Deep learning based medical image recognition

Hichma KARI

3rd year MPCI

From November 26th to December 21st 2018

Internship tutor : Ronan SICRE

Host institution : LIS

Acknowledgements

I would like to express my very great appreciation to my tutor Ronan Sicre, for his guidance, and suggestions during the whole internship.
I would also like to thank the QARMA team for welcoming me in their laboratory.

Table of contents

Introduction

This report is about the medical images classification problem. Image classification consists in knowing the category of an input image, based on the learning of a certain number of categories, thanks to an annotated image set. Those categories can be anything, so we can use image classification in many fields, such as remote sensing or security check, as we can see in the paper [1] which present an application of classification with identity documents, and explain the methods we'll use here. In this report, we focus on an application of classification on medical images.

During these last years, methods based on deep learning have been used in classic image classification, and this report present some of them.

We focus on the use of pre-trained network to extract the needed features of an images, and pooling methods to obtain a global representation of the image, that can be used in recognition tasks.

Firstly, we'll see the methods used to extract the needed features of an images and with these features recognize the category of the image. Then, we will present some results with medical images sets.

Presentation of the laboratory

The "Laboratoire d'Informatique et Système" (LIS) is the result of the merger of the "Laboratoire d'Informatique Fondamentale de Marseille" (LIF) and the "Laboratoire des Sciences de l'Information et des Systèmes" (LSIS). The LIS brings together more than 375 members : 190 permanent researchers and research professors, more than 125 doctoral students, more than 40 post-docs and 20 IT/IATSS. Research in the LIS is organized around four pole : image and signal, calculation, system analysis and control, and data science.
During my internship, I worked with the QARMA team, that is specialized in data science and calculation.

1 Methods to classify images

To classify images, there is two things to do : find the needed features of images, and discriminate them automatically.
Deep learning can be use for both tasks : Support Vector Machine is used to discriminate data based on some features, that can be extract from images thanks to Artificial Neural Network.

1.1 Support Vector Machine (SVM)

Support Vector Machine, or SVM, are supervised learning algorithms. Which means that it is used to find a predict function with annotated examples.
To understand how SVM works, let's take an example in a 2-dimensional space. :

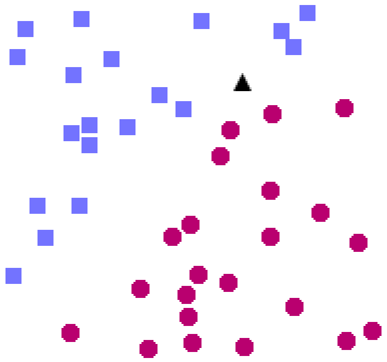


Figure 1 – Example of data to classify, red circles represent a label, and blue squares another one

Here, there is two different kind of data : red circles and blue squares. These data are annotated examples, which means they are put in the SVM with their labels. There is also a black triangle : this is a new non-annotated data, and the purpose is to know its category, to know if this is a red circle or a blue square.

In order to do that, SVM are trained on the annotated data, and from these, SVM find a linear separation between different categories. On the example, here's a border between the two classes :

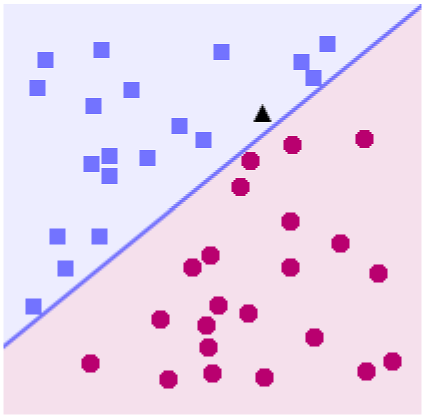


Figure 2 – Border between red circles and blue squares

However, this is not the only border possible. Usually, there's an infinity of possible boundary. Here's some other possibilities on the example :

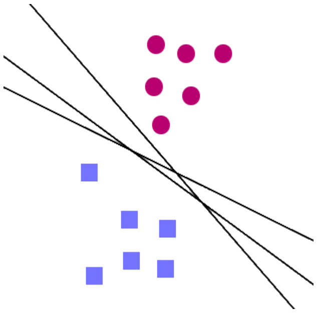


Figure 3 – Some possible borders between red circles and blue squares

SVM purpose is to find the optimal border between data. In order to do that, it maximize both area, by taking the furthest border of data. As we can see in the image below, with a non-optimal border, the pink circle, which is labeled as a red circle, is classify by the SVM as a blue square. However, with an optimal border, the pink circle is well classified.

