A Master of Science

Forest Fire Detection Using Convolutional Neural Networks and Attention Mechanisms

The Graduate School of the University of Ulsan

Department of Electrical, Electronic and Computer Engineering

QUY QUYEN HOANG

May 2023

Forest Fire Detection Using Convolutional Neural Networks and Attention Mechanisms

Under the supervision of

Prof. Hoon Oh

A Thesis Submitted to

the Graduate School of the University of Ulsan

In partial Fulfillment of the Requirements

for the Degree of

A MASTER OF SCIENCE

by

QUY QUYEN HOANG

Department of Electrical, Electronic and Computer Engineering

University of Ulsan, Korea

May 2023

Forest Fire Detection Using Convolutional Neural Networks and Attention Mechanisms

This certifies that

this thesis of QUY QUYEN HOANG is approved.

Committee chair: **Prof. Seok-Hoon Yoon** (sign)

Committee member: **Prof. Dae-Hwan Kim** (sign)

Committee member: **Prof. Hoon Oh (Advisor)** (sign)

Department of Electrical, Electronic and Computer Engineering

University of Ulsan, Korea

May 2023

Acknowledgments

I would like to express my sincere appreciation to my advisor, Hoon Oh professor, for giving me a chance to join the Ubiquitous Computing (Ubicom) lab. During two years working here, I have gained a lot of experience in both doing research and doing real projects. I appreciate his support, encouragement, and advice in both studies and life.

I would also like thank my friends for their sincere friendship, enthusiastic help, and cheerfulness during my stay in Ulsan.

I am deeply indebted to my family for their love and support. This is the biggest motivation and encouragement for me throughout my master journey.

VITA

Quy Quyen Hoang received the B.S. degree in mechatronics engineering from the University of Science and Technology, University of Da Nang, Da Nang City, Vietnam, in 2019. He is currently working toward the M.S. degrees in the department of IT Convergence, University of Ulsan, Ulsan, Korea. His current research interests include computer vision and deep learning.

Forest Fire Detection Using Convolutional Neural Networks and Attention Mechanisms

by

Quy Quyen Hoang

Submitted in partial fulfillment of the requirements for the degree of

Master of Science (Electrical, Electronic and Computer Engineering)

May 2023

Abstract

This study explores a way of detecting smoke plumes effectively as the early signs of forest fire. Convolutional neural networks (CNNs) have been widely used for forest fire detection; they were not customized or optimized for smoke characteristics. This paper proposes a CNN-based forest smoke detection model featuring a novel backbone architecture that can increase detection accuracy and reduce computational load. The proposed backbone detects the plume of smoke through different views using different sized kernels, it can better detect smoke plumes of different sizes. The conventional convolution of square kernels is decomposed into the depth-wise convolution of coordinate kernels to not only can better extract the features of smoke plumes spreading along the vertical dimension but also reduce the computational load. Attention mechanism was applied to allow the model to focus on important information while suppressing less relevant information. Experiments show that our model outperforms other popular ones by achieving detection accuracy of up to 52.9 average precision (AP) and reduces the number of parameters and giga floating-point operations (GFLOPs) significantly compared to the popular models.

Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Overview of forest fire detection

Forest fires often cause enormous damage to human life and the environment. Many great forest fire disasters can be found in history like the Camp Fire in California, in 2018, that claimed 85 lives and destroyed 153,336 ha of forest. Even worse, global warming has led to increased temperature extremes and longer dry periods, which increase the risk of forest fires. Thereby, the number of forest fires was increasing. According to the annual Wildfires Report from the National Centers for Environmental Information, forest fire in United States burned over 7 million acres of wildland in 2021 [1].

A major reason for the great damage is that forest fires can spread quickly, making them difficult or even impossible to extinguish before being detected. This study considers the evolution of the existing vision-based model to effectively detect forest smoke that is the early sign of a forest fire. The proposed model is based on convolutional neural networks (CNNs) and attention mechanism and focuses on increasing the accuracy and reducing the computational complexity of smoke plume detection.

1.2 Related work

According to survey papers [2-4], many forest fire detection methods have been proposed. Early methods relied on fire lookout towers that often rely on tools like the Osborne Fire finder [5]; however, they were not effective due to continuous human intervention and potential for human error. Some evolved methods used sensors that can detect signs of a fire outbreak, such as rising temperature, smoke, flames and lack of oxygen; they had the difficulty of collecting data reliably from the sensors deployed in a vast forest area [6]. In addition, they suffered from the problem of a fire alarm that does not work until

the parameter values for fire detection exceed their respective preset thresholds. Recently, the direction of research has been shifting toward a vision-based approach that relies on artificial intelligence [7].

1.3 Evaluation methodology

This study introduces a forest smoke detection model featuring a new backbone architecture that is customized to increase the accuracy of smoke detection and reduce computational complexity. The proposed backbone is designed to effectively extract smoke features. First, it extracts features of objects through different views using different kernel sizes. This allows the model to better detect smoke plumes of different scales. Second, it decomposes the conventional convolution of a square kernel into the depth-wise convolutions of coordinate kernels to not only better extract the features of smoke plumes that spread along vertical dimension, but also reduce the computational load. Finally, it employs an attention mechanism that focuses on important features in the image while suppressing irrelevant features. As a result, the proposed model could achieve up to 52.9 average precision (AP), which far exceeds the accuracy of other models such as YOLOv3 [18], RetinaNet [25], Faster-RCNN [20], and SSD [26], while reducing the number of parameters and GFLOPs significantly

1.4 Thesis Outline

The thesis consists of six chapters structured as follows:

Chapter 1 presents the fundamental knowledge about forest fire detection, problems in designing a forest fire detection of previous approaches. Then, the evaluation methodology and organization of the thesis are given.

Chapter 2 discusses the background of forest fire detection model and presents our motivation in the design of a new forest fire detection model.

Chapter 3 provides a survey of forest fire detection that includes traditional, and CNN based approaches.

Chapter 4 introduces a detailed description of the proposed forest fire detection model.

Chapter 5 provides the performance evaluation of the proposed forest fire detection model with the experiments setups and results on the collected dataset.

Finally, Chapter 6 gives conclusions on the thesis and the future research direction.

Chapter 2. Background

2.1 Convolution neural networks

Convolutional Neural Networks (CNNs) are a type of neural network commonly used in deep learning for image recognition, classification, and segmentation tasks. They are inspired by the structure and function of the visual cortex of the brain, where neurons are arranged in a hierarchical manner to extract increasingly complex features from sensory input.

2.1.1 Convolution

The basic building block of a CNN is a convolutional layer, which applies a set of learnable kernels to the input image to extract features. The filters slide over the input image, performing a dot product operation at each location to generate a feature map. Multiple filters can be applied to the same input to extract different features.

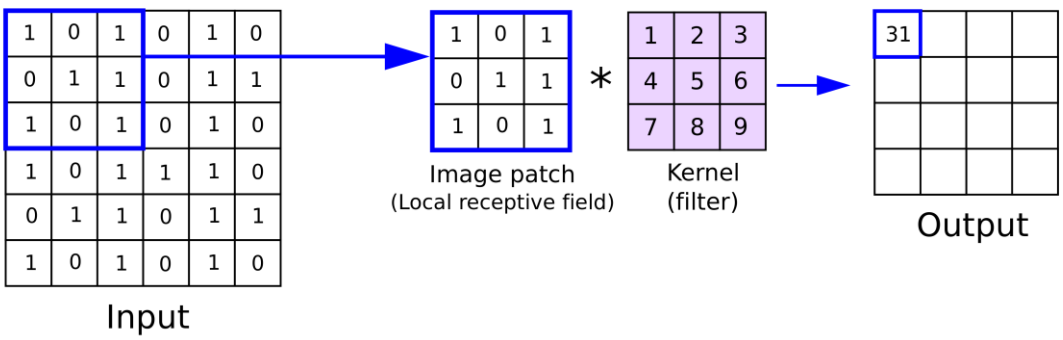


Figure 2.1. An example of the convolution

2.1.2 Pooling

Pooling is a technique used to reduce the spatial size of the feature maps while retaining the most important information. There are several types of pooling layers, including max pooling, average pooling, and L2 pooling. In max pooling, the maximum value in each pooling window is selected and passed on to the next layer. In average pooling, the average value in each window is computed. L2 pooling takes the root of the sum of the squared values in each window.

