Université de technologie de Belfort Montbéliard
Reeferpulse

Internship report

Detection of container zones by Artificial Intelligence

Author: **Yuxin Yang**[1]
Advisor: **Vu Nam Anh LE**[1], **Rémy Ntshaykolo**[2]
Note: Data Scientist, Reeferpulse[1]
Data Scientist, Reeferpulse[2]
Data Scientist Director, Reeferpulse [3]

AIX-EN-PROVENCE, JANUARY 2023

Contents

1 Introduction

In order to gain professional experience, I am working as Data Scientist Intern for six months in the Autumn 2022 semester at the Bluepulse Group, more specifically working for the object detection part. The main purpose is to find suitable models under different requirements and environments.

In the maritime industry, containers are crucial elements, therefore, the management of containers is an important requirement in the world. To better manage these containers, a model to track the movement of containers is a real demand, it can examine the positions of container zones and analyze the situation of containers.

This research is based on some deep learning methods and satellite photos then predict if there is a zone of containers, the target is to find the method with the highest accuracy and the shortest running time. It will be used to create a 'geofencing' database for all the maritime terminals in the world based on GPS data.

2 Acknowledgement

I would like to give my heartfelt thanks to all the people who have ever helped me with this internship.

Firstly, My sincere and hearty thanks and appreciations go firstly Bluepulse for allowing me to work as the Data Scientist Intern in this company and for bringing me knowledge in this field. I would also thank to my mentor Vu Nam Anh LE for his time and kindness, whose suggestions and encouragement have given me much insight into these studies. It has been a great privilege and joy to study under his guidance and supervision.

Finally, I would like to thank UTBM for allowing me to intern at Bluepulse and my follower teacher Oumaya baala.

3 Organisation and Context

3.1 Bluepulse

Bluepulse is a start-up founded in June 2019. The start-up provides the maritime industry with AI-BASED PREDICTIVE MAINTENANCE. In a more detailed manner, Bluepulse gives maritime industry enterprises and associations important data to improve performance and make decisions exactly.

The global target of Bluepulse is to enhance the marketplace with AI technology and meet the rapidly growing reefer market, because a large percentage of the world's 35 million shipping containers are refrigerated, also known-as Reefers. This refrigeration is mandatory to ship fruits, vegetables, and perishable goods, but also all goods which may rust or degrade during the long, saline air-soaked trips. Over 100,000 new Reefers are built every year and over 4 million Reefers are in operation on the seas at any given time.

The service of maritime container tracking services has the following characteristics: global market of 10 very large shipping companies, growing 25% per year for connected containers (exp. To reach 5 to 6 million units in 2026, on a total of 22 million containers in service in 2021).

Since no shipping line has designed, manufactured and installed its container sensors, the tracking service is performed either by the IoT vendor (they design, build, sell and install sensors for shipping lines), or sometimes

by an analytic service company – such as Bluepulse service, reselling the solution to IoT vendors.
The company mainly provides the following four services:

- Shipment analytics data from existing telematics data
- Real-time notifications for problems with the electric power system on the ship
- Predict energy, based on consumption metrics at present
- Monitor energy in/out of port

3.2 Context

Figure 1: The general solution for the maritime industry

The maritime industry is collecting a massive amount of data, but is not utilizing it at its full potential. As a result, shipment delays, food waste, and containers are misplaced and lost, and client complaints are enormous. This affects the overall supply chain for any business and decreases its effectiveness.

Bluepulse helps the shipping industry become more efficient by extracting helpful insight from IoT data. They make things simple as Bluepulse provides only necessary data, is cost-effective as there is no need to invest in hardware or sensors, and make it easy to analyze by providing customized KPIs and ad-hoc analytics.

The innovation is unique because it solves a pain point for analytics. Usually, analytics from manufacturers are proprietary, and don't adapt to customers' needs. As the figure **??**, Bluepulse designs and sells customized analytical solutions shipping companies and ports can use with any reefers, ship, and equipment despite their manufacturers.

The project is inspired by the requirement of tracking containers, which requires precision to the 6th decimal place. All container zones in the world will be collected and determined exactly by AI-based containers detection.

4 Work plan

The work plan shows in the following pictures 3.

R&D work "Detection of zone container" forecast							
		sept.-22	oct.-22	nov.-22	déc.-22	janv.-23	févr.-23
WP	MILESTONES						
WP1	Understanding the project and the basic practice						
WP2	Basic knowledge of AI and Data Collection						
WP3	Test different methods						
WP4	Setting the best parameters						
WP5	Write a report and check the models						
WP6	Project management						

Figure 2: The work plan forecast

R&D work "Detection of zone container" real							
		sept.-22	oct.-22	nov.-22	déc.-22	janv.-23	févr.-23
WP	MILESTONES						
WP1	Understanding the project and the basic practice						
WP2	Basic knowledge of AI and Data Collection						
WP3	Test different methods						
WP4	Setting the best parameters						
WP5	Write a report and check the models						
WP6	Project management						

Figure 3: The work plan real

Where the WP is the 'work plan'. From the pictures above, the work plan expected and the work plan real are almost containers, some parts cost more time than the expected plan. Because of the amount of theoretical knowledge, understanding of the project and the basic practice took one more week.

For the part of writing the report and checking the models, the parameters that need to be reset are more than expected. Also, because there is no specific team for collecting the image information. So the image-collecting process also consumes more time.

5 Introduction of template

The first step to take before judging whether there is a zone container or not is to find a zone container. The Template is a method to find the most similar part of one (target) image that matches the other template image. The template image refers to an image already known, then compare the template image from pixel to pixel with a target image that has the same size, dimension, and orientation. The process of detection contains 3 steps as following:

1. Data Preparation
2. Detection of template
3. Visualization and Remove the duplicates

6 Detection of containers zone by template matching

6.1 Data Preparation

The dataset is collected from Google Earth at the camera height of 1198 m, it contains 25 target pictures and 85 template pictures from 3 locations:Le Havre (3 terminals), Fos Marseille (3 terminals), Dunkerque (1 terminal).

To detect the template image, it is important to properly define the template since this method requires the detected zone and template image to have the same size, dimension, and orientation. The steps should be as follows :

- Get the height and width of the picture
- Define a label corresponding to the template picture and a color associated with the template picture
- All information will be displayed in the visualization results and a matching threshold
- Results are decided by the judging criteria of whether the two pictures matches or not

6.2 Detection of template

The detection of template is realized based on the Computer Visual Library OpenCV, this library allows each template loops over from pixel to pixel and compared with the target picture. Because the judging criteria is a threshold, so it is necessary to choose a normalized similarity metric, which transforms the similarity of two pictures into digits, so that it can be compared with the threshold.

The normalized similarity metric is **Correlation coefficient normalized(TM_CCOEFF_NORMED)**, which is selected, this method matches the mean relative value of the template picture to the mean correlation value of the target picture: 1 denotes a perfect matching;-1 denotes a poor matching; 0 represents its random and no correlative matching.

The formula is as follows:

$$R(x,y) = \frac{\sum_{x',y'}(T'(x',y') \cdot I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T'(x',y')^2 \cdot \sum_{x',y'} I'(x+x',y+y')^2}} \tag{1}$$

6.3 Visualization and Remove the duplicates

This visualization is realized based on OpenCV.As the result shows in Fig.4, the output bounding box is thick. Because in the process of loop and comparison, if an object is detected in one location, the surrounding pixels are likely to have highly similar scores too. Therefore, the duplicated detection results should be deleted.

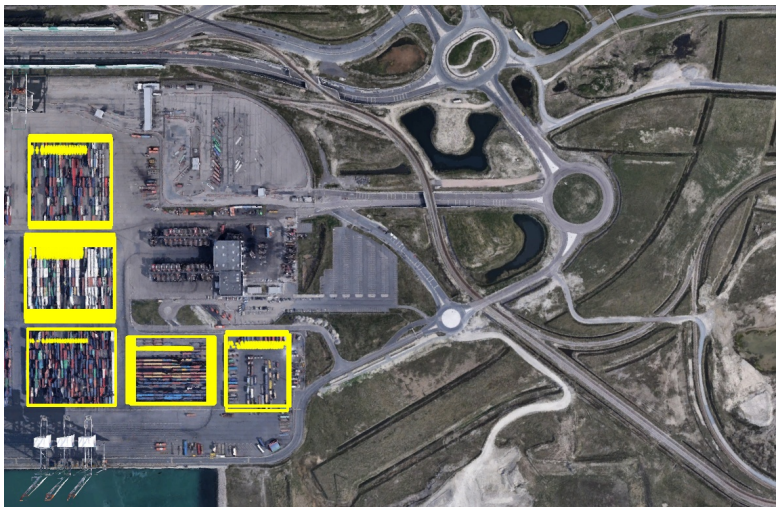


Figure 4: Duplicated detected objects

1. Remove the duplicates
   The duplicate is the detection result which is highly overlapped with the already verified detection result. In order to delete the duplicates, given a threshold of overlapping, if the IoU (Intersection over Union) is above the threshold then they delete the detecting result.
   The definition of IOU:
   $$IOU = \frac{Area\_of\_Overlap}{Area\_of\_Union} \tag{2}$$
2. Result
   The detecting objects contain containers, ships, and terminals.The result shows in Fig.5, 6, 7. Besides, some irrelevant images have also been added to verify the accuracy, the result shows in Fig.8.

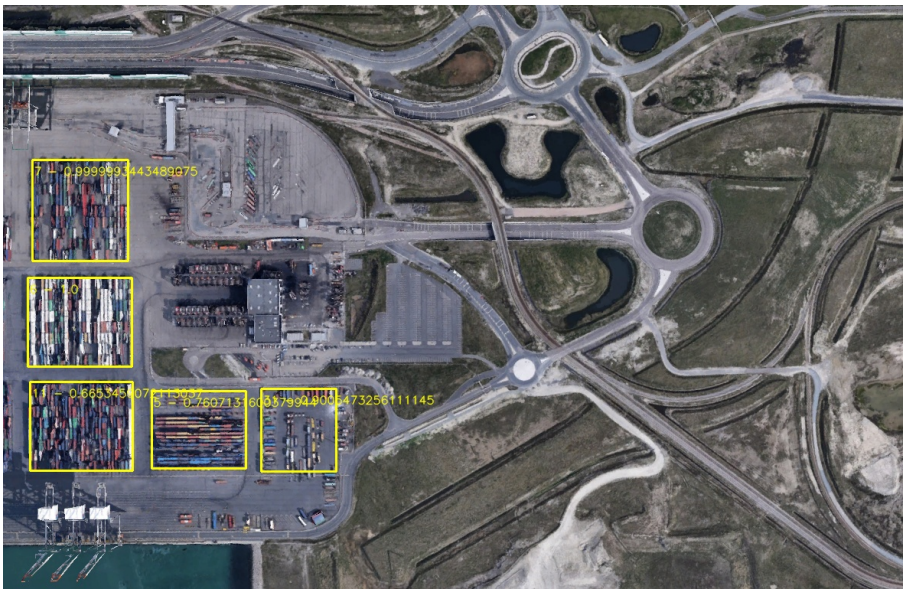


Figure 5: Result Container

Figure 6: Result Terminal

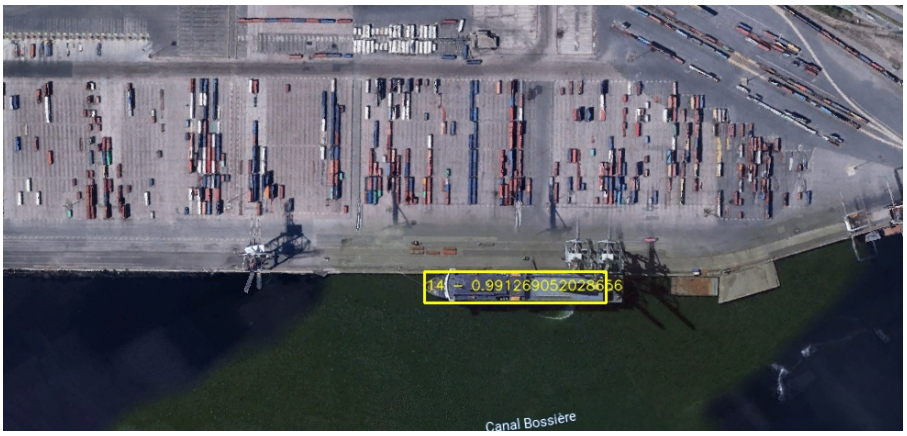


Figure 7: Result ship

Figure 8: Result Irrelevant

The confidence of all object detection is over 90% and for the irrelevant pictures, the confidence values are all less than 40%, some of them are undetectable.
The results above show that template matching is simple and straightforward way and has high confidence.

Find the full testing results in the 17.1.

7 Introduction of image classification

The purpose of these parts is to predict if the picture contains the zone container or not, to be brief, it can be considered as a binary classification problem. There are many methods and networks for the classification of images. In this research, there are two networks concentrated and optimized : A simple convolutional neural network(CNN) (also called simple CNN) and VGG16.

The CNN is a network which mainly focuses on image processing. The VGG model is a type of CNN model proposed by Oxford Visual Geometry Group at the University of Oxford. The purpose is to demonstrate that the final performance of a network can be impacted by increasing network depth to a certain extent. This article mainly compares the different performances between simple convolutional neural networks and VGG16.

These two models are based on neural networks, the image classification algorithm based on neural networks is divided into the following steps:

1. Data preprocessing
2. Construction of convolutional neural networks
3. Train the network and fine-tune

8 Data preprocessing

The dataset is collected from Google Earth at the camera height of 1198 m, it contains 352 pictures from 3 locations:Le Havre (3 terminals), Fos Marseille (3 terminals), Dunkerque (1 terminal).

8.1 Dataset Organization

To properly train and evaluate the implemented models, The dataset is divided into 3 sets: training set of 200 pictures (57%), validation set of 100 pictures (28%), test set of 52 pictures (15%). The training set and validation set contain 2 classes : port (which contains zone containers) and no_port(which does not contain the zone containers).

Since it is a small data set, in order to reach a better result, the data needs to be augmented.

8.2 Data Augmentation

1. Rotation and Zooming

   (1)Rotation

   Rotation is realized based on the function $rotate(img, angle)$ of Library SciPy. SciPy is a popular software package for mathematics, science, engineering that handles image processing etc. In this part, all the pictures in the training set are rotated 90 degrees.

   (2)Zooming

   Zooming is realized based on the function $zoom(img, zoom_tuple, **kwargs)$ of Library SciPy. For multi-channel images (like RGB images), instead of applying the zoom factor to the RGB dimension, it is better to create a tuple of zoom factors, each factor corresponds to a dimension. In this part, the pictures in the training set are reduced to half the size of the original image.

   After the two processing above, the number of images increased to 842. The distribution of the dataset becomes: training set of 690 pictures (82%), validation set of 100 pictures (12%), test set of 52 pictures(6%).

2. ImageDataGenerator
   ImageDataGenerator is an image generator from Library keras, it allows applying batch processing (transformation, normalization) to the image to achieve data augmentation without saving a lot of images on the disk.

   There are two *ImageDataGenerator* objects:
   train_datagen: Process the training set.(1)Rescales all the images to the same size of 150×150, because CNN network cannot directly process images of different sizes. (2)Rotate all the images at a random angle from 0 °to 40 °, to pass more images to the model and perform the data augmentation
   valid_datagen: Process the validation set. Only rescales all the images to the same size of 150×150.

9 Convolutional neural networks

Convolutional neural network (CNN, or ConvNet) is a type of artificial neural network(ANN), most commonly used in image processing. The biggest difference from general artificial neural networks is that CNN has convolutional layer(s) because the input data has two dimensions (image).

An artificial neural network is a non-linear model composed with node layers, each node layer contains an input layer, one or more hidden layers, and an output layer becomes is the j-th node of the network, with respect to parameter θ, input x, output $y = f(x, \theta)$.

The calculation process of this layer is as follows:

Suppose there is a function f_j of the input $x = (x_1, x_2, ..., x_d)$,then the output is the result of the function f_j.

To calculate the function f_j, there is:

1. A vector of connection weights $w_j = (w_{j,1}, ..., w_{j,d})$
2. A neuron bias b_j

These values are associated with an activation functionϕ.

$y_j = f_j(x) = \phi(< w_j, x > +b_j)$

A typical CNN consists of 3 parts:

1. Convolutional layer——Extract features
2. Pooling layer——Data dimensionality reduction to avoid overfitting
3. Fully connected layer——Output the result

9.1 Convolutional layer

The main building blocks of convolutional neural networks are convolutional layers.
The convolutional layer and its parameters are composed of some filters (kernels). In each layer, each filter has a small size, for example, the most commonly used are 3×3,5×5 and 7×7followsput depth for each filter and the number of filters are the same. The main function of a convolution layer is to repeat applied convolution

operation to the same filter and the input image (input feature image), then output a feature map. This process will continuously extract features through one filter after another, from local features to overall features, finally realize the image feature extraction.

2D-Convolution is often used in image processing, because an image is 2-dimension data. The formula is as follows:

Discrete 2D-Convolutional:

$$(f * g)(x, y) = \sum \sum f(n, m) \cdot g(x - m, y - n) \tag{3}$$

The image has 2 Dimensions, but the image data can be a multidimensional array, for example, the colored image has 3 dimensions (width, height, channel(RGB)).

$$(F * Img)(x, y) = \sum_{c} \sum_{m} \sum_{n} F^{c}(n, m) \cdot Img^{c}(x + m, y + n) \tag{4}$$

9.2 Pooling layer

The main function of the pooling layer is to increase the speed of operation and prevent the model from overfitting by extracting the main features of a certain area (The size of filter)and reducing the number of parameters.
Pooling layer does not have convolutional operations. The two most commonly used methods are *Max Pooling* and *Average Pooling*.

1. Max Pooling: it takes the maximum value in the sliding area of the filter.
2. Average Pooling: it takes the average value in the sliding area of the filter.

Maxpooling is more often used in real case because large number means that strengthened certain characteristics and ignored other values, so it can reduce noise effects and performs better.

9.3 Fully connected layer

A fully connected layer means that all the nodes of this layer are connected to the previous layer. It maps the high-dimension feature map to a one-dimension feature vector, the vector contains all the feature information and can be converted into the probability of each category. The main function of a fully connected layer is to synthesize the features extracted from the front. Due to its characteristics (fully connected), it also has the most parameters.

The input image size at the beginning of the network must be fixed to ensure that the size of the feature map transmitted to the fully connected layer matches the weight matrix of the fully connected layer. On the other hand, the size of the input of each feature map can be multiplied with the weight matrix because the weight matrix of the fully connected layer is fixed. For this reason, the images require rescaling to the same size.

9.4 Construction of Convolutional neural network

Layer	Output Layer	Param#
conv2d (Conv2D)	(None, 148, 148, 32)	896
activation (Activation)	(None, 148, 148, 32)	0
max_pooling2d (MaxPooling2D)	(None, 74, 74, 32)	0
conv2d_1 (Conv2D)	(None, 72, 72, 32)	9248
activation_1 (Activation)	(None, 72, 72, 32)	0
max_pooling2d_1 (MaxPooling2	(None, 36, 36, 32)	0
conv2d_2 (Conv2D)	(None, 34, 34, 64)	18496
activation_2 (Activation)	(None, 34, 34, 64)	0
max_pooling2d_2	(None, 17, 17, 64) 0	
flatten (Flatten)	(None, 18496)	0
dense (Dense)	(None, 64)	1183808
activation_3 (Activation)	(None, 64)	0
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65
activation_4 (Activation)	(None, 1)	0

Total parameters: 1,212,513
Trainable parameters: 1,212,513
Non-trainable parameters: 0

1. **Model architecture**
   The simple CNN model is composed of 3 convolution blocks with the following form :
   1)A Conv2D layer with 32-3×3 filters and a Relu activation function:$param = (3 * 3 * 3 + 1)X32 = 896$
   2)A MaxPooling layer with 2×2 window.

   Followed by
   3)A flatten layer.
   4)A Dense layer with 64 neurons and a Relu activation function:$param = (18496 + 1)X64 = 1183808$
   5)A Dropout layer with a 50% drop rate.
   6)A Dense layer with 1 neuron and a softmax activation function.

2. **Loss Function**

The loss function is a function used to measure the degree of inconsistency between the predicted value $f(x)$ of the model and the real value Y. Because this is a binary classification problem, the used loss function is Binary cross-entropy:

$$Loss = \frac{-1}{N_{train}} \sum_{i=1} N_{train}[y_i log\widetilde{y}_i + (1 - y_i)log(1 - \widetilde{y}_i)] \tag{5}$$

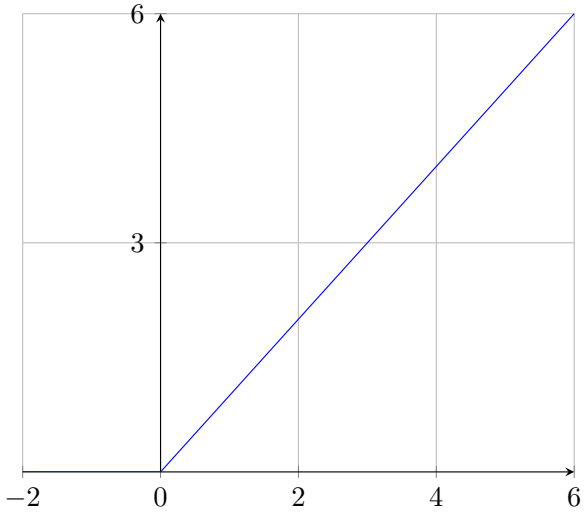
wherey_i is a binary label, $y_i \in [0, 1]$,$\widetilde{y}_i$ is the probability that the output belongs to y_i, $\widetilde{y}_i \in P(X_i = 1)$. Binary cross-entropy is used to evaluate the degree of quality of the prediction results of a binary classification model, in the situation where the label y_i is 1, if the predicted value $\widetilde{y}_i$ approaches 1. Then, the loss function value should approach 0. On the contrary, if the predicted value $\widetilde{y}_i$ approaches 0 at this time, then the loss function value should be very large.

3. **Activation Function**

The Activation Function is to add the non-liner structure in the neural network. In this model, ReLu Activation and Sigmoid Activation are used.

1)ReLu activate the convolutional layer:

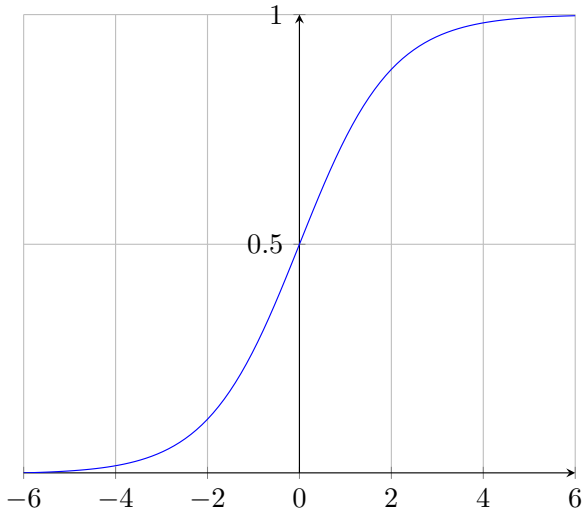
$$ReLu = max(0, x) \tag{6}$$



Choosing Relu as the activation function can ensure that the gradient = 1 when x > 0, thereby improving the operation speed of the gradient descent algorithm. However, when x < zero, there is a disadvantage of a gradient = 0.

2)Sigmoid activate the last layer:

$$Sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{7}$$



The sigmoid is generally chosen as the activation function because the output value of the binary classification problem is $\{0, +1\}$.

4. **Optimizer**

   The purpose of the optimizer is to train the network and find the appropriate parameters to make the loss function of the neural network as small as possible.

   RMSProp (Root Mean Square Prop) can make different parameters have personalized learning rates.

   This algorithm calculates the differential squared weighted mean for the gradient. This procedure is conducive to eliminating the direction of the large oscillation amplitude, which is used to correct the oscillation amplitude, so that to keep the oscillation amplitude of each dimension small. On the other hand, this practice also makes the network model converge faster.

   Its formula:

$$s_t \leftarrow \gamma s_{t-1} + (1-\gamma) g_t \odot g_t \tag{8}$$

$$\theta_t \leftarrow \theta_{t-1} - \frac{\eta}{\sqrt{s_t + \epsilon}} \odot g_t \tag{9}$$

   where γ is the decay rate,$0 < \gamma < 1$,s_t is the state variable at time t, and is the sum of squares of all gradients from the start of training until time t,θ_t is the independent variable of time t,η is the learning rate, ϵ is a constant to maintain numerical stability (to avoid numerical overflow), usually very small.

10 VGG16

The reason for choosing to use the VGG16 network is because of their good performance in the image classification problem. VGGNet is a model proposed by the Visual Geometry Group, and it has achieved good results in the ILSVRC held in 2014 for the 1st positioning task and the 2nd place for the classification task. VGGNet has a simple structure and performs well, which make the model popular.

The structure of the VGG network is simple and consistent, it uses 3×3 convolution (with a unified size of 1,unified padding of 1)and 2×2 max pooling (with a step ize of 2,unified padding of 0) instead of large convolutional kernels (11×11, 7×7, 5×5) ,this model contains 16 hidden layers (13 convolutional layers and 3 fully connected layers), these two important features proved that the performance of the model can be improved more effectively by using smaller convolutional kernels and increasing the depth of the convolutional neural network.

10.1 Model Architecture

The complexity of the data makes it difficult to build a classifier from scratch quickly, so using a pre-trained model can be more efficient. To make the process easier, it is possible to download VGG16 model from Keras, it is a very complex model that have been trained on a ImageNet in order to perform the classification valid. The table below represents the architecture of VGG 16. This model is composed of 5 convolutional blocks, which allows building features on the images. The last block is a fully convolutional block, used on the features build by the convolutional block.

The data will be sent through the 5 convolutional blocks in order to build features, these blocks have already been trained on a huge amount of data. Then, the parameters of the fully connected layer will be changed.

10.2 Parameter

Layer	Output Layer	Param#
input_1 (InputLayer)	(None, 150, 150, 3)	0
block1_conv1 (Conv2D)	(None, 150, 150, 64)	1792
block1_conv2 (Conv2D)	(None, 150, 150, 64)	36928
block1_pool (MaxPooling2D)	(None, 75, 75, 64)	0
block2_conv1 (Conv2D)	(None, 75, 75, 128)	73856
block2_conv2 (Conv2D)	(None, 75, 75, 128)	147584
block2_pool (MaxPooling2D)	(None, 37, 37, 128)	0
block3_conv1 (Conv2D)	(None, 37, 37, 256)	295168
block3_conv2 (Conv2D)	(None, 37, 37, 256)	590080
block3_conv3 (Conv2D)	(None, 37, 37, 256)	590080
block3_pool (MaxPooling2D)	(None, 18, 18, 256)	0
block4_conv1 (Conv2D)	(None, 18, 18, 512)	1180160
block4_conv2 (Conv2D)	(None, 18, 18, 512)	2359808
block4_conv3 (Conv2D)	(None, 18, 18, 512)	2359808
block4_pool (MaxPooling2D)	(None, 9, 9, 512)	0
block5_conv1 (Conv2D)	(None, 9, 9, 512)	2359808
block5_conv2 (Conv2D)	(None, 9, 9, 512)	2359808
block5_conv3 (Conv2D)	(None, 9, 9, 512)	2359808
block5_pool (MaxPooling2D)	(None, 4, 4, 512)	0

Total parameters: 14,714,688

Trainable parameters: 14,714,688

Non-trainable parameters: 0

The model VGG16_without_top is composed of 5 convolution blocks.

1)input layer The default input data of the VGG16 convolutional neural network must be an image with dimensions of 150×150×3,and the color channels it has are R, G, and B.

2)block1_conv1: The convolution kernel used is (3*3*3)*64 (convolution kernel size is 3×3, input channel

is 3, output channel is 64), step size is 1, and padding is 1. The Parameter of this layer is (3*3*3+1)*64 = 1792.

3)block1_conv2:The convolutional kernel used is (3*3*64)*64, the step size is 1, and the padding is 1. The parameter of this layer is (64*3*3+1)*64 = 36928.

4)Maxpooling: The pooled kernel size is 2×2, and the step size is 2.

5)block2_conv1: The convolution kernel used is (3*3*64)*128, the step size is 1, and the padding is 1. The parameter of this layer is (64*3*3+1)*128 = 73856.

6)block2_conv2: The convolutional kernel used is (3*3*128)*128, the step size is 1, and the padding is 1. The parameter of this layer is (128*3*3+1)*128 = 147584.

7)Maxpooling: The pooled core size is 2×2, and the step size is 2.

8)block3_conv1: The convolutional kernel used is (3*3*128)*256, the step size is 1, and the padding is 1.The parameter of this layer is (128*3*3+1)*256 = 295168.

