Smart Car Parking System using Deep Learning.

October 28, 2017

Author: Syed Sheece Raza Gardezi
Supervisor: Anoop Cherian
Supervisor: Peter Strazdins

Project undertaken at:

U5752631
sheecegardezi@gmail.com

Individual Computing Project
COMP8755

27 October 2017
Canberra, ACT, Australia

Except where otherwise indicated, this report is my own original work.
Sheece
October 28, 2017

This report is dedicated to François Chollet founder of kerase library.

1 Acknowledgments

I wish to acknowledge the generous support of the following people:

Anoop as my supervisor for her invaluable guidance, support, and belief throughout this project.

Peter (ANU RSCS) for his project coordination and research skills training.

Syeda Raiha Raza Gardezi and Aksit for voluntering and helping throughout the course the report.

John for proof reading and providing valuable insight.

The participants of my survey for generously volunteering their time for this study.

2 Acknowledgement of Country

I acknowledge the First Australians on whose traditional lands I live and study, and pay my respects to the elders of the Ngunnawal people past and present.

3 Abstract

. Finding parking spaces can be time consuming and frustrating, leading to a decreased productivity and increased carbon emissions. This paper aims to model an effective solution by creating an automated program which uses affordable cameras to detect the real-time availability of parking spaces. To realize this system, the most effective deep learning architecture was selected from four different architectures. A comprehensive analysis of 32 different parking lots to check if parking spaces were empty or not was conducted; using algorithms based around the chosen deep learning architecture. The success rate of 99% was achieved, indicating a successful approach. Dataset SSRLot was created for the purpose of this study, encompassing 45,000 images, along with using datasets PKLot and CNR-Park for standardization purposes. A mobile application was created to view the live updates of parking lots on google maps. Our approached solution outperformed and was a success in being generalized over different carparks.

Keywords: Machine Learning, Classification, Deep Learning, Convolutional Neural Networks, Parking space dataset

Contents

4 Introduction

Every day many drivers face the problem of finding a car park. Drives typically waste 30 minutes each day to find a parking space(Shoup, D.C., 2006). A solution to this problem is real-time information of empty parking spaces being available. This would allow drivers to plan their route ahead of time, saving them time and also being more environmentally friendly. This system could also be used in security to notify an owner if their car has been moved from the parking spot. In countries with low economic strength, this system would provide an economical solution to tackle car theft.

To create a real-time application able to indicate an empty car park, the solutions include either installing sensors at every parking space or using pre-existing security cameras. The latter is more viable.

In this proposed system the coordinates of the parking spaces of a parking lot are stored. When new images are received from the camera they are segmented into individual parking space images. A pertained classier that predicts whether the image represents an empty or occupied space. We used CNNs as the classifiers for this task. The challenge is to train an algorithm that is flexible to different weather conditions, orientations of the car and able to identify partially concealed car images. Despite the pre-segmentation restriction, such a system could still be practical as it could be applied to any stationary camera feed where the regions of interest (i.e. the bounding boxes of each parking space) remain fixed. One could use pre-existing cameras (like security cameras) for this task, or install one or more cameras that overlook the whole parking lot.

A major challenge faced was the lack of availability of a consistent and reliable dataset that captured images of car parked in car from various angles. To this end a dataset, SSRLot, was created with 45,000 images of 20 different cameras covering various parking lots. Using this data 28581 images (17334 empty spaces and 11247 occupied spaces) were labeled during the course of this semester. The labeling of each parking space was carried out manually and was categorized into ‘vacant’ or ‘occupied’. The images were also manually put into category by weather of ‘sunny’, ‘cloudy’ or ‘raining’. Two existing dataset PKlot and CNR-Park were also standardized. In order to ensure that an adequate generalization approach is established, parking lots data was used which had many varying characteristics. The CNN were established by being trained through different datasets which ensured this approach has high accuracy.

Our method allows us to monitor parking lots at a much cheaper cost than sensors with comparable accuracy results. Moreover, technology capable of detecting empty parking spaces can be further used in applications which require video surveillance, facial recognition etc. The cameras capture the parking spaces of a parking lot to determine if the parking space is vacant or occupied by using a trained CNN.

5 Related Work

Intelligent parking system used already existent image detection softwares. Gives insight to similar model developments as the proposed parking detection system mentioned in this report.(Al-Kharusi and Al-Bahadly, 2014)
Explores methodology of mapping parking lots for automatic system of parking spaces which detection. This methodology can be adopted to integrate with the proposed structure and provides some interesting insights into parking space detection systems.(Choeychuen, 2013)
Studies the parking structure in America and further studies trends and planning requirements needed for Lots of Parking which have a profound impact upon our daily lives. This emphasises the need for more efficient systems such as this app suggests. (Jakle and Sculle, 2005)
ImageNet is a challenge undertaken to classify the category where an image belongs. Participants submit their top five categories for the prediction of images, leading to the first breakthrough of AlexNet with a low error rate of 15.3
"Investigating the effect of the convolutional network depth on its accuracy in the large-scale image recognition setting. Uses an evaluation of networks of increasing depth using an architecture with very small convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16–19 weight layers. Vgg16 is similar to vgg19 but with more convolution layers."(Simonyan and Zisserman, 2014)
This is a study into behaviorial patterns of drivers and their dependencies on parking condition in the area they commute to. This study shocases the influence on drivers created by carparks.(van der Waerden, Janssens and da Silva, 2017)
Almeida, P., Oliveira, L. S., Silva Jr, E., Britto Jr, A., Koerich, A., PKLot – A robust dataset for parking lot classification, Expert Systems with Applications, 42(11):4937-4949, 2015.
They CNR-Ext Alex net (Amato et al., 2017) used a varion of Alexnet to detect empty car spaces. The also worked on developed a smart camera to decentralise the identification task. They also created the dataset CNRPark-Ext which we used for our experiments as well.
They PKLot (Almeida et al., 2017) collect a dataset for three parkinglots. Each picture had a hierarchy day, parking lot and weather conditions and was accompanied with a meta data file informing about the occupancy of parking space. We used the same format to create our dataset and standardise CNRPark.

6 Dataset

There are only two datasets relevant to car parking lots openly available are namely PKLot (Almeida et al., 2017) and CNRPark (Amato et al., 2017). Both of these datasets have an issue of having limited amounts of cameras with pictures in PKLot being from 3 cameras and CNRPark being from 9 cameras. To this end SSRLot dataset was created, encompassing pictures from 20 different cameras. Pictures were stored at an interval of 30 mins over the course of two months. Almost 45,000 pictures were collected but due to time constraint only a fraction of these images were labeled. Metadata about each image in encoded in xml file with the same name. The set includes data about their location, name of the parking lot, date, occupancy and weather conditions. Images are of various size depending on specification of the camera this also gives variance to the dataset. The specifications of each camera are shown below:

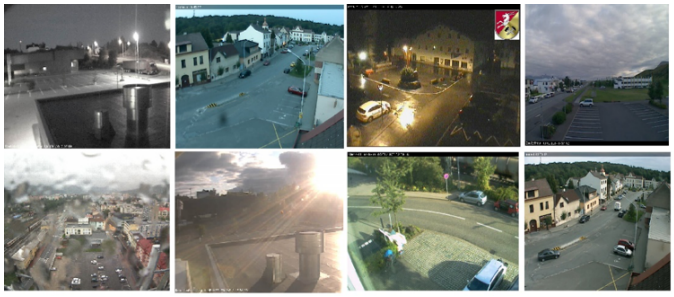


Figure 1: **Examples of parking spaces in dataset.**

To create SSRLot a corn job was set up on Linux machine to save an image from each of the 20 cameras after 30 mins. Initially, the interval was 15 mins, but this caused a high volume of images with no change in the condition of the parking lot to be downloaded; therefore, time was increased to 30 mins. The original quality of the images was maintained.
Two tools were created one to create a wireframe describing the position of each parking space and secondly a labelling tool to label each parking space. There are two tools mentioned in literature MIT LabelMe (Labelme.csail.mit.edu, 2017) and Matlab's Labeler (Labeler, 2017) but both would have caused the wireframes to be drawn and redraw for each image and they don't provide a functionality to save custom data. To this labelling tools were created. The labelling tools was gamified; it showed how many images were left to label for each parking lot this introduced a factor of challenge. Inception v3 trained on CNRExt was used to give labels to each segments on dataset. This made the manually job to label parking segments easier. Secondly using a threshold on black colour pixel night time images were deleted.
Some problems in this dataset arise, such as images with bright lighting making the cars look too bright and the trees causing uneven lighting and shadows. Heavy rainfall also presents an outlook of night time due to lighting issues. These problems were taken care of by increasing the variance of test data.

Dataset	Total Images	Total Spaces	Interval Between Pictures	No of cameras
SSRLot	45000	28581	30 mins	20
PKLot	12417	695899	5 mins	3
CNRPark	-	144965	-	9

Table 1: Details about cameras

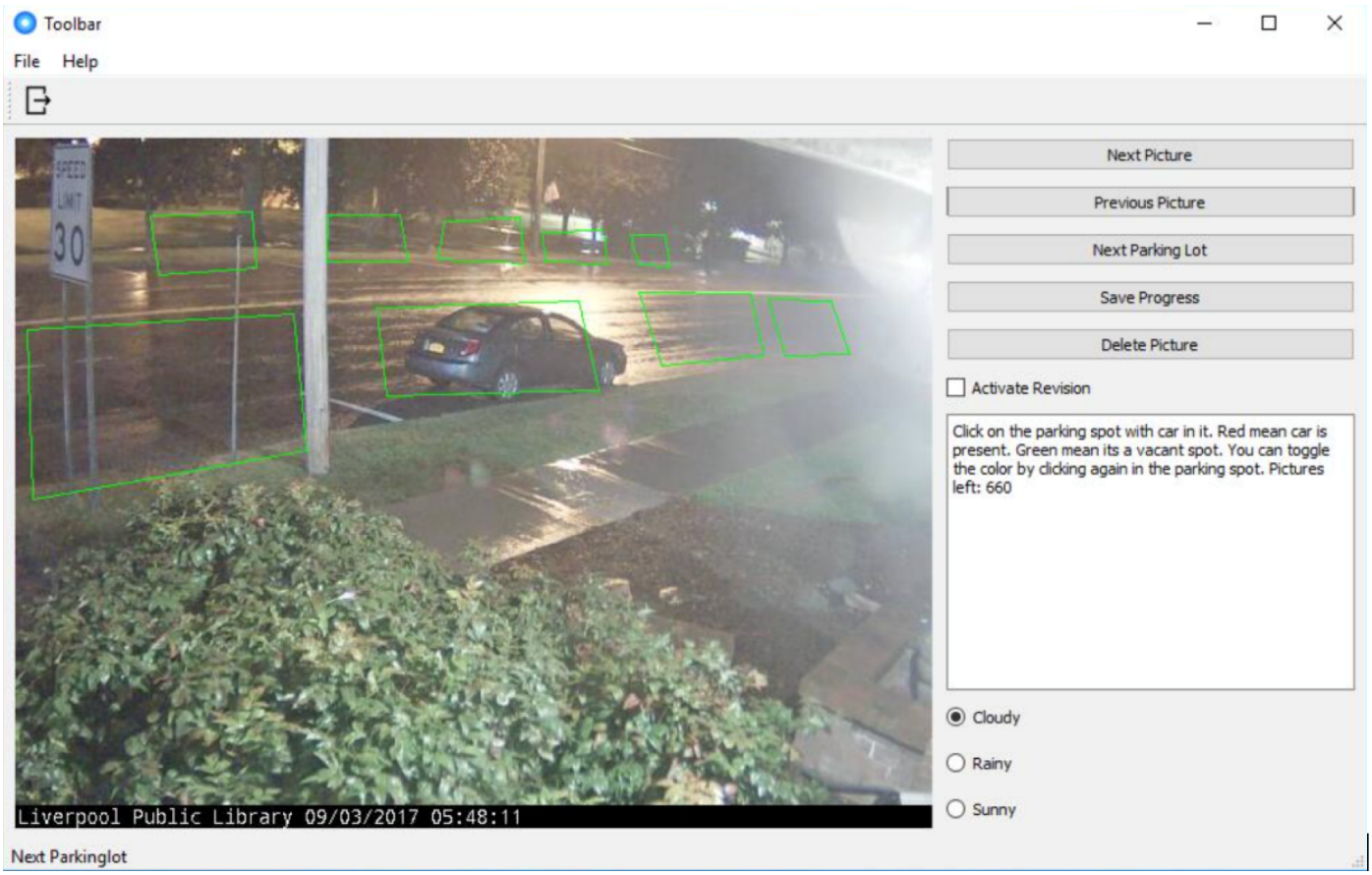


Figure 2: **Labeling tool that was used to label parking spots.**

Features in dataset:
1. Images have variance in lighting, such as sunshine, rainy weather, shadows etc.
2. Camera position is at different angles and heights with cameras placed over different parking lots.
3. Multiple uses
4. Various climatic settings
5. Detection is not zoomed in, as in most cases the images is taken from quite an altitude

The variance of cameras' environment and general setup is vast and an important strength of this database as it allows the training of a more accurate algorithms. In addition, due to these variances in environment, this data set can be used for normalization algorithms, modelling parking spaces etc.

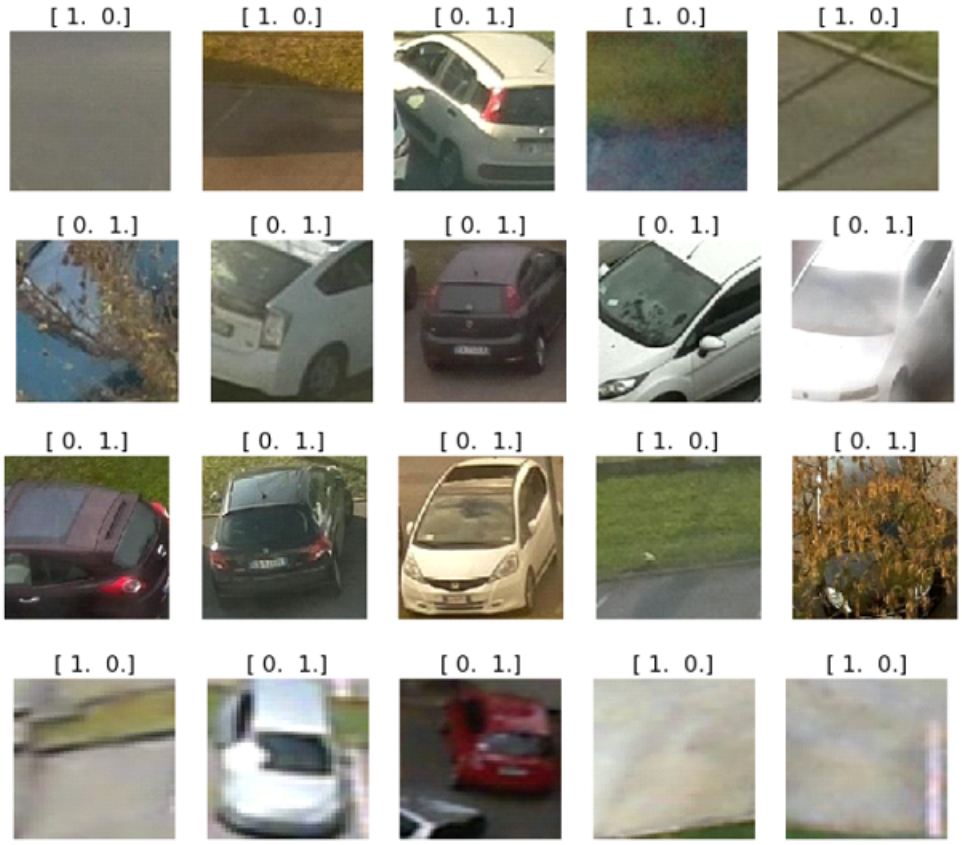


Figure 3: **Examples of difficult parking spaces to label in dataset.**

Camera	**Dimensions(pixels x pixels)**	**Resolution (dpi)**	**Bit depth**
1	640x480	96	24
2	640x480	96	24
3	704x576	96	24
4	640x480	96	24
5	1280x1024	96	24
6	640x480	96	24
7	640x480	96	24
8	1280x960	96	24
9	640x480	96	24
10	640x480	96	24
11	640x481	96	24
12	640x482	96	24
13	640x483	96	24
14	640x484	96	24
15	640x485	96	24
16	640x485	96	24
17	640x487	96	24
18	640x488	96	24
19	640x489	96	24
20	640x490	96	24

Table 2: The specification of pictures captured by the cameras.

7 Method

7.1 Convolutional Neural Nets

Initial experiment were done using hard coded rules for threshold on number of corner pixels present, values of hue. It was discovered that using such an approach will not scale well to either changes lighten conditions or if the parking spot was obstructed with some other object like leaves. A Convolutional Networks were used is a deep learning data structure that has proved to work well for image classification tasks over the past years. It comprises of 3 layers Fully-Connected Layer, Convolutional Layer and Pooling Layer. The first layer is an input layer. Its size will be same as the shape of input image i.e. (no of color channel, width of image ,height of image). The convolution layer has a filter of 2D of some arbitrary size defined by the architecture. The filter is initialized to some weight values. The shape that this filter represents corresponds to features like edges, circles, diagonals etc. Convolution layer performs a convolution between filter and the output of the previous layer. This results in an image which has the pixels that corresponds to the feature of the filter to have a higher values and pixels that don't correspond to the features in the filter to have low value. This result passes through a non-linear activation function like Relu or tanh. Relu is defined by the formula:
f(x)=max(0,x)
The combination of non-liner function and filter can be used to model any function as proved by universal approximation function. This gives the convolutional network the ability to model complex pattern in images. Max pooling layer is used to down sample spatial dimensions. This down sampled output is feed into the next layer. The process is repeated over and over agin according to the architecture of the convolutional neural network. The last layer is fully connect layer which has an output of dimensions 1 by 1 by number of classes. The values in the last dimention correspond to the likelihood of that class being represented by the input image. The most common classifier is softmax classifier which has the formula :

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}$$

To improve the CNN's performance, a loss function is used to estimate the quality of the CNN's answer (i.e. its class scores). We used the categorical cross entropy, which is the following: Minimizing this function, then, will allow the CNN to maximize the probability that it chooses

$$H(y, \hat{y}) = \sum_i y_i \log \frac{1}{\hat{y_i}} = -\sum_i y_i \log \hat{y_i}$$

the correct output given the input.

7.2 Batch Normalization

The performance of a CNN often depends heavily on the initialization of the parameters of the network, and since those are initialized randomly, the performance can vary widely even when the same hyper parameters are used. To solve this problem, one can either train the model multiple times with the same hyper parameters to get the best possible performance, or normalize the input to each layer with a technique called batch normalization (Ioffe and Szegedy 2015). During training, each dimension of the mini-batches is normalized using the mean and variance of that specific batch. However, to keep the expressiveness of the model, additional parameters beta and gama are introduced, that scale and shift the normalized value and are learned during training. (Just normalizing, without giving the net the option to scale and shift the normalization, would constrain the net to specific outputs and therefore reduce its expressiveness).
The algorithm then looks as follows, as described in (Ioffe and Szegedy 2015):

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Batch normalization is usually added to the network after convolutional or fully-connected layers, but before the non-linearity. During test time, the normalization is done using a sample mean and variance.
During the training process for the binary classifier, we ran into the above issue. Occasionally, due to an un- lucky parameter initialization, the validation accuracy of our model would plummet. After applying batch normalization layers after each convolutional layer, this problem disappeared.

7.3 Receptive field

Images have very high dimensions. Each neuron of convolution layer cannot be connected to each neuron of input layer. Rather we define a small regain of neurons of input layer that contributes to the values of neuron of output layer. We call this receptive field or filter. The depth of the filter is always equal to the depth of the input image. The most common filter size is 3x3.

7.4 Stride

Stride is how much a filter is shifted on an image with each step. The filter slides over the image, stopping at each stride length, and performs the necessary operations at that step.

7.5 Dropout

It stops the Neural Network from overfitting randomly dropping weights. This ensures that the network dose not over fits to the training data.

7.6 Universal approximation theorem

"In the mathematical theory of artificial neural networks, the universal approximation theorem states that a feed-forward network with a single hidden layer containing a finite number of neurons (i.e., a multilayer perceptron), can approximate continuous functions on compact subsets of Rn, under mild assumptions on the activation function. The theorem thus states that simple neural networks can represent a wide variety of interesting functions when given appropriate parameters; however, it does not touch upon the algorithmic learnability of those parameters."(En.wikipedia.org, 2017)

7.7 Stochastic Gradient Descent

The method involves estimating the magnitude of error in neuron by first processing and analysing numerous data sets and generalizing the error. And then adjust the weights of the neurons. It takes any function and finds us the parameters that fit our outputs. This is mathematical optimization of a function. This finds us a local minimal. Because there are millions of parameters for which a point that have to be found

The error function over n training examples can be written as an average:

$$E = \frac{1}{2n} \sum_{x} \|(y(x) - y'(x))\|^2$$

and the partial derivative with respect to the outputs:

$$\frac{\partial E}{\partial y'} = y' - y$$

7.8 Symbolic Derivatives

The change in gradient is required by backpropagation to adjust weights. Instead of calculating millions of gradients for each weight in real time gradient, symbolic gradient can be calculate using chain rule. Modern libraries have the functionality building; given the model architecture and they will calculate the symbolic derivative that will be used by scholastic gradient descent to find a optimal solution given a learning rate.

8 Implementation

Due to the limitation of memory of graphic cards available, we were unable to train a CNN with more than 2 layers of filter size 512 by 512 or parameters more than 50,000. Keeping these limitations in mind we can train a model with architecture of 3 convolution layers with filter size increasing from 32 to 256 and receptive field of 5 by 5. Each of those layers was then generalized and analyzed though normalization of batch and finally through a ReLu activation function. Finally, we used a Softmax layer to compute class scores. Model was trained from scratch on all three parking lot data sets, but the network got saturated at around 54% and was biased towards giving the prediction as empty lot. This was due to the fact that there were not enough parameters trained to encompass all the features that could lead to differentiation between the two classes. So, the state of art CNN architectures over the past five years were selected for transfer learning. They were trained for this particular problem. Most of weights of their layers were kept as they were trained on ImageNet. These models were fine-tuned. The fully connected layers and the final convolution layers were trained again. A dropout of 50% was used. The activation function Relu was used for this transfer learning.
Four differed architectures were selected to experiment with namely, vgg16, vgg19, resnet50 and Inception v3. Various experiments were conducted to gauge their performance.
A study to test the effectiveness of different architectures have not been conducted on this sort of datasets. Only one team (Amato et al., 2017) used AlexNet to solve this problem.

8.1 Preprocessing

For VGG models, input shapes of the images dimensions were set to (3,224,224) and the channels were set to RGB while the mean (123.68, 116.779, 103.939) was subtracted as given in the original implementation by the authors. For resnet50 image dimensions were set to (3,224,224). For Inception v3 image, dimensions were set to 3 by 299 by 299.
Wireframe was built manually to store the location of the parking spots in an image captured by the camera. Each parking space from the wireframe was a cropped out of the image. These cropped out images were given to the model to predict whether the image is of an empty space or whether a car is parked there. Some of the examples of the wireframe are shown below:

Figure 4: **Examples of wireframe**

8.2 Architecture of the CNN

Complete architecture structure of each models implement are shared in the appendix.

