Universität Osnabrück
Fachbereich Humanwissenschaften
Institute of Cognitive Science

Bachelor thesis

Relational investigation of input resolution, receptive field size and depth of VGG models.

Paribartan Humagain
975747
Bachelor's Program Cognitive Science
October 2017 - March 2021

First supervisor:	M.Sc. Mats Leon Richter Institute of Cognitive Science Osnabrück
Second supervisor:	Prof. Dr. phil. Kai-Uwe Kühnberger Institute of Cognitive Science Osnabrück

Declaration

I hereby certify that the work presented here is, to the best of my knowledge and belief, original and the result of my own investigations, except as acknowledged, and has not been submitted, either in part or whole, for a degree at this or any other university.

Paribartan Humagain, 11.02.2021, Osnabrück

Abstract

The choice of input image resolution affects the performance of the VGGNet models. But there is a lack of studies that investigate the relationship between the depth of VGGNet models, receptive field size, and input resolution on the VGGNet model's accuracy. The input image resolution for training VGGNet models chose by VGG Group is 224×224, and since it is common practice to choose 224×224 as input image resolution to train VGGNet models. But there has not been much explanation on why 224×224 and how to choose the appropriate input image resolution. We found that the accuracy of a VGGNet model is dependent on input image resolution and the receptive field size of the VGGNet models, which is dependent on various factors including the depth of a model. So to choose an appropriate input image resolution, we need to consider the receptive field size of VGGNet models.

Acknowledgement

First of all, I would like to express my gratitude towards M.Sc. Mats Leon Richter for giving me the opportunity to complete my bachelor thesis on this topic. I would like to thank Prof. Dr. phil. Kai-Uwe Kühnberger, for showing interest in my project and for being a secondary supervisor. My special thanks go to Faisal Khalil, who helped and supported me greatly to make successful completion of my bachelor thesis. I also would like to thank him for helping me in running the experiments in the University grid. Finally, I would like to thank my family and my wife Yashoda bhurtel for the constant support and encouragement during my studies and bachelor thesis.

Contents

1. Introduction

1.1. Introduction and Motivation

Convolutional Neural Networks (CNN) (LeCun et al., 1995) are obtaining very promising results, and they are being widely used in image classification e.g. Krizhevsky et al. (2012), Natural Language Processing (NLP) e.g. Kim (2014) and speech recognition e.g. Abdel-Hamid et al. (2014). Despite their popularity finding an architecture that fits the given problem is still a hard task because the accuracy of CNN highly depends on how well the architecture fits the problem at hand.

There are ongoing attempts to optimize the original architecture of Krizhevsky et al. (2012). And in an attempt of such improvement VGGNet (Simonyan, Zisserman, 2014) introduced the idea of increasing the depth of a CNN architecture to improve accuracy in prediction. It has already been shown that increasing the depth of a CNN leads to a higher computational cost and the prediction accuracy of a model decreases after a certain depth limit (He et al., 2016; Srivastava et al., 2015). Such deep CNN are also very difficult to train because of the vanishing gradient problem (Zagoruyko, Komodakis, 2016).

There are several automated methods available for optimizing the design choices like selecting through a meta-modeling procedure based on reinforcement learning (Baker et al., 2016) and Hyper-Parameters Optimization based on Genetic Algorithms (Loussaief, Abdelkrim, 2018). But these AutoML are computationally expensive.

A Convolutional Neural Network's (CNN) performance can be affected by several design factors like down-sample rate, network's depth, and receptive field size (Li et al., 2019). There are very few works published that study the impact of the receptive field on CNN's performance in isolation. In this thesis, we will investigate if we can increase CNN's performance by other means than increasing the depth of a model, which is also computationally less expensive. We will try to find the relation between the final layer receptive field of a model and input resolution so that we can increase the CNN model's performance by finding the right input resolution for a particular model. Establishing a relationship between the model's performance to model's final layer receptive field and input resolution will provide us a simple way to design and choose a VGG style CNN and this will also reduce one parameter to optimize for AutoML like those mentioned above.

1.2. Motivation and Related work

Possidente (2019) showed that Convolutional Neural Networks (CNN) with a receptive field (RF) size that results in maximal mutual information (MI) per RF learn more effectively than CNN with RFs of other sizes. Their finding suggests that finding a good enough RF is also very important for a model to perform well.

Wang et al. (2019) found out that the performance of Single Image Super-Resolution (SISR) is affected more by the change in the size of the receptive field than a change in depth. And their another important finding is that model's depth must be congruent with the receptive field size to produce improved performance.

A similar idea of adjusting the depth of CNN to its receptive field size for improvement in the model's performance was done by Koutini et al. (2019), where they tested the effect of increasing receptive field of CNN in its performance. Koutini et al. (2019) found out that for an acoustic scene classification model, increasing RF in a certain range will make a model predict well. But if we keep on increasing the RF size of the model then that leads to a bad performance of a model.
Koutini et al. (2019) conducted an experiment on acoustic data and Wang et al. (2019) used CNN for recovering a high-resolution image from its corresponding single low-resolution image.
Wu et al. (2015) concluded that training a model on higher-resolution images shows an improvement in the classification accuracy. But Shenk et al. (2020) not only showed that increasing input resolution improves the classification accuracy but also showed that only by increasing the input resolution a model doesn't give better accuracy but finding a good input resolution has a greater influence on test accuracy. In this thesis, we will explore the idea of finding a good input resolution in more depth. Tan, Le (2019) concluded in their work that it is very important to have a balance between all dimensions of network width/depth/resolution, for a model to perform well. Following their conclusions, we would like to use CNN for the image classification problem and find out the relation between three design parameters (i.e. size of the receptive field, depth of a model, and size of input image resolution).

1.3. Structure of the Thesis

This thesis will be divided into two main sections. The first section focuses on the literature survey. Firstly the basic concepts of CNN will be provided and a brief explanation of some of the basics and famous CNN will be discussed. After providing the basic ideas of CNN, VGGNet and their different versions presented in the original VGGNet paper will be explored. Additionally, different ways to manipulate VGGNet will be explained, which will be used in the experimental part of this thesis.
In the second part, experiments on VGGNets will be conducted. The main goal of the experiments is to establish a fixed relation between input resolution and receptive field size for the better performance of a model. Furthermore, it will be attempted to establish the existence of some ratio between receptive field size of a final layer of a VGGNet to input resolution, which results in a better performing model. An experiment will be conducted to find out if it is better to find a good input resolution for the model to increase the prediction accuracy than to increase the depth of a model. Several VGGNets with different depths will be examined. This helps to find out the role of depth for a better performing model with respect to input image resolution and receptive field size.

2. Basics

2.1. Convolutional Layer

Convolutional layers are important building blocks of Convolutional neural networks. The main task of convolutional layers is to capture local features from the previous layer and map their existence to a feature map. It is a mathematical operation that takes an image matrix and a filter or kernel as inputs and gives output feature maps. A schematic illustration of a convolutional operation is shown in Fig. 2.1

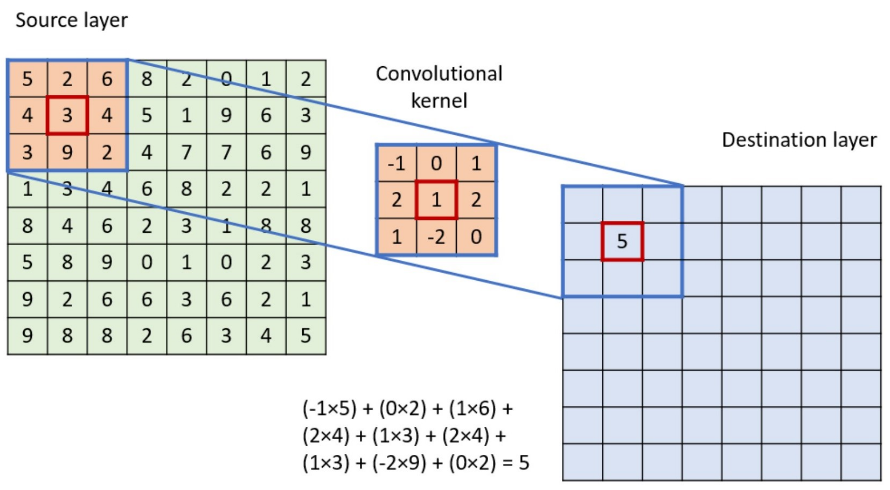


Figure 2.1: The convolutional kernel shifts over the source layer (input layer), filling the pixels in the destination layer. Image by (Podareanu et al., 2019)

2.2. Convolutional Neural Networks (CNN)

General artificial neural networks (ANN) uses general matrix multiplication in all of their layers. A deep convolutional neural network is nothing more but is a special kind of ANNs which uses convolution instead of general matrix multiplication in at least one of their layers (Goodfellow et al., 2016). David Hubel and Torsten wiesel's model (Hubel, Wiesel, 1968) describes the structure of the mammalian visual cortex, which is the inspiration for the architectural design of Convolutional Neural Networks (CNN). CNN architecture is typically deep with different types of layers like convolution, pooling layer, and fully-connected layers. With help of convolution operation, CNN extracts useful features from locally correlated data points. The activation function that follows the convolution operation helps in learning abstractions and also embeds non-linearity in the feature space. Learning of semantic differences in images is accomplished by different patterns generated by non-linearity. Sub-sampling, which is followed by the activation function helps to summarize the results (LeCun et al., 2010). CNN consists of several connections with deep architecture which helps CNN to map complicated features and functions. This special architecture of CNN is the reason for CNN to perform well in several AI-level tasks like vision and language (Ferreira, Giraldi, 2017). Nowadays researchers are focused on bringing advancements in CNNs by tuning and experimenting on different aspects of CNN like using different activation and loss functions, optimizing the parameters, and innovating the architecture of CNN. I will provide a very brief overview on other CNN than VGGNet for a historical context, which includes AlexNet, ZfNet, and GoogLeNet.

