COMP9444 Report

Term 3 2022

Group Assignment

Yuyan Li	Z5270606
David Zhou	Z5157517
Joshua Rozario	Z5115700
Lachlan Ting	z5264855

Table of Contents

Introduction

Background

The applications of machine learning have become ubiquitous with the advancement of computing hardware and machine learning algorithms. In the recent few years, machine learning has matured significantly as a field with applications in many industries from Tesla's self-driving cars to Microsoft's GitHub Copilot. Although machine learning has grown as a whole, individual machine learning trends are constantly being replaced and improved by newer more efficient or accurate technologies based on the purpose of the application.
For instance, traditional machine learning requires manual feature extraction and relies on the machine-learning model to classify the inputs and outputs. This is different to deep learning which is a subset of machine learning which includes both feature extraction and classification.

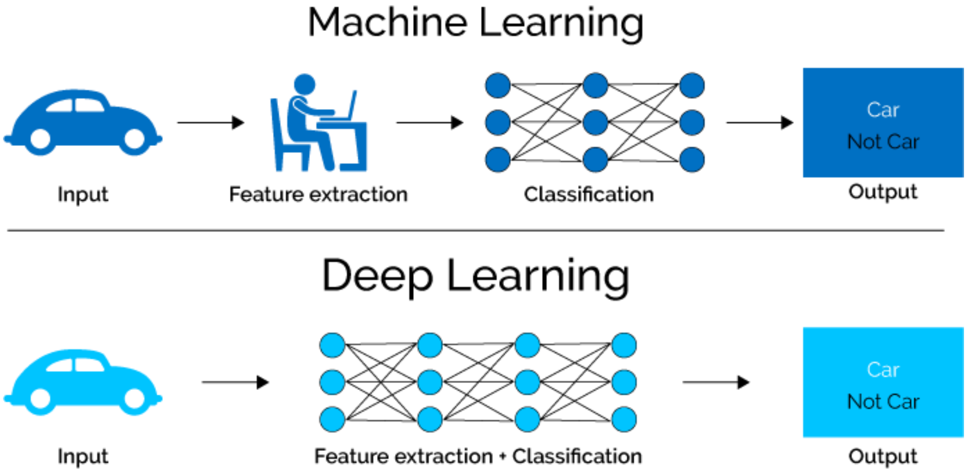


Figure 1: Traditional Machine Learning Verses Deep Learning [1]

An obvious advantage of this is the machine-learning model can pick up features that a human inputting the data may miss. However, this introduces additional risks as the model may optimise for the wrong thing. For instance, a model which is written to detect if there is a dog in the image may in fact be detected if the background is a beach or a backyard. These are just a few of the things one must account for when developing a robust machine-learning model.

Problem Statement

The objective of our project was to develop a model that can count and identify the number of people within a scene in order to help enforce social distancing. As there are many solutions to this traditional problem—the focus of this project is to measure and compare the performance of many bleeding-edge models with that of a simple CNN approach.

Literature Review

In the investigation stage, we looked at a variety of models. Including object detection [11], object counting [6], and more specifically people counting [12]. We concluded that a convolutional neural network (CNN) [5] and its variations are best suited for the task – thus eliminating networks such as BERT (which is a language model) and LSTMs. We then looked at a few models that are made for this task as well as experimented with basic CNN models this narrowed the options down to the following. These models include MaskRCNN [4], DETR-Resnet50 [3], and Faster R-CNN [7].

Models

Simple model

The simple models have 3 or 4 layers. The hidden layers are simple 2D convolutional layers [5], which are very common in image recognition neural networks. This model uses 50 outputs to find up to 50 bounding boxes as well as the model's confidence in the accuracy of each box.

DETR-Resnet50

The DETR Resnet 50 model [3] was selected as a solution as it exhibits speed, faster networks (due to the number of parameters) [8], and accuracy with other RCNN models without having the same complexity. It does not require additional libraries too which is a major plus!

The reason it is faster than even the "gold standard" Faster RCNN, is that it is trained using a set operation called bipartite matching loss which compares the ground truth to the predictions made by the model [3].

Faster R-CNN with MobileNet backbone

The Faster R-CNN model [7] was chosen as it was a model that could run object detection evaluations in a fast and speedy manner, while also providing decently accurate results.

The reason Faster R-CNN is faster than normal R-CNN is that you do not need to feed in multiple region proposals—1000's even, to the CNN every time. Instead, the Convolution operation is only done once per image and the feature map is generated from it. This means that Faster R-CNN is a single-stage process rather than a multiple-stage process. This lends it to be much faster and usually more accurate due to the ROI pooling being shared across the region proposal network and the CNN.