8.2.1 VGG

VGG16 has 13 convolution layers with filter size increasing from 64 to 512. Every convolution layer is further analysed by Max pooling layer. In total, there are two fully connect dense layers of size 4096 with a drop out of 50%. The last layer is a dense layer which has an equal magnitude as to the classes; i.e. 2. Softmax is used as activation function for last layer. For all previous layers, Relu is used as activation function. Receptive field of size 3x3 is used with a stride of 2. The network architecture of VGG19 is very similar to vgg16 but with more convolution layers. (Simonyan and Zisserman, 2014)

8.2.2 Resnet

Resent is a branched convolutional network where the input to a block of convolution layers is added to the output of that block. This help in combatting the saturation that might occur due to batch normalisation. There are five stages in Resnet. After each convolution layer there is batch normalised. Relu is used as activation. Stride of 1 step was used in first layer and 2 for the rest. (He et al., 2016)

8.2.3 Inception

According to (Valigi, 2017) " The key insight about this architecture was to realize that conventional convolutional "filters" can only learn linear functions of their inputs. Why not increase their learning abilities and abstraction power by having more complex "filters"? This paper gets rid of the linear convolutions that are the bread and butter of CNNs and instead connects convolutional layers through multi-layer perceptrons that can learn non-linear functions. Luckily, it turns out that these perceptrons are mathematically equivalent to 1x1 convolutions, and thus fit neatly within the CNN framework." (Szegedy et al., 2015)

8.3 Technical Approach

We used Keras v2 (Keras.io, 2017) to build and train CNNs with backend theano (Theano: A Python framework for fast computation of mathematical expressions, 2016) to compile code for GPU. For various other scripts, such as compressing the dataset, running cross validation, etc., we typically used Python v3+ with NumPy. Anaconda[ref] was used as the packaging framework for python. We used the OpenCV library (Opencv.org, 2015) for better understanding the dimensions of each camera. Initially weights were taken from Caffe model zoo (Caffe.berkeleyvision.org, 2017).
Nvidia GTX 970 Single Precision Arithmetic's was used as the graphical processing unit, with CPU intel core i5, ram 4 GB.
On our system the classification of 1 parking spaces takes about 0.01 seconds. The figure below shows an example of our approach being able to deal with various visualisation challenges such as variant lighting, obstacles etc.
Figure: Example of difficult parking lot classifications

9 Training

For all classification tasks, the primary metric used was the classification accuracy, which is the fraction of correctly classified examples divided by the total images classified. Training was performed by optimizing the softmax loss using the Adam optimizer (Kingma, D. and Ba, J., 2014) with the learning rate of 0.0001. Step per epoch . Only the last convolution layer and the dense layers were trained. Because the initial convolution layers encode primitive features like edges and they had been trained by a huge data set from ImageNet. The initial weight that had been trained by the huge data set of ImageNet were used because they had been trained to perform a similar task of identifying patterns in image data. To avoid unsteady gradients in deeper neural networks, appropriate adjustments were performed. When batch normalization was conducted, after and before each layer activation the original weight was applied with training being carried for at least 8 epochs for all systems. The determination of initial sizes of batches and the learning rate was conducted through a grid search leading to a learning rate of 0.0001 and a mini-batch size of 16 being ideal with the largest decrease in training loss within the given epochs. A decay factor of 0.1 every 2 epochs was used to adjust the learning rate. In figure below accuracy is seen to be improve at each epoch. For tests conducted on particular dataset 90% of the dataset was used in direct correspondence to examples for training. For validation, 20% of these examples were used and training was stopped when the loss of validation and accuracy reached maximum epochs. The rest 10% of the data was used testing. In case of weather testing all the data belonging to weather was used in training i.e. 90% for training and 10% for validation and all the data points from other weather conditions was used for testing. For tests conducted to gauge diversity in orientation CNNs were trained on one dataset and tested with data from other two datasets as the camera angles vary across datasets.

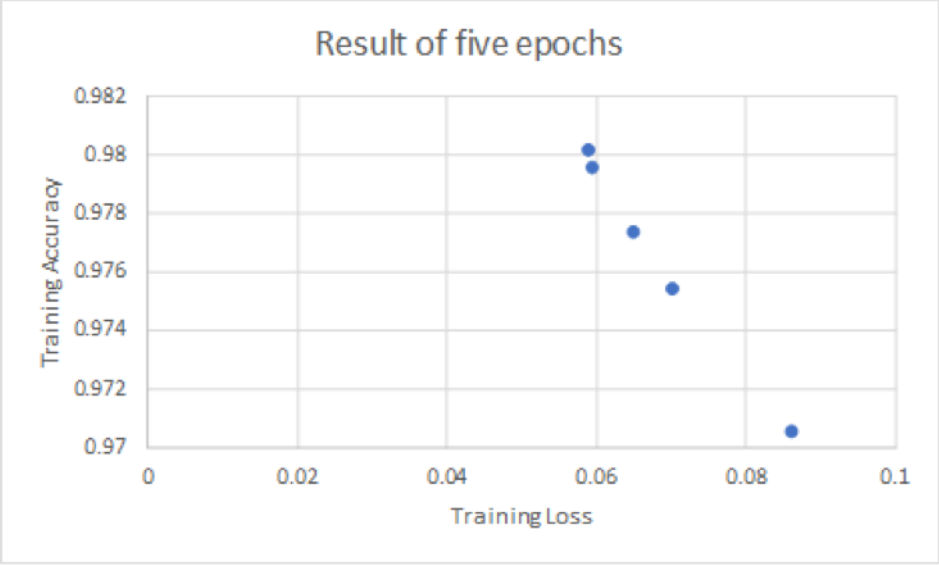


Figure 5: **Loss vs accuracy for five epoches on sample data**

labeling-tool

10 Testing and Evaluation

This section discusses the methodology used to evaluate the results compiled from experiments:
1. How the speed to training and prediction of each approach compares?
2. How much does the architectures over-fits or under-fits?
3. How much can the architectures generalize to different weather(light) conditions?
4. How robust are the architectures to change in viewpoints?
"missing"
For testing, the remaining samples that were not used for training were instead used as the test set. The images used at test time are preprocessed in the same way as the training images. The test images were also pre-segmented and cropped.
The images from a single day could only belong to only one of the sets namely training, testing or validation. This ensured that if a car has been parked at a certain parking spot for a longer period of time and is undergoing only weather difference (for example if the car is parked in the sun during the day and remains there till evening when it starts to rain) will not create any errors in the training set.

10.1 Performance Estimation

Overall Error Rate equation can be used to calculate the performance of each classifier. It divides the sum of test cases with misclassified label with total number of test cases. Which is given by the formula:

$$\frac{FP+FN}{TP+TN+FP+FN}$$

Figure 6: **2x2 confusion matrix**

where, FP,FN,TP, TN = False Positive, False Negative, True Positive, and True Negative. This can also be shown by the 2 by 2 confusion matrix as following:

Figure 7: **2x2 confusion matrix**

Data Set	Test Number	Total Training Images	Total Validation Images	Total Test Images
CNRExtInternal	test 1	94563	15449	26615
PKLotInternal	test 2	61248	17498	8748
SSRLotInternal	test 3	21646	6182	3090
Cloudy	test 4	59020	14754	154170
Rainy	test 5	38673	9667	179604
Sunny	test 6	84665	21165	122114
Pklot	test 7	44799	42695	140450
CNRExt	test 8	73018	63566	91360
SSRLot	test 9	3094	772	224078

Table 3: Training and testing sets for each test

11 Results and Discussion

As mentioned previously, the classification accuracy was used as the primary metric for all tasks. All classification results are summarized in table below. The best classification accuracies are highlighted in bold. Generally, CNN's with greater depth lead to better classification accuracies through validation set. This is because of the increased parameters that are being introduced to the model with each additional weight layer. The following sections provide detailed comparisons between the models for each classification task.

11.1 Measurement for performance

To measure the performance of the models accuracy was used. Which is defined as the total misclassification divided by total images classified. For most of the test CNNs preformed really better than any other approach in literature that used feature based approaches. For comparison the performance of each model on each test is given in table below. Using a simple CNN for this task is therefore a successful approach and has the advantage that no features need to be defined. It is also robust to overfitting, scoring over 99% for the training, validation and test set, and converges within one training epoch was achieved.

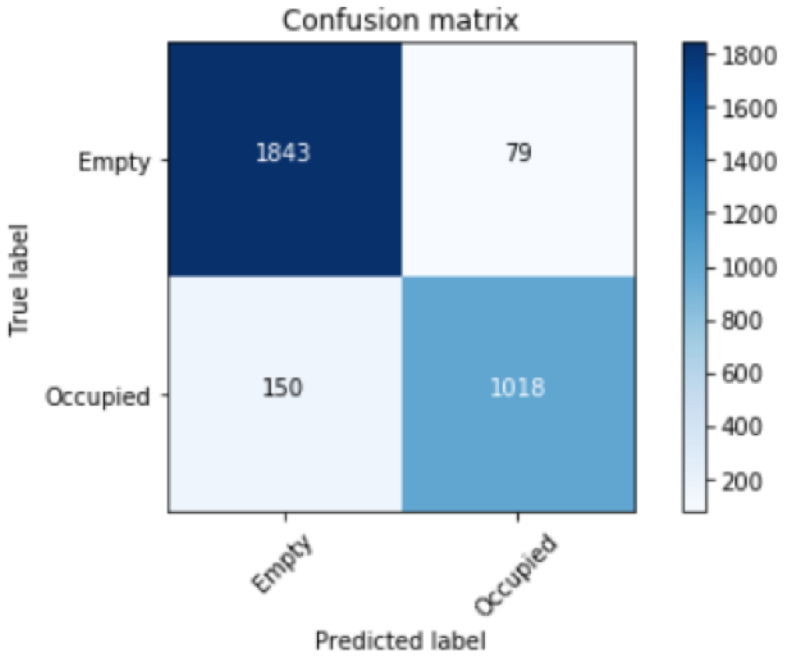


Figure 8: **Confusion Matrix for inception v3 for test 3**

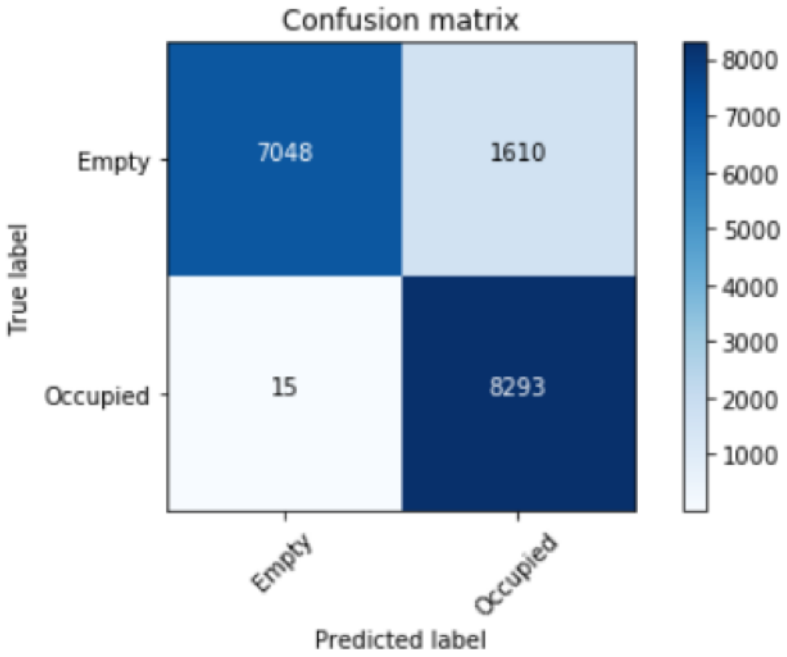


Figure 9: **Confusion Matrix for Resnet50 for test 1**

11.2 Generalization

To test how well our model generalizes across different parking lots or weather conditions, we trained our net on a specific condition and tested how well it performed on the other conditions. The details about dataset distribution are given in table below. More specifically, we trained it on each of the three weather conditions in our training set, namely sunny, rainy, and cloudy, and then tested the test performance on all other weather conditions for each of those, rendering a confusion matrix in figure below.
It can be seen that the performance on the test set varies as the weather conditions are changed, hinting that when the net is trained on the entire dataset, it learns to adapt to different lighting conditions. However, even when trained on one lighting condition, the net stays above 90% in all cases, which is fairly robust. The best performance over all weather conditions was obtained when training on sunny
In a second round of tests, we tested how well the net generalizes over images of different parking lots. Here, the main differentiator between the images is the camera angle, resulting in a different perspective distortion. The images in our dataset are drawn from three different parking lots (namely CNRExt, PKLot and SSRLot). The results can be seen in table below. The CNN also performed reasonably well, though not as well as during the weather test. This suggests that the CNN is more sensitive to changes in the viewing angle of the parking spaces than it is to changes in lighting.
In the third round of tests test we tested on 80% data taken from same parking lot and test it on reaming 20% data from same parking lot. This showed how well the model preforms if they were trained specifically particular parking lots.
Experiments on CNRPark revealed that it was more difficult to given classification on our dataset as compared with PKLot. It was also challenging training on single camera from SSRLot because individually the training examples were not enough.

11.3 Results

Vgg19 performed poorly in all the tests. This was either due to initially weights used not properly trained or this networks needs more epochs to train well.
1. How the speed to training and prediction of each approach compares?
Inception v3 took the least time to train on larger datasets and gave higher accuracy results. Vgg19 took the most time to train per epoch and gave the least accurate reuslts with error rate around 30%. Vgg16 and Resnet50 took roughly equal amount of time to train.
2. How much does the architectures over-fits or under-fits?
By comparing the accuracy in train and validation one can judge weather the model is overfitting or underfitting. If the training accuracy is low this mean that the model is under fitted. This can be fixed by using augmented data like rotating , skewing or including some distortion to generate more data entries for the model to train on. If the training accuracy is more that the validation accuracy that mean that the data is overfitting. This can be fixed by increasing the learning rate or increasing the drop out. This can also be fixed by using normally distributed data for training. Except vgg19 all there models were ideally fit for most of the tests. In test 4 models were underfitted due to the fact the SSRLot is the smallest dataset. But even than vgg16 , resnet50 and inception v3 preformed really well getting an error rate of 6.28%, 4.63% and 7.41% respectively.
3. How much can the architectures generalize to different weather(light) conditions?
Things like the fluctuations in lighting conditions, occlusions and reflection along with weather conditions can cause errors in the detection in the availability of parking spaces. Not only this

but the low generalization properties of classifiers can also create errors; an important point being the classifiers trained on a low variation of camera angles will be unable to detect different angles. If the classifier is not generalizing to an acceptable level then it will only work in the conditions it was trained for, creating all these errors.
Resnet50 preformed the best when it was just trained on cloudy images and tested on images from Rainy and Sunny datasets. It had an error of 0.01%. Vgg16 and Inception v3 preformed fairly each with error of 2.79% and 6.31% respectively
4. How robust are the architectures to change in viewpoints?
We trained models on cameras from one data set and tested it on the camera angles from different dataset. When the model were trained on just SSRLot which is the smallest datasets of the three and tested on complete other sets. All models preformed really well they had an average misclassification error of 3.4%. Vgg19 had the lowest error of 3.02% and resnet50 had the highest error of 4.15%.
5. How the performance time of the models compare?
Inception v3 performs well for larger datasets and takes less time to train on data set and to predict a label for an image.

Test	**Model**	**Training Accuracy(%)**	**Validation Accuracy (%)**	**Training Time per epoch (min)**	Error (%)	Time to perdict
1	vgg16	97.01	98.09	86.05	1.42	0.017
	vgg19	63.65	84.7	59.11666666667	16.84	0.018707011
	resnet50	9614	98.02	4368333333333	9.57	0.009333577
	Inception_V3	93.76	97.25	61.9833333333	2.96	
2	vgg16	95.2	99.07	64	1.0402	0.016060622
	vgg19	67.21	70.42	63	18.016	0.019084554
	resnet50	97.82	99.4	12.81666667	0.503	0.010424422
	Inception_V3	93.75	97.95	15.8333333	1.9662	0.012577641
3	vgg16	81.97	94.89	8.9333333	6.2783	0.118527277
	vgg19	58.42	63.72	10.85	34.919	0.018523019
	resnet50	88.51	95.32	4.2	4.6278	0.013200149
	Inception_V3	82.11	92.38	24.86666667	7.411	.0.026559849
4	vgg16	95.45	98.61	43.4	2.7904	0.01657707
	vgg19	62.64	78.12	40.816666667	310359	0.018757431
	resnet50	95.31	98.12	39.683333333	0.011	4.177855614
	Inception_V3	91.96	95.77	40.7333333	6.308	0.008949252
5	vgg16	34.4	38.09	26.51666666667	3.273312	0.017641132
	vgg19	55.54	58.33	23.74	37.92009	0.019755391
	resnet50	93.38	97.7	25.8	3.834547	.0009500253
	Inception_V3	30.2	35.24	23.2	5.782722	0.008950147
6	vgg16	-				
	vgg19					
	resnet50					
	Inception_V3					
7	vgg16	0.9479	0.9901	48.8	6.2948	0.015995954
	vgg19	0.6356	0.8641	282.16666667	44.769	0.120541457
	resnet50	96.84	99.29	51.2	17.91	0.0103895551
	Inception_V3	92.85	97.64	51.9	16.717	
8	vgg16	69.76	89.42	3.33	25.56	0.026
	vgg19	55.19	62.01	3.02	52.24	0.031
	resnet50	73.68	78.44	4.15	39.02	0.029
	Inception_V3	67.76	71.75	3.10	33.23	0.015

Table 4: My caption

12 Mobile Application

12.1 Background

One demo app was created to evaluate whether or not, getting real time information about parking lot is useful. Currently the procedure to navigate using google map is as follows:
1. Google Maps is opened
2. It shows the initial position of the user
3. User enter a location e.g. Australian National University
4. Map jumps to that location
5. The standard google map allows the functionality to edit route at this point by dropping pin and it also allows to zoom into the map
6. On this screen we showed general polygon of three colours green (less than 70% is empty), yellow (more than 70% and less than 100% is free) and red (if there are no free parking spots)
7. User to could click the parking lot and see the individual parking spots that are free.
8. At both of the previous steps user can choose to stick with the route and start navigation or he/she could drop pin and change the route.

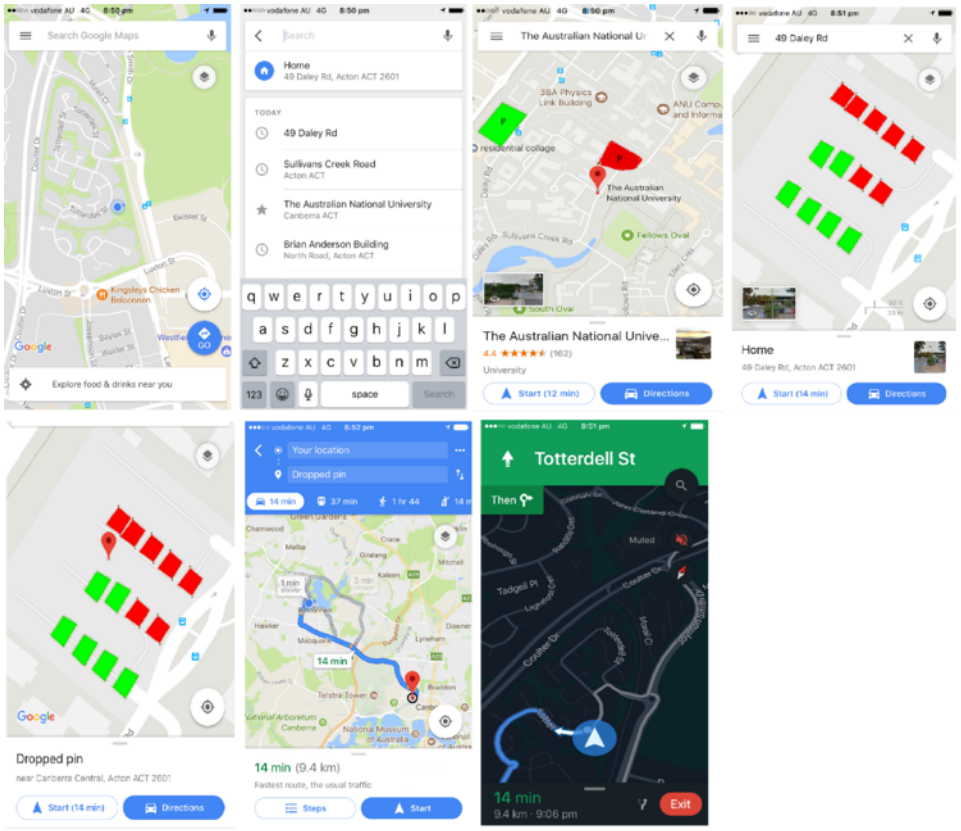


Figure 10: **Top Left 1. Enter address 2. Selects destination i.e. Australian National University 3. Initial way pin is dropped in the centre of Australian Nation University 4. User selects the green parking lot 5. Map is zoomed in to show empty parking spaces 6. User drops pin to change his/her destination 7. User starts navigation**

12.2 Survey findings

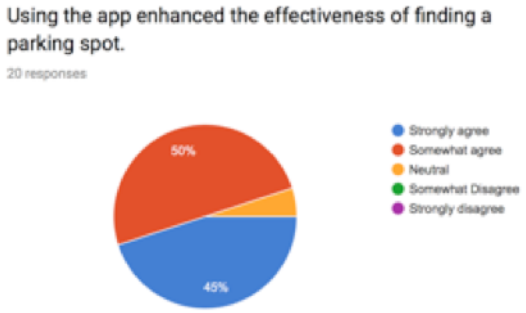
Ethical clearance was taken from the Ethics Department at Australian National University for this study. The concent form, particpant information sheet and results are attached in the appendix.

12.2.1 Background of respondents

Due to limited time constrain, 21 responses were received within 24 hours after the survey was made available to the public. People who answered the surveyed belonged to the age group of between 18 till 40, 52.4% of them were females and 47.6% of them were male which ensures that the survey results are not gender biased. Majority of the respondents have high school or equivalent qualification (42.9%), the remaining have a higher educational background which implies that respondents go to an educational institute and are the potential target users for the app (as 60% of them use a car on either on daily basis or weekly basis to commute and hence would use the parking app potentially in the future when travelling to their educational institute) and the respondents did not face any difficulties in understanding how to use the technology of the application.

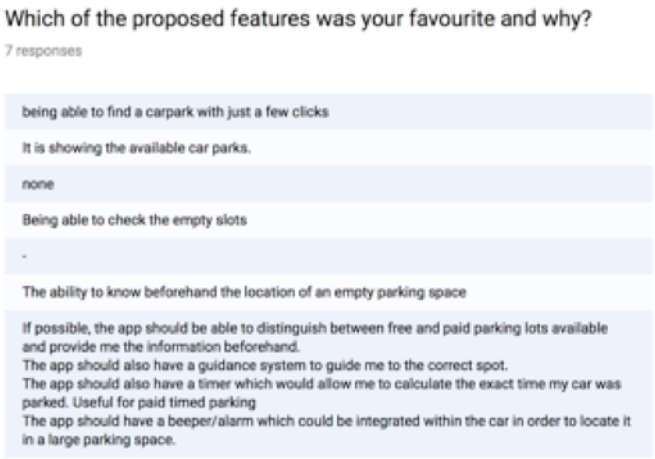
12.2.2 Findings

The result of the survey shows that most majority of the people prefer to use their personal car as a mean of transport as compared to any other mean of transportation, out of which 14.3% of the people never use public transportation.
80.9% of the users agreed that using the app will help them find a car park faster which implies that currently there is a strong demand for such an application and students are finding difficulties in finding a parking space in their educational institutes. 71% of the users believed that the app will improve the way they plan their travel route, 23.8% were neutral and the remaining disagreed, which shows that the even though the application has a huge potential and it is something that users demand but in order to build their trust (in ensuring them that the application will be able to work in real time and prove to be beneficial to them) a prototype needs to be launched so that users can experience its benefit first handed. This can further be justified because 95% of the respondents believed that using the app enhanced the effectiveness of finding a parking spot (as shown in the figure below).