2.2.1. AlexNet

The base of CNN presented by LeCun et al. (1995) were good only at hand digit recognition tasks and didn't perform well to all classes of images. AlexNet (Krizhevsky et al., 2012) is one of the first CNN architecture to give a good performance on image classification tasks. Krizhevsky et al. (2012) enhanced the capability of CNN by making the architecture deep. There were two main problems of making an architecture deeper: first, the computational capacity at that time was not good enough to handle such a deep CNN model, and second increasing depth of a model causes the model to over-fit. The architecture of AlexNet is shown in Fig. 2.2. In the first convolutional layer max pooling is done with Local Responsive Normalization (LRN) and in this layer 96 different receptive filters are used. All the filters in this layer are 11×11 in size, whereas the max-pooling is achieved by 3×3 filters with a stride size of 2. In the second layer, all the operations are done as in the first layer but with 5×5 filters. In the third, fourth, and fifth layer 3×3 filters are used and have 384, 384, and 296 feature maps respectively. After convolutional layers, AlexNet has two fully connected(FC) layers, which is followed by soft-max at the end of the architecture.

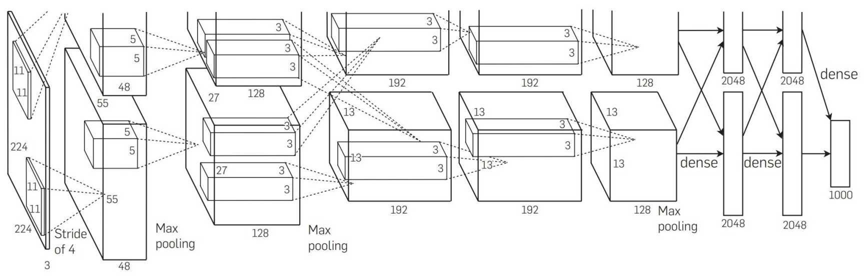


Figure 2.2: Architecture of AlexNet: AlexNet architecture consists of 5 convolutional layers, 3 max-pooling layers, 2 normalization layers, 2 fully connected layers, and 1 softmax layer. Image by (Krizhevsky et al., 2012)

2.2.2. ZfNet

Zeiler, Fergus (2014) modified AlexNet architecture to get better accuracy in image classification tasks by altering some parameters. ZfNet uses 7×7 filters instead of 11×11 sized filters used by AlexNet. By using a smaller filter size Zeiler, Fergus (2014) reduced the number of weights and improved overall classification accuracy. Using a smaller filter size helps ZfNet to retain a lot of original pixel information in input volume.

2.2.3. GoogLeNet

GoogLeNet (Szegedy et al., 2014) won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) in 2014. It introduced a new module called the inception module which solves the problem of computational expense and over-fitting by dimensional reduction with stacked 1×1 convolutions. The architecture of the Inception module is shown in Fig. 2.3. GoogLeNet, which is 22 layers deep including 27 pooling layers, has 9 inception modules stacked linearly in total.

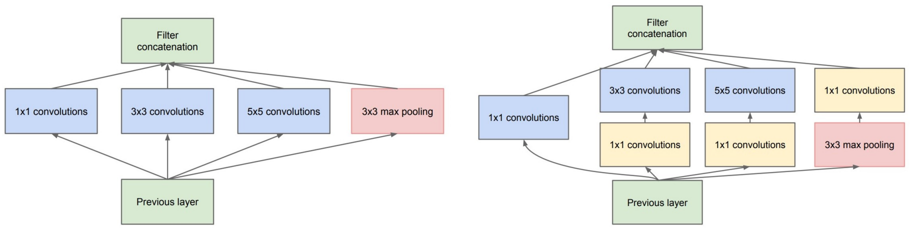


Figure 2.3: Inception module. Image by (Szegedy et al., 2014)

GoogLeNet also introduced another change to previously standard CNN by replacing fully-connected layers at the end with a simple global average pooling. This helps in reducing the total number of parameters. Due to the use of large network width and depth, GoogLeNet can remove the Fully connected (FC) layer without affecting the accuracy.

2.2.4. VGGNet

VGGNet (Simonyan, Zisserman, 2014) was the first runner-up of the ILSVRC (ImageNet Large Scale Visual Recognition Competition) in 2014 in the classification. VGGNet is developed by the Visual Geometry Group(VGG) from the University of Oxford. VGG put forward the idea of increasing the depth of the architecture for better accuracy. As compared to other CNN like AlexNet (the winner in 2012) and ZFnet (the winner in 2013), VGGNet used a smaller filter size. VGGNet used 3×3 filter size, whereas AlexNet (Krizhevsky et al., 2012) used 11x11 and 5x5 filters at the first two layers and Zfnet (Zeiler, Fergus, 2014) used 7x7 filters. By reducing the filter size and increasing the depth, VGGNet reduces the number of parameters and that results in the reduction of the network's complexity. In VGGNet input images are passed through a stack of convolutional (Conv.) layers. All configurations use 3×3 filter size in all of its convolutional layers except for one configuration where they used 1×1 convolution filters as well. After the VGG group used small convolutional kernels (3×3 units), it has become now the standard kernel size in modern papers. VGGNet models use the rectified linear unit (ReLU) as an activation function, which is the default activation function in all big deep learning frameworks, for e.g. Keras, Tensorflow, and PyTorch. VGGNet also introduced the pyramidal shape of the network shown in Fig. 2.4, which is considered standard nowadays for deep neural architectures.

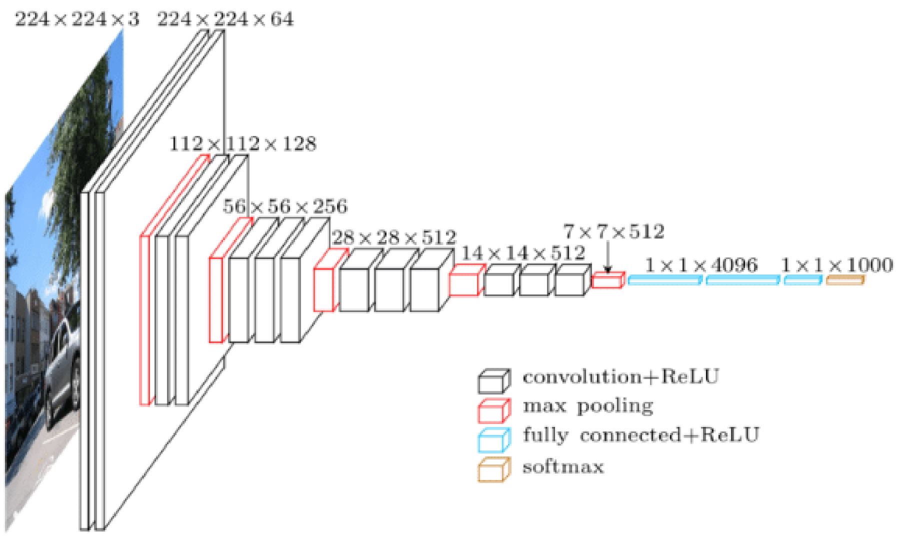


Figure 2.4: The VGG16 architecture. Note the pyramidal shape with decreasing size of filters, while the size of the filter stacks increase. Image by (Simonyan, Zisserman, 2014)

The configuration of the VGGNet used Simonyan, Zisserman (2014) is depicted in Table 1. From A to E the depth of the configuration increases. The configuration denoted by A has the 11 layers with the lowest number of layers, with 8 convolutional layer and 3 Fully connected layers. The configuration denoted by E has the 19 layers with the highest number of layers. The bold written layers denotes the added layers. The convolutional layer parameters are denoted as ”conv<receptive field size> - <number of channels>”

Table 1: **ConvNet configurations**. (Simonyan, Zisserman, 2014)

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layer	11 weight layer	13 weight layer	16 weight layer	16 weight layer	19 weight layer
input (224×224 RGB image)					
conv3-64	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64
	LRN	**conv3-64**	conv3-64	conv3-64	conv3-64
maxpool					
conv3-128	conv3-128	conv3-128	conv3-128	conv3-128	conv3-128
		conv3-128	conv3-128	conv3-128	conv3-128
maxpool					
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
			conv1-256	**conv3-256**	conv3-256
					conv3-256
maxpool					
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
			conv1-512	**conv3-512**	conv3-512
					conv3-512
maxpool					
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
			conv1-512	**conv3-512**	conv3-512
					conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

In this work, the VGGNet shown in table 1 will be modified by reducing the number of channels in convolutional layers. The aim of this work is not to find the higher accuracy of a model by increasing the number of parameters; rather, by changing the input resolution and receptive field size. So reducing the number of channels will not have a bad influence on the experiments; however, this will also help to run the experiments in less time and with a less computational cost. The modified VGGNet is shown in Table 2.

Table 2: **ConvNet configurations with reduced number of channels**

ConvNet Configuration			
VGG11_XXS	VGG13_XXS	VGG16_XXS	VGG19_XXS
11 weight layer	13 weight layer	16 weight layer	19 weight layer
input (different resolution RGB image)			
conv3-8	conv3-8 **conv3-8**	conv3-8 conv3-8	conv3-8 conv3-8
maxpool			
conv3-16	conv3-16 **conv3-16**	conv3-16 conv3-16	conv3-16 conv3-16
maxpool			
conv3-32 conv3-32	conv3-32 conv3-32	conv3-32 conv3-32 **conv3-32**	conv3-32 conv3-32 conv3-32 **conv3-32**
maxpool			
conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64 **conv3-64**	conv3-64 conv3-64 conv3-64 **conv3-64**
maxpool			
conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64 **conv3-64**	conv3-64 conv3-64 conv3-64 **conv3-64**
maxpool			
FC-4096			
FC-4096			
FC-1000			
soft-max			

2.3. Why VGGNet

First of all the VGGNet is very simplistic in structure and has become the very definition of a simple CNN structure. VGGNet has set standards like 3×3 convolution which are still standard in today's context. Although GoogLeNet was responsible for setting a new state-of-the-art for classification and detection in the ILSVRC, it complicates the computation of receptive fields and their respective receptive field sizes. This is due to the non-sequential structure of GoogLeNet. We are interested in VGGNet due to its simple architecture than GoogLeNet. GoogLeNet's heterogeneous topology needs to be customized from module to module, which makes GoogLeNet a complex architecture to design. The bottleneck representation of GoogLeNet drastically reduces the feature space in the following layer and that could sometimes cause loss of useful information (Khan et al., 2020). As VGGNet is very simple in structure, there will be fewer potential influences to control for, and this makes VGGNet a very good model Organism to study for the effects. In Simonyan, Zisserman (2014), the authors have also mentioned the incorporation of

1×1 convolutional layers in the VGGNet, which allows us to alter the resolution of a model without changing the weights of a model.

