A Simple Label Approach to Deep Learning for Crowd Counting

December 2020

Abstract

Annotating images for building a Deep Neural Network (DNN) for crowd counting is one of the most time-consuming steps of the development. This report aims to build a DNN regression model of close to equal accuracy as comparable object detection models for crowd counting by only using the number of people in each image as annotation (referred to as simple labelling). The dataset consists of 2000 surveillance camera images from a shopping mall with the number of people in the image as label. The data is split into train (1600 obs.), validation (200 obs.) and test (200 obs.). A pre-trained ResNet50 encoder with a custom two-layer decoder is found to be best for solving the problem of counting the number of people in the image given the dataset. Findings reveal that all models trained on images with simple labelling fail to learn and only slightly outperform constant mean predictions. Simple labelling does not provide the needed information for a CNN to identify which features to extract for people counting, where comparable object detection models from previous literature clearly outperforms. Based on this it is proposed to not use simple labelling for crowd counting, but rather the SD-CNN presented by Basalamah, Khan and Ullah, 2019 as it performs better and requires arguably shorter time for annotation.

Contents

1 Introduction

There are several reasons why crowd counting is useful in the current society. One relevant reason in the time of this pandemic is to count the number of people in a room, like a restaurant for I.e., where a crowd counter warns the owner if the number of people in their restaurant exceeds the current regulatory limitations. Another example is for businesses to accurately track the number of customers in a store in a given time of the day, to analyse peak visit hours. Implementing this model could be done via a security camera or a secondary camera placed in the venue/room one wants to survey.

There are a lot of challenges surrounding crowd counting examples of challenges includes occlusion, diversity in clothing and size among people etc. These challenges are affecting the generalizability of a model and makes it difficult to successfully apply it in new domains or situations. This report aims to build a specific deep learning model which can predict the amount of people present in an image from a specific surveillance camera in a mall.

One of the most time-consuming tasks of building and training a crowd counting neural network is to annotate the images used for training and validation. The best performing crowd counting models are usually object detection models, which requires detailed annotation that includes coordinates of each person in the images. Because annotation is costly, both time and money wise, it is interesting to explore how accurate a model can become by using only one simple label consisting of the number of people in each image (referred to as simple labelling) and thus, solving the problem of people counting as a regression problem. More specifically the problem this report aims to solve is defined as follows:

Is it possible to create a regression model of close to equal accuracy as comparable object detection models for crowd counting, by using the number of people in each image as only annotation?

To solve this problem the report takes a starting point in the crowd counting Kaggle dataset, which contain 2000 labelled images from a single surveillance camera overlooking a corridor in a mall. The baseline model for benchmarking is a pre-trained model with a ResNet50 encoder, a single dense layer decoder, and no augmentation of the training images. The output layer makes one value, which is the number of people predicted in the image. This single output approach makes it a difficult problem to solve, as the model does not have much information to learn from compared to models trained on data annotated for object detection, where the element(s) of interest is marked on each training image. However, the ability to create an accurate model for crowd counting using only a single label increases the number of potential cases where a model like this can be applied.

To evaluate the model, one uses Mean Absolute Error (MSE) from the validation set as well as MAE from the test set to compare the model with previous research.

2 Related Work

Prior research tackles the problem of crowd counting in many ways, and the evolution among the techniques used is moving rapidly. The initial problem with regular CNN's for crowd counting is that persons in an image may have different locations, varying aspect ratio, differing colours of clothing etc., which means that the number of regions a CNN must look at for in each image increases drastically with the variation of the images. One of the early methods to mitigate this problem is a CNN capable of selectively extract 'just' 2,000 Regions with CNN features (R-CNN) from an image. (Ross Girshick et al., 2014) However, this is not a perfect solution and shortly after followed the Fast R-CNN, then Faster R-CNN, YOLO etc. and now there are various CNN techniques for crowd counting. (R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms, 2020)

In most research for crowd counting, scientist commonly train their model to count annotated heads. This requires time-consuming annotation of training images like in the example of Gao et al., 2016. Motivated by the R-CNN for object detection, which takes general object proposals as input, Geo et al., 2016 used the cascade Adaboost algorithm to predict the head region proposals which are fed into a CNN. The CNN is then only used for feature extraction, which saves time as classification is quite time consuming for a CNN. To do the classification Gao et al., 2016, uses a fast linear Support Vector Machine (SVM) and thereby mitigates the time-consuming classification problem of the R-CNN. The results of the study is measured as precision among predictions of crowds divided into three groups: low (¡ 30 persons), middle (30 ¡ 50 persons), and high (50 ¡ 83 persons). The average precision across all three groups is 0.80. However, even though classification time is reduced, Gao et al., 2016 still had to manually annotate 15,292 samples of heads in frames/images and collected further 24,891 samples of empty frames/images (like empty classrooms etc.) to have a proper foundation for building their models. This annotation of more than 15,000 heads is a relatively time-consuming part of the study and far from ideal. (Gao et al., 2016)

