Smoke Detection Using Deep Convolutional Neural Network

by

Wahidul Hasan Niloy
18341009
Mostafa Kamal Ornab
1824120
Saurav Saha
13101148

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
BRAC University
August 2019

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at BRAC University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Wahidul Hasan Niloy
18341009

Mostafa Kamal Ornab
18241020

Saurav Saha
13101148

Approval

The thesis titled "Smoke Detection Using Deep Convolutional Neural Network" submitted by

1. Wahidul Hasan Niloy (18341009)
2. Mostafa Kamal Ornab (18241020)
3. Saurav Saha (13101148)

Of Summer, 2019 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on August 28, 2019.

Examining Committee:

Supervisor:
(Member)

Dr. Jia Uddin
Associate Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Dr. Jia Uddin
Associate Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Mahbubul Alam Majumdar
Professor and Chairperson
Department of Computer Science and Engineering
BRAC University

Abstract

In a densely populated country like Bangladesh, fire accidents have become a frequent disaster that primarily be formed as a consequence of unconsciousness among the people. Therefore, detection of smoke, is a must in order to have an earlier caution before the damages caused by fire. Thereby, in this paper, we have approached a deep convolutional neural network in the identification of smoke from images by using the process of image processing. The detection of smoke images recognized as a difficult task for having of a larger differentiation in textures, colors and structures. In competing with the challenges of detecting smoke, the model has developed with the help of the methodology of image processing and computer vision, through the deep convolutional neural network in the identification of smoke images. We have succeeded to gain the accuracy in a sufficient ratio.Using the model of Deep CNN, "VGG-19" and "Inception-v3" we have gained the accuracy of 82.33% and 84.67%. Moreover, for reducing the overfitting problem, we have structured an increasing amount of training data sets through the data augmentation techniques. Thus, the Deep Convolutional Neural Network has been utilized to perform in a more accurate way by gathering the accuracy in a more preferable way in the procedure of smoke detection.

Keywords: Deep Convolutional Neural Network;computer vision; VGG-19; Inception-v3; Smoke detection

Dedication

We like to dedicate our thesis to our honorable faculty members and our beloved parents. We are also thankful to our seniors and all the well-wishers of our department, for whom we are able to successfully accomplish our goal.

Acknowledgement

We would like to express our heartiest gratitude towards the Almighty Allah. Secondly, we would like to share our sincere gratitude to our advisor Dr. Jia Uddin for his constant motivation, support and immense knowledge to our research. It would not have been possible without his constant guidance in all phases of our project. Special thanks to Mr. Reza Tanzim for perfectly guiding us in Image Processing. Finally, we would like to thank our gratefulness to parents, brothers, sisters and beloved friends for encouraging and supporting us from the very beginning.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

$BFSCD$ Bangladesh Fire Service and Civil Defense

CNN Convolutional Neural Network

$DCNN$ Deep Convolutional Neural Network

GAN Generative Adversarial Network

$SDRIP$ Smoke Detection Using Image Processing

VGG Visual Geometry Group

LRN Linear Response Normalization

YOLO You only look once

β shifting

γ scaling

μ mean of x in mini-batch

σ std of x in mini batch

π $\simeq 3.14\ldots$

Chapter 1

Introduction

Smoke detection has become an important and significant way of decreasing damages caused by fire.The detection of smoke at an early stage, has become an important need in avoiding a large scale of damages. There exists several approaches in visual scenes of detecting smoke. Among those most commonly used model signifying as of the physical sensor based approach, it has showcased the outcomes with a small amount of effectiveness . It has been used the days only for it has the cheap availability, simplicity and of having a bit amount of accuracy. The primary shortcomings of a sensor-based smoke detection, has already performed with the poor outcomings, defining as the incapability of detecting the source of smoke, the amount of it and as well as the direction of the actual ranges, it actually covered. Moreover, smoke blurs the visual scenes tending to an unstable position. So, based on the framework handcrafted feature some of the most algorithms could not make out with the desired output. Therefore, in this paper we have come through with the novel model of deep convolutional neural network, that performed in a proper manner gaining the acquisition of a higher amount of accuracy and higher detection rate. The model signifies the state-of-the-art in the deep learning procedures with strongly demonstrating procedures featuring some of the pre-trained convolutional neural networks defining "VGG19", "InceptionV3". This model of networks are compatible with the capability in a fully connected layer with the feature of extracting on a multi-level phase in computing with the convolutions within the same module of networks.

1.1 Motivation

Since, over the last two decades, we have been facing a large number of fire incidents across the country, causing a huge damage to our properties and taking many lives. As indicated by the official information, the degree of property harm caused because of flame occurrences saw an upward pattern since 1998 while number of setbacks brought about by flame saw both good and bad times in the middle of 2006 and 2018 [30]. A sum of 1,762 individuals were murdered and 10,625 others injured in flame during the period somewhere in the range of 2006 and 2018, as indicated by the information accessible with Bangladesh Fire Service and Civil Defense (BFSCD). Moreover, The quantity of flames has expanded more than triple crosswise over Bangladesh since 1997; with the year 2018 seeing a day by day normal of 53[2]. So, we are facing the most disastrous issues throughout our present time, and we are not assured about how the forthcoming tragedies going to cause our lives in a more

devastating way. To control the casualties, everyone should come forward and work over the solution. The first and foremost work continues with the raise of awareness among the general people with the scenarios. Therefore, our motivation comes alive to make out a solution of the problem and in all, for the betterment of our country in the forthcoming days.

1.2 Objectives

The main purpose of our thesis is to introduce with the technological approach determining the prior caution of fire, defining as smoke detection, which has now turned into a huge need for our country. Our goal structured on the process of detecting smoke by image processing using the Deep Convolutional Neural Networks. In order to stop the casualties dropping over as a consequence of fire damages, we must detect the smoke as a strong form of precautions. Moreover, the prior identification of fire, resulting in the detection of the smoke images, could remain us to forfeit those devastating tragedies and help us to save thousands of lives, properties and wealth of our nation. Thus, we expect to grow the awareness among the general people from our research, as well to encourage others to get engaged with this sector with their profound knowledge, resulting to be existed in the form of a more secure country across the world.

1.3 Thesis outline

Chapter 2: Discusses the "Literature Review" of related works in this field

Chapter 3: Discusses the "Proposed Model" that includes the description of our improved model and implementation process of Smoke detection using Deep Convolutional Neural Network

Chapter 4: Discusses the Experimental Results and Discussion

Chapter 5: Conclusion and future findings

Chapter 2

Literature review

This part of our thesis, replicates the work that was developed and structured on the previous researches, upon the deep convolutional neural network in recognition of pattern. The part also covers up the areas of developing the techniques, for improving the sector of algorithm determination through the data augmentation phase.

2.1 Convolutional Neural Networks

Convolutional Neural Networks known as the ConvNets define a class of Neural Networks. They are extremely beneficial for image classification. The convolutional neural networks signifying a deep learning algorithm. It has the capability of distinguishing images that are assigned, based on the learning through the phases of weights of those images. It take less time during the classification phase in comparison with the other algorithms similar to the classification procedures or phases. The structure implying the similarity in the phase of connectivity pattern of Human brain Neurons. The development brought prominent upgrades in the presentation of the advancement of CNN. The availability in a large scale of datasets, made the researchers to be applicable of allowing to go deeper with CNN. Moreover, the development of the powerful GPUs also can easily defined of getting deeper through it. As refer to this, a deep CNN of Alexnet [14] was introduced and established by Krizhevsky et al [5] that placed excellent performance making top-1 and top-5 error rates of 37.7% and 17.0% in ImageNet large Scale Visual Recognition Challenge. Afterwards, Zeiler et al also showed good classification performance through the visualization technique.[9] For instance: diagram of convolutional neural networks with TensorFlow as Fig. 2.1

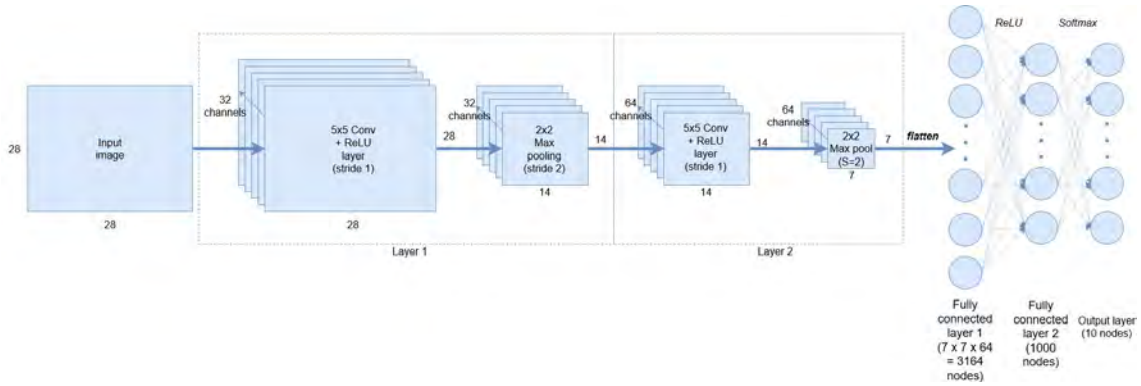


Figure 2.1: Convolutional neural network

In the diagram of convolutional neural networks showed on Fig. 2.1, dimension of 28×28 greyscale of digits in Mnist images was implied. A 5×5 convolutional channels plus ReLu is been activated with pace with the creation of 32. After that applying the max pooling operation of a 2×2 , it implements the down-sampling. Afterwards, with the 64 channels with along the stride 2 max pool down sampling, the output being flattened for getting a fully connected layer. It consists of total 3164 nodes with of a hidden layer consisting 1000 nodes. The layers are performed then with ReLu activations node. At last, getting of the 10 digit possibilities in terms by using the softmax identification.