2.4. Receptive Field

A typical Convolutional Neural Networks(CNN) are composed of several convolutions, pooling, and fully connected layers. Within several layers of the CNN, there are a set of units or neurons which gets inputs from corresponding units from a similar subsection in the earlier layer. Neurons learns patterns like edges, lines, and small details that constitutes in creating images as each neurons within a CNN is accountable for a designated region of the input data. A stimulation in a limited region of certain layers of CNN leads to the response of particular sensory neurons/units in the next layer (Gál et al., 2004), this defined region of space portion of space or spatial construct containing units that provide input to a set of units within a corresponding layer is called a receptive field (RF) (Alake, 2020).

2.5. Mathematics of Receptive Field

We can compute the receptive field size of a VGGNet analytically as it is a neural network with a sequential structure. There are a number of variables we need to take into consideration to calculate the size of the receptive field in each layer, which is the receptive field size of previous layer $r_{(l-1)}$, kernel size of current layer k_l and stride size which is denoted by s_i for stride size of layer i.
The mathematical formula for calculating the receptive field for all layers $l > 0$ in the convolutional part of the network is given in equation 1.

$$r_l = r_{l-1} + ((k_l - 2) * \sum_{i=o}^{l-1} s_i) \tag{1}$$

As a single position of first layer input layers exactly maps to a single pixel on the input image, the receptive field size of the input layer is 1(i.e $r_0 = 1$). If we go one step deep in the network(i.e first convolutional layer), the receptive field size(r_1) with stride size of 1 is equal to the size of the kernel. If we Stack more convolutional layers in the network with kernel size $k_l \neq 1$ then this will cause expansion of receptive field size for next layer l+1 because the filter integrates information of an area of input feature map into a single output position.

2.6. Modifying the Receptive Field of a CNN

For testing an effect of the Receptive field of a VGGNet model on models performance receptive field size of VGGNet will be modified. There are various ways to change the receptive field size of CNN. According to equation 1, if we change the filter sizes k (by dilating the convolutional kernel or by replacing kernels with different size) or the strides s of different layers or change the depth of network by change the total number of layers(l), we can change the receptive field size. Fig.2.5 shows the relation between kernel size and Receptive field in each layer.

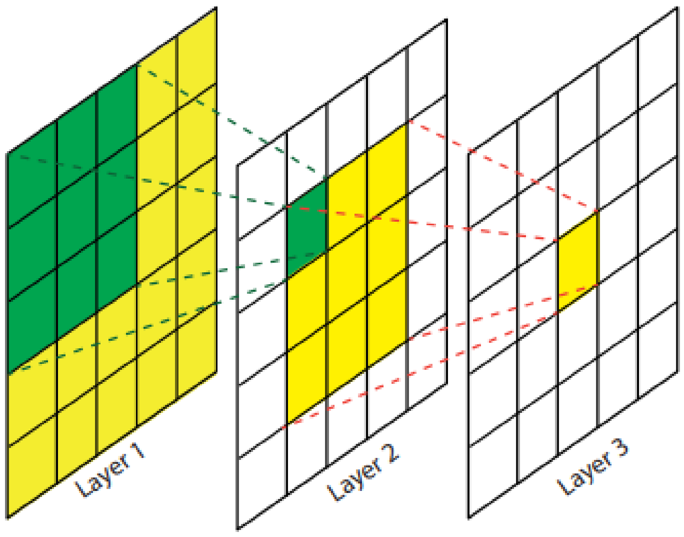


Figure 2.5: The receptive field of each convolution layer with a 3×3 kernel. The green area denotes the receptive field of one pixel in Layer 2, and the yellow area denotes the receptive field of one pixel in Layer 3. Image by (Lin et al., 2017)

If we use different kernel sizes for e.g 7 ×7 instead of 3 ×3 the receptive field will increase in each layer. As 3×3 kernel size has become standard kernel size, we will not simply replace the 3×3 kernel size of VGGNets for our experiments, but we will change kernel size by the method explained in 2.6.3.
We use the term “the receptive field of a VGGNet model” to denote the receptive field of the final layer of the model.

2.6.1. Changing Depth

Adding additional layers i.e. increasing the depth of VGGNet will also increase the Receptive field of a model for e.g. VGG16 has a final layer receptive field size of 196 and VGG19 has 252. For a fair and simple comparison, we will not compare the performance of different models with different receptive field sizes, which is the result of increasing the depth of a model. We will definitely conduct experiments on VGGNets with different depths, but we will only compare VGGNets with the same depth.

2.6.2. Changing Strides

The number of pixels a filter moves over the input image per step is strides. In all the models by Simonyan, Zisserman (2014) they fixed stride to 1 for all the convolutional layers and 2 for max polling layers. If we increase the size of stride in convolutional layers or in pooling layers then we will also increase the receptive field size in the following layer of a network. In this thesis, we will not be changing the size of the stride for changing the receptive field size. But we will follow the idea of Wang et al. (2019), and increase receptive field size by using dilated convolutional layers.

2.6.3. Dilated Convolutional Layers

Wang et al. (2019) used Dilated convolutional layers to increase the receptive size of a model without significantly increasing the number of learning parameters. The idea of dilated convolution(or *Atrous convolution*) was originally used for wavelet decomposition (Holschneider et al., 1990). In a dilated network, zeros (holes) are inserted between the regular filter samples, as shown in the Fig. 2.6. The convolutional filter (kernel) obtained by such dilation is sparse when compared to a standard convolutional filter. Dilated convolution not only increases the receptive field size of the network but also keeps the size of feature maps unchanged. With help of dilation, we will make the receptive field size larger without increasing the number of arguments and computational cost (Li et al., 2019). We will modify the VGGNet models shown in Table 2 by using the dilated convolutions with different dilation rates. The receptive field size of VGGNets after dilation is depicted in Table 3

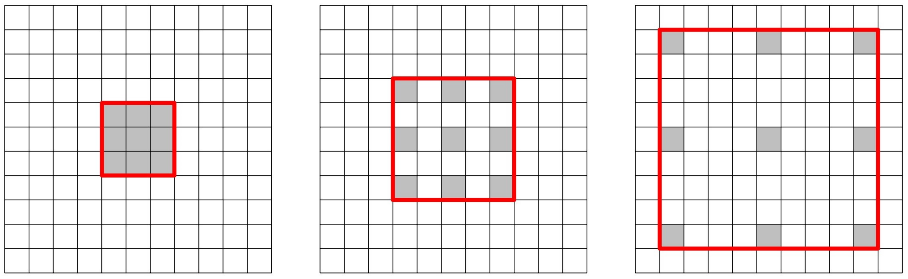


Figure 2.6: 2-D Dilated convolutions with a kernel size of 3 ×3, with dilation rate 1 (on the left-side), dilation rate 2 (in the middle) and dilation rate 4 (on the right-side)

Table 3: **The receptive field size of VGGNet models with different dilation rate.**

	VGG11_XXS	VGG13_XXS	VGG16_XXS	VGG19_XXS
Dilation rate 1	134	140	196	252
Dilation rate 2	252	264	376	488
Dilation rate 3	370	388	556	724
Dilation rate 4	488	512	736	960
Dilation rate 5	606	636	916	1196
Dilation rate 6	724	760	1096	1432

3. Experiments

The following section will describe the experimental setup of this thesis. There are 3 experimental sections describing the different experimental designs. I will first discuss the outlines of the experiments by looking at data sets and the metrics. After that, we will look into the methodology of the experiments, the exact configurations, and the objective behind the configurations will be explained in more detail.

3.1. Data Sets

We will conduct all the experiments using the CIFAR10 data sets. In this data set, there are 10 different classes of images with each class containing 600 images (500 training, 100 validation). The data set consists of various real-world pictures of different categories like an airplane, automobile, bird, etc, and are 32×32 color images. The classes are completely mutually exclusive. A sample of the data set is shown in Fig. 3.1.

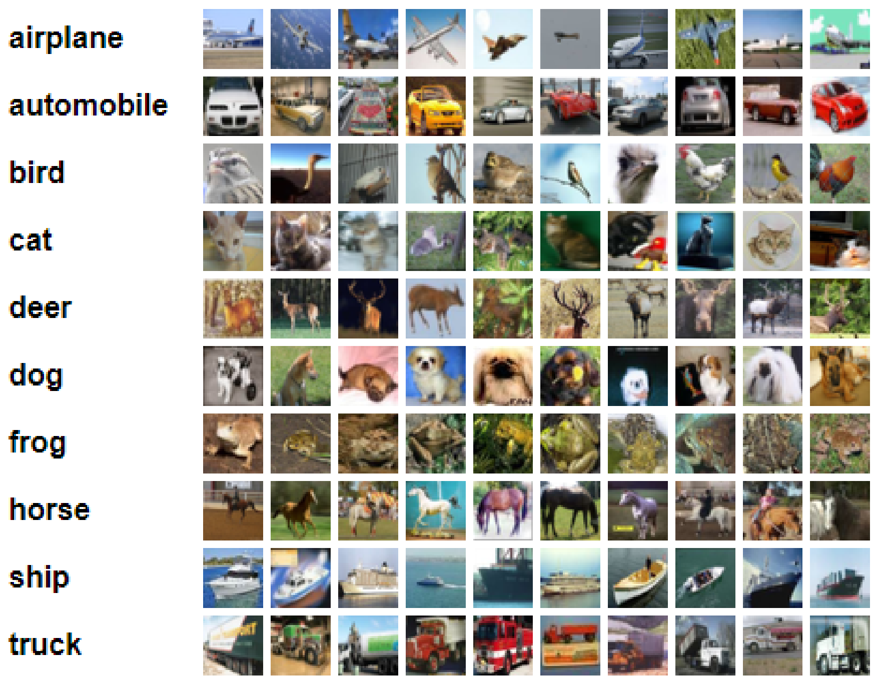


Figure 3.1: Randomly selected images from CIFAR10 data set. Image by (Krizhevsky et al., 2012)

3.2. Metrics

In this section, we will elaborate on the metric that we will use to visualize and analyze the predictive quality of trained models. For our purpose, we will calculate the test accuracy of the model and saturation.