One of the benefits of pooling is that it helps to make CNN more robust to small translations of the input. For example, if a small object is located in a slightly different position in two images, pooling can help CNN to recognize that the same object is present in both images. Pooling can also help to reduce overfitting by enforcing a form of spatial regularization on the feature maps.

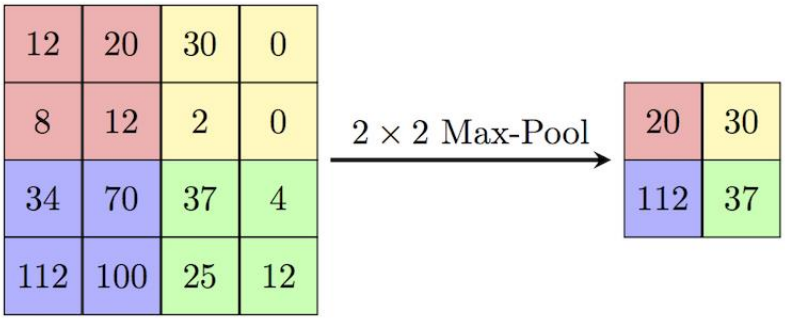


Figure 2.2. An example of the max-pooling

2.2 Forest fire detection model

The considered forest fire detection model consists of three modules: Backbone, Neck, and Head, as shown in Figure 2.3. Backbone consists of four stages labeled S_1, S_2, S_3 and S_4 and each stage generates *one stage feature map* (i.e., the last generated feature map from the in-stage convolutional network) where the stage feature map of stage S_i is constructed by taking the stage feature map of S_{i-1} as input and going through convolutional layers while the stage feature map of S_1 is constructed from the input image. Early stages tend to capture low-level information like edges, corners, etc., while later stages capture high-level or specific information.

Neck has five levels labeled P_1, P_2, P_3, P_4, and P_5 and each level generates *one level feature map* (i.e., the last generated feature map from the in-level convolutional network) where the level feature map of level P_3 is built by applying convolutions to the stage feature map of S_4. The level feature map of P_2 is created by up-sampling the level feature map of P_3 and adding it to the stage feature map of S_3. Note that the level feature map of P_3 is also used to generate the level feature map of P_4. The level feature map of P_1 is created similarly. Two more level feature maps of P_4 and P_5 are constructed by down-sampling those of P_3 and P_4, respectively to have more abundant features. In this way, having a multi-level pyramidal structure, Neck can not only balance the information of different feature maps, but also help the model easily detect objects of different scales.

Head includes two specific tasks: object classification and bounding box regression. Each one is represented as a small convolutional network that consists of five serially connected convolution layers. A class feature map and a box feature map are represented by A×W×H (Anchors × Width × Height) and by 4A×W×H, respectively, where 4 indicates 4 relative offset values between the anchor and the ground truth box. The former

is used to determine the probability of the presence of a specific object at each spatial position while the latter is used to regress the 4 offset values from each anchor box to a nearby ground truth object.

In this study, we focus on developing a new Backbone which is suitable for extracting features from the images of fire and smoke in the forest.

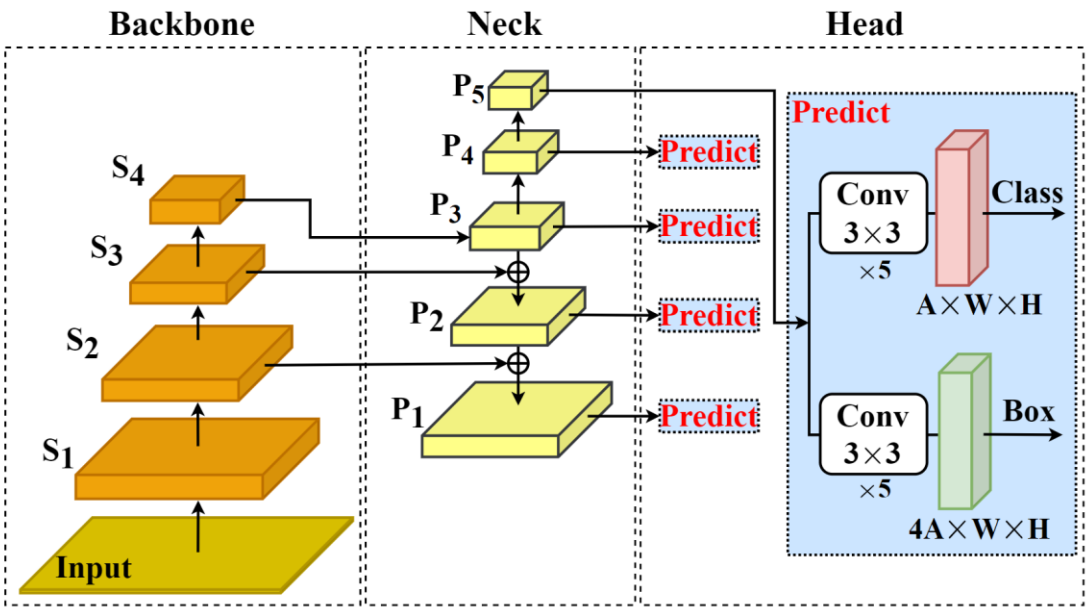


Figure 2.3. The architecture of the forest fire detection model

2.3 Motivation

Backbone plays an important role in improving the accuracy of object detection since it generates feature maps of objects. In addition, it can incur significant computational load since it deals with a lot of convolutional layers.

Recent forest fire detection models actually used well known backbones designed using the ImageNet [27] data set. However, ImageNet does not have smoke and fire classes even though it is a large dataset with over one million images and one thousand classes. This means that those backbones are not suitable for forest fire detection models. Moreover, researchers have been trying to design backbones with more layers and large kernels to extract more information from ImageNet. However, this can demand more computational load. The proposed backbone addresses the two problems as follows.

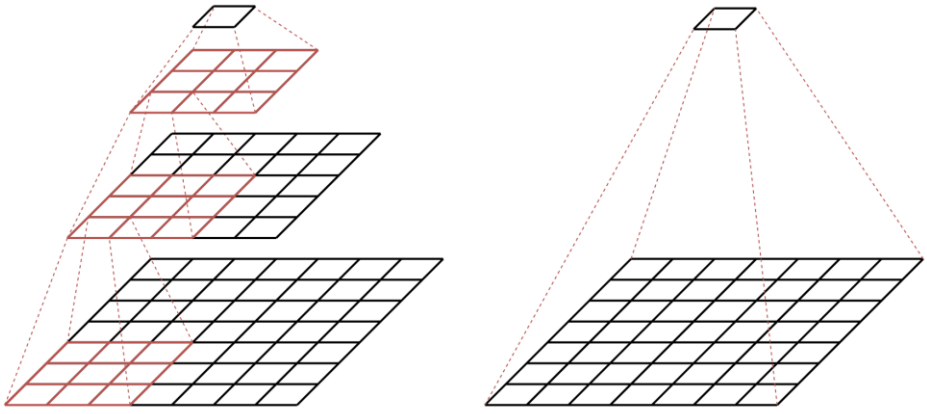
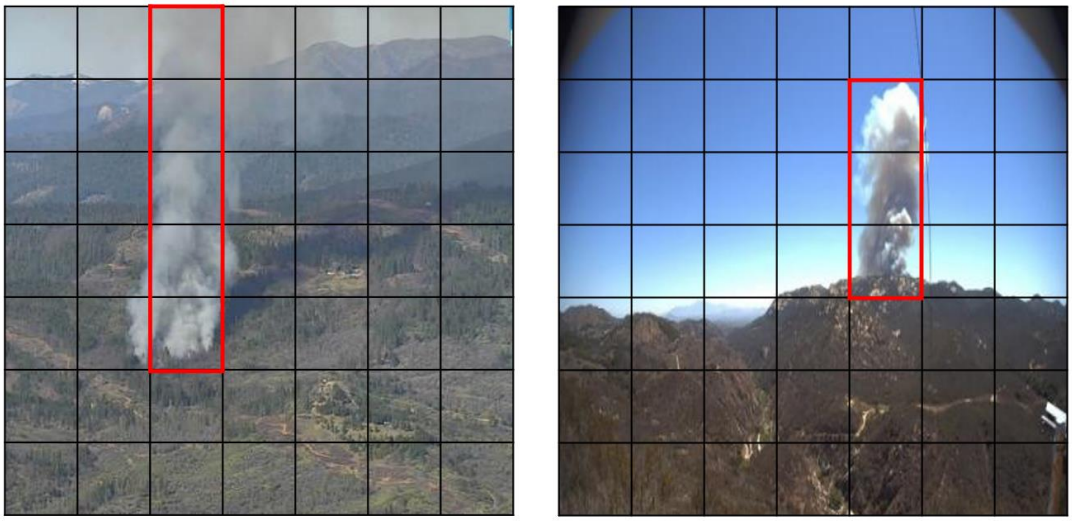