2.2 Batch Normalization

Batch normalization [10] is a process of accelerating the performance in terms of speed and stability of the artificial neural networks. Also known as the "BN-Inception" or "Inception-v3". It is used for normalizing the input with the actions performing margin the activations. For instance, from features of 1 to 1000 we need to normalize it for a better learning in other sense, speed up the learning. The hidden layers values go with the same procedures which results in 10 times faster in course of training speed. Batch normalization [17] adds both of the parameters that stayed in training category for the multiplication by a standard deviation parameter in the normalized output procedure. Then following the procedure of adding a mean parameter. The whole process is as Fig. 2.2

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

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

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

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

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

Figure 2.2: Batch Normalization Transformation

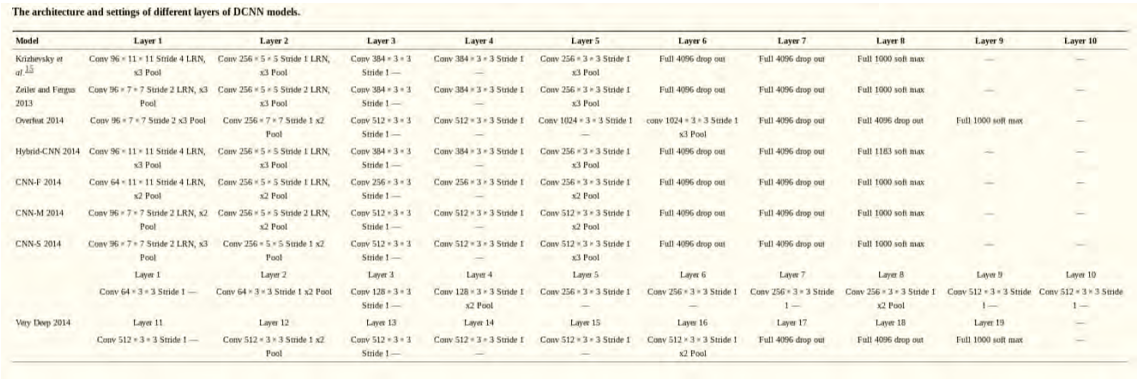
2.3 Background

Hereby, we have used the deep CNN(Convolutional Neural Network) as per following the "VGG-19" and "Inception-v3", which brings the efficiency in our accuracy. We also able to solve the data overfitting problem, that most of the previous algorithms faces. Moreover, in the detection part we have utilized the "YOLO-v3" algorithm which is a popular real-time object detection algorithm defined as 'You only look once'. It is better and stronger in terms of image detection than "YOLO-v2" as well came with the improved incremental with a better feature extraction mode.

2.3.1 Overview of Deep Convolutional Neural Network

Deep convolutional neural network generally structured in a combination of deep learning procedures with the convolutional neural networks. It is to be noted that

the deep convolutional neural networks have been recognized presently [3] with much more attention. It has a great deal of persuasion towards the area of performance, performing in the identification of the object categories from natural image databases. The hierarchy is based on a several back to back feature detecting layers.The higher the layer it gets, the higher complexity it could handle. There are various algorithms already defined for the deep convolutional neural networks. Amongst them the supervised learning methodologies have performed the higher rate of success. Here, the layers with the higher tendencies tend to perform in the identification of more complex features. As following the layers with lower tendencies tend the performance of the identification to the simple features. The structured raised the similarities in some extent with the human visual system. Their similarities also break down in the extraction of more features by the number of a hierarchy layers. In the deep convolutional neural networks the convolution is considered as the main stream where every layer seems to be followed by a number of back to back operations defining as max pooling, output normalization. Hubel and Wiesel introduced with a similar architecture in between the human visual system and the convolutional neural networks [1]. The concept of neocognitron then considered as the fundamental of the CNN by Fukushima et al [4]. Later on, LeCun et al [3] came up with a milestone of the convolutional neural networks named "LeNet-5" [16], becoming the pioneering convolutional neural networks. Afterwards, in 2012 Krizhevsky et al came up with a model [5] that performed on the ImageNet database in an impressive manner. The model inherits five feature detectors as convolution with three fully connected layer. It applied the Rectified Linear Unit(ReLu) over the neurons for the activation function. This boosted up the learning phase in an efficient way. Here, on the layers the max pooling operation is trained over 60 million free parameters for avoiding overfitting. The data augmentation plays a notable part in the reduction phase of overfitting [8] as well enlarging the capacity of training set with the dropout procedure in the learning phase. A pre-trained structured details of this model shown on Fig. 2.3



The architecture and settings of different layers of DCNN models.

Model	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7	Layer 8	Layer 9	Layer 10
Krizhevsky et al. [15]	Conv 96 × 11 × 11 Stride 4 LRN, x3 Pool	Conv 256 × 5 × 5 Stride 1 LRN, x3 Pool	Conv 384 × 3 × 3 Stride 1 —	Conv 384 × 3 × 3 Stride 1 —	Conv 256 × 3 × 3 Stride 1 x3 Pool	Full 4096 drop out	Full 4096 drop out	Full 1000 soft max	—	—
Zeiler and Fergus 2013	Conv 96 × 7 × 7 Stride 2 LRN, x3 Pool	Conv 256 × 5 × 5 Stride 2 LRN, x3 Pool	Conv 384 × 3 × 3 Stride 1 —	Conv 384 × 3 × 3 Stride 1 —	Conv 256 × 3 × 3 Stride 1 x3 Pool	Full 4096 drop out	Full 4096 drop out	Full 1000 soft max	—	—
Overfeat 2014	Conv 96 × 7 × 7 Stride 2 x3 Pool	Conv 256 × 7 × 7 Stride 1 x2 Pool	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 —	Conv 1024 × 3 × 3 Stride 1 —	conv 1024 × 3 × 3 Stride 1 x3 Pool	Full 4096 drop out	Full 4096 drop out	Full 1000 soft max	—
Hybrid-CNN 2014	Conv 96 × 11 × 11 Stride 4 LRN, x3 Pool	Conv 256 × 5 × 5 Stride 1 LRN, x3 Pool	Conv 384 × 3 × 3 Stride 1 —	Conv 384 × 3 × 3 Stride 1 —	Conv 256 × 3 × 3 Stride 1 x3 Pool	Full 4096 drop out	Full 4096 drop out	Full 1183 soft max	—	—
CNN-F 2014	Conv 64 × 11 × 11 Stride 4 LRN, x2 Pool	Conv 256 × 5 × 5 Stride 1 LRN, x2 Pool	Conv 256 × 3 × 3 Stride 1 —	Conv 256 × 3 × 3 Stride 1 —	Conv 256 × 3 × 3 Stride 1 x2 Pool	Full 4096 drop out	Full 4096 drop out	Full 1000 soft max	—	—
CNN-M 2014	Conv 96 × 7 × 7 Stride 2 LRN, x2 Pool	Conv 256 × 5 × 5 Stride 2 LRN, x2 Pool	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 x2 Pool	Full 4096 drop out	Full 4096 drop out	Full 1000 soft max	—	—
CNN-S 2014	Conv 96 × 7 × 7 Stride 2 LRN, x3 Pool	Conv 256 × 5 × 5 Stride 1 x2 Pool	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 x3 Pool	Full 4096 drop out	Full 4096 drop out	Full 1000 soft max	—	—
	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7	Layer 8	Layer 9	Layer 10
	Conv 64 × 3 × 3 Stride 1 —	Conv 64 × 3 × 3 Stride 1 x2 Pool	Conv 128 × 3 × 3 Stride 1 —	Conv 128 × 3 × 3 Stride 1 x2 Pool	Conv 256 × 3 × 3 Stride 1 —	Conv 256 × 3 × 3 Stride 1 —	Conv 256 × 3 × 3 Stride 1 —	Conv 256 × 3 × 3 Stride 1 x2 Pool	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 —
Very Deep 2014	Layer 11	Layer 12	Layer 13	Layer 14	Layer 15	Layer 16	Layer 17	Layer 18	Layer 19	—
	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 x2 Pool	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 —	Conv 512 × 3 × 3 Stride 1 x2 Pool	Full 4096 drop out	Full 4096 drop out	Full 1000 soft max	—