The problem of annotating each person of every image can be mitigated as Basalamah, Khan and Ullah, 2019, show by using a Scale Driven Convolutional Network (SD-CNN) to count the number of heads in each frame. In the paper, they annotate only a set of heads at random size and location of each image, to develop a scale map representing the range of head sizes of the image, thus, no need to annotate every single head in the training set. After developing the scale map, they feed it into the SD-CNN which uses the scale-aware proposals to count the number of heads in the image. The model is tested on 3 major datasets (WorldExpo'10, UCSD, and UCF-CC-50) with varying frame size and people density. The results show that this approach with rather simple annotating can outperform state-of-the-art methods in terms of both frame-level and pixel-level analyses. Comparable models which is outperformed by this SD-CNN includes Faster R-CNN, MCNN, Liping et al. (2019), and more. It is however hard to compare the precision of this study with the precision of Gao et al., 2016 as

the datasets has more variations. The precision on the three datasets of the study is WorldExpo'10 = 0.6946, UCSD = 0.7358, and UCF-CC-50 = 0.4567. (Basalamah, Khan and Ullah, 2019)

Like Basalamah, Khan and Ullah, 2019, this report also aims to reduce the time spent on annotating the training data, rather than reducing the time spent by the CNN's, as Gao et al., 2016 investigates. The approach of this report differs, by using the before mentioned simple labelling of just the number of people in each image, thus not considering the location of people in the image. This greatly simplifies the problem as it no longer is a classification + location problem but can be treated as a regression problem. This approach is presented in a Kaggle competition, where few people have submitted their take on the problem, from which the best of the solutions are the benchmark model for this report.

3 Methods

3.1 Data set

The dataset consists of 2000 RGB images of frames in a video recording from a security camera in a mall, all with the size 480x640 pixels. The data is manually labelled with an integer of the number of people in each image. Each image includes a varying number of people, with constant camera angle and lighting. This means that the models build on this dataset will be tailored to this specific setting and is likely not very useful at counting crowds in other settings. However, the advantage of a consistent setting is that the models require less training data to become good.

Table 1: Training set description

Name	Year	Attributes	Resolution
Mall	2012	1 Fixed Scene	480x640

No. Samples	Avg. Count	Max. Count	Min.Count
1,600	31	53	13

In table 1 above, one can see a summary of the training set. To avoid data leakage the dataset is randomly shuffled and split into a training set (1600 obs.), a validation set (200 obs.), and a test set (200 obs.) before one do anything with it.

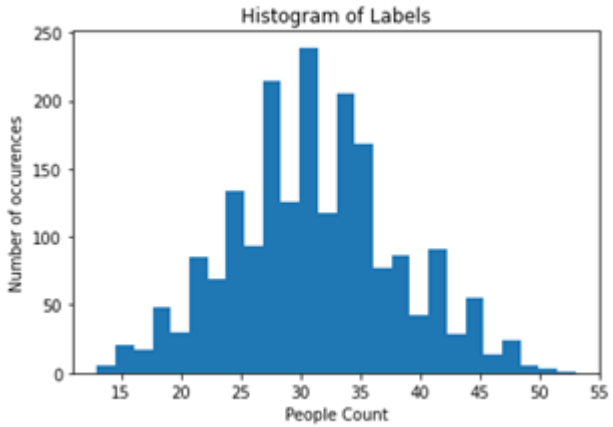

Figure 1: Histogram of the training set

In Figure 1 above the distribution of the labels (no. people in the images) are visualized, where one can see that the distribution follows a rough normal distribution. A potential risk of the training set is that the distribution of the

number persons in each image is rather narrow, meaning that a model constantly predicting the mean (31 persons) potentially could be hard to beat. To check for this, a model that only predicts the mean will be tested on the test set as comparison in the results section.

3.2 Network Architecture

The final model is built on a ResNet50 encoder trained on ImageNet. Using transfer learning to solve the problem this report aims to answer is convenient, as ResNet50 have been trained for classification on the +14 million images of ImageNet. Furthermore, a ResNet50 architecture pretrained on ImageNet can identify 1000 classes and a person is one of them. However, even though a ResNet50 network can identify a person, it is not able to count the number of people in an image out of the box. In continuation, ResNet50 is used as the encoder and a custom decoder is made to make the network learn to count people. To help the network to learn even further, the last 10 layers of the ResNet50 encoder is also made trainable, so the model gets more free memory to adjust to the specific task at hand.