Figure 2.4. The same receptive field of using of three 3×3 kernels and one 7×7 kernel

First, a large size kernel can sometimes help the model speed up object detection by capturing more pixels in one step, but it produces too many numbers of parameters. Therefore, it is replaced by multi smaller size kernels that can capture pixels from the

same receptive field (Figure 2.4) while using fewer parameters. Second, it decomposes the conventional convolution of n×n kernels (Conv n×n) into the depth-wise convolution of n×1 (DWconv n×1) and 1×n kernel (DWconv 1×n) to better extract the features of smoke plumes that spread along vertical dimension (Figure 2.5). The decomposition also helps the model reduce the number of parameters. Third, it extracts features of objects through different views using different kernel sizes (Figure 2.5). This allows the model to better detect smoke plumes of different scales.



(a) DWconv 5×1 (b) DWconv 3×1

Figure 2.5. Depth-wise convolutions of multi kernel size

Fourth, it employs an attention mechanism that focuses on important features in the image while suppressing irrelevant features to increase detection accuracy. Finally, the

model was trained and improved using a dataset containing over 4,000 forest smoke images.

Chapter 3. Survey of Forest Fire Detection

3.1 Traditional approaches

Existing vision-based approaches can be divided into two categories: image processing approaches and CNN-based ones. The former relied on image processing techniques to explore fire and smoke characteristics such as color, shape and motion. The authors in [8], [9] and [10] used RGB, YCbCr, and Lab color models, respectively, to extract fire and smoke pixels. The authors in [11] used wavelet and fast Fourier transform methods to analyze the contours of the fire area in videos. The authors in [12] combined the properties of color, shape, and motion using a multi-expert framework to increase detection accuracy. One recent approach utilized background subtraction and color segmentation to detect regions containing motion [13]. Since these approaches do not use high computational power, they can be used for devices with limited computational power, such as drones or surveillance cameras. However, to achieve a reasonable level of accuracy, they require careful image pre-processing steps and may need the use of different feature extraction algorithms for forest fire images in different situations.

3.2 CNN-based approaches

In contrast, CNN-based approaches use deep learning techniques to automatically extract features from different images. The authors in [14] proposed a lightweight forest fire detection model by replacing the backbone network of YOLOv4 [15] with MobileNetv3 [16]. This method could reduce the computational load greatly, but with reduced detection accuracy. The authors in [17] used YOLOv3 [18] to detect forest fires with the utilization of UAV (unmanned aerial vehicle) that can capture high-resolution videos and images. However, it did not perform well against small smoke or fire. Another approach [19] tried to detect forest smoke using Faster R-CNN [20]; They improved the

accuracy to some extent, but the experiments were limited because they did not use diverse forest fire data sets. The authors in [21] employed Inceptionv3 [22] to train satellite images for forest fire detection. This satellite-based approach can capture large fire images only after the fire has spread to a large area. Furthermore, since Inceptionv3 only returns a fire or non-fire decision without boxing the fires, it requires an extra step to determine the regions of the fires, which takes time and effort. One recent approach [23] used R-CNN [24] for forest fire detection. The high computational complexity of this model hinders its portability to monitoring devices. In summary, the existing approaches have limited improvement in detection accuracy because it uses popular models such as the YOLO series and Faster R-CNN as they are. Moreover, they require a high computational load.

Chapter 4. Methodology

4.1 Backbone architecture

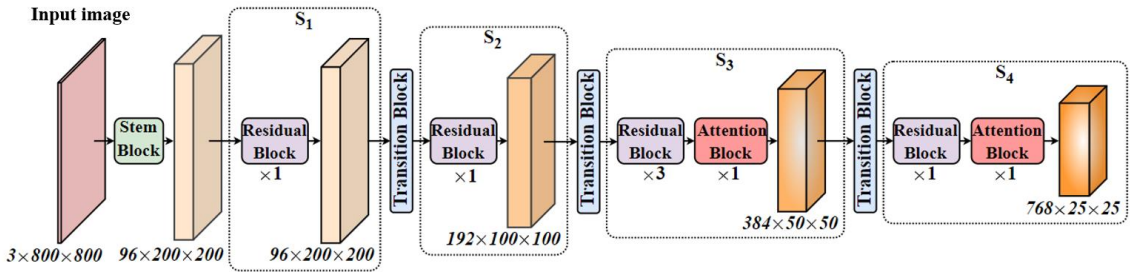


Figure 4.1. Backbone architecture

The structure of the proposed Backbone is shown in Figure 3.1. The features of input data are comprehensively extracted through a 4-stage hierarchical structure, each stage consisting of one or more residual block(s). Note that the third stage comprises three residual blocks followed by one attention block, while the fourth stage has one residual block followed by one attention block. The design aims to enhance the Backbone's ability to extract smoke features while minimizing computational requirements.

4.1.1 Stem Block

The structure of the stem block is illustrated in Figure 3.2. The stem block is utilized to quickly reduce the spatial dimension of the input without losing feature information. To perform this task, the previous studies have often used a large kernel size, such as 7×7, thus requiring a larger number of parameters. In our design, three 3×3 kernels with stride size

of 2, 1, and 1 are stacked to replace a single 7×7 kernel. It can fully extract features from input by having the same effective receptive field, but fewer parameters. Batch Normalization (*BN*) and Rectified Linear Unit (*ReLU*) are additionally applied to the output of each convolution layer to speed up and stabilize the training process. At the end of the stem block, one 3×3 max pooling is applied to reduce the size of the feature map. The spatial dimension across stages of input data will be reduced sixteen times from 800×800 to 200×200 through the stem block.

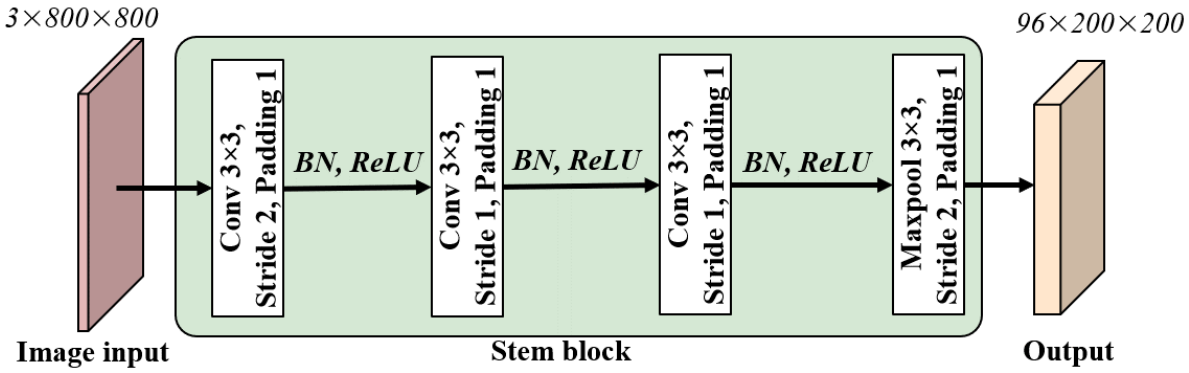


Figure 4.2. Stem block

4.1.2 Residual block

The structure of the residual block is illustrated in Figure 3.3. The residual block plays the main roles in the smoke feature extraction of forest fire model. This block is constructed by using the residual structure to avoid the vanishing gradient problem and stabilize the optimization. The feature map from the previous layer is split into four sub-features, and then the sub-features are fed into different operators that can capture a wide range of receptive fields to improve detection of objects of different sizes. Two upper branches are constructed by decomposing the conventional convolution of a square kernel into depth-wise convolutions of coordinate kernels. This way can relax the computational complexity while enhancing the capability of detecting the vertical spread smoke plumes. The remaining branch is used to mix information along spatial and channel dimensions.