Figure 2.3: Different layers of DCNN models architecture

The subtleties of convolutional layers (marked as Conv) are given in three sub-pushes: the first demonstrates the number following the size of the convolution channels as $Num \times Size \times Size$; the convolution walk is allowed in the subsequent sub-push; and the third one demonstrates the maximum pooling down-testing rate. Each line of the table alludes to a DCNN model and every segment contains the subtleties of a layer where the Linear Response Normalization (LRN) [11] was set. Along with, subtleties of completely associated layers are displayed in two sub-pushes indicating the first demonstrates the quantity of neurons whereas the second one whether dropout max tasks are connected.

2.3.2 Data Augmentation

Building a successful deep CNN [15] in visual identification procedure, is not only just defining a greater architecture or an incredible computational environments but also the presence of a large number of image data. There is no doubt, with a greater amount of data comes with the greater achievements, resulting in the success of a well trained datasets. But, in the availability of those big datasets in some cases like our smoke detection are not available for training. Thus, resulting in a overfitting problem. As a result, incapability of validation and testing the datasets occured. Meaning, not able to generalize the data in a well procedure. Here, data augmentation is a must, as considered to be the easiest method of reducing the overfitting problem. By using the data augmentation techniques we can easily generate new training samples. Using the generative models, new training images can be created.For example, GANs algorithms [6] that uses the min-max strategy, where a generative model G creates comparative examples from the first information circulation, and a discriminative model D is prepared to assess the likelihood that an example came from preparing information instead of G. Moreover, specialized images often faced with insufficient amount of data. In extend, in the medical industry, accessing the data is strictly maintained for having a better privacy environment. The systems have been created which consolidate master space information with pre-prepared models. Significant assignments, for instance, characterizing disease types [24] are obstructed by this absence of information.

2.3.3 VGG model

The VGG network system is portrayed by its straightforwardness. Having a 3×3 convolution layers stacked in the verse of the top of each other in order to have an expansion in depth. It was first proposed by Simonyan and Zisseman in the paper of "Very Deep Convolutional Networks for Large Scale Image recognition" in 2014 [7]. Here, the reduction of size volume is controlled by the max pooling. Each with 4,096 hubs are trailed by a softmax classifier with the two completely associated layers, it comes in working with the final layer of neural network-based classifier.

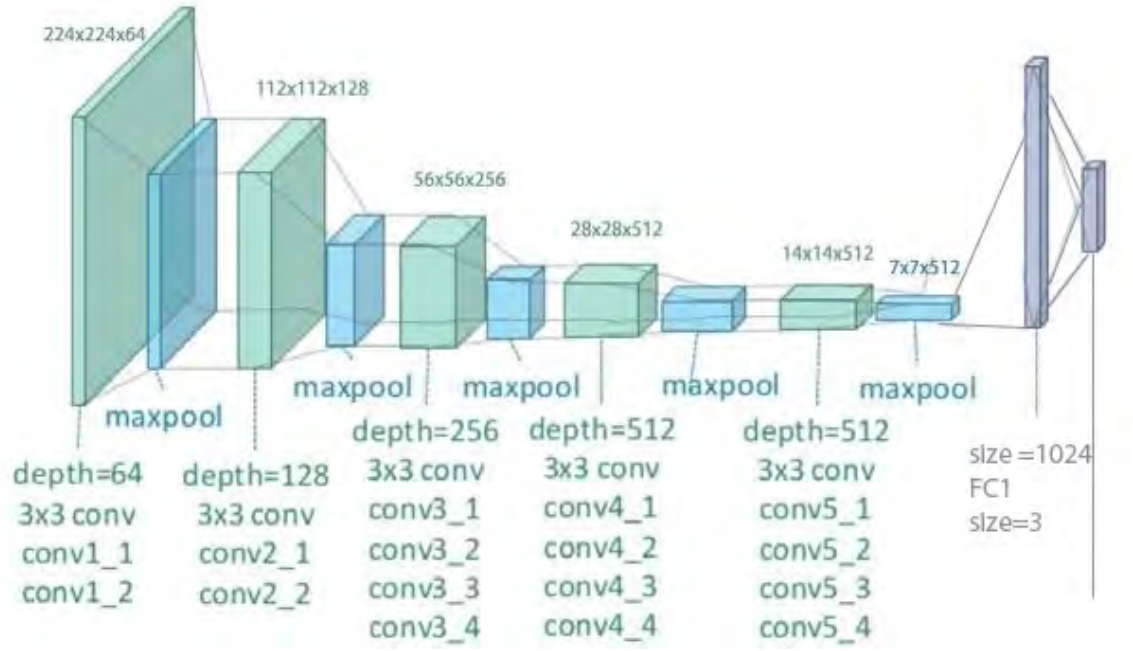


Figure 2.4: VGG architecture

VGG is released in two different CNN models [21], defining "VGG-19" and "VGG-16" which represents a 16-layer model and a 19-layer model respectively. This '16' and '19' also known as the number of weights. They are good for the purpose of new models that utilized image inputs. Defining as of the very two powerful models, they are worthy, as a form of image identifiers. We are using the Keras which is a neural network library in python. Keras provided both the VGG-16 and VGG-19 layer version according to their respective classes. When the VGG is loaded, Keras will first determine the weights as a form of 16 or 19 indicating the VGG-16 or VGG-19 classes respectively. In first run the weights are loaded into the directory in such that the next time it will load from that saved local directory. The VGG model expect inputs in the size of 224×224 pixels consists of three channels [20].

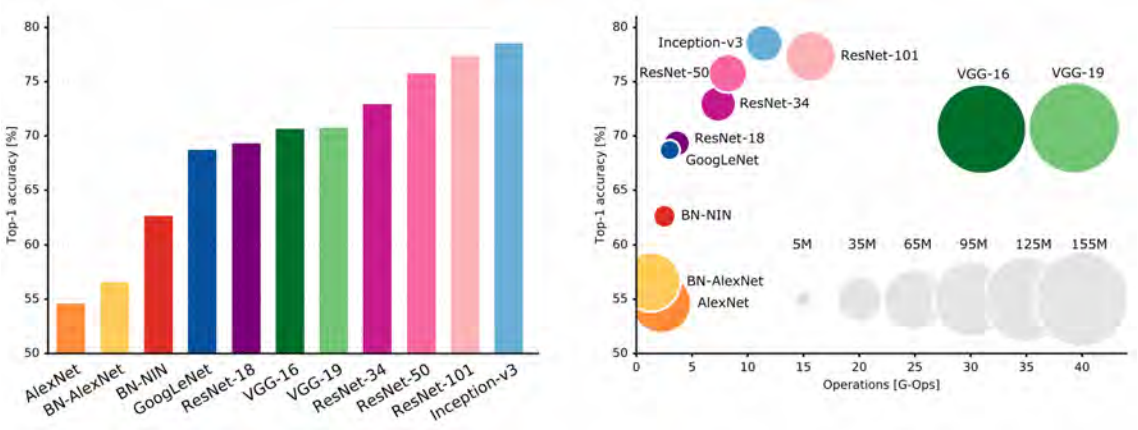


Figure 2.5: VGG in analysis of the deep Neural Network Models

The reason using the VGG network, is the number of parameters are few, [7] covering 1 filter per layer excluding the bias with 1 layer at the input. Moreover, the VGG architecture used 2 layers for 3×3 filters which covered 5×5 and using the 3 layers of 3×3 filters, [18]it respectively covered the 7×7 area. Thus, no need of using the large-size filters. For instance: the Alexnet 11×11 or the ZFNet 7×7.