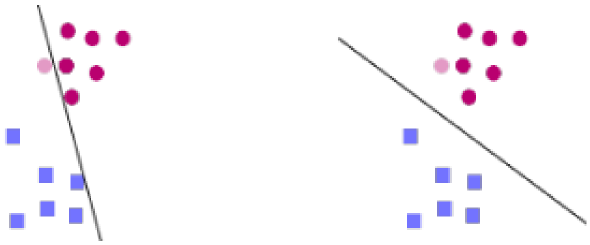


Figure 4 – One of the non-optimal borders and the optimal border

Once this border is found, SVM can determine the category of the new data, by reading its coordinate relative to the border.

On this example, the black triangle is a blue square.

The example presented here is a two dimensional problem, but usually, the problem is in bigger dimension. The purpose is also to find a linear separation, so the border will not be a straight line anymore, but a plane for a tree dimensional problem, ans a hyper-plane for a bigger dimension.

As shown, SVM works with vectors, but at the beginning, the data are images. So the issue is to represent these images with vectors by extracting features of images.

1.2 Artificial Neural Networks

To extract the needed features of an image, we start by transforming it into a three dimensional array : a dimension for width and height, and a third dimension for the colors channels RGB.
Then we can treat information by using Convolutional Neural Network, or Fully Convolutional Network.

1.2.1 The VGG-19 Model

The network on which every analysis done here are based is the VGG-19 network.
VGG-19 is a convolutional neural network that is pretrained on more than a million images from the ImageNet database. The network is composed by 19 layers, organized into 5 blocks of convolutional layers, and 3 fully connected layers, as shown in the image below :

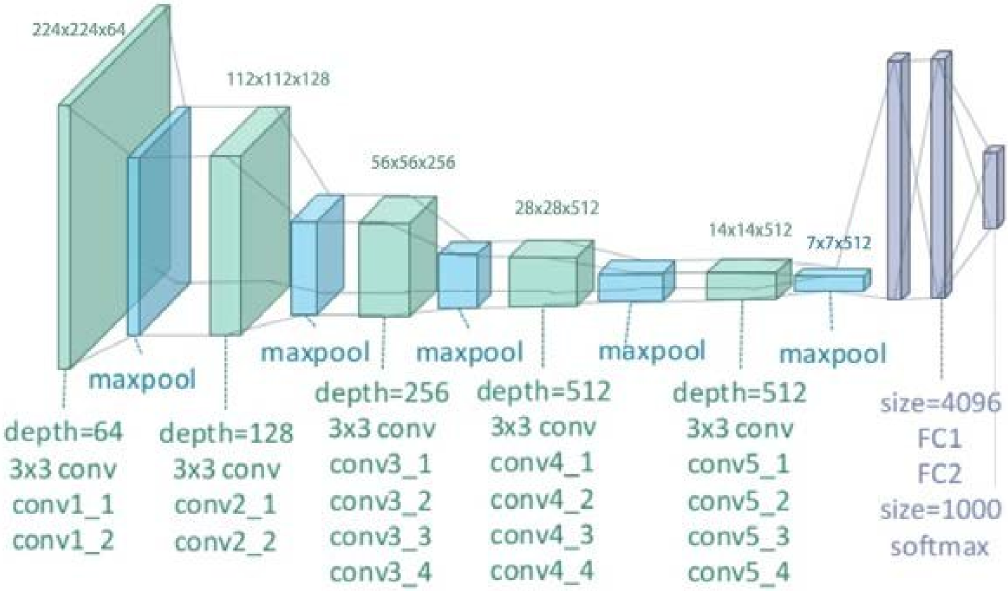


Figure 5 – Illustration of the network architecture of VGG-19 model

1.2.2 Convolutional Neural Network

Convolutional Neural Network, or CNN is an artificial neural network with multiple layers. A CNN takes an image as input, which is treated by the different layers of the CNN, and in the end, it gives a vector as output. CNN are composed by two kinds of layers, that analyze the image in two different ways.