Specifically, depending on the number of channels in the input, The feature map from the previous layer is split into four small feature maps along the channel dimension and each map is processed along different convolution layers. One of the two upper branches uses two depth-wise convolutions, as DWconv 1×5 and DWconv 5×1, and another as DWconv 1×3 and DWconv 3×1. Note that each branch uses the coordinate kernels of different sizes instead of one large square kernel. This way of decomposed processing allows the backbone module to better extract the features of the smoke plumes spreading vertically. It allows for reducing the number of parameters and GFLOPS. The third branch can enhance feature extraction from small smoke plumes by using DWconv 1x1 with a small-sized kernel. The last branch sequentially applies one max pooling 3×3 and one DWconv 1×1. By taking the maximum value within each pooling region, max pooling retains the most important features while discarding less important or noisy features. One DWconv 1×1 is applied on the output of the max pooling layer that can help to perform

channel mixing, which can improve the accuracy of the model. The outputs of all branches are concatenated along the channel dimension to produce a fine-grained feature map. Then, two point-wise convolution (PWconv) 1×1s are added serially to mix the information along channel dimension. ReLU activation function in between them is utilized to reinforce the feature non-linearing on large space via expand ratio of 4. Finally, the output through the above layers is element-wise added to the residual branch to produce the feature map for the next layer. This addition allows residual block to keep the previous layer features that help the model avoid the vanishing gradient problem [28] and stabilize the optimization.

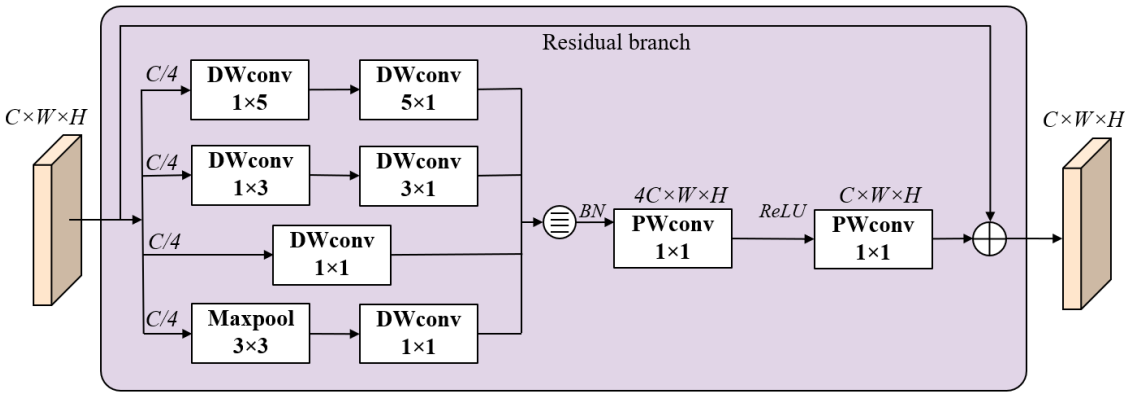


Figure 4.3. Residual block

4.1.3 Transition block

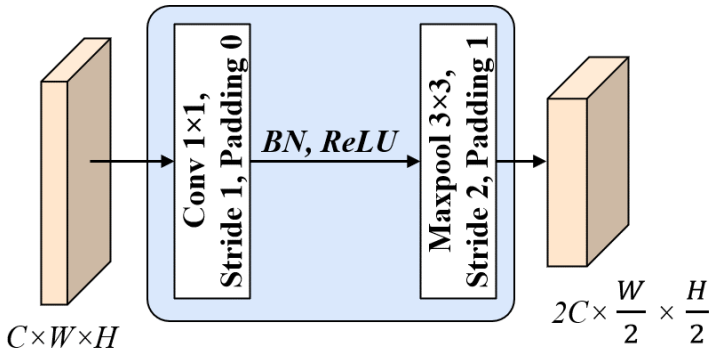


Figure 4.4. Transition block

The structure of transition block is illustrated in Figure 3.4. The transition block is used to shrink the size of feature map between two adjacent stages. The conv 1×1 is utilized to double the number of channels and then, followed by 3×3 max pooling to reduce the spatial dimension by half. This way helps to shrink the size of the feature map without losing information while saving the number of required parameters.

4.1.4 Attention block

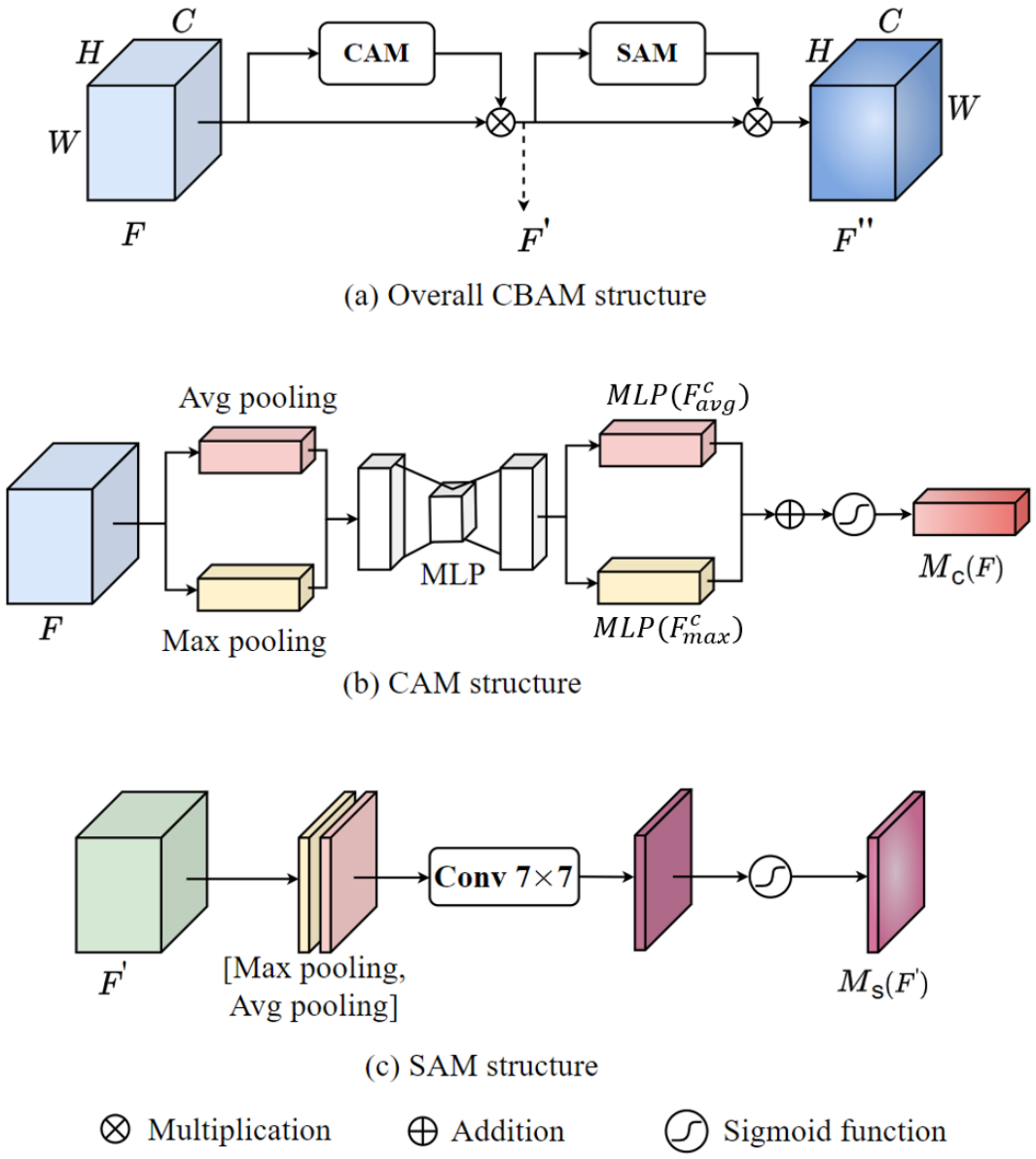


Figure 4.5. CBAM architecture

The attention block is added to the outputs of residual blocks of stage 3 and stage 4 in Fig. 3a. The structure of the attention block is illustrated in Figure 3.5(a). The attention mechanism helps the model to focus on important features of the image while suppressing