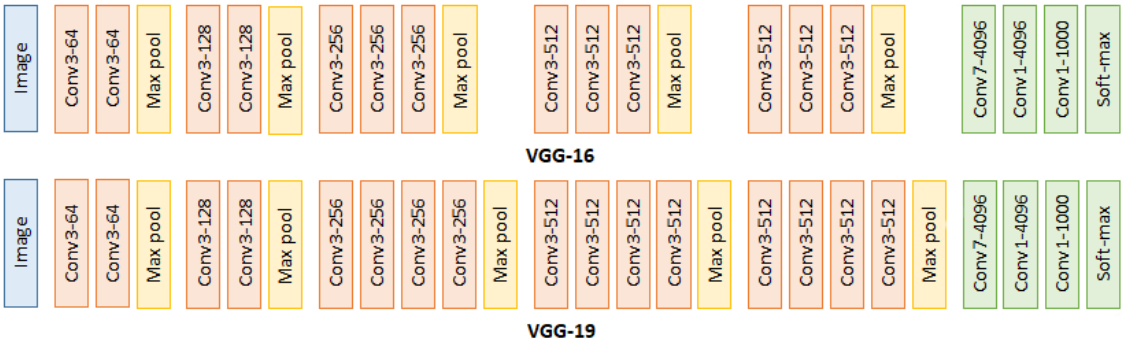


Figure 2.6: VGG in analysis of the deep Neural Network Models

VGG obtains 8.8% error rate which is a sign of improving in the phase of deep learning network by lower error rate.

2.3.4 Inception-v3

The Inception-v3 [19] was implied for the feature extraction phase with the convolutional neural network and the identification of images with a fully connected process. It also inherits the softmax layer defining the function for the last layer. The number of parameters are fewer in this case with a strong computational efficiency, consists of 42-layers deep learning network. It carries the same complex environment as the VGGNet. Here, in the last layer of 17×17 layer, an auxiliary classifier is utilized [23], instead of defining two auxiliary classifiers which makes it efficient in some more extent of VGGNet. Inception-v2 talks about the batch normalization whereas the Inception-v3 talks about the factorization convolution, which is more efficient. They perform well in classifying the images with low resolutions or which are small. As the goal defined in the inception-v3 is acting on the extraction of multi-level feature, the computation 1×1 respectively followed by the other two 3×3 and 5×5

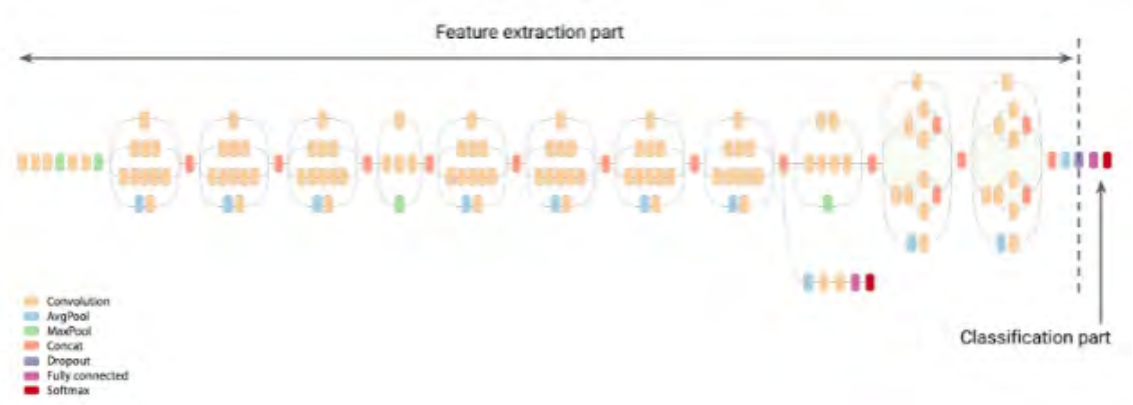


Figure 2.7: Inception-v3

convolutions get stacked in the form of output with the channel dimension before going onto the next layer. In the [5] ImageNet, we can see the original concept which is shown on Fig. 2.8

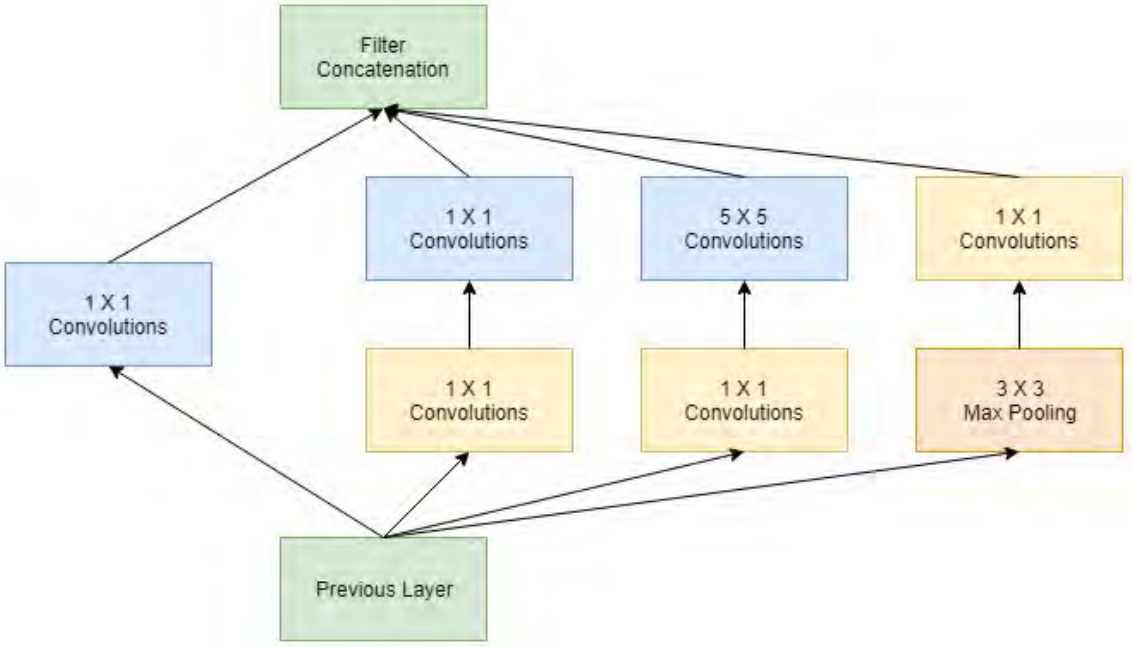


Figure 2.8: Inception module of GoogLeNet

2.3.5 YOLO-v3

The word "YOLO" stands for 'You only look once'. The "YOLO-v3" is the simplified and upgradable version of the "YOLO v2". It is popular for the real-time object detection. The incremental upgrades of the "YOLO v3" structured from the previous versions, named as the "Darknet-53", which is regarded as a better object detector [28]. It has the capabilities of a better feature extraction mode with short connections. Moreover, it performs in a great way in terms of an object detection with the quality of concatenation and feature map upsampling. By this YOLO v3, we can perform multilabel classification in detection of objects on images [22].

Uploading a variant from the Darknet the YOLO-v3 works [26]. The YOLO-v3 is normally trained on ImageNet, which consists of 53 layers of networks. But in our image detection system, we have used 106 layers that means another more 53 layers adding to this. Here, we used the logistic regression for the class prediction, solving the squared errors replacing by cross-entropy error [25]. The detection is done in three different scales. A fully convolutional network [12], making the detection at three different scales,the YOLO v3 [29] is generating the output by the process of applying 1×1 kernal at a time on the feature map. In three different sizes at three different places of the network, the identification is finished by applying 11 kernels on feature maps. At the multilabel classification, for having overlapping labelings [27], the independent logistic classifiers are utilized along with the binary cross-entropy loss. In YOLO-v3, several logistic regression is trained instead of softmax to normalize the results for getting a multi-class classifier. As because the YOLO-v3 is no longer simple, in moving towards other complex domain, such as the Open Images Dataset. The last layer came up with the prediction of the bounding box as well the class predictions. In YOLO-v3 the number of anchor boxes we used is 9, each of 3 scale. Using the K-means clustering to generate 9 anchors, arranged them in a descending order of the dimension. The bigger 3 anchors are assigned for first scale and respectively the last 6 anchors for the rest of the second and third scale. For the input images we can generate more bounding boxes than the previous model of YOLO-v2. The architecture of YOLO-v3 is as Fig. 2.9