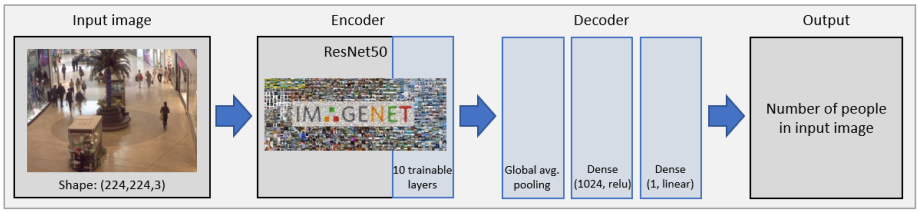

Figure 2: Network architecture

As presented in figure 2, the decoder consists of a global average pooling layer followed by two dense layers where the last dense layer outputs the predicted number of people in the input image. The activation function of the last dense layer is linear, as this activation function together with ReLU works best for regression problems.

3.3 Experiments

Several experiments are conducted before arriving at the final model architecture and setup. These experiments include choice of pre-trained encoder, learning rate optimizers and decoder architecture.

A MobileNet-v2 and a ResNet50 encoder are built and evaluated on the dataset, to discover which one is more suitable for the task. The two pre-trained encoders are selected based on their accuracy and number of operations from the online article by Canziani, A. Et al. (2018). Here both ResNet50 and MobileNet-v2 have a high accuracy with relatively low number of operations compared to e.g., VGG-16 or VGG-19. Figure 3 below show the training and validation Mean

Absolute Error (MAE) for two simple models trained with a MobileNet-v2 and a ResNet50 as encoder. The MobileNet-v2 model quickly reach a plateau on 6 MAE on the training set, while the ResNet50 model MAE decrease with every epoch. The validation MAE of both models does not converge properly, and none of them have a satisfactory validation MAE. Even though the ResNet50 seem to suffer from overfitting, it is still the best model in terms of the validation MSE. However, regularization is needed to reduce overfitting and increase the robustness of the predictions.

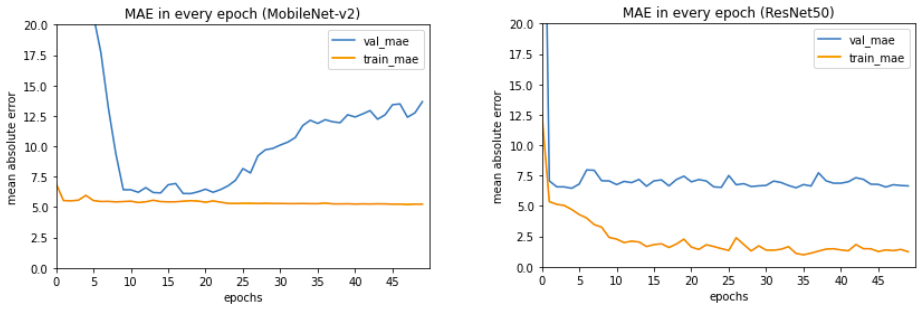

Figure 3: Model training loss comparison (Encoder: MobileNet.v2 & ResNet50)

The learning rate is controlled by a learning rate optimizer, which adjust if the validation loss does not improve in three epochs. The main concerns of training this model are either getting stuck in a local minimum or overshooting the global minimum. By decreasing learning rates, the model lowers the risk of overshooting the minimum as the change in gradients become too large. The best optimizer is the one that traverse the loss of the problem the best, empirical testing is done to find a fitting optimizer for this specific project. Both stochastic gradient decent (SDG) and adaptive learning rate optimizer (ADAM) are tested and compared against each other. However, the ADAM optimizer slightly outperformed the SDG optimizer on the validation loss when training the model, thus it is used for the final network.

The decoder architecture of this project is simple and consists of two dense layers. The reason for this simple two-layer dense decoder is that the last 10 layers of the ResNet50 encoder are made trainable. Additionally, several slightly more complicated decoder architectures are tested but yield no impact on validation loss. In continuation, increasing the number of trainable layers in ResNet50 do not decrease validation loss either.