irrelevant ones. In this paper, the Convolution Block Attention Module (CBAM) [28] is employed, where CBAM consists of two main components: Channel Attention Module (CAM) as shown in Figure 3.5(b), and the Spatial Attention Module (SAM) as shown in Figure 3.5(c). CAM is designed to help the model focus on the most relevant channels in feature maps. SAM, on the other hand, is designed to capture spatial dependencies in feature maps. These two attention modules compensate for each other's weaknesses, making the model focus on the important features of the feature map.

Let $\boldsymbol{F}$ and $\mathbb{R}^{\boldsymbol{C}\times\boldsymbol{H}\times\boldsymbol{W}}$ represent the input feature map and a set of all possible feature maps of the target object, respectively such that $\boldsymbol{F} \in \mathbb{R}^{\boldsymbol{C}\times\boldsymbol{H}\times\boldsymbol{W}}$. Input feature map $\boldsymbol{F}$ is processed by CAM to produce channel attention weight $\boldsymbol{M_c(F)}$ as detailed in Figure 3.5(b). Then, the refined feature map $\boldsymbol{F'}$ is obtained by performing the element-wise matrix multiplication between $\boldsymbol{M_c(F)}$ and $\boldsymbol{F}$ to redistribute the information in the input feature map $\boldsymbol{F}$ along the channel dimension as follows:

$$\boldsymbol{F' = M_c(F) \otimes F} \qquad \textbf{(4.1)}$$

Referring to Figure 3.5(b), CAM uses average-pooling and max-pooling along spatial dimension to aggregate the spatial information, which generate the average-pooled features $\boldsymbol{F^c_{avg}}$ and the max-pooled features $\boldsymbol{F^c_{max}}$, respectively. These two features are then passed to the Multilayer Perceptron (MLP) to generate two channel attention maps, $\boldsymbol{MLP(F^c_{avg})}$ and $\boldsymbol{MLP(F^c_{max})}$, which are merged using element-wise addition. Finally, the sigmoid function, denoted by σ, is applied to produce the channel attention weight $\boldsymbol{M_c(F)}$ as follows:

$$\boldsymbol{M_c(F) = \sigma\big(MLP(F^c_{avg}) \oplus MLP(F^c_{max})\big).} \quad \textbf{(4.2)}$$

The refined feature map $\boldsymbol{F'}$ is then fed into SAM module to generate spatial attention weight $\boldsymbol{M_s(F')}$ as detailed in Figure 3.5(c). Then, $\boldsymbol{M_s(F')}$ is multiplied with feature map $\boldsymbol{F'}$ to refine the feature map $\boldsymbol{F'}$ in the spatial dimension, thereby producing the feature $\boldsymbol{F''}$ as follows:

$$\boldsymbol{F'' = M_s(F') \otimes F',} \quad \textbf{(4.3)}$$

referring to Figure 3.5(c), SAM also uses both max-pooling and average-pooling along channel dimension to generate two features $\boldsymbol{F^s_{avg}}$ and $\boldsymbol{F^s_{max}}$ that represent the aggregated channel information. Then, they are concatenated and mixed using 7x7 convolution, $\boldsymbol{\mathcal{F}^{7\times7}}$, to produce a spatial attention map. Finally, the sigmoid function $\boldsymbol{\sigma}$ is applied to produce the spatial attention weight $\boldsymbol{M_s(F)}$ as follows:

$$\boldsymbol{M_s(F) = \sigma\big(\mathcal{F}^{7\times7}(F^c_{avg}\,; F^c_{max})\big),} \quad \textbf{(4.4)}$$

In conclusion, our model focuses on the important features and suppress the irrelevant ones along both channel dimension and spatial dimension, by applying channel attention weight (via CAM) and spatial attention weight (via SAM), respectively, to refine input feature.

4.2 Neck architecture

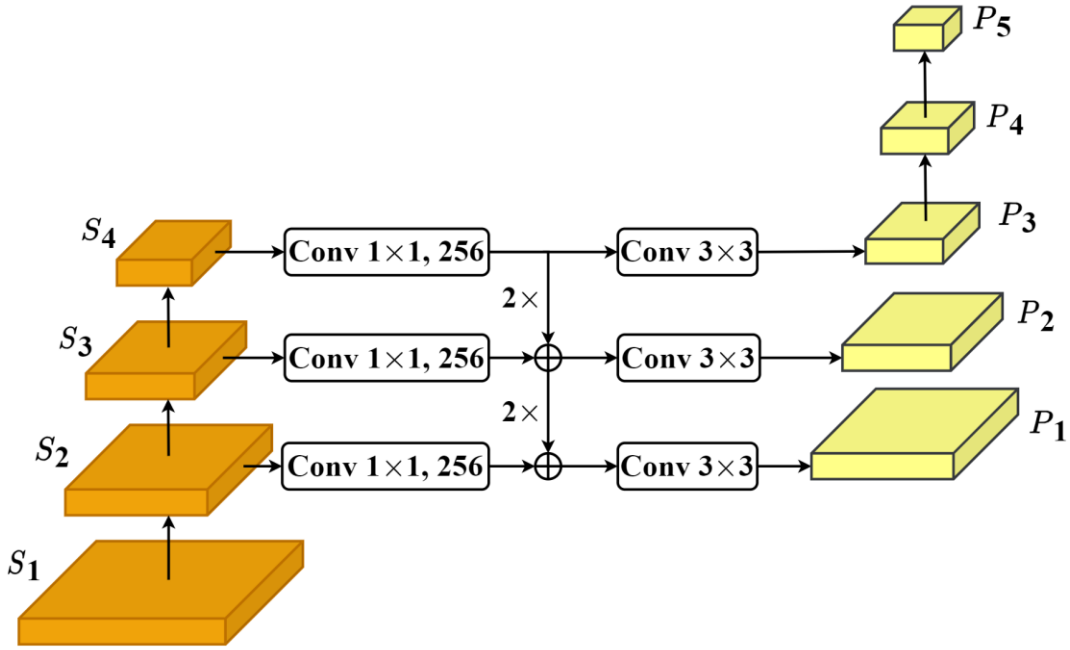


Figure 4.6. Neck architecture

As illustrated in Figure 3.6, the Neck model based on Feature Pyramid Network [28] consists of multiple levels so that the model can easily detect objects of different scales as well as balance the information via multiple stages. Specifically, the Neck consists of five levels labeled $P_1, \ldots, P_5$. Initially, the feature maps from S_2 to S_4 in Backbone is reduced by shrinking the number of channels to 256 based on 1×1 convolution. The feature map at P_3 level is produced by directly applying 3×3 convolution to the feature map of S_4. Meanwhile, the feature map at P_2 are created by up-sampling the feature map at P_3 through nearest algorithm and then adding it to the corresponding feature map (S_3) in Backbone. The same principle is applied for P_1. Note that stage S_1 is not used because it is computationally expensive. The P_4 and P_5 levels are added as in [25] by down-sampling the feature map at

P_3 by 1/2 and 1/4, respectively, using 3×3 conv with stride 2. This addition allows Neck to create more levels of pyramids and helps the model better detect larger objects.