3.2.1. Accuracy

This is one of the widely used metrics to estimate the quality of the classifier. This metric gives us a simple and easy way of interpreting the estimation of the predictive performance. The accuracy can be defined as the percentage of correctly classified instances which is calculated using the following formula:

$$Accuracy = \frac{TP}{N} * 100\%$$

Where TP denotes the number of true positives i.e. number of instances where models label the given input correctly and N denoted the total number of data points. We will look at top-1 accuracy for a comparison. The prediction of a model is generally given in percentage. For example, if a model has to predict a picture of a cat then its output will be 60% cat and 5% Dog, 4% cow, 11% donkey, 16% ship. Here the value TP will be increased by 1 only if the cat has the highest predictive value of all.

3.2.2. Layer Saturation

Layer saturation is a simple and computationally inexpensive method proposed Shenk et al. (2020) for analyzing the information processing in a neural network. The saturation layer indicates what proportion of spatial dimension is occupied by the information in a layer l. With the help of saturation layer calculation, we can find what fraction is useful in output space Shenk et al. (2020). We can see how an input resolution size with respect to the receptive field of a model is affecting the inference process between the layers by examining the distribution of saturation in a model. Layer saturation is calculated using the following formula.

$$s_l = \frac{dim\, E_l^k}{dim\, Z_l}$$

Where S_l indicates layer saturation, E denotes eigenspace, k∈{0, ..., n} at training time, l indicates a layer of the Network and Z_l is the mean layer output vector. Z_l is defined as:

$$Z_l = \sum_{(i=1)}^{n} \frac{z_{(l,i)}}{n}$$

where n is total number of training samples. When at least 3 successive layers in a feed-forward network have average saturation at least 50% lower relative to the rest of the network then the saturation pattern is defined to have a tail. This definition of the tail is not perfect, but this definition is more accurate than pure visual observation. The tail pattern forms when there is a mismatch between input resolution and network architecture (Shenk et al., 2020). We can observe how tail-saturation-pattern behaves when we change the input resolution to VGG models. We want to see if the length of a tail pattern is shortened when the input resolution matches the receptive field size of a model. The saturation at each layer can be calculated during the model's training with almost no computational expense.

3.3. Training Setup and Parameters

All the models used in all the experiment is trained with the same training setup. The hyper-parameter setting used in all experiments is depicted in Table 4.

Table 4: **Hyper-parameters common to each of experiments in section 3.**

Parameters	**Values**
Epochs	20
Batch size	32
Optimizer	ADAM
ADAM : beta1	0.9
ADAM : beta2	0.999
ADAM : epsilon	le-8
ADAM :learning rate	0.001

3.4. Experiment 1

The first experiment is designed to investigate whether less deep VGGNet can predict better or equally good, with appropriate input resolution than deeper VGGNet.
We train VGG11, VGG13, VGG16, VGG19 with input resolution 32 which is the smallest resolution we can have as the original data-sets consists of 32×32 resolution images. Reducing the resolution further low will drop image classification performance significantly (Pei et al., 2018), so we will not go below 32. The models are trained 3 times and the VGG11 model is again trained 3 times with input resolution 92. We then calculate the mean test accuracy of a model by taking an average of all three runs. We will also calculate the receptive field size of each VGGNet model during the training process.

3.5. Experiment 2

The second experiment is designed to test the effect of changing the receptive field of VGGNet on prediction accuracy.
We train VGG11_XXS, VGG13_XXS, VGG16_XXS, and VGG19_XXS models, the configuration to which are shown in Table 2. For this experiment, we change the receptive field size of each model with the method described in the section2.6.3. Each model is dilated with dilation rates 1, 2, 3, 4, 5, and 6. We train 24 models and each model are trained 3 times and in total, we have 72 runs. We pick the input resolution for all models in such a way so that we can see how the models behave when the input resolution below and above the receptive field size of a model. The receptive field size for each model is shown in Table 3. The input resolutions used for VGG11_XXS, VGG13_XXS, VGG16_XXS, and VGG19_XXS models are 370, 388, 556, and 724 respectively, which are the receptive field size of each model at dilation rate 3. The receptive field size of each model is also calculated during model training. As the receptive field size grows exponentially we will not be able to see explicitly which receptive filed size is suitable for the given input resolution. That is why we

also tested some dilated models with different input resolutions as well. We trained all dilated version of VGG13_XXS with input resolution 224.

3.6. Experiment 3

The third experiment is developed in a way to find out the influence of changing input resolution on prediction accuracy on a model.
We train VGG11_XXS, VGG13_XXS, VGG16_XXS, and VGG19_XXS models with different input resolutions. VGG11_XXS is trained with 34 to 274 with each new resolution 20 more than the previous one. We are investigating how the model's prediction behaves when the input resolution size is near to the receptive field of VGGNet models. To investigate this we first calculated the receptive field of each model and taking the same size input resolution as the middle point we selected other input resolution based on the middle point. The selected input resolutions for each model are depicted in Table 5.

Table 5: **Different input resolution for VGGNet models. The input resolutions in bold are of the same size of the receptive field of respective models.**

VGG11_XXS	VGG13_XXS	VGG16_XXS	VGG19_XXS
34	32	32	32
54	40	56	124
74	60	76	132
94	80	96	152
114	100	116	172
124	120	136	180
134	**140**	146	192
154	160	156	212
174	180	176	232
194	200	**196**	**252**
214	220	216	272
234	240	236	292
254	260	256	312
274	280	276	332
724	300	296	352
-	724	724	724

We would also like to find out if there exists any kind of optimal ratio between the receptive field of a VGGNet model and the size of input resolution. To test this we will calculate the ratio of receptive field size(RF) of a model to the size of input resolution(IR) with the highest accuracy using the formula 2. We will calculate this ratio for all three runs for each of the models and then take a mean. With this ratio in hand, we can find an input resolution which gives us the highest accuracy for different VGGNet model. In this experiment, each model is trained with 10 different input resolutions. Each model with different resolutions was trained 3 times, and we take a mean of 3 runs. The experiment has a total of 168 runs.

$$Optimalratio = \frac{RF}{IR} \tag{2}$$

4. Analysis

4.1. Experiment 1

This experiment is showcasing the main purpose of this thesis. This shows us, what can be achieved if we can find a relation between receptive field size, depth, and input resolution of VGGNet models.

4.1.1. Effect on Model's Test Accuracy

As the figure 4.1 shows us that VGG11 with input resolution 92 outperforms all other models with input resolution 32. Interestingly the VGG13 and VGG16 have a higher test accuracy than the VGG11 with input resolution 32, but VGG19 has the lowest test accuracy. This result shows us that making a network deep doesn't always increase the model's accuracy as VGG19 performs worst in comparison to other less deep models. If input resolution is not matching enough, making a model deeper doesn't improve the model's accuracy and if we can find good enough input resolution for a less deep model it can outperform deeper models with mismatching input resolutions.

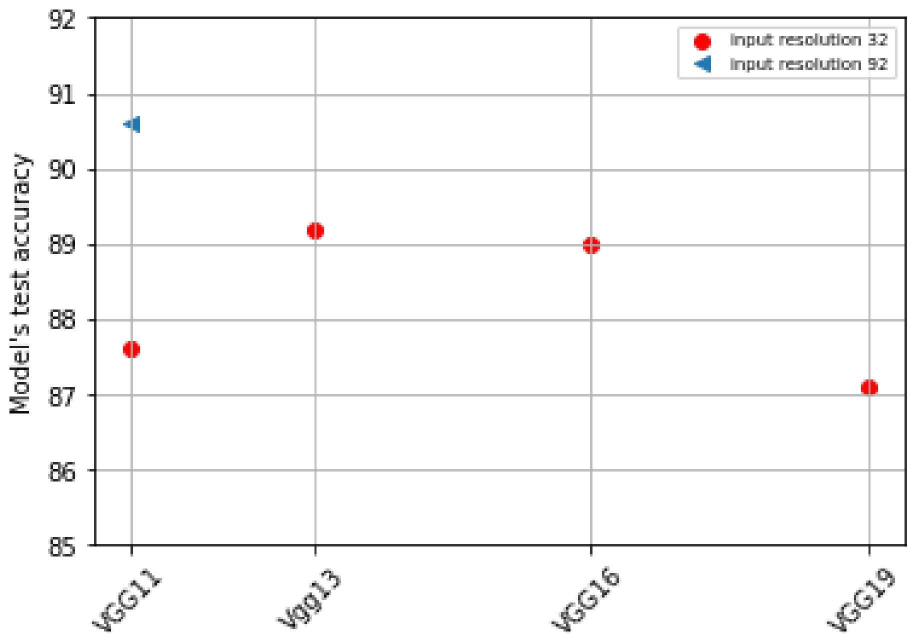


Figure 4.1: VGG11 trained with input resolution (32 and 92) and VGG13, VGG16 and VGG19 trained with input resolution 32 indicating that shallower VGGNet models, given an appropriate input resolution, can outperform deeper VGGNet model

4.1.2. Effect on Saturation Tails

Having an appropriate input resolution is key to removing a tail pattern (Shenk et al., 2020). Fig.4.2 shows us that with input resolution size 92 tail pattern is removed in VGG11_XXS, and we can also see that with the tail removed model's test accuracy is also improved as shown in Fig.4.1. Another way to remove the tail pattern is by reducing the depth of a model (Shenk et al., 2020), which can be seen in Fig.4.2.