Convolutional layers Convolutional layers are the first to analyze the input image, and it consists as a local analyze. Convolutional layers performs a convolution product between the input image and a layer-specific filter, as shown in the figure 6 :

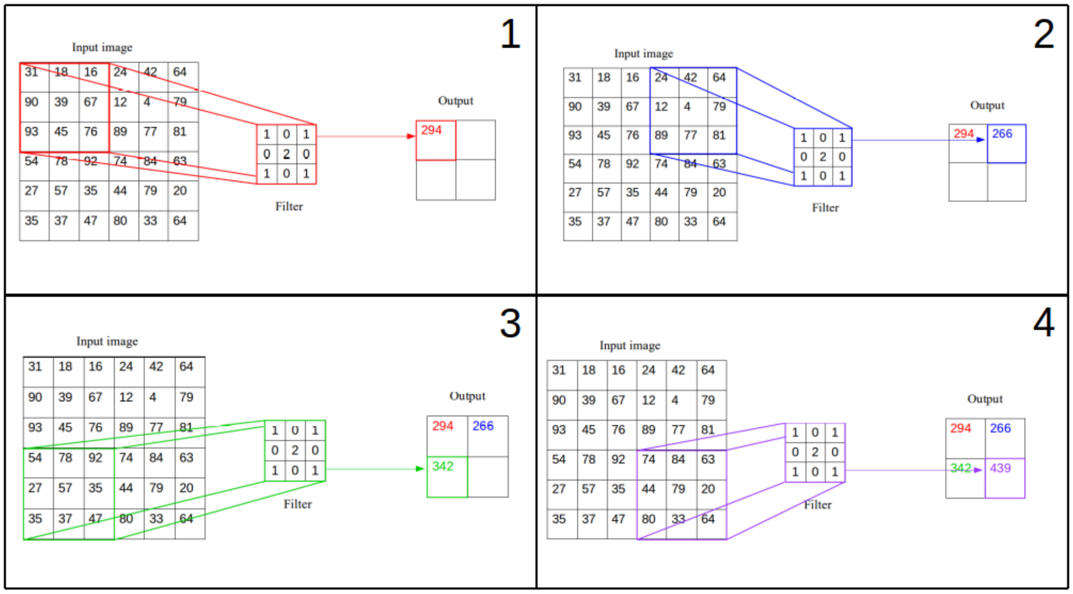


FIGURE 6 – Simple example of a convolution product between an input image and a filter

As we can see, the filter, which has a smaller size than the input image, is applied on groups of pixels which have the same size. And the output have a smaller size than the input image.
Let's notice that in this example, the input is a two dimensional array, however images are represented by a three dimensional array. In reality, the filters are also three dimensional array, with a third dimension for the colors channels RGB. The output keep the same dimensions (two dimensional array) but the result of the convoltion product will be different.
The network used here and schematized in figure 5 contains five blocks of convolutional layers, each followed by a max pooling, which consists in taking the bigger elements of each group of elements of the convolutional layers output array.

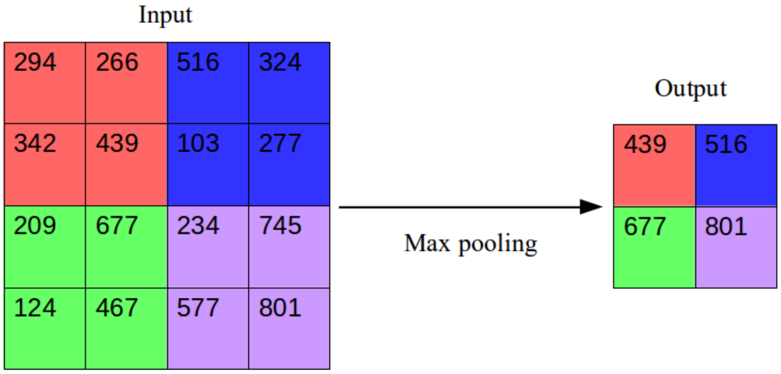


FIGURE 7 – Simple example of a max pooling

The input image is analyze by all the convolutional layer by applying every filters on the image, and the outputs are stacked together.
As the network used contains a total of 512 filters, the output of the convolutional layers is a three dimensional array with a depth of 512.

Fully connected layers Fully connected layers take the output array of convolutional layers and perform a pooling on the different elements by performing operations with all of them. The output is a one dimensional array.