4.3 Head architecture

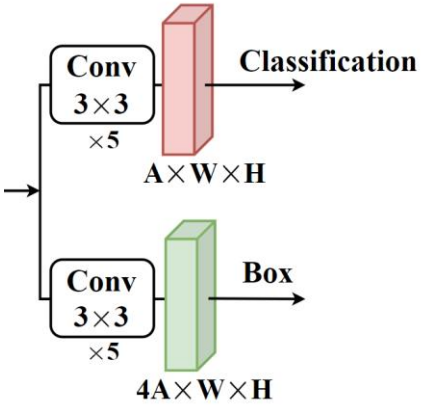


Figure 4.7. Head architecture

The Head of our model is borrowed from [25]. Specifically, it requires multi-task learning using two task branches: object classification branch and bounding box regression branch. The classification branch predicts the probability of object presence at each spatial position for each of the A anchors and K object classes. It is a series of 3×3 convolutional networks connected to each FPN level where the output is a class feature map denoted by A×W×H (Anchors × Width × Height), parameters of this branch are shared across all pyramid levels. We use K = 1 and A = 9 in most experiments.

The bounding box regression also consists of a series of 3×3 convolutional networks connected to each FPN level where the output is a bounding box feature map denoted by 4A×W×H (Anchors × Width × Height), where 4 represent the four relative offset values

between the anchor and the ground truth box. For details on the Head, refer to the paper [25].

4.4 Loss function

The focal loss (*FL*) function [25] is used in the model since it is suitable for smoke detection scenario where foreground and background classes are extremely imbalanced during training. $FL(p_t)$ as the focal loss function for classification score p_t, is expressed as follows:

$$\boldsymbol{FL(p_t) = -(1-p_t)^{\gamma}\log(p_t),} \tag{4.5}$$

where $-(1-p_t)^{\gamma}$is the modulating factor, with tunable focusing parameter γ = 2, and

$$\boldsymbol{p_t = \begin{cases} p & if\ y = 1 \\ 1-p & otherwise, \end{cases}} \tag{4.6}$$

where $y \in \{\pm 1\}$ specifies the ground-truth class and $p \in [0,1]$ is the model's estimated probability for the class with label $y = 1$. As suggested in the paper [20], we measure the difference between the offsets and the ground truth boxes using the bounding box regression loss function denoted by L_1.

Then, the total loss, L_{total}, is expressed as a linear combination of $FL(p_t)$ and L_1:

$$\boldsymbol{L_{total} = \alpha FL(p_t) + \beta L_1,} \tag{4.7}$$

where α and β are balancing terms. According to experiments [25] and [20], the optimal values of both α and β are given as 1.

Chapter 5. Experiments

5.1 Dataset

Large datasets are available for researchers in the field of object detection to perform benchmarking by training their models and comparing them to other methods. Unfortunately, the forest fire datasets are not available. In this study, a forest fire dataset was created by collecting data from several sources. The collected data set contains 4,350 forest fire images, of which 2,190 images are collected from the HPWREN Public Database [29] and the remaining images are manually collected from other sources on the Internet. These images are labeled and boxed using the tool in [30], and are then divided into a training set of 3,915 images and an evaluation set of 435 images. The dataset adequately accounted for a variety of forest fire scenarios by including forest fire images varying in fire intensity, time of day, smoke shape, etc.

5.2 Experimental setup

The model was implemented using the Python programming language and Pytorch framework. Then, it was trained and evaluated using a computer with a GeForce RTX 3060 GPU card. The training process took 60 epochs with a batch size of 6. The learning rate was initialized as 2.5×10^{-3} and then decreased by 10 times and 100 times after 40 epochs and 55 epochs, respectively.

Our model was compared with various existing models, including RetinaNet [25], YOLOv3 [18], Faster-RCNN [20], SSD [26] with the same implementation settings in number of epochs and learning rate for fair comparison. We also compared our backbone

with other backbones like VGG16 [33], Convnext [34], EfficientNet [35], InceptionV1 [36], and InceptionV4 [37], using the same Head and Neck.

5.3 Evaluation metrics

In this paper, we use the average precision (AP) metric of MS-COCO [36], which is one of the evaluation criteria widely used in target detection tasks, to evaluate the accuracy of the model. Precision and Recall are used to calculate AP and are expressed as:

$$\boldsymbol{Precision = \frac{TP}{TP + FP}} \tag{5.1}$$

$$\boldsymbol{Recall = \frac{TP}{TP + FN}} \tag{5.2}$$

$$\boldsymbol{AP = \int_0^1 P(r)dr\,,} \tag{5.3}$$

where True Positive (TP) indicates that the model predicted the presence of smoke (Positive) and the prediction was correct (True), False Positive (FP) indicates that the model predicted the presence of a smoke (Positive), but the prediction was incorrect (False), True Negative (TN) indicates that the model predicted the absence of smoke (Negative), but the prediction was correct (True), and False Negative (FN) indicates that the model predicted the absence of smoke (Negative), but the prediction was incorrect (False). Precision indicates the ratio of the correct predictions to all predictions, while Recall indicates the ratio of the correct predictions to all labeled smokes. In addition to AP, some other metrics are used to evaluate the performance of the model.

AP_{50} and AP_{75} indicate the AP values at 50% and 75% IoU (Intersection over Union) thresholds, respectively, and AP_S, AP_M, and AP_L are AP values for small, medium, and large objects, respectively. GFLOPs (giga floating-point operations) and #Params (the number of parameters) are used to evaluate the computational complexity of the model, and FPS (frames per second) is used to evaluate detection speed.

5.4 Experimental results

Table 5.1. Performance Comparison of our model and the other models

Model	**AP**	**AP_{50}**	**AP_{75}**	**AP_S**	**AP_M**	**AP_L**	**#Parameters(M)**	**GFLOPS**	**FPS**
Our model	***52.9***	***85.7***	***53.3***	***27.8***	***50.2***	***85.8***	***18.61***	***120.63***	***21.5***
RetinaNet [24]	50.8	82.1	49.3	24.3	46.6	85.6	36.10	127.82	20.4
Faster-RCNN [20]	48.3	79.5	46.7	27.5	45.3	78.3	41.12	134.38	17.7
YOLOv3 [18]	45.5	81.0	44.0	21.0	46.5	73.3	61.52	121.15	22.0
SSD [25]	43.0	77.8	42.4	21.6	47.1	70.8	24.39	214.18	17.4

The performance comparison is shown in Table 4.1. Overall, our model achieved the best values for AP and its sub-metrics while keeping the low number of parameters and GFLOPS. RetinaNet, which has neck and head similar to our model, achieved competitive accuracy, especially in terms of AP_L, with nearly identical values. However, RetinaNet required a higher computational cost because it uses Resnet as its backbone. In particular, it shows more than twice the number of parameters and 6% higher GFLOPS compared to our model. As a two-stage object detection model, Faster R-CNN typically achieves high

accuracy by using more stages in its architecture. However, while this model requires many parameters and GFLOPS, it achieved 8.7% lower AP compared to our model. Note that YOLOv3 showed slightly higher FPS than our model but achieved significantly lower accuracy. This demonstrates the importance of customizing and optimizing the model to suit detection of fire and smoke objects.

Table 5.2. Performance Comparison of proposed backbone and other backbones

Backbone	**AP**	**AP_{50}**	**AP_{75}**	**AP_S**	**AP_M**	**AP_L**	**#Parameters (M)**	**GFLOPS**	**FPS**
Proposed	***52.9***	***85.7***	***53.3***	***27.8***	***50.2***	***85.8***	***18.61***	***120.63***	***21.5***
VGG16 [33]	49.7	83.7	48.8	25.6	45.5	82.6	142.93	331.82	12.2
Convnext [34]	48.0	81.0	46.3	19.7	45.6	78.9	19.61	90.11	22.0
EfficientNet [35]	44.0	70.9	42.0	17.2	40.7	73.1	14.58	25.75	26.1
Inceptionv1 [36]	41.2	69.4	40.4	9.6	33.8	82.1	16.13	52.25	23.8
Inceptionv4 [37]	41.0	66.4	40.3	7.5	39.2	82.0	52.92	120.43	21.0

Table 4.2 compares the performance of the proposed backbone with other popular backbones using the same Neck, Head and other settings. Overall, the proposed backbone achieves the best AP values while keeping fairly favorable values in #Parameters and GFLOPS. VGG16 also achieved good AP values, but with significantly higher #Parameters and GFLOPS values. On the other hand, EfficientNet and InceptionV1 generated significantly fewer parameters but with significantly lower AP values of 44.0 and 41.2, respectively. It can be concluded that the proposed backbone achieves both efficiency and

effectiveness for forest fire detection surpassing recent methods by a clear gain in AP and latency criterion.