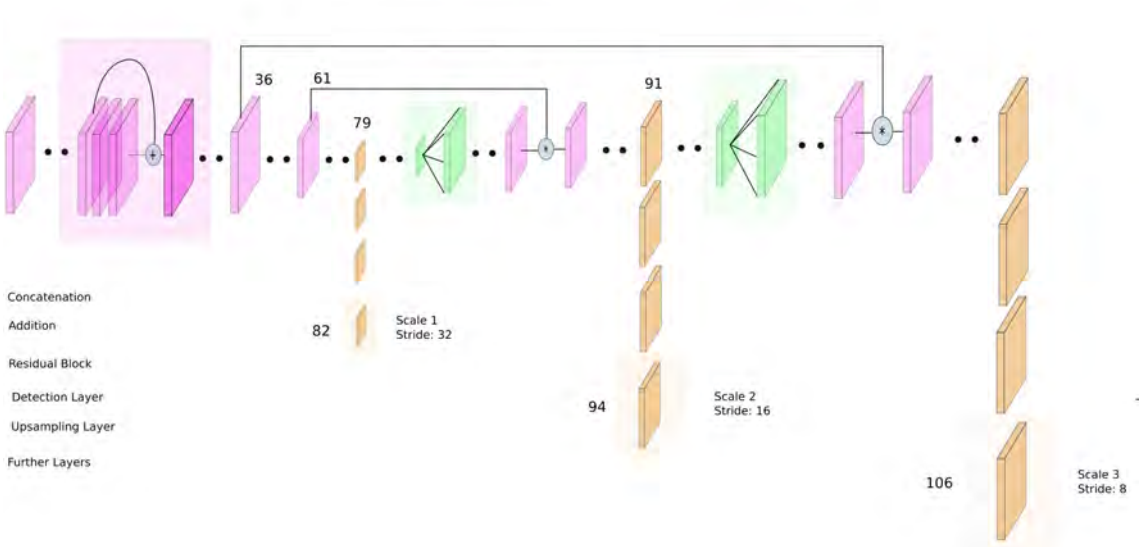


Figure 2.9: YOLO-v3 architecture

As the YOLO v3 uses the predicts boxes at three different scales [28]. The detecting kernel is shaped in the structure of $1 \times 1 \times (B \times (5 \div C)$,where B defines the number of bounding boxes and C stands for the number of classes. Here, the value of b is 3 and the value of c is 80 resulting the kernel size $1 \times 1 \times 255$, with the kernel containing an identical width and height differ from the previous feature map used in YOLO-v2. As the YOLO-v3 make predictions at three scales by downsampling the input images in 32, 16 and 8. The first 81 layers contains the stride of 32 by downsampling image, making the first detection at the 82th layer. Using the 1×1

detection kernel, the first one detection is made which gives us a detection feature map of $13 \times 13 \times 255$ over the input image of 416 x 416 leading to a resultant feature map of 13×13. After that, the layer 79 feature map is going through a few convolutional layers before sampled by $2\times$ to dimensions of 26×26, concatenated then with the layer 61 feature map. This combined feature map is again deal with a few 1×1 convolutional layers in making second detection with layer 61 with the help of 94th layer, with the identification feature map of $26 \times 26 \times 255$. In a similar way, a feature map from layer 91 deal with a few convolutional layers before depth concatenated with the feature map of layer 36. Then, as the same procedure, again a few 1×1 convolutional layers follow to gather the information from the 36th layer, making of the third 106th layer consisting of the feature map of $52 \times 52 \times 255$.

Chapter 3

Proposed Model

In this section, we have discussed about the model we have worked with. The dataset acquisition is an essential section for any model at first glance, to be utilized through which we could generate the sampling process to measure the datasets through different approaches of our respective models. In our proposed model of smoke image detection, we have used a dataset consists of 3000 images [13]. We then divide our datasets into two categories. They are respectively stands for the training and testing datasets. In our proposed model, almost 80% of the data belongs to the training dataset and the rest of the 20% data, is for the testing dataset. The training data will be trained by over those 80% of the datasets and implement those learnings by making predictions running on the rest of the 20% of the testing datasets. The proposed model of our thesis illustrated on Fig. 3.1

3.1 Data acquisition

The dataset acquisition is an essential section for any model at first glance, to be utilized through which we could generate the sampling process to measure the datasets through different approaches of our respective models. In our proposed model of smoke image detection, we have used a dataset consists of 3000 images. They are consisting of fire, smoke and neutral images each one of containing 1000 images.

3.2 Train and test split

Splitting of the datasets is an essential part for a better learning over our datasets. We divide our datasets into two categories. They are respectively signifying the training and testing datasets. In our proposed model, almost 80% of the data belongs to the training dataset and the rest of the 20% data, is for the testing dataset. The training data will be trained over of those 80% of the datasets and implement those learnings by making the predictions running on the rest of the 20% of the testing datasets. Thus, the learning achieved by the training part of the datasets, used for the testing datasets to make predictions as a part of the learning procedures.

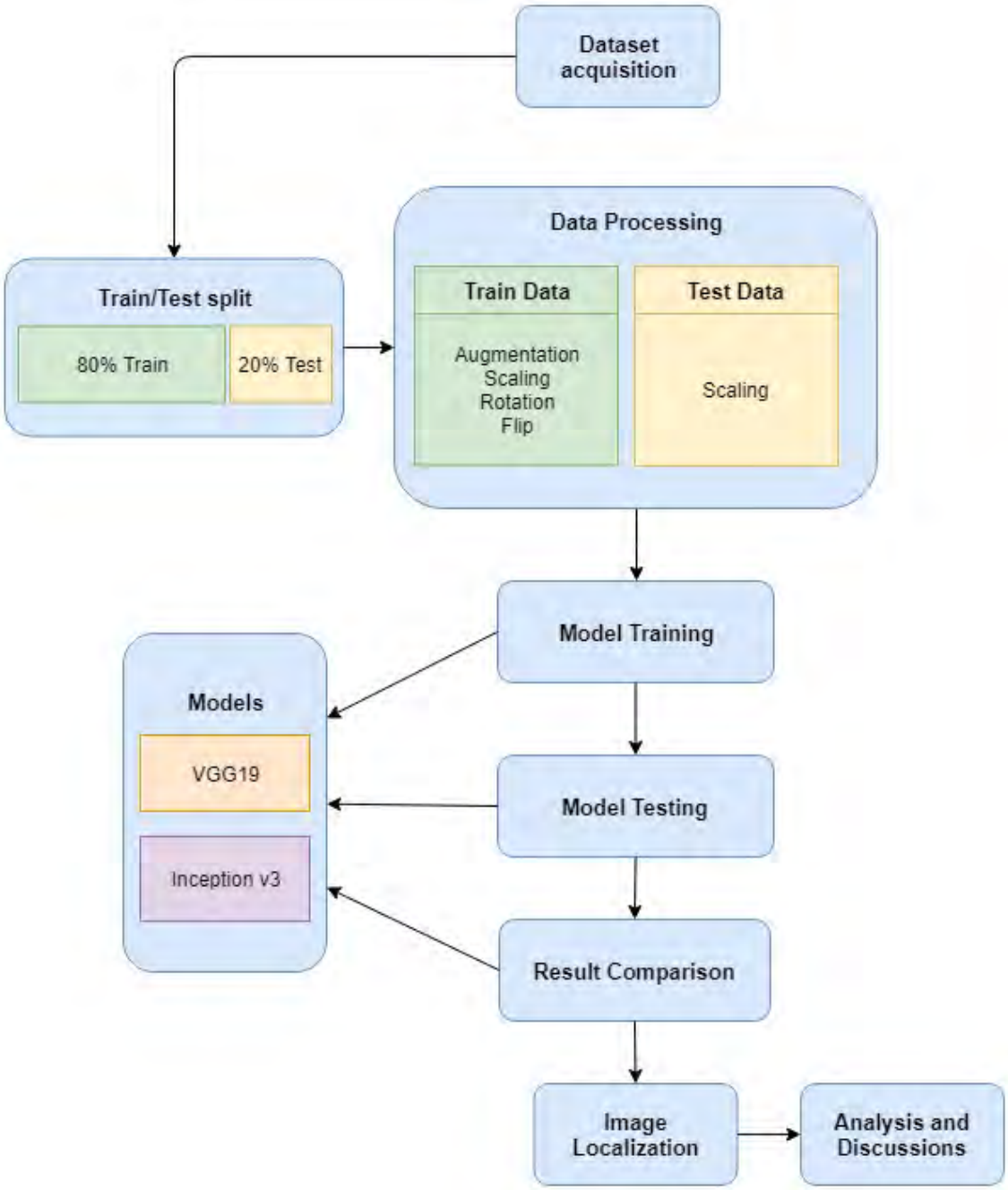


Figure 3.1: Proposed method

3.3 Data processing

In our deep convolutional neural networks, the training of deep learning model of the dataset, makes the dataset stable, resulting in a more efficient way. For data processing, we use the data augmentation part to train our datasets. Added to that, this technique of training the dataset makes the augmentation procedure easier. As a result, to learn on new images by creating variations between the images, improving to be capable of fitting into the models. Resulting in becoming more easier in the procedural phases. We have used the rotation in an angle of 30 degrees of angularity so that the images have a visual stability to process. Moreover, we flipped our images in the x-axis to check whether in shifting the image still trainable in both sides. We exclude the y-axis or the vertical flip as it will make the images in an unstable position. We used the scale of -10 to +10 in both the training and testing part, as a consequence it will provide with a clear visual of the images in the real hands, not tiny or larger in size to make the visual unclear.