12.2.3 Conclusions

Overall the feedback from the respondents was positive and they showed excitement for such an application to be developed. 80.9% of the respondents agreed that it was useful for searching a parking spot, majority agreed that the app was easy to use/interact with/understand and they were able to do what they wanted out of it which implies that not only such an application is easy to use but it is also filling in the gap of what users want to accomplish (find a parking space when they travel).
The following tables show the feedback collected from respondents regarding the app.



The feedback shows that respondents are excited for such an application to be launched and they even provided thoughtful insight on how to incorporate further features into it to make it even more beneficial for them. 83.8% of the respondents agreed on seeing themselves using such an application for their day-to-day commute and 80.9% of them plan to begin using such an app if made available, which means that there is an urgent need to cater to the present demand.

13 Conclusions

CNN are a well establish solution for this problem. As demonstrated in the results they have preformed quit well in identifying occupancy of parking spots. A clear market need is established by survey. This is a viable solution we will be working on to implement it.

13.0.1 Future work

The model has numerous future implications and have potential room for improvement, some of which are discussed below.

13.0.2 Additional App Features

Allowing users to add information about when they are about to leave will benefit other users in planning there trips. Adding information about paid and free parking spots. Notifing users when their parking is running.

13.0.3 Satellite Imagery

By using satellite imagery, better resolution of images can be gathered, the images would incorporate a more generalized top-down view and thus by increasing the sample size of the data a higher accuracy model can be achieved. However, to implement such an approach more resources would be required.

13.0.4 Reduced Positional Data & Attention Models

The current model adopted an approach in which four coordinates are required to map a box image at each parking space to identify it. This can be further improved by using only one coordinate to image each parking space (Tang, Srivastava and Salakutdinov 2014). The modified approach, 'Attention model', involves identifying the central coordinate of each parking space and mapping the parking space using zero coordinates into CNN under a binary classification. Such an approach would also require more resources to build up an intelligent model that can identify the parking spaces with minimum data for the coordinates in each space.

13.0.5 Parking Space Detection

Another effective approach that can be used is to identify the parking spaces using detection systems. Detection models have previously been adopted for many applications and have proven to be most efficient. However, a lot of further development would be required in order to make such a model work because it does not support rotation of bounding boxes. Building upon the 'Spatial Transformer Model' (Jaderberg et al. 2015), this problem can be solved. It would make the rotation of the bounding boxes stationary and would open the door to be able to applications that can use live camera feed. This approach is very challenging and would require extensive amount of research before it can be implemented.

13.0.6 Smart Camera System

The use of smart camera system is one of the most effective approaches as it would decrease the communication between the computer and the parking spaces, making the model very intelligent.

The model will identify empty spaces automatically, making it easier to use. The only downside being a requirement for bigger data.

13.0.7 Single Parking Lot Training and Testing

This method involves studying a single parking lot extensively, identifying all the classifiers using a pool of images from it, and creating a training set. This training set would then be used to create more training subsets ensuring the error is close to minimum. Even though such a model is highly accurate, but it would require a lot of time to develop it.

13.0.8 Single and Multiple Parking Lot Testing

Method involves focusing on one parking lot and then implementing the findings to multiple parking lots. This will reduce the time constraint as only one parking would be required to be fully studied, identifying all the classifiers for it and then extending them to be generalized over other parking lots. This technique involves creating a training set and then using those matrices to mirror on other testing sets. Extensive analysis will need to be carried out in order to ensure that the training set has no errors and can be mirrored onto other testing sets.

13.0.9 Multiple Parking Lot Training

Another approach is to study multiple parking lots to then use the data set as the basic training set. The trained classifiers will represent a variety of variables like different parking lot surfaces, different angles, different temperature settings etc. The method involves developing multiple training sets by using multiple training subsets, which will give it high accuracy however a lot time would need to be invested in creating such a model.

13.0.10 Visual detection

The technique involves installing multiple cameras at the parking lot site, using the 'Raspberry Pi Platform', which can identify any vacant parking space. Even though it would require greater resources for implementation, but the data gathered would be highly accurate. Such a deep level CNN design would ensure that every vacant parking spot is identified in real time making it most useful to end users. The CNN architecture would work on binary coding, making it a simpler model to design, however, the implementation requires more resources.

14 References

Choeychuen, K. (2013). Automatic parking lot mapping for available parking space detection. 2013 5th International Conference on Knowledge and Smart Technology (KST). [online] Available at: http://ieeexplore.ieee.org/document/6512799/ [Accessed 25 Oct. 2017].

Al-Kharusi, H. and Al-Bahadly, I. (2014). Intelligent Parking Management System Based on Image Processing. World Journal of Engineering and Technology, [online] 02(02), pp.55-67. Available at: scirp.orgpdfWJET_2014050611391380.pdf.

Jakle, J. and Sculle, K. (2005). Lots of parking. 1st ed. Charlottesville: University of Virginia Press.

Krizhevsky, A., Sutskever, I. and Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. 1st ed. [ebook] Curran Associates, Inc., pp.1097-1105.

Simonyan, K. and Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. CoRR, [online] abs/1409.1556. Available at: http://dblp.org/rec/bib/journals/corr/SimonyanZ14a [Accessed 25 Oct. 2017].

van der Waerden, P., Janssens, D. and da Silva, A. (2017). The influence of parking facility characteristics on car drivers' departure time decisions. Transportation Research Procedia, [online] 25, pp.4058-4067. Available at: http://www.sciencedirect.com/science/article/pii/S2352146517306312.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: CoRR abs/1502.03167. URL: http://arxiv.org/abs/ 1502.03167.

Jaderberg, Max et al. (2015). "Spatial Transformer Net- works". In: CoRR abs/1506.02025. URL: http : / / arxiv.org/abs/1506.02025.

Tang, Yichuan, Nitish Srivastava, and Ruslan R Salakhut- dinov (2014). "Learning generative models with visual attention". In: Advances in Neural Information Processing Systems, pp. 1808–1816.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: CoRR abs/1502.03167. URL: `http://arxiv.org/abs/1502.03167`.

De Almeida, P.R., Oliveira, L.S., Britto, A.S., Silva, E.J. and Koerich, A.L., 2015. PKLot–A robust dataset for parking lot classification. Expert Systems with Applications, 42(11), pp.4937-4949.

Amato G, Carrara F, Falchi F, Gennaro C, Meghini C, Vairo C. Deep learning for decentralized parking lot occupancy detection. Expert Systems with Applications. 2017 Apr 15;72:327-34.

Davami, E. and Sukthankar, G., 2015, May. Improving the performance of mobile phone crowdsourcing applications. In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (pp. 145-153). International Foundation for Autonomous Agents and Multiagent Systems.

Zhang, C., Dong, H., Jia, L., Qin, Y. and Yang, Z., 2017, September. Robust vehicle detection and identification with single magnetic sensor. In Intelligent Transportation Engineering (ICITE), 2017 2nd IEEE International Conference on (pp. 94-98). IEEE.

Ling, X., Sheng, J., Baiocchi, O., Liu, X. and Tolentino, M.E., 2017, June. Identifying parking spaces & detecting occupancy using vision-based IoT devices. In Global Internet of Things Summit (GIoTS), 2017 (pp. 1-6). IEEE.

Keras.io. (2017). Keras Documentation. [online] Available at: `https://keras.io` [Accessed 27 Oct. 2017].

Theano: A Python framework for fast computation of mathematical expressions. (2016). arXiv

e-prints, [online] abs/1605.02688. Available at: http://arxiv.org/abs/1605.02688 [Accessed 27 Oct. 2017].
Opencv.org. (2015). OpenCV. [online] Available at: http://opencv.org/[Accessed 27 Oct. 2017].
Caffe.berkeleyvision.org. (2017). Caffe — Model Zoo. [online] Available at: http://caffe.berkeleyvision.org//model_zoo.html [Accessed 27 Oct. 2017].
He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).
Valigi, N. (2017). Short history of the Inception deep learning architecture - Nicolò Valigi. [online] Nicolovaligi.com. Available at: http://nicolovaligi.com/history-inception-deep-learning-architecture.html [Accessed 27 Oct. 2017].
Kingma, D. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
Shoup, D.C., 2006. Cruising for parking. Transport Policy, 13(6), pp.479-486.
Caffe.berkeleyvision.org. (2017). Caffe — Model Zoo. [online] Available at: http://caffe.berkeleyvision.org/model_zoo.html [Accessed 27 Oct. 2017].
Labeler, T. (2017). Label Images for Classification Model Training - MATLAB & Simulink - MathWorks United Kingdom. [online] Au.mathworks.com Available at: https://au.mathworks.com/help/vision/ug/label-images-for-classification-model-training.html [Accessed 27 Oct. 2017].
K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. CoRR, abs/1502.01852, 2015.
S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR, abs/1502.03167, 2015. D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. CoRR, abs/1412.6980, 2014.
En.wikipedia.org. (2017). Universal approximation theorem. [online] Available at: https://en.wikipedia.org/wiki/Universal_approximation_theorem [Accessed 28 Oct. 2017].

15 Appendix - Personal Information Sheet

Participant Information Sheet

Researcher:
Syed Gardezi, Master of Computing student, Research School of Computing, Australian National University

Project Title: Mobile App: Smart Car Parking System; Finding empty parking spots using computer vision and deep learning

General Outline of the Project:

- **Description and Methodology:** The purpose of this research project is to investigate the usefulness of a mobile app that gives live updates about free parking spaces in a given parking lot. The intent of the research project is to measure the benefits of such a utility to motorists. You will be provided with a survey form with a structured questionnaire to record your experience of using the application.

- **Participants:** The main component of the research will be a survey of approximately 20 people who own a car. They will be presented with questions asking whether such a mobile application will be useful for them.

- **Use of Data and Feedback:** The data from the survey findings will be used in the development of my master thesis along with using it as a reference in conference and journal publications. A password protected link will be shared with testing users who can use this link to find the outcomes of this survey in a neatly presented report. This link will be live; data will be updated on it, once processed. https://drive.google.com/open?id=0B8iTmoLLdsoEZ2FOQzVNZnlGRGM

Participant Involvement:

- **Voluntary Participation & Withdrawal:** You have received this invitation to participate in this research as you have indicated your interest in the development of this project. Participation in this survey is entirely voluntary and the identity of those who participate and those that decline to participate will not be recorded. Participants may withdraw from the research at any time without providing an explanation and their data will be destroyed if requested. The testing user can refuse to answer any individual questions they wish without negative consequences.

- **What does participation in the research entail?** The survey will be made available to you online. Neither your name nor other identifying details will be requested, beyond some demographic information regarding your age, gender and what you are (or were) studying. You will be asked to preform a task i.e. use the app to find a free parking spot in a said area and then answer a few questions about your experience.

- **Location and Duration:** The survey will take place in a meeting room in CSIT building. It will take not more than 15 minutes. While you use the mobile application, you will be observed to note what are the natural expectations from the application and on which step of the task most time was consumed.

Australian National University

- **Risks:** You should be aware even with best of my ability to aggregate data to hide individual data, 3rd parties in conjunction with other sources of data may be able to identify individual compromising their privacy.

- **Benefits:** The benefit of this study will be to establish a genre and demand for this technology and take it further in the direction of development needed. If this study is a success, then a mobile application could be developed with real time information available about parking spaces. This will not only lead to effective route planning but also save fuel, save time, reduce stress but also will contribute to a reduction in carbon footprint.

Confidentiality:

- **Confidentiality:** The survey findings will be used in the development of my master thesis and I hope to present them in one or more conference presentations and published journal articles. I will ensure any organisations that have distributed an invitation to participate in the survey also have an opportunity to review key findings and have the option to distribute such findings to participants. I will be unable to do that directly as the contact details of student participants will be unknown to me. Neither your name nor other identifying details will be requested, beyond some demographic information regarding your age, gender and what you are (or were) studying.

Privacy Notice:

In collecting your personal information within this research, the ANU must comply with the Privacy Act 1988. The ANU Privacy Policy is available at https://policies.anu.edu.au/ppl/document/ANUP_010007 and it contains information about how a person can:

· Access or seek correction to their personal information;
· Complain about a breach of an Australian Privacy Principle by ANU, and how ANU will handle the complaint.

Data Storage:

- **Where:** The data collected through this survey will be collated in an electronic database, held securely within the Australian National University and otherwise on a password-protected laptop. I intend to analyse the data and may publish some or all of it (in its de-identified form).

- **How long:** The data will be kept for 5 years after the date of collection.

- **Handling of Data following the required storage period:** The data will be archived and stored in a backup password protected hard drive by me. The data will be in de-identifiable format. The purpose will be to provide future researchers accesses to the material collected so they may use it in their work.

Queries and Concerns:

- **Contact Details for More Information:** Please feel welcome to contact me directly, as principle investigator of this research project:
  Syed Gardezi, Master of Computing, Australian National University.
  Phone: 0451440429 u5752631@anu.edu.au

  You are also able to contact my academic supervisor:
  Professor Jacqueline Lo
  Phone: (02) 6125 6778 Jacqueline.Lo@anu.edu.au

Ethics Committee Clearance:

The ethical aspects of this research have been approved by the ANU Human Research Ethics Committee (Protocol 2017/729). If you have any concerns or complaints about how this research has been conducted, please contact:

Ethics Manager
The ANU Human Research Ethics Committee
The Australian National University
Telephone: +61 2 6125 3427
Email: Human.Ethics.Officer@anu.edu.au

16 Appendix - Written Consent

Australian National University

WRITTEN CONSENT for Participants

Mobile App: Smart Car Parking System; Finding empty parking spots using computer vision and deep learning

I have read and understood the Information Sheet you have given me about the research project, and I have had any questions and concerns about the project (listed here

_____

_____)

addressed to my satisfaction.

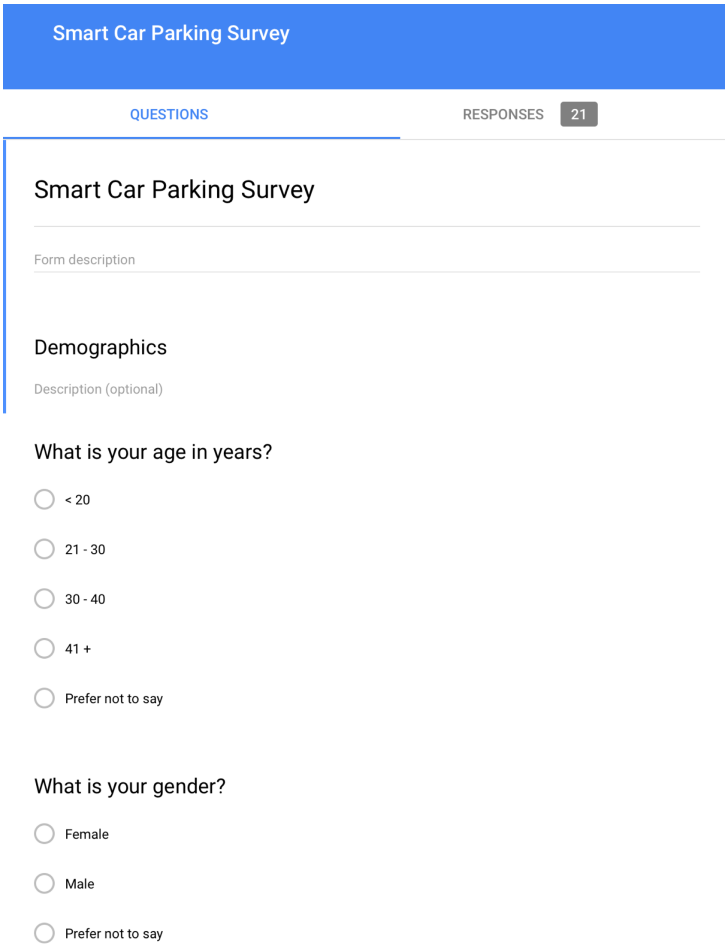
I agree to participate in the project. YES ☐ NO

☐

Signature:……

Date:……

17 Appendix - Questioner



- ○ Other...

What is your highest level of education completed?

- ○ High school or equivalent
- ○ Vocational or diploma qualification (Certificate I - IV, diploma, associate diploma/degree)
- ○ Bachelor degree (includes honours)
- ○ Masters degree
- ○ PhD
- ○ Other...

Car Use

Description (optional)

How often do you use a car?

- ○ Daily
- ○ Weekly
- ○ Fortnightly
- ○ Monthly
- ○ Quarterly
- ○ Biannually
- ○ Annually
- ○ Less than once a year

Which mode of transport do you usually prefer to use within the city?

- [ ] Uber
- [ ] Taxi
- [ ] Personal car
- [ ] Bicycle
- [ ] Motorbike
- [ ] Walking
- [ ] Bus

How often do you use a public car park?

- ( ) Always
- ( ) Often
- ( ) Sometimes
- ( ) Seldom
- ( ) Never

Assessment of the proposed application

Description (optional)

Using the app will enable me to park my car faster.

- ( ) Strongly agree

- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat disagree
- ○ Strongly disagree

Using the app will improve the way I plan my travel.

- ○ Strongly agree
- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat Disagree
- ○ Strongly disagree

Using the app enhanced the effectiveness of finding a parking spot.

- ○ Strongly agree
- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat Disagree
- ○ Strongly disagree

Using the app will make searching for a parking spot easier.

- ○ Strongly agree

- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat Disagree
- ○ Strongly disagree

The app was useful for searching a parking spot.

- ○ Strongly agree
- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat Disagree
- ○ Strongly disagree

Learning to use the app was easy.

- ○ Strongly agree
- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat disagree
- ○ Strongly disagree

It was easy to get the app to do what I wanted.

- ○ Strongly agree

- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat disagree
- ○ Strongly disagree

Interaction with the app was easy to understand.

- ○ Strongly agree
- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat disagree
- ○ Strongly disagree

The app was flexible to interact with.

- ○ Strongly agree
- ○ Somewhat agree
- ○ Neutral
- ○ Somewhat disagree
- ○ Strongly disagree

It is easy for me to become skillful at using the app.

- ○ Strongly agree

- Somewhat agree
- Neutral
- Somewhat disagree
- Strongly disagree

The app was easy to use.

- Strongly agree
- Somewhat agree
- Neutral
- Somewhat disagree
- Strongly disagree

Do you have any further comments about the app or its use experience?

Long answer text

Final comments

Description (optional)

In my opinion, a smart parking app supports searching better than the current practice.

- Strongly agree
- Somewhat agree
- Neutral

- Somewhat disagree
- Strongly disagree

The smart parking app will support me in my route planning task.

- Strongly agree
- Somewhat agree
- Neutral
- Somewhat disagree
- Strongly disagree

I see myself using app of this kind in my day-to-day commute.

- Strongly agree
- Somewhat agree
- Neutral
- Somewhat disagree
- Strongly disagree

I plan to begin using apps of this kind in my life.

- Strongly agree
- Somewhat agree
- Neutral
- Somewhat disagree
- Strongly disagree

Which of the proposed features was your favourite and why?

Long answer text

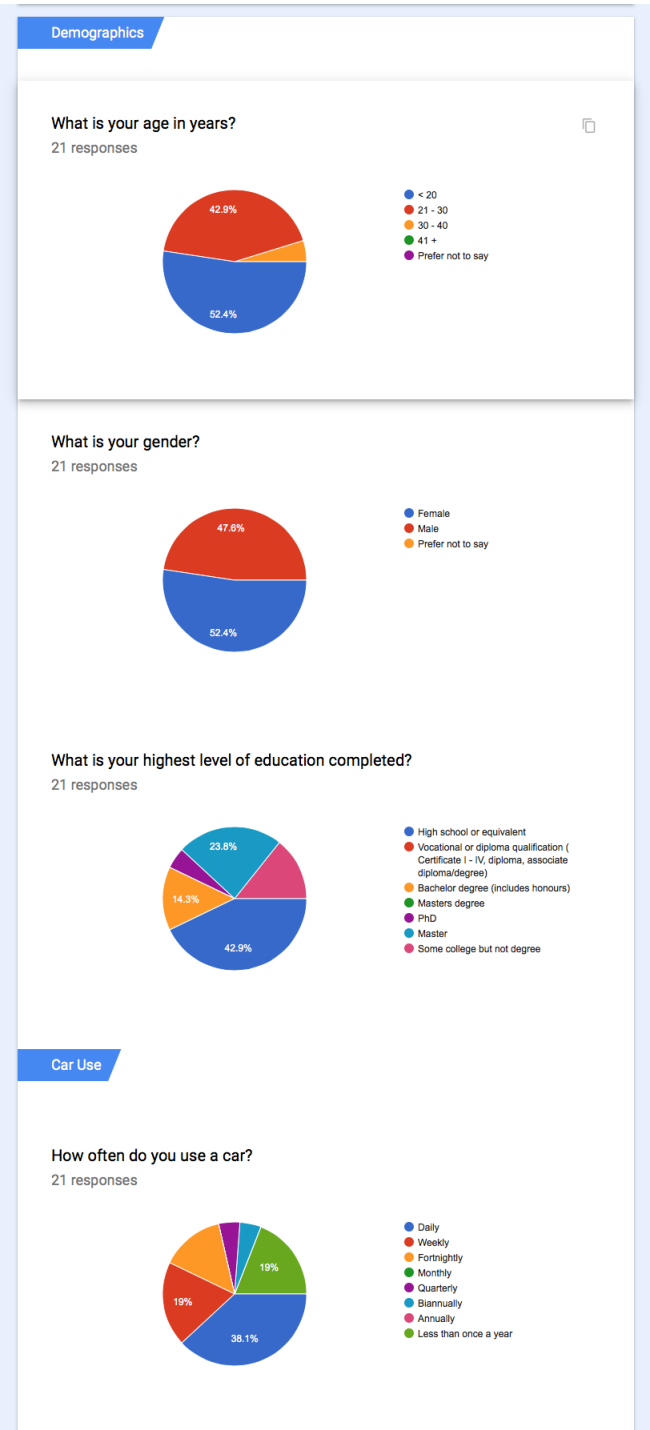
Survey feedback

Description (optional)

Do you have any questions or comments about the survey?

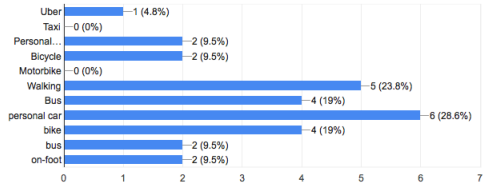
Long answer text

18 Appendix - Survey Results



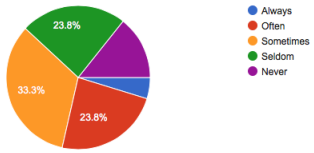
Which mode of transport do you usually prefer to use within the city?

21 responses



How often do you use a public car park?

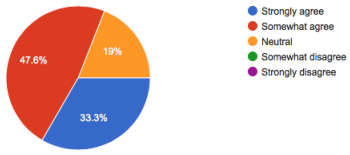
21 responses



Assessment of the proposed application

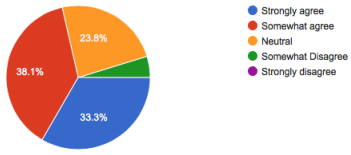
Using the app will enable me to park my car faster.

21 responses



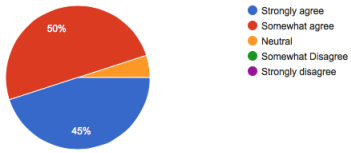
Using the app will improve the way I plan my travel.

