PROJECT REPORT

Topic:

FPGA-Based Convolution Neural Network Classification Using Inception v1 Architecture

University of Information Technology

April 2021

Student Information

Student name:	Student ID:
Tran Gia Bao	18520498
Nguyen Thien An	18520433

Supervisor

Truong Van Cuong

Acknowledgments

Here is your acknowledgment. Please giving your thanking to all people who have support or contribution to your work.

Tran Gia Bao
Nguyen Thien An

Contents

List of Figures

Chapter 1

Introduction

1.1 Abstract

A Convolutional Neural Network (CNN) is a class of deep neural network in deep learning. Based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation equivariant responses known as feature maps.

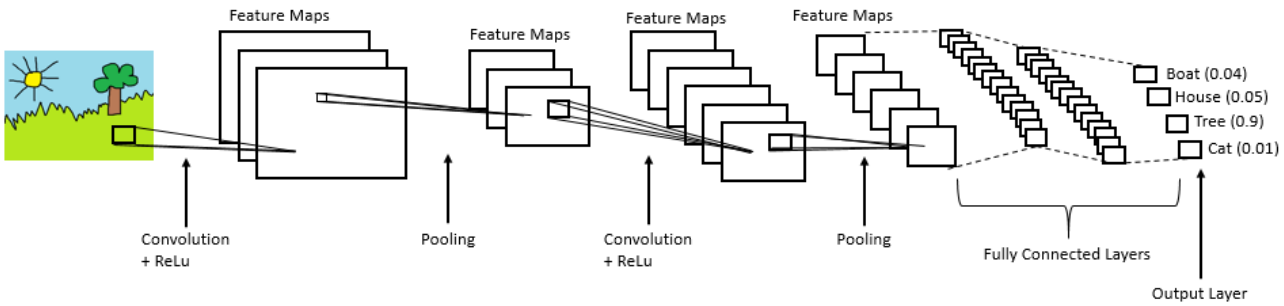

Figure 1.1: Base CNN

Some of the application areas of CNN include Image Classification and Segmentation, Object Detection, Video Processing, Natural Language Processing, and Speech Recognition.

Inception v1 to proposed by a group of researchers of Google, University of Michigan, University of North Carolina when joined in the ILSVRC 2014 competition. GoogleNet regulates the computations by adding a bottleneck layer of 1x1 convolutional filter, before employing large size kernels.

Inception Layer is a combination of all those layers (namely, 1×1 Convolutional layer, 3×3 Convolutional layer, 5×5 Convolutional layer) with their output filter banks concatenated into a single output vector forming the input of the next stage.

Inception v1 included 27 layers deep with 5M parameters. [1]

1.2 Inception v1 architecture.

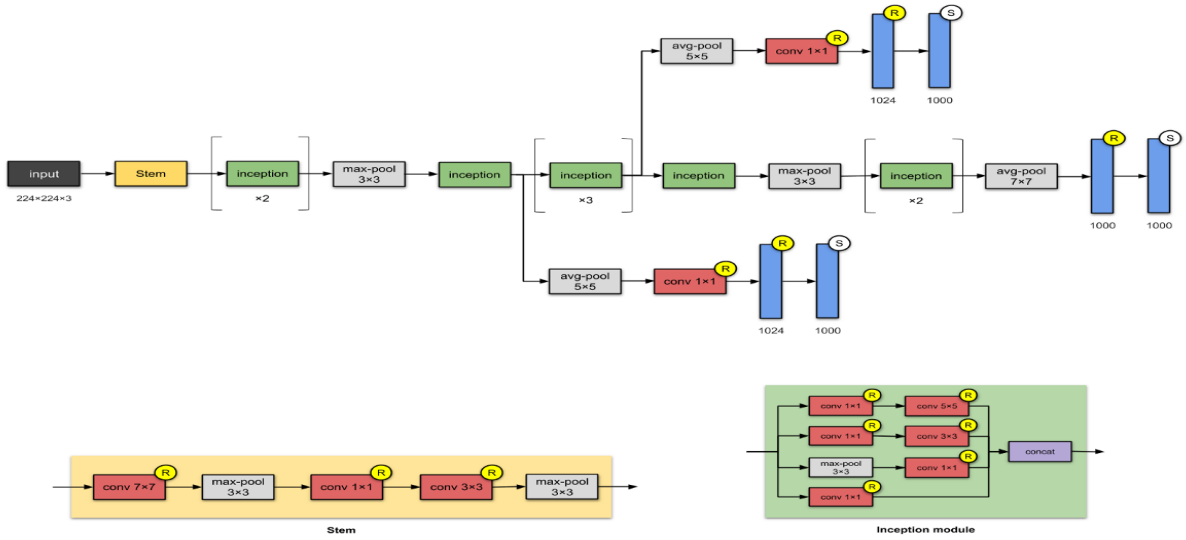

Figure 1.2: This figure is based on architecture in paper [1]