3.4 Model training and testing

We have used the Keras deep learning neural network library which provides the ImageDataGenerator class. Utilizing this class, we were able to successfully fit our models: the "VGG-19' and "Inception-v3', by using image data augmentation. Moreover, using this image data augmentation, we accelerated the performance of our models by expanding the training datasets. As the data augmentation creates new training data from the existing training data we provide, the procedure came up with the transformed versions of the images belonging to the same nature or class as the original, providing with be trained. This image transformation includes some certain operations defined as the flipping, zooming, rotating. The zooming procedure was from a ratio of negative ten to positive ten, so that the image does not get blur or big or small to read. The scaling is an essential part for both training and testing dataset. As because, whilst getting in the real hands, the image could be tiny or big causing the problem of image diversity. Our batch size is 20, carrying the dataset worth of 100 batches, each with 20 samples. As a result, the weights of our model will update after each batch of 20 samples. We provide with the input shape of (224, 224 ,3) with the output shape ration of (1:3). The dropout is 0.5 meaning the number of layer outputs to be ignored is 0.5 to approximates our large number of training dataset. We have used the max pooling and lastly flatten the output by combining the previous convolutional layers. In the max pooling procedure, we down-sample the input image for reducing the size dimensionally for allowing the assumptions to be made over the features being contained in the sub-regions. The actual network layer that feeds the output from 20 Proposed Model previous layer to all the neurons used is 1024 in the new model. Resulting the dense 1024, exists 1024 neurons to deal with providing one output each to the next layer.

3.5 Result comparison

After training and testing the dataset through the model of VGG-19 and Inception v3 we have found the comparison between them. The Inception-v3 came up with

a better amount of output in terms of accuracy of 85% whereas the VGG-19 came up with the accuracy of 82%. The number of parameters in Inception-v3 is 2.4M, which greater in number to process the procedures in the model in a better way. Moreover, in parallel phases the model convolutions are destined to work, makes the model more efficient. On the other hand, VGG-19 consists of 19 layers compared to Inception-v3 consists of 42 layers. The VGG-19 runs only 1 convolution of 3×3 over this 19 layers, makes the procedure a bit less efficient with the Inception-v3. Thus, we provide with a better outcome with the model Inception-v3.

3.6 Image localization

In the image localization part, we have used the YOLO-v3 for our smoke image detection with the three different scales of the architecture of 106th layer. It is three steps of procedures. Through the image localization process, first we detect the object which actually signifying the smoke in our smoke images, by putting a bounding box around it. This process is called classification with localization. The height and width of the bounding box are signifying the symbols of h and w. We have a starting point of our bounding box defined as (x, y) and the ending point becomes (x + w, y + h). From this bounding box we came up with the five outputs. They are respectively x, y, h, w and the last one detecting the fire or smoke, known as labelling.

Figure 3.2: Image localization

From the input image the localization will put the bounding box surrounding the smoke object, running the convolutional process through the hidden layers and came up with these five outputs, making the detection complete.

Table 3.1: Image information of Fig. 3.2

	File name	Original Image Size		x (pixel)	y (pixel)	x + w (pixel)	y + h (pixel)
		width (pixel)	height (pixel)				
01	Image001	266	189	18	0	199	109

Chapter 4

Experimental Results and Discussion

In this section, we have discussed about the model we have worked with. Adding to this, some of the basics of deep convolutional neural networks as well. We have came with the two model of the deep convolutional neural networks defining as "VGG-19" and "Inception-v3", that brings out an essential efficiency in the accuracy as well solving the data overfitting problem, that most algorithms faces. Generally, the deep learning neural network performs well in the presence of a good amount of data. The more training data we provide, the better improvement we get. So, we used a dataset worth of 3000 images [13] and respectively we came up with the following results.

4.1 Results

4.1.1 Image Localization

We have successfully detect the smoke in an efficient manner using the deep convolutional neural networks. To detect our smoke images we have used "YOLO-v3" model with the Keras deep learning library. The process works with a deep convolutional neural network called the "Darknet 53". We split our input images to a grid of cells where each cell going to predict the bounding box. Then the convolution through the hidden layers and the backbone network "Darknet53", also known as the upsampling network extracts from the feature map from the input images. Thus, concatenate the output layers of Keras "YOLO-v3" model. The detection of smoke images as per our result is shown on Fig. 4.1 and Fig. 4.2

(a) (b)

Figure 4.1: (a) Input image, (b) Output image

(a) (b)

Figure 4.2: (a) Input image, (b) Output image

Table 4.1: Output of the images showed on Fig 4.1 and Fig. 4.2

	File name	Original Image Size		x (pixel)	y (pixel)	x + w (pixel)	y + h (pixel)
		width (pixel)	height (pixel)				
01	Image001	266	189	18	0	199	109
02	Image002	272	185	0	61	240	124
03	Image003	206	245	0	0	168	200
04	Image004	275	183	91	0	275	160
05	Image005	184	275	0	0	153	233
06	Image006	275	183	0	0	184	120

4.1.2 Training and Testing

In this section, we have shown some of the output samples using the "VGG-19" and "Inception-v3" . Added to that we showed the results with a sufficient amount of rate regarding the accuracy.
In the "VGG-19" model, the accuracy comes with almost 82.33%, which is a good amount of ratio in terms of accuracy. And in "Inception-v3" the number of accuracy comes in a better amount of ratio with almost 84.67%, defining this model even more efficient than the "VGG-19" model.

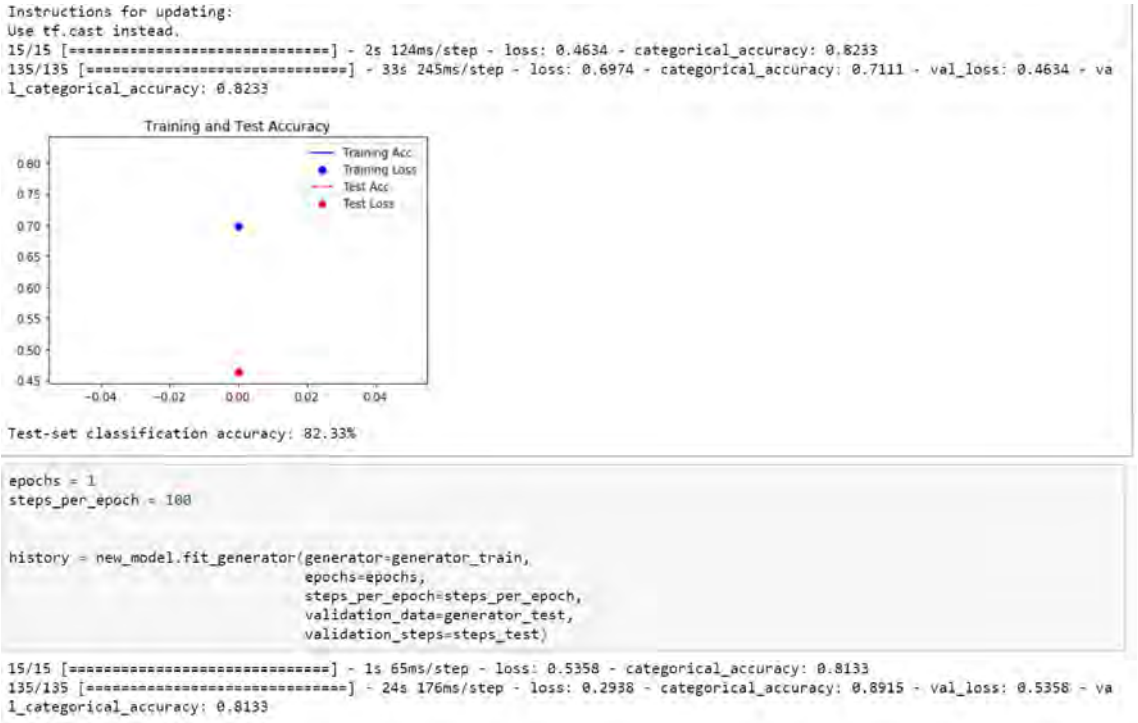


Figure 4.3: Testing result using VGG-19

4.1.3 VGG-19

From the Fig. 4.7, we can see the number of steps per epoch is 100, meaning the number of testing images we have used in our "VGG 19" model. The step loss is 46.34% as shown fig.4.6 with the categorical accuracy of 71.11% .Therefore, the categorical accuracy value is 0.8233 meaning a accuracy of almost 82%, signifying a good amount of ratio.