21 responses



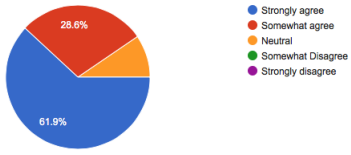
Using the app enhanced the effectiveness of finding a parking spot.

20 responses



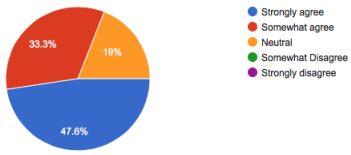
Using the app will make searching for a parking spot easier.

21 responses



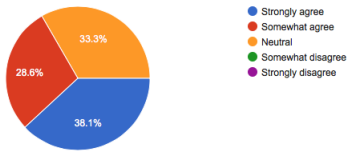
The app was useful for searching a parking spot.

21 responses



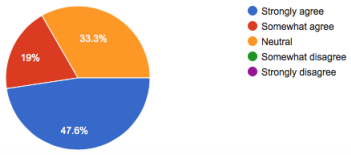
Learning to use the app was easy.

21 responses



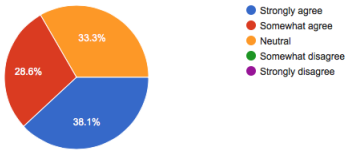
It was easy to get the app to do what I wanted.

21 responses



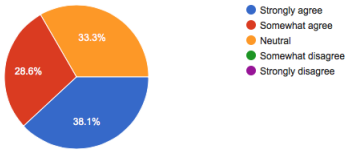
Interaction with the app was easy to understand.

21 responses



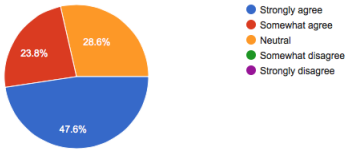
The app was flexible to interact with.

21 responses



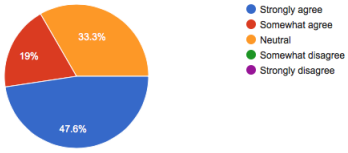
It is easy for me to become skillful at using the app.

21 responses



The app was easy to use.

21 responses



Do you have any further comments about the app or its use experience?

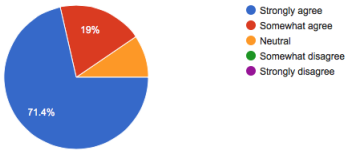
6 responses

When are you making it for ANU?
When is it coming in the market?
It was really obvious to use the application and i can't wait for it to become available to be used commonly.
no
-
Probably add a feature in-real time to update whether the parking is still available while travelling to it

Final comments

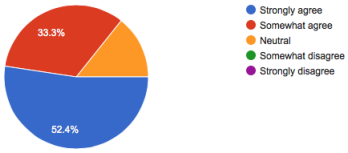
In my opinion, a smart parking app supports searching better than the current practice.

21 responses



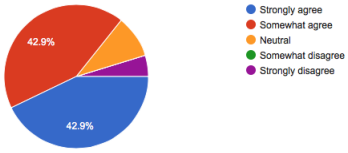
The smart parking app will support me in my route planning task.

21 responses



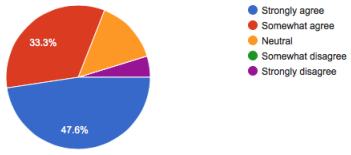
I see myself using app of this kind in my day-to-day commute.

21 responses



I plan to begin using apps of this kind in my life.

21 responses



Which of the proposed features was your favourite and why?

7 responses

being able to find a carpark with just a few clicks
It is showing the available car parks.
none
Being able to check the empty slots
-
The ability to know beforehand the location of an empty parking space
If possible, the app should be able to distinguish between free and paid parking lots available and provide me the information beforehand. The app should also have a guidance system to guide me to the correct spot. The app should also have a timer which would allow me to calculate the exact time my car was parked. Useful for paid timed parking The app should have a beeper/alarm which could be integrated within the car in

Survey feedback

Do you have any questions or comments about the survey?

2 responses

no
-

19 Appendix - Presentation

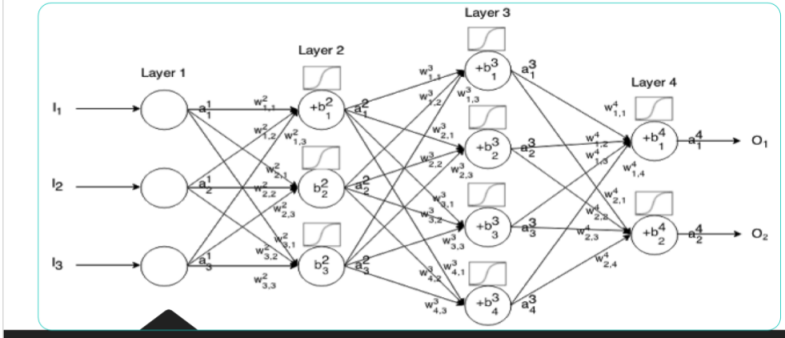
Smart Parking System using Deep Learning

Sheece Gardezi
Supervised By:
Anoop Cherian
Peter Strazdins

Australian National University

Content

- Labeling tool
- Neural Networks
- Visual
- Road Map
- Labeling tool
- Data set
- Vgg16
- Resnet50
- Inception_v3
- Test
- Real Time App

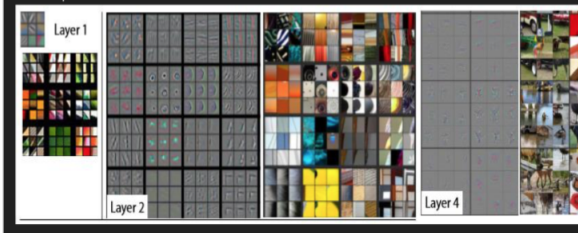


Neural Networks

- Infinitely flexible function
- All purpose parameter fitting
- Fast and scalable

Visualizing Convolutional Netw

Filters learn to find features from pictures and each filter activate shape of the filter



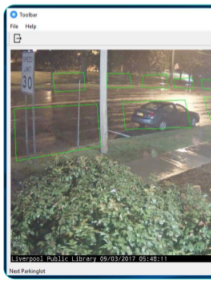
Ref: Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. InEuropean conference on computer vision 2014 Sep 6 (pp. 818-833). Springer, Cham.

Road Map

- Collect data
- Preprocesses Data
  - Fillter data – remove pictures with people, night time pictures
  - Create wireframe
  - Label data – create labeling tool
  - Segment data
- Create test cases
  - Create test/train/ validitation datasets for each case
- Implement Models
  - Vgg16, Vgg19, resnet50 and Inception_v3
  - Create mini-Vgg16
  - Train from scratch
- Evaluate
  - Run tests on models
  - Gadge their preformance

Labelling Tool

- It automatically create standard wireframe for each image
- Gamification



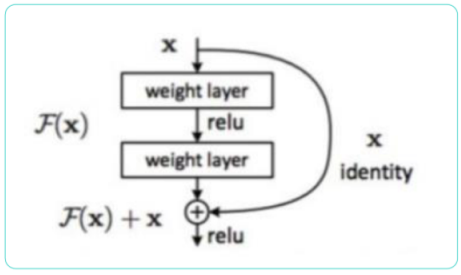
Dataset

Dataset	Free spaces	Busy spaces	Total Spaces
CNRPark	65,658	79,307	144,965
PkLot	337,780	358,119	695,899
SSRLot	17,334	11,247	28,581

Vgg

- Major Features:
  - 3x3 Convolution Layers Stacked
  - 16 weight layers
  - fixed image size 224 x 224 and three color channels RGB
  - Spatial pooling was carried out by five max-pooling layers each had 4096 channels
- Major drawbacks:
  - It is extremely slow to train and consumes a lot of time and resources of the machine.
  - The network architecture's weights themselves are quite large (in terms of the disk space/ bandwidth covered)

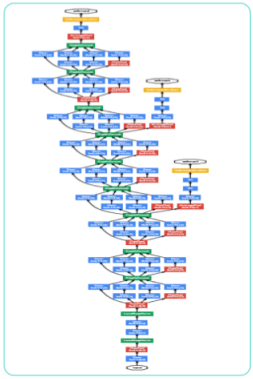
Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. 2014 Sep 4.



He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. InProceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 770-778).

Resnet

- As we go deeper; the training of neural network becomes difficult and also the accuracy starts saturating and then degrades also. Residual Learning tries to solve both these problems.
- Residual can be simply understood as subtraction of feature learned from input of that layer. ResNet does this using shortcut connections (directly connecting input of nth layer to some (n+x)th layer. It has proved that training this form of networks is easier than training simple deep convolutional neural networks and also the problem of degrading accuracy is resolved



Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. InProceedings of the IEEE conference on computer vision and pattern recognition 2015 (pp. 1-9).

Purpose of conducting tests of various models

- Different architectures have different strengths
- Using stochastic gradient decent and universal approximation theorem given enough time any model could be trained for any task
- Evaluate which model trains fastest for car parking problem
- Evaluate which model is more porous/flexible to changed light conditions
- Evaluate which model is more flexible to orientations of cars

Evaluation Method

Model	Number of Epoch	Training Time(s)	Training Accuracy(%)
Vgg16	1	3015	98.27
Vgg19	1	567	75.71
Resnet50	1	578	99.43
Inception-v3	1	622	99.9

Results for test case PkLot-Complete

Real Time information about the parking lot in google maps.

Comercial Limitation

- No cost reduction for parking lot owners
- No added benfit added to commerical entities
- Places for implemention
  - University where organization cares for its populous life improvement
  - Government owned parkinglots where object is to work for betterme

Questions

20 Appendix - Architectures

20.1 Vgg16

Layer (type)	Output Shape	Param #
lambda_layer_1 (Lambda)	(None, 3, 224, 224)	0
zero_padding2d_ layer_1 (ZeroPaddin	(None, 3, 226, 226)	0
convolution2d_ layer_1 (Convolution2D)	(None, 64, 224, 224)	1792
zero_padding2d_ layer_2 (ZeroPaddin	(None, 64, 226, 226)	0
convolution2d_ layer_2 (Convolution2D)	(None, 64, 224, 224)	36928
max_pooling2d_ layer_1 (MaxPooling2	(None, 64, 112, 112)	0
zero_padding2d_ layer_3 (ZeroPaddin	(None, 64, 114, 114)	0
convolution2d_ layer_3 (Convolution2D)	(None, 128, 112, 112)	73856
zero_padding2d_4 (ZeroPaddin	(None, 128, 114, 114)	0
convolution2d_ layer_4 (Convolution2D)	(None, 128, 112, 112)	147584
max_pooling2d_ layer_2 (MaxPooling2	(None, 128, 56, 56)	0
zero_padding2d_ layer_5 (ZeroPaddin	(None, 128, 58, 58)	0
convolution2d_ layer_5 (Convolution2D)	(None, 256, 56, 56)	295168
zero_padding2d_6 (ZeroPaddin	(None, 256, 58, 58)	0
convolution2d_ layer_6 (Convolution2D)	(None, 256, 56, 56)	590080
zero_padding2d_ layer_7 (ZeroPaddin	(None, 256, 58, 58)	0
convolution2d_ layer_7 (Convolution2D)	(None, 256, 56, 56)	590080
max_pooling2d_ layer_3 (MaxPooling2	(None, 256, 28, 28)	0
zero_padding2d_8 (ZeroPaddin	(None, 256, 30, 30)	0
convolution2d_ layer_8 (Convolution2D)	(None, 512, 28, 28)	1180160
zero_padding2d_9 (ZeroPaddin	(None, 512, 30, 30)	0
convolution2d_ layer_9 (Convolution2D)	(None, 512, 28, 28)	2359808
zero_padding2d_ layer_10 (ZeroPaddi	(None, 512, 30, 30)	0
convolution2d_ layer_10 (Convolution2D)	(None, 512, 28, 28)	2359808

max_pooling2d_4 (MaxPooling2	(None, 512, 14, 14)	0
zero_padding2d_11 (ZeroPaddi	(None, 512, 16, 16)	0
convolution2d_11 (Convolution2D)	(None, 512, 14, 14)	2359808
zero_padding2d_12 (ZeroPaddi	(None, 512, 16, 16)	0
convolution2d_12 (Convolution2D)	(None, 512, 14, 14)	2359808
zero_padding2d_13 (ZeroPaddi	(None, 512, 16, 16)	0
convolution2d_13 (Convolution2D)	(None, 512, 14, 14)	2359808
max_pooling2d_5 (MaxPooling2	(None, 512, 7, 7)	0
flatten_1 (Flatten)	(None, 25088)	0
dense_1 (Dense)	(None, 4096)	102764544
dropout_1 (Dropout)	(None, 4096)	0
dense_2 (Dense)	(None, 4096)	16781312
dropout_2 (Dropout)	(None, 4096)	0
dense_4 (Dense)	(None, 2)	8194

Total params: 134,268,738

20.2 Vgg19

Layer (type)	Output Shape	Param #
lambda_2 (Lambda)	(None, 3, 224, 224)	0
zero_padding2d_17 (ZeroPaddi	(None, 3, 226, 226)	0
conv2d_17 (Conv2D)	(None, 64, 224, 224)	1792
zero_padding2d_18 (ZeroPaddi	(None, 64, 226, 226)	0
conv2d_18 (Conv2D)	(None, 64, 224, 224)	36928
max_pooling2d_6 (MaxPooling2	(None, 64, 112, 112)	0
zero_padding2d_19 (ZeroPaddi	(None, 64, 114, 114)	0
conv2d_19 (Conv2D)	(None, 128, 112, 112)	73856
zero_padding2d_20 (ZeroPaddi	(None, 128, 114, 114)	0
conv2d_20 (Conv2D)	(None, 128, 112, 112)	147584
max_pooling2d_7 (MaxPooling2	(None, 128, 56, 56)	0
zero_padding2d_21 (ZeroPaddi	(None, 128, 58, 58)	0
conv2d_21 (Conv2D)	(None, 256, 56, 56)	295168
zero_padding2d_22 (ZeroPaddi	(None, 256, 58, 58)	0
conv2d_22 (Conv2D)	(None, 256, 56, 56)	590080
zero_padding2d_23 (ZeroPaddi	(None, 256, 58, 58)	0
conv2d_23 (Conv2D)	(None, 256, 56, 56)	590080
zero_padding2d_24 (ZeroPaddi	(None, 256, 58, 58)	0
conv2d_24 (Conv2D)	(None, 256, 56, 56)	590080
max_pooling2d_8 (MaxPooling2	(None, 256, 28, 28)	0
zero_padding2d_25 (ZeroPaddi	(None, 256, 30, 30)	0
conv2d_25 (Conv2D)	(None, 512, 28, 28)	1180160
zero_padding2d_26 (ZeroPaddi	(None, 512, 30, 30)	0
conv2d_26 (Conv2D)	(None, 512, 28, 28)	2359808

zero_padding2d_27 (ZeroPaddi (None, 512, 30, 30) 0

_____

conv2d_27 (Conv2D) (None, 512, 28, 28) 2359808

_____

zero_padding2d_28 (ZeroPaddi (None, 512, 30, 30) 0

_____

conv2d_28 (Conv2D) (None, 512, 28, 28) 2359808

_____

max_pooling2d_9 (MaxPooling2 (None, 512, 14, 14) 0

_____

zero_padding2d_29 (ZeroPaddi (None, 512, 16, 16) 0

_____

conv2d_29 (Conv2D) (None, 512, 14, 14) 2359808

_____

zero_padding2d_30 (ZeroPaddi (None, 512, 16, 16) 0

_____

conv2d_30 (Conv2D) (None, 512, 14, 14) 2359808

_____

zero_padding2d_31 (ZeroPaddi (None, 512, 16, 16) 0

_____

conv2d_31 (Conv2D) (None, 512, 14, 14) 2359808

_____

zero_padding2d_32 (ZeroPaddi (None, 512, 16, 16) 0

_____

conv2d_32 (Conv2D) (None, 512, 14, 14) 2359808

_____

max_pooling2d_10 (MaxPooling (None, 512, 7, 7) 0

_____

flatten_2 (Flatten) (None, 25088) 0

_____

dense_5 (Dense) (None, 4096) 102764544

_____

dropout_3 (Dropout) (None, 4096) 0

_____

dense_6 (Dense) (None, 4096) 16781312

_____

dropout_4 (Dropout) (None, 4096) 0

_____

dense_8 (Dense) (None, 2) 8194

==========

Total params: 139,578,434

_____

20.3 Resnet50

```
____________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
====================================================================================================
input_1 (InputLayer)         (None, 3, 224, 224)       0
____________________________________________________________________________________________________
zero_padding2d_1 (ZeroPadding2D) (None, 3, 230, 230)   0           input_1[0][0]
____________________________________________________________________________________________________
conv1 (Conv2D)               (None, 64, 112, 112)      9472        zero_padding2d_1[0][0]
____________________________________________________________________________________________________
bn_conv1 (BatchNormalization)  (None, 64, 112, 112)    256         conv1[0][0]
____________________________________________________________________________________________________
activation_1 (Activation)    (None, 64, 112, 112)      0           bn_conv1[0][0]
____________________________________________________________________________________________________
max_pooling2d_1 (MaxPooling2D)  (None, 64, 55, 55)     0           activation_1[0][0]
____________________________________________________________________________________________________
res2a_branch2a (Conv2D)      (None, 64, 55, 55)        4160        max_pooling2d_1[0][0]
____________________________________________________________________________________________________
bn2a_branch2a (BatchNormalizatio (None, 64, 55, 55)    256         res2a_branch2a[0][0]
____________________________________________________________________________________________________
activation_2 (Activation)    (None, 64, 55, 55)        0           bn2a_branch2a[0][0]
____________________________________________________________________________________________________
res2a_branch2b (Conv2D)      (None, 64, 55, 55)        36928       activation_2[0][0]
____________________________________________________________________________________________________
bn2a_branch2b (BatchNormalizatio (None, 64, 55, 55)    256         res2a_branch2b[0][0]
____________________________________________________________________________________________________
activation_3 (Activation)    (None, 64, 55, 55)        0           bn2a_branch2b[0][0]
____________________________________________________________________________________________________
res2a_branch2c (Conv2D)      (None, 256, 55, 55)       16640       activation_3[0][0]
____________________________________________________________________________________________________
res2a_branch1 (Conv2D)       (None, 256, 55, 55)       16640       max_pooling2d_1[0][0]
____________________________________________________________________________________________________
bn2a_branch2c (BatchNormalizatio (None, 256, 55, 55)   1024        res2a_branch2c[0][0]
____________________________________________________________________________________________________
bn2a_branch1 (BatchNormalization (None, 256, 55, 55)   1024        res2a_branch1[0][0]
```

```
____________________________________________________________________________________________________
____________________
add_1 (Add)                 (None, 256, 55, 55)   0         bn2a_branch2c[0][0]
                                                            bn2a_branch1[0][0]
____________________________________________________________________________________________________
____________________
activation_4 (Activation)   (None, 256, 55, 55)   0         add_1[0][0]
____________________________________________________________________________________________________
____________________
res2b_branch2a (Conv2D)     (None, 64, 55, 55)    16448     activation_4[0][0]
____________________________________________________________________________________________________
____________________
bn2b_branch2a (BatchNormalizatio (None, 64, 55, 55)    256       res2b_branch2a[0][0]
____________________________________________________________________________________________________
____________________
activation_5 (Activation)   (None, 64, 55, 55)    0         bn2b_branch2a[0][0]
____________________________________________________________________________________________________
____________________
res2b_branch2b (Conv2D)     (None, 64, 55, 55)    36928     activation_5[0][0]
____________________________________________________________________________________________________
____________________
bn2b_branch2b (BatchNormalizatio (None, 64, 55, 55)    256       res2b_branch2b[0][0]
____________________________________________________________________________________________________
____________________
activation_6 (Activation)   (None, 64, 55, 55)    0         bn2b_branch2b[0][0]
____________________________________________________________________________________________________
____________________
res2b_branch2c (Conv2D)     (None, 256, 55, 55)   16640     activation_6[0][0]
____________________________________________________________________________________________________
____________________
bn2b_branch2c (BatchNormalizatio (None, 256, 55, 55)   1024      res2b_branch2c[0][0]
____________________________________________________________________________________________________
____________________
add_2 (Add)                 (None, 256, 55, 55)   0         bn2b_branch2c[0][0]
                                                            activation_4[0][0]
____________________________________________________________________________________________________
____________________
activation_7 (Activation)   (None, 256, 55, 55)   0         add_2[0][0]
____________________________________________________________________________________________________
____________________
res2c_branch2a (Conv2D)     (None, 64, 55, 55)    16448     activation_7[0][0]
____________________________________________________________________________________________________
____________________
bn2c_branch2a (BatchNormalizatio (None, 64, 55, 55)    256       res2c_branch2a[0][0]
____________________________________________________________________________________________________
____________________
activation_8 (Activation)   (None, 64, 55, 55)    0         bn2c_branch2a[0][0]
____________________________________________________________________________________________________
____________________
res2c_branch2b (Conv2D)     (None, 64, 55, 55)    36928     activation_8[0][0]
```

bn2c_branch2b (BatchNormalizatio (None, 64, 55, 55) 256 res2c_branch2b[0][0]

activation_9 (Activation) (None, 64, 55, 55) 0 bn2c_branch2b[0][0]

res2c_branch2c (Conv2D) (None, 256, 55, 55) 16640 activation_9[0][0]

bn2c_branch2c (BatchNormalizatio (None, 256, 55, 55) 1024 res2c_branch2c[0][0]

add_3 (Add) (None, 256, 55, 55) 0 bn2c_branch2c[0][0]
activation_7[0][0]

activation_10 (Activation) (None, 256, 55, 55) 0 add_3[0][0]

res3a_branch2a (Conv2D) (None, 128, 28, 28) 32896 activation_10[0][0]

bn3a_branch2a (BatchNormalizatio (None, 128, 28, 28) 512 res3a_branch2a[0][0]

activation_11 (Activation) (None, 128, 28, 28) 0 bn3a_branch2a[0][0]

res3a_branch2b (Conv2D) (None, 128, 28, 28) 147584 activation_11[0][0]

bn3a_branch2b (BatchNormalizatio (None, 128, 28, 28) 512 res3a_branch2b[0][0]

activation_12 (Activation) (None, 128, 28, 28) 0 bn3a_branch2b[0][0]

res3a_branch2c (Conv2D) (None, 512, 28, 28) 66048 activation_12[0][0]

res3a_branch1 (Conv2D) (None, 512, 28, 28) 131584 activation_10[0][0]

bn3a_branch2c (BatchNormalizatio (None, 512, 28, 28) 2048 res3a_branch2c[0][0]

bn3a_branch1 (BatchNormalization (None, 512, 28, 28) 2048 res3a_branch1[0][0]