9)block3_conv2: The convolutional kernel used is (3*3*256)*256, the step size is 1, and the padding is 1. The parameter of this layer is (256*3*3+1)*256 = 590080.

10)block3_conv3: After (3*3*256)*256 convolutional kernel, The parameter of this layer is (256*3*3+1)* 256 = 590080.

11)Maxpooling:(2*2) maxpool.

12)block4_conv1: After (3*3*256)*512 convolutional kernel, the parameter of this layer is (256*3*3+1) * 512 = 1180160.

13)block4_conv2: After (3*3*512)*512 convolutional kernel, the parameter of this layer is (512*3*3+1)* 512 = 2359808.

14)block4_conv3the parameter of this layer is (256*3*3+1)*512=1180160.

15)Maxpooling 2*2 convolution.

16)block5_conv1: After (3*3*512)*512 convolutional kernel, the parameter of this layer is (512*3*3+1)*512 = 2359808.

17)block5_conv2:After (3*3*512)*512 convolutional kernel, the parameter of this layer is (512*3*3+1)*512 = 2359808.

18)block5_conv3:After (3*3*512)*512 convolutional kernel, the parameter of this layer is (512*3*3+1)*512 = 2359808.

19)Maxpooling 2*2 convolution.

10.3 Fine-tuning

The fine-tuning process is often used in some complex models, so it is only applied in the VGG16 model, not the CNN model.

Because the available dataset is not large enough, so instead of training the model from scratch, it is better to use a pre-trained VGG16 network and apply fine-tuning. Fine-tuning will modify the output layer and fine-tune the parameters of several layers before the last layer. This method can avoid the need to design a complex model and time-consuming training. So to really improve the VGG16 model, it would be better to use fine-tuning and make them fit our problem.

In this part, to build a model which is composed of the 5 convolutional blocks of the VGG16 model (with its weights learned on ImageNet) and the classifier block of simple CNN (its weights are shown previously).

1. Model Creation
   Firstly, download the model *model_ VGG16_ without_ top* as done previously.

2. Freeze Block
   The model contains a huge number of parameters that the computer may not handle, so fine-tuning only the last block of convolution of the model can solve this problem.
   This is possible by updating the trainable arguments of the layers that should not be updated.

3. Learning rate
   The learning rate is a parameter that controls the converge speed of the loss function at each iteration, it moves toward the minimum of the loss function.

4. Callback
   Callbacks are a group of functions to be applied at particular times throughout the training process. In Keras, callback is a specific Library that is used to interrupt training, save the model, charge different weights, or override the model state according to different performance and status of the model.
   EarlyStopping() is a type of callback function, in this article, it is used to stop the training process in advance when the loss is no more decreasing and also avoid overfitting.

10.4 Optimizer

The optimizer used is a Stochastic Gradient Descent Algorithm with momentum.

Stochastic gradient descent(SGD) is a very popular optimization method, when each data is read in, it will update the parameters by calculating the loss function, so its learning process can be slow sometimes. The momentum method is designed to accelerate the gradient descent, the momentum parameter α determines how quickly the contribution of the previous gradient decays. In that case, this method has both accuracy and speed.

Algorithm 1. Stochastic Gradient Descent Algorithm with momentum

Require: learning rate ϵ , momentum $\alpha \in [0, 1]$
Require: initial parameter θ , initial velocity v
  while Stopping criterion not met **do**
    Sample a minibatch B of size m from the learning sample
    Compute the gradient estimate:
    $g \leftarrow \frac{1}{m} \sum_{i \in B} \nabla_\theta \mathcal{L}(f(X_i, \theta), Y_i)$
    Update the velocity :$v \leftarrow \alpha v - \epsilon \beta$
    Update the parameter:$\theta \leftarrow \theta + v$
  end while

11 Evaluation and Comparison

During the training process, there may be problems of under-fitting (due to excessive deviation)and over-fitting (caused by excessive variance).In order to show the performance of the model more intuitively, it is important to have a set of evaluation indicators.There are many indicators to evaluate the models, for example, AUC, precision and recall.

In the research, it is set to input image size = 150 × 150, epoch = 10, batch_size= 10, learning rate = 1e-5, then evaluate the two models from following indicators: Accuracy and Loss, confusion matrix, precision and recall, F-1 score.

11.1 Accuracy and Loss

The most commonly used indicator in model evaluation is accuracy and loss.

Accuracy is a percentage value, It refers to the proportion of samples that were correctly predicted to all samples that were predicted.

Loss is not a percentage value, it is the sum of all errors from one epoch, calculated by a loss function.is For example, in the research, the loss is calculated by Binary Cross-Entropy.

When the model is not overfitting, the smaller the loss, the better the model is.

1. CNN
   The result shows in Fig.9. The performance is improving over time.
   For the training set, after 10 epochs, the accuracy of CNN reaches at around 90%, the loss of CNN reaches at around 0.2.
   For the validation set, there is a big vibration, but the performance is still improving over time.The accuracy reaches at around 90%, the loss reaches at around 0.6.

2. VGG16
   The result shows in Fig.10. The performance is improving over time.
   For the training set, after 10 epochs the accuracy of the VGG16 reaches at around 98%, the loss of VGG16 reaches at 0.06.
   For the validation set, there is also a vibration, but smaller than the CNN model. The accuracy reaches at around 98%, and the loss reaches at 0.08.

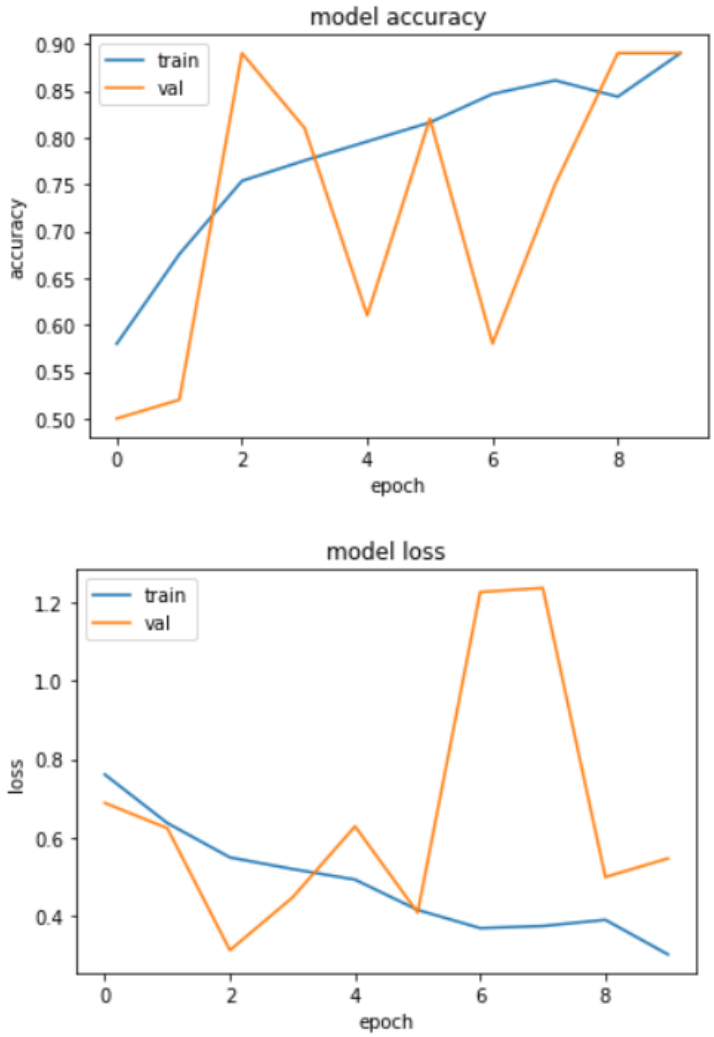


Figure 9: Model Accuracy and Loss of CNN

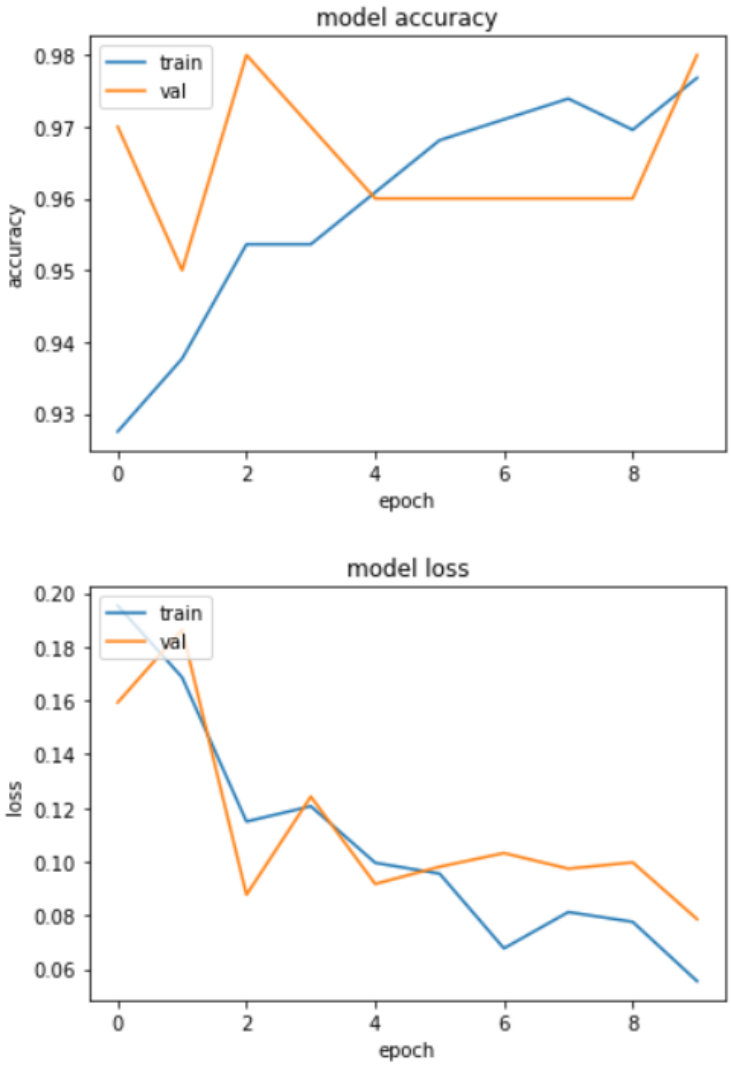


Figure 10: Model Accuracy and Loss of VGG16

11.2 Confusion Matrix

Confusion matrix is also a commonly used indicator, it is a specific table layout that shows the relationship between the predicted value and the actual value. The instances in the real class are represented in each row of the matrix, whereas the instances in the predicted class are represented in each column.

Test set contains 52 pictures, 27 of 'port' pictures and 25 of 'no_port' pictures.

1. CNN

   The result shows in Fig.11.The CNN model incorrectly predicted 15 'port' images as 'no_port' images, and the 'no_port' images are all correctly predicted.

2. VGG16

   The result shows in Fig.12.The VGG16 model correctly predicted all the 'port' and 'no_port' images.

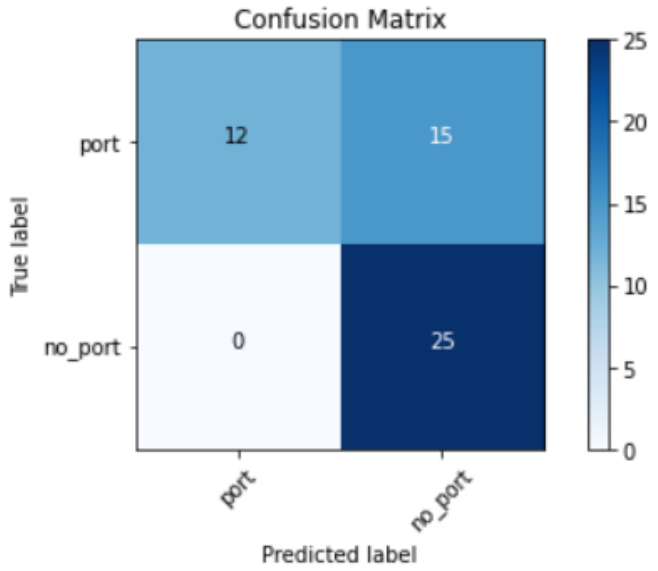


Figure 11: Confusion Matrix CNN

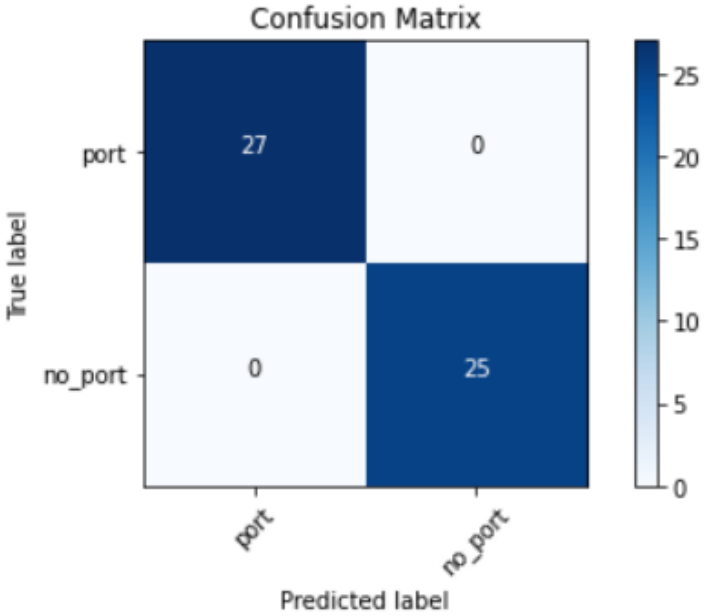


Figure 12: Confusion Matrix VGG16

11.3 Confidence

The confidence of the model is an indicator of how accurate is the result estimation.The confidence of the simple CNN is at around 60% and the confidence of VGG16 is at almost 100%.

The full confidence results can be found in the Annex 17.2.

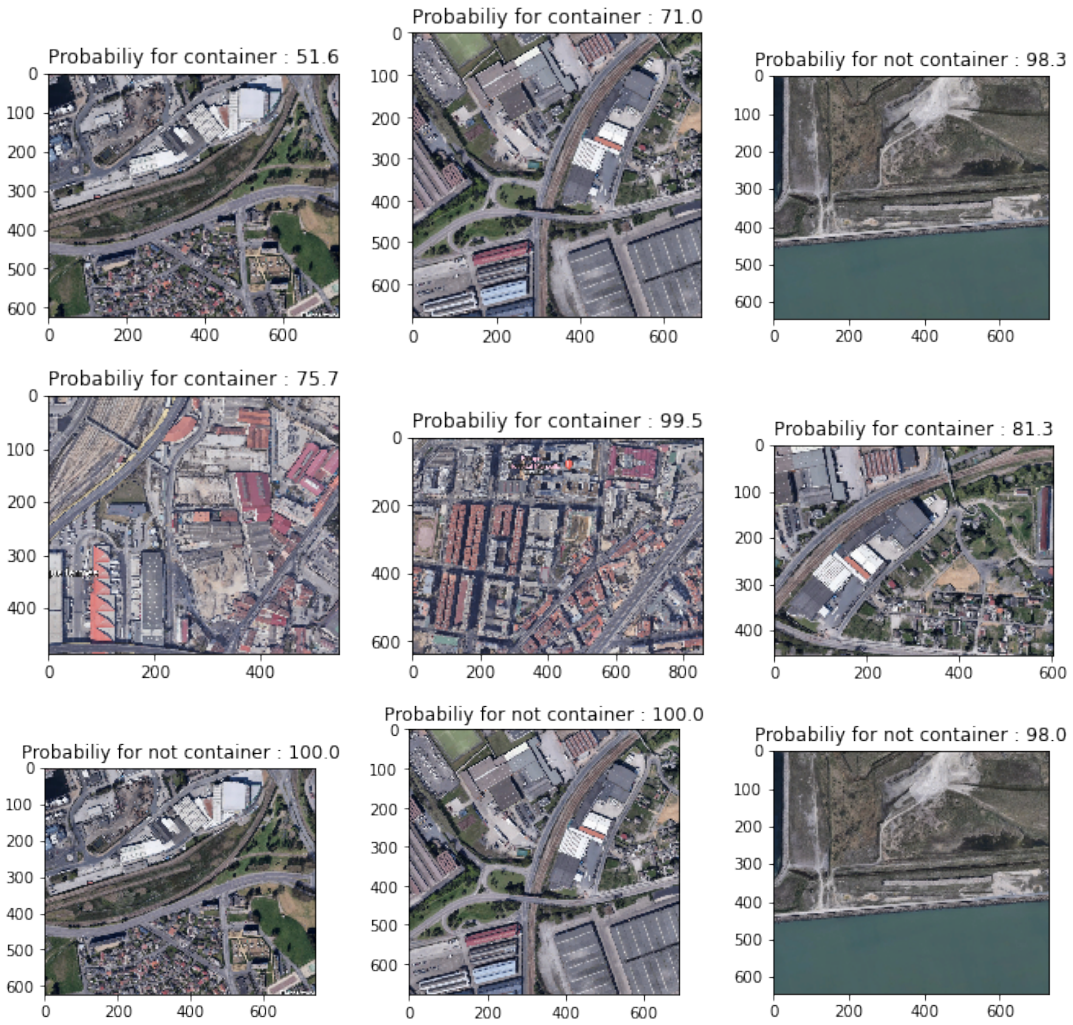


Figure 13: The Confidence of CNN

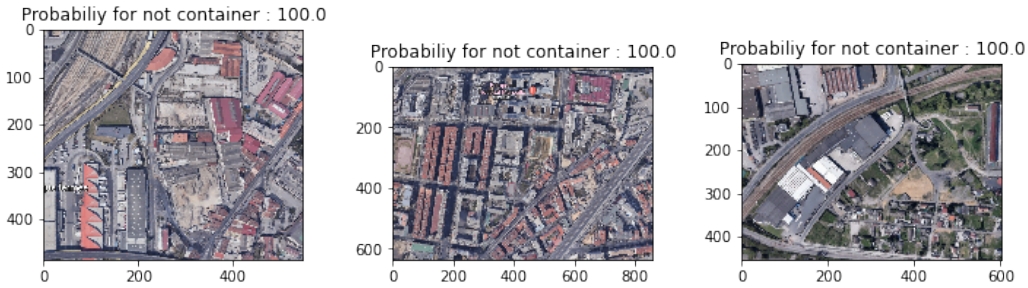


Figure 14: The Confidence of VGG16

11.4 Classification report

Classification report is a function in sklearn, it is used to evaluate the performance of classification-based model and display the text report :the precision, recall, F1 score and support of the model.

The CNN result shows in Fig.15,the VGG16 result shows in Fig.16.

	precision	recall	f1-score	support
0	1.00	0.48	0.65	27
1	0.64	1.00	0.78	25
accuracy			0.73	52
macro avg	0.82	0.74	0.72	52
weighted avg	0.83	0.73	0.71	52

Figure 15: Report of CNN

	precision	recall	f1-score	support
0	1.00	1.00	1.00	27
1	1.00	1.00	1.00	25
accuracy			1.00	52
macro avg	1.00	1.00	1.00	52
weighted avg	1.00	1.00	1.00	52

Figure 16: Report of VGG16

The classification report has two labels (categories), '0' means the 'no_port' samples, '1' means 'port' samples.

The labels on vertical axis:**'accuracy'** refers to the ration of correctly predicted samples to all the predicted samples,**'macro avg'** refers to the average of the corresponding indicator for all categories; **'weighted avg'** refers the sum of the proportion of categorical samples in the total sample and the product of the corresponding index.

The labels on horizontal axis:**'precision'**refers to the proportion of true positive examples in the sample whose prediction result is true; **'recall'**refers to the proportion of the sample that is predicted to be positive in the true positive sample;**'f1-score'** refers to the robustness of the model.

12 Fine-tuning of VGG16

12.1 Epochs and batch_size

1. epoch:one epoch refers to all the samples run through once, but it is necessary to go through the entire dataset numerous times in the same neural network in order to achieve a better outcome. It is important to choose the appropriate number of epochs because as the epochs increases (network runs), the neural network will also update the weights, and the curve changes from under-fitting to over-fitting.

2. batch_size:batch_size is the quantity of samples in one training. When training the data set, it generally does not input all the samples into the model at one time, but train in batches. To choose an appropriate batch_size is to find the best balance between memory efficiency and memory capacity.

12.2 Learning rate

The learning rate(lr) is a hyperparameter that directs how to modify the network weights by controlling the gradient of the loss function. The loss function changes more slowly as the learning rate decreases. Utilizing a low learning rate guarantees that no local minimums are missed, but it also prolongs the convergence process.

12.3 Comparison and Result

To reach the best performance of the model, different learning rates are tested. From the figures below, it is easy to see the different performances of the following models:

- **1.Model1_learning rate**=0.0004
- **2.Model2_learning rate**=0.0005
- **3.Model3_learning rate**=0.0006

1. Accuracy and Loss
   (2)lr = 0.0004
   The model converges at the 14th epoch, the accuracy reaches at around 98% and the loss reaches at around 0.070.

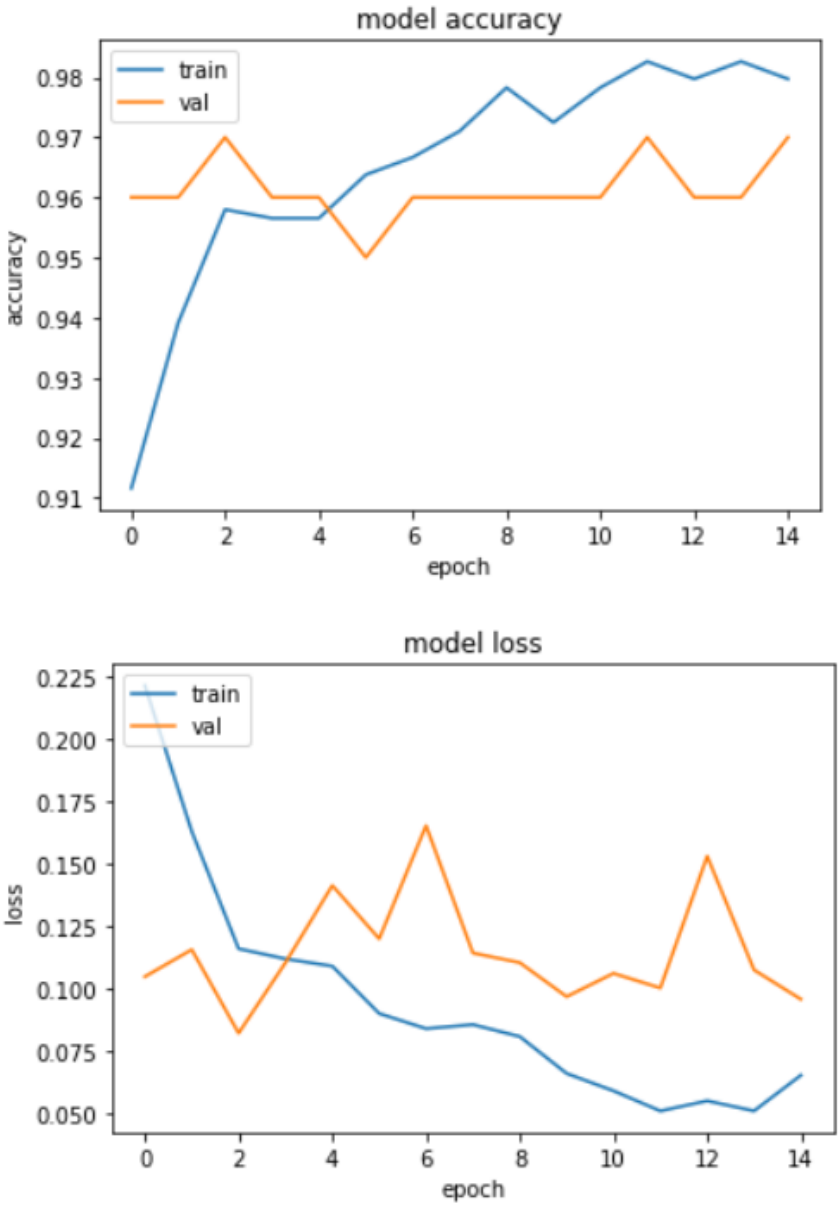


Figure 17: Model Accuracy and Loss at lr = 0.0004

(1)lr = 0.0005

The model converges at the 25th epoch, the accuracy reaches at around 99% and the loss reaches at around 0.035.

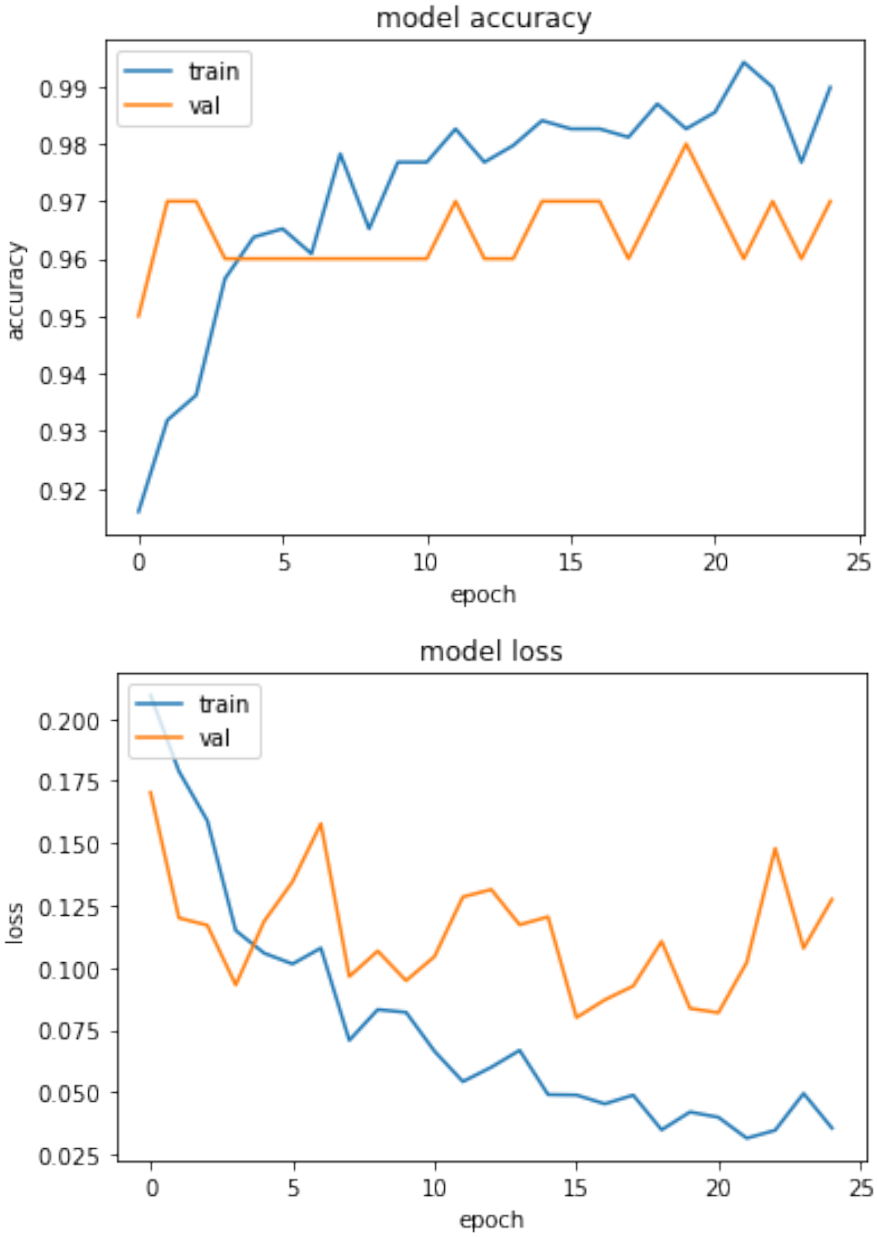


Figure 18: Model Accuracy and Loss of VGG16 at lr=0.0005

(3)lr = 0.0006

The model converges at the 27th epoch, the accuracy reaches at around 100% and the loss reaches at around 0.020.