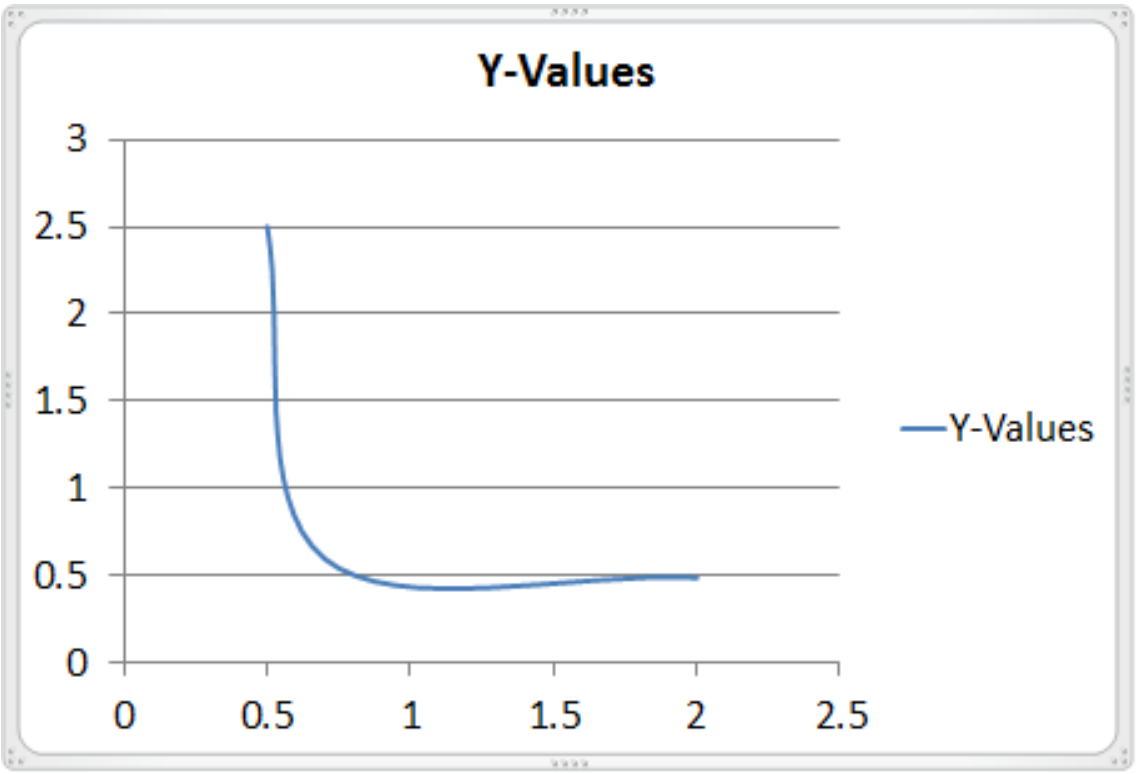


Figure 4.4: Step loss in VGG-19

4.1.4 Inception v3

From the Fig. 4.8 , we can see the number of steps per epoch is as same as the "VGG-19" which is 100, meaning the number of testing images we have used in our "Inception v3" model. The step loss is 42.27% as shown Fig. 4.8 with the categorical accuracy of 71.44% . Therefore, the categorical accuracy value is 0.8467 meaning a accuracy of almost 84.67%, which is even more better than the "VGG-19" model.

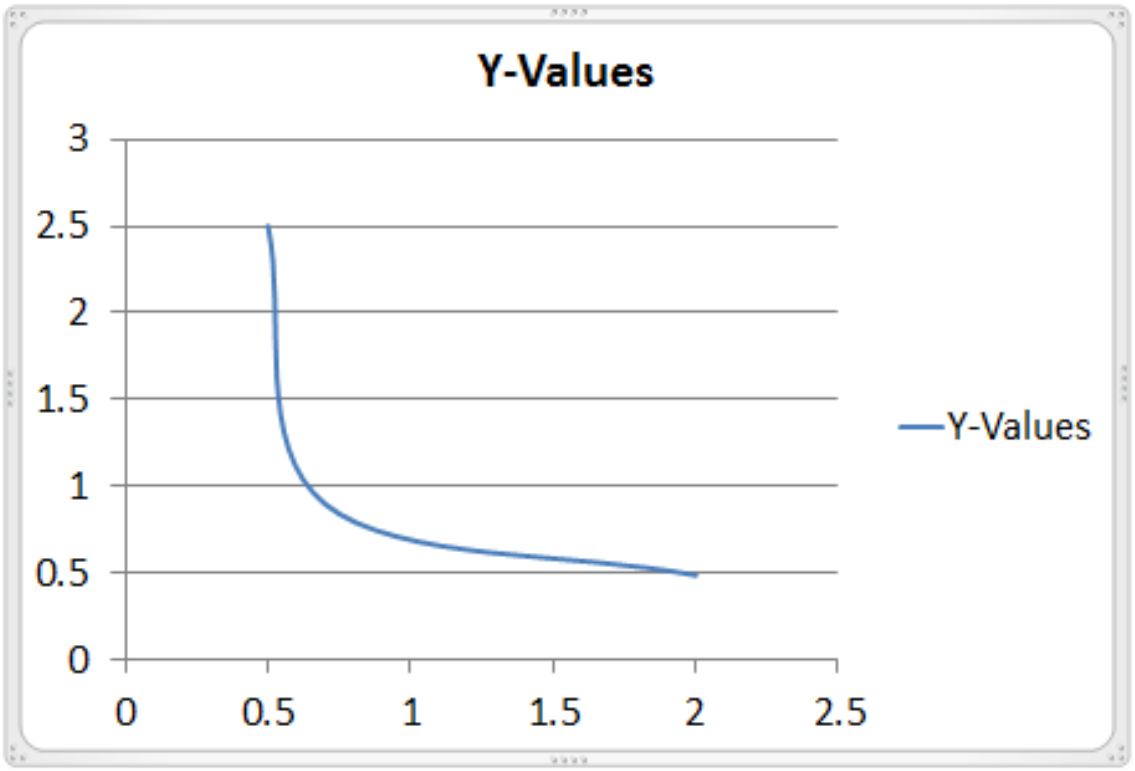


Figure 4.5: Step loss in Inception-v3

The Inception v3 runs on four operational phases. Three convolutional layers and the max pooling. The convolutional layers are respectively the 1×1 , 3×3 and 5×5. Here, the depth reduction works with using the 1×1 convolutional layer. After getting the results from the 3 convolutional layers and as well the max pooling outcomes, we combine or concatenate according to the depth wise. The process of this feature extraction and concatenate must be done before going to the next layer. Thus, flatten the output to a fully connected layer.

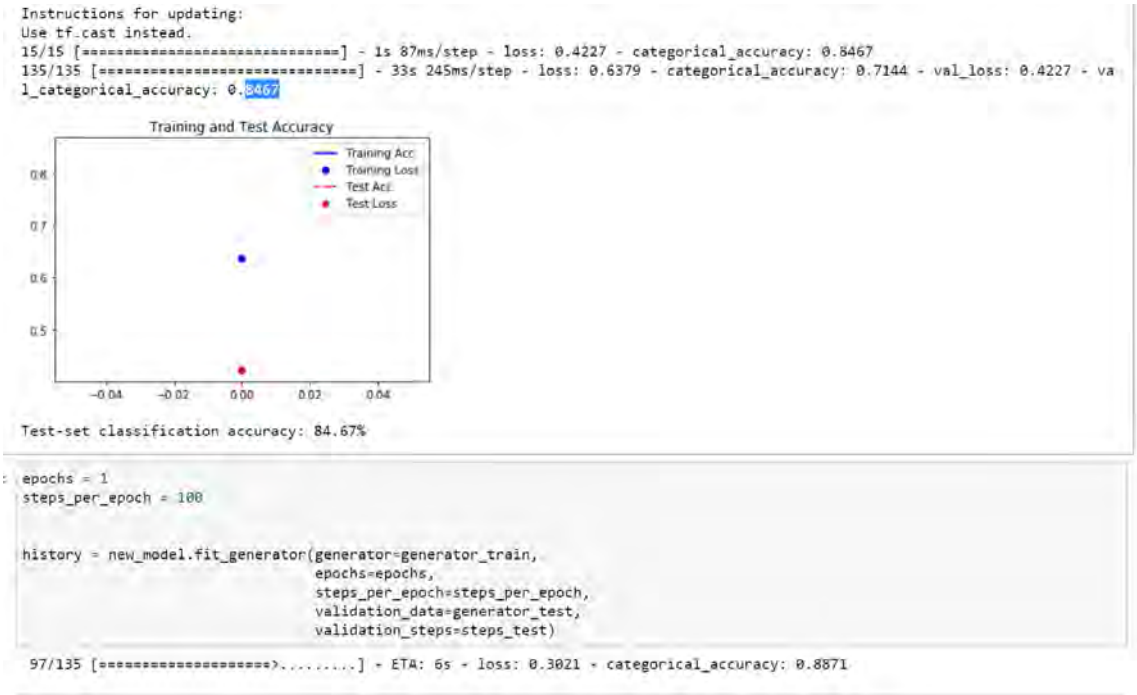


Figure 4.6: Testing result using Inception-v3

4.2 Discussion

In this procedure of smoke detection of image processing, using the deep convolutional neural networks, we have overcome the overfitting problem with a large number of images trained datasets. Moreover, we have gained the accuracy in a more preferable way, with a better amount of percentage which makes our project more efficient. We have also came up with a lower ratio of error rate which is illustrated on Fig. 4.4