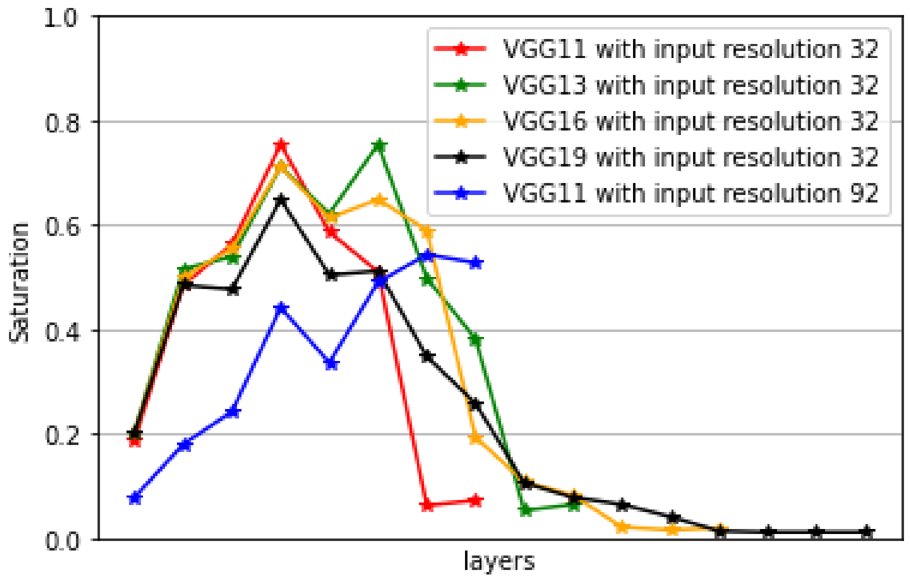


Figure 4.2: VGG 11 trained with input resolution (32 and 92) and VGG13, VGG16 and VGG19 trained with input resolution 32. Graph shows removed tail pattern with reduced model's depth

4.2. Experiment 2

This experiment shows us the effect of changing the receptive field size of a model without changing the depth.

4.2.1. Effect on Model's Test Accuracy

In Fig. 4.3, we can see that the VGG11_XXS model has the lowest mean prediction accuracy (79.6%) when a model is dilated with a dilation rate of 1. As the receptive field size increases the mean prediction accuracy increases as well until it is equal to the size of input resolution which is 370. When the receptive field size of VGG11_XXS becomes much larger than the input resolution size, then the mean prediction accuracy decreases.

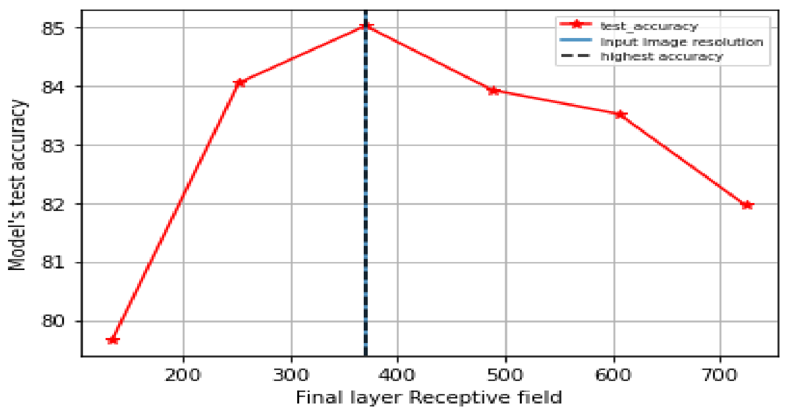
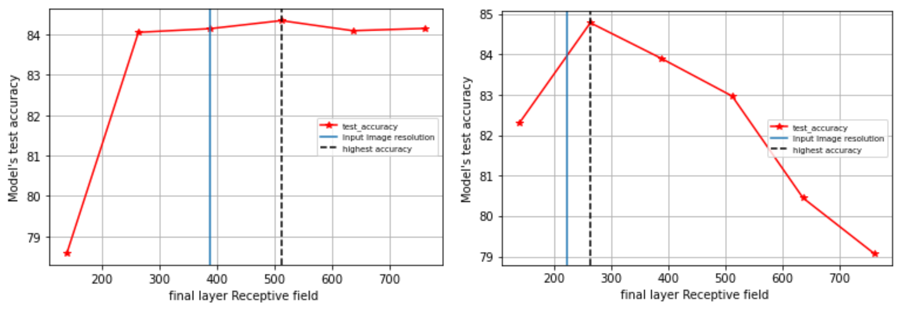


Figure 4.3: When the input image resolution size matches the size of Receptive field of VGG11_XXS then it has the optimal performance. VGG11_XXS (dilated with dilation rate 1, 2, 3, 4, 5, 6) trained with input resolution 370. Dilated VGG11_XXS has receptive field size 134, 252, 370, 488, 606, 724 respectively

In the case of the VGG13_XXS model with input resolution 388, it has the lowest mean prediction rate (78.59%) when it is dilated with dilation rate 1. VGG13_XXS also shows almost the same behavior as VGG11_XXS. The mean prediction accuracy increases as we increase the receptive field size of a model. We are not able to see the falling prediction accuracy when the receptive field size is larger than the input resolution. This effect is not visible with input resolution 388 because the remaining receptive field size is not big enough to cause a reduction in mean prediction accuracy. We can see the downhill movement of prediction accuracy by changing the input resolution to 224.



(a) (b)

Figure 4.4: Input resolution size closer to receptive field size of a VGG13_XXS gives optimal performance on accuracy. VGG13_XXS models with different receptive field size and fixed input resolution (a) VGG13_XXS with input resolution 388; (b) DVGG13_XXS with input resolution 224;

A similar pattern can be seen in VGG16_XXS with dilation 1, 2, 3, 4, 5, 6. As the receptive field size approaches the size of input resolution, the prediction accuracy increases as well. When the receptive field size grows much bigger than the input resolution then mean prediction accuracy starts decreasing.

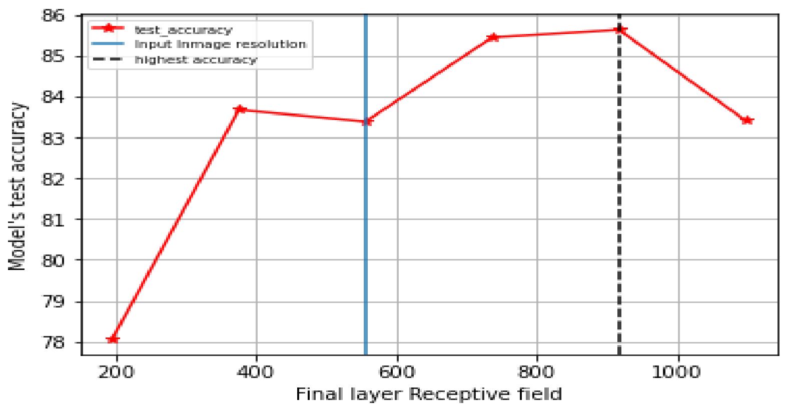
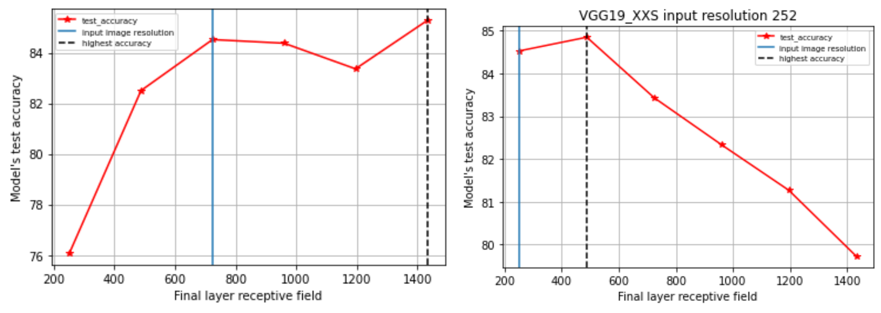


Figure 4.5: Input resolution size closer to receptive field size of a VGG16_XXS gives optimal performance on accuracy.VGG16_XXS (dilated with dilation rate 1, 2, 3, 4, 5, 6) trained with input resolution 556. Dilated VGG16_XXS has receptive field size 196, 376, 556, 736, 916, 1096 respectively

VGG19_XXS also shows analogous results when trained with input resolution 724. With input resolution size 724 VGG19_XXS has the highest mean prediction accuracy when the model is dilated with a dilation rate of 6 (which has receptive field size 1432. Same as in VGG13_XXS we can't see the downhill movement of the model's accuracy with input resolution 724, but this can be seen with input resolution 252.



(a) (b)

Figure 4.6: Input resolution size closer to receptive field size of a VGG19_XXS gives optimal performance on accuracy. (a) Dilated VGG19_XXS with dilation rate 1,2,3,4,5 and 6 trained with input resolution 724; (b) Dilated VGG19_XXS with dilation rate 1, 2, 3, 4, 5 and 6 trained with input resolution 252

These results show us that for a fixed input resolution size changing the receptive field size increases the model's test accuracy for a certain range and starts decreasing afterward. The highest accuracy is when the input resolution size is either equal or smaller than the receptive field size of a model. Comparable results are also shown by Koutini et al. (2019), where they tested the effect of increasing receptive field of CNN in its performance on acoustic scene classification and showed that increasing the receptive field size of a model increases the model's prediction accuracy but for a very large receptive field this effect is not valid. We found out that too large and too small receptive field causes a decrease in models performance, a similar result was obtained by Wang et al. (2019) on L10 CNN model for Single Image Super-Resolution (SISR) tasks.

4.2.2. Effect on Saturation Tails

Fig. 4.7 shows that when the receptive field size of a model is much larger than the input image resolution size we can see the tail pattern at the end layers. And if the receptive field size of a model is much smaller than the input image resolution size, then the tail pattern is seen at the beginning layers but with the input image resolution size closer to the receptive field size of a model the tail pattern can be removed. This indicated that if we choose the input resolution size that is closer to the receptive field size of a model then we can increase the prediction accuracy of a model. Choosing a very large or a very small input resolution size with respect to the receptive field size of a model causes the tail pattern in a saturation plot and low prediction accuracy as well.