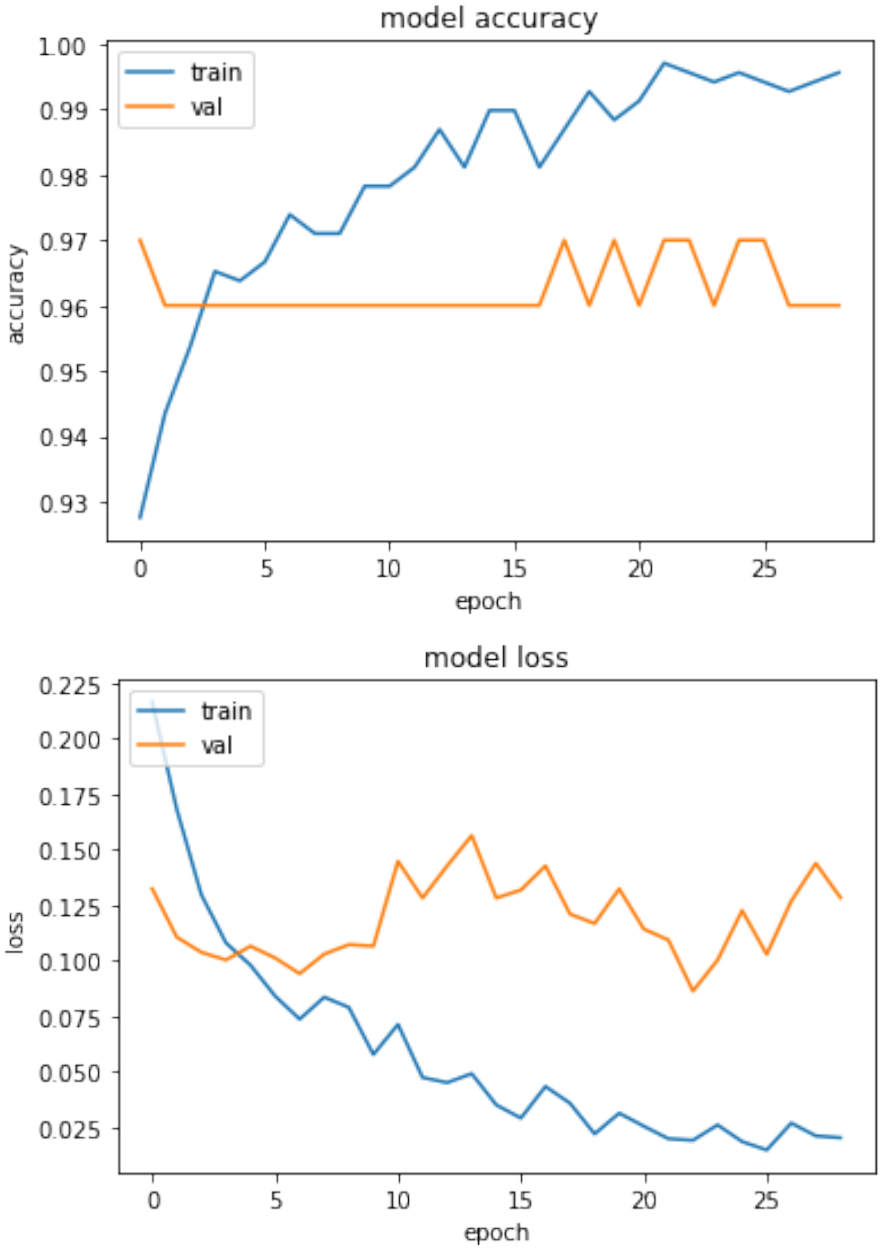


Figure 19: Model Accuracy and Loss at lr = 0.0006

2. Confidence The confidences of the three models are all approach to 100%.Find the full samples of Confidence at Annex 17.3.

From the result above, the best model is at input image size = 150 X 150, epoch = 10, batch_size= 10,learning rate =0.0005.

13 VGG16 with no fixed-orientation picture

The data set used above only has two orientations: parallel or vertical. But in practice, the orientation of pictures are no-fixed, so in this part, given 700 no-fixed orientation pictures collected from Google Earth. The data set used is divided into three parts: training set of 484 pictures, validation set of 180 pictures and test set 36 pictures.

In order to reach the best result, changing the parameters input image size, epoch, batch_size and learning rate to find the most appropriate parameter.

13.1 Comparison

In this part, the comparison of the different performances are illustrated in the following 5 models:

1. **Model1_150**:Input image size = 150 × 150, epoch = 10,batch_size = 30,learning rate = 1e-5
   Find the full result in Annex 17.4
2. **Model2_224**:Input image size = 224 × 224, epoch = 20,batch_size = 20,learning rate = 0.0004
   Find the full result in Annex 17.5
3. **Model3_300**:Input image size = 300 × 300, epoch = 20,batch_size = 20,learning rate = 0.0003
   Find the full result in Annex 17.6
4. **Model4_325**:Input image size = 325 × 325, epoch = 40,batch_size = 20,learning rate = 0.0003
   Find the full result in Annex 17.7
5. **Model5_350**:Input image size = 350 × 350, epoch = 40,batch_size = 20,learning rate = 0.0003
   Find the full result in Annex 17.8

model	Accuracy	Loss	Precision	Recall	T*(sec)
model1_150	92.6%	0.205	0.81	0.81	65.72
model2_224	99.6%	0.018	0.92	0.92	147.44
model3_300	97.9%	0.052	0.93	0.92	277.57
model4_325	99.2%	0.023	0.95	0.94	379.92
model5_350	99.6%	0.009	0.95	0.94	577.14

Where T is the time Prediction for 36 pictures.Under the environment:AMD Ryzen 5 5625U with Radeon Graphics(12GPUs, 2.3GHz)

From the table above ,it is easy to see that the best performance is at the model4_325. The full results are shown below:

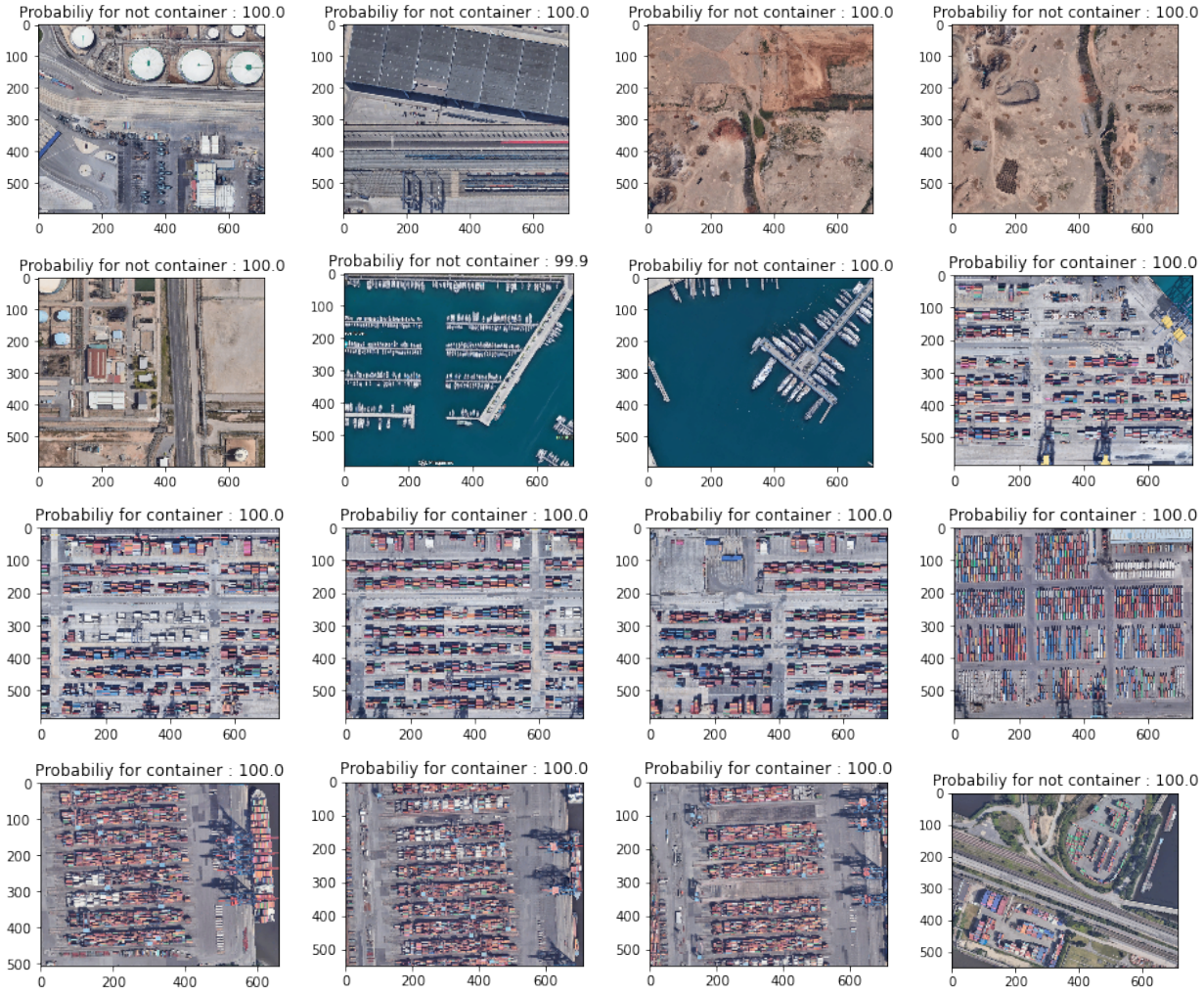


Figure 20: The Confidence of VGG16 at input image size =325X325(1)

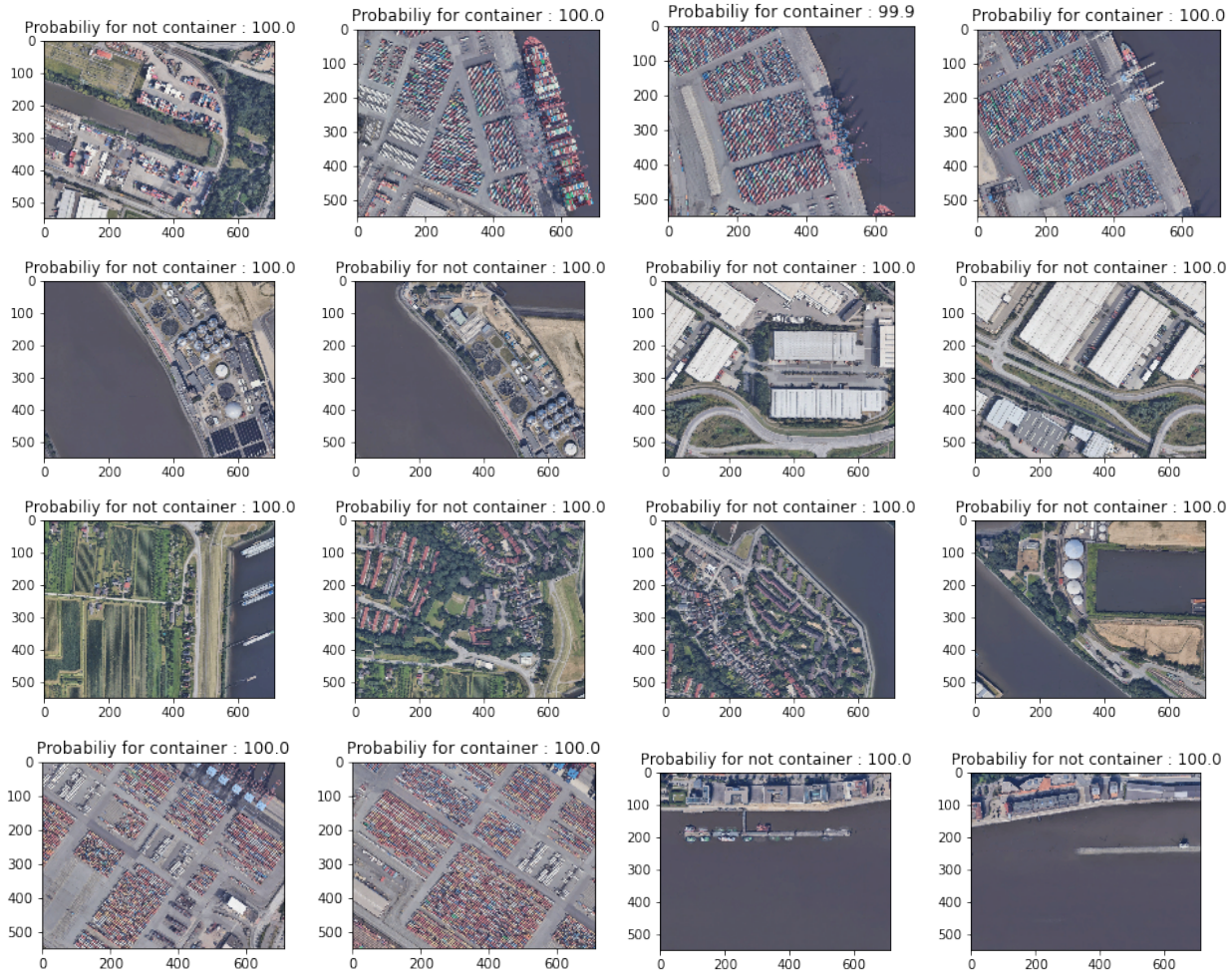


Figure 21: The Confidence of VGG16 at input image size =325×325(2)

13.2 Results of Model4_325

1. Accuracy and Loss
   Under this condition, the model converges at the 25th epoch:
   The train accuracy reaches at 99.2%,train loss reaches at 0.023.
   The validation accuracy reaches at 98.9%, validation loss 0.057.

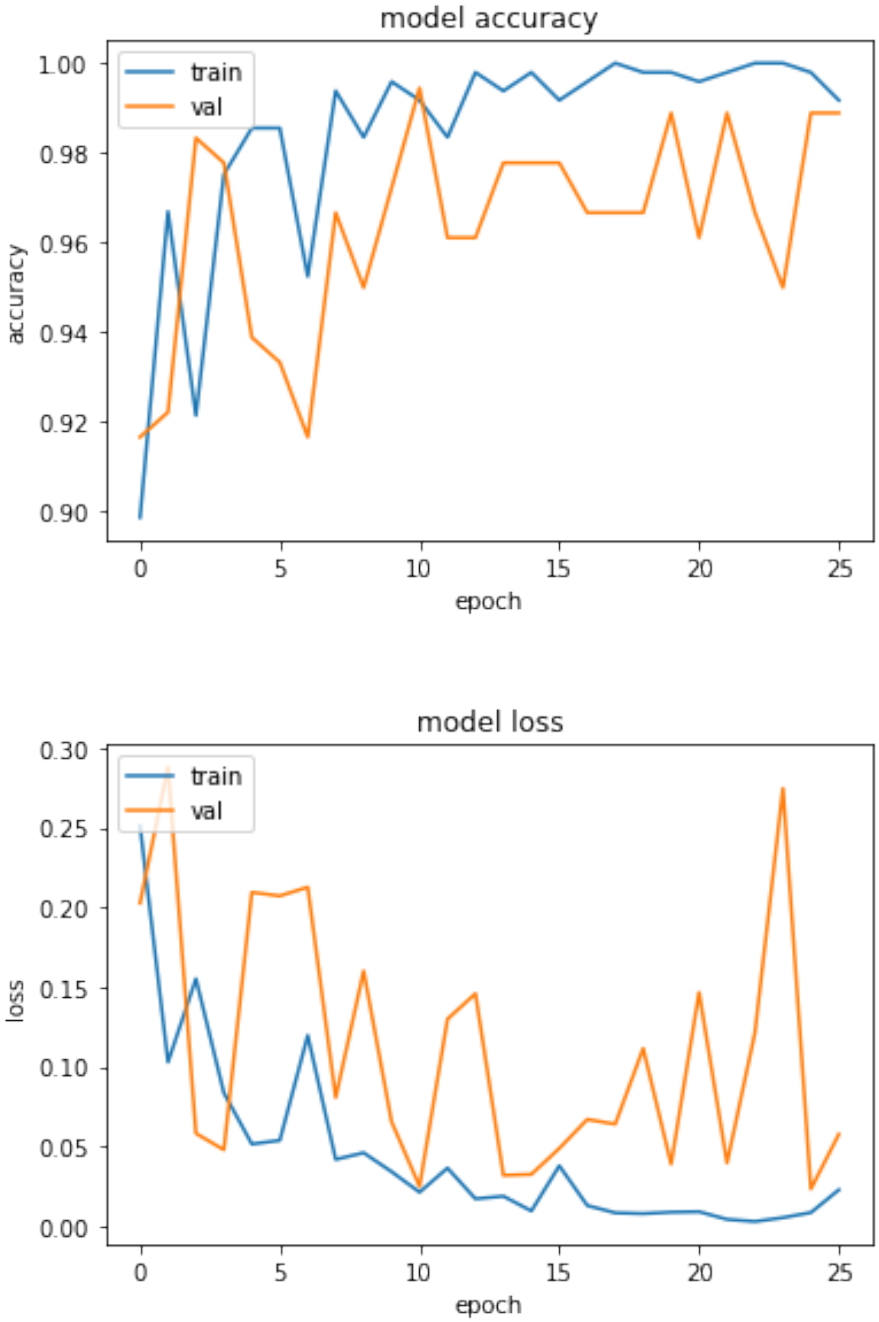


Figure 22: Accuracy and Loss 325

2. Confidence

   The confidence under this condition is at around 80%.

   Find the full results of confidence at **??**.

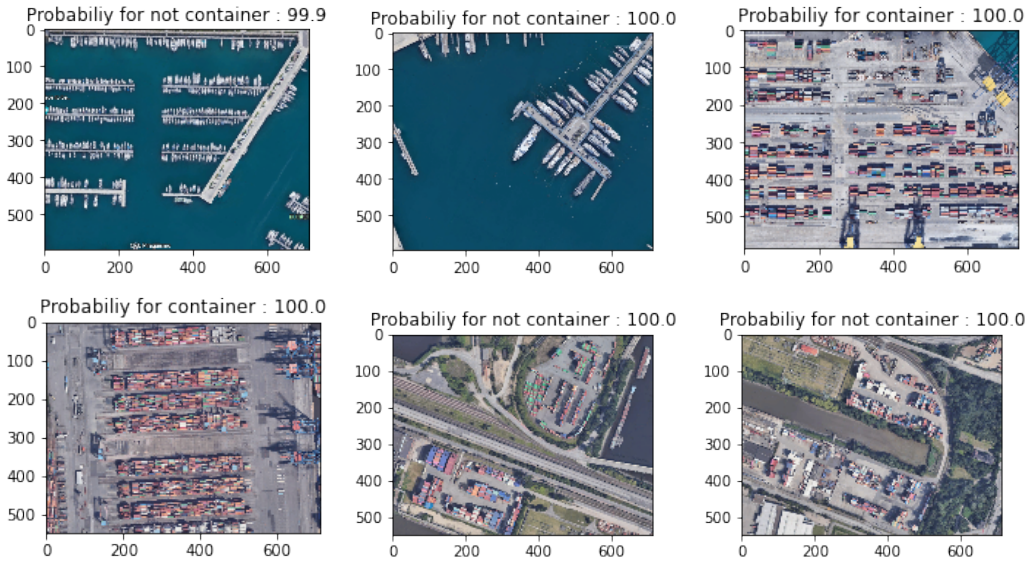


Figure 23: The Confidence of CNN

3. Confusion Matrix

   The accuracy of the model is not 100% correct. The two incorrectly predicted pictures are zone containers far away from the sea. So these two error results don't have a big impact on the model.

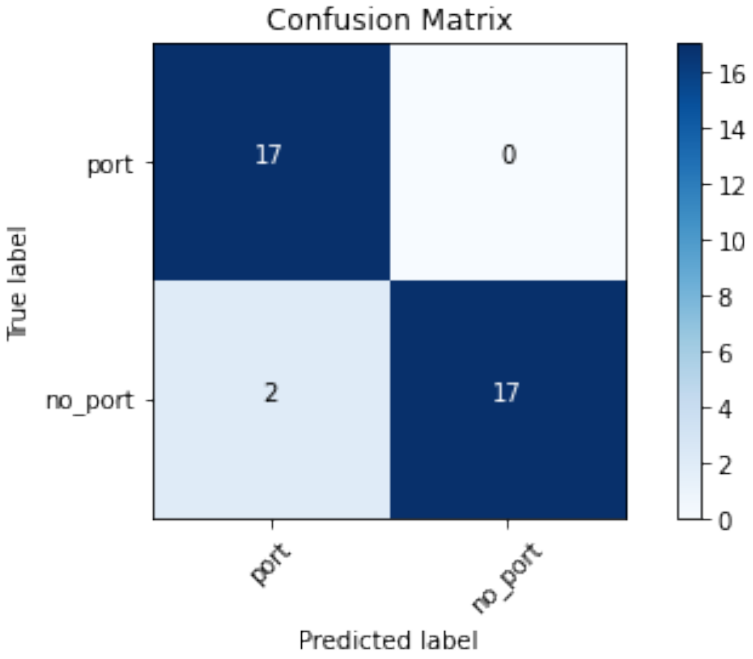


Figure 24: The Confusion Matrix 325

14 Detection of container zones based on Region-proposal CNN

The final goal is to realize the object detection, determine whether the target object exists and determine the position of the target object. Region-Proposal CNN (RCNN) is proposed to solve this problem.

14.1 RCNN

The RCNN is one of the most used object detection algorithms. It mainly solves two problems of object detection: The speed of finding region proposals and the method of feature extraction with different input pictures size.

The process of RCNN mainly contains 3 steps:

1. Extract Region proposal
2. Calculate the features for each region proposals
3. Predict each Region proposals

The expected result is shown in the figure below:

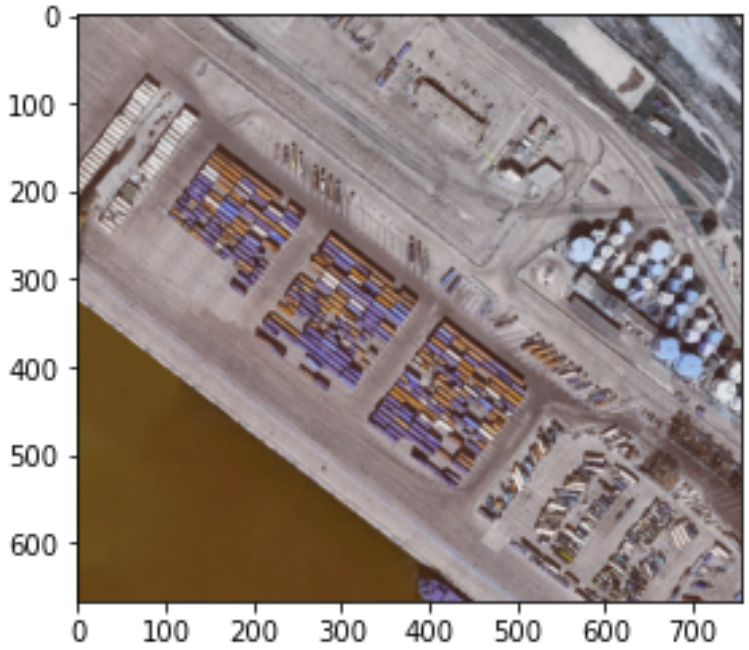


Figure 25: The target picture

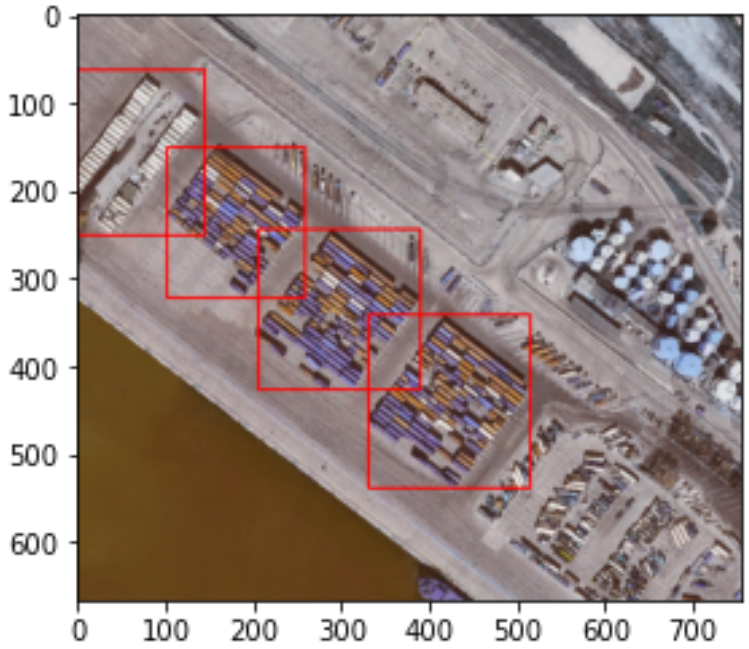


Figure 26: The Boxes

14.2 Extract Region Proposal

The essence of the fully connected layer is the linear transformation from one feature space to another, in that case, the CNN is not able to handle many items and find the region proposals. Theoretically, the method of Exhaustive search can be applied to solve this problem: using a sliding window to perform the brute force search and to choose a region area, then, applying the CNN model to that region can be used. However, this technique has a disadvantage, which is that the same item might be represented in images of various sizes. There are lots of region proposals while taking the considerations into account, and if CNN is applied to all of those areas, it would cost a lot of computation cost.

So in order to reduce the time and lower the computation power, firstly, the selective search method is applied to select about 2000 region proposals on each image, the region proposal is the location where the object is likely to appear.

The selective search uses both Exhaustive search and segmentation, which is a method to separate objects of different shapes in the image by assigning them different colors, in order to boost the speed.

The region proposals have both positive and negative samples. If the IOU of a region proposal and the one with the largest overlapping area of all ground truths on the current image is greater than or equal to 0.65, the region proposal is taken as a positive sample based on this case, otherwise, it will be taken as a negative sample.

The step of Selective Search contains 3 steps as following:

- From a set of regions, choose two that are most likely to contain expected objects.
- Combine them into a single, larger region.
- Repeat the above steps for multiple iterations.

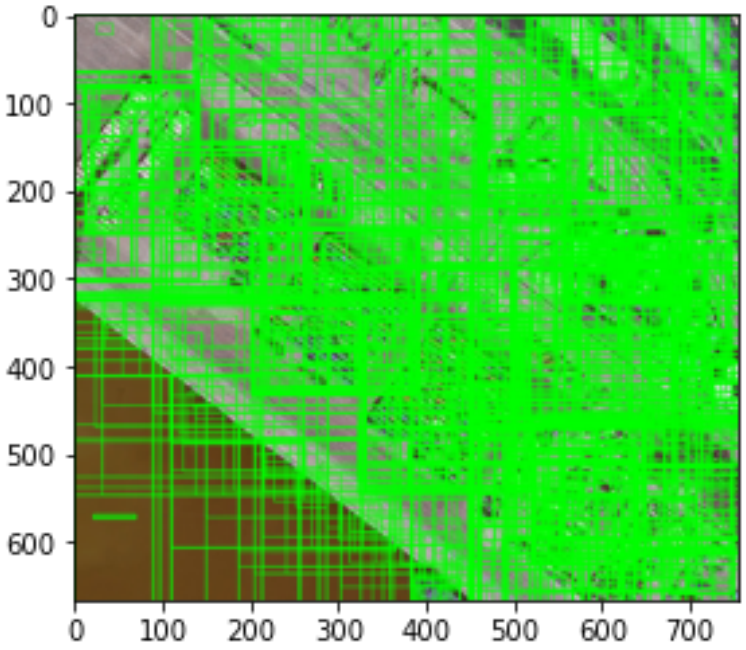


Figure 27: Return of selective search

14.3 Calculate the features for each region proposals

After getting region proposals, the second step is to input them into a CNN network, input each region proposal once, and return a feature vector corresponding to the region proposal.

Because the size of region proposals are different from one to another. The method of converting all input images into a uniform size output is called **warping**. This process can be realized based on OpenCV.

Image wrapping is the method to resize all region proposal boxes to a size of 224×224.

The process of wrapping contains the following steps:

1. Take an area around the target of the original area and scale it equally to the image size required by CNN.
2. Remove the original target area and then fill the target area, and then scale to the image size required by CNN.
3. Directly scale the original target area non-equip metrically to the image size required by CNN.

In this part, The network used is VGG16, import the VGG16 model and put the ImageNet weights into the model. The first 15 layers of VGG16 are frozen, and then a 2-unit softmax dense layer is added.

14.4 Predicted each Region proposal

In this part, after obtaining the feature vectors of each region proposal. The model needs to receive the images that passed from the selective search, to realize this process, the function model_final.predict(img) is used to pass all results of selective search to the model as input.

If the output of the model indicates that the confidence of the box of images is above the defined threshold, a bounding box is created on the original image at the coordinates of the proposed region.

pass the test image to selective search, then pass the top 2000 suggested regions in the trained model and predict the categories of those region proposals.

14.5 Results

The dataset used contains 152 no-fixed orientation pictures and the coordinates of the object area that corresponds to each image, collected from Google Earth at the height of from 550 m to 610 m.

In order to reach the best result, change the parameters input image size, epoch, batch_size, and learning rate to find the most appropriate parameter.