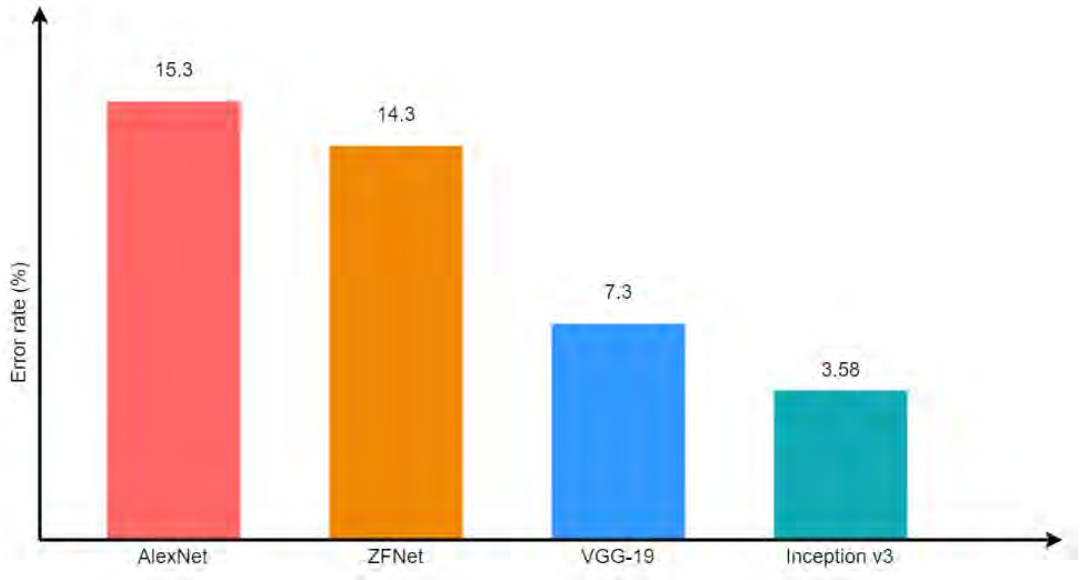


Figure 4.7: Error rates comparison

The AlexNet succeed to improve the results of the ImageNet challenge in 2012 and it has shown with the error rate of 15.3%. Whereas the other models showed lower error rate including the VGG-19 with 7.3% and respectively the Inception v3 with the error rate of 3.58% . As the Inception v3 runs Parallel kernels, it performed with better accuracy than the other models shown in Fig. 4.7
Thus, our proposed models VGG-19 and Inception-v3 provides with a better result than the other models, AlexNet and ZFNet. The comparison of the models are shown in Table 4.2

Table 4.2: Comparison of models

Model	Step loss (%)	Error rate (%)
AlexNet	32.9	15.3
ZFNet	37.5	14.3
VGG-19	46.34	7.3
Inception-v3	42.27	3.58

Chapter 5

Conclusion

5.1 Conclusion

In this project, to have an effective performance, we have utilized the algorithm of deep convolutional neural network for detecting smoke. For a better outcome in terms of accuracy and to get rid of the overfitting problem, we have used a large amount of training dataset of 3000 images and go for testing with the collection of 100 images. The model of our Deep CNN, VGG-19 and Inception-v3 provide us with the accuracy of 82.33% and 84.67%. Our deep convolutional neural network also could extract the features in a form of automatic procedure. Moreover, the results came with a better percentage in terms of accuracy, decreasing the overfitting problems and lower the false alarming rate, in comparison with the previous projects.

5.2 Future Improvements

We have been using the artificial images and a lower number of images which is not sufficient for a better smoke detection. It could be improved, if we could gather more legitimate images, natural images in more pixels than we are using 224×224. Thus, our future motivation is to work upon on our current project, in a sense to extend it to a newer proportion of creating a platform for gathering more artificial images, and developing it towards a legitimate smoke detecting system to upgrade the capability in a more preferable way.

Bibliography

[1] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex", *The Journal of physiology*, vol. 160, no. 1, pp. 106–154, 1962.

[2] J. McGuire, "Control of smoke in building fires", *Fire Technology*, vol. 3, no. 4, pp. 281–290, 1967.

[3] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, *et al.*, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[4] S. G. Wysoski, L. Benuskova, and N. Kasabov, "Evolving spiking neural networks for audiovisual information processing", *Neural Networks*, vol. 23, no. 7, pp. 819–835, 2010.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets. advances in neural information processing systems", 2014.

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

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[9] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks", in *European conference on computer vision*, Springer, 2014, pp. 818–833.

[10] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *arXiv preprint arXiv:1502.03167*, 2015.

[11] H. Jang, H.-J. Yang, D.-S. Jeong, and H. Lee, "Object classification using cnn for video traffic detection system", in *2015 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*, IEEE, 2015, pp. 1–4.

[12] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection", in *european conference on computer vision*, Springer, 2016, pp. 354–370.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[14] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size", *arXiv preprint arXiv:1602.07360*, 2016.

[15] G. Li and Y. Yu, "Visual saliency detection based on multiscale deep cnn features", *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5012–5024, 2016.

[16] A. El-Sawy, E.-B. Hazem, and M. Loey, "Cnn for handwritten arabic digits recognition based on lenet-5", in *International Conference on Advanced Intelligent Systems and Informatics*, Springer, 2016, pp. 566–575.

[17] M. Simon, E. Rodner, and J. Denzler, "Imagenet pre-trained models with batch normalization", *arXiv preprint arXiv:1612.01452*, 2016.

[18] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[19] M. Z. Alom, M. Hasan, C. Yakopcic, T. M. Taha, and V. K. Asari, "Improved inception-residual convolutional neural network for object recognition", *Neural Computing and Applications*, pp. 1–15, 2017.

[20] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation", *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[21] S. Chaib, H. Yao, Y. Gu, and M. Amrani, "Deep feature extraction and combination for remote sensing image classification based on pre-trained cnn models", in *Ninth International Conference on Digital Image Processing (ICDIP 2017)*, International Society for Optics and Photonics, vol. 10420, 2017, p. 104203D.

[22] M. B. Jensen, K. Nasrollahi, and T. B. Moeslund, "Evaluating state-of-the-art object detector on challenging traffic light data", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 9–15.

[23] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning", in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[24] C. N. Vasconcelos and B. N. Vasconcelos, "Increasing deep learning melanoma classification by classical and expert knowledge based image transforms", *CoRR, abs/1702.07025*, vol. 1, 2017.

[25] J. Du, "Understanding of object detection based on cnn family and yolo", in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1004, 2018, p. 012 029.

[26] R. Huang, J. Pedoeem, and C. Chen, "Yolo-lite: A real-time object detection algorithm optimized for non-gpu computers", in *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, 2018, pp. 2503–2510.

[27] W. Liu, L. Ma, J. Wang, *et al.*, "Detection of multiclass objects in optical remote sensing images", *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 5, pp. 791–795, 2018.

[28] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement", *arXiv preprint arXiv:1804.02767*, 2018.

[29] X. Zhang, W. Yang, X. Tang, and J. Liu, "A fast learning method for accurate and robust lane detection using two-stage feature extraction with yolo v3", *Sensors*, vol. 18, no. 12, p. 4308, 2018.

[30] S. Rahman, "Alarming rise in fire incidents", *The Financial Express*, Feb. 2019. [Online]. Available: https://thefinancialexpress.com.bd/national/alarming-rise-in-fire-incidents-1551336199.