors still happen in all fine-tuned models. The ability of the fine-tuned models for object detection can be improved by a number of ways. For example, one can improve the architecture model to enhance its performance and generalization ability (Zhai et al., 2020), or by performing the models training and detection process on a cloud platform, which might raise the detection efficiency. Indeed, it is evident that the Faster R-CNN detects pedestrians with more accuracy, clearly with a greater processing time (i.e., more training steps to reach the plateau) than the SSD. This is because Faster R-CNN comprises two sequential stages, the first one to propose regions, followed by the second one that classifies the proposed regions. Conversely, the SSD have a lower processing time, which is a useful

advantage when designing applications in embedded systems, mainly because the model handle object detection as a regression problem.

5. Conclusion

We conducted sets of experiments to explore the performance of fine-tuning models with a variation in the number of training data for object detection on images, and compare state of art Convolutional Object Detection systems particularly, Faster R-CNN and SSD with Inception v2 base feature extractors and SSD Mobilenet v2. The set of experiments evaluate precision, accuracy, recall and F1-score of fine-tuned object detection models via Faster R-CNN Inception v2, SSD Inception v2 and SSD Mobilenet v2 systems with different amount of training samples. Qualitative and quantitative assessments indicate that SSD produces worse results compared to Faster R-CNN, but had quick convergence during the training indicating stronger learning ability of the detector.

In fine-tuning the detectors, the precision, accuracy, recall, and F1-score had the same trend: the increase in training data increases the accuracy of the detector but the allocation of 90% of total data (e5) for training degrades the values of accuracy, precision, F1-score, and recall. The optimal number of data allocation for training is likely to be around 80% (e4) of the training data.

The proper selection of the proportion of training data plays a significant contribution to successful fine-tuning models. If the proportion is too low (50%) or too high (90%) the models performance degrades, since de accuracy, precision, recall, and F1-score (Table 1) decrease and not all the pedestrians will be detected. Therefore, the use of at least 60%-80% total of images for training is suggested to provide safety margin for transfer learning, using Faster R-CNN Inception v2, SSD Inception v2 and SSD Mobilenet v2.

From the experiment, the fine-tuning of the Faster R-CNN Inception v2, SSD Inception v2 and SSD Mobilenet v2 detectors tends to generate a coherent increase in the values of accuracy, precision, recall, and F1-score when retrained in our small dataset (only 700 images, taking around 5000 training steps), this demonstrates the potential of transfer learning technique in practical applications.

Future work should test with a different number of classes and use fine-tuning with model topology adjustment by varying some model parameters such as IoU, graph, Batch size, Moment, Weight decay and anchors boxes.

ACKNOWLEDGEMENT

The authors would like to thank CAPES/CNpq for financial support during this study.

AUTHOR´S CONTRIBUTION

Caisse Amisse: designed and implemented the algorithm, conducted experimental study and wrote the paper; Mario Jijón-Palma: prepared the data and carried out manual labelling of images and Jorge António Silva Centeno: collected the data and revised the paper.

REFERENCES

Bai, J., Lu, Z., Liao, Q. 2019. Single shot relation detector for pedestrian detection. In *Eleventh International Conference on Digital Image Processing (ICDIP 2019)*, vol. 11179, p. 1117929.

Cordts, M., Omran, M., Ramos, S., Scharwächter, T., Enzweiler, M., Benenson, R.,...Schiele, B. 2015. The cityscapes dataset. In *CVPR Workshop on the Future of Datasets in Vision*, vol.2.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248-255.

Du, X., El-Khamy, M., Lee, J. Davis, L., 2017. Fused DNN: A deep neural network fusion approach to fast and robust pedestrian detection. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pp. 953-961.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., Zisserman, A. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2), 303-338.

Girshick, R., 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 1440-1448.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580-587.

Goutte, C., Gaussier, E. 2005. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In *European conference on information retrieval*, pp. 345-359. Springer, Berlin, Heidelberg.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., Lew, M. S. 2016. Deep learning for visual understanding: A review. *Neurocomputing*, 187, 27-48.

Hu, Q., Wang, P., Shen, C., van den Hengel, A., Porikli, F. 2017. Pushing the limits of deep cnns for pedestrian detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(6), 1358-1368.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., ...Murphy, K. 2016. Speed/accuracy trade-offs for modern convolutional object detectors. *arXiv preprint arXiv*:1611.10012.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S. and Murphy, K., 2017. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7310-7311.

Huh, M., Agrawal, P., Efros, A. A. 2016. What makes ImageNet good for transfer learning? *arXiv preprint arXiv*:1608.08614.

Jiang, X., Pang, Y., Li, X., Pan, J. 2016. Speed up deep neural network based pedestrian detection by sharing features across multi-scale models. *Neurocomputing*, 185, 163-170.

Jiang, Z., Huynh, D. Q. 2017. Multiple pedestrian tracking from monocular videos in an interacting multiple model framework. *IEEE transactions on image processing*, 27(3), 1361-1375.

Jin, G., Taniguchi, R. I., Qu, F. 2020. Auxiliary Detection Head for One-Stage Object Detection. *IEEE Access*, 8, 85740-85749.

Li, J., Liang, X., Shen, S., Xu, T., Feng, J. and Yan, S., 2017. Scale-aware fast R-CNN for pedestrian detection. *IEEE transactions on Multimedia*, 20(4), pp.985-996.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D.... Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740-755. Springer, Cham.