In this part, the comparison of the different performances are illustrated in the following 4 models:

1. **Model1_10**: The epochs =20, the step_per_epoch = 10, the validation_step = 2;
2. **Model2_50**: The epochs =20 and the step_per_epoch = 50, the validation_step = 5;
3. **Model3_64**: The epochs =20 and the step_per_epoch = 64, the validation_step = 10;
4. **Model4_128**:The epochs =20 and the step_per_epoch = 128,the validation_step = 32;

After comparing the 4 models, the best result is at the step_per_epoch = 128 and validation_step =32.The full result can be seen at the 17.9.

The results of Model4_128 shown at the Fig.28 and 29.

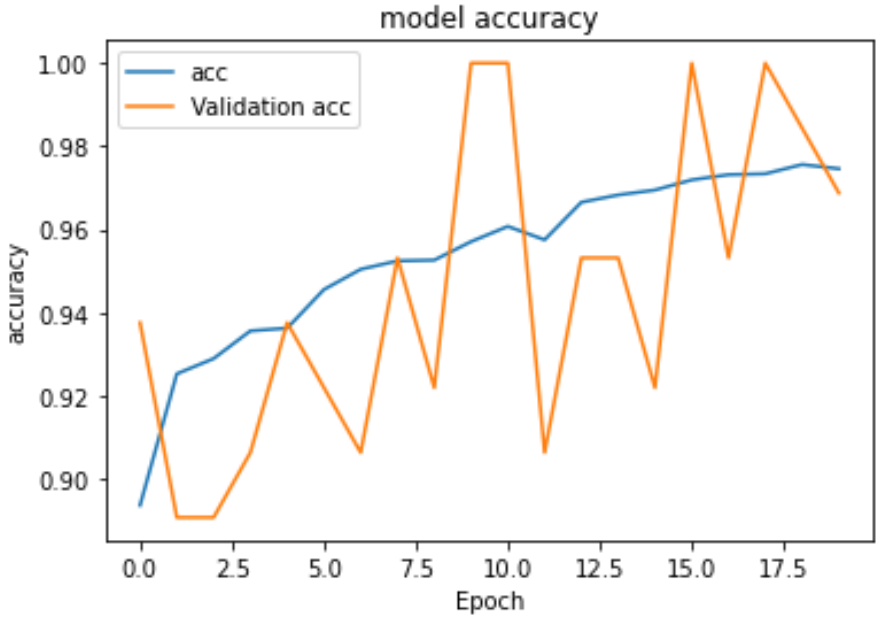


Figure 28: RCNN accuracy

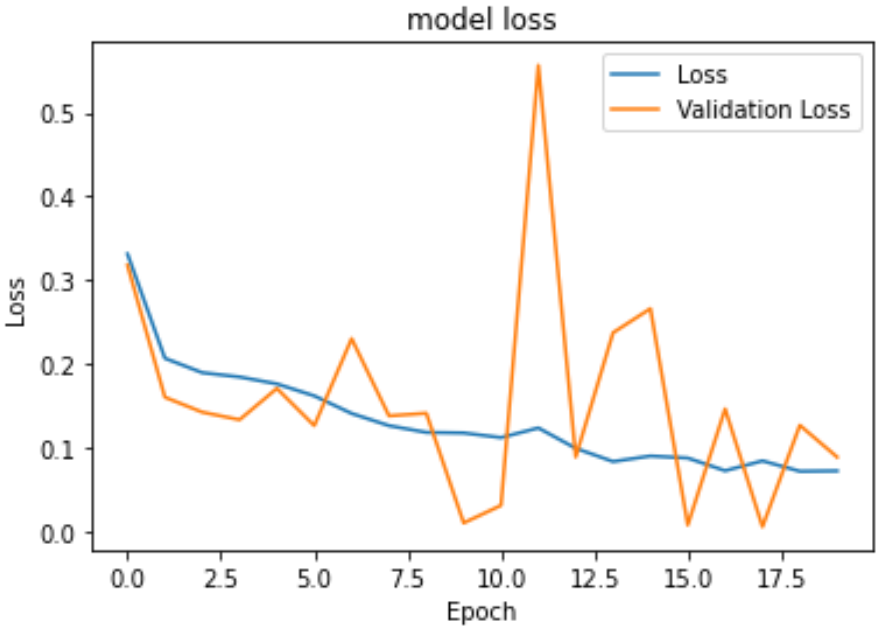


Figure 29: RCNN loss

Under the condition of the best performance out of 4, the outcome results are not the best, so in order to reach the best outcome, it is necessary to change the different threshold of IOU under the same condition.

First, it is set to the threshold at 0.6.
The results at the threshold of 0.65 shown in Fig.**??**

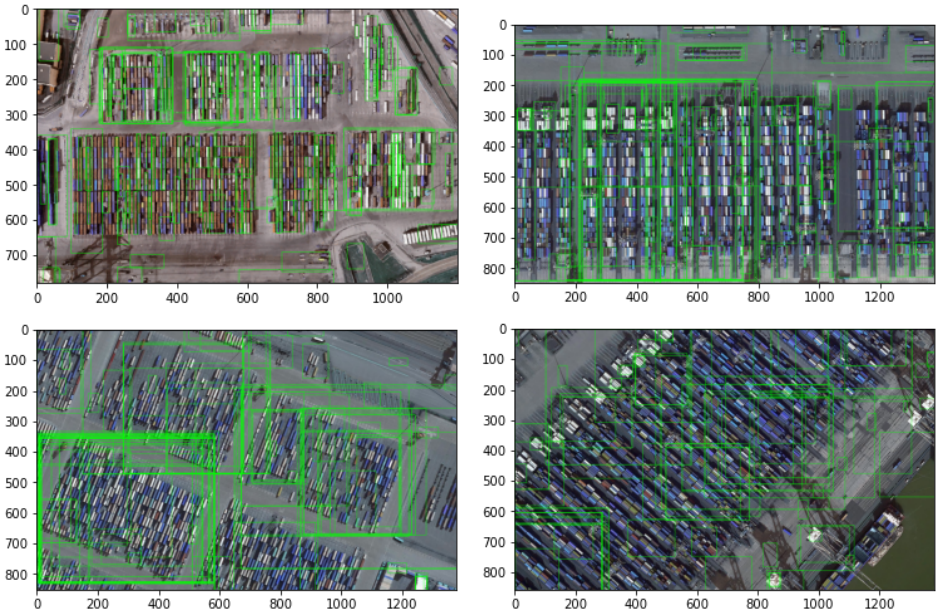


Figure 30: The result at the threshold at 0.6

The results at the threshold of 0.70 shown in Fig.31

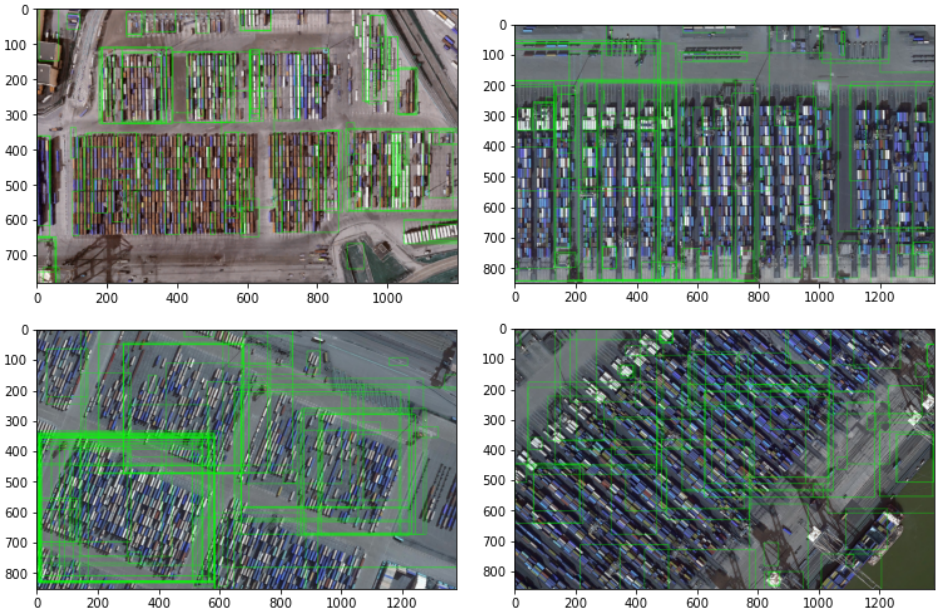


Figure 31: The result at the threshold at 0.70

The results at the threshold of 0.80 shown in Fig.32

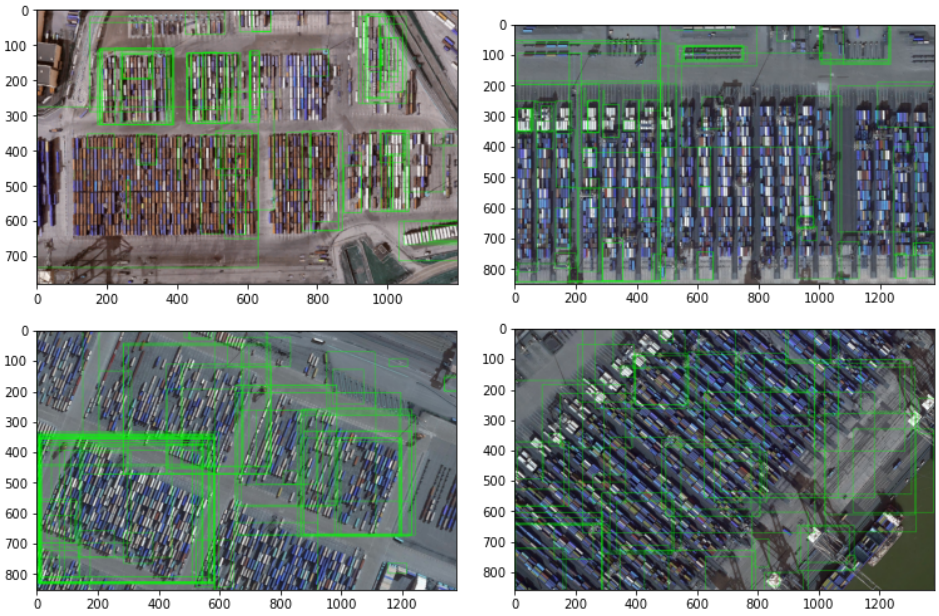


Figure 32: The result at the threshold at 0.80

From the pictures above, the best performance of this RCNN model is at the threshold of 0.7.It performs better when the zone-container images are complete and horizontal.

The full results can be seen at the 17.10

15 U-Net Image Segmentation

In the previous models, it only detected the approximate location of the target or simply determined whether there is a target image or not.

U-net Image segmentation can get a more precise image position because it will classify each pixel with a category (label) and make the precise segmentation.

Figure 33: The image segmentation result of Algeciras port

There are two types of image segmentation:

1. **Semantic segmentation**: it will classify each pixel with a label.
2. **Instance Segmentation**: it will classify and categorize each pixel into different instances of the label.

Figure 34: image segmentation real image

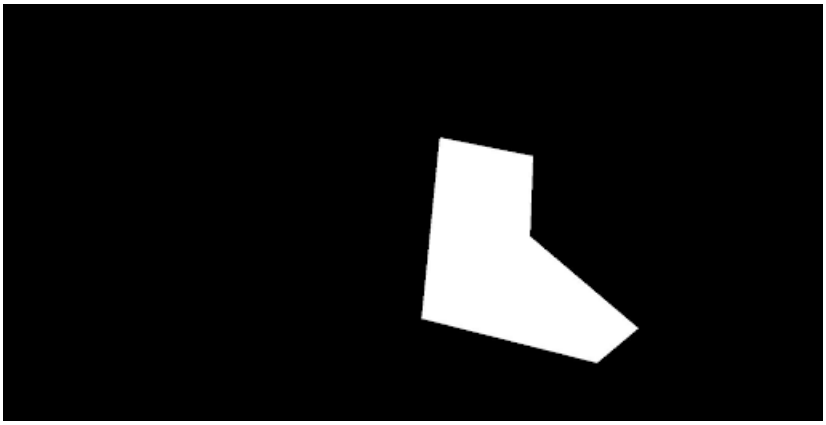


Figure 35: image segmentation mask image

U-net segmentation is a one of semantic segmentation. In this part, each image includes one or more corresponding labels and masks. From the picture above, it is easy to see that for each pixel, there are two types of masks:

1. pixel belongs to the container
2. pixel does not belong to the container

15.1 Data Preprocessing

The dataset is collected from Google Earth from Marseille.
The data preprocessing process contains the following steps:

1. Normalization The pictures collected are of different sizes, so the first step is to transform all pictures into the same size of x by using the openCV library *resize()*.

2. Data augmentation The number of pictures collected is not much. In that case, randomly applying the function *flip* can help augment the number of pictures.

15.2 U-net

The model architecture is like a structure of Encoder-Decoder.

UNet, which evolved from the traditional convolutional neural network, was first designed and applied in 2015 to process biomedical images. In general, a convolutional neural network focuses its task on image classification, where the input is an image and the output is one label. U-net is able to localize and distinguish borders by doing classification on every pixel, so the input and output share the same size.

From the picture above, it is easy to see that the U-net segmentation model is like an Encoder-Decoder architecture. In the picture above, the input of the image is a 572 X 572 image.

1. Contracting path: the main purpose is to extract features through convolutional layers.

2. Expanding path:the main purpose is to upsample the feature map in order to make pixel-level predictions on the image.

The left part of the model is the contracting path, it is a series of downsampling operations consisting of convolution and Max Pooling. The Contracting path consists of 4 blocks. Each block uses 2 convolutional layers and 1 maxpooling layer to perform downsampling, after each downsampling, the number of feature maps doubled. Therefore, there are finally 32×32 feature maps.

The right part of the model is the expanding path, it also consists of 4 blocks. Before each block, the size of the feature map doubled through the operation of deconvolution, then cut the number of the feature map in half, and merged with the feature map on the left side contracting path. Since the size of the two feature maps are different, so U-net will normalize the feature maps by clipping the feature map of the contracting path to the same size as the feature map in the expanding path.

Deconvolution

From above, it can be seen that convolution is the process of contracting the features of pictures through the filters.

The process of deconvolution is to get the picture with the input features.

15.3 Model architecture

The U-net model architecture shows in the following pictures:

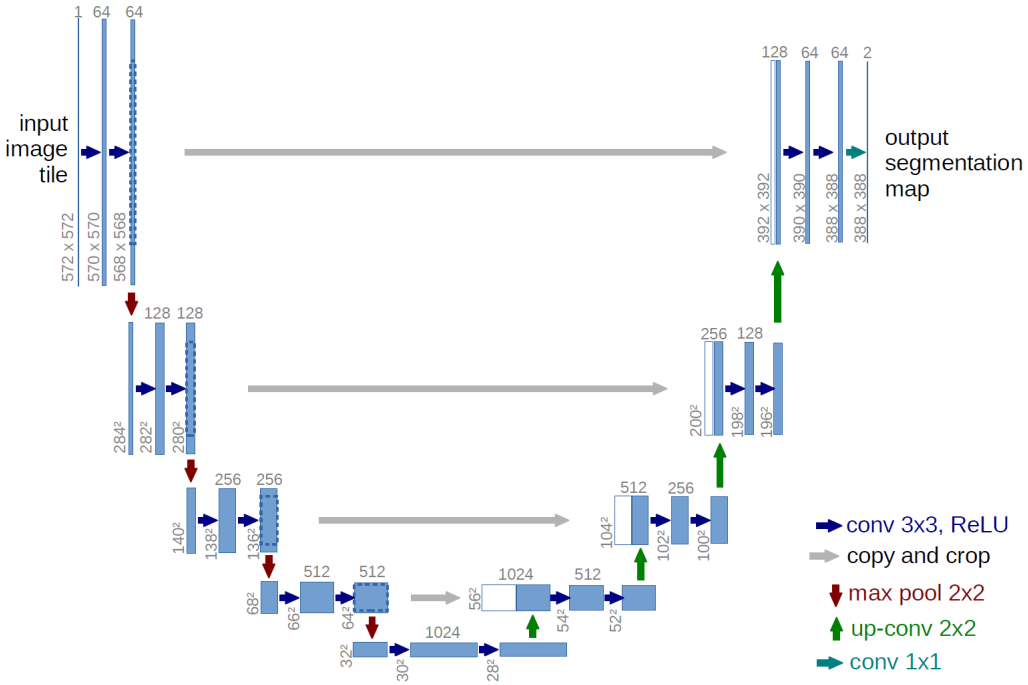


Figure 36: U-net segmentation architecture

15.4 Result and Test

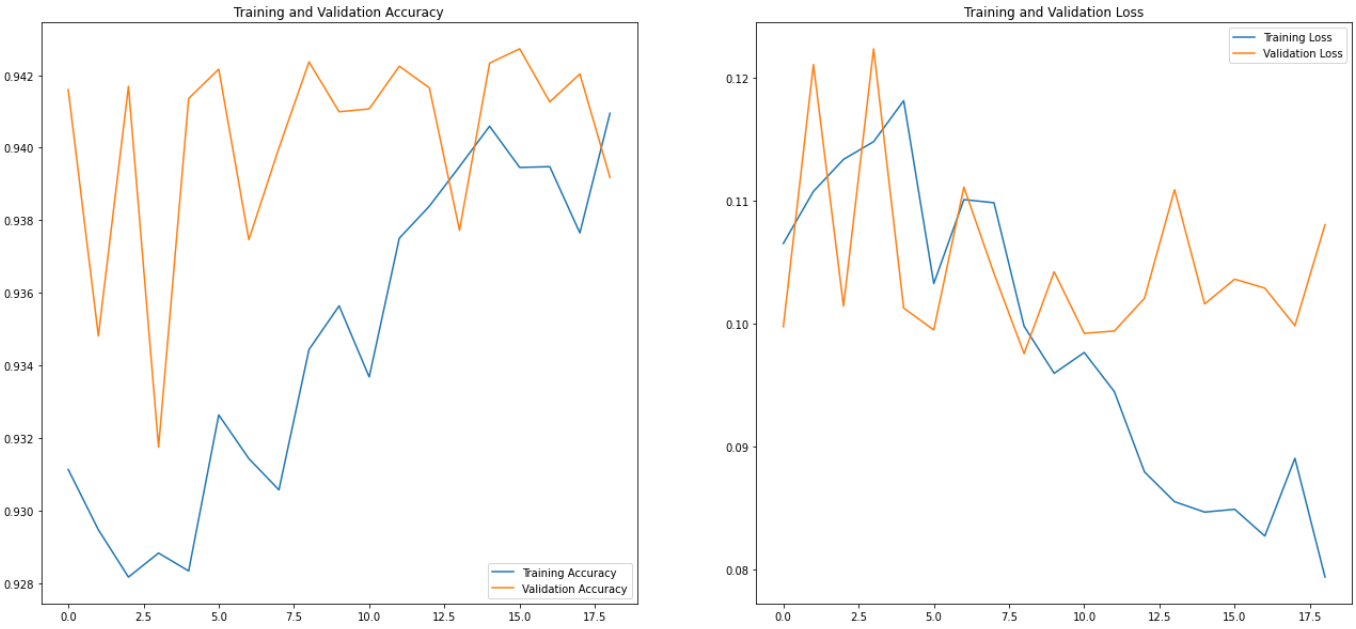


Figure 37: U-net result

As the figure 37, the training and validation pictures above show that the result is not perfect. The Accuracy of the training set can reach around 94%, and the loss of the training set can reach around 0.08. But for the validation set, the performance is not good, it has many vibrations, and the improvement of accuracy and the decrease of loss is not significant, which is shown in the pictures 38, 39, 40 and 41 below:

Figure 38: The result in Algeciras port

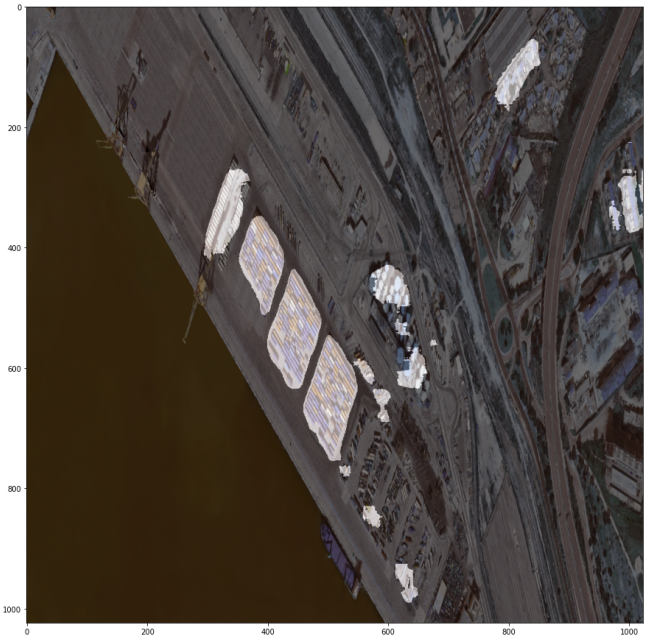


Figure 39: The result in Marseille port

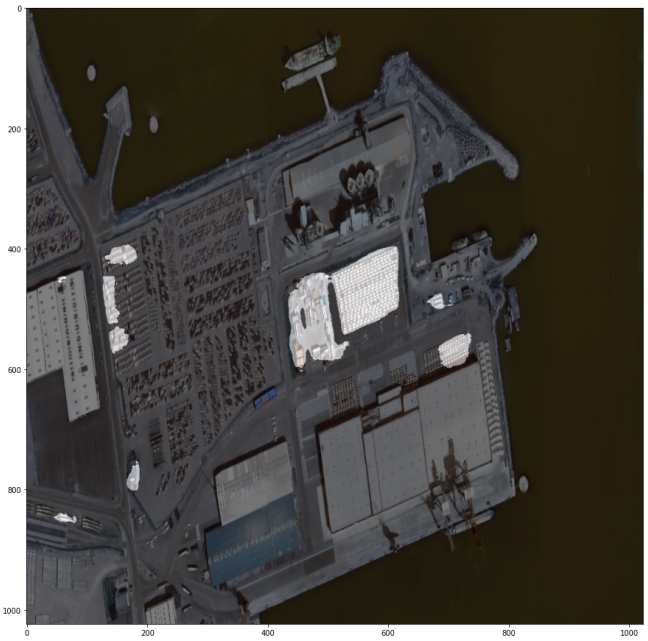


Figure 40: The result in Sète port

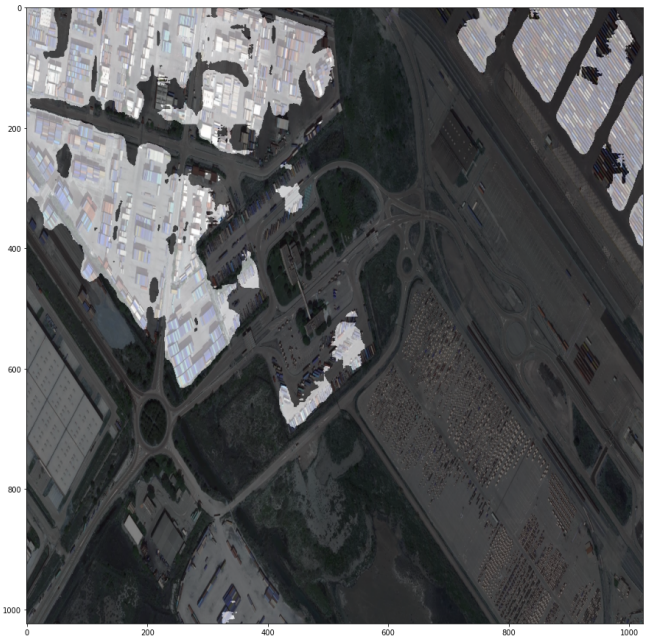


Figure 41: The result in Fos Marseille port

16 Conclusion

16.1 General assume of the project

In this paper, the project is mainly focused on the detection of the container zone because there should be a track of the movement of the containers. It concentrates on 5 methods to perform the detection: TemplateMatching, simple CNN, VGG16, RCNN, and U-net Segmentation.

These methods gradually improve the accuracy of the detection, from searching for the comparing template, and determination of whether it exists or not to localization and classification at the pixel level, thus getting a more precise result and tracking the movement of containers.

In this project also included the collection and preprocessing of images from different ports around the world of different sizes and colors values.

In summary, this paper proposed a method for examining the positions and analyzing the situation of the containers.

Following is the comparison of the different models:

Comparison \ Model	Classification	RCNN	U-net Segmentation
Advantages	High accuracy; Short time-consuming	High accuracy; Recapture large area	High accuracy; Each pixel is segmented; Recapture large area
Disadvantages	Only get a result on a small particular zone	Long time-consuming; Difficult to controll window choices	Too detailed; Possibly get wrong predicted pixels
Results	Region-level result	Overall result	pixel-level result

The three different models have their own advantages and disadvantages. The U-net segmentation has high accuracy, but it is not as high as the result of the classification, because the result of the classification is the overall result and the U-net is the pixel-level result.

16.2 Experience of the science research

Firstly, in the process of search process finding, the goal of the paper is to find a method to detect the container zones under different situations. The goal is set clearly. Then for the realization part, the data collection is the more time-consuming part. The image collected is stored and gets on AWS.

Because the dataset used is different in different models, the preprocessing process is also different. The additional information, for example, the location information of the container in the data format *.json.* on the pictures also need to be processed.

Besides, I have participated in the treatment of the image process from scratches, which lets me know that it is long and takes much effort. Especially, when I collect data, the images should be collected meticulously from height to size. Moreover, I also learn how to use one item to annotate images called SuperAnnotae. Finally, the whole project will be piped on Github.

16.3 Experience in the professional environment

During the internship, the professional environment is good at Bluepulse. The physique surrounding and the sympathetic relationships with other employees provide a great atmosphere for improving the experience.

The most important thing I learned is teamwork: Data Science cannot stand alone. I have met the Data Engineering team and CEO who gives me an overview of an AI company. It is not only Data Science research but also the crucial Data Engineering operation architecture and the media promotion of the marketing team.

16.4 Last words

In summary, after this half-year internship, based on these conclusions, work process and environment, I have gained a lot of experience both in life and work, I have a more detailed and profound understanding of my major and, a clearer thinking and understanding of the future direction, and I have realized many things that cannot be learned only in school, no longer limited to the classroom, but have a more professional and comprehensive understanding.

References

[1] Beatrice-Laurent. Image classification. https://github.com/wikistat/High-Dimensional-Deep-Learning/blob/master/ImageClassification/CatsVSDogs.ipynb, 2019.

[2] Ayush Gupta. A comprehensive guide on deep learning optimizers. https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/.

[3] nnart. Why is the vgg network commonly used? https://nnart.org/why-is-vgg-commonly-used/.

[4] openCVdoc. Template matching. https://docs.opencv.org/3.4/de/da9/tutorial_template_matching.html.

[5] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[6] Jean Rovani. End-to-end object detection with template matching using python. https://www.sicara.fr/blog-technique/object-detection-template-matching.

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

17 Annex

17.1 Result Template Matching

1. Container

   The results of containers confidence at the place of Havre, Dunkerque and Marseille.

