Human Counting on RGB images

Zesen Huang
zesen.huang@studenti.unipd.it

Abstract

Computer vision about counting human in public places, such as stations or parks, plays an important role in city management. Knowing the number of people in a public place will help control the flow of people and provide better services for citizens. In this work I exploited and compared different convolutional network models to compute the number of people from images of shopping mall. Some classical CNN models were used in experiments to compared their performance on people counting, and I also tried to use a more effective model to discuss the reason why it performs better.

1. Introduction

In the process of urbanization, more and more people choose to live in cities. The rapid growth of urban population has brought a series of challenges to urban management, including the problem of too many people in public places, such as sports events or shopping malls. With too many people in a public places, stampede and casualties will be easily caused once panic occurs. Moreover, in the current world of the COVID-19 pandemic, governments around the world also have certain restrictions on the number of people and social distancing in public places. Therefore, it is of great significance to use computer vision to calculate the number of people in pictures or surveillance videos.

In the development of computer vision, convolutional neural network (CNN) has excellent performance and has many different varieties. In this work, I used and compared two common network structures, VGG16 [6] and Resnet50 [4] for deep learning on Mall Dataset. All the models were trained on the Mall Dataset and then the best model were saved. In order to improve efficiency and obtain better performance, I used a technique of transfer learning and fine-tuning to train models and count the number of people from images. The structure of models and performance were reported and discussed respectively to find out what kinds of networks are suitable for human counting job.

2. Related work

2.1. VGG model

There are many convolution neural networks developed since 2012, and the VGG is the one of famous model for very deep convolutional networks for large-scale image recognition. VGG has 16-19 weight layers with 224×224 input and 3×3 convolution filters. Max-pooling is performed over a 2×2 pixel window with stride 2 and the final layer of model is the soft-max layer. The following figure(1) shows the VGG16 architecture:

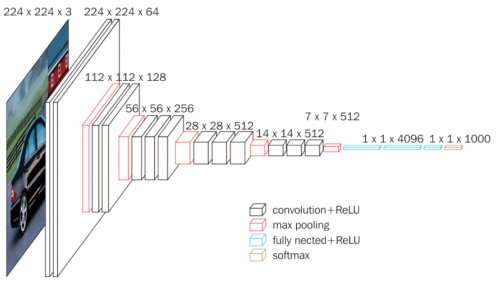

Figure 1. VGG16 architecture

The main contribution of VGG model is to demonstrate that the performance and accuracy of the model can be effectively improved through smaller convolution kernel and deeper network structure. By using multiple smaller convolution kernels to replace larger convolution kernels, parameters can be reduced on the one hand and the performance in nonlinear representation ability can be improved on the other hand, thus the feature learning ability of CNN can be enhanced.

2.2. ResNet50

The key of ResNet50 model is the residual learning(2) and identity mapping by shortcuts, which effectively solve the problem of gradient vanishing and reduce the training error caused by stacking more networks. In Residual Network, the data output of one of the earlier layers is directly introduced into the input part of the later data layer by skipping the other layers. This means that the content of the following feature layer will be contributed linearly by one of the preceding layers.

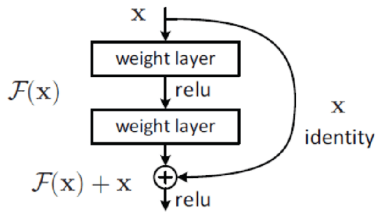

Figure 2. Residual learning: a building block

ResNet uses two types of residual units, as shown in Figure(3). The image on the left corresponds to the shallow network, while the image on the right corresponds to the deep network. For shortcuts connections, the input can be directly added to the output when the input and output dimensions are the same. But when the dimensions are inconsistent (corresponding to a doubling of the dimensions), these cannot be added directly. There are two strategies: (1) zero-padding is used to add dimensions. (2) New mapping shortcut is adopted, generally 1x1 convolution is adopted, which will increase parameters and the amount of calculation. In addition to using identity mapping directly, short-circuit connections can of course be made using Shortcut projection.

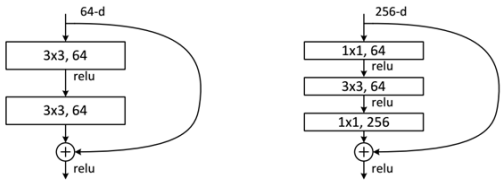

Figure 3. Different residual units

3. Dataset

In the experiment, I used Mall Data [1] [2] [3] [5] collected from a publicly accessible webcam for crowd counting and profiling research to train the models. There are 2000 video frames in this dataset and over 60000 pedestrians were labelled by detecting the head position in all frames. Every frame is a RGB 640×480 image. According to the distribution plot(4), the number of people concentrated in 25-35 people, which is not very much.