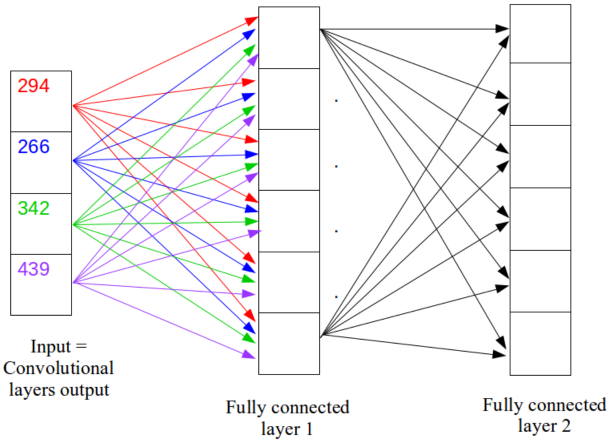


FIGURE 8 – Fully connected layers schema : simplified explanation of how they work

As shown in figure 5, the VGG-19 network contains two fully connected layers, and both of them have a 4096 size output.
While convolutional layers perform a local analyze, fully connected one perform a global analyze, which

permit to obtain a global representation of the image. This representation can then be used by the SVM in order to do classification.

We saw that the output of the second fully connected layer has a size of 4096for each image. Thus, the array given to the SVM for n images will be a 4096 x n array.
Knowing the execution time of SVM depend on the input array size, and to save time, the use of the convolutional layers output, which is 512 deep, instead of fully connected ones is possible. However, this output is a three-dimensional array, and we need a two-dimensional one to use SVM. Thereby, a pooling is necessary in order to obtain the needed dimensions. This way, the classification is faster, but less accurate because the array contains less informations.

The use of convolutional network is limited. Indeed, the input scale is fixed, here, it is 224 x 224 x 3, and it cannot be changed. Fully Convolutional Nerwork are used in order to allow the use of bigger input images, which can permits to obtain a more accurate classification.

1.2.3 Fully Convolutional Network

Fully Convolutional Network, or FCN, are obtained from CNN, and are only composed of convolutional layers. Thereby, to create a FCN from a CNN, the convolutional layers are kept, and the fully connected ones are transformed to convolutional layers. The function *to_fully_conv* performs this transformation. The result of the transformation of the VGG-19 network into a FCN is the network presented in the annex (reference). As we can see, the input scale is multiple, and the output isn't a one-dimensional array anymore. So, a max or average pooling has to be performed on the output in order to then use SVM.
As FCN can take bigger images as input, the array given to the SVM contains more informations, which should give more accurate results. However, the execution time is bigger, and the power needed to effectuate the classification can become too important.

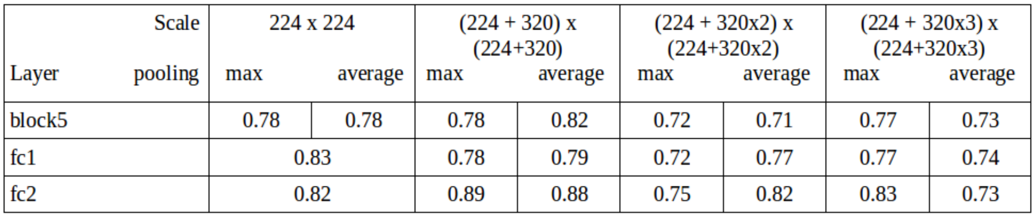
2 Experiments

2.1 Mini MIT data set

Presentation of the data set

Presentation of the results

As the data set is reduced, we could performed classification with multiple scales, from 224 x 224 to (224 + 960) x (224 + 960). The accuracy of the classification with the output of the fifth block of convolutional layers, the first and the second fully connected layers, and with a max or an average pooling are presented in the following table :



Scale	224 x 224		(224 + 320) x (224+320)		(224 + 320x2) x (224+320x2)		(224 + 320x3) x (224+320x3)	
Layer pooling	max	average	max	average	max	average	max	average
block5	0.78	0.78	0.78	0.82	0.72	0.71	0.77	0.73
fc1	0.83		0.78	0.79	0.72	0.77	0.77	0.74
fc2	0.82		0.89	0.88	0.75	0.82	0.83	0.73

Figure 9 – Results on the mini-MIT set

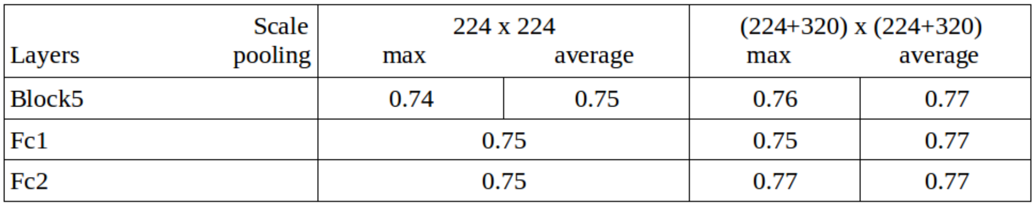
As we can see, the better results are obtain with an input scale of (224 + 320) x (224 + 320), and with an average pooling.