add_4 (Add)	(None, 512, 28, 28)	0	bn3a_branch2c[0][0] bn3a_branch1[0][0]
activation_13 (Activation)	(None, 512, 28, 28)	0	add_4[0][0]
res3b_branch2a (Conv2D)	(None, 128, 28, 28)	65664	activation_13[0][0]
bn3b_branch2a (BatchNormalizatio	(None, 128, 28, 28)	512	res3b_branch2a[0][0]
activation_14 (Activation)	(None, 128, 28, 28)	0	bn3b_branch2a[0][0]
res3b_branch2b (Conv2D)	(None, 128, 28, 28)	147584	activation_14[0][0]
bn3b_branch2b (BatchNormalizatio	(None, 128, 28, 28)	512	res3b_branch2b[0][0]
activation_15 (Activation)	(None, 128, 28, 28)	0	bn3b_branch2b[0][0]
res3b_branch2c (Conv2D)	(None, 512, 28, 28)	66048	activation_15[0][0]
bn3b_branch2c (BatchNormalizatio	(None, 512, 28, 28)	2048	res3b_branch2c[0][0]
add_5 (Add)	(None, 512, 28, 28)	0	bn3b_branch2c[0][0] activation_13[0][0]
activation_16 (Activation)	(None, 512, 28, 28)	0	add_5[0][0]
res3c_branch2a (Conv2D)	(None, 128, 28, 28)	65664	activation_16[0][0]
bn3c_branch2a (BatchNormalizatio	(None, 128, 28, 28)	512	res3c_branch2a[0][0]
activation_17 (Activation)	(None, 128, 28, 28)	0	bn3c_branch2a[0][0]
res3c_branch2b (Conv2D)	(None, 128, 28, 28)	147584	activation_17[0][0]

```
____________________________________________________________________________________________
____________________
bn3c_branch2b (BatchNormalizatio (None, 128, 28, 28)   512        res3c_branch2b[0][0]
____________________________________________________________________________________________
____________________
activation_18 (Activation)      (None, 128, 28, 28)   0          bn3c_branch2b[0][0]
____________________________________________________________________________________________
____________________
res3c_branch2c (Conv2D)         (None, 512, 28, 28)   66048      activation_18[0][0]
____________________________________________________________________________________________
____________________
bn3c_branch2c (BatchNormalizatio (None, 512, 28, 28)   2048       res3c_branch2c[0][0]
____________________________________________________________________________________________
____________________
add_6 (Add)                     (None, 512, 28, 28)   0          bn3c_branch2c[0][0]
                                                                   activation_16[0][0]
____________________________________________________________________________________________
____________________
activation_19 (Activation)      (None, 512, 28, 28)   0          add_6[0][0]
____________________________________________________________________________________________
____________________
res3d_branch2a (Conv2D)         (None, 128, 28, 28)   65664      activation_19[0][0]
____________________________________________________________________________________________
____________________
bn3d_branch2a (BatchNormalizatio (None, 128, 28, 28)   512        res3d_branch2a[0][0]
____________________________________________________________________________________________
____________________
activation_20 (Activation)      (None, 128, 28, 28)   0          bn3d_branch2a[0][0]
____________________________________________________________________________________________
____________________
res3d_branch2b (Conv2D)         (None, 128, 28, 28)   147584     activation_20[0][0]
____________________________________________________________________________________________
____________________
bn3d_branch2b (BatchNormalizatio (None, 128, 28, 28)   512        res3d_branch2b[0][0]
____________________________________________________________________________________________
____________________
activation_21 (Activation)      (None, 128, 28, 28)   0          bn3d_branch2b[0][0]
____________________________________________________________________________________________
____________________
res3d_branch2c (Conv2D)         (None, 512, 28, 28)   66048      activation_21[0][0]
____________________________________________________________________________________________
____________________
bn3d_branch2c (BatchNormalizatio (None, 512, 28, 28)   2048       res3d_branch2c[0][0]
____________________________________________________________________________________________
____________________
add_7 (Add)                     (None, 512, 28, 28)   0          bn3d_branch2c[0][0]
                                                                   activation_19[0][0]
____________________________________________________________________________________________
____________________
activation_22 (Activation)      (None, 512, 28, 28)   0          add_7[0][0]
```

```
____________________________________________________________________________________________________
____________________
res4a_branch2a (Conv2D)         (None, 256, 14, 14)   131328      activation_22[0][0]
____________________________________________________________________________________________________
____________________
bn4a_branch2a (BatchNormalizatio (None, 256, 14, 14)   1024        res4a_branch2a[0][0]
____________________________________________________________________________________________________
____________________
activation_23 (Activation)      (None, 256, 14, 14)   0           bn4a_branch2a[0][0]
____________________________________________________________________________________________________
____________________
res4a_branch2b (Conv2D)         (None, 256, 14, 14)   590080      activation_23[0][0]
____________________________________________________________________________________________________
____________________
bn4a_branch2b (BatchNormalizatio (None, 256, 14, 14)   1024        res4a_branch2b[0][0]
____________________________________________________________________________________________________
____________________
activation_24 (Activation)      (None, 256, 14, 14)   0           bn4a_branch2b[0][0]
____________________________________________________________________________________________________
____________________
res4a_branch2c (Conv2D)         (None, 1024, 14, 14)  263168      activation_24[0][0]
____________________________________________________________________________________________________
____________________

res4a_branch1 (Conv2D)          (None, 1024, 14, 14)  525312      activation_22[0][0]
____________________________________________________________________________________________________
____________________
bn4a_branch2c (BatchNormalizatio (None, 1024, 14, 14)  4096        res4a_branch2c[0][0]
____________________________________________________________________________________________________
____________________
bn4a_branch1 (BatchNormalization (None, 1024, 14, 14)  4096        res4a_branch1[0][0]
____________________________________________________________________________________________________
____________________
add_8 (Add)                     (None, 1024, 14, 14)  0           bn4a_branch2c[0][0]
                                                                  bn4a_branch1[0][0]
____________________________________________________________________________________________________
____________________
activation_25 (Activation)      (None, 1024, 14, 14)  0           add_8[0][0]
____________________________________________________________________________________________________
____________________
res4b_branch2a (Conv2D)         (None, 256, 14, 14)   262400      activation_25[0][0]
____________________________________________________________________________________________________
____________________
bn4b_branch2a (BatchNormalizatio (None, 256, 14, 14)   1024        res4b_branch2a[0][0]
____________________________________________________________________________________________________
____________________
activation_26 (Activation)      (None, 256, 14, 14)   0           bn4b_branch2a[0][0]
____________________________________________________________________________________________________
____________________
res4b_branch2b (Conv2D)         (None, 256, 14, 14)   590080      activation_26[0][0]
```

```
____________________________________________________________________________________________
____________________
bn4b_branch2b (BatchNormalizatio (None, 256, 14, 14)   1024       res4b_branch2b[0][0]
____________________________________________________________________________________________
____________________
activation_27 (Activation)      (None, 256, 14, 14)   0          bn4b_branch2b[0][0]
____________________________________________________________________________________________
____________________
res4b_branch2c (Conv2D)         (None, 1024, 14, 14)  263168     activation_27[0][0]
____________________________________________________________________________________________
____________________
bn4b_branch2c (BatchNormalizatio (None, 1024, 14, 14)  4096       res4b_branch2c[0][0]
____________________________________________________________________________________________
____________________
add_9 (Add)                     (None, 1024, 14, 14)  0          bn4b_branch2c[0][0]
                                                                 activation_25[0][0]
____________________________________________________________________________________________
____________________
activation_28 (Activation)      (None, 1024, 14, 14)  0          add_9[0][0]
____________________________________________________________________________________________
____________________
res4c_branch2a (Conv2D)         (None, 256, 14, 14)   262400     activation_28[0][0]
____________________________________________________________________________________________
____________________
bn4c_branch2a (BatchNormalizatio (None, 256, 14, 14)   1024       res4c_branch2a[0][0]
____________________________________________________________________________________________
____________________
activation_29 (Activation)      (None, 256, 14, 14)   0          bn4c_branch2a[0][0]
____________________________________________________________________________________________
____________________
res4c_branch2b (Conv2D)         (None, 256, 14, 14)   590080     activation_29[0][0]
____________________________________________________________________________________________
____________________
bn4c_branch2b (BatchNormalizatio (None, 256, 14, 14)   1024       res4c_branch2b[0][0]
____________________________________________________________________________________________
____________________
activation_30 (Activation)      (None, 256, 14, 14)   0          bn4c_branch2b[0][0]
____________________________________________________________________________________________
____________________
res4c_branch2c (Conv2D)         (None, 1024, 14, 14)  263168     activation_30[0][0]
____________________________________________________________________________________________
____________________
bn4c_branch2c (BatchNormalizatio (None, 1024, 14, 14)  4096       res4c_branch2c[0][0]
____________________________________________________________________________________________
____________________
add_10 (Add)                    (None, 1024, 14, 14)  0          bn4c_branch2c[0][0]
                                                                 activation_28[0][0]
____________________________________________________________________________________________
____________________
activation_31 (Activation)      (None, 1024, 14, 14)  0          add_10[0][0]
```

```
____________________________________________________________________________________________________
____________________
res4d_branch2a (Conv2D)         (None, 256, 14, 14)   262400     activation_31[0][0]
____________________________________________________________________________________________________
____________________
bn4d_branch2a (BatchNormalizatio (None, 256, 14, 14)   1024       res4d_branch2a[0][0]
____________________________________________________________________________________________________
____________________
activation_32 (Activation)      (None, 256, 14, 14)   0          bn4d_branch2a[0][0]
____________________________________________________________________________________________________
____________________
res4d_branch2b (Conv2D)         (None, 256, 14, 14)   590080     activation_32[0][0]
____________________________________________________________________________________________________
____________________
bn4d_branch2b (BatchNormalizatio (None, 256, 14, 14)   1024       res4d_branch2b[0][0]
____________________________________________________________________________________________________
____________________
activation_33 (Activation)      (None, 256, 14, 14)   0          bn4d_branch2b[0][0]
____________________________________________________________________________________________________
____________________
res4d_branch2c (Conv2D)         (None, 1024, 14, 14)  263168     activation_33[0][0]
____________________________________________________________________________________________________
____________________
bn4d_branch2c (BatchNormalizatio (None, 1024, 14, 14)  4096       res4d_branch2c[0][0]
____________________________________________________________________________________________________
____________________
add_11 (Add)                    (None, 1024, 14, 14)  0          bn4d_branch2c[0][0]
                                                                 activation_31[0][0]
____________________________________________________________________________________________________
____________________
activation_34 (Activation)      (None, 1024, 14, 14)  0          add_11[0][0]
____________________________________________________________________________________________________
____________________
res4e_branch2a (Conv2D)         (None, 256, 14, 14)   262400     activation_34[0][0]
____________________________________________________________________________________________________
____________________
bn4e_branch2a (BatchNormalizatio (None, 256, 14, 14)   1024       res4e_branch2a[0][0]
____________________________________________________________________________________________________
____________________
activation_35 (Activation)      (None, 256, 14, 14)   0          bn4e_branch2a[0][0]
____________________________________________________________________________________________________
____________________
res4e_branch2b (Conv2D)         (None, 256, 14, 14)   590080     activation_35[0][0]
____________________________________________________________________________________________________
____________________
bn4e_branch2b (BatchNormalizatio (None, 256, 14, 14)   1024       res4e_branch2b[0][0]
____________________________________________________________________________________________________
____________________
activation_36 (Activation)      (None, 256, 14, 14)   0          bn4e_branch2b[0][0]
____________________________________________________________________________________________________
____________________
```

res4e_branch2c (Conv2D) (None, 1024, 14, 14) 263168 activation_36[0][0]

bn4e_branch2c (BatchNormalizatio (None, 1024, 14, 14) 4096 res4e_branch2c[0][0]

add_12 (Add) (None, 1024, 14, 14) 0 bn4e_branch2c[0][0]
activation_34[0][0]

activation_37 (Activation) (None, 1024, 14, 14) 0 add_12[0][0]

res4f_branch2a (Conv2D) (None, 256, 14, 14) 262400 activation_37[0][0]

bn4f_branch2a (BatchNormalizatio (None, 256, 14, 14) 1024 res4f_branch2a[0][0]

activation_38 (Activation) (None, 256, 14, 14) 0 bn4f_branch2a[0][0]

res4f_branch2b (Conv2D) (None, 256, 14, 14) 590080 activation_38[0][0]

bn4f_branch2b (BatchNormalizatio (None, 256, 14, 14) 1024 res4f_branch2b[0][0]

activation_39 (Activation) (None, 256, 14, 14) 0 bn4f_branch2b[0][0]

res4f_branch2c (Conv2D) (None, 1024, 14, 14) 263168 activation_39[0][0]

bn4f_branch2c (BatchNormalizatio (None, 1024, 14, 14) 4096 res4f_branch2c[0][0]

add_13 (Add) (None, 1024, 14, 14) 0 bn4f_branch2c[0][0]
activation_37[0][0]

activation_40 (Activation) (None, 1024, 14, 14) 0 add_13[0][0]

res5a_branch2a (Conv2D) (None, 512, 7, 7) 524800 activation_40[0][0]

bn5a_branch2a (BatchNormalizatio (None, 512, 7, 7) 2048 res5a_branch2a[0][0]

activation_41 (Activation) (None, 512, 7, 7) 0 bn5a_branch2a[0][0]

res5a_branch2b (Conv2D) (None, 512, 7, 7) 2359808 activation_41[0][0]

bn5a_branch2b (BatchNormalizatio (None, 512, 7, 7) 2048 res5a_branch2b[0][0]

activation_42 (Activation) (None, 512, 7, 7) 0 bn5a_branch2b[0][0]

res5a_branch2c (Conv2D) (None, 2048, 7, 7) 1050624 activation_42[0][0]

res5a_branch1 (Conv2D) (None, 2048, 7, 7) 2099200 activation_40[0][0]

bn5a_branch2c (BatchNormalizatio (None, 2048, 7, 7) 8192 res5a_branch2c[0][0]

bn5a_branch1 (BatchNormalization (None, 2048, 7, 7) 8192 res5a_branch1[0][0]

add_14 (Add) (None, 2048, 7, 7) 0 bn5a_branch2c[0][0]
bn5a_branch1[0][0]

activation_43 (Activation) (None, 2048, 7, 7) 0 add_14[0][0]

res5b_branch2a (Conv2D) (None, 512, 7, 7) 1049088 activation_43[0][0]

bn5b_branch2a (BatchNormalizatio (None, 512, 7, 7) 2048 res5b_branch2a[0][0]

activation_44 (Activation) (None, 512, 7, 7) 0 bn5b_branch2a[0][0]

res5b_branch2b (Conv2D) (None, 512, 7, 7) 2359808 activation_44[0][0]

bn5b_branch2b (BatchNormalizatio (None, 512, 7, 7) 2048 res5b_branch2b[0][0]

activation_45 (Activation) (None, 512, 7, 7) 0 bn5b_branch2b[0][0]

res5b_branch2c (Conv2D) (None, 2048, 7, 7) 1050624 activation_45[0][0]

```
____________________________________________________________________________________________________
bn5b_branch2c (BatchNormalizatio (None, 2048, 7, 7)    8192       res5b_branch2c[0][0]
____________________________________________________________________________________________________
add_15 (Add)                    (None, 2048, 7, 7)    0          bn5b_branch2c[0][0]
                                                                  activation_43[0][0]
____________________________________________________________________________________________________
activation_46 (Activation)      (None, 2048, 7, 7)    0          add_15[0][0]
____________________________________________________________________________________________________
res5c_branch2a (Conv2D)         (None, 512, 7, 7)     1049088    activation_46[0][0]
____________________________________________________________________________________________________
bn5c_branch2a (BatchNormalizatio (None, 512, 7, 7)     2048       res5c_branch2a[0][0]
____________________________________________________________________________________________________
activation_47 (Activation)      (None, 512, 7, 7)     0          bn5c_branch2a[0][0]
____________________________________________________________________________________________________
res5c_branch2b (Conv2D)         (None, 512, 7, 7)     2359808    activation_47[0][0]
____________________________________________________________________________________________________
bn5c_branch2b (BatchNormalizatio (None, 512, 7, 7)     2048       res5c_branch2b[0][0]
____________________________________________________________________________________________________
activation_48 (Activation)      (None, 512, 7, 7)     0          bn5c_branch2b[0][0]
____________________________________________________________________________________________________
res5c_branch2c (Conv2D)         (None, 2048, 7, 7)    1050624    activation_48[0][0]
____________________________________________________________________________________________________
bn5c_branch2c (BatchNormalizatio (None, 2048, 7, 7)    8192       res5c_branch2c[0][0]
____________________________________________________________________________________________________
add_16 (Add)                    (None, 2048, 7, 7)    0          bn5c_branch2c[0][0]
                                                                  activation_46[0][0]
____________________________________________________________________________________________________
activation_49 (Activation)      (None, 2048, 7, 7)    0          add_16[0][0]
____________________________________________________________________________________________________
avg_pool (AveragePooling2D)     (None, 2048, 1, 1)    0          activation_49[0][0]
____________________________________________________________________________________________________
flatten_2 (Flatten)             (None, 2048)          0          avg_pool[0][0]
____________________________________________________________________________________________________
dense_1 (Dense)                 (None, 2)             4098       flatten_2[0][0]
```

```
================================================================================
==================
```

Total params: 23,591,810

20.4 Inception-v3

```
____________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
====================================================================================================
input_2 (InputLayer)         (None, 3, 299, 299)       0
____________________________________________________________________________________________________
conv2d_95 (Conv2D)           (None, 32, 149, 149)      864         input_2[0][0]
____________________________________________________________________________________________________
batch_normalization_95 (BatchNor (None, 32, 149, 149)  96          conv2d_95[0][0]
____________________________________________________________________________________________________
activation_95 (Activation)   (None, 32, 149, 149)      0           batch_normalization_95[0][0]
____________________________________________________________________________________________________
conv2d_96 (Conv2D)           (None, 32, 147, 147)      9216        activation_95[0][0]
____________________________________________________________________________________________________
batch_normalization_96 (BatchNor (None, 32, 147, 147)  96          conv2d_96[0][0]
____________________________________________________________________________________________________
activation_96 (Activation)   (None, 32, 147, 147)      0           batch_normalization_96[0][0]
____________________________________________________________________________________________________
conv2d_97 (Conv2D)           (None, 64, 147, 147)      18432       activation_96[0][0]
____________________________________________________________________________________________________
batch_normalization_97 (BatchNor (None, 64, 147, 147)  192         conv2d_97[0][0]
____________________________________________________________________________________________________
activation_97 (Activation)   (None, 64, 147, 147)      0           batch_normalization_97[0][0]
____________________________________________________________________________________________________
max_pooling2d_5 (MaxPooling2D)   (None, 64, 73, 73)    0           activation_97[0][0]
____________________________________________________________________________________________________
conv2d_98 (Conv2D)           (None, 80, 73, 73)        5120        max_pooling2d_5[0][0]
____________________________________________________________________________________________________
batch_normalization_98 (BatchNor (None, 80, 73, 73)    240         conv2d_98[0][0]
____________________________________________________________________________________________________
activation_98 (Activation)   (None, 80, 73, 73)        0           batch_normalization_98[0][0]
____________________________________________________________________________________________________
conv2d_99 (Conv2D)           (None, 192, 71, 71)       138240      activation_98[0][0]
____________________________________________________________________________________________________
batch_normalization_99 (BatchNor (None, 192, 71, 71)   576         conv2d_99[0][0]
```

```
____________________________________________________________________________________________________
activation_99 (Activation)      (None, 192, 71, 71)   0           batch_normalization_99[0][0]
____________________________________________________________________________________________________
max_pooling2d_6 (MaxPooling2D)   (None, 192, 35, 35)   0           activation_99[0][0]
____________________________________________________________________________________________________
conv2d_103 (Conv2D)             (None, 64, 35, 35)    12288       max_pooling2d_6[0][0]
____________________________________________________________________________________________________
batch_normalization_103 (BatchNo (None, 64, 35, 35)    192         conv2d_103[0][0]
____________________________________________________________________________________________________
activation_103 (Activation)     (None, 64, 35, 35)    0           batch_normalization_103[0][0]
____________________________________________________________________________________________________
conv2d_101 (Conv2D)             (None, 48, 35, 35)    9216        max_pooling2d_6[0][0]
____________________________________________________________________________________________________
conv2d_104 (Conv2D)             (None, 96, 35, 35)    55296       activation_103[0][0]
____________________________________________________________________________________________________
batch_normalization_101 (BatchNo (None, 48, 35, 35)    144         conv2d_101[0][0]
____________________________________________________________________________________________________
batch_normalization_104 (BatchNo (None, 96, 35, 35)    288         conv2d_104[0][0]
____________________________________________________________________________________________________
activation_101 (Activation)     (None, 48, 35, 35)    0           batch_normalization_101[0][0]
____________________________________________________________________________________________________
activation_104 (Activation)     (None, 96, 35, 35)    0           batch_normalization_104[0][0]
____________________________________________________________________________________________________
average_pooling2d_10 (AveragePoo (None, 192, 35, 35)   0           max_pooling2d_6[0][0]
____________________________________________________________________________________________________
conv2d_100 (Conv2D)             (None, 64, 35, 35)    12288       max_pooling2d_6[0][0]
____________________________________________________________________________________________________
conv2d_102 (Conv2D)             (None, 64, 35, 35)    76800       activation_101[0][0]
____________________________________________________________________________________________________
conv2d_105 (Conv2D)             (None, 96, 35, 35)    82944       activation_104[0][0]
____________________________________________________________________________________________________
conv2d_106 (Conv2D)             (None, 32, 35, 35)    6144        average_pooling2d_10[0][0]
____________________________________________________________________________________________________
batch_normalization_100 (BatchNo (None, 64, 35, 35)    192         conv2d_100[0][0]
```

```
__________________________________________________________________________________________________
batch_normalization_102 (BatchNo (None, 64, 35, 35)    192       conv2d_102[0][0]
__________________________________________________________________________________________________
batch_normalization_105 (BatchNo (None, 96, 35, 35)    288       conv2d_105[0][0]
__________________________________________________________________________________________________
batch_normalization_106 (BatchNo (None, 32, 35, 35)    96        conv2d_106[0][0]
__________________________________________________________________________________________________
activation_100 (Activation)     (None, 64, 35, 35)    0         batch_normalization_100[0][0]
__________________________________________________________________________________________________
activation_102 (Activation)     (None, 64, 35, 35)    0         batch_normalization_102[0][0]
__________________________________________________________________________________________________
activation_105 (Activation)     (None, 96, 35, 35)    0         batch_normalization_105[0][0]
__________________________________________________________________________________________________
activation_106 (Activation)     (None, 32, 35, 35)    0         batch_normalization_106[0][0]
__________________________________________________________________________________________________
mixed0 (Concatenate)            (None, 256, 35, 35)   0         activation_100[0][0]
                                                                  activation_102[0][0]
                                                                  activation_105[0][0]
                                                                  activation_106[0][0]
__________________________________________________________________________________________________
conv2d_110 (Conv2D)             (None, 64, 35, 35)    16384     mixed0[0][0]
__________________________________________________________________________________________________
batch_normalization_110 (BatchNo (None, 64, 35, 35)    192       conv2d_110[0][0]
__________________________________________________________________________________________________
activation_110 (Activation)     (None, 64, 35, 35)    0         batch_normalization_110[0][0]
__________________________________________________________________________________________________
conv2d_108 (Conv2D)             (None, 48, 35, 35)    12288     mixed0[0][0]
__________________________________________________________________________________________________
conv2d_111 (Conv2D)             (None, 96, 35, 35)    55296     activation_110[0][0]
__________________________________________________________________________________________________
batch_normalization_108 (BatchNo (None, 48, 35, 35)    144       conv2d_108[0][0]
__________________________________________________________________________________________________
batch_normalization_111 (BatchNo (None, 96, 35, 35)    288       conv2d_111[0][0]
__________________________________________________________________________________________________
activation_108 (Activation)     (None, 48, 35, 35)    0         batch_normalization_108[0][0]
```