The qualitative test results for forest fires are shown in Figure 4.1 that includes 15 test images. The proposed model was able to detect various shapes of smokes and/or fires correctly regardless of daytime or nighttime. Moreover, the model could detect small smokes in images such as 11, 12, 13, and 14, blurred smokes such as 8, 9 and 12, and far-away smokes such as 11, 12, 13, and 14, that are difficult for humans to discern. The detection of smoke implies that the model can detect a forest fire at an early stage.

To show how well the attention mechanism works, heat maps of images using the Grad-CAM technique with different models applied are compared in Fig. 4.2. The first column of the table shows four different smoke images and each of the next five columns shows the heat map of the corresponding image when each model is applied. Looking at the heap map, hot colors such as red and yellow indicate high attention, while cool colors such as blue and green indicate low attention.

It is clearly seen that the attention area of the heat map generated by our proposed model depicts the shape of the original smoke image much better` compared to other heat maps. It is also seen that our model is better able to suppress the less relevant regions. For example, in our model's heat map to the last smoke image, the attention area has a shape very similar to that of smoke.

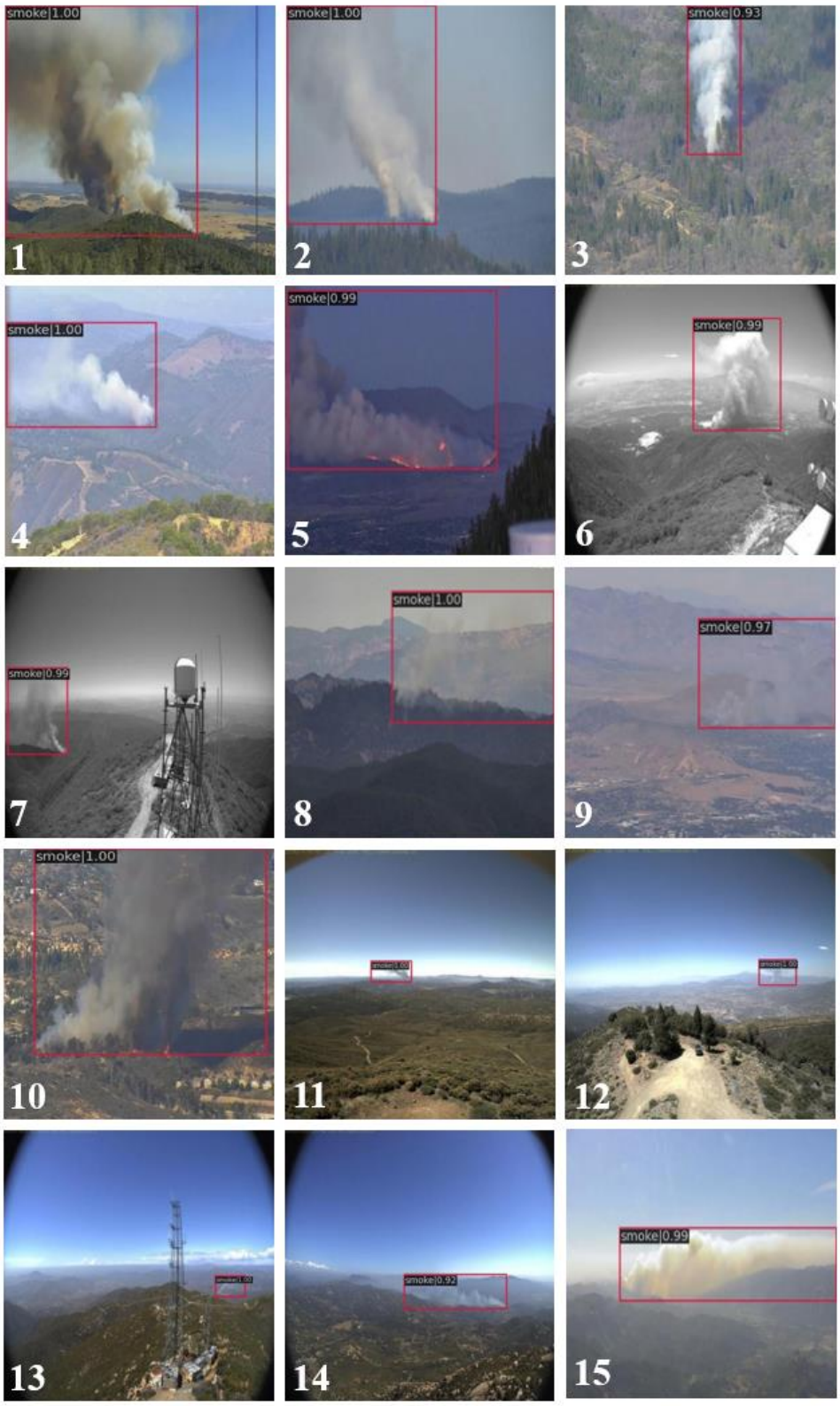


Figure 5.1. The qualitative results for forest fire detection on our dataset.

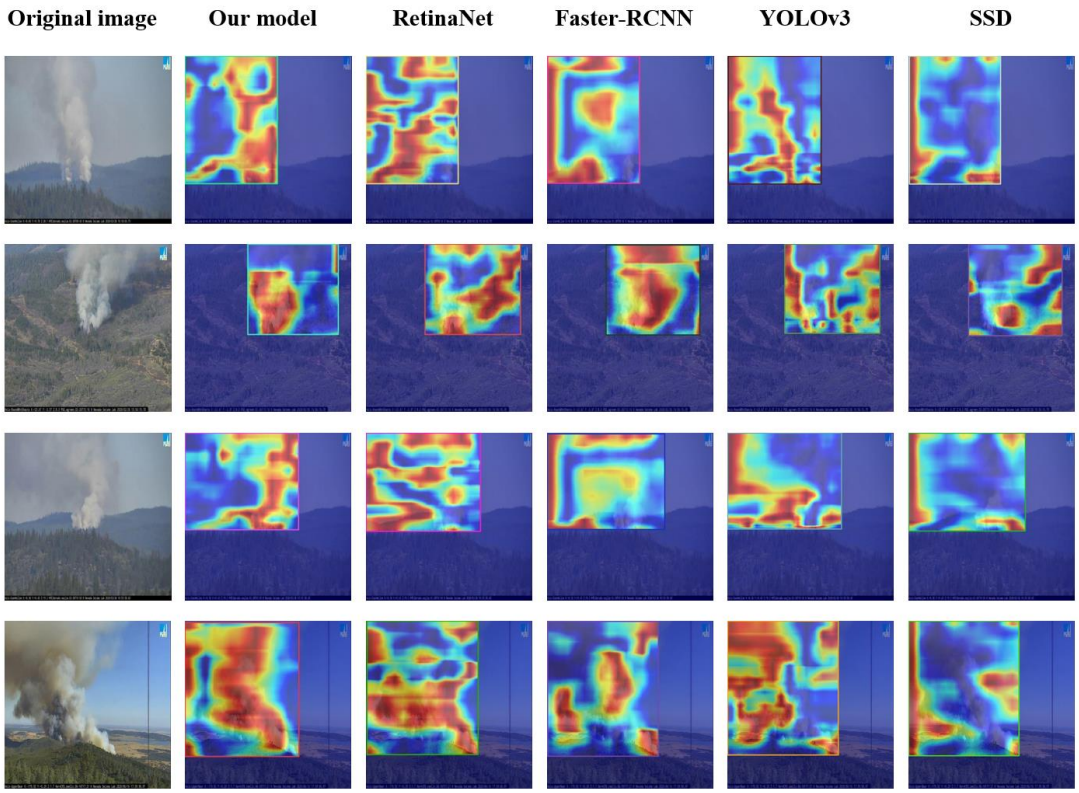


Figure 5.2. Grad-CAM visualization

5.5 Ablation study

Table 5.3. Ablation study on backbone modules with different techniques

Basic	Splitting	DW-coordinate	CBAM	AP	#Parameters(M)	GFLOPS
√				49.9	28.21	140.49
√	√			50.7	20.93	125.55
√	√	√		52.6	18.52	120.62
√	√	√	√	52.9	18.61	120.63

Finally, we conducted an ablation study by examining the effect of using techniques such as splitting (Splitting), depth-wise convolution of coordinate kernels (DW-coordinate), and attention mechanism (CBAM) on the basic backbone of our model (Basic). According to the results in Table 4.3 when each of those techniques is added, the proposed model can not only improve the accuracy, but also reduce the number of parameters. Especially, when our model employs all above techniques together, it could achieve 6% higher AP while reducing #Parameters and GFLOPS by 34% and 14%, respectively, compared to the basic model. It is also worth noting that the techniques, depth-wise convolution of coordinate kernels, contribute the most to the model's accuracy.