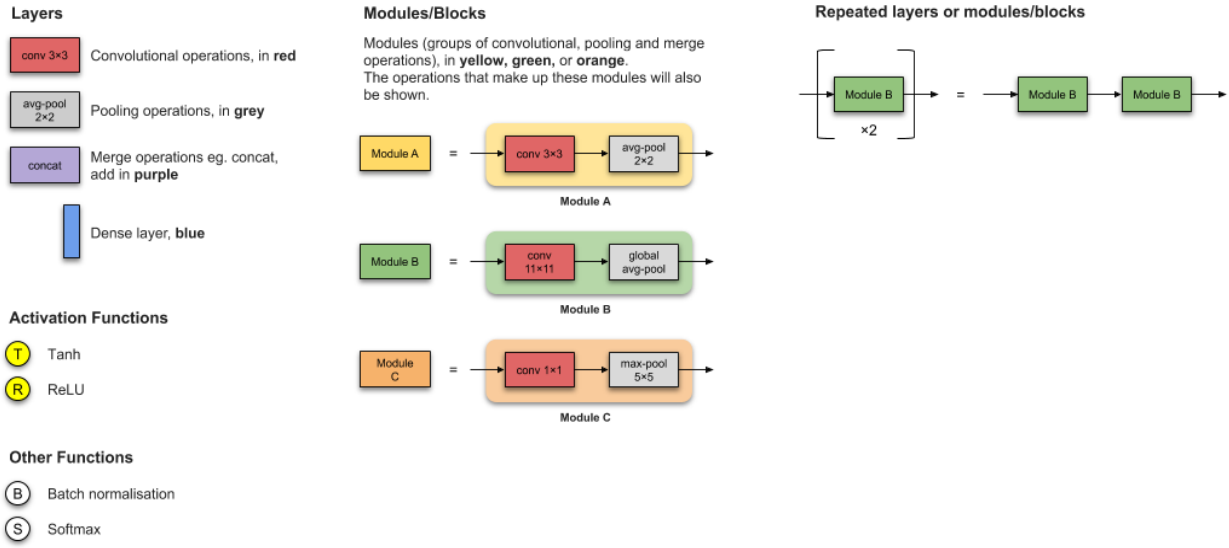

Figure 1.3: Description

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Figure 1.4: Configuration

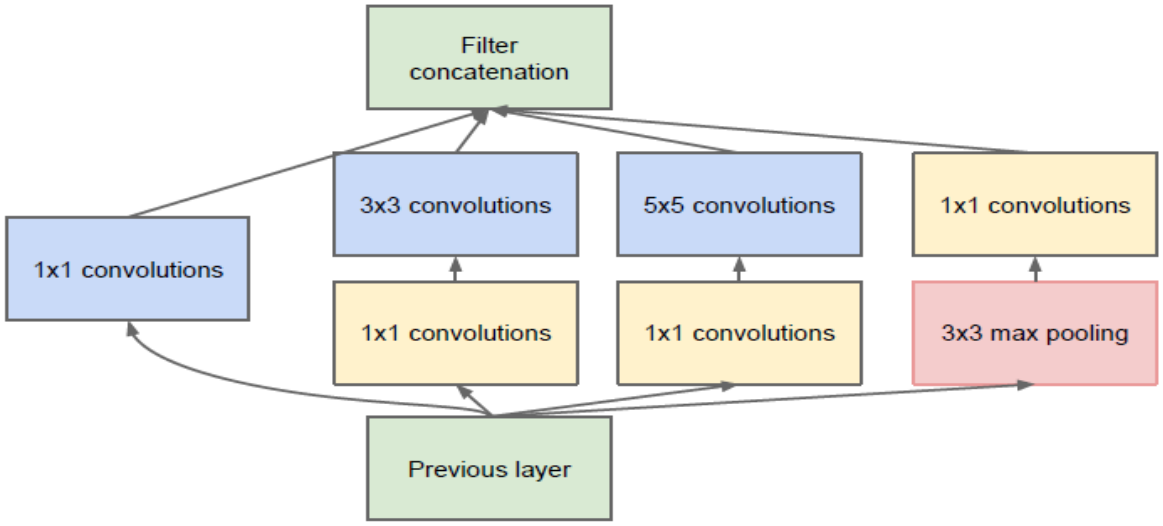

Figure 1.5: Reduce computation costs by introducing a 1 x 1 convolution

1.3 Layers

1.3.1 Convolution layers

In a CNN, the input is a tensor with a shape: (number of inputs) x (input height) x (input width) x (input channels). After passing through a convolution layer, the image becomes abstracted to a feature map, also called an activation map, with shape: (number of inputs) x (feature map height) x (feature map width) x (feature map channels). Example: No padding and stride is 1.