2.2 Chest X-ray data set

Presentation of the data set

This data set present chest x-ray images selected from pediatric patients of one to five years old from Guangzhou Women and Children Medical Center, at Guangzhou. The data set contains 5,863 chest x-ray images which can be classify into two categories : those of pneumonia patients and those of healthy patients. The data set is split into three folders, a train set, a test set and a validation set, and each of them contains two folders for the two categories. However, we didn't use the validation set for this classification.

Presentation of the results



Scale Layers pooling	224 x 224 max	 average	(224+320) x (224+320) max	 average
Block5	0.74	0.75	0.76	0.77
Fc1	0.75		0.75	0.77
Fc2	0.75		0.77	0.77

Figure 10 – Results on the chest x-ray data set

2.3 Kvasir data set

Presentation of the data set

Presentation of the results

2.4 HAM1000 data set

Presentation of the data set

The "Humain Against Machine with 10000 training images" or HAM10000 data set consist in 10015 dermatoscopic images collected from different populations, acquired and stored by different modalities. These images can be classified into seven categories, that represent the important diagnostic categories in the realm of pigmented lesions : Bowen's disease, basal cell carcinoma, benign keratosis-like lesions, dermatofibroma, melanoma, melanocytic nevi and vascular lesions.
These lesions have been confirmed through histopathology, follow-up examination, expert consensus or by in-vivo confocal microscopy.
The categories of images can be found thanks to a metadata file that gathers all these informations.

Presentation of the results

Conclusion

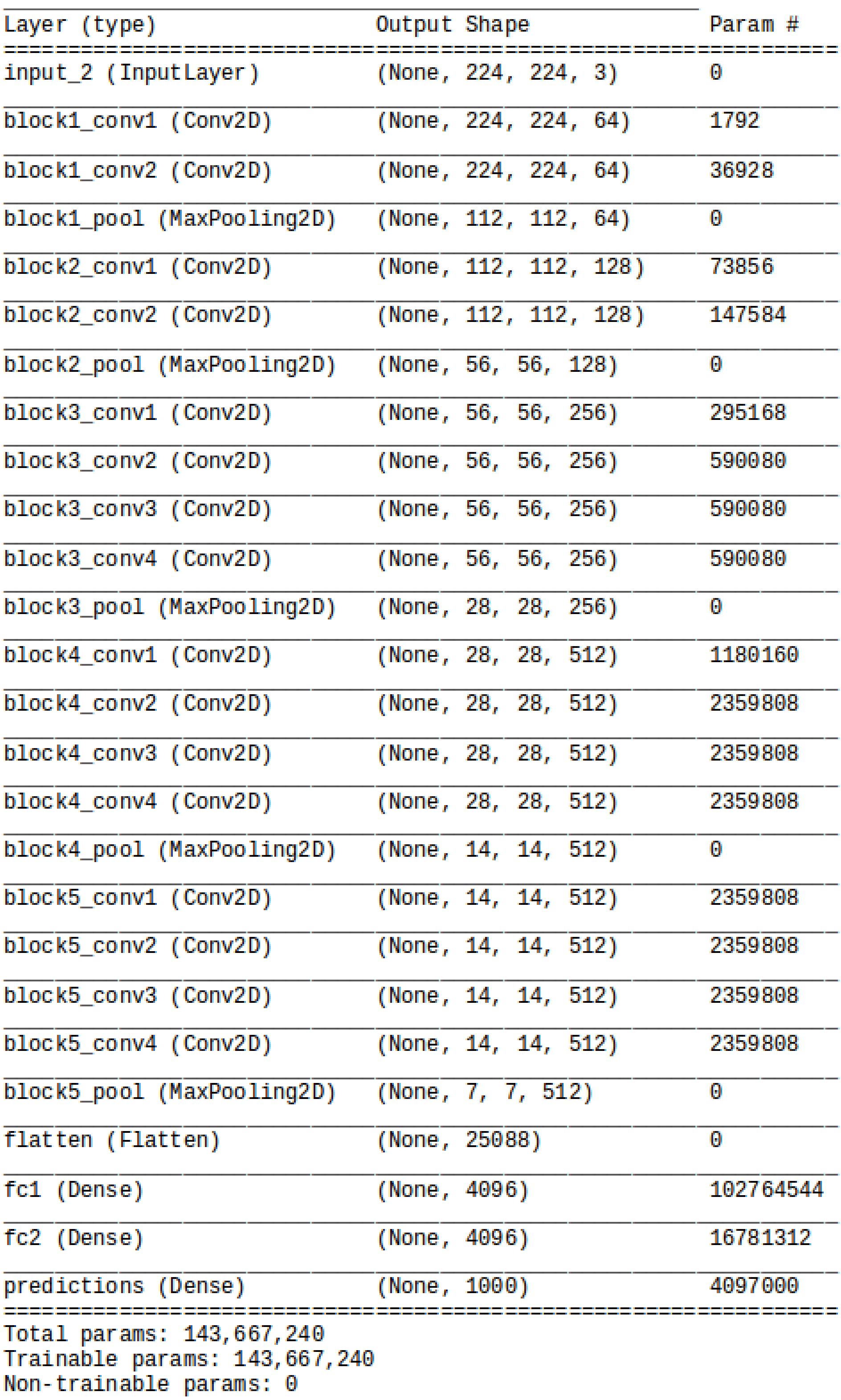
This report present classification with the use of artificial neural network. However, there's other ways to perform classification that weren't presented here. As presented in the paper [1], aggregation is another way to extracted the features of an images, with methods like Bag-Of-Words or VLAD. Moreover, we only use the VGG-19 network, but the use of other networks, as ResNet, is also possible, and could give better results.