Experimental Setup

Dataset 1

The STEP-ICCV21 dataset [9] we used to train the models consists of 1125 images taken from 2 videos with fixed cameras and a large number of people walking past them. These image sets are both 1920 by 1080 in size. The two videos are of different times of the day so the model could have higher accuracy in different lighting conditions. One has 600 images with a low angle, and the other has 525 images with a wide angle. We chose this dataset so that the model could be trained to be accurate for images with differently angled shots and different lenses.

The dataset also provides the ground truths in images. They are panoptic maps of the original images, which use blue to fill the shapes of people with a distinct red border around them. However, this format has been developed with a specific model in mind. In order to make the dataset adapt to different models that we would like to test; the ground truths must be turned into a more flexible format.

First, we would like to find the bounding boxes of every person in each image. We did this by calculating the maximum and minimum pixel coordinates of the blue channel on all the ground truth images. Using a simple python script, we were able to render these bounding boxes on the original images.

Then we exported all the bounding box data using another simple python script to JSON files. We created these files for each of the images so we can use them to train different models with data that are portable and adaptive.

Dataset 2

The Penn-Fudan Dataset [10] was also used along with the STEP-ICCV21 dataset as there was some difficulty getting the first dataset to train on the Faster R-CNN model. This dataset albeit smaller was of a large enough size to train the model and receive decent results.

The 170 images in this dataset are around various locations at the University of Pennsylvania and Fudan University. Each shot was taken around the campus and the urban streets around the campus, The heights of labelled pedestrians in this database are within 180 by 390 pixels. All labelled pedestrians are straight up.

The dataset also contains the ground truths for each pedestrian. There is a mask and bounding box for each of the 345 pedestrians in the pictures. Since all this data is provided for us the only work required was to load the data into the `torch.utils.data.Dataset` class provided by PyTorch.

Pre-processing

A major risk of deep learning is self-classification as the model may optimise for incorrect targets, overfit the training set, or build an 'optimal' model with a biased dataset. To mitigate these risks, the important aspects of the image should be focused on.

Generally, the first step is to ensure that the scaling of the input data is equal for all the images which are verified on the training set.

The next step involves normalizing the images which means adjusting the distribution of the image such that the mean of the histogram is zero with a standard deviation of one. The chosen image set already has normalized images but needs to be renormalized after certain image-augmentation techniques.

Another commonly used technique is the rotations and flips but the only flip which makes sense is a horizontal flip as any other orientation would result in unrealistic data. Furthermore, a gaussian blur is also used on the image set to increase the robustness of the model at handling low-resolution images.

The final image augmentation technique used is random cut-outs which is essentially a black box that obstructs part of the image [2]. This technique can be further optimized by only covering part of the people in the image.

Results

Simple model

We encountered 3 major issues building this model:

1. It is difficult to input the images into the models.
   Since this is a simple model we created ourselves, there is no pre-existing dataset class we can use. And the dataset model that is built in PyTorch does not support image files. We solved this by turning the images into tensors.
2. The size of the data.
   The built-in dataset and models are made with image classification in mind. However, our goal is not to classify images but to find the number of people. Hence the nature of the dataset is varied. The simple model can only take datasets with a set number of inputs and does not allow inputs with varying lengths. We decided to try a different method of building the dataset. However, they encountered the same issue.
3. The simple model cannot output varied-length answers.
   To get around this we considered making it output a level of confidence in the person found so it can output all 50 boxes with some low-confidence ones filtered out. However, the threshold of this confidence level is difficult to determine. We then decided to use more advanced models to reach our goal.

These problems make it very difficult to perform this task on the simple model, we decided we should explore other models.

DETR-ResNet50

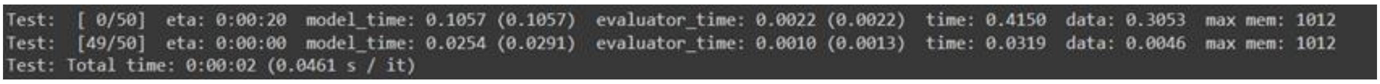
The DETR-ResNet50 can easily identify people in the images in a relatively short time using the pre-trained weights, however, there was the occurrence of some false positives as well as some missing ones. It should be noted that this model struggled to detect people hidden behind the foreground which is a shortcoming as a potential solution to deal with the problem statement.

The main issue with the DETR was converting the panoptic maps which were the ground truth into a format that could allow for the DETR to be trained. However, this was not possible and therefore we could not train the modal on the ground truth to pick up things that it was supposed to. The DETR modal expected a COCO format dataset which we attempted; however, this still did not work.

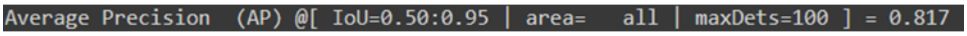
Faster R-CNN with MobileNet backbone

In the end, the results we received from this model were of sound standards, overall, the accuracy was 81.7% for the Penn-Fudan Image Dataset. Not only that but it only took 0.0461 seconds to run an evaluation on 50 images, which highlighted this model's specific strength of providing quick results while also being decently accurate.