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X. Pietikäinen, M., 2020. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2), pp.261-318.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., Berg, A. C. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*, pp. 21-37. Springer, Cham.

Liu, W., Liao, S., Hu, W., Liang, X., Chen, X. 2018. Learning efficient single-stage pedestrian detectors by asymptotic localization fitting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 618-634.

Luo, P., Wang, G., Lin, L., Wang, X. 2017. Deep dual learning for semantic image segmentation. In *Proceedings of the IEEE international conference on computer vision*, pp. 2718-2726.

Mao, J., Xiao, T., Jiang, Y., Cao, Z. 2017. What can help pedestrian detection? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pp. 3127-3136.

Masita, K. L., Hasan, A. N., Paul, S. 2018. Pedestrian detection using R-CNN object detector. In *2018 IEEE Latin American Conference on Computational Intelligence (LA-CCI),* pp. 1-6.

Ning, C., Zhou, H., Song, Y., Tang, J., 2017. Inception single shot multibox detector for object detection. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pp. 549-554.

Pang, Y., Xie, J., Khan, M.H., Anwer, R.M., Khan, F.S. and Shao, L., 2019. Mask-guided attention network for occluded pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4967-4975.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 779-788.

Ren, S., He, K., Girshick, R., Sun, J. 2017. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6), 1137-1149.

Ren, S., He, K., Girshick, R., Sun, J. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91-99.

Rosebrock, A., 2017. Deep Learning for Computer Vision with Python: ImageNet bundle. *PyImageSearch*. New York, NY, USA.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., McCool, C. 2016. Deep fruits: a fruit detection system using deep neural networks. *Sensors,* v. 16, n. 8, p. 1222.

Sharif Razavian, A., Azizpour, H., Sullivan, J., Carlsson, S. 2014. CNN features off-the-shelf: An astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops,* pp. 806-813.

Shen, Z., Liu, Z., Li, J., Jiang, Y. G., Chen, Y., Xue, X. 2017. Dsod: Learning deeply supervised object detectors from scratch. In *Proceedings of the IEEE international conference on computer vision,* pp. 1919-1927.

Shin, H. C., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., ... Summers, R. M. 2016. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5), 1285-1298.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D. ... Rabinovich, A. 2015. Going deeper with convolutions. In P*roceedings of the IEEE conference on computer vision and pattern recognition,* pp. 1-9.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 2818-2826.

Tian, Y., Luo, P., Wang, X., Tang, X. 2015. Pedestrian detection aided by deep learning semantic tasks. In *Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 5079-5087.

Torrey, L., Shavlik, J. 2010. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques,* pp. 242-264.

Wei, H., Kehtarnavaz, N. 2019. Semi-supervised faster RCNN-based person detection and load classification for far field video surveillance. *Machine Learning and Knowledge Extraction*, 1(3), 756-767.

Xie, J., Cholakkal, H., Anwer, R. M., Khan, F. S., Pang, Y., Shao, L., Shah, M. 2020. Count-and similarity-aware R-CNN for pedestrian detection. In *European Conference on Computer Vision,* pp. 88-104. Springer, Cham.

Xu, J. 2017. Deep learning for object detection: A comprehensive review. *Towards Data Science*. Retrieved December 10, 2018.

Yosinski, J., Clune, J., Bengio, Y., Lipson, H. 2014. How transferable are features in deep neural networks? In *Advances in neural information processing systems,* pp. 3320-3328.

Zeng, X., Ouyang, W., Wang, M., Wang, X. 2014. Deep learning of scene-specific classifier for pedestrian detection. In *European Conference on Computer Vision,* pp. 472-487. Springer, Cham.

Zhai, S., Dong, S., Shang, D., Wang, S. 2020. An improved faster R-CNN pedestrian detection algorithm based on feature fusion and context analysis. *IEEE Access,* 8, 138117-138128.

Zhai, S., Shang, D., Wang, S., Dong, S. 2020. DF-SSD: An improved SSD object detection algorithm based on denseNet and feature fusion. *IEEE Access*, 8, 24344-24357.

Zhang, L., Lin, L., Liang, X. and He, K., 2016. Is faster R-CNN doing well for pedestrian detection? In *European conference on computer vision,* pp. 443-457. Springer, Cham.

Zhang, S., Benenson, R., Omran, M., Hosang, J., Schiele, B. 2017. Towards reaching human performance in pedestrian detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(4), 973-986.

Zhang, W., Wang, K., Liu, Y., Lu, Y., Wang, F. Y. 2020. A parallel vision approach to scene-specific pedestrian detection. *Neurocomputing*, 394, 114-126.

Zhao, Z. Q., Bian, H., Hu, D., Cheng, W., Glotin, H. 2017. Pedestrian detection based on fast R-CNN and batch normalization. In *International Conference on Intelligent Computing,* pp. 735-746. Springer, Cham.

Zhao, Z. Q., Zheng, P., Xu, S. T., Wu, X. 2019. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11), 3212-3232.

Zhou, C., Yuan, J. 2018. Bi-box regression for pedestrian detection and occlusion estimation. In *Proceedings of the European Conference on Computer Vision (ECCV),* pp. 135-151.

Zhu, R., Zhang, S., Wang, X., Wen, L., Shi, H., Bo, L., Mei, T. 2019. ScratchDet: Training single-shot object detectors from scratch. In *Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 2268-2277.