Figure 42: Havre Container 01

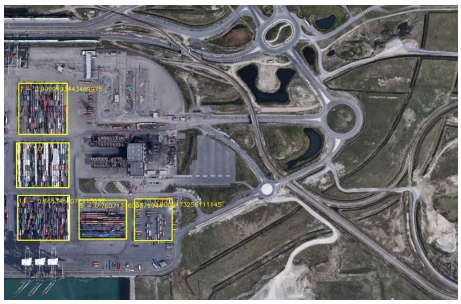


Figure 43: Havre Container 02

Figure 44: Havre Container 03

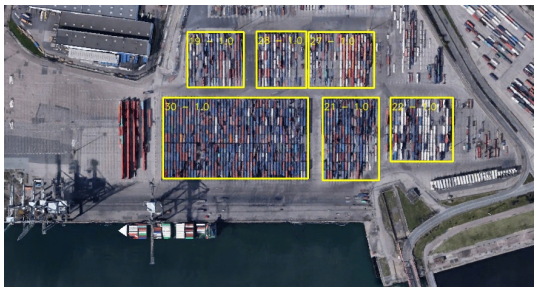


Figure 45: Havre Container 04

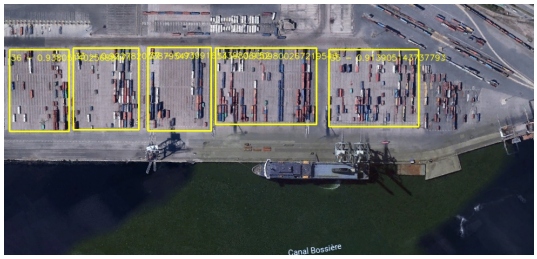


Figure 46: Havre Container 05

Figure 47: Dunkerque Container 01

Figure 48: Marseille Container 01

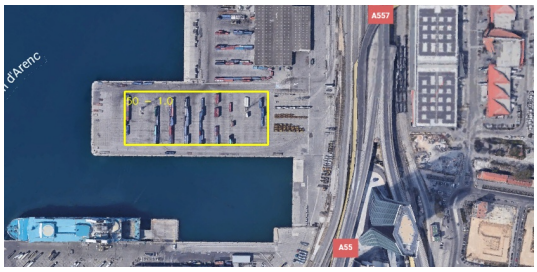


Figure 49: Marseille Container 02

Figure 50: Marseille Container 03

Figure 51: Marseille Container 04

Figure 52: Marseille Container 05

2. Ship

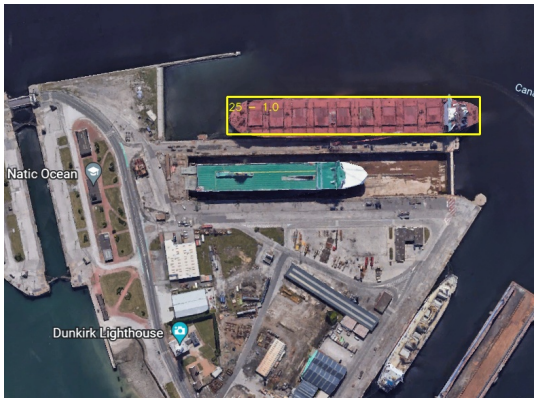


Figure 53: Dunkerque ship 01

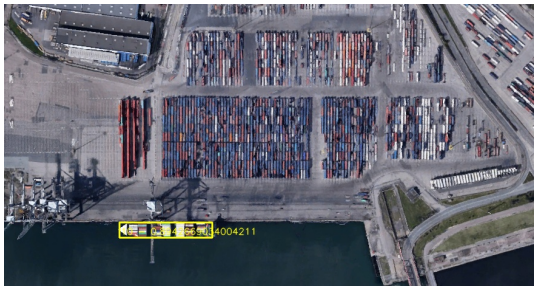


Figure 54: Havre ship 01

Figure 55: Havre ship 02

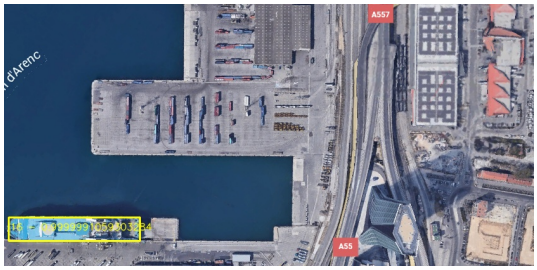


Figure 56: Marserille ship 01

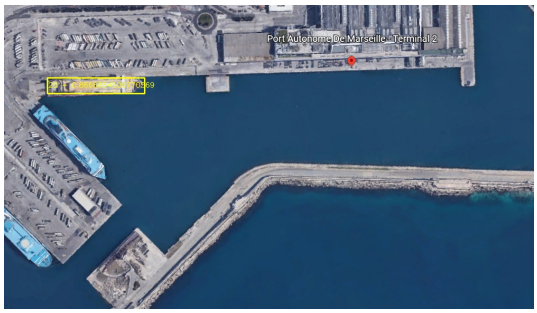


Figure 57: Marserille ship 02

Figure 58: Marserille ship 03

Figure 59: Marserille ship 04

3. Terminal

Figure 60: Havre Terminal 01

Figure 61: Havre Terminal 02

Figure 62: Dunkerque Terminal 03

Figure 63: Marseille Terminal 01

Figure 64: Marseille Terminal 02

4. Others(do not have template)

Figure 65: not port 01

Figure 66: not port 02

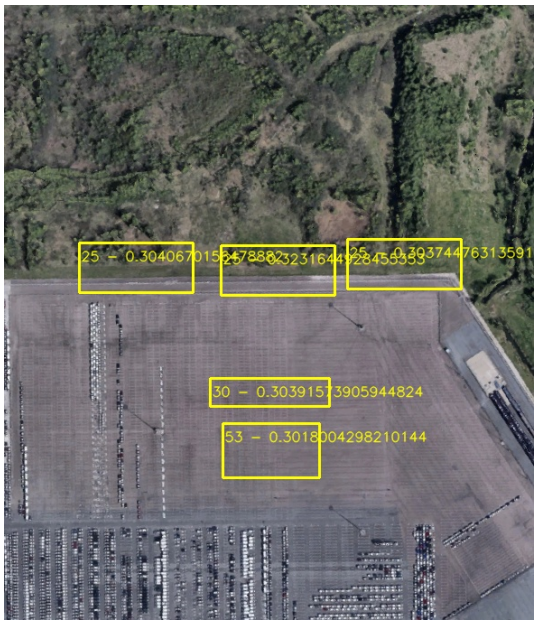


Figure 67: not port 03

17.2 Result of Classification

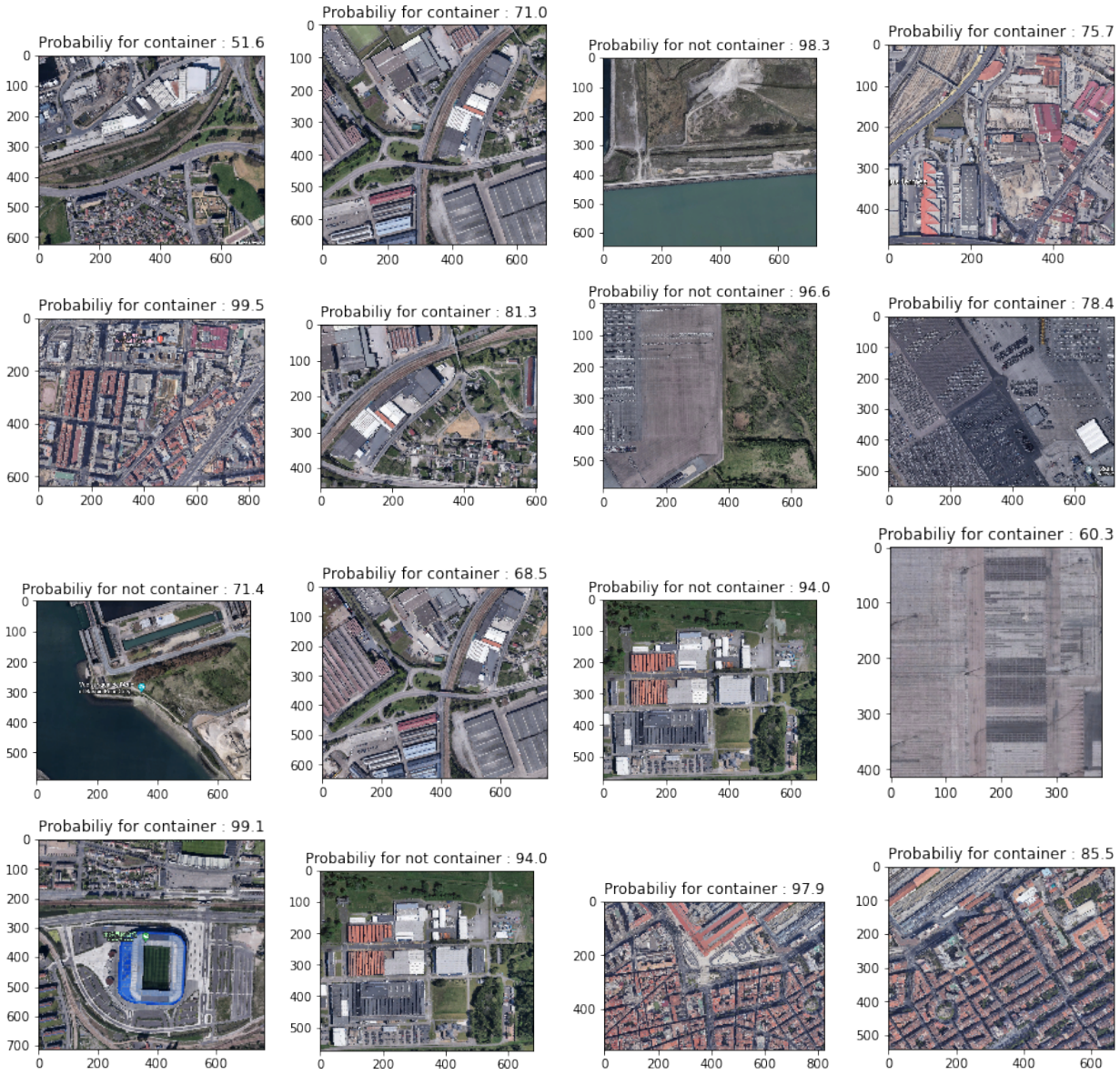


Figure 68: The Confidence of CNN(1)

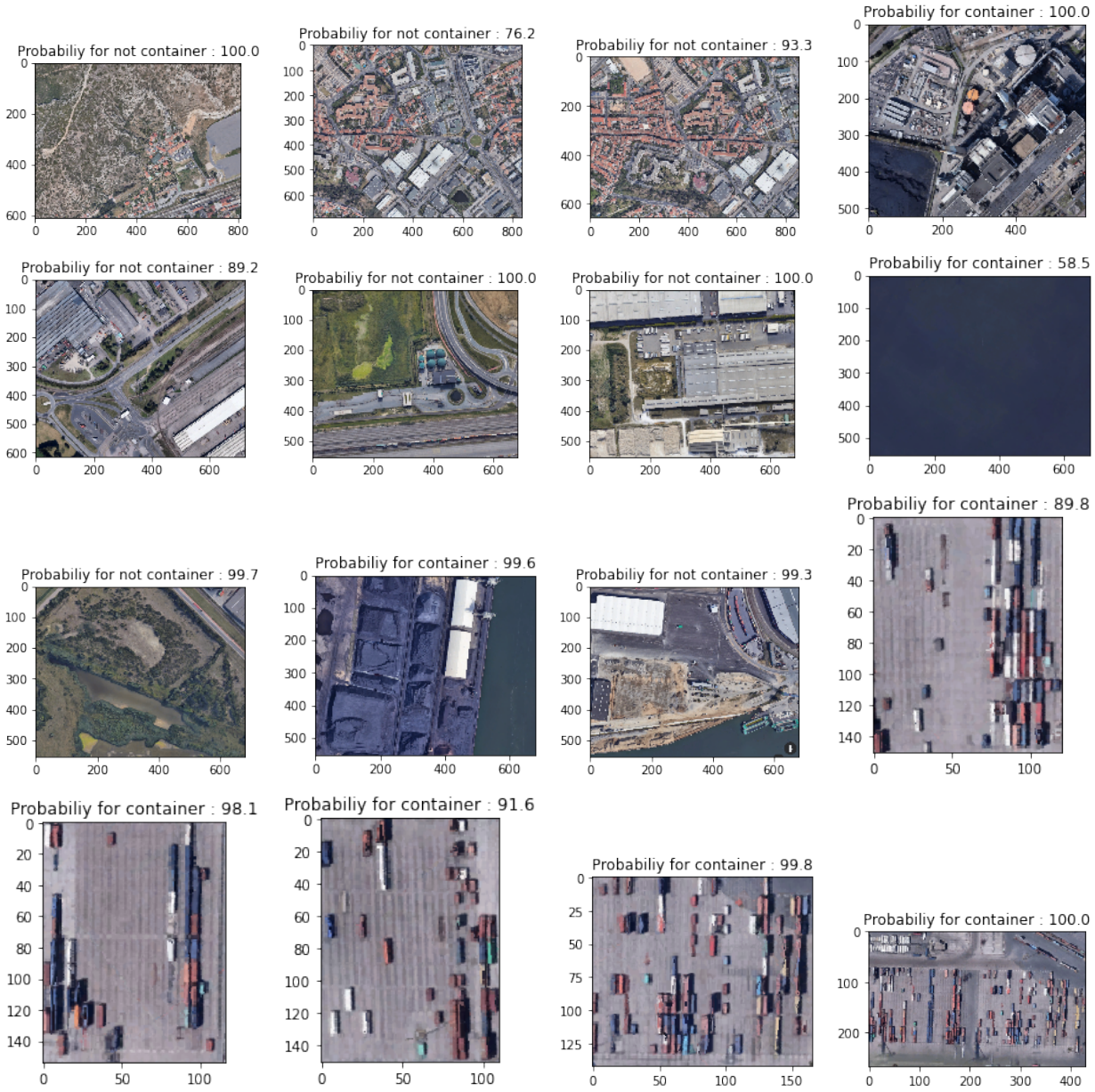


Figure 69: The Confidence of CNN(2)

Figure 70: The Confidence of CNN(3)

Figure 71: The Confidence of VGG16(1)

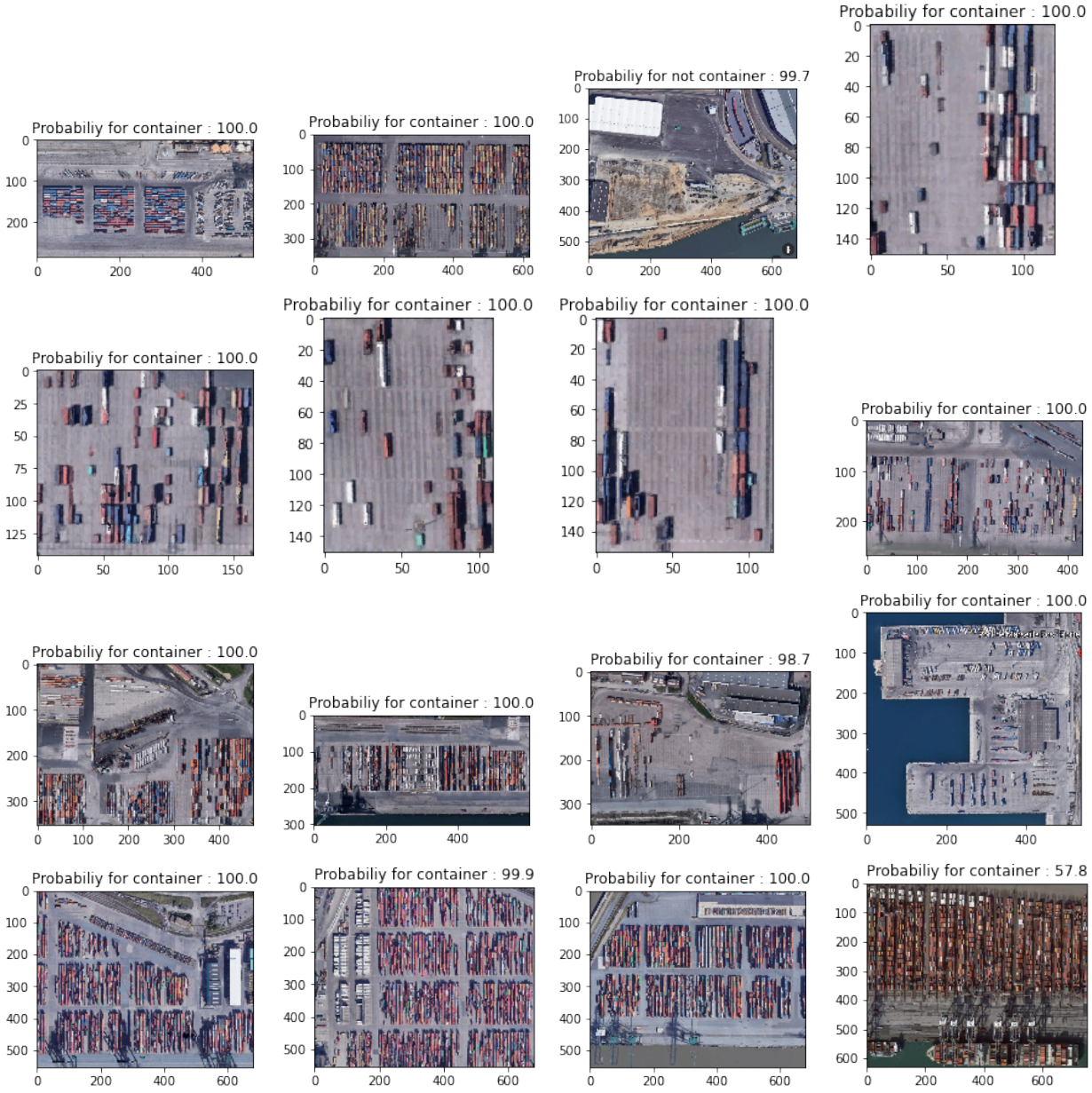


Figure 72: The Confidence of VGG16(2)

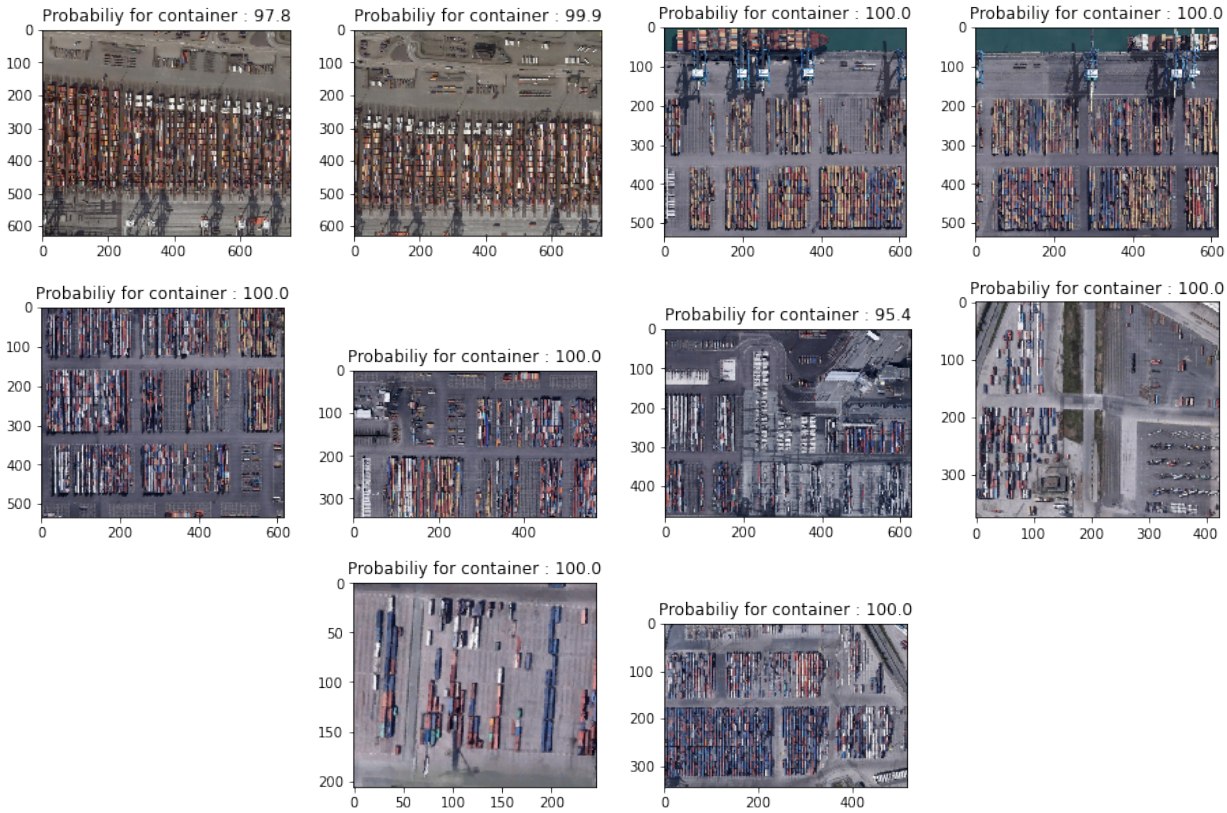


Figure 73: The Confidence of VGG16(3)

17.3 Result of VGG16

1.learning rate =0.0004

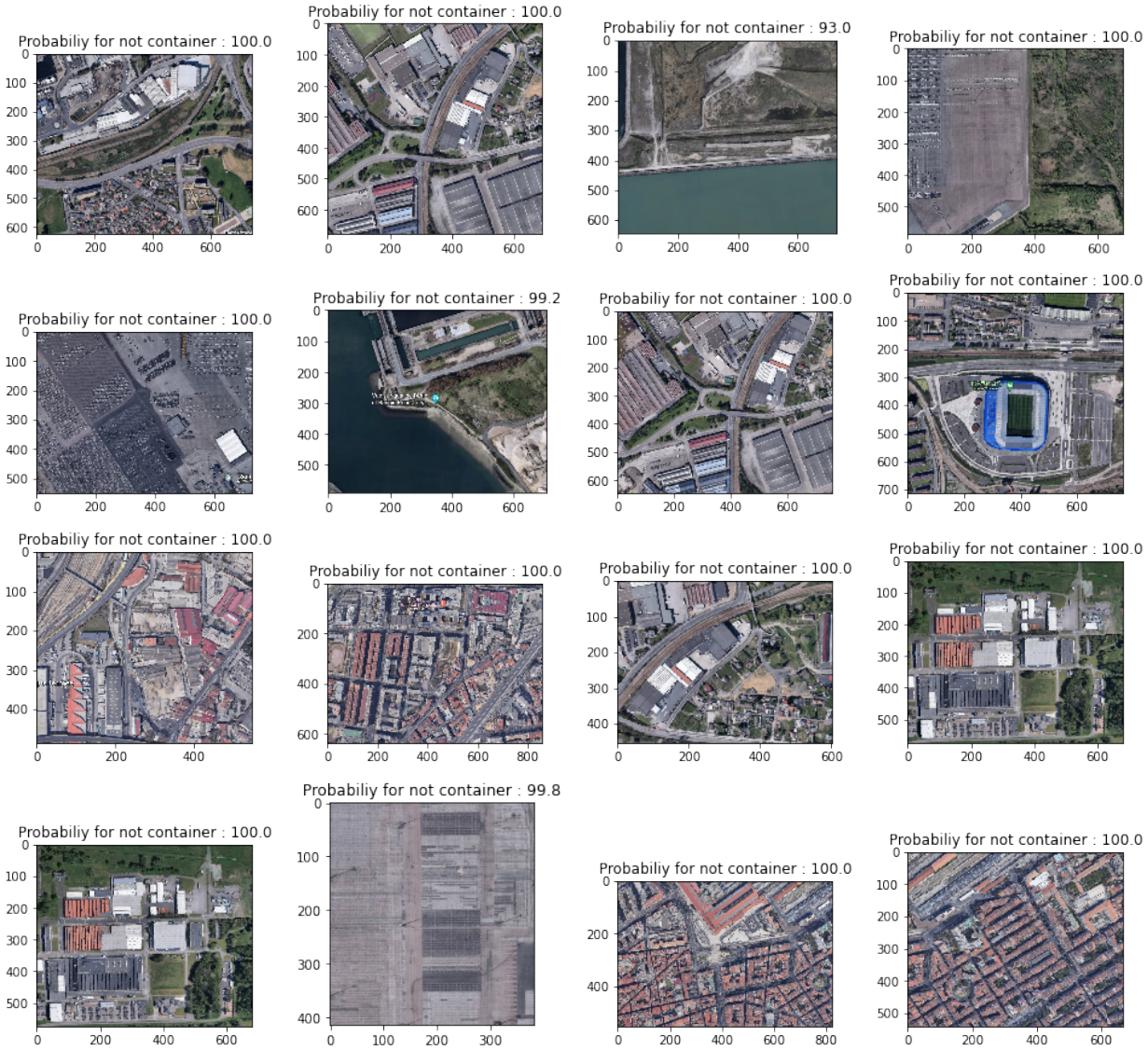


Figure 74: The Confidence of VGG16 at lr = 0.0004(1)

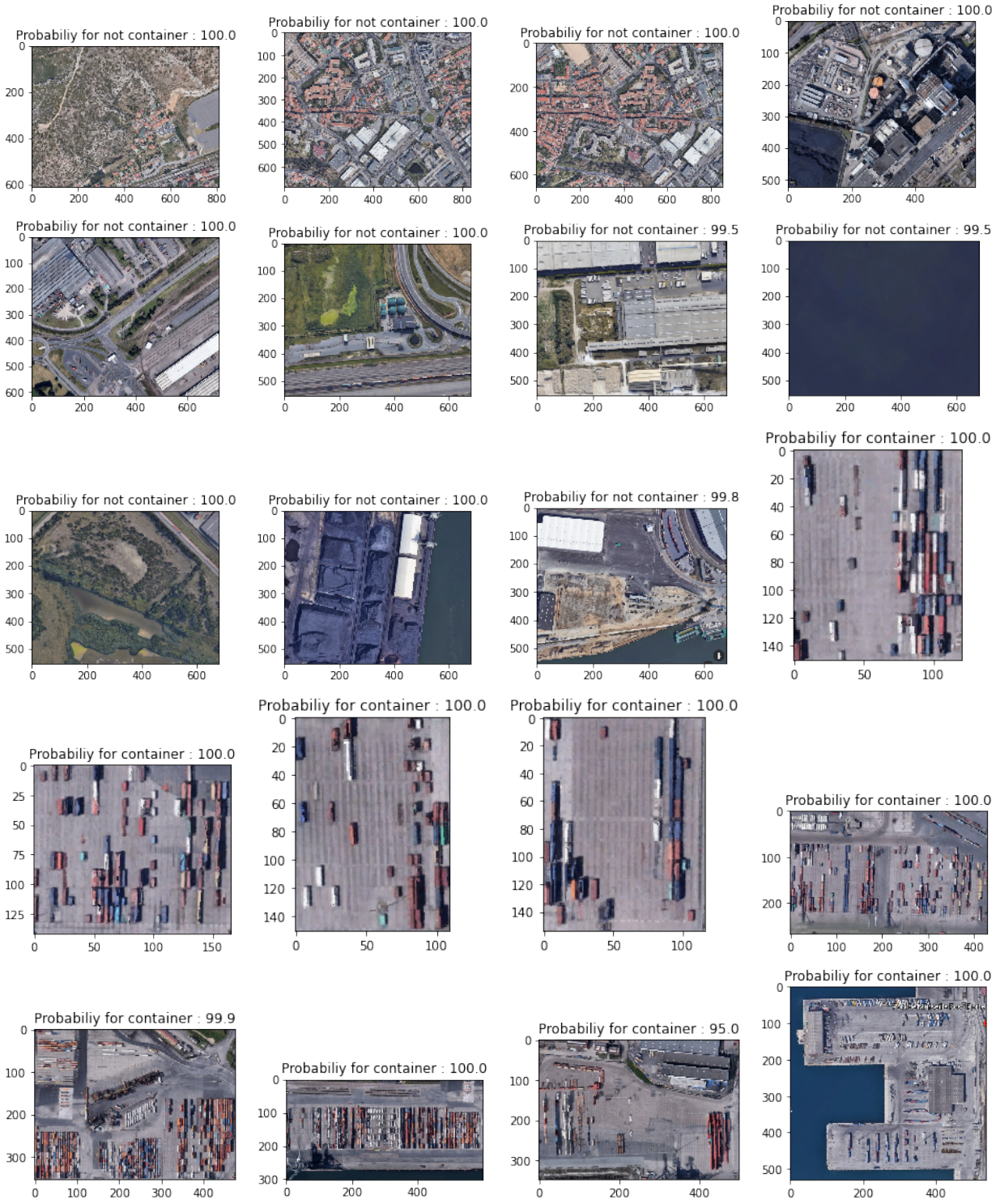


Figure 75: The Confidence of VGG16 at lr = 0.0004(2)

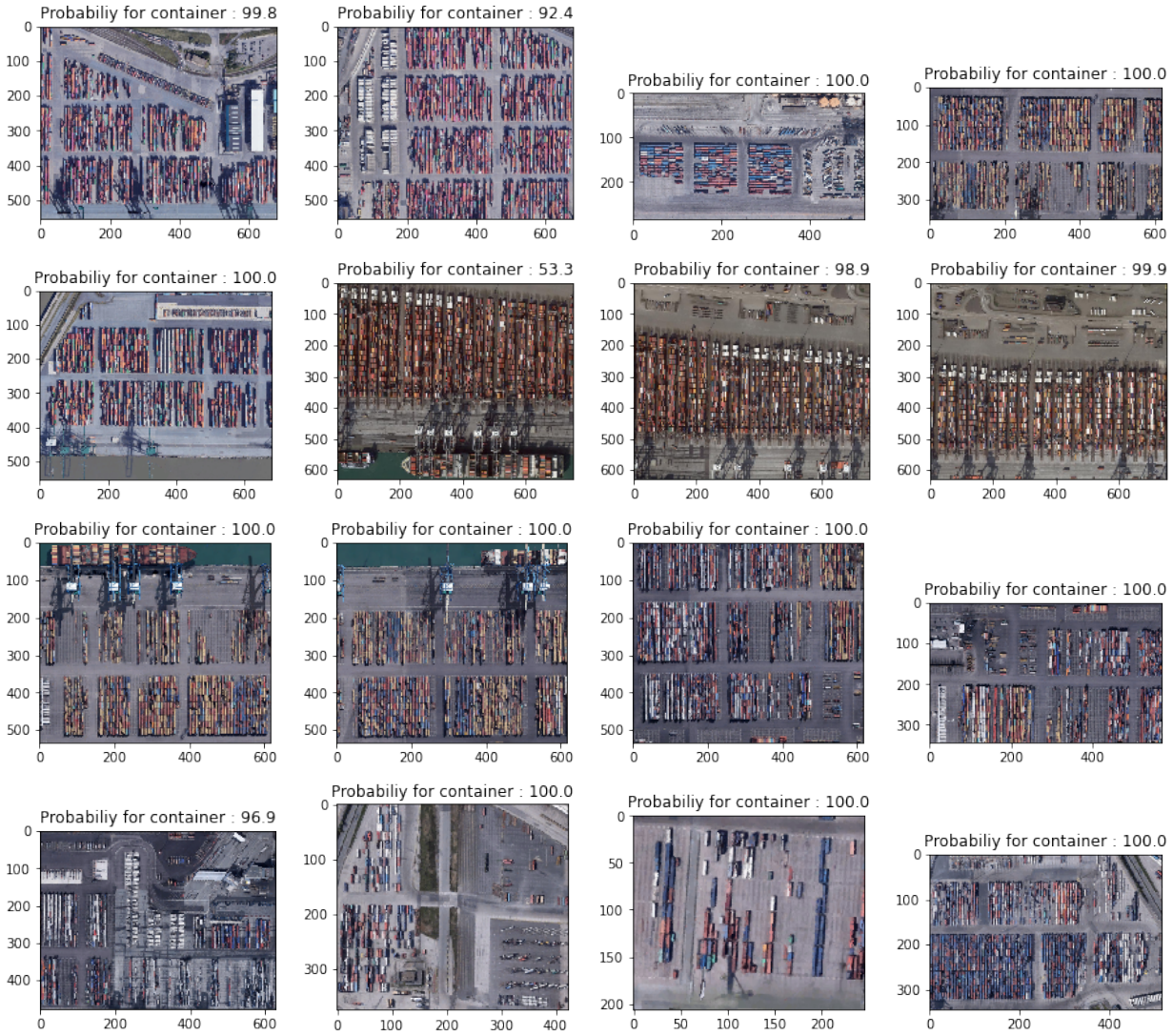


Figure 76: The Confidence of VGG16 at lr = 0.0004(3)

2.learning rate =0.0005.