4 Results

In table 2 below , the results of the convolutional neural network build in this report is compared to the results of related work. It is important to note that the comparable models are trained on annotated images, where the persons of the training images are marked, while the models of this report and the Kaggle-model are trained without any annotation of a person but the simple labelling. Furthermore, the MAE of the comparable models are calculated on a different test set than the models of this report, however, one assumes that both the test set of this report and the test set of the comparable articles are sufficient representations of the same theoretical population of images with similar scenes and similar number of persons on each image.

For each of the models of this report one has tested multiple combinations of different settings of the hyperparameters to ensure the best possible fit. Table 2 summarises the performance results of only the best model built with each encoder. The comparable models in table 2 below are presented in Basalamah, Khan and Ullah, 2019.

Table 2: Model evaluation against models from previous research.

Regression Based Models	MAE
Kernel Ridge Regression	2.16
Cumulative Attribute Regression	2.07
Lempitsky et al.	1.70
CNN Based Models	
Kaggle-model*	***5.46***
Our Model*	***5.32***
Faster R-CNN	2.89
MCNN	1.07
Liping et al.	1.03
SD-CNN	1.01
Mean Prediction*	***5.42***

**The models built in this report are marked with bold and italic*

The results reveal that the best model built in this report is closely followed by the mean prediction model and far behind the comparable object detection models. These results indicate that the model fail to learn doing training and since the comparable models perform significantly better, one can conclude that this report fails to build a model with simple labelling which performs close to similar accuracy as the object detection models. It seems like there is a lower boundary of how good the model can become with the simple labels, as the validation loss of the model reach a plateau no matter how the parameters are tuned.

5 Discussion

The final model did not reach a satisfactory MAE when comparing to competing models from existing literature. The simple labels ease the manual annotation but leaves more work for the model. The model struggles to determine exactly what features to look for to improve the loss. This is also apparent from the plot of training and validation loss in Figure 3, which flattens out in a few epochs. The three heatmaps below give a sense of what features the model extract from the images when training. The heatmaps are made with grad-CAM, which captures the activations of the second last convolutional layer of the model. The second last convolutional layer is used, as it has a size of 7x7 pixels while the last convolutional layer only has a size of 1x1 pixel, which is not suitable for a heatmap. Warm colours of the heatmap indicate areas with important features that the model use to predict the number of people, while cold colours indicate the opposite. An ideal model would have high activation only on the people, and low activation for everything else. As stated in the conclusion, this is not an ideal model, which is further supported by the heatmaps. Several people in the heatmaps below are coloured blue indicating that the model fails to identify the relevant targets with the single simple labelling approach, thus the model fails to become an adequate people counter.

Figure 4: HeatMaps of 3 randomly chosen images from the training set

There is also an additional issue with the shape of training images, as they need to be reshaped to fit the input shape of the pre-trained encoder. ResNet50 and MobileNet-v2 requires a quadratic input shape of 224x224 pixels, while the images used for this problem have the shape of 480x640. Reshaping the images changes aspect ratio and slightly distorts the people of in the images, which makes it harder for the model to recognise them. Based on previous literature and the findings of this report, one can conclude that a better way to tackle the problem of people counting with reduced annotation, is the approach by Basalamah, Khan and Ullah, 2019 described in the related work section.

References

[1] Kaggle.com. 2020. Crowd Counting. [online] Available at:
https://www.kaggle.com/fmena14/crowd-counting
[Accessed 6 December 2020].

[2] GitHub. 2020. Gjy3035/Awesome-Crowd-Counting. [online] Available at:
https://github.com/gjy3035/Awesome-Crowd-Counting
[Accessed 6 December 2020].

[3] GitHub. 2020. Gjy3035/Awesome-Crowd-Counting. [online] Available at:
https://github.com/gjy3035/Awesome-Crowd-Counting/blob/master/src/Datasets.md
[Accessed 6 December 2020].

[4] Crowd-counting.com. 2020. JHU-CROWD++ — A Large-Scale Unconstrained Crowd Counting Dataset. [online] Available at:
http://www.crowd-counting.com/download
[Accessed 6 December 2020].

[5] Gao, C., Li, P., Zhang, Y., Liu, J. and Wang, L., 2016. Crowd counting based on head detection combining Ada-boost and CNN in crowded surveillance environment. Neurocomputing, 208, pp.108-116.

[6] Basalamah, S., Khan, S. and Ullah, H., 2019. Scale Driven Convolutional Neural Network Model for Crowd counting and Localization in Crowd Scenes. IEEE Access, 7, pp.71576-71584.

[7] Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object de-tection and semantic segmentation.

[8] Medium. 2020. R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms. [online] Available at:
https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e
[Accessed 22 November 2020].