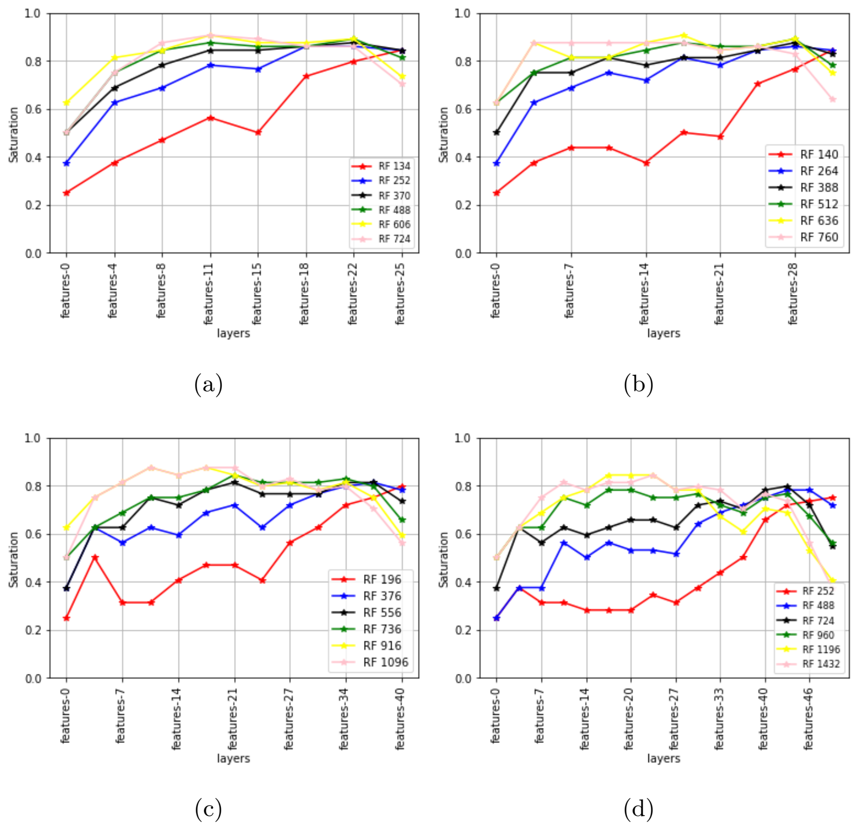


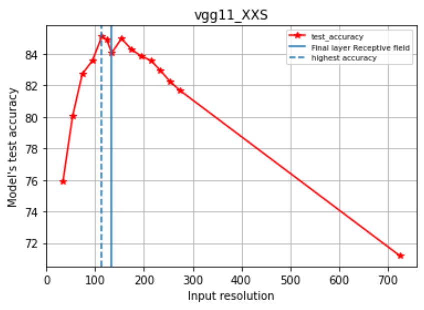
Figure 4.7: Saturation plot of Different VGGNet models dilated with dilation rate 1, 2, 3, 4, 5 and 6 and trained with different input image resolution size: (a) Dilated VGG11_XXS with input resolution size 370; (b) Dilated VGG13_XXS with input resolution size 388; (c) Dilated VGG16_XXS with input resolution size 556; and (d) Dilated VGG19_XXS with input resolution size 724.

4.3. Experiment 3

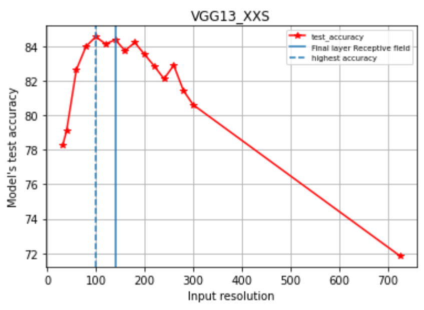
Different input resolution was given to training a model to see the effect of changing the input resolution on the model's performance. Having the same model trained on different input resolutions and plotting the model's performance against different input resolution gives us an insight into how changing the input resolution changes the model's performance. If we also plot the final layer receptive field size(RF) of a model then we can also see how an input resolution around the RF changes a model's performance.

4.3.1. Effect on Model's Test Accuracy

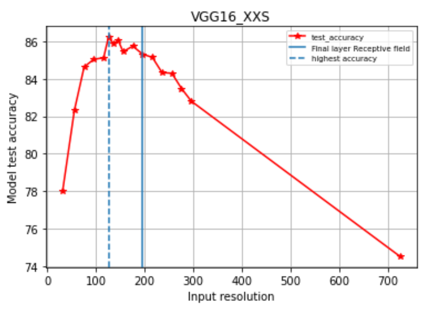
VGG11_XXS has the lowest mean test accuracy when the input resolution is 34. The highest mean test accuracy is at 114 (85.1%), which is slightly smaller than the receptive field size (134) of VGG11_XXS. The mean test accuracy decreases after input resolution size 154. An almost similar pattern can be seen in VGG13_XXS (receptive field size 134), which has the highest average test accuracy (84.6%) when trained with input resolution size 100. VGG16_XXS has a receptive field size of 196 and has the highest mean test accuracy with an input resolution size of 124 (86.2%). VGG19_XXS has a receptive field size of 252 and has the highest test accuracy (85.7%) when trained with input resolution size 152. Other input resolutions like 132, 172, 180, 192 also give almost the same accuracy but the accuracy starts decreasing after input resolution size 212.
Fig. 4.8 shows that for all the models we have trained the highest accuracy is obtained when input resolution is smaller than the receptive field size of a model. When input resolution is too small or is too large then the accuracy of a model decreases which is depicted by tails at the beginning and end of the plots. The prediction accuracy of a model increases for a certain range when we increase the input resolution without adding further detail. The same conclusion was also made by Shenk et al. (2020). These outcomes indicate that for a model to perform well, the input resolution should be in alignment with the receptive field size of a model.



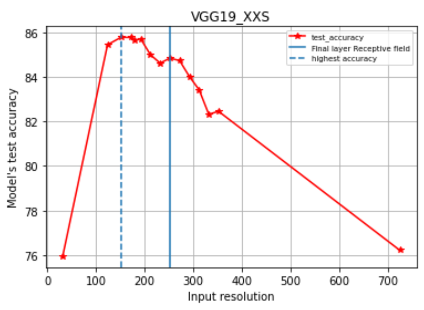
(a)



(b)



(c)



(d)

Figure 4.8: Different VGGNet models with varying input resolution indicating that input resolution size should be closer to receptive field size of a VGGNet models to optimal performance on accuracy. (a) VGG11_XXS with receptive field size 134; (b) VGG13_XXS with receptive field size 140; (c) VGG16_XXS with receptive field size 196; and (d) VGG19_XXS with receptive field size 252.

We have also calculated the Optimal-ratio for all VGGNet models. The calculated mean ratios of all the 3 runs for each model are shown in Table 6.

Table 6: **Ratio of receptive field size(RF) of a model to size of input resolution(IR) with highest accuracy. Ratios are rounded to hundredths.**

Models	Ratio
VGG11_XXS	1.07
VGG13_XXS	1.09
VGG16_XXS	1.45
VGG19_XXS	1.41

4.3.2. Effect on Saturation Tails

If we increase the input resolution, the tail pattern can be removed in the saturation plot (Shenk et al., 2020). Fig. 4.9 shows that having a small size input resolution compared to the receptive field size of a model causes a tail pattern. The tail pattern can be removed if we give we choose the input resolution for a model taking in the size of the receptive field of a model and the ratio given in Table 6 in consideration. If we train VGG11_XXS with input resolution size 124 which is approximately 1.07

times smaller than the receptive field size of VGG11_XXS, the tail pattern is removed. Having too high (724×724) or too low (32×32) resolutions also results in tail patterns in the back and front of the network.

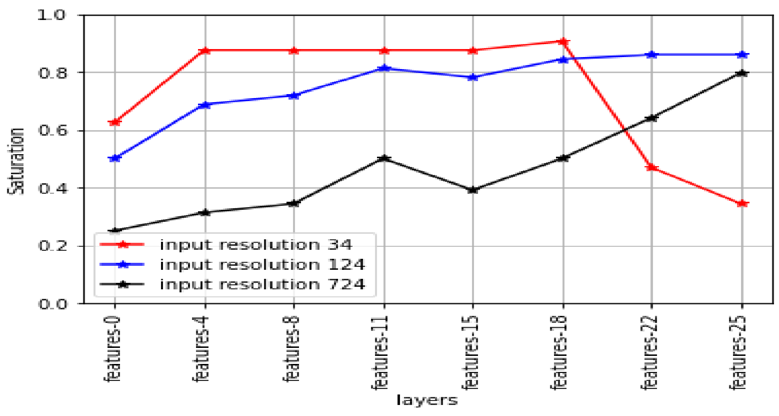


Figure 4.9: The tail pattern is removed with Input resolution in accordance with receptive field size. VGG 11_XXS trained with input resolution 34, 124, 254.

A similar pattern can be seen in VGG13_XXS, VGG16_XXS, and VGG19_XXS. If we choose the input resolution size with respect to the receptive field size of a model then we can remove the tail pattern. If we choose input resolution size 120 for training and testing the VGG13_XXS model, which is approximately 1.09 times smaller than the receptive field size of VGG13_XXS, the tail pattern is removed as shown in Fig. 4.10.

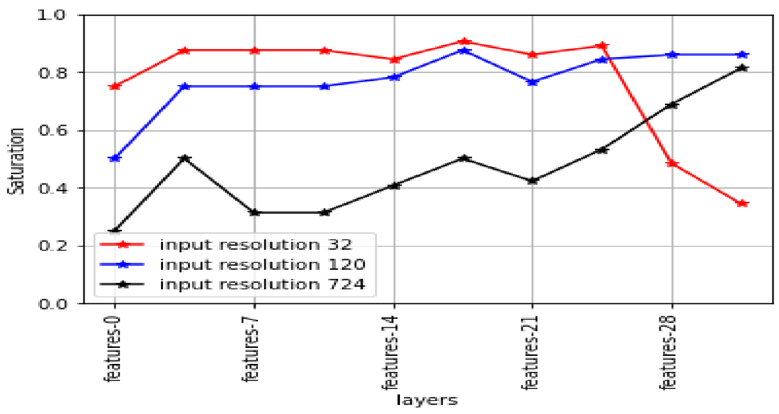


Figure 4.10: The tail pattern is removed with Input resolution in accordance with receptive field size. VGG 13_XXS trained with input resolution 32, 120, 724.