Figure 77: The Confidence of VGG16 at lr = 0.0005(1)

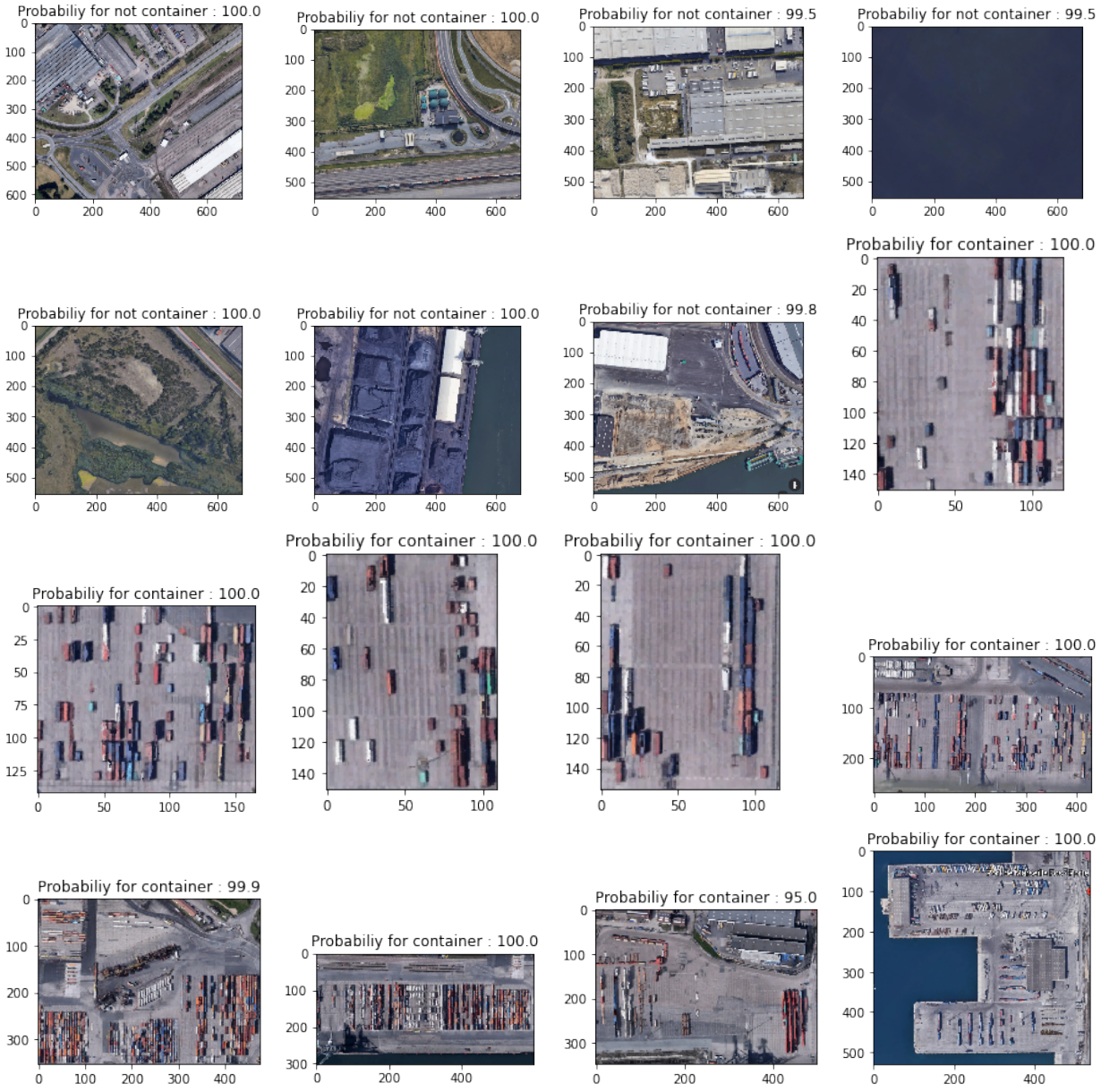


Figure 78: The Confidence of VGG16 at lr = 0.0005(2)

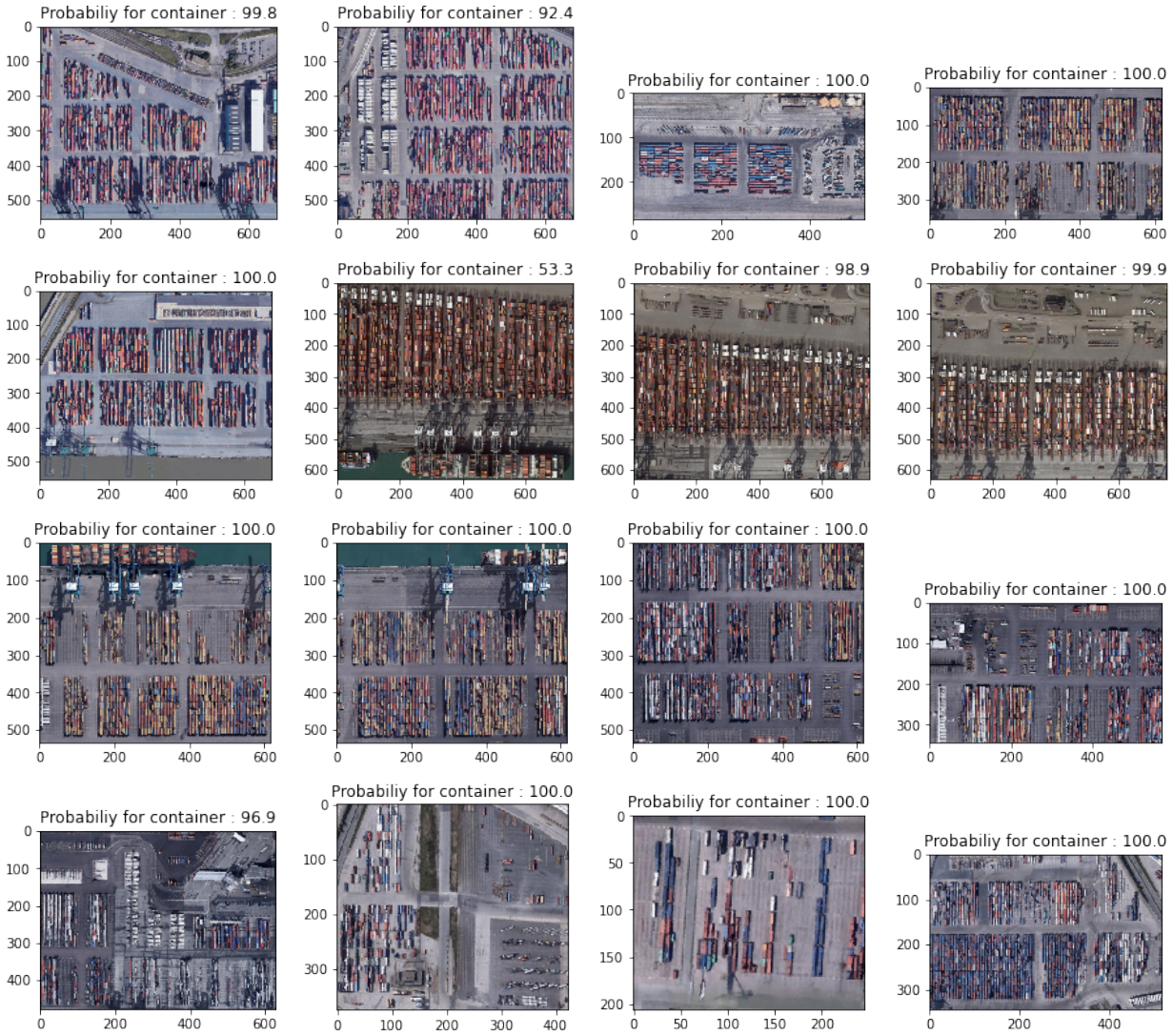


Figure 79: The Confidence of VGG16 at lr = 0.0005(2)

3.learning rate =0.0006

Figure 80: The Confidence of VGG16 at lr = 0.0006(1)

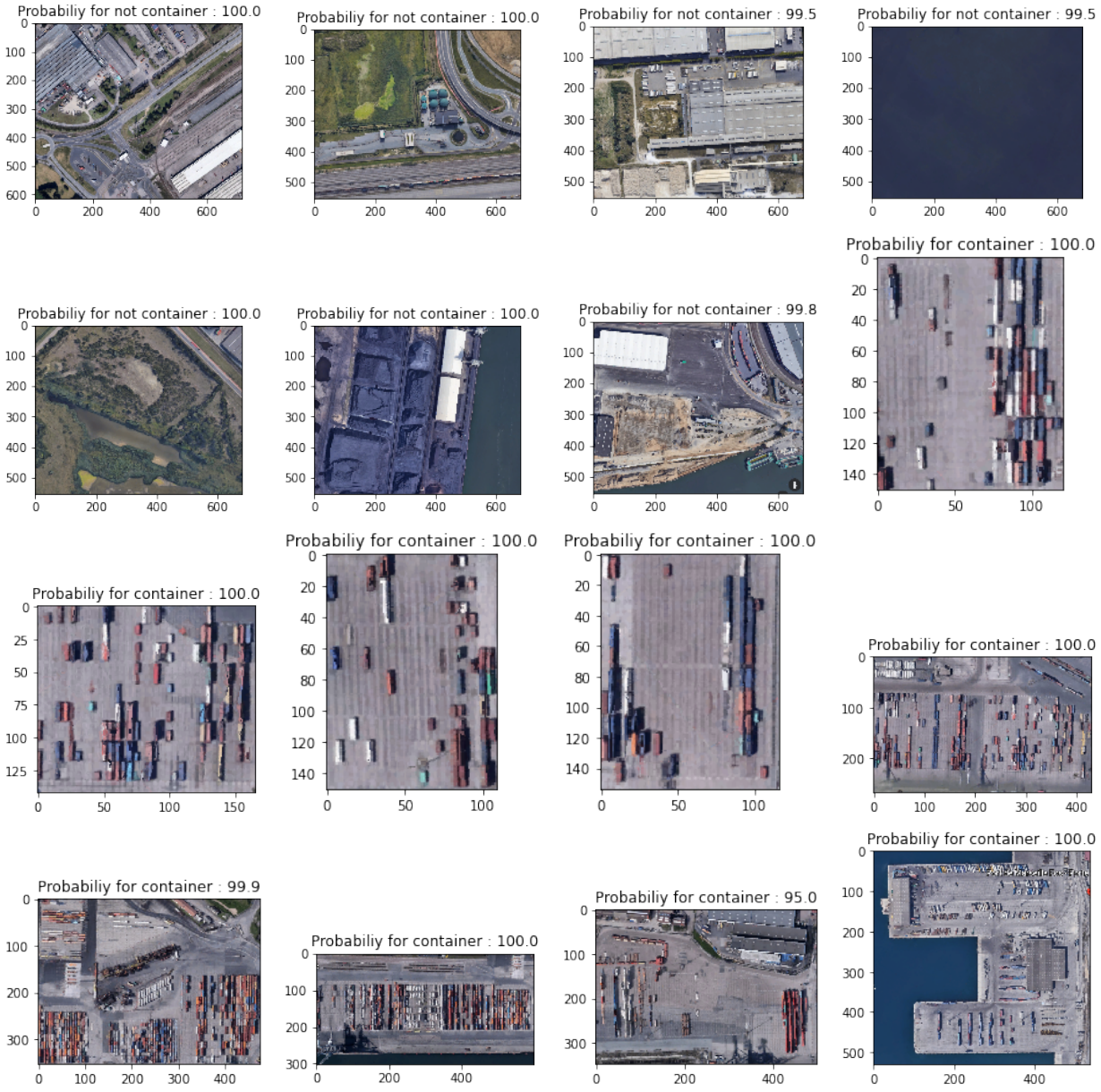


Figure 81: The Confidence of VGG16 at lr = 0.0006(2)

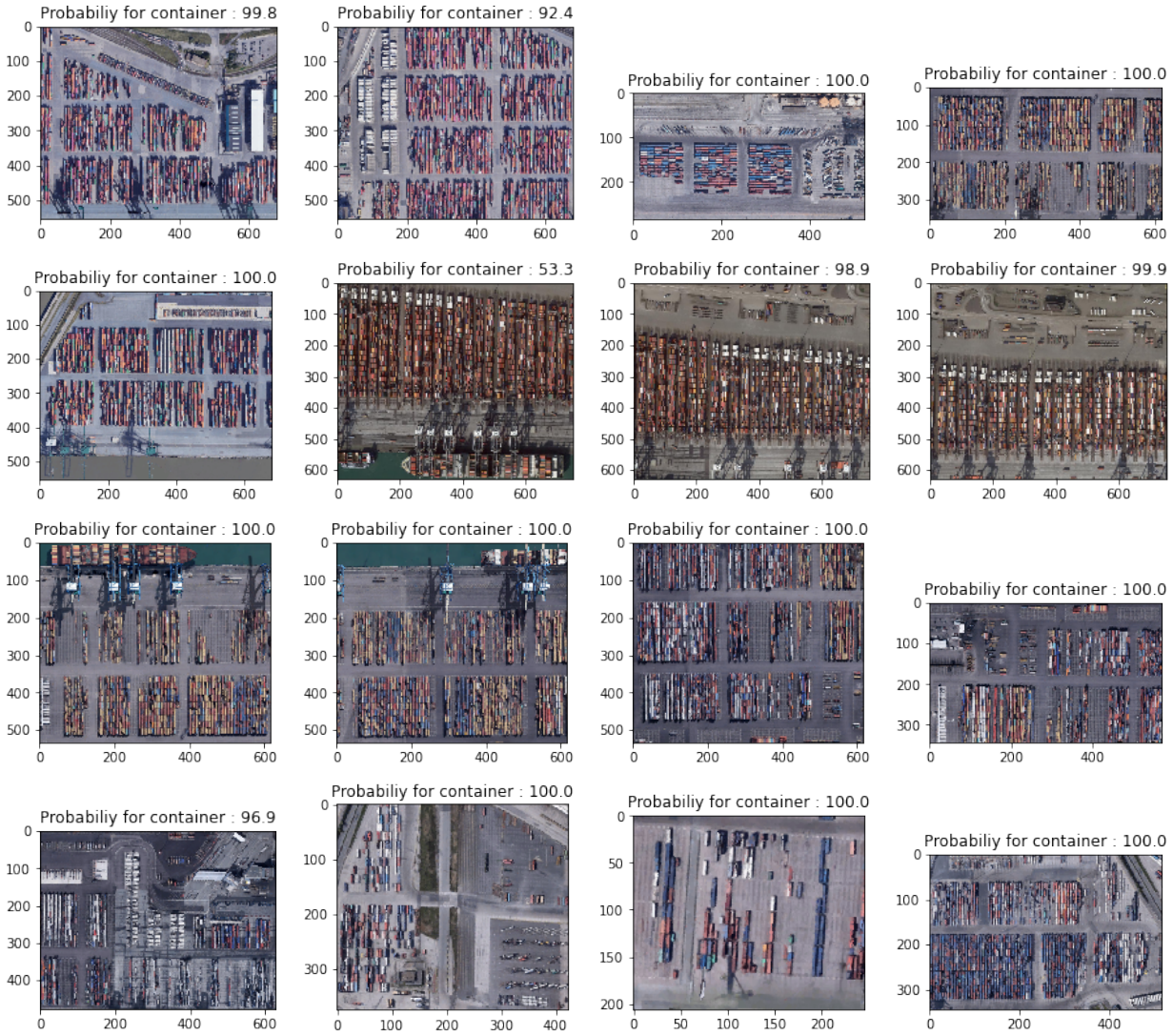


Figure 82: The Confidence of VGG16 at lr = 0.0006(3)

17.4 Model1_150

1. Accuracy and Loss

   Under this condition,the model converges at the 14th epoch:

   The train accuracy reaches at 92.6%, train loss reaches at 0.205.

   The validation accuracy reaches at 94.9%,validation loss reaches at 0.159.

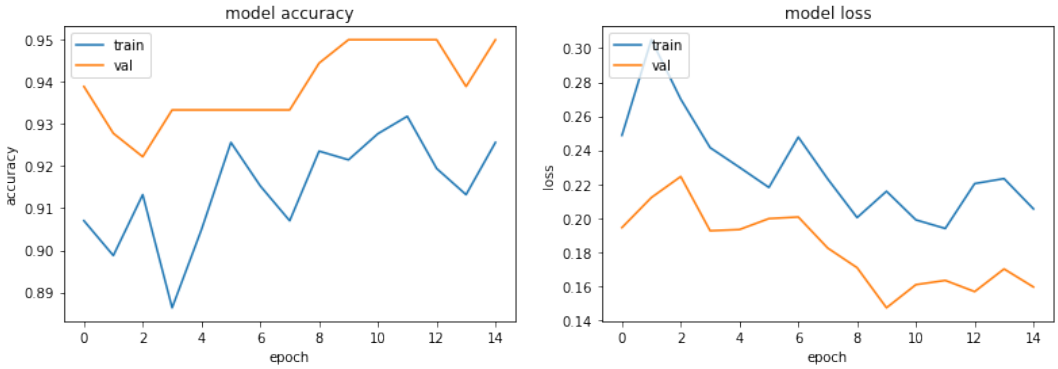


Figure 83: Accuracy and Loss 150

2. Confusion Matrix

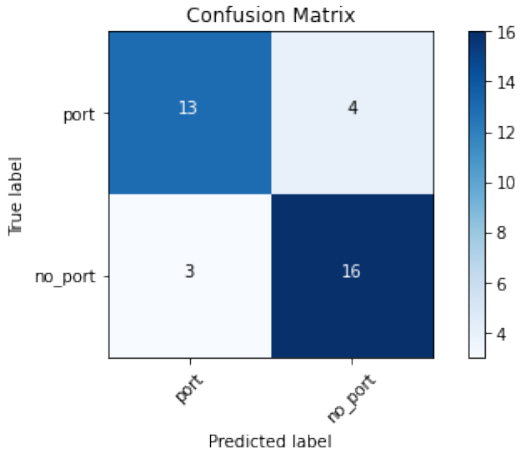


Figure 84: Confusion Matrix 150

3. Confidence

   The confidence under this condition is at around 80%.

Figure 85: The Confidence of VGG16 at input image size =150X150(1)

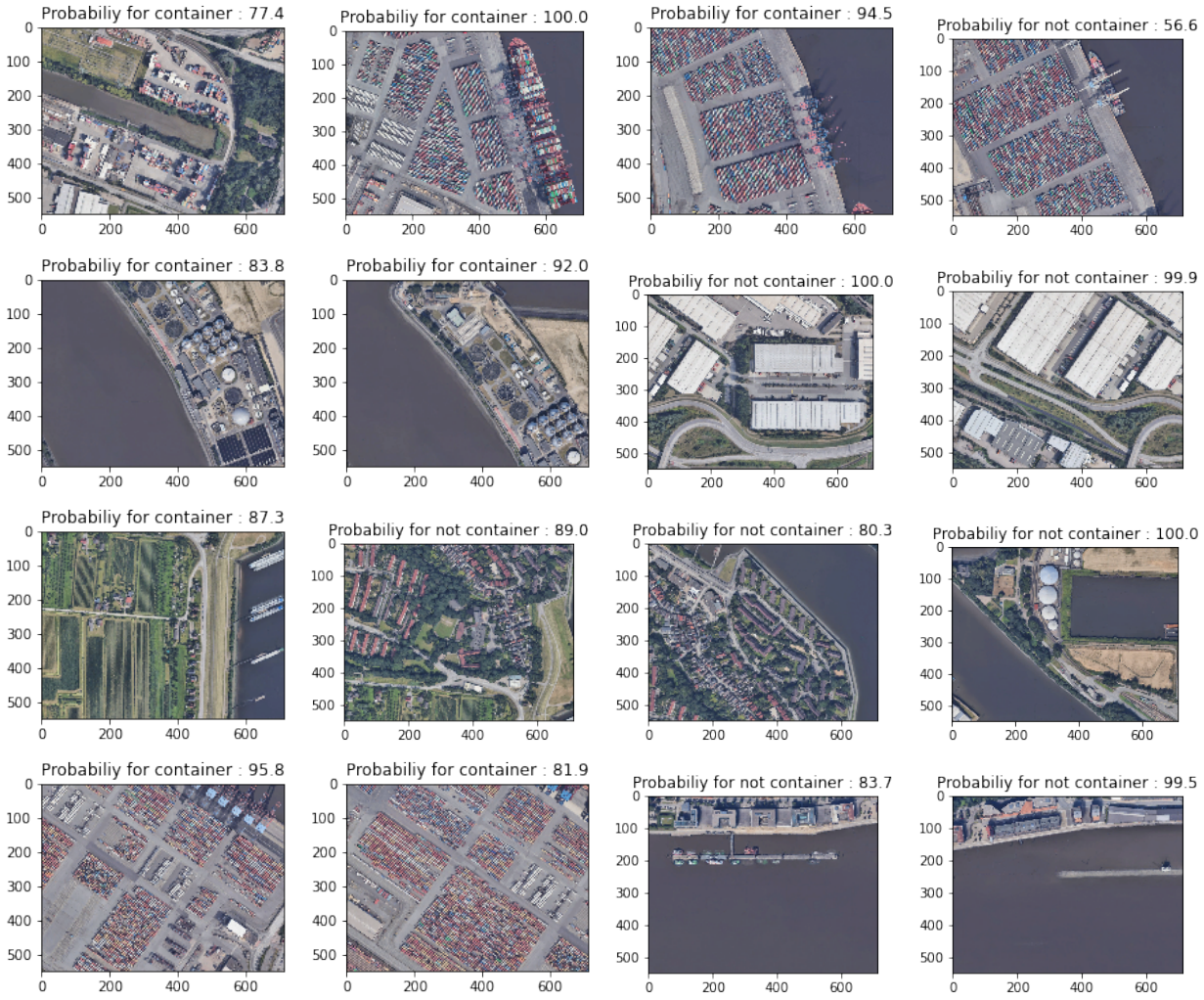


Figure 86: The Confidence of VGG16 at input image size =150X150(2)

17.5 Model2_224

1. Accuracy and Loss

   Under this condition,the model converges at the 17th epoch:

   The train accuracy reaches at 99.6% ,train loss reaches at around 0.018.

   The validation accuracy reaches at 97.2%,the validation loss reaches at 0.128.

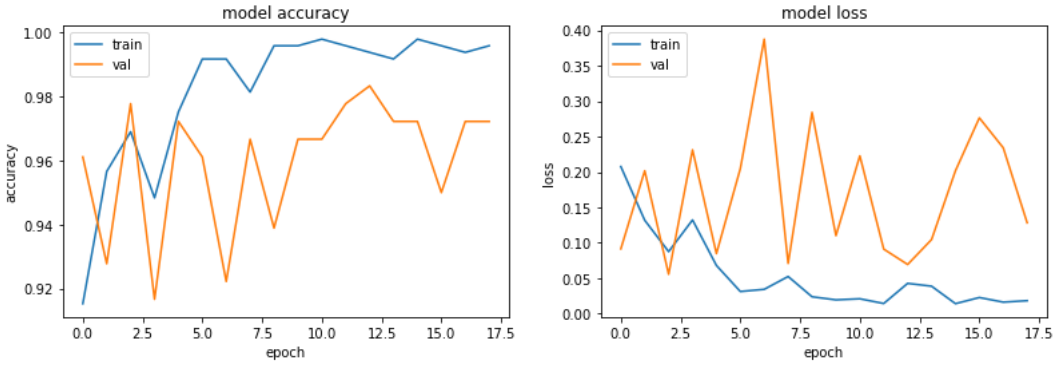


Figure 87: Accuracy and loss 224

2. Confusion Matrix

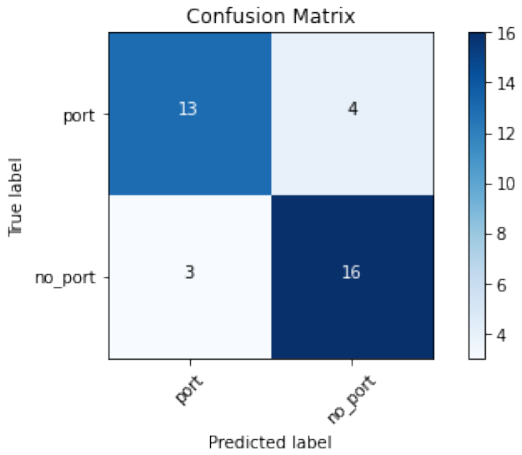


Figure 88: Confusion Matrix 224

3. Confidence

   The confidence under this condition is at around 90%.