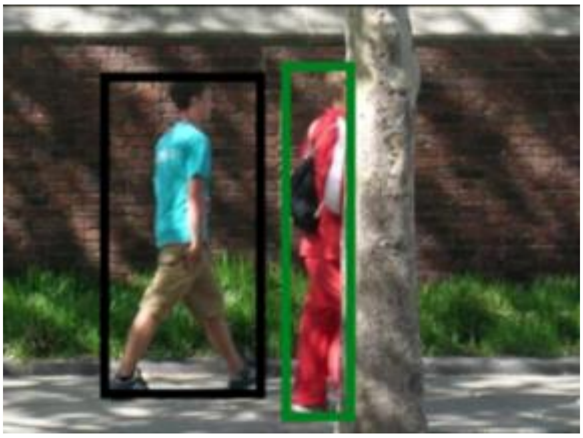
Overall, the model managed to pick out most of the pedestrians and did so, at such a speed that it would be reasonable to use this model for low-resolution videos that go up to 30fps.



```
Test:  [ 0/50]  eta: 0:00:20  model_time: 0.1057 (0.1057)  evaluator_time: 0.0022 (0.0022)  time: 0.4150  data: 0.3053  max mem: 1012
Test:  [49/50]  eta: 0:00:00  model_time: 0.0254 (0.0291)  evaluator_time: 0.0010 (0.0013)  time: 0.0319  data: 0.0046  max mem: 1012
Test: Total time: 0:00:02 (0.0461 s / it)
```



```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.817
```



Conclusions

Considering all the results we obtained we can say that while there were some troubles getting some models to train, we did receive results that point to the valid use of object detection models in order to count and detect the number of people within a scene and we were also able to obtain accuracy ratings of over 80%.

This means that with the use of such models a real-world system could be implemented in with the working model to count the number of people within a scene – and considering the models' evaluation speed it would be safe to say that this could be done on data that is in video format.

The augmented dataset is unable to be tested it is processed on dataset one, but the team was unable to create a working model with dataset one. As part of future work, this pre-processing will also need to be performed on dataset two which uses a different label format. However, based on the papers presented image augmentation should decrease the likelihood of overfitting and improve the robustness of the models.

References

[1] S. Mahapatra, “Why deep learning over traditional machine learning?”, 22-Mar-2018. [Online]. Available: https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063. [Accessed: 20-Nov-2022].

[2] DeVries, Terrance & Taylor, Graham. (2017). Improved Regularization of Convolutional Neural Networks with Cutout.

[3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-End Object Detection with Transformers,” 2020.

[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” 2017.

[5] PyTorch Contributors, “Conv2d¶,” *Conv2d - PyTorch 1.13 documentation*, 2022. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html#torch.nn.Conv2d. [Accessed: 20-Nov-2022].

[6] E. Kilic and S. Ozturk, “An accurate car counting in aerial images based on convolutional neural networks,” *Journal of Ambient Intelligence and Humanized Computing*, 2021.

[7] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” 2015.

[8] Goswami, S. (2020, November 2). *Comparison of Faster-RCNN and Detection Transformer (DETR)*. Medium. https://whatdhack.medium.com/comparison-of-faster-rcnn-and-detection-transformer-detr-f67c2f5a2a04#:~:text=DETR%20offers%20a%20number%20of

[9] M. Weber, J. Xie, M. Collins, Y. Zhu, P. Voigtlaender, H. Adam, B. Green, A. Geiger, B. Leibe, D. Cremers, A. Ošep, L. Leal-Taixé, and L.-C. Chen, “Step-ICCV21,” *MOT Challenge - Data*, 2021. [Online]. Available: https://motchallenge.net/data/STEP-ICCV21/. [Accessed: 20-Nov-2022].

[10] L. Wang, J. Shi, G. Song, and I.-fan Shen, “Penn-Fudan Database for Pedestrian Detection and Segmentation,” *Pedestrian Detection Database*, 2007. [Online]. Available: https://www.cis.upenn.edu/~jshi/ped_html/. [Accessed: 20-Nov-2022].

[11] W. Wang, J. Dai, Z. Chen, Z. Huang, Z. Li, X. Zhu, X. Hu, T. Lu, L. Lu, H. Li, X. Wang, and Y. Qiao, “InternImage: Exploring Large-Scale Vision Foundation Models with Deformable Convolutions,” 2022.

[12] Q. Song, C. Wang, Z. Jiang, Y. Wang, Y. Tai, C. Wang, J. Li, F. Huang, and Y. Wu, “Rethinking Counting and Localization in Crowds: A Purely Point-Based Framework,” 2021.