With input image resolution size(136) 1.45 smaller than the receptive field size of VGG16_XXS the tail pattern is removed in the VGG16_XXS model, which is depicted in Fig.4.11

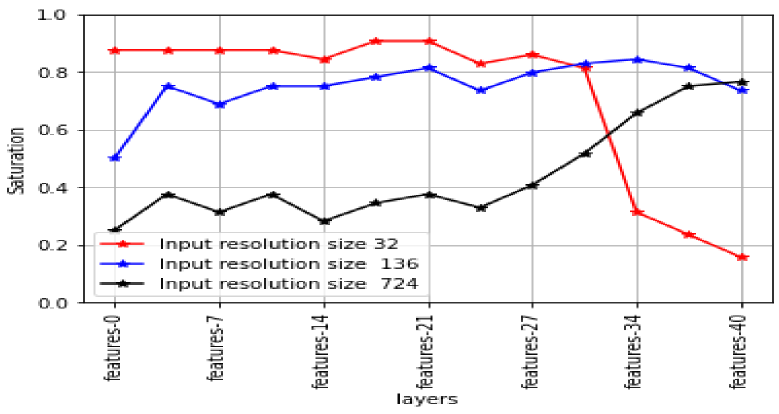


Figure 4.11: The tail pattern is removed with Input resolution in accordance with receptive field size. VGG 16_XXS trained with input resolution 32, 136, 296.

The figure shown above showed that for the optimal performance of the model the input resolution should be chosen according to the receptive field size of the model.

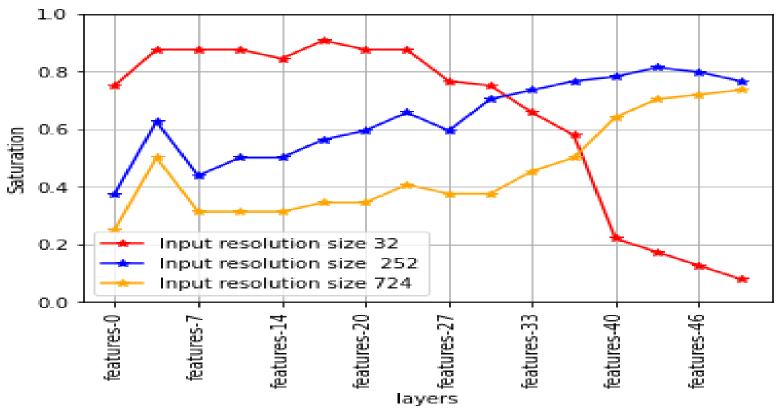


Figure 4.12: The tail pattern is removed with Input resolution in accordance with receptive field size. VGG 19_XXS trained with input resolution 32, 252, 352.

5. Conclusion

The experiments on CIFAR10 showed us that finding an appropriate input resolution to a VGGNet model leads to higher performance. This thesis has shown that the shallower VGGNet models (e.g. VGG11), given an appropriate input resolution, can outperform deeper VGGNet models (e.g. VGG19), with inappropriate input resolution. Just increasing the depth of a model doesn't lead to better performance, we also have to choose the right input resolution for a model to perform better. And we can decide the appropriate input resolution for VGGNet models based on their receptive field size and the depth of the architecture. I also introduced the idea of "Optimalratio", which gives us the heuristic to choose a better input image resolution for a given VGGNet Model. VGGNet models are suggested to be trained with an input image resolution size of 224 but it has not been mentioned Simonyan, Zisserman (2014), why they choose input image resolution size 224 for training VGGNet models. The results of this thesis indicate that input image resolution size should be closer to the receptive field size of VGGNet models to give an optimal performance, which is the case with VGG19 (Receptive field size 256) in the experiment by Simonyan, Zisserman (2014). This also Suggests that VGG19 performed better, on experiments by Simonyan, Zisserman(2014), than other less deep VGGNet Models (i.e. VGG11, VGG13, and VGG16), not only because the VGG19 was deeper than other models, but also it was provided with the appropriate input image resolution.

5.1. Further work

Repeating the experiments done in this thesis many times for averaging the results is the next step we can do. This multiple repetition and averaging reduce the possible influence of random factors like weight initialization and shuffling of the data-set.
Luo et al. (2016) introduced the way to calculate effective receptive field and showed that all pixels in a receptive field do not have an equal contribution to an output unit's response. We used the receptive field as the indicator for choosing a perfect input resolution for a model but instead of calculating the final layer receptive field size, we can calculate the effective receptive field.
We can improvise on this thesis by enlarging the experimental setup. Instead of training model with input resolutions which have 20 resolution size increment in each step we can change the input resolution size by 1 in each step. In this way, we can approximate more accurately the appropriate size of input resolution.
The "Optimalratio" introduced in this not very stable and meaningful but further exploring this ratio with proper mathematical calculation would be helpful in understanding the role of input resolution and receptive field size of a model to model's performance.
This thesis only deals with VGG models, for future work we can also test this on other different models like Resnet, Zfnet, GoogLeNet, and so on and see if this relation holds for these architectures as well.

A. Appendix

A.1. Source code

The source code can be found with the following hyperlink:
https://github.com/pariyashu/Bsc_Thesis
The source code is a clone of another GitHub repository. The original code was needed to be added with some models for this thesis. The configurations used for the experiments are also included in the source code. The source code of this original repository can be found here:
https://github.com/MLRichter/phd-lab.

A.2. Experiment 2- Further Details

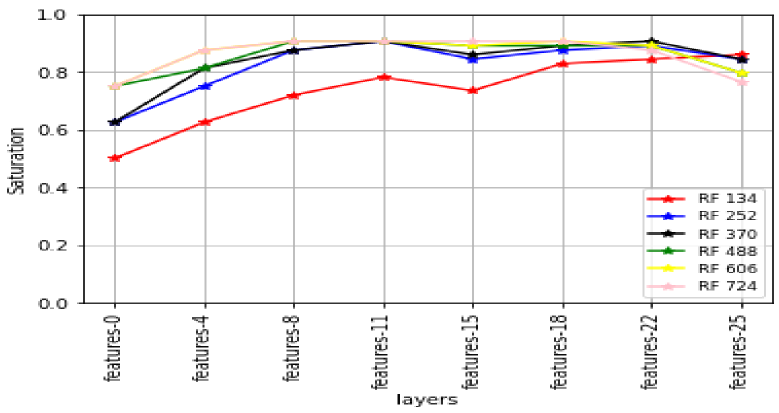


Figure A.1: Saturation plot of all dilated VGG11_XXS trained with input resolution size 156

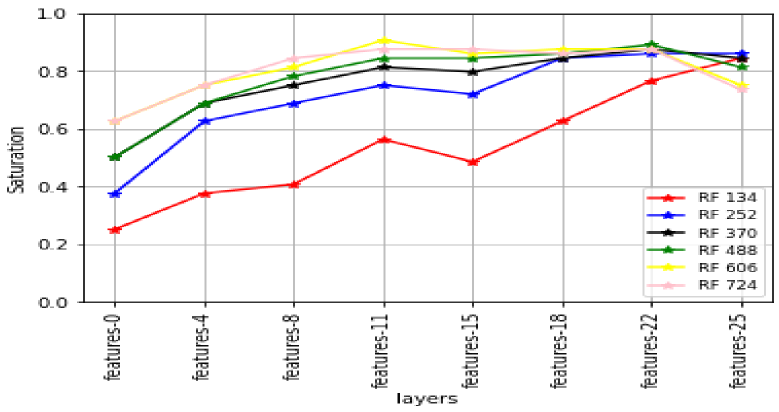


Figure A.2: Saturation plot of all dilated VGG11_XXS trained with input resolution size 456

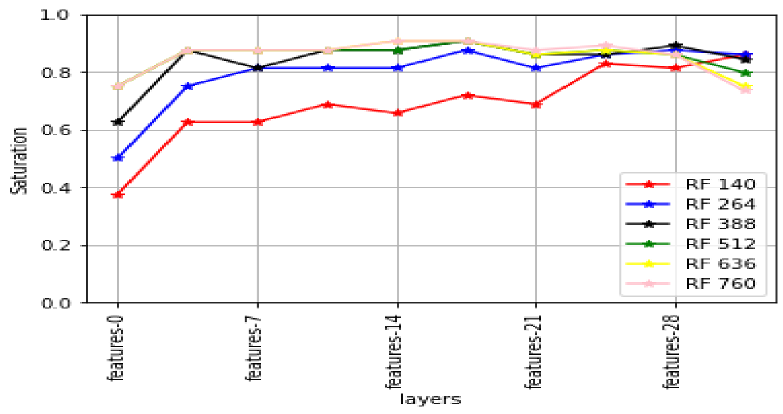


Figure A.3: Saturation plot of all dilated VGG_13, dilated with dilation rate 1, 2, 3, 4, 5, 6 and trained with input resolution size 224