In this work, I randomly split the dataset into 1800 training data, 200 testing data and 180 validation data. The training data was used to train models, and the validation data and test data was used to adjust model hyperparameters and estimate generalization ability respectively. Because the input of VGG and Resnet is $224 \times 224 \times 3$, I resized every image to 224×224 to fit the requirement. Moreover, training time can also be shortened by reducing images size. In order to see if a model has better performance with larger

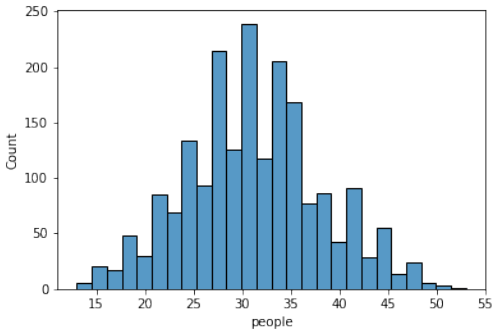

Figure 4. People number distribution

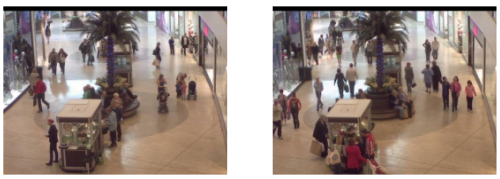

Figure 5. Mall dateset images

input size, I also resized the images to 112×112 in the experiment. Since using VGG and Resnet expects a specific kind of input preprocessing, I converted the input images from RGB to BGR, then zero-centered each color channel with respect to the ImageNet dataset, without scaling.

Figure 6. Images processing

4. Method

4.1. Transfer learning

With the emergence of more and more machine learning application scenarios, the existing supervised learning with better performance requires a large amount of annotated data, which is a tedious and costly task, so transfer learning has attracted more and more attention. Transfer learning consists of taking features learned on one problem, and leveraging them on a new and similar problem. Generally speaking, for CNN, the first few layers have little influence on the result output, and the main useful information and features are mostly learned at the top layer, so the top layer is more important to the output of the model, which is why transfer learning can be carried out. Because of the large data and shortage of hardware resource, transfer learning is a better choice to be used in this work. In this case, I used VGG16 and Resnet50 that have been trained on Im-

ageNet to learn images representation and predict on Mall Data, and fine-tune the last few convolutional layers to fit the target domain. The most common incarnation of transfer learning in the context of deep learning is shown as the following workflow:

1. Import previously trained models;
2. Freeze them, so as to avoid destroying any of the information they contain during future training rounds.
3. Insert new, trainable layers on the top of frozen layers. They will learn new features based on the old features and make prediction;
4. Train the new layers on new dataset.

4.2. Model architecture

When creating the top layers for VGG16 and Resnet50 model, I mainly considered three kinds of layers to stack: flatten layer, dense layer and dropout layer. As the result is the number of people, which is a continuous numerical value, thus the output layer is dense layer with one dimension. Besides, several dense layers with 1024 dimension and relu activation were used to stack together to extract more features. Normally, deeper network can learn more representation of images and improve its performance, but we also need to care about the overfitting. The flatten layer was used after dense layers to flatten the output of dense layers. Considering of the overfitting problem, I exploited dropout layer before the output layer.

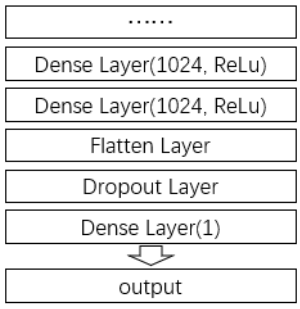

Figure 7. One of the architecture of top layers

4.3. Evaluation metric

Since the output result is a numerical value about number of people, this is a regression problem. Therefore, I selected Mean Squared Error(MSE) as loss function and Mean Absolute Error(MAE) as estimation metrics respectively during training, which are defined as follows:

$$MSE = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(y_i - \hat{y}_i)^2}, \quad MAE = \frac{1}{N}\sum_{i=0}^{N}|y_i - \hat{y}_i| \tag{1}$$

where N is the number of images, y_i is the is the actual number of people in the ithimage and $\hat{y_i}$ is the prediction of ith image. Roughly speaking, MAE indicates the accuracy of the estimates, and MSE indicates the robustness of the estimates.