activation_111 (Activation)	(None, 96, 35, 35)	0	batch_normalization_111[0][0]
average_pooling2d_11 (AveragePoo	(None, 256, 35, 35)	0	mixed0[0][0]
conv2d_107 (Conv2D)	(None, 64, 35, 35)	16384	mixed0[0][0]
conv2d_109 (Conv2D)	(None, 64, 35, 35)	76800	activation_108[0][0]
conv2d_112 (Conv2D)	(None, 96, 35, 35)	82944	activation_111[0][0]
conv2d_113 (Conv2D)	(None, 64, 35, 35)	16384	average_pooling2d_11[0][0]
batch_normalization_107 (BatchNo	(None, 64, 35, 35)	192	conv2d_107[0][0]
batch_normalization_109 (BatchNo	(None, 64, 35, 35)	192	conv2d_109[0][0]
batch_normalization_112 (BatchNo	(None, 96, 35, 35)	288	conv2d_112[0][0]
batch_normalization_113 (BatchNo	(None, 64, 35, 35)	192	conv2d_113[0][0]
activation_107 (Activation)	(None, 64, 35, 35)	0	batch_normalization_107[0][0]
activation_109 (Activation)	(None, 64, 35, 35)	0	batch_normalization_109[0][0]
activation_112 (Activation)	(None, 96, 35, 35)	0	batch_normalization_112[0][0]
activation_113 (Activation)	(None, 64, 35, 35)	0	batch_normalization_113[0][0]
mixed1 (Concatenate)	(None, 288, 35, 35)	0	activation_107[0][0] activation_109[0][0] activation_112[0][0] activation_113[0][0]

conv2d_117 (Conv2D) (None, 64, 35, 35) 18432 mixed1[0][0]

batch_normalization_117 (BatchNo (None, 64, 35, 35) 192 conv2d_117[0][0]

activation_117 (Activation) (None, 64, 35, 35) 0 batch_normalization_117[0][0]

conv2d_115 (Conv2D) (None, 48, 35, 35) 13824 mixed1[0][0]

conv2d_118 (Conv2D) (None, 96, 35, 35) 55296 activation_117[0][0]

batch_normalization_115 (BatchNo (None, 48, 35, 35) 144 conv2d_115[0][0]

batch_normalization_118 (BatchNo (None, 96, 35, 35) 288 conv2d_118[0][0]

activation_115 (Activation) (None, 48, 35, 35) 0 batch_normalization_115[0][0]

activation_118 (Activation) (None, 96, 35, 35) 0 batch_normalization_118[0][0]

average_pooling2d_12 (AveragePoo (None, 288, 35, 35) 0 mixed1[0][0]

conv2d_114 (Conv2D) (None, 64, 35, 35) 18432 mixed1[0][0]

conv2d_116 (Conv2D) (None, 64, 35, 35) 76800 activation_115[0][0]

conv2d_119 (Conv2D) (None, 96, 35, 35) 82944 activation_118[0][0]

conv2d_120 (Conv2D) (None, 64, 35, 35) 18432 average_pooling2d_12[0][0]

batch_normalization_114 (BatchNo (None, 64, 35, 35) 192 conv2d_114[0][0]

batch_normalization_116 (BatchNo (None, 64, 35, 35) 192 conv2d_116[0][0]

batch_normalization_119 (BatchNo (None, 96, 35, 35) 288 conv2d_119[0][0]

batch_normalization_120 (BatchNo (None, 64, 35, 35) 192 conv2d_120[0][0]

activation_114 (Activation) (None, 64, 35, 35) 0 batch_normalization_114[0][0]

activation_116 (Activation) (None, 64, 35, 35) 0 batch_normalization_116[0][0]

activation_119 (Activation) (None, 96, 35, 35) 0 batch_normalization_119[0][0]

activation_120 (Activation) (None, 64, 35, 35) 0 batch_normalization_120[0][0]

mixed2 (Concatenate) (None, 288, 35, 35) 0 activation_114[0][0]
activation_116[0][0]
activation_119[0][0]
activation_120[0][0]

conv2d_122 (Conv2D) (None, 64, 35, 35) 18432 mixed2[0][0]

batch_normalization_122 (BatchNo (None, 64, 35, 35) 192 conv2d_122[0][0]

activation_122 (Activation) (None, 64, 35, 35) 0 batch_normalization_122[0][0]

conv2d_123 (Conv2D) (None, 96, 35, 35) 55296 activation_122[0][0]

batch_normalization_123 (BatchNo (None, 96, 35, 35) 288 conv2d_123[0][0]

activation_123 (Activation) (None, 96, 35, 35) 0 batch_normalization_123[0][0]

conv2d_121 (Conv2D) (None, 384, 17, 17) 995328 mixed2[0][0]

conv2d_124 (Conv2D) (None, 96, 17, 17) 82944 activation_123[0][0]

batch_normalization_121 (BatchNo (None, 384, 17, 17) 1152 conv2d_121[0][0]

batch_normalization_124 (BatchNo (None, 96, 17, 17) 288 conv2d_124[0][0]

```
__________________________________________________________________________________________________
activation_121 (Activation)     (None, 384, 17, 17)   0           batch_normalization_121[0][0]
__________________________________________________________________________________________________
activation_124 (Activation)     (None, 96, 17, 17)    0           batch_normalization_124[0][0]
__________________________________________________________________________________________________
max_pooling2d_7 (MaxPooling2D)  (None, 288, 17, 17)   0           mixed2[0][0]
__________________________________________________________________________________________________
mixed3 (Concatenate)            (None, 768, 17, 17)   0           activation_121[0][0]
                                                                    activation_124[0][0]
                                                                    max_pooling2d_7[0][0]
__________________________________________________________________________________________________
conv2d_129 (Conv2D)             (None, 128, 17, 17)   98304       mixed3[0][0]
__________________________________________________________________________________________________
batch_normalization_129 (BatchNo (None, 128, 17, 17)  384         conv2d_129[0][0]
__________________________________________________________________________________________________
activation_129 (Activation)     (None, 128, 17, 17)   0           batch_normalization_129[0][0]
__________________________________________________________________________________________________
conv2d_130 (Conv2D)             (None, 128, 17, 17)   114688      activation_129[0][0]
__________________________________________________________________________________________________
batch_normalization_130 (BatchNo (None, 128, 17, 17)  384         conv2d_130[0][0]
__________________________________________________________________________________________________
activation_130 (Activation)     (None, 128, 17, 17)   0           batch_normalization_130[0][0]
__________________________________________________________________________________________________
conv2d_126 (Conv2D)             (None, 128, 17, 17)   98304       mixed3[0][0]
__________________________________________________________________________________________________
conv2d_131 (Conv2D)             (None, 128, 17, 17)   114688      activation_130[0][0]
__________________________________________________________________________________________________
batch_normalization_126 (BatchNo (None, 128, 17, 17)  384         conv2d_126[0][0]
__________________________________________________________________________________________________
batch_normalization_131 (BatchNo (None, 128, 17, 17)  384         conv2d_131[0][0]
__________________________________________________________________________________________________
activation_126 (Activation)     (None, 128, 17, 17)   0           batch_normalization_126[0][0]
__________________________________________________________________________________________________
activation_131 (Activation)     (None, 128, 17, 17)   0           batch_normalization_131[0][0]
```

```
____________________________________________________________________________________________________
____________________
conv2d_127 (Conv2D)             (None, 128, 17, 17)   114688     activation_126[0][0]
____________________________________________________________________________________________________
____________________
conv2d_132 (Conv2D)             (None, 128, 17, 17)   114688     activation_131[0][0]
____________________________________________________________________________________________________
____________________
batch_normalization_127 (BatchNo (None, 128, 17, 17)   384        conv2d_127[0][0]
____________________________________________________________________________________________________
____________________
batch_normalization_132 (BatchNo (None, 128, 17, 17)   384        conv2d_132[0][0]
____________________________________________________________________________________________________
____________________
activation_127 (Activation)     (None, 128, 17, 17)   0          batch_normalization_127[0][0]
____________________________________________________________________________________________________
____________________
activation_132 (Activation)     (None, 128, 17, 17)   0          batch_normalization_132[0][0]
____________________________________________________________________________________________________
____________________
average_pooling2d_13 (AveragePoo (None, 768, 17, 17)   0          mixed3[0][0]
____________________________________________________________________________________________________
____________________
conv2d_125 (Conv2D)             (None, 192, 17, 17)   147456     mixed3[0][0]
____________________________________________________________________________________________________
____________________
conv2d_128 (Conv2D)             (None, 192, 17, 17)   172032     activation_127[0][0]
____________________________________________________________________________________________________
____________________
conv2d_133 (Conv2D)             (None, 192, 17, 17)   172032     activation_132[0][0]
____________________________________________________________________________________________________
____________________
conv2d_134 (Conv2D)             (None, 192, 17, 17)   147456     average_pooling2d_13[0][0]
____________________________________________________________________________________________________
____________________
batch_normalization_125 (BatchNo (None, 192, 17, 17)   576        conv2d_125[0][0]
____________________________________________________________________________________________________
____________________
batch_normalization_128 (BatchNo (None, 192, 17, 17)   576        conv2d_128[0][0]
____________________________________________________________________________________________________
____________________
batch_normalization_133 (BatchNo (None, 192, 17, 17)   576        conv2d_133[0][0]
____________________________________________________________________________________________________
____________________
batch_normalization_134 (BatchNo (None, 192, 17, 17)   576        conv2d_134[0][0]
____________________________________________________________________________________________________
____________________
activation_125 (Activation)     (None, 192, 17, 17)   0          batch_normalization_125[0][0]
____________________________________________________________________________________________________
____________________
activation_128 (Activation)     (None, 192, 17, 17)   0          batch_normalization_128[0][0]
```

```
__________________________________________________________________________________________________
activation_133 (Activation)     (None, 192, 17, 17)  0           batch_normalization_133[0][0]
__________________________________________________________________________________________________
activation_134 (Activation)     (None, 192, 17, 17)  0           batch_normalization_134[0][0]
__________________________________________________________________________________________________
mixed4 (Concatenate)            (None, 768, 17, 17)  0           activation_125[0][0]
                                                                 activation_128[0][0]
                                                                 activation_133[0][0]
                                                                 activation_134[0][0]
__________________________________________________________________________________________________
conv2d_139 (Conv2D)             (None, 160, 17, 17)  122880      mixed4[0][0]
__________________________________________________________________________________________________
batch_normalization_139 (BatchNo (None, 160, 17, 17)  480         conv2d_139[0][0]
__________________________________________________________________________________________________
activation_139 (Activation)     (None, 160, 17, 17)  0           batch_normalization_139[0][0]
__________________________________________________________________________________________________
conv2d_140 (Conv2D)             (None, 160, 17, 17)  179200      activation_139[0][0]
__________________________________________________________________________________________________
batch_normalization_140 (BatchNo (None, 160, 17, 17)  480         conv2d_140[0][0]
__________________________________________________________________________________________________
activation_140 (Activation)     (None, 160, 17, 17)  0           batch_normalization_140[0][0]
__________________________________________________________________________________________________
conv2d_136 (Conv2D)             (None, 160, 17, 17)  122880      mixed4[0][0]
__________________________________________________________________________________________________
conv2d_141 (Conv2D)             (None, 160, 17, 17)  179200      activation_140[0][0]
__________________________________________________________________________________________________
batch_normalization_136 (BatchNo (None, 160, 17, 17)  480         conv2d_136[0][0]
__________________________________________________________________________________________________
batch_normalization_141 (BatchNo (None, 160, 17, 17)  480         conv2d_141[0][0]
__________________________________________________________________________________________________
activation_136 (Activation)     (None, 160, 17, 17)  0           batch_normalization_136[0][0]
__________________________________________________________________________________________________
activation_141 (Activation)     (None, 160, 17, 17)  0           batch_normalization_141[0][0]
__________________________________________________________________________________________________
```

conv2d_137 (Conv2D) (None, 160, 17, 17) 179200 activation_136[0][0]

conv2d_142 (Conv2D) (None, 160, 17, 17) 179200 activation_141[0][0]

batch_normalization_137 (BatchNo (None, 160, 17, 17) 480 conv2d_137[0][0]

batch_normalization_142 (BatchNo (None, 160, 17, 17) 480 conv2d_142[0][0]

activation_137 (Activation) (None, 160, 17, 17) 0 batch_normalization_137[0][0]

activation_142 (Activation) (None, 160, 17, 17) 0 batch_normalization_142[0][0]

average_pooling2d_14 (AveragePoo (None, 768, 17, 17) 0 mixed4[0][0]

conv2d_135 (Conv2D) (None, 192, 17, 17) 147456 mixed4[0][0]

conv2d_138 (Conv2D) (None, 192, 17, 17) 215040 activation_137[0][0]

conv2d_143 (Conv2D) (None, 192, 17, 17) 215040 activation_142[0][0]

conv2d_144 (Conv2D) (None, 192, 17, 17) 147456 average_pooling2d_14[0][0]

batch_normalization_135 (BatchNo (None, 192, 17, 17) 576 conv2d_135[0][0]

batch_normalization_138 (BatchNo (None, 192, 17, 17) 576 conv2d_138[0][0]

batch_normalization_143 (BatchNo (None, 192, 17, 17) 576 conv2d_143[0][0]

batch_normalization_144 (BatchNo (None, 192, 17, 17) 576 conv2d_144[0][0]

activation_135 (Activation) (None, 192, 17, 17) 0 batch_normalization_135[0][0]

activation_138 (Activation) (None, 192, 17, 17) 0 batch_normalization_138[0][0]

activation_143 (Activation) (None, 192, 17, 17) 0 batch_normalization_143[0][0]

activation_144 (Activation) (None, 192, 17, 17) 0 batch_normalization_144[0][0]

mixed5 (Concatenate) (None, 768, 17, 17) 0 activation_135[0][0]
activation_138[0][0]
activation_143[0][0]
activation_144[0][0]

conv2d_149 (Conv2D) (None, 160, 17, 17) 122880 mixed5[0][0]

batch_normalization_149 (BatchNo (None, 160, 17, 17) 480 conv2d_149[0][0]

activation_149 (Activation) (None, 160, 17, 17) 0 batch_normalization_149[0][0]

conv2d_150 (Conv2D) (None, 160, 17, 17) 179200 activation_149[0][0]

batch_normalization_150 (BatchNo (None, 160, 17, 17) 480 conv2d_150[0][0]

activation_150 (Activation) (None, 160, 17, 17) 0 batch_normalization_150[0][0]

conv2d_146 (Conv2D) (None, 160, 17, 17) 122880 mixed5[0][0]

conv2d_151 (Conv2D) (None, 160, 17, 17) 179200 activation_150[0][0]

batch_normalization_146 (BatchNo (None, 160, 17, 17) 480 conv2d_146[0][0]

batch_normalization_151 (BatchNo (None, 160, 17, 17) 480 conv2d_151[0][0]

activation_146 (Activation) (None, 160, 17, 17) 0 batch_normalization_146[0][0]

activation_151 (Activation) (None, 160, 17, 17) 0 batch_normalization_151[0][0]

conv2d_147 (Conv2D) (None, 160, 17, 17) 179200 activation_146[0][0]

conv2d_152 (Conv2D) (None, 160, 17, 17) 179200 activation_151[0][0]

batch_normalization_147 (BatchNo (None, 160, 17, 17) 480 conv2d_147[0][0]

batch_normalization_152 (BatchNo (None, 160, 17, 17) 480 conv2d_152[0][0]

activation_147 (Activation) (None, 160, 17, 17) 0 batch_normalization_147[0][0]

activation_152 (Activation) (None, 160, 17, 17) 0 batch_normalization_152[0][0]

average_pooling2d_15 (AveragePoo (None, 768, 17, 17) 0 mixed5[0][0]

conv2d_145 (Conv2D) (None, 192, 17, 17) 147456 mixed5[0][0]

conv2d_148 (Conv2D) (None, 192, 17, 17) 215040 activation_147[0][0]

conv2d_153 (Conv2D) (None, 192, 17, 17) 215040 activation_152[0][0]

conv2d_154 (Conv2D) (None, 192, 17, 17) 147456 average_pooling2d_15[0][0]

batch_normalization_145 (BatchNo (None, 192, 17, 17) 576 conv2d_145[0][0]

batch_normalization_148 (BatchNo (None, 192, 17, 17) 576 conv2d_148[0][0]

batch_normalization_153 (BatchNo (None, 192, 17, 17) 576 conv2d_153[0][0]

batch_normalization_154 (BatchNo (None, 192, 17, 17) 576 conv2d_154[0][0]

activation_145 (Activation) (None, 192, 17, 17) 0 batch_normalization_145[0][0]

activation_148 (Activation) (None, 192, 17, 17) 0 batch_normalization_148[0][0]

activation_153 (Activation) (None, 192, 17, 17) 0 batch_normalization_153[0][0]

```
__________________________________________________________________________________________________
activation_154 (Activation)     (None, 192, 17, 17)   0           batch_normalization_154[0][0]
__________________________________________________________________________________________________
mixed6 (Concatenate)            (None, 768, 17, 17)   0           activation_145[0][0]
                                                                  activation_148[0][0]
                                                                  activation_153[0][0]
                                                                  activation_154[0][0]
__________________________________________________________________________________________________
conv2d_159 (Conv2D)             (None, 192, 17, 17)   147456      mixed6[0][0]
__________________________________________________________________________________________________
batch_normalization_159 (BatchNo (None, 192, 17, 17)   576         conv2d_159[0][0]
__________________________________________________________________________________________________
activation_159 (Activation)     (None, 192, 17, 17)   0           batch_normalization_159[0][0]
__________________________________________________________________________________________________
conv2d_160 (Conv2D)             (None, 192, 17, 17)   258048      activation_159[0][0]
__________________________________________________________________________________________________
batch_normalization_160 (BatchNo (None, 192, 17, 17)   576         conv2d_160[0][0]
__________________________________________________________________________________________________
activation_160 (Activation)     (None, 192, 17, 17)   0           batch_normalization_160[0][0]
__________________________________________________________________________________________________
conv2d_156 (Conv2D)             (None, 192, 17, 17)   147456      mixed6[0][0]
__________________________________________________________________________________________________
conv2d_161 (Conv2D)             (None, 192, 17, 17)   258048      activation_160[0][0]
__________________________________________________________________________________________________
batch_normalization_156 (BatchNo (None, 192, 17, 17)   576         conv2d_156[0][0]
__________________________________________________________________________________________________
batch_normalization_161 (BatchNo (None, 192, 17, 17)   576         conv2d_161[0][0]
__________________________________________________________________________________________________
activation_156 (Activation)     (None, 192, 17, 17)   0           batch_normalization_156[0][0]
__________________________________________________________________________________________________
activation_161 (Activation)     (None, 192, 17, 17)   0           batch_normalization_161[0][0]
__________________________________________________________________________________________________
conv2d_157 (Conv2D)             (None, 192, 17, 17)   258048      activation_156[0][0]
__________________________________________________________________________________________________
conv2d_162 (Conv2D)             (None, 192, 17, 17)   258048      activation_161[0][0]
```

batch_normalization_157 (BatchNo	(None, 192, 17, 17)	576	conv2d_157[0][0]
batch_normalization_162 (BatchNo	(None, 192, 17, 17)	576	conv2d_162[0][0]
activation_157 (Activation)	(None, 192, 17, 17)	0	batch_normalization_157[0][0]
activation_162 (Activation)	(None, 192, 17, 17)	0	batch_normalization_162[0][0]
average_pooling2d_16 (AveragePoo	(None, 768, 17, 17)	0	mixed6[0][0]
conv2d_155 (Conv2D)	(None, 192, 17, 17)	147456	mixed6[0][0]
conv2d_158 (Conv2D)	(None, 192, 17, 17)	258048	activation_157[0][0]
conv2d_163 (Conv2D)	(None, 192, 17, 17)	258048	activation_162[0][0]
conv2d_164 (Conv2D)	(None, 192, 17, 17)	147456	average_pooling2d_16[0][0]
batch_normalization_155 (BatchNo	(None, 192, 17, 17)	576	conv2d_155[0][0]
batch_normalization_158 (BatchNo	(None, 192, 17, 17)	576	conv2d_158[0][0]
batch_normalization_163 (BatchNo	(None, 192, 17, 17)	576	conv2d_163[0][0]
batch_normalization_164 (BatchNo	(None, 192, 17, 17)	576	conv2d_164[0][0]
activation_155 (Activation)	(None, 192, 17, 17)	0	batch_normalization_155[0][0]
activation_158 (Activation)	(None, 192, 17, 17)	0	batch_normalization_158[0][0]
activation_163 (Activation)	(None, 192, 17, 17)	0	batch_normalization_163[0][0]
activation_164 (Activation)	(None, 192, 17, 17)	0	batch_normalization_164[0][0]

mixed7 (Concatenate)	(None, 768, 17, 17)	0	activation_155[0][0] activation_158[0][0] activation_163[0][0] activation_164[0][0]	
conv2d_167 (Conv2D)	(None, 192, 17, 17)	147456	mixed7[0][0]	
batch_normalization_167 (BatchNo	(None, 192, 17, 17)	576	conv2d_167[0][0]	
activation_167 (Activation)	(None, 192, 17, 17)	0	batch_normalization_167[0][0]	
conv2d_168 (Conv2D)	(None, 192, 17, 17)	258048	activation_167[0][0]	
batch_normalization_168 (BatchNo	(None, 192, 17, 17)	576	conv2d_168[0][0]	
activation_168 (Activation)	(None, 192, 17, 17)	0	batch_normalization_168[0][0]	
conv2d_165 (Conv2D)	(None, 192, 17, 17)	147456	mixed7[0][0]	
conv2d_169 (Conv2D)	(None, 192, 17, 17)	258048	activation_168[0][0]	
batch_normalization_165 (BatchNo	(None, 192, 17, 17)	576	conv2d_165[0][0]	
batch_normalization_169 (BatchNo	(None, 192, 17, 17)	576	conv2d_169[0][0]	
activation_165 (Activation)	(None, 192, 17, 17)	0	batch_normalization_165[0][0]	
activation_169 (Activation)	(None, 192, 17, 17)	0	batch_normalization_169[0][0]	
conv2d_166 (Conv2D)	(None, 320, 8, 8)	552960	activation_165[0][0]	
conv2d_170 (Conv2D)	(None, 192, 8, 8)	331776	activation_169[0][0]	

batch_normalization_166 (BatchNo (None, 320, 8, 8) 960 conv2d_166[0][0]

batch_normalization_170 (BatchNo (None, 192, 8, 8) 576 conv2d_170[0][0]

activation_166 (Activation) (None, 320, 8, 8) 0 batch_normalization_166[0][0]

activation_170 (Activation) (None, 192, 8, 8) 0 batch_normalization_170[0][0]

max_pooling2d_8 (MaxPooling2D) (None, 768, 8, 8) 0 mixed7[0][0]

mixed8 (Concatenate) (None, 1280, 8, 8) 0 activation_166[0][0]
activation_170[0][0]
max_pooling2d_8[0][0]

conv2d_175 (Conv2D) (None, 448, 8, 8) 573440 mixed8[0][0]

batch_normalization_175 (BatchNo (None, 448, 8, 8) 1344 conv2d_175[0][0]

activation_175 (Activation) (None, 448, 8, 8) 0 batch_normalization_175[0][0]

conv2d_172 (Conv2D) (None, 384, 8, 8) 491520 mixed8[0][0]

conv2d_176 (Conv2D) (None, 384, 8, 8) 1548288 activation_175[0][0]

batch_normalization_172 (BatchNo (None, 384, 8, 8) 1152 conv2d_172[0][0]

batch_normalization_176 (BatchNo (None, 384, 8, 8) 1152 conv2d_176[0][0]

activation_172 (Activation) (None, 384, 8, 8) 0 batch_normalization_172[0][0]

activation_176 (Activation) (None, 384, 8, 8) 0 batch_normalization_176[0][0]