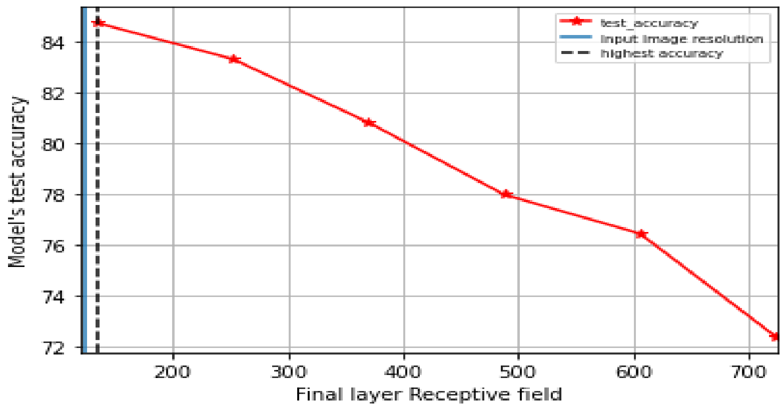


Figure A.4: Dilated VGG11_XXS trained with input resolution size 156

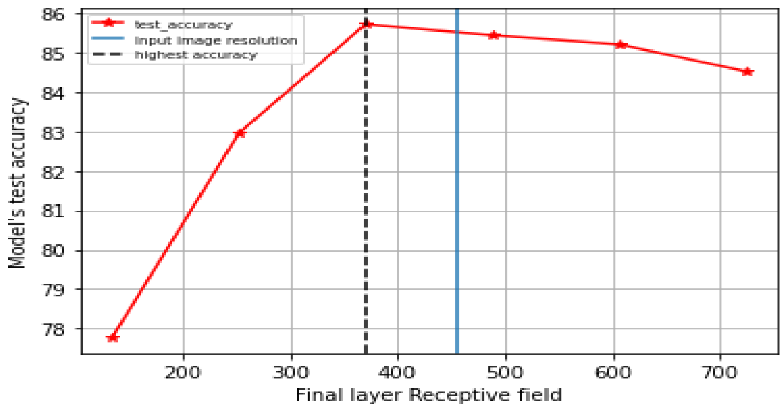


Figure A.5: Dilated VGG11_XXS trained with input resolution size 456

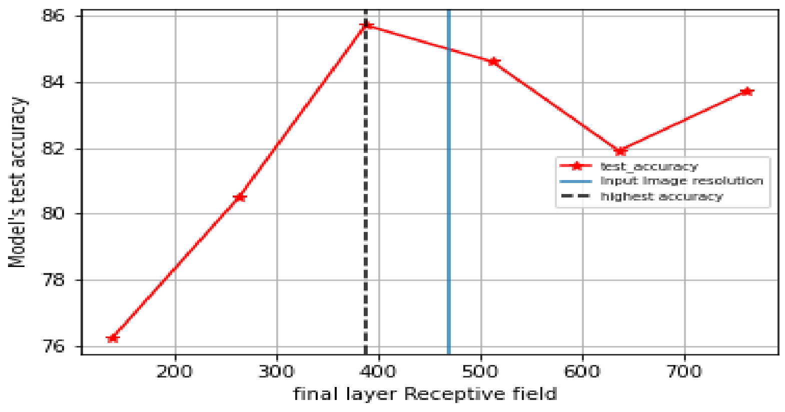


Figure A.6: Dilated VGG13_XXS trained with input resolution size 496

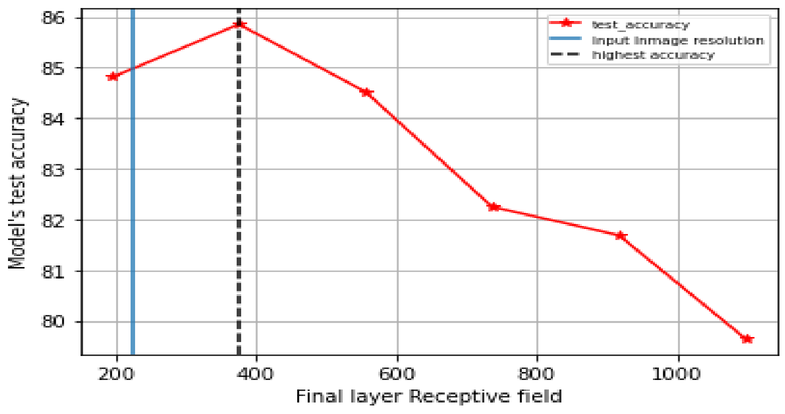


Figure A.7: Dilated VGG16_XXS trained with input resolution size 224

A.3. Experiment 3 - Further Details

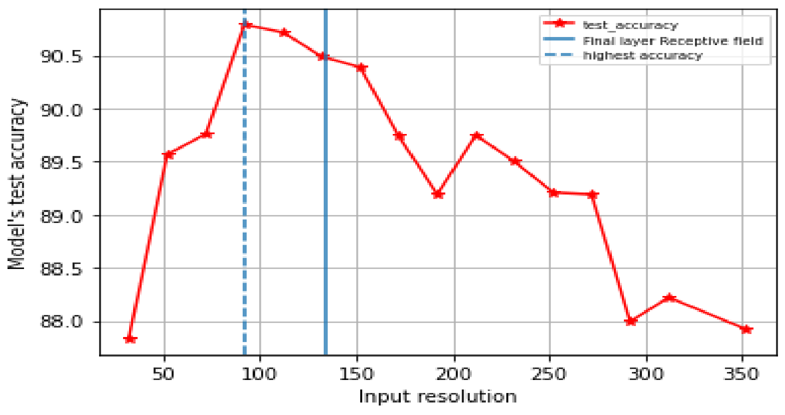


Figure A.8: Dilated VGG11 trained with different input image resolution sizes

References

Abdel-Hamid Ossama, Mohamed Abdel-rahman, Jiang Hui, Deng Li, Penn Gerald, Yu Dong. Convolutional neural networks for speech recognition // IEEE/ACM Transactions on audio, speech, and language processing. 2014. 22, 10. 1533–1545.

Alake Richmond. Understand Local Receptive Fields In Convolutional Neural Networks. 12 June 2020.

Baker Bowen, Gupta Otkrist, Naik Nikhil, Raskar Ramesh. Designing neural network architectures using reinforcement learning // arXiv preprint arXiv:1611.02167. 2016.

Ferreira Anselmo, Giraldi Gilson. Convolutional Neural Network approaches to granite tiles classification // Expert Systems with Applications. 2017. 84. 1–11.

Gál Viktor, Hámori József, Roska Tamás, Bálya Dávid, Borostyánkői ZS, Brendel Mátyás, Lotz Károly, Négyessy Laszlo, Orzó László, Petrás István, others . Receptive field atlas and related CNN models // International Journal of Bifurcation and Chaos. 2004. 14, 02. 551–584.

Goodfellow Ian, Bengio Yoshua, Courville Aaron, Bengio Yoshua. Deep learning. 1. 2016.

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

Holschneider Matthias, Kronland-Martinet Richard, Morlet Jean, Tchamitchian Ph. A real-time algorithm for signal analysis with the help of the wavelet transform // Wavelets. 1990. 286–297.

Hubel David H, Wiesel Torsten N. Receptive fields and functional architecture of monkey striate cortex // The Journal of physiology. 1968. 195, 1. 215–243.

Khan Asifullah, Sohail Anabia, Zahoora Umme, Qureshi Aqsa Saeed. A survey of the recent architectures of deep convolutional neural networks // Artificial Intelligence Review. 2020. 53, 8. 5455–5516.

Kim Yoon. Convolutional neural networks for sentence classification // arXiv preprint arXiv:1408.5882. 2014.

Koutini Khaled, Eghbal-Zadeh Hamid, Dorfer Matthias, Widmer Gerhard. The receptive field as a regularizer in deep convolutional neural networks for acoustic scene classification // 2019 27th European signal processing conference (EUSIPCO). 2019. 1–5.

Krizhevsky Alex, Sutskever Ilya, Hinton Geoffrey E. Imagenet classification with deep convolutional neural networks // Advances in neural information processing systems. 2012. 1097–1105.

LeCun Yann, Bengio Yoshua, others . Convolutional networks for images, speech, and time series // The handbook of brain theory and neural networks. 1995. 3361, 10. 1995.

LeCun Yann, Kavukcuoglu Koray, Farabet Clément. Convolutional networks and applications in vision // Proceedings of 2010 IEEE international symposium on circuits and systems. 2010. 253–256.

Li Yanghao, Chen Yuntao, Wang Naiyan, Zhang Zhaoxiang. Scale-aware trident networks for object detection // Proceedings of the IEEE international conference on computer vision. 2019. 6054–6063.

Lin Haoning, Shi Zhenwei, Zou Zhengxia. Maritime Semantic Labeling of Optical Remote Sensing Images with Multi-Scale Fully Convolutional Network // Remote Sensing. 05 2017. 9. 480.

Loussaief Sehla, Abdelkrim Afef. Convolutional neural network hyper-parameters optimization based on genetic algorithms // International Journal of Advanced Computer Science and Applications. 2018. 9, 10. 252–266.

Luo Wenjie, Li Yujia, Urtasun Raquel, Zemel Richard. Understanding the effective receptive field in deep convolutional neural networks // Advances in neural information processing systems. 2016. 4898–4906.

Pei Yanting, Huang Yaping, Zou Qi, Zang Hao, Zhang Xingyuan, Wang Song. Effects of image degradations to CNN-based image classification // arXiv preprint arXiv:1810.05552. 2018.

Best Practice Guide - Deep Learning. // . 02 2019.

Investigation of optimal receptive field size for maximizing mutual information and increasing learning efficiency. // . 2019.

Shenk Justin, Richter Mats L, Byttner Wolf, Arpteg Anders, Huss Mikael. Feature Space Saturation during Training // arXiv preprint arXiv:2006.08679. 2020.

Simonyan Karen, Zisserman Andrew. Very deep convolutional networks for large-scale image recognition // arXiv preprint arXiv:1409.1556. 2014.

Srivastava Rupesh K, Greff Klaus, Schmidhuber Jürgen. Training very deep networks // Advances in neural information processing systems. 2015. 2377–2385.

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A, others . Going deeper with convolutions. arXiv 2014 // arXiv preprint arXiv:1409.4842. 2014. 1409.

Tan Mingxing, Le Quoc V. Efficientnet: Rethinking model scaling for convolutional neural networks // arXiv preprint arXiv:1905.11946. 2019.

Wang Ruxin, Gong Mingming, Tao Dacheng. Receptive Field Size Versus Model Depth for Single Image Super-Resolution // IEEE Transactions on Image Processing. 2019. 29. 1669–1682.

Wu R, Yan S, Shan Y, Dang Q, Sun G. Deep image: Scaling up image recognition. arXiv 2015 // arXiv preprint arXiv:1501.02876. 2015.

Zagoruyko Sergey, Komodakis Nikos. Wide residual networks // arXiv preprint arXiv:1605.07146. 2016.

Zeiler Matthew D, Fergus Rob. Visualizing and understanding convolutional networks // European conference on computer vision. 2014. 818–833.