5. Experiments

I compared VGG16 and Resnet50 network performance by changing their top layers architecture, Dropout rate, optimizers(Adam) learning rate, training batch size and image size. At the beginning of training, I only used half of the data because the more data used in the training, the longer the training time it took. I made the models run on the subset of data to find the suitable architectures and hyperparameters, and used all data after setting up several reasonable architectures and hyperparameters. In the end, I evaluated and compared different models by observing the loss plot and computed the MAE on test dataset. For example, if the MAE is 1, that means the different between prediction and true value is 1 people. The smaller MAE is, the more accurate the predicted value is.

5.1. VGG16 experiment

First of all, I resized the images to 112×112 to train and imported VGG16 that has been pre-trained and froze its all layers and did the fine-tuning work. I stacked 1 flatten layer, 1-5 dense layers and 1 Dropout layer on the top of VGG16, and set up dropout rate to 0.4 and Adam learning rate to 0.001 because these values are reasonable to choose. Besides, I also used VGG16 to run on 224×224 images and made a comparison after training on 112×112 images.

Architecture	Image size	MAE
Flatten+ Dropout	112×112	3.76413
Flatten+ Dense(1024)+Dropout	112×112	2.78171
Flatten+ 2 Dense(1024)+Dropout	112×112	2.86833
Flatten+ 5 Dense(1024)+Dropout	112×112	2.56536
Flatten+ 5 Dense(1024)+Dropout	224×224	2.30762

Table 1. VGG16 experiment result with Dropout rate 0.4 and Adam learning rate 0.001.

According to the results, adding 1 dense layer can bring obvious improvement, with 0.98242 of MAE value decreased. However, the model's value of MAE with 2 dense layers was a little higher than the one with 1 dense layer. And when I stacked 5 dense layers on the top of model, I got better performance this time with a little improvement. Therefore, it can be inferred that using more dense layer can improve model performance. Compared with 112×112 images, using 224×224 images to train model also can improve performance. This is because more information is

included in larger images and model can learn more features and boost representation ability.

5.2. Resnet50 experiment

When training with Resnet50, I used 224×224 images to train directly because I learned that larger images were better based on the VGG16 experiment. And I also stacked 1, 3 and 5 dense layers on the top of model to see whether Resnet50 can get better result with more layers. The dropout rate and Adam learning rate were selected to 0.1 and 0.001 respectively.

Architecture	Image size	MAE
Dense(1024)+Flatten+Dropout	224×224	2.78311
3 Dense(1024)+Flatten+Dropout	224×224	2.48081
4 Dense(1024)+Flatten+Dropout	224×224	2.45907
5 Dense(1024)+Flatten+Dropout	224×224	2.39781

Table 2. Resnet50 experiment result with Dropout rate 0.1 and Adam learning rate 0.001.

It's easy to see that stacking more dense layers can get lower MAE. However, with more and more layers, the improvement of model become more and more insignificant.

5.3. VGG16 VS Resnet50

According to the results I obtained in this experiment, I made a comparison between VGG16 and Resnet50 performance. As we can see, both model can improve their performance by easily stacking more dense layers with dropout and Adam optimization. However, the VGG16 only needs to be trained with 112×112 images to achieve similar performance as the Resnet50 with 224×224 images, and also save more time than Resnet50. The smallest MAE I got in this experiment is 2.30762, which was obtained by VGG16 trained on 224×224 images. And the second smallest MAE, 2.39781, was from Resnet50 with 5 dense layers, which is close to the smallest one.

6. Conclusion

In this work, I exploited transfer learning and developed different top layer architectures of VGG16 and Resnet50 to count number of human on 2000 images from Mall Dataset, and made a comparison of models performance in this experiment. Most of models outperform here achieved 2.7 MAE on average by fine-tuning, and the best models was the VGG16 with 5 dense layers and dropout layer(0.4) and Adam optimization(0.001), which obtained 2.30762 MAE. All the works and experiments showed that VGG16 and Resnet50 can achieve a good performance on counting human from images of mall.

7. References

References

[1] Chen Change Loy, Shaogang Gong, and Tao Xiang. From semi-supervised to transfer counting of crowds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2256–2263, 2013.

[2] Ke Chen, Shaogang Gong, Tao Xiang, and Chen Change Loy. Cumulative attribute space for age and crowd density estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2467–2474, 2013.

[3] Ke Chen, Chen Change Loy, Shaogang Gong, and Tony Xiang. Feature mining for localised crowd counting. In *Bmvc*, volume 1, page 3, 2012.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[5] Chen Change Loy, Ke Chen, Shaogang Gong, and Tao Xiang. Crowd counting and profiling: Methodology and evaluation. In *Modeling, simulation and visual analysis of crowds*, pages 347–382. Springer, 2013.

[6] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *Computer Science*, 2014.