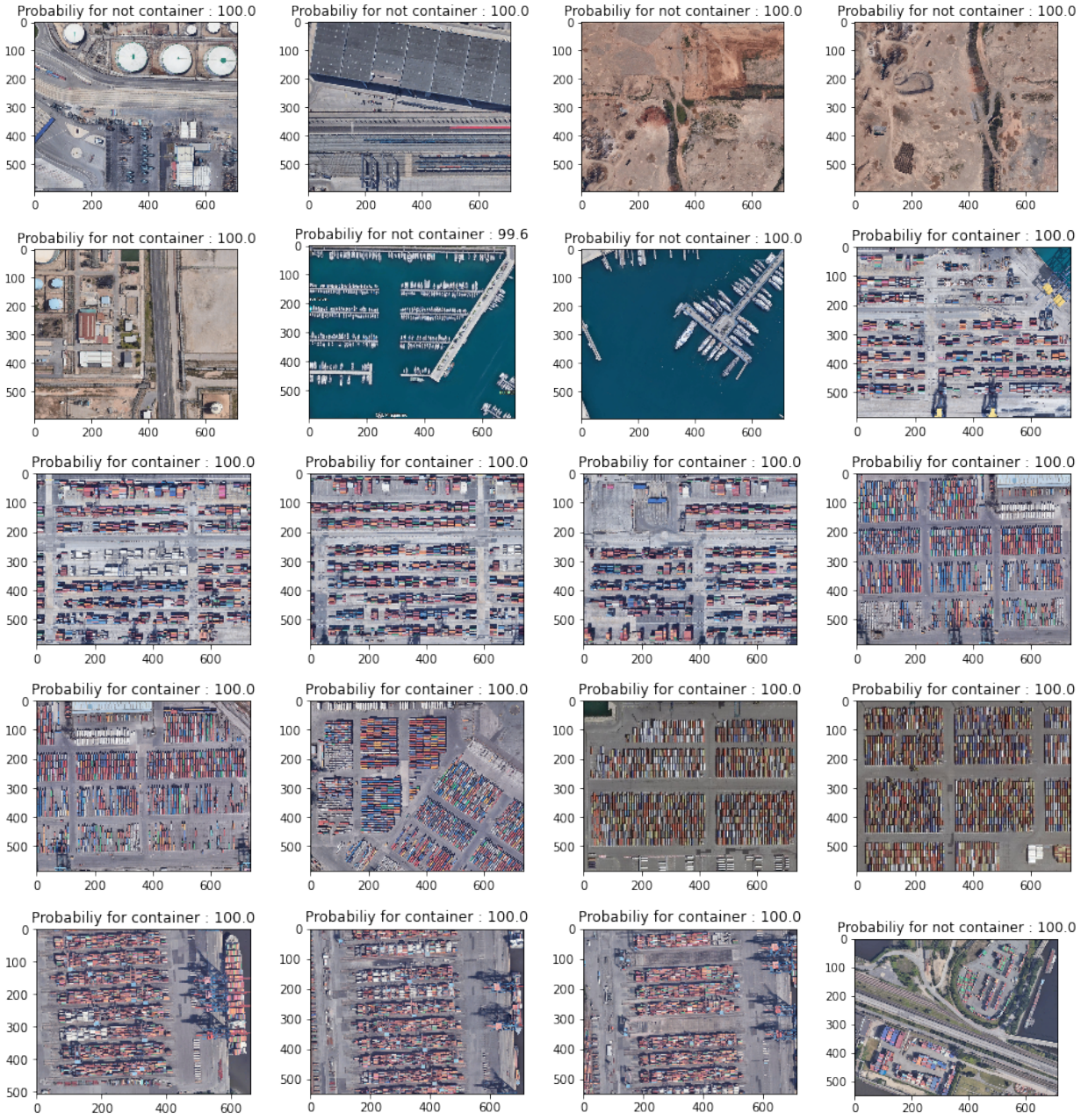


Figure 89: The Confidence of VGG16 at input image size =224X224(1)

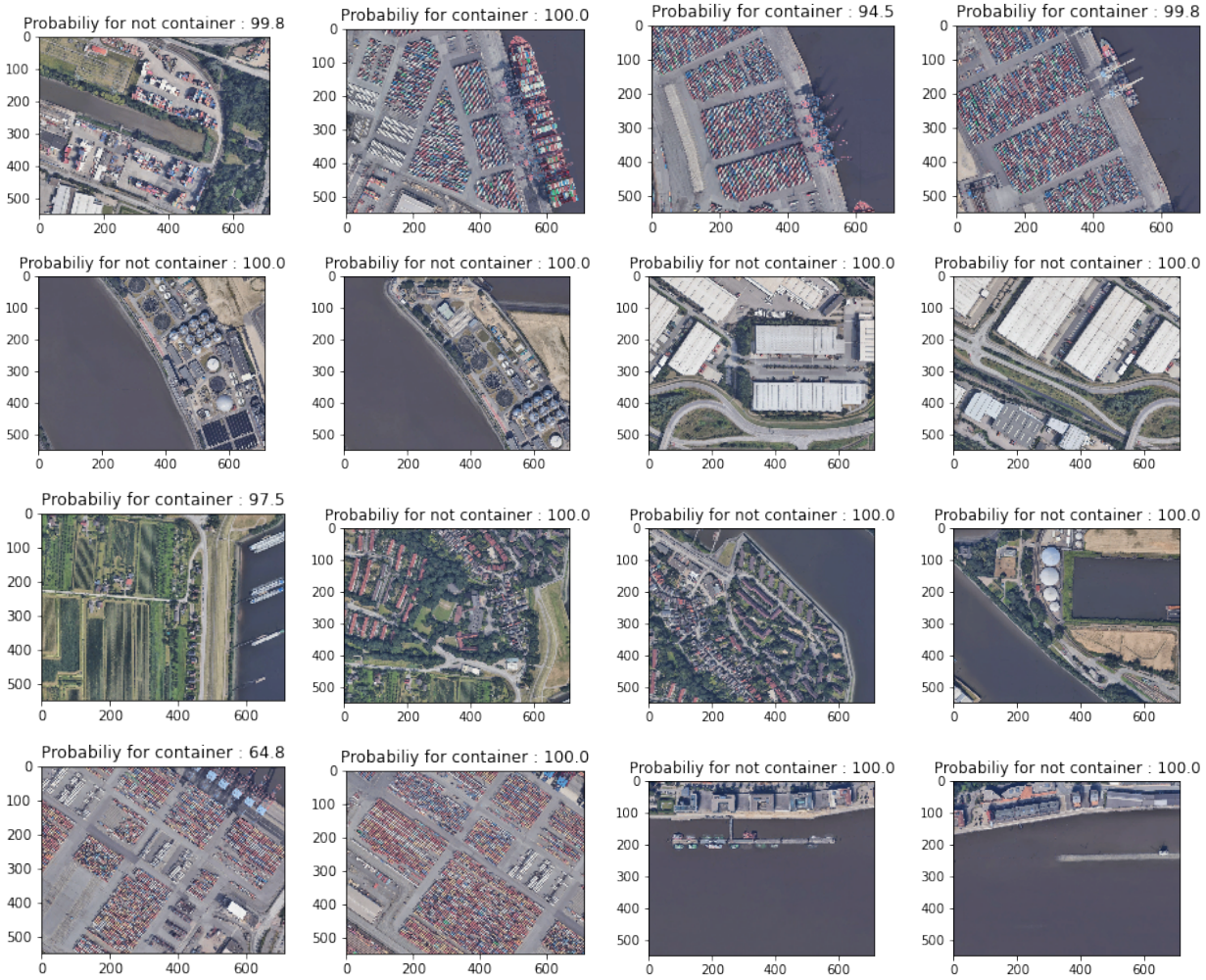


Figure 90: The Confidence of VGG16 at input image size =224X224(2)

17.6 Model3_300

1. Accuracy and Loss

   Under the condition,the model converges at the 14th epoch:

   The train accuracy reaches at 97.9% ,train loss reaches at 0.052.

   The validation accuracy reaches at 97.2%, validation loss reaches at 0.132.

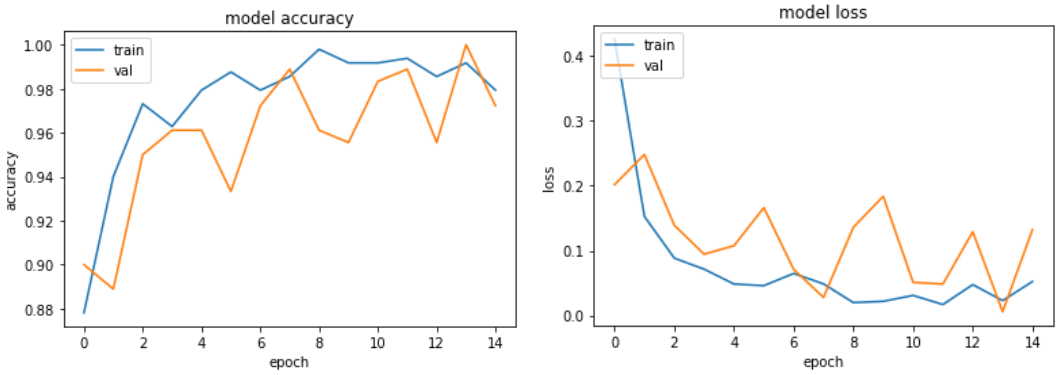


Figure 91: Accuracy and Loss 300

2. Confusion Matrix

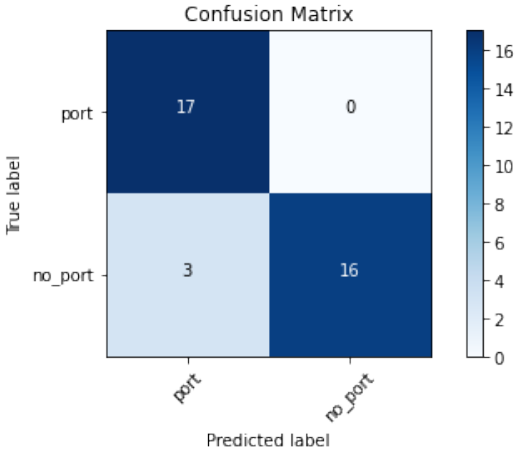


Figure 92: Confusion Matrix 300

3. Confidence

   The confidence under this condition is at around 80%.

   Find the full results of confidence at **??**.

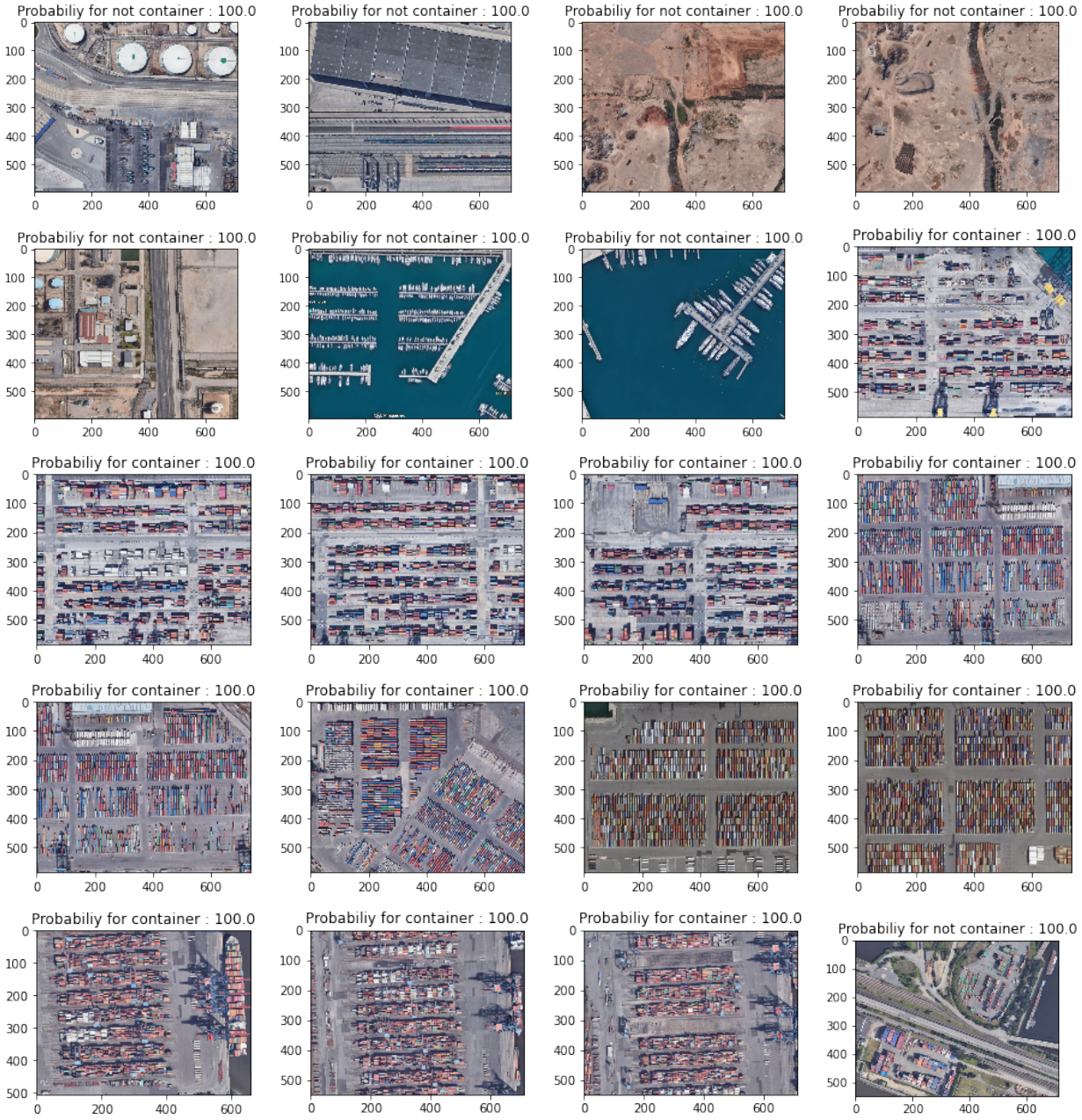


Figure 93: The Confidence of VGG16 at input image size =300X300(1)

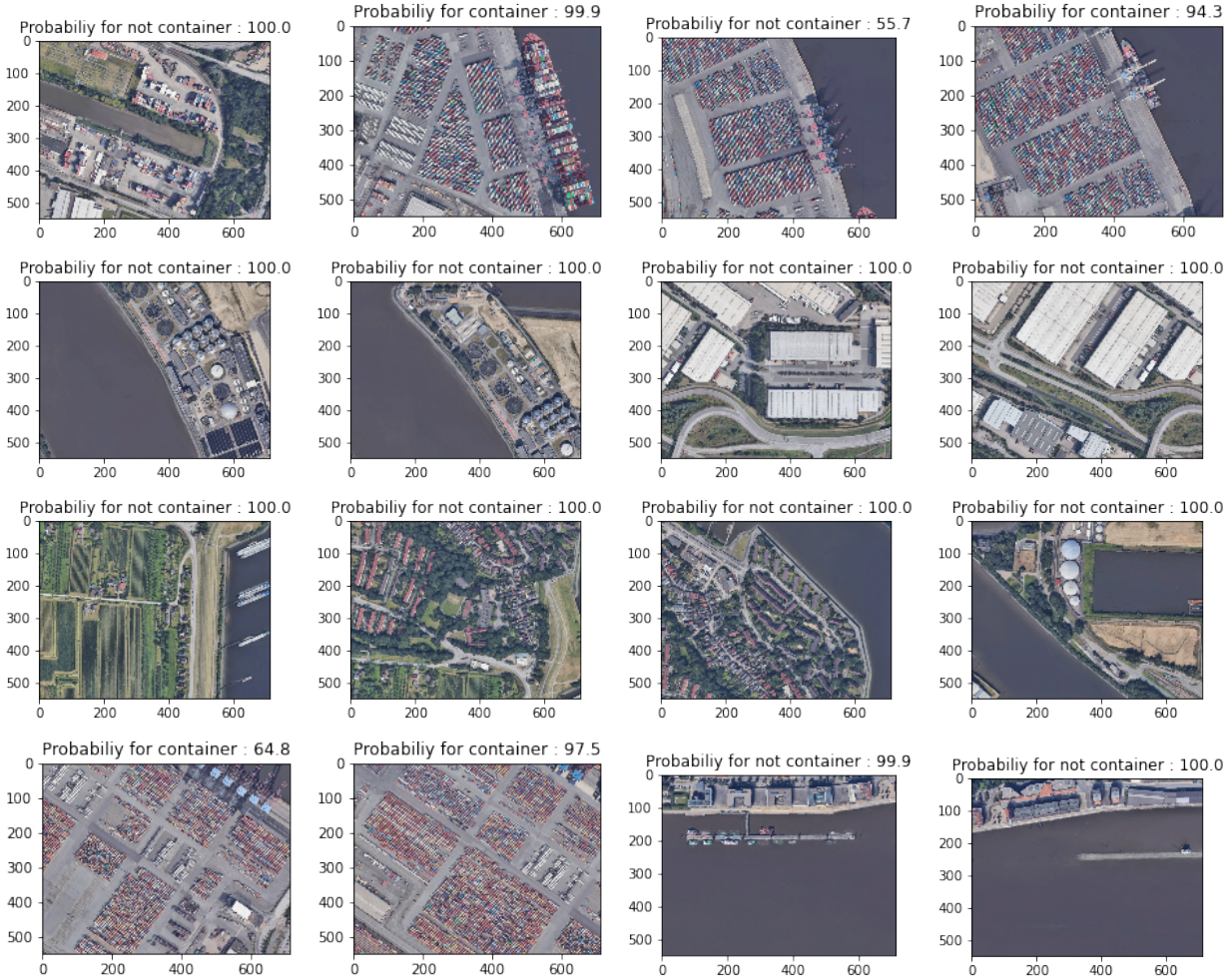


Figure 94: The Confidence of VGG16 at input image size =300X300(2)

17.7 Model4_325

1. Accuracy and Loss

   Under this condition,the model converges at the 25th epoch:

   The train accuracy reaches at 99.2%,train loss reaches at 0.023.

   The validation accuracy reaches at 98.9%,validation loss 0.057.

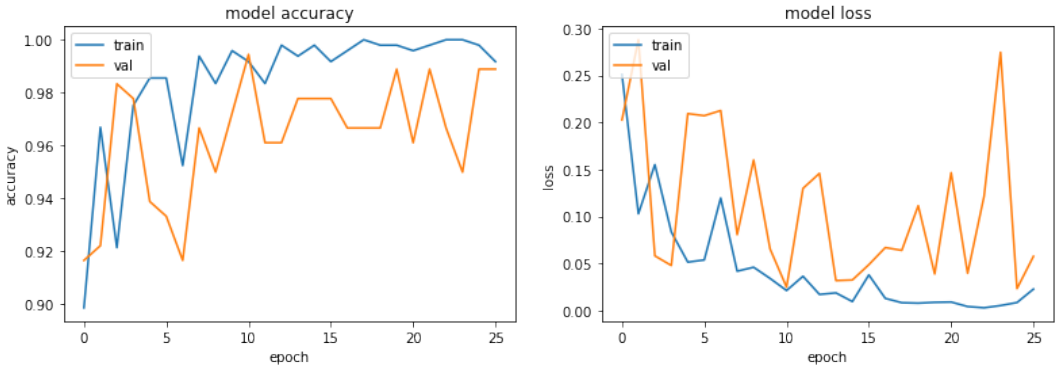


Figure 95: Accuracy and Loss 325

2. ConfusionMatrix

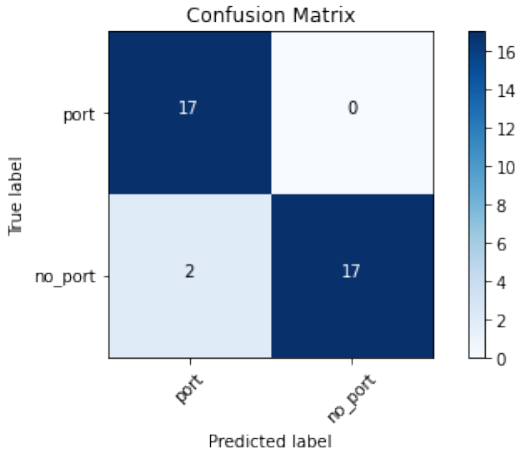


Figure 96: The Confusion Matrix 325

3. Confidence

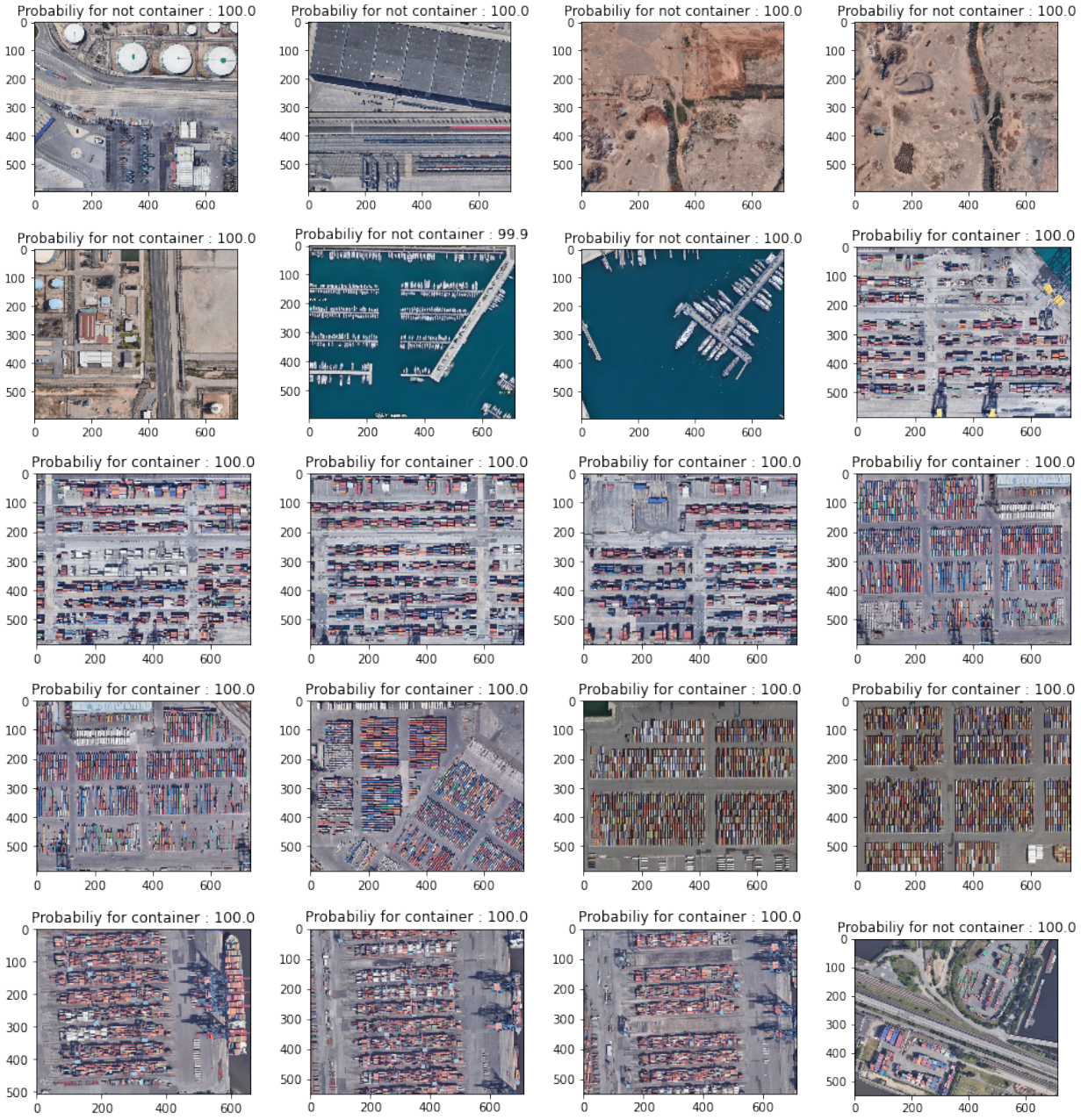


Figure 97: The Confidence of VGG16 at input image size =325×325(1)

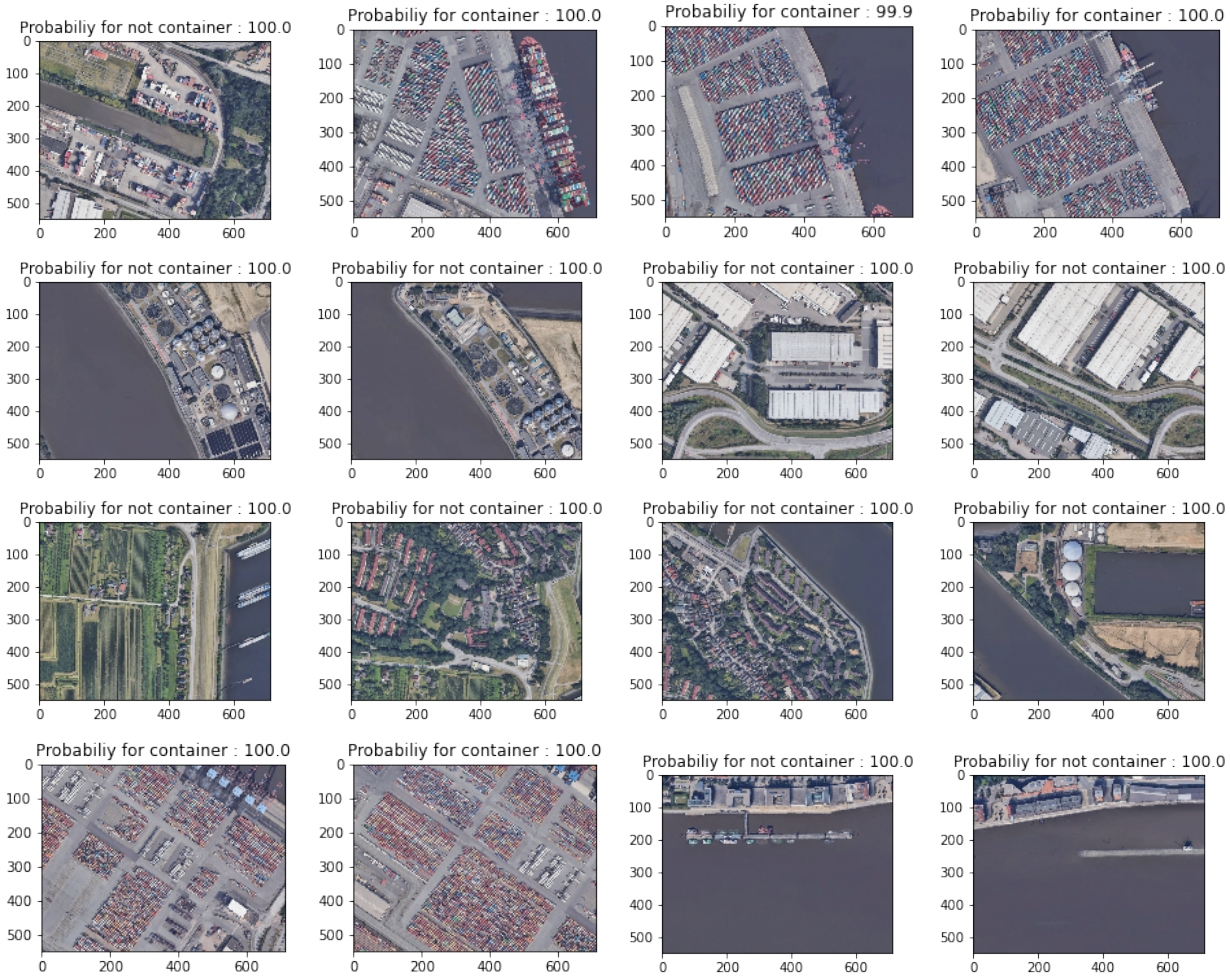


Figure 98: The Confidence of VGG16 at input image size =325×325(2)

17.8 Model5_350

1. Accuracy and Loss

   Under the condition,the model converges at the 19th epoch:

   The train accuracy reaches at 99.6%, train loss reaches at 0.009.

   The validation accuracy reaches at 99.4%, validation loss reaches at the 0.015.

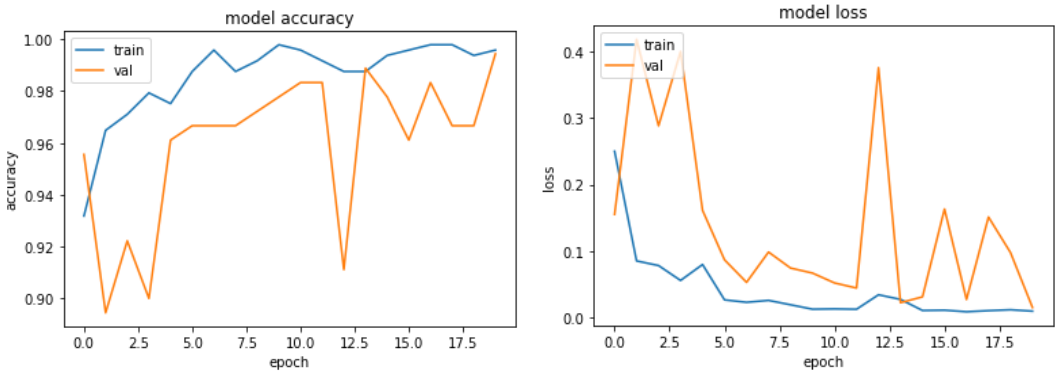


Figure 99: Accuracy and Loss 350

2. Confusion Matrix

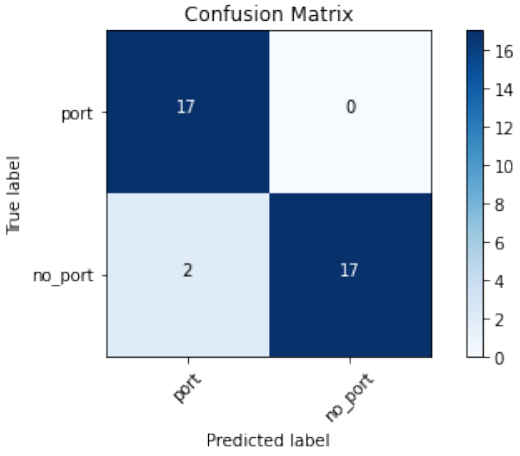


Figure 100: Confusion Matrix 350

3. Confidence

   The confidence under this condition is at around 80%.