Références

[1] R. Sicre, A. Awal, and T. Furon. Identity documents classification as an image classification problem. *ICIAP 2017 - 19th International Conference on Image Analysis and Processing*, pages 602–613, 2017.

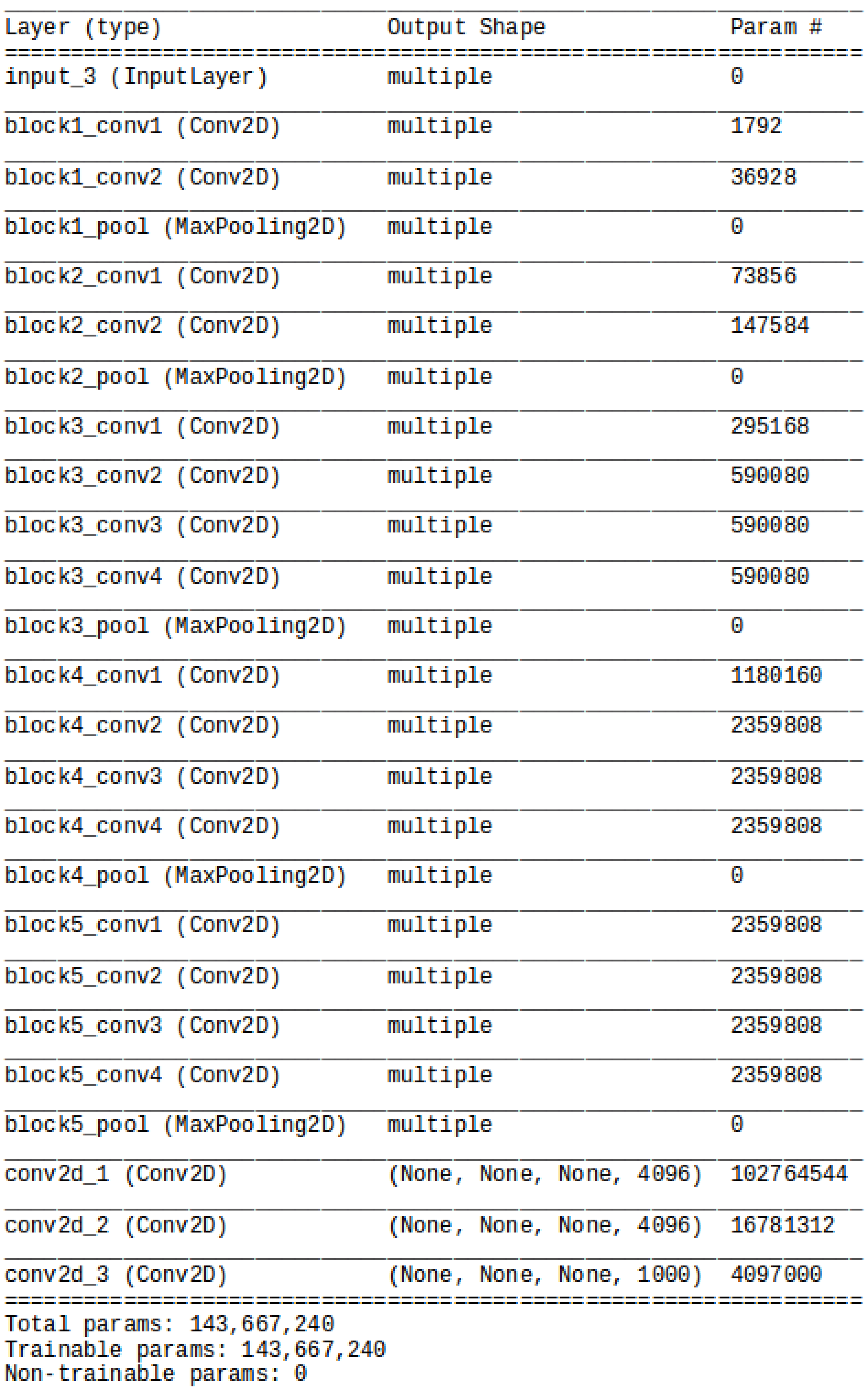
3 Appendices

3.1 Summary of the VGG19 network



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv4 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv4 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
=================================================================
Total params: 143,667,240
Trainable params: 143,667,240
Non-trainable params: 0
```

3.2 Summary of the FCN based on the VGG16 network



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_3 (InputLayer)         multiple                  0         
_________________________________________________________________
block1_conv1 (Conv2D)        multiple                  1792      
_________________________________________________________________
block1_conv2 (Conv2D)        multiple                  36928     
_________________________________________________________________
block1_pool (MaxPooling2D)   multiple                  0         
_________________________________________________________________
block2_conv1 (Conv2D)        multiple                  73856     
_________________________________________________________________
block2_conv2 (Conv2D)        multiple                  147584    
_________________________________________________________________
block2_pool (MaxPooling2D)   multiple                  0         
_________________________________________________________________
block3_conv1 (Conv2D)        multiple                  295168    
_________________________________________________________________
block3_conv2 (Conv2D)        multiple                  590080    
_________________________________________________________________
block3_conv3 (Conv2D)        multiple                  590080    
_________________________________________________________________
block3_conv4 (Conv2D)        multiple                  590080    
_________________________________________________________________
block3_pool (MaxPooling2D)   multiple                  0         