Chapter 6. Contribution Summary and Further Work

6.1 Contribution summary

This study introduced a variant of a vision-based fire detection model that relies on CNN for early and efficient detection of forest fires. The model focuses on developing a new Backbone which is suitable for extracting features from the images of smoke in the forest. Specifically, we applied a splitting technique as well as the use of depth-wise and coordinate convolutions to efficiently detect different types of smoke from forest fires. The attention mechanism is also integrated into the backbone architecture to improve detection accuracy. Our model was evaluated using a dataset that contains 4350 images of forest fires. According to the experiment results, our forest fire detection model performed better than the existing models in terms of accuracy and computational cost reduction.

6.2 Future work

Future work may consider a variety of mechanisms to further improve CNN's performance for forest fire detection, particularly in distinguishing smoke from other objects with similar characteristics such as clouds, fog, etc. Besides, we also consider the lightweight structures in our design to make the model easily employing on the low computational devices.

Bibliography

[1] *Facts + Statistics: Wildfires*, [cited 2023 Apr 1st], Available from: https://www.iii.org/fact-statistic/facts-statistics-wildfires.

[2] V. Chowdary, and M.K. Gupta, "Automatic forest fire detection and monitoring techniques: a survey," *Intelligent Communication, Control and Devices: Proceedings of ICICCD 2017*, pp. 1111-1117, 2018

[3] A. A. A. Alkhatib, "A review on forest fire detection techniques*," International Journal of Distributed Sensor Networks*, 10.3, p. 597368, 2014

[4] P. Barmpoutis, P. Papaioannou, K. Dimitropoulos and N.Grammalidis, "A review on early forest fire detection systems using optical remote sensing," *sensor*, p. 6442, 2022

[5] *History of the Osborne Firefinder*, [cited 2023 Apr 1st], Available from: http://www.nysforestrangers.com/archives/osborne%20firefinder%20by%20kresek.pdf.

[6] K. Bouabdellah, H. Noureddine, and S. Larbi, "Using wireless sensor networks for reliable forest fires detection," *Procedia Computer Science 19 (2013),* pp. 794-801, 2013.

[7] A. Gaur, A. Singh, A. Kumar, A. Kumar and K. Kapoor "Video flame and smoke based fire detection algorithms: A literature review," *Fire technology 56* (2020), pp: 1943-1980, 2020.

[8] T.H. Chen, P.H. Wu, and Y.C. Chiou, "An early fire-detection method based on image processing," *2004 International Conference on Image Processing*, ICIP'04, 2004, IEEE, Vol. 3, pp. 1707-1710, 2004.

[9] V. Vipin, "Image processing based forest fire detection," *International Journal of Emerging Technology and Advanced Engineering* 2.2, pp. 87-95, 2012.

[10] C. Yuan, Z. Liu, Y. Zhang, "UAV-based forest fire detection and tracking using image processing techniques," *2015 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE*, pp. 639-643, 2015.

[11] Z. Zhang, J. Zhao, D. Zhang, C. Qu, Y. Ke and B. Cai, "Contour based forest fire detection using FFT and wavelet," *2008 International conference on computer science and software engineering, IEEE*, Vol. 1, pp. 760-763, 2008.

[12] P. Foggia, A. Saggese, and M. Vento, “Real-time fire detection for video-surveillance applications using a combination of experts based on color, shape, and motion,” *IEEE transactions on circuits and systems for video technology 25(9)*, pp. 1545-1556, 2015.

[13] M.A. Mahmoud, and H. Ren, “Forest fire detection using a rule-based image processing algorithm and temporal variation,” *Mathematical Problems in Engineering 2018*, 2018.

[14] S. Wang, T. Chen, X. Lv, J. Zhao, X. Zou, X. Zhao, M. Xiao and H. Wei, “Forest fire detection based on lightweight Yolo,” *2021 33rd Chinese Control and Decision Conference (CCDC)*, IEEE, pp. 1560-1565, 2021.

[15] A. Bochkovskiy, C.Y Wang, and H.Y.M Liao, “Yolov4: Optimal speed and accuracy of object detection,” *arXiv preprint arXiv:2004.10934*, 2020.

[16] A. Howard, M. Sandler, G. Chu, L. C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q.V. Le, and H. Adam, “Searching for mobilenetv3,” *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1314-1324, 2019.

[17] Z. Jiao, Y. Zhang, J. Xin, L. Mu, Y. Yi, H. Liu and D. Liu, “A deep learning based forest fire detection approach using UAV and YOLOv3,” *2019 1st International conference on industrial artificial intelligence (IAI)*, IEEE, pp. 1-5, 2019.

[18] J. Redmon, and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018.

[19] Q. X. Zhang, G. H. Lin, Y.M Zhang, G. Xu, and J. T. Wang, “Wildland forest fire smoke detection based on faster R-CNN using synthetic smoke images,” *Procedia engineering*, pp. 441-446, 2018

[20] S. Ren, K. He, R. Girshick and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, 28, 2015.

[21] K. Vani, “Deep learning based forest fire classification and detection in satellite images,” *2019 11th International Conference on Advanced Computing (ICoAC)*, IEEE, pp. 61-65, 2019.

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

[23] U. Meena, G. Munjal, S. Sachdeva, P. Garg, D. Dagar, and A. Gangal, "RCNN Architecture for Forest Fire Detection," *2023 13th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, IEEE, pp. 699-704, 2023.

[24] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580-587, 2014.

[25] T. Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal loss for dense object detection," *Proceedings of the IEEE international conference on computer vision*, pp. 2980-2988, 2017.

[26] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu and A. C. Berg, "Ssd: Single shot multibox detector," *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands*, October 11–14, 2016, Proceedings, Part I 14, Springer International Publishing, pp. 21-37, 2016.

[27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S.Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, pp. 211-252, 2015.

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

[29] S. Woo, J. Park, J. Y. Lee and I. S. Kweon, "Cbam: Convolutional block attention module," *Proceedings of the European conference on computer vision (ECCV)*, pp. 3-19, 2018.

[30] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature pyramid networks for object detection*," Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117-2125, 2017.

[31] *High Performance Wireless Research and Education Network*, [cited 2023 Apr 1st], Available from: http://hpwren.ucsd.edu/index.html.

[32] *Roboflow*, [cited 2023 Mar 1st], Available from: https://roboflow.com/.

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

[34] Z. Liu, H. Mao, C. Y. Wu, C. Feichtenhofer, T. Darrell and S. Xie, "A convnet for the 2020s," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976-11986, 2022.

[35] M. Tan, and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *International conference on machine learning*, PMLR, pp. 6105-6114, 2019.

[36] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D, Erhan, V. Vanhoucke and A. Rabinovich, "Going deeper with convolutions," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1-9, 2015.

[37] C. Szegedy, S. Ioffe, V. Vanhoucke and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," *Proceedings of the AAAI conference on artificial intelligence*, Vol. 31, No. 1, 2017.

[38] T.Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár and C. L. Zitnick, "Microsoft coco: Common objects in context," *Computer Vision–ECCV 2014: 13th European Conference*, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, Springer International Publishing, pp. 740-755, 2014.