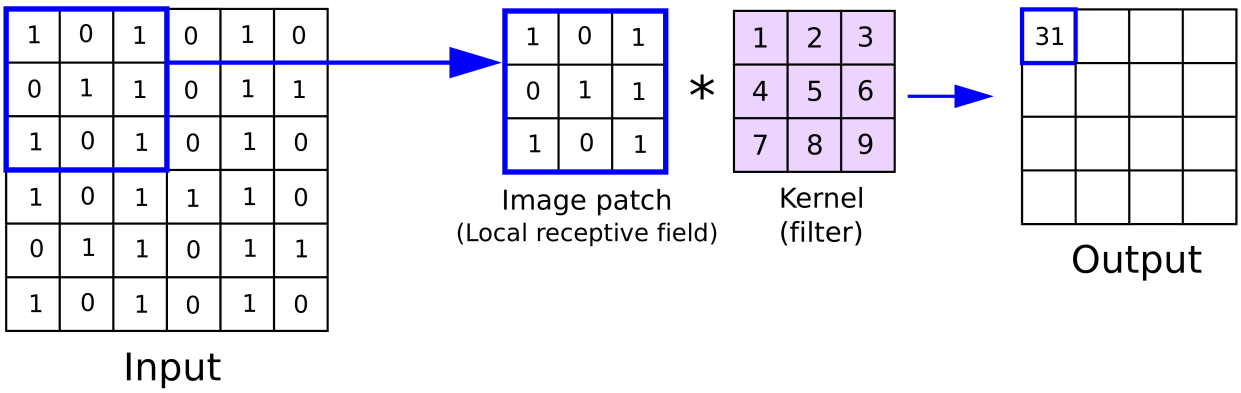

Figure 1.6: Convolution layer

1.3.2 Pooling layers

Pooling or down-sampling is an interesting local operation. It sums up similar information in the neighborhood of the receptive field and outputs the dominant response within this local region. There are two common types of pooling in popular use: max and average. Max pooling often to use preserving the importantly features. Average pooling often to use reducing the values. Example: pooling layer 2x2, stride = 2.

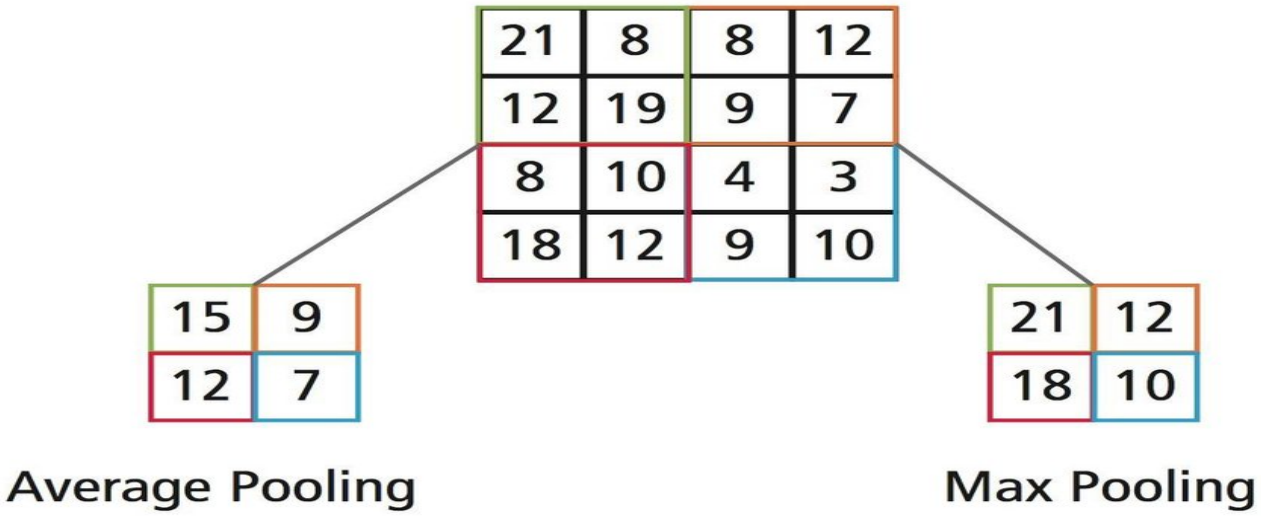

Figure 1.7: Pooling layer

1.3.3 Activation Function

Activation function serves as a decision function and helps in learning of intricate patterns. The selection of an appropriate activation function can accelerate the learning process. There are many functions like ReLU, Sigmoid, Tanh, Leaky ReLU, Maxout. ReLU is often preferred to other functions because it trains the neural network several times faster without a significant penalty to generalization accuracy.