   Find the full results of confidence at **??**.

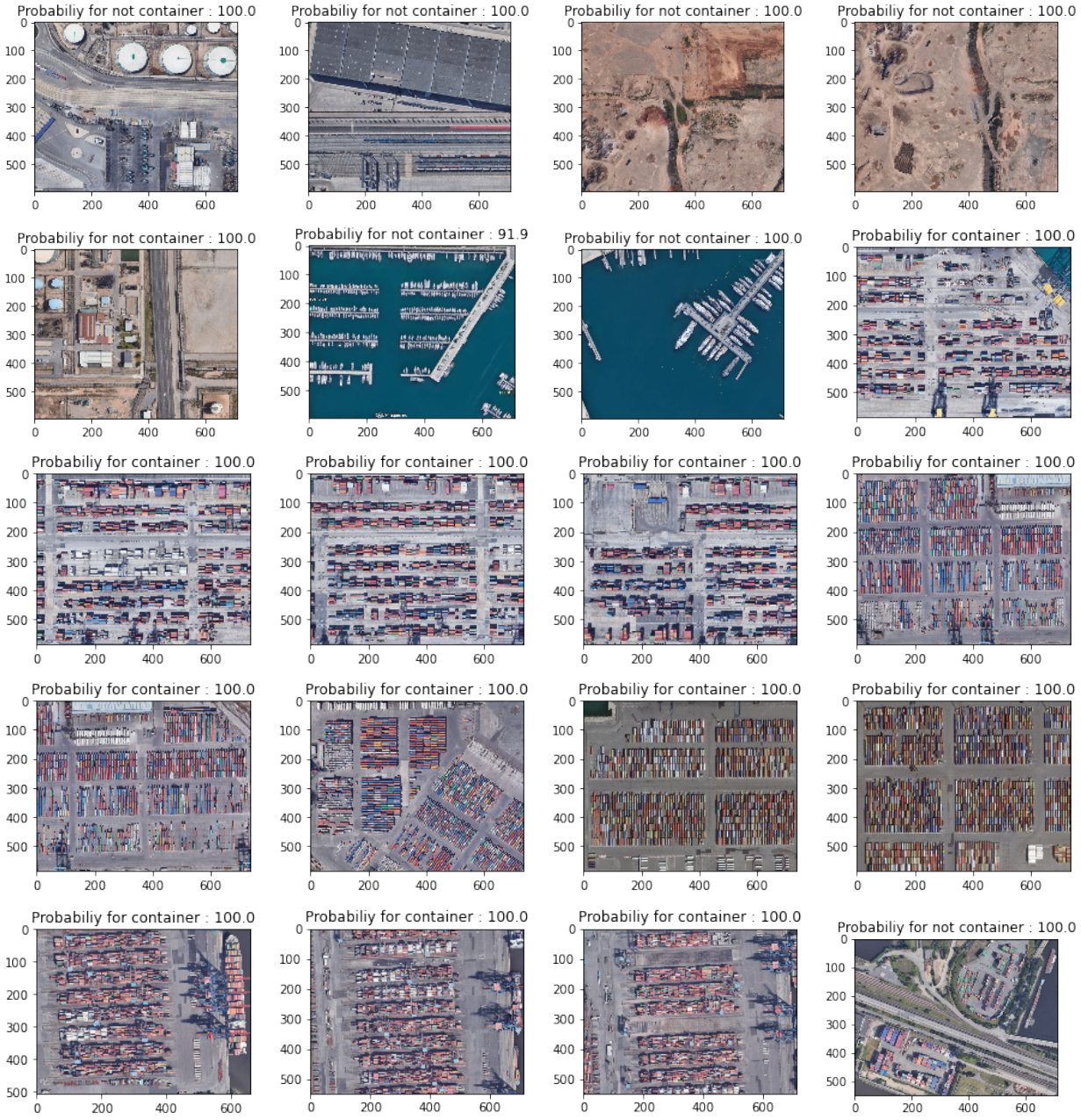


Figure 101: The Confidence of VGG16 at input image size =350X350(1)

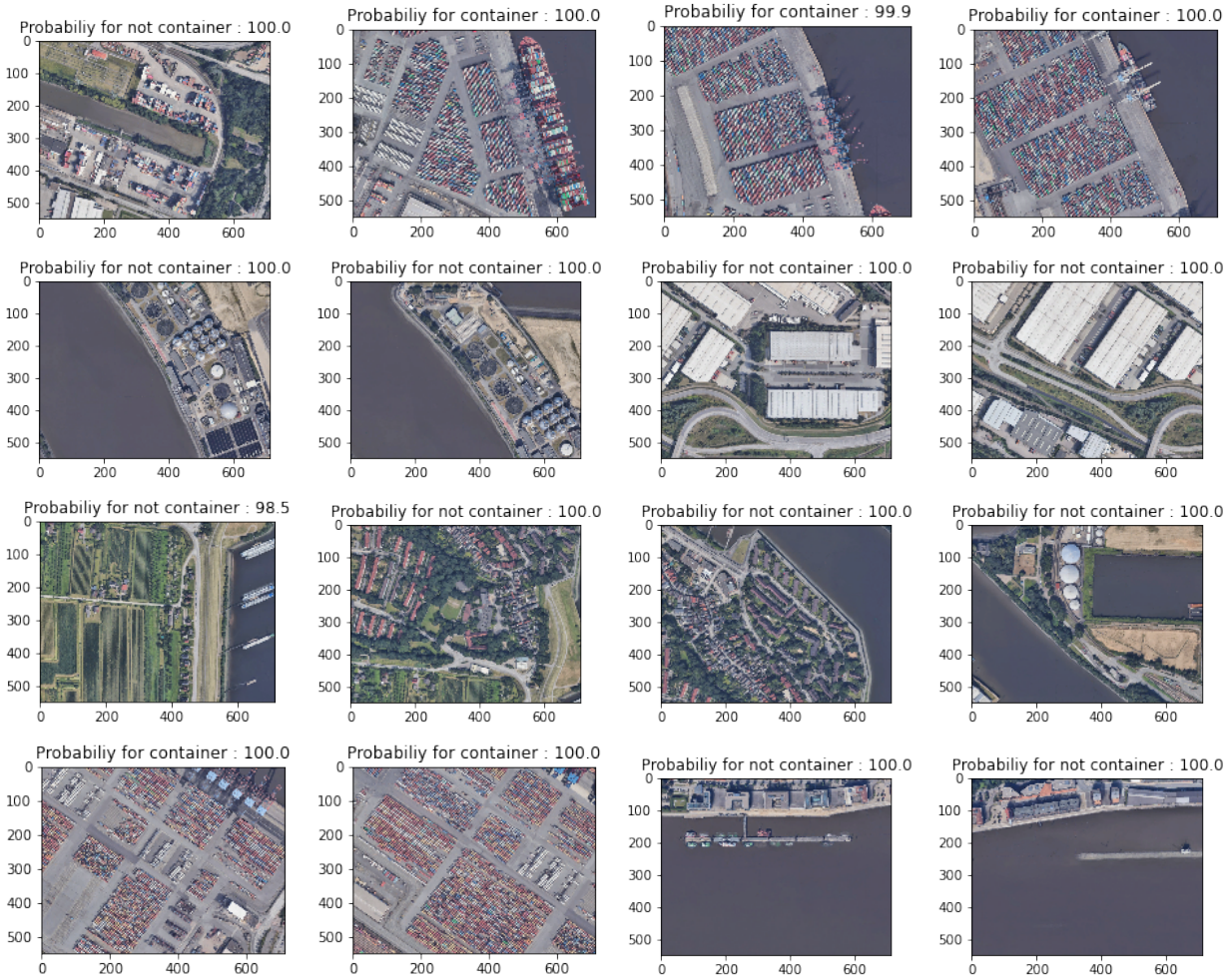


Figure 102: The Confidence of VGG16 at input image size =350X350(2)

17.9 Result of RCNN

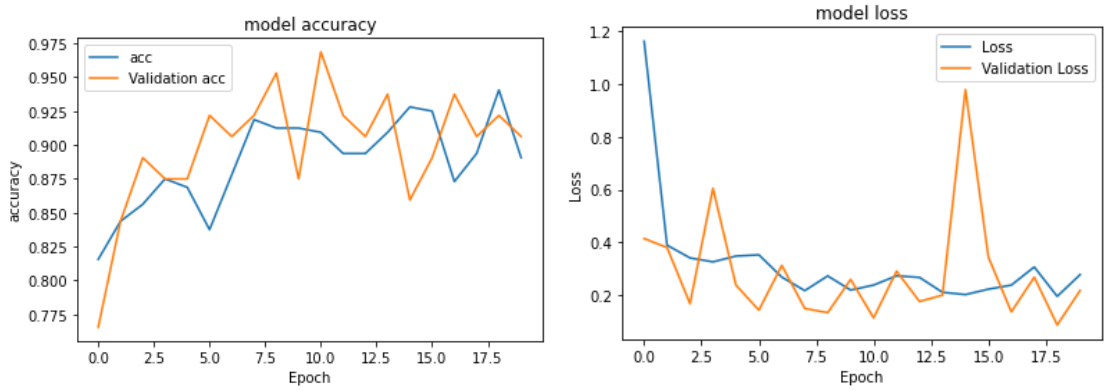


Figure 103: Model1_10 accuracy and loss

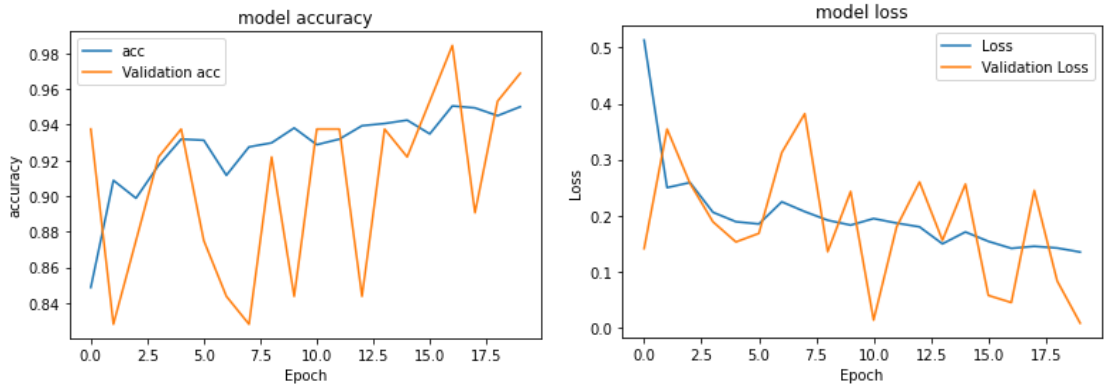


Figure 104: Model2_50 accuracy and loss

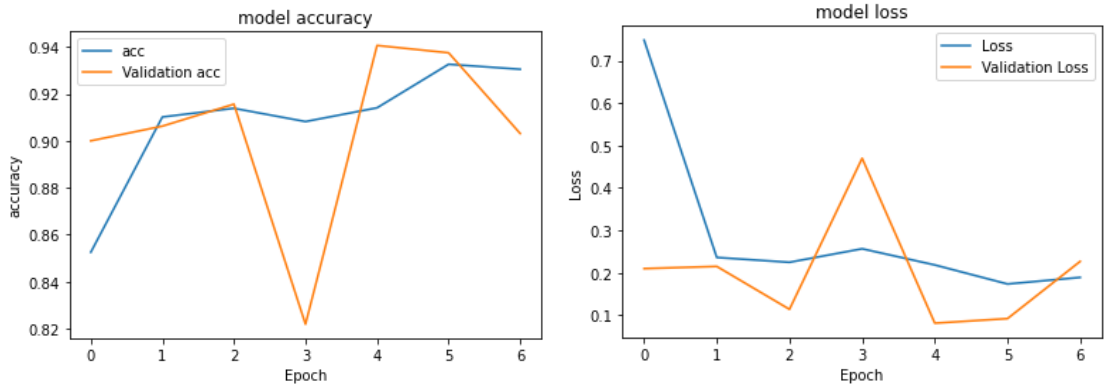


Figure 105: Model3_64 accuracy and loss

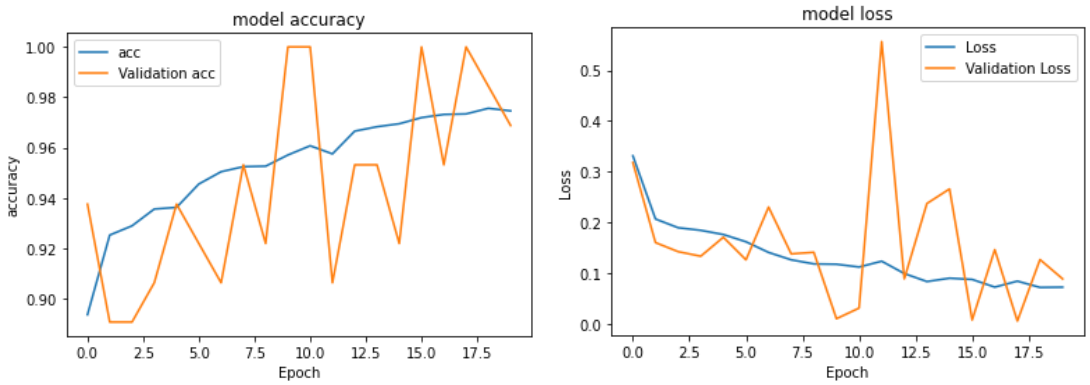


Figure 106: Model4_128 accuracy and loss

17.10 Threshold RCNN

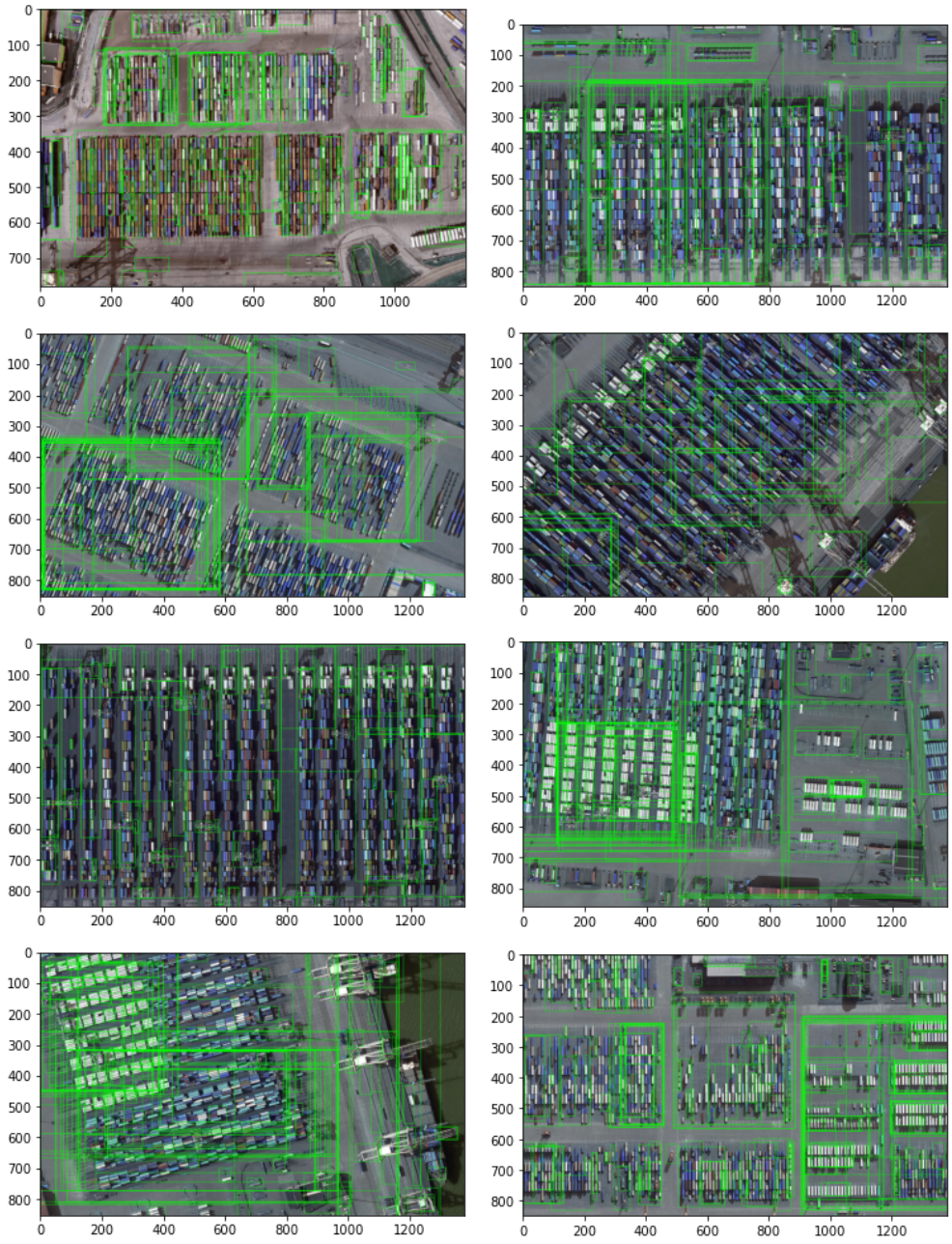


Figure 107: The result at the threshold at 0.6(1)

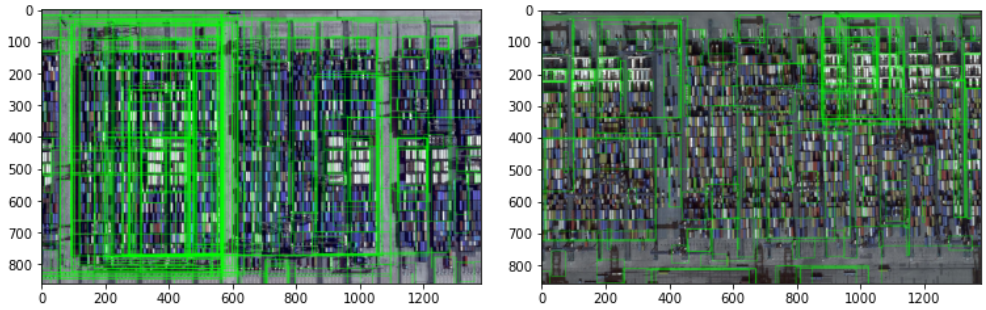


Figure 108: The result at the threshold at 0.6(2)

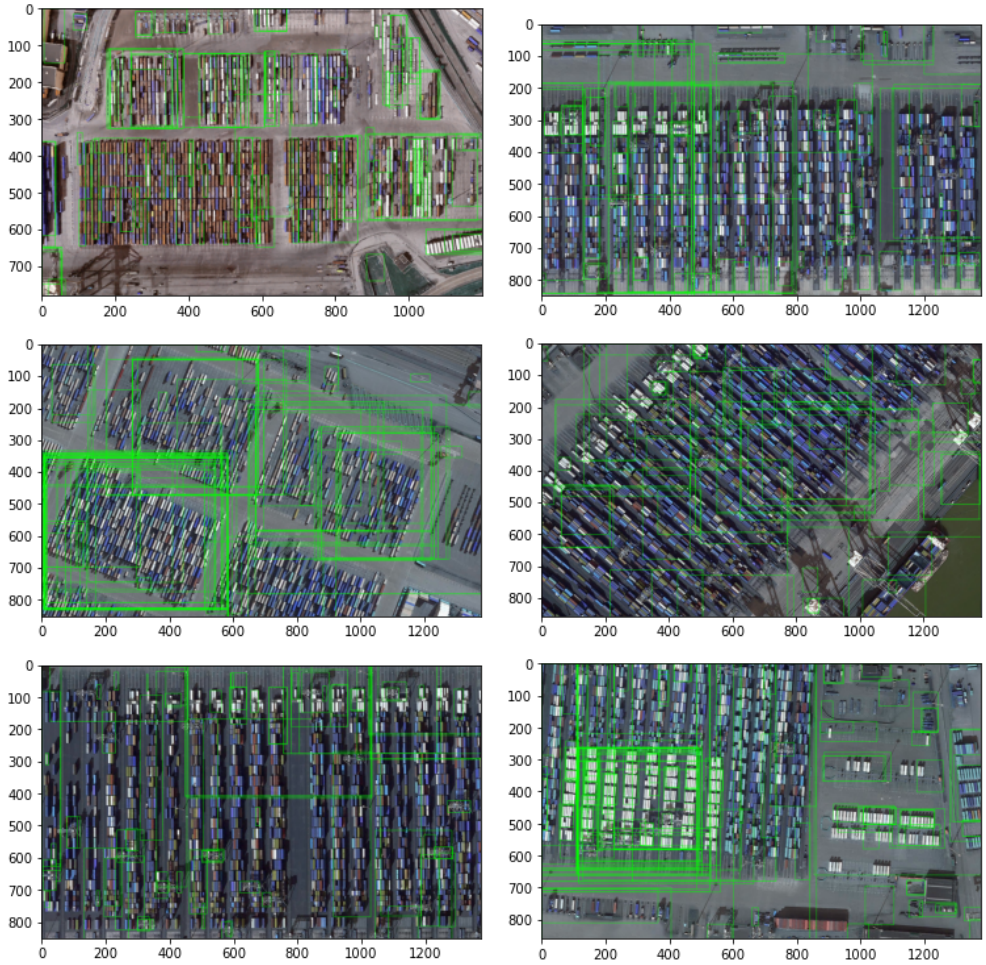


Figure 109: The result at the threshold at 0.70(1)

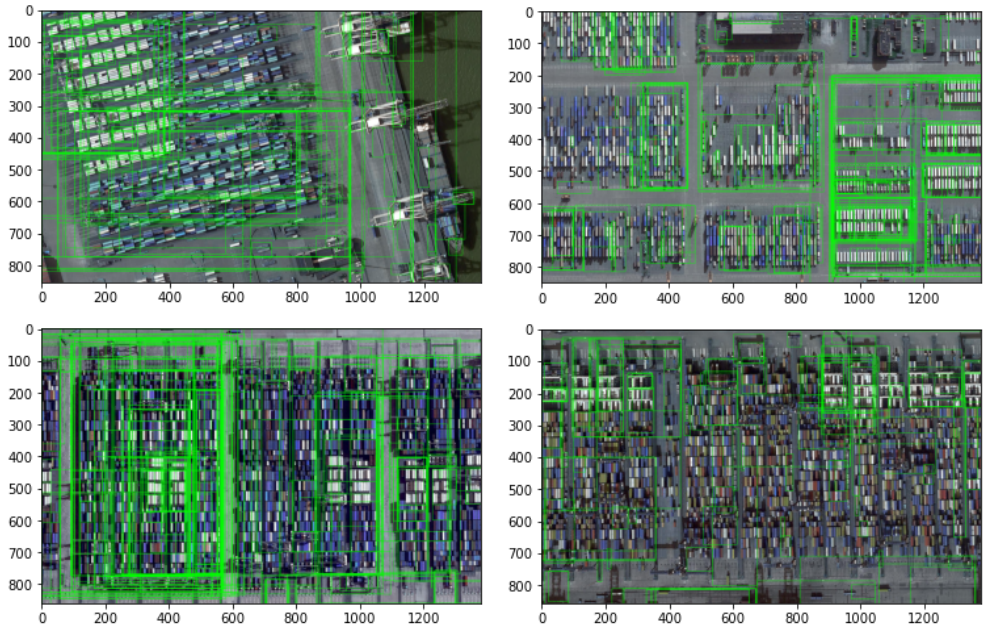


Figure 110: The result at the threshold at 0.70(2)

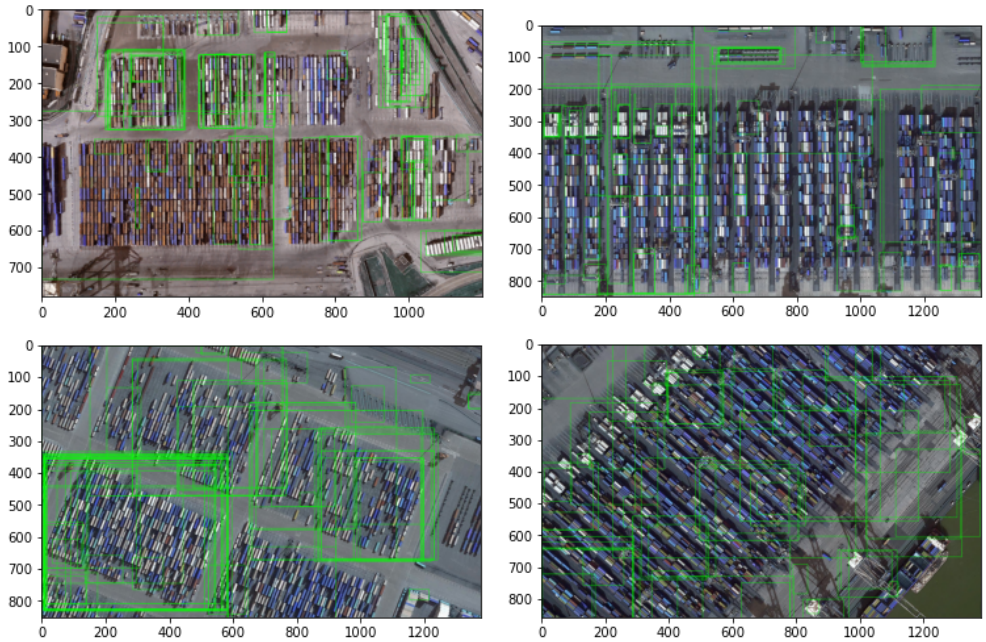


Figure 111: The result at the threshold at 0.80(1)

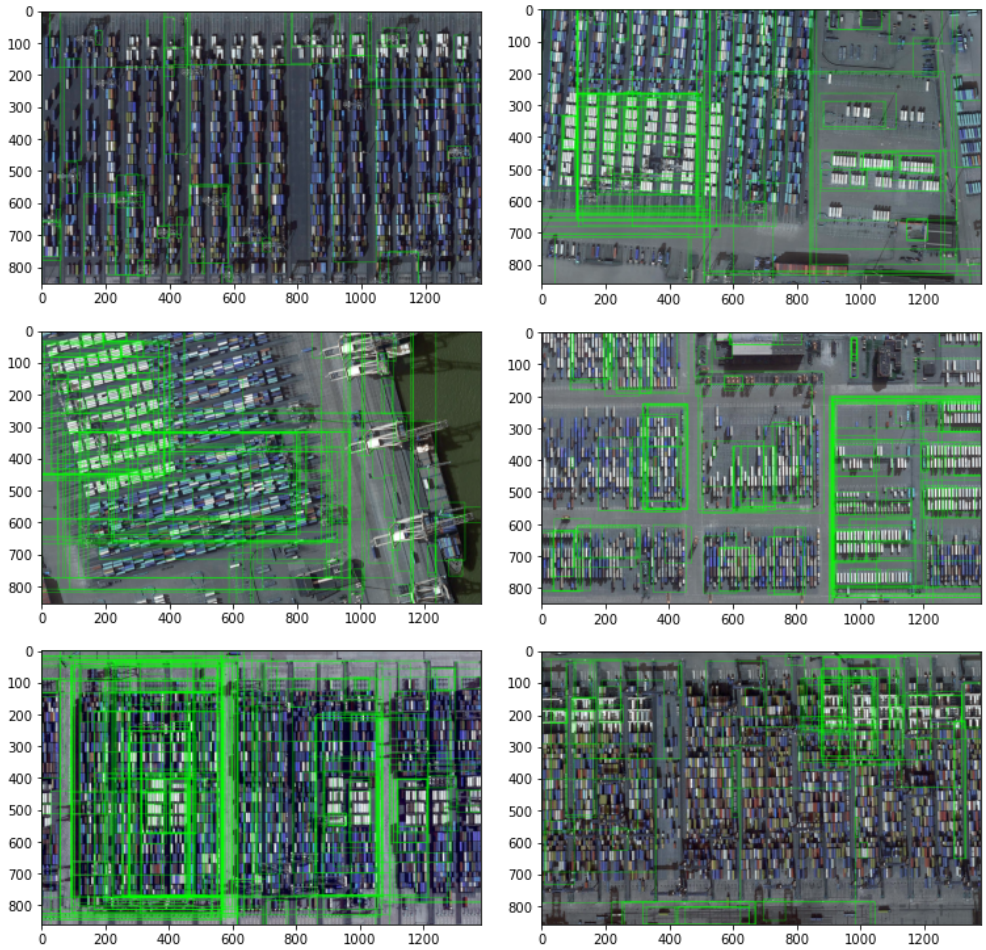


Figure 112: The result at the threshold at 0.80(2)