conv2d_173 (Conv2D) (None, 384, 8, 8) 442368 activation_172[0][0]

conv2d_174 (Conv2D) (None, 384, 8, 8) 442368 activation_172[0][0]

conv2d_177 (Conv2D)	(None, 384, 8, 8)	442368	activation_176[0][0]	
conv2d_178 (Conv2D)	(None, 384, 8, 8)	442368	activation_176[0][0]	
average_pooling2d_17 (AveragePoo	(None, 1280, 8, 8)	0	mixed8[0][0]	
conv2d_171 (Conv2D)	(None, 320, 8, 8)	409600	mixed8[0][0]	
batch_normalization_173 (BatchNo	(None, 384, 8, 8)	1152	conv2d_173[0][0]	
batch_normalization_174 (BatchNo	(None, 384, 8, 8)	1152	conv2d_174[0][0]	
batch_normalization_177 (BatchNo	(None, 384, 8, 8)	1152	conv2d_177[0][0]	
batch_normalization_178 (BatchNo	(None, 384, 8, 8)	1152	conv2d_178[0][0]	
conv2d_179 (Conv2D)	(None, 192, 8, 8)	245760	average_pooling2d_17[0][0]	
batch_normalization_171 (BatchNo	(None, 320, 8, 8)	960	conv2d_171[0][0]	
activation_173 (Activation)	(None, 384, 8, 8)	0	batch_normalization_173[0][0]	
activation_174 (Activation)	(None, 384, 8, 8)	0	batch_normalization_174[0][0]	
activation_177 (Activation)	(None, 384, 8, 8)	0	batch_normalization_177[0][0]	
activation_178 (Activation)	(None, 384, 8, 8)	0	batch_normalization_178[0][0]	
batch_normalization_179 (BatchNo	(None, 192, 8, 8)	576	conv2d_179[0][0]	
activation_171 (Activation)	(None, 320, 8, 8)	0	batch_normalization_171[0][0]	
mixed9_0 (Concatenate)	(None, 768, 8, 8)	0	activation_173[0][0]	

```
                                                    activation_174[0][0]
__________________________________________________________________________________________________
concatenate_3 (Concatenate)     (None, 768, 8, 8)    0           activation_177[0][0]
                                                    activation_178[0][0]
__________________________________________________________________________________________________
activation_179 (Activation)     (None, 192, 8, 8)    0           batch_normalization_179[0][0]
__________________________________________________________________________________________________
mixed9 (Concatenate)            (None, 2048, 8, 8)   0           activation_171[0][0]
                                                    mixed9_0[0][0]
                                                    concatenate_3[0][0]
                                                    activation_179[0][0]
__________________________________________________________________________________________________
conv2d_184 (Conv2D)             (None, 448, 8, 8)    917504      mixed9[0][0]
__________________________________________________________________________________________________
batch_normalization_184 (BatchNo (None, 448, 8, 8)    1344        conv2d_184[0][0]
__________________________________________________________________________________________________
activation_184 (Activation)     (None, 448, 8, 8)    0           batch_normalization_184[0][0]
__________________________________________________________________________________________________
conv2d_181 (Conv2D)             (None, 384, 8, 8)    786432      mixed9[0][0]
__________________________________________________________________________________________________
conv2d_185 (Conv2D)             (None, 384, 8, 8)    1548288     activation_184[0][0]
__________________________________________________________________________________________________
batch_normalization_181 (BatchNo (None, 384, 8, 8)    1152        conv2d_181[0][0]
__________________________________________________________________________________________________
batch_normalization_185 (BatchNo (None, 384, 8, 8)    1152        conv2d_185[0][0]
__________________________________________________________________________________________________
activation_181 (Activation)     (None, 384, 8, 8)    0           batch_normalization_181[0][0]
__________________________________________________________________________________________________
activation_185 (Activation)     (None, 384, 8, 8)    0           batch_normalization_185[0][0]
__________________________________________________________________________________________________
conv2d_182 (Conv2D)             (None, 384, 8, 8)    442368      activation_181[0][0]
__________________________________________________________________________________________________
conv2d_183 (Conv2D)             (None, 384, 8, 8)    442368      activation_181[0][0]
__________________________________________________________________________________________________
conv2d_186 (Conv2D)             (None, 384, 8, 8)    442368      activation_185[0][0]
```

conv2d_187 (Conv2D)	(None, 384, 8, 8)	442368	activation_185[0][0]	
average_pooling2d_18 (AveragePoo	(None, 2048, 8, 8)	0	mixed9[0][0]	
conv2d_180 (Conv2D)	(None, 320, 8, 8)	655360	mixed9[0][0]	
batch_normalization_182 (BatchNo	(None, 384, 8, 8)	1152	conv2d_182[0][0]	
batch_normalization_183 (BatchNo	(None, 384, 8, 8)	1152	conv2d_183[0][0]	
batch_normalization_186 (BatchNo	(None, 384, 8, 8)	1152	conv2d_186[0][0]	
batch_normalization_187 (BatchNo	(None, 384, 8, 8)	1152	conv2d_187[0][0]	
conv2d_188 (Conv2D)	(None, 192, 8, 8)	393216	average_pooling2d_18[0][0]	
batch_normalization_180 (BatchNo	(None, 320, 8, 8)	960	conv2d_180[0][0]	
activation_182 (Activation)	(None, 384, 8, 8)	0	batch_normalization_182[0][0]	
activation_183 (Activation)	(None, 384, 8, 8)	0	batch_normalization_183[0][0]	
activation_186 (Activation)	(None, 384, 8, 8)	0	batch_normalization_186[0][0]	
activation_187 (Activation)	(None, 384, 8, 8)	0	batch_normalization_187[0][0]	
batch_normalization_188 (BatchNo	(None, 192, 8, 8)	576	conv2d_188[0][0]	
activation_180 (Activation)	(None, 320, 8, 8)	0	batch_normalization_180[0][0]	
mixed9_1 (Concatenate)	(None, 768, 8, 8)	0	activation_182[0][0] activation_183[0][0]	

```
concatenate_4 (Concatenate)     (None, 768, 8, 8)    0          activation_186[0][0]
                                                                activation_187[0][0]
__________________________________________________________________________________________________
activation_188 (Activation)     (None, 192, 8, 8)    0          batch_normalization_188[0][0]
__________________________________________________________________________________________________
mixed10 (Concatenate)           (None, 2048, 8, 8)   0          activation_180[0][0]
                                                                mixed9_1[0][0]
                                                                concatenate_4[0][0]
                                                                activation_188[0][0]
__________________________________________________________________________________________________
avg_pool (GlobalAveragePooling2D (None, 2048)         0          mixed10[0][0]
__________________________________________________________________________________________________
new_predictions (Dense)         (None, 2)            4098       avg_pool[0][0]
==================================================================================================
Total params: 21,806,882
```