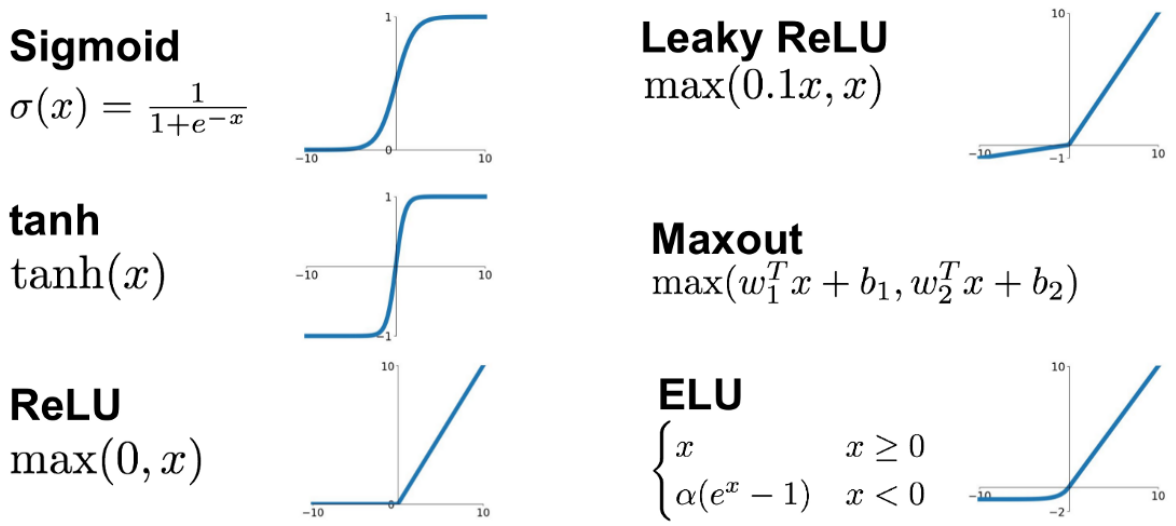

Figure 1.8: Activation function

1.3.4 Fully Connected layers

Fully connected layers connect every neuron in one layer to every neuron in another layer. The flattened matrix goes through a fully connected layer to classify the images.

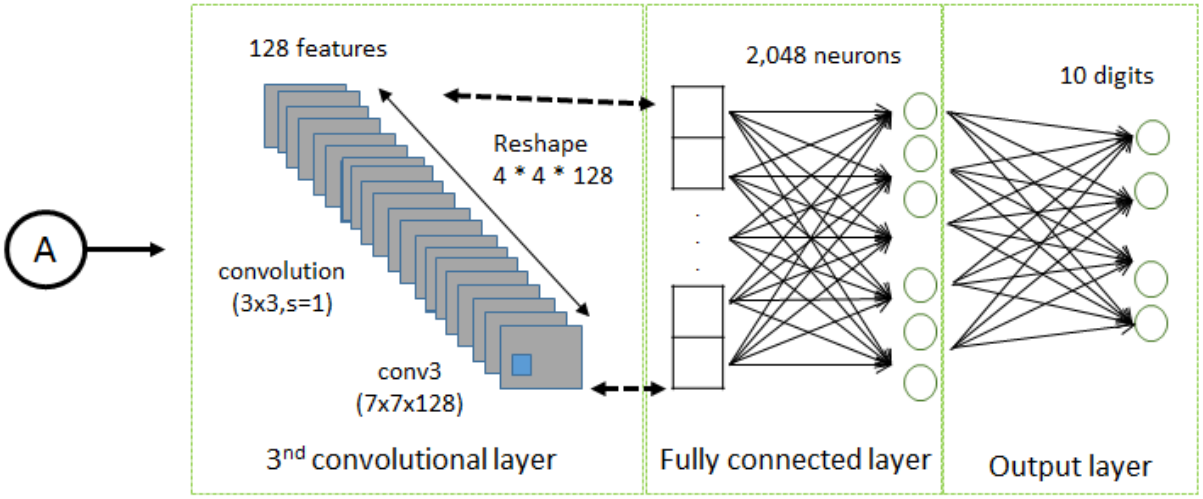

Figure 1.9: Fully Connected layer

Chapter 2

Ojectives

- Understand CNN and Inception v1 architecture theory.
- Can use python, matlab... to simulation.
- Implement Inception v1 with verilog HDL code with:

  Input: picture 224x224x3 (RGB image)
  Output: 10 classes
  Datasets: CIFAR-10.
  Classes:

  - airplane
  - automobile
  - bird
  - cat
  - deer
  - dog
  - frog
  - horse
  - ship
  - truck

Chapter 3

Implement and Result

References

[1] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich. Google, University of Michigan, University of North Carolina. "Going Deeper with Convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).