_________________________________________________________________
block4_conv1 (Conv2D)        multiple                  1180160   
_________________________________________________________________
block4_conv2 (Conv2D)        multiple                  2359808   
_________________________________________________________________
block4_conv3 (Conv2D)        multiple                  2359808   
_________________________________________________________________
block4_conv4 (Conv2D)        multiple                  2359808   
_________________________________________________________________
block4_pool (MaxPooling2D)   multiple                  0         
_________________________________________________________________
block5_conv1 (Conv2D)        multiple                  2359808   
_________________________________________________________________
block5_conv2 (Conv2D)        multiple                  2359808   
_________________________________________________________________
block5_conv3 (Conv2D)        multiple                  2359808   
_________________________________________________________________
block5_conv4 (Conv2D)        multiple                  2359808   
_________________________________________________________________
block5_pool (MaxPooling2D)   multiple                  0         
_________________________________________________________________
conv2d_1 (Conv2D)            (None, None, None, 4096)  102764544 
_________________________________________________________________
conv2d_2 (Conv2D)            (None, None, None, 4096)  16781312  
_________________________________________________________________
conv2d_3 (Conv2D)            (None, None, None, 1000)  4097000   
=================================================================
Total params: 143,667,240
Trainable params: 143,667,240
Non-trainable params: 0
```

3.3 Classification algorithm

Summary

This report is about medical images classification with deep leaning, by using Artificial Neural Network and Support Vector Machines.
This report present a way use of these methods for classification, and also the results of classification on some image set found on the internet.

Keywords : Classification, neural network, deep learning, medical images, support vector machines