21 Appendix - Test Results

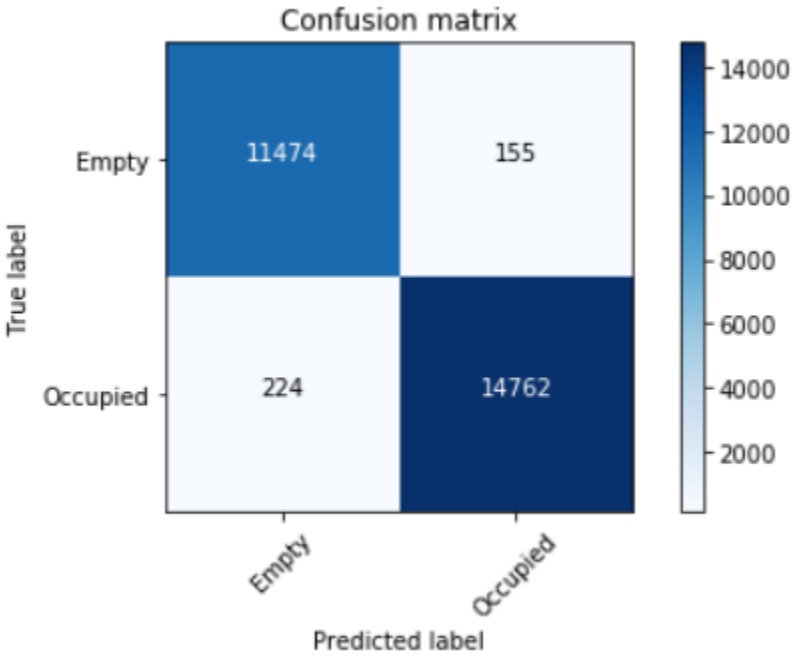


Figure 11: **Confusion matrix for Vgg16 for test 1.**

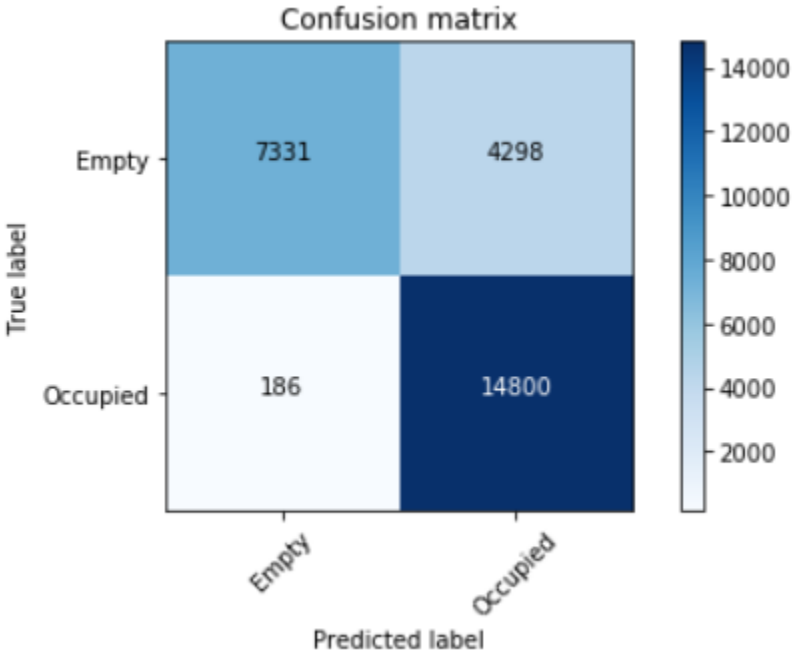


Figure 12: **Confusion matrix for Vgg19 for test 1.**

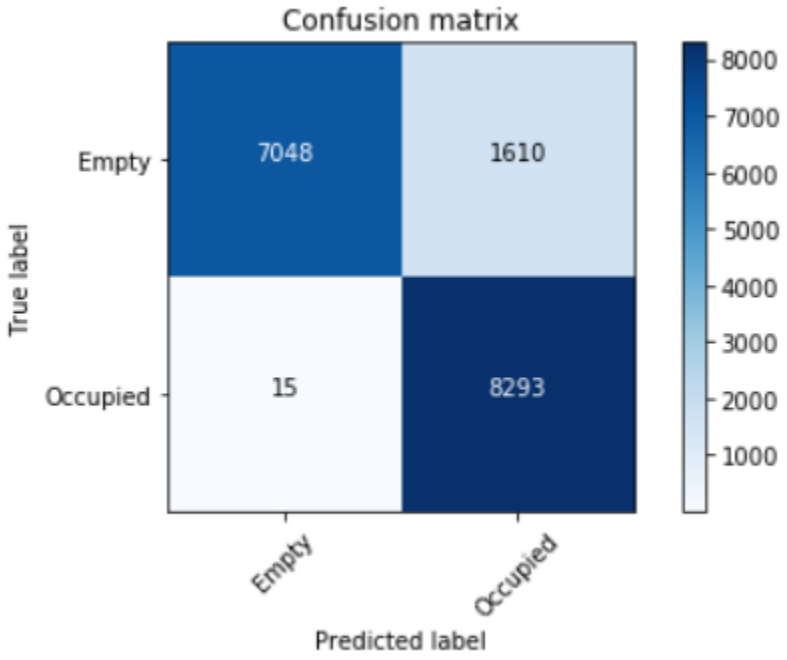


Figure 13: **Confusion matrix for Resnet50 for test 1.**

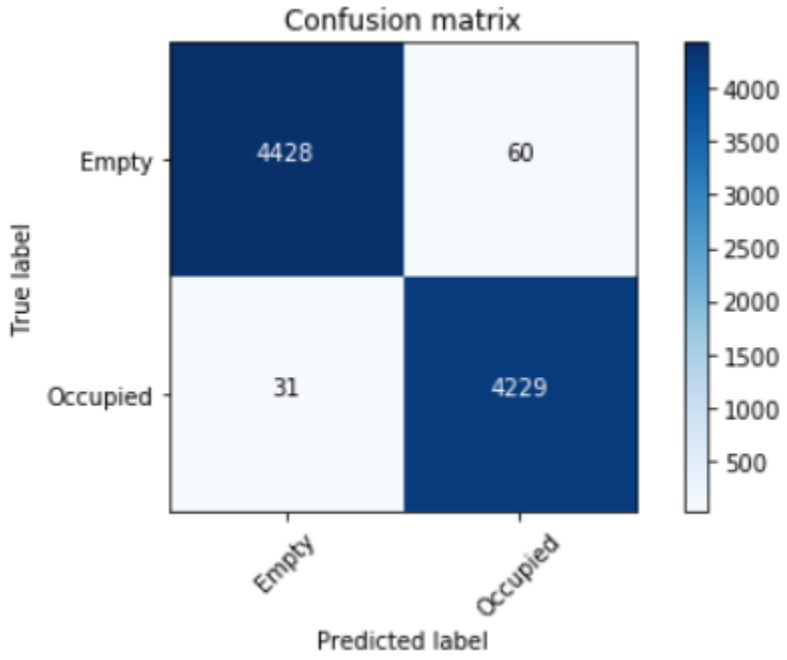


Figure 14: **Confusion matrix for Vgg16 for test 2.**

22 Appendix - Study Contract

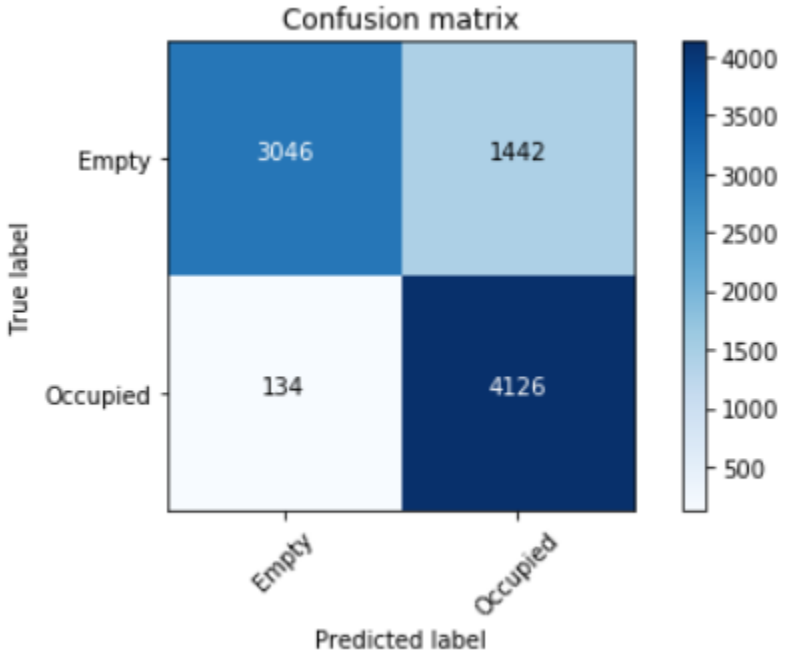


Figure 15: **Confusion matrix for Vgg19 for test 2.**

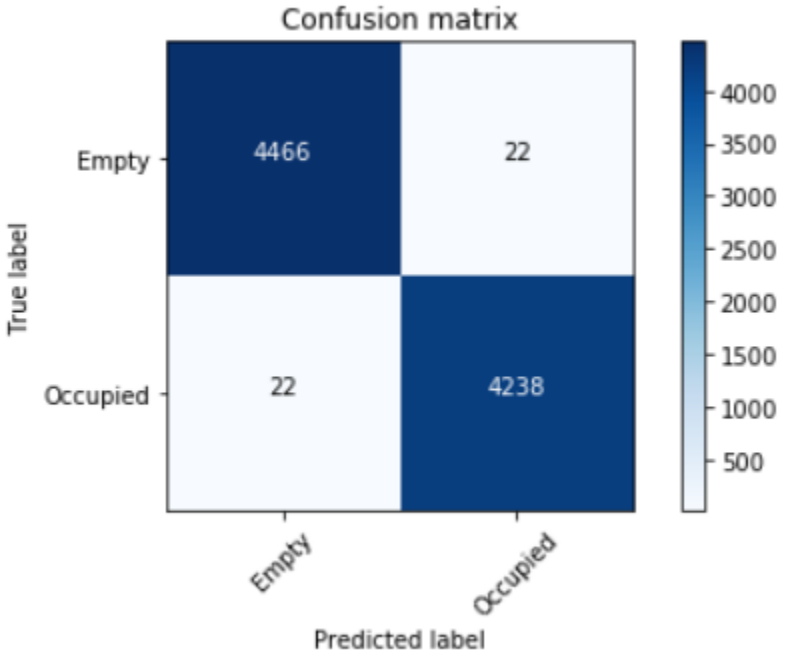


Figure 16: **Confusion matrix for Resnet50 for test 2.**

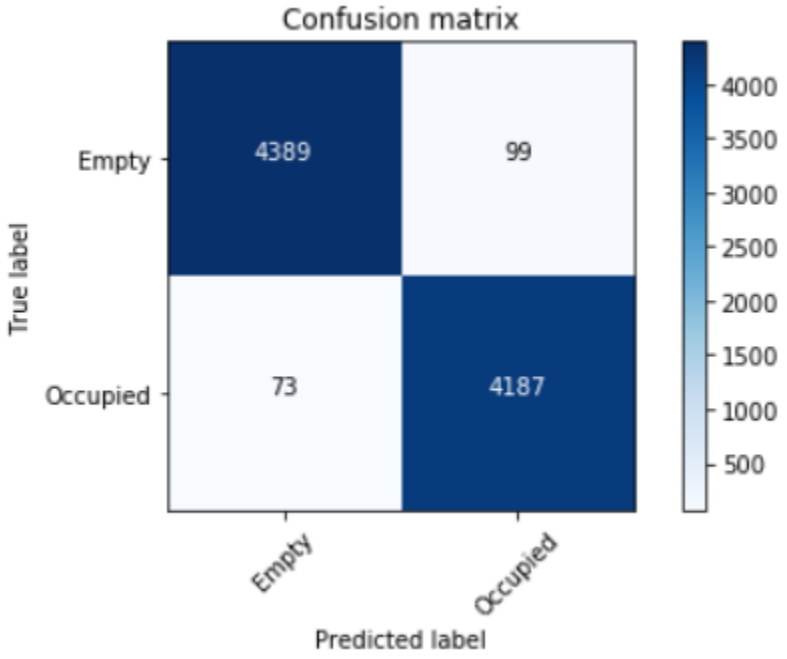


Figure 17: **Confusion matrix for Inception v3 for test 2.**

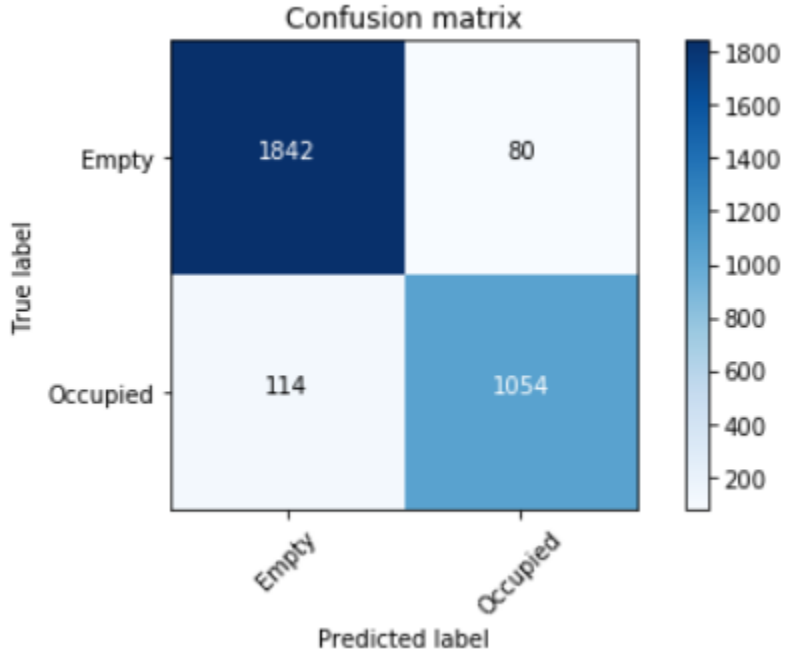


Figure 18: **Confusion matrix for Vgg16 for test 3.**

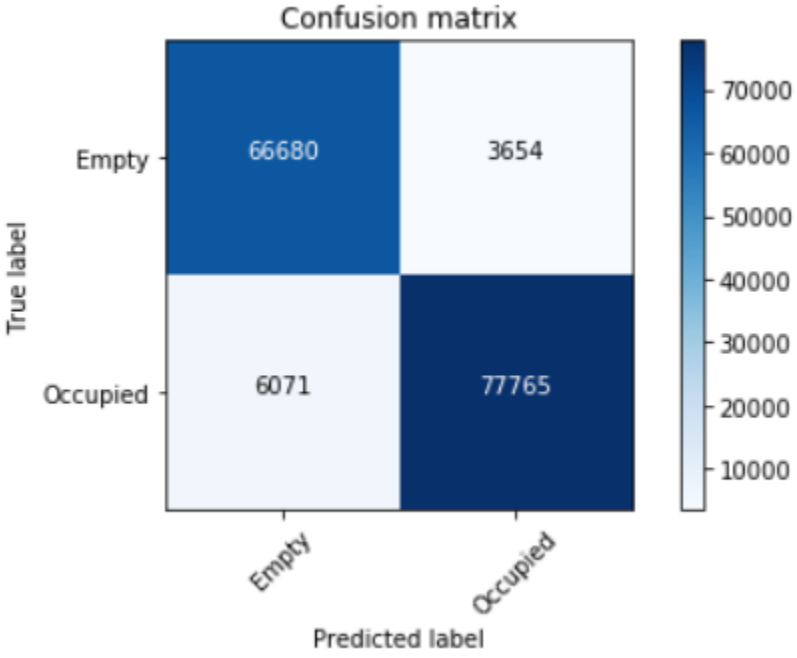


Figure 19: **Confusion matrix for Vgg19 for test 3.**

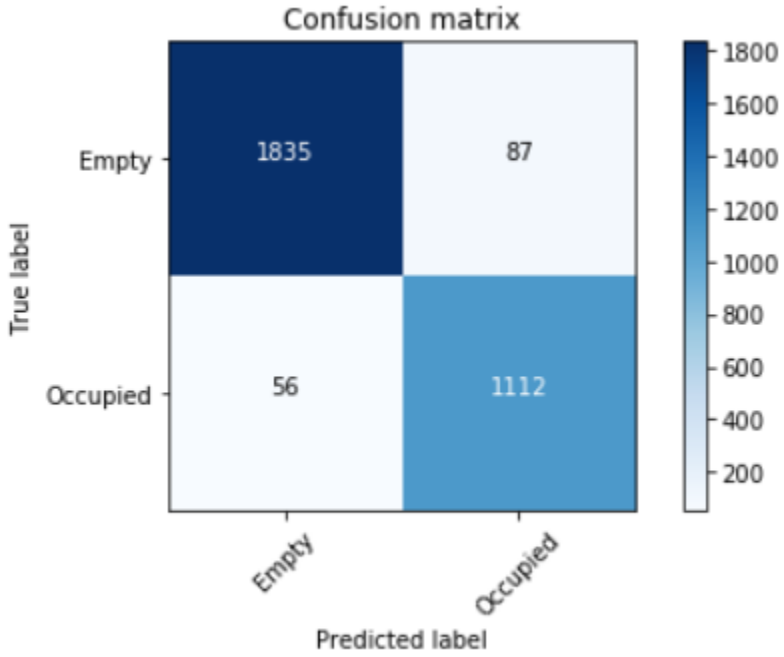


Figure 20: **Confusion matrix for Resnet50 for test 3.**

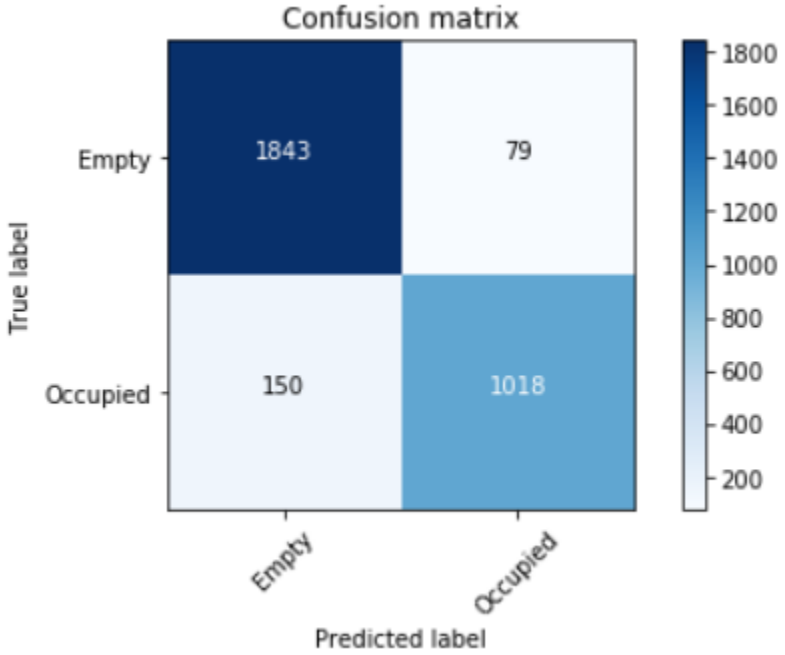


Figure 21: **Confusion matrix for Inception v3 for test 3.**

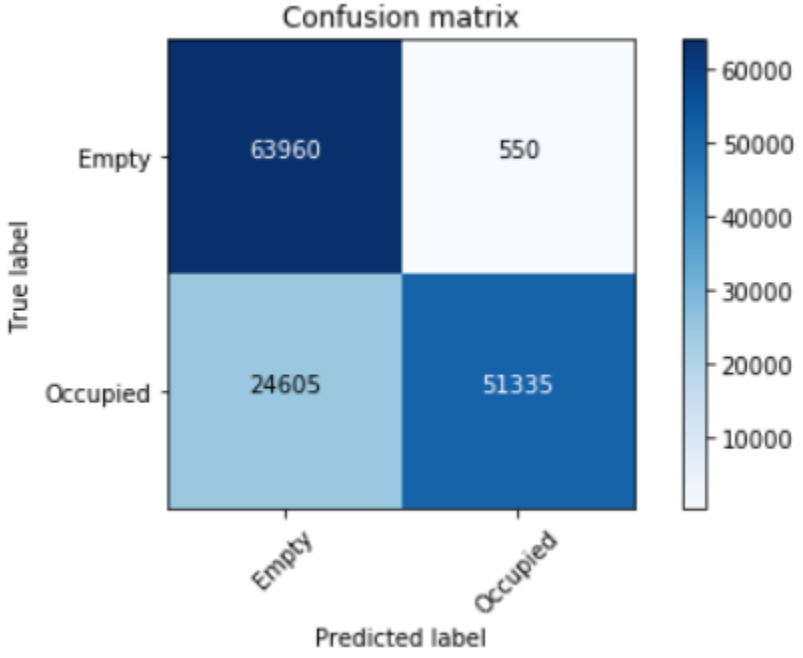


Figure 22: **Confusion matrix for Resnet50 for test 4.**

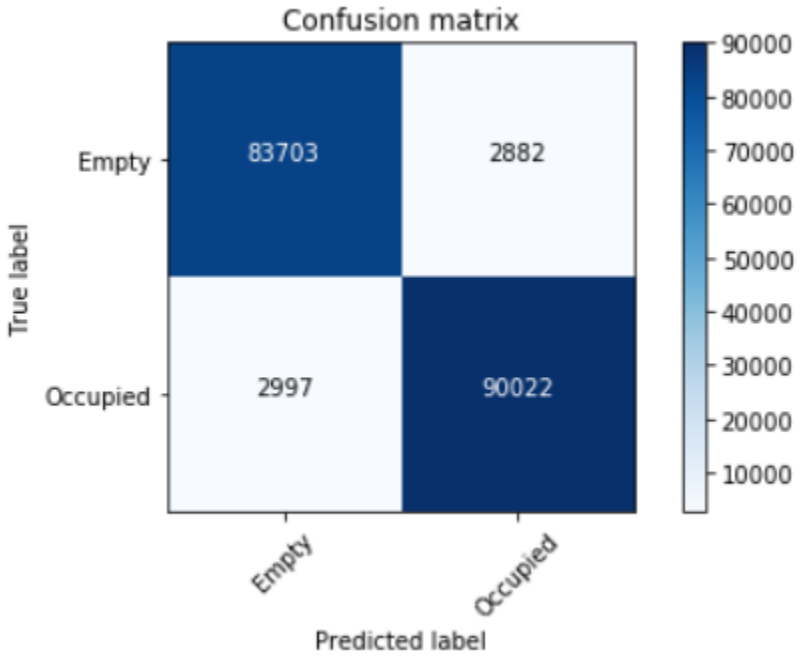


Figure 23: **Confusion matrix for Vgg16 for test 5.**

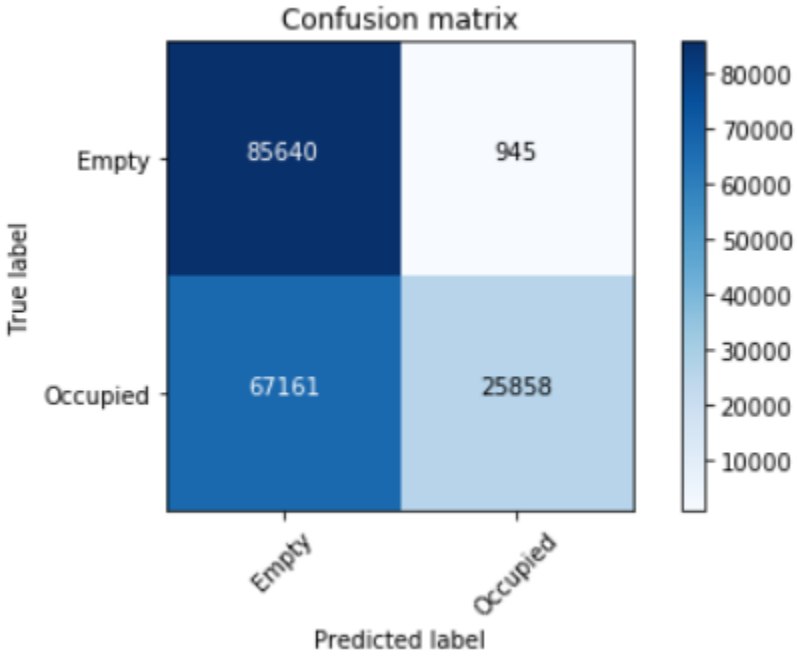


Figure 24: **Confusion matrix for Vgg19 for test 5.**

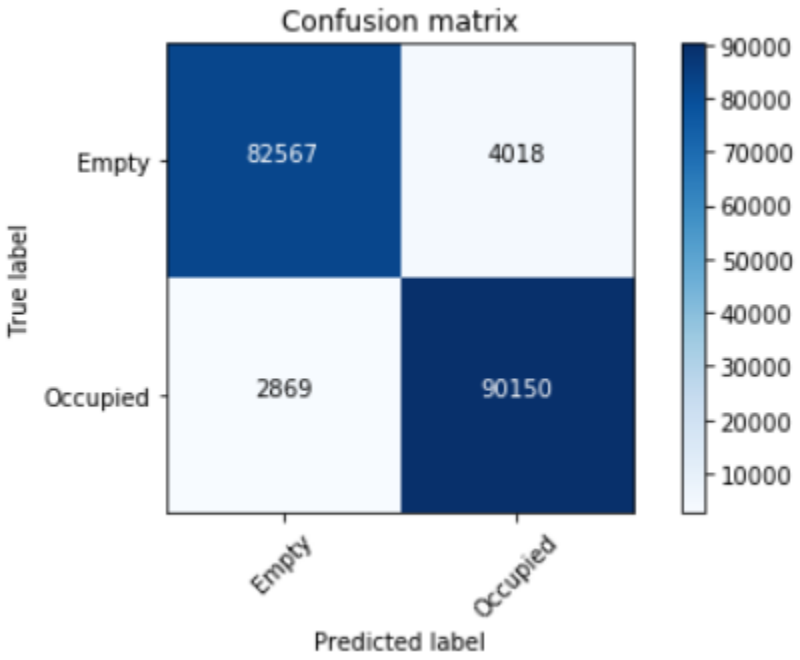


Figure 25: **Confusion matrix for Resnet50 for test 5.**

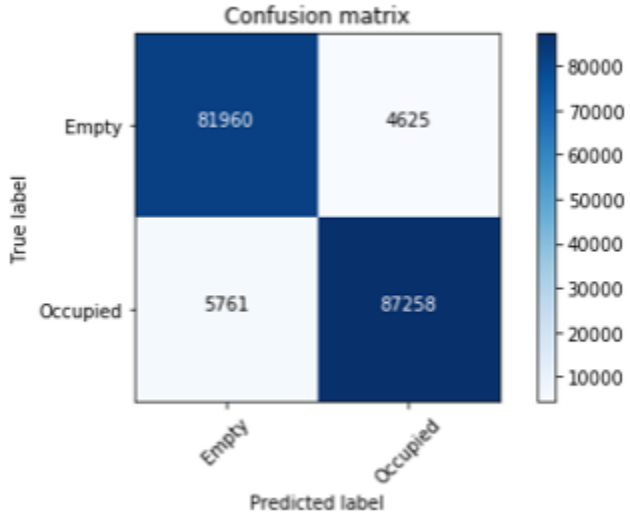


Figure 26: **Confusion matrix for Inception v3 for test 5.**

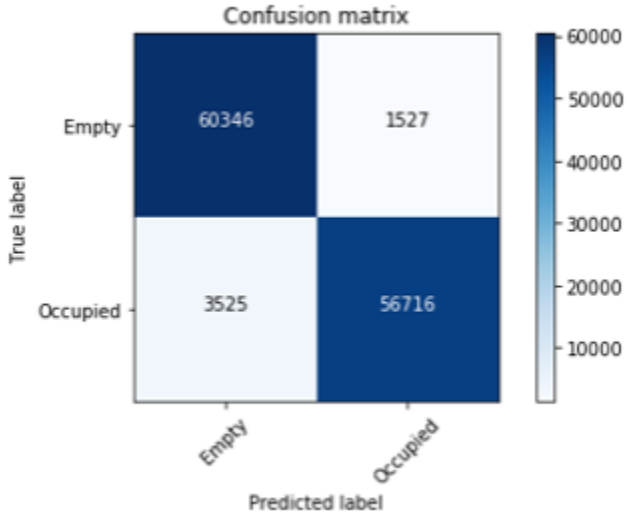


Figure 27: **Confusion matrix for Inception v3 for test 6.**

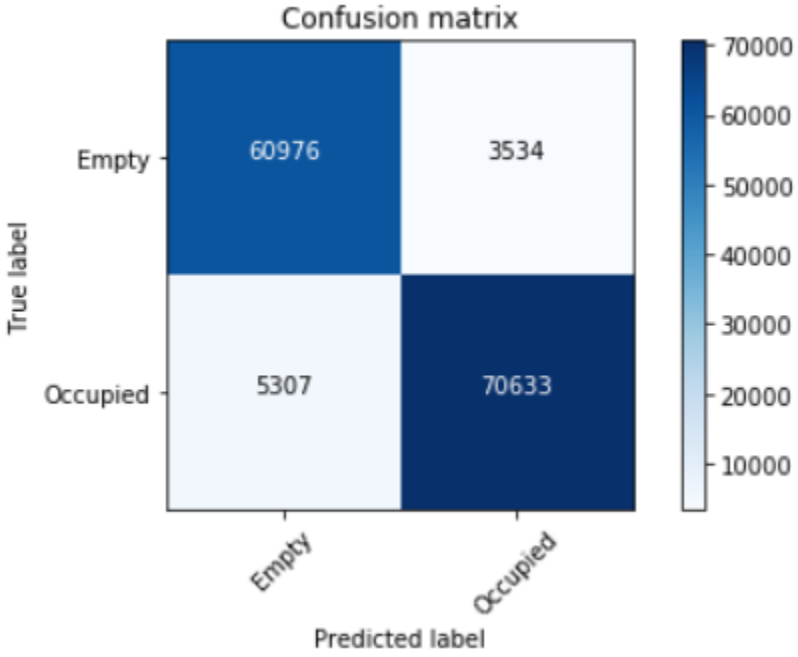


Figure 28: **Confusion matrix for Vgg16 for test 8.**

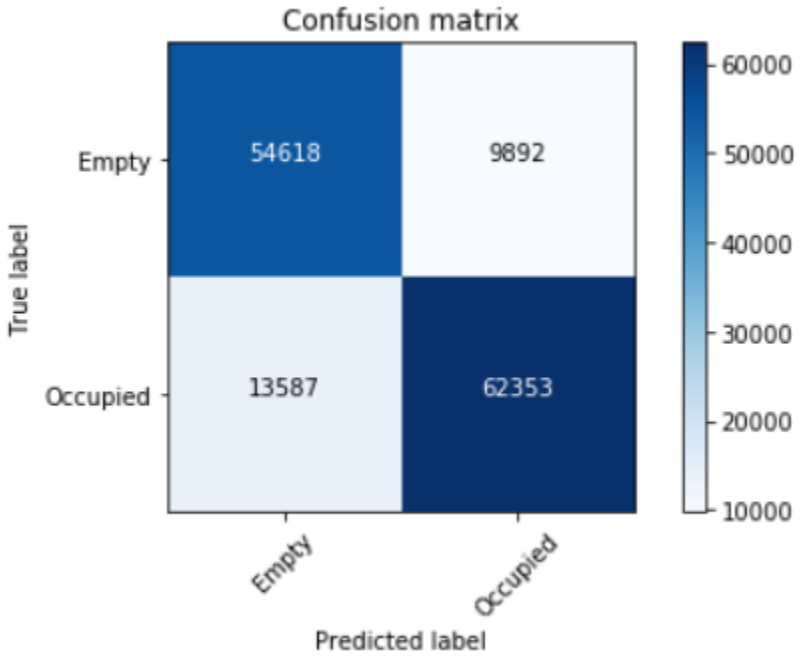


Figure 29: **Confusion matrix for Inception v3 for test 8.**

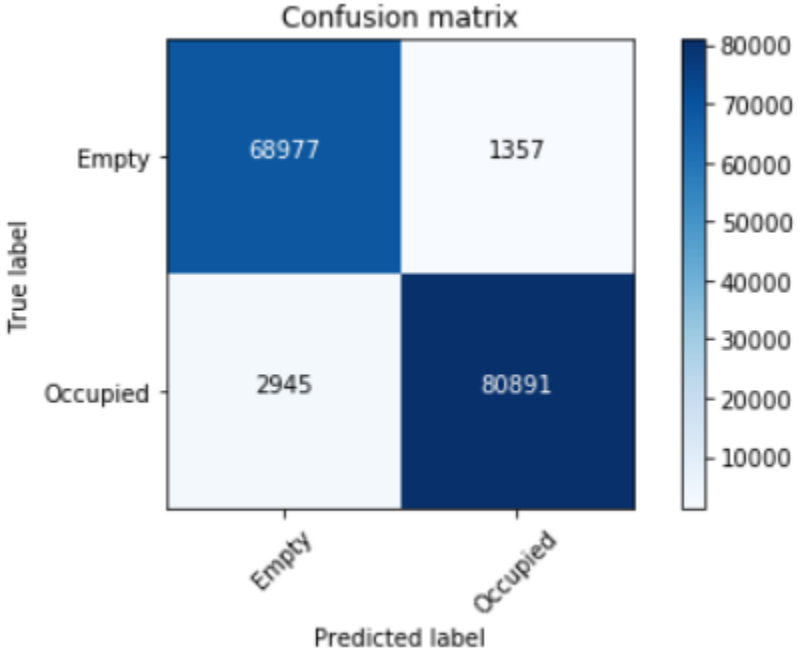


Figure 30: **Confusion matrix for Vgg16 for test 9.**

```
error:  31.3588895375
```

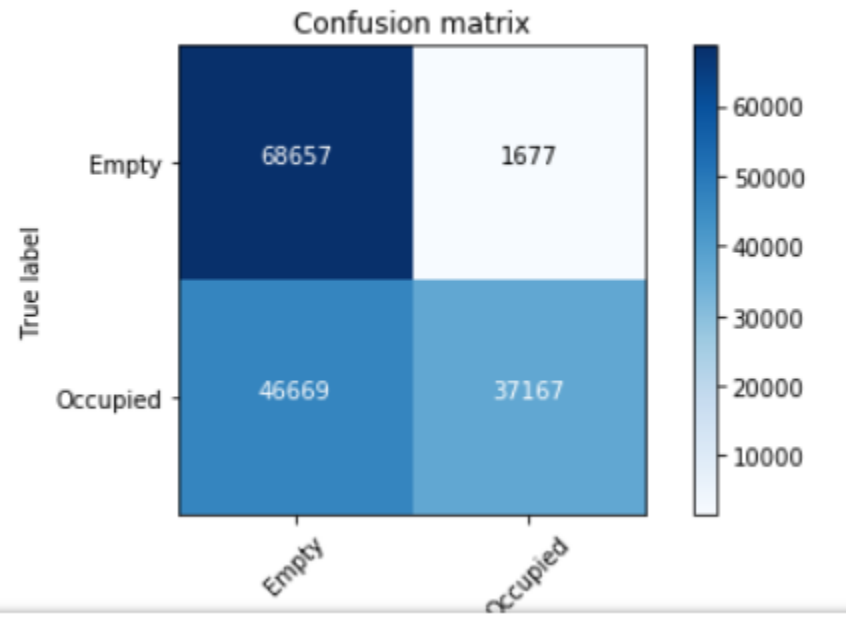


Figure 31: **Confusion matrix for Vgg19 for test 9.**

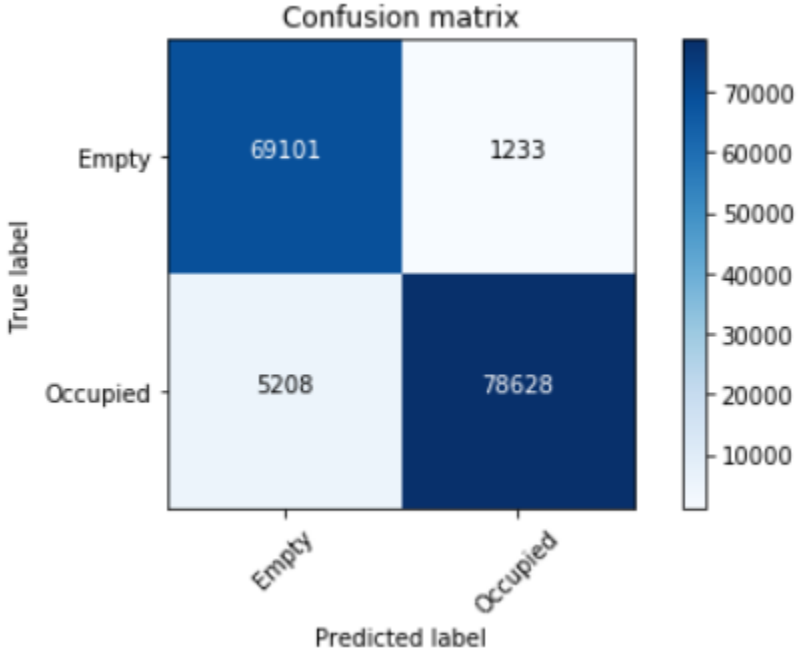


Figure 32: **Confusion matrix for Resnet50 for test 9.**

The Australian National University
School of Computer Science

Independent Study Contract

Note: Enrolment is subject to approval by the projects coordinator.

Section A (Student & Supervisor(s))

UniID: u5752631

Surname: Gardezi First names: Syed Sheece Raza

Project supervisor (may be external): Anoop Cherian

Course supervisor (a SoCS academic): Peter Strazdins TBA

Course code, title and unit: COMP8755 - Individual Computing Project

Semester: ☐ S1 ☑ S2 Year: 2017

Project title:

Identifying Availability of Parking Spot Using Deep Learning

Learning Objectives:

1) Reviewing related work done in this area
2) Designing a deep learning framework for parking spot recognition
3) Collecting a dataset of (>10000 items) annotated with regions labelled as 'occupied' or 'free'.
4) Training the deep learning framework classify image regions as 'occupied' or 'free'.
5) Development of a demo website that shows the results of the scheme.

Project Description:

It is often a difficult task to find a free parking spots. As most parking lots are equipped with surveillance cameras, we could use the video feed from such cameras to locate free parking spots. In this project, we propose to implement a deep learning based software system that will identify parking spots as 'occupied' and 'free'. Our algorithm is expected to work under varying lighting conditions, view point changes, weather conditions, locations of the camera, and indoor/outdoor settings. Further, we assume the camera is uncalibrated, however the perimeters of the parking lot and spots can be estimated from the images. As part of this project, we will collect a large dataset of more than 10K images/clips containing parking lots whose regions are annotated as 'occupied' or 'free'. Our dataset will cater to the diversity requirements specified above.

Deliverables on which the project will be evaluated on:

1. A sufficiently diverse dataset (with changes in viewpoints, weather, and lighting conditions), with atleast 10K images/videos, and their ground truth annotations. A test set of video feeds, with ground truth annotations (40%)

2. A deep learning framework written in Caffe/Torch/Tensorflow (and associated trained models) that takes as input a video feed from a parking lot (subsampled at 1 min / sec or another temporal sampling scheme) and produces as output the occupancy map of the parking lot over time. (50%).

3. A website showing the occupancy map of a parking lot that can be viewed online (10%)

Assessment (as per course's project rules web page, with the differences noted below):

Assessed project components:	% of mark	Due date:	Evaluated by:
Report: name style: Research Report (e.g. research report, software description, ...)	45		
Artefact: name kind: Dataset + software + website (e.g. software, user interface, robot, ...)	45		
Presentation:	10		

Meeting dates (if known): On thursday weekly

Student declaration: I agree to fulfil the above defined contract:

Signature ..... Date 26/7/2017

Section B (Supervisor)

I am willing to supervise and support this project. I have checked the student's academic record and believe this student can complete the project:

Signature ..... Date 29/07/2017

Required department resources:

Section C (Course coordinator approval)

Signature ..... Date 04/08/17

Section D (Projects coordinator approval)

Signature ..... Date

23 Appendix - Test Results