iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** XII **Month of publication:** December 2023

DOI: https://doi.org/10.22214/ijraset.2023.57847

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue XII Dec 2023- Available at www.ijraset.com

Advances in Ecological Surveillance: Real-Time Wildlife Detection using MobileNet-SSD V2 Convolutional Neural Network

Geethanjali P[1], Metun[2], Dr. M Rajeshwari[3]
[1]*Research Scholar, Department of Electronics and Telecommunication, Bangalore Institute of Technology, Bangalore, India*
[2] *Research Scholar, Department of Computer Science, New York University, Brooklyn, New York – 11201*
[3] *Head of Department, Department of Electronics and Telecommunication, Bangalore Institute of Technology, Bangalore, India*

Abstract: ***Human-wildlife conflicts have escalated due to increased encroachment on natural habitats, necessitating advanced surveillance to mitigate potential threats and preserve biodiversity. Traditional monitoring methods are labor-intensive and inefficient when dealing with the volume and velocity of data generated by camera traps. This research introduces an innovative, automated wildlife detection system utilizing a Convolutional Neural Network (CNN) - the MobileNet-SSD V2 - to process images for real-time animal detection. The paper elaborates on a comprehensive methodology, from dataset curation to model training and deployment, leveraging TensorFlow Lite for on-device inference. The approach includes building a versatile CNN framework with labeled images from benchmark datasets, annotated using LabelImg and deployed it in Raspberry Pi Model B using TensorFlow Lite. The proposed system is not only accurate but also conserves memory resources by employing an optimized CNN architecture that requires less computational storage. With regular updates using fresh camera-trap images, the system ensures continual improvement in species recognition precision. The system exhibits exceptional accuracy, robustness in diverse environmental settings, and real-time alert mechanisms for rapid response. Through extensive experiments, this automated detection model achieves a higher accuracy rate in clear conditions and maintains precision of with considerable resilience in lower-quality image scenarios. The findings in this research demonstrate the profound capabilities of CNNs in ecological surveillance and their potential to assist conservation efforts.***
Keywords: Convolutional Neural Network, MobileNet-SSD V2, Raspberry Pi, Computer vision, TensorFlow.

I. INTRODUCTION

Human-animal conflicts pose a significant problem, resulting in the loss of vast resources and endangering human lives. In recent times, the frequency of these conflicts has been on the rise. As humans encroach into forests to secure their livelihoods or claim land for agricultural practices, rapid industrialization leads to the expansion of urban areas. This encroachment often results in animals entering nearby villages and damaging vegetation in farmlands. Therefore, constant monitoring of these intrusion prone areas is essential to prevent the entry of such animals or any other unwanted intrusions.

This paper proposes a novel solution to address this issue by a method for wildlife animal detection using image processing and Convolutional Neural Networks (CNNs) based on camera-trap images by integration with Raspberry Pi microcontroller and utilizing Solar panels for charging the Battery continuously to cut down the overall cost of the detection model. Sensor cameras are strategically installed on a pole or trees within a specific area to form a fixed camera-trap network for wildlife monitoring. These camera traps are triggered to capture images whenever movement is detected.

This study is centered on the enhancement of wildlife monitoring through the deployment of camera traps that yield motion-triggered video sequences with GSM location tracking. Traditional surveillance techniques often suffer from low detection rates, particularly in videos where the distinction between the animal and its environment is minimal. Such methods also struggle with high rates of false positives, largely due to the dynamic nature of background elements. The proposed hardware and software architecture in this paper will provide an effective monitoring system, expediting the analysis of wildlife research data and facilitating resource management decisions. The creation of diverse automated tools to process the vast amount of images from camera traps is urgently needed. This includes tools for identifying animals, segmenting images, extracting relevant data, and tracking animal movements. This innovative approach aims to provide more effective means of mitigating conflicts between humans and animals, offering a promising avenue to address this global challenge.

II. LITERATURE REVIEW

The swift pace of industrial development often drives wildlife into adjacent rural areas, where they encroach on agricultural lands for sustenance. This interaction poses dangers to both wildlife and human populations, frequently resulting in the depletion of resources and occasionally in the loss of life. Additionally, these animals frequently wander onto roads, leading to vehicular collisions and property damage. The model's real-time effectiveness should be consistent with its theoretical capabilities. The core hypothesis of this research is the inadequacy of images showing animals behind bars in the training sets of current models, with specific emphasis on pandas and deer. This study involves collecting targeted training data and creating a unique dataset to explore methods for enhancing the CNN's performance. The model's performance was improved by integrating images from both caged and non-caged animal datasets. However, the outcomes in real-time applications fell short compared to theoretical performance. Although there have been many attempts to utilize CNNs for animal detection, the efficiency of these models is often limited by factors like the volume of training data and the duration of model training.[1-4]

The research proposes a new algorithm using Deep CNN where the method focuses on improving the accuracy of object classification within the framework. Optimal accuracy was noted with image sizes of 50x50 pixels and training over 100 epochs.[5]

The study details the creation of a specific dataset using Google Open Images and COCO datasets by evaluating several advanced neural network architectures including YOLOv3, RetinaNet, R-50-FPN, Faster RCNN R-50-FPN, and Cascade RCNN R-50-FPN.[6] The YOLOv3 architecture showcased the best performance metrics, achieving over 35 fps and an mAP of 0.78 across ten classes, and 0.92 for a combined class. Meanwhile, the RetinaNet R-50-FPN architecture achieved a recognition speed of over 44 fps but had a 13 percent lower mAP compared to YOLOv3.[7]

Traditionally, farms have relied on electric fences to deter animals from entering, which may cause animals to behave abnormally. To address these challenges, the proposed system can be deployed in areas where it is needed. There is a need to enhance the accuracy of the Convolutional Neural Network (CNN) model while reducing computational complexity. This problem is addressed by a novel approach for identifying rare animals in images using Convolutional Neural Networks (CNNs) is presented.[8-9] This method focuses on automatically extracting image features from the training set to develop a system capable of recognizing rare animals. Key components of this system include an image acquisition and preprocessing module, which processes images in real-time to reduce noise and enhance recognition accuracy. Additionally, a module for locating target rare animals within images is incorporated.[10] This localization module efficiently identifies the positions of rare animals in the images. The recognition algorithm further refines the network by adjusting the weight parameters in each layer based on the unique characteristics of rare animal images.[11]

This paper proposes a system where the main focus is on developing a wild animal detection and real-time alert system to prevent wild animal intrusions and mitigate potential harm with a higher level of accuracy for optimal speed. The system identifies animal intrusions and promptly sends alert notifications to address the situation. This research provides solutions to the problems identified in the existing system such as developing an animal detection and real-time alert system that can be used to prevent wild animal assaults, capture the image of animals and categorize it using image processing, and to alerting farm owners and forest officials about animal intrusion. The proposed system introduced in this study capitalizes on solar power for sustainability, driving a Raspberry Pi via a battery module. Incorporating GSM for live animal tracking achieves a synergy of eco-friendliness and technological advancement. This cost-effective solution stands out for its autonomy and low environmental impact, marking a significant step forward in remote ecological monitoring.

III.METHODOLOGY

The following is a detailed methodology for an autonomous wildlife detection system designed to identify lions, tigers, and elephants using advanced machine learning algorithms. The system is built around the MobileNet SSD V2 architecture, utilizing its two variants: MobileNet-SSD V2 coco and MobileNet-SSD V2 320x320, for image processing and object detection.

A. Hardware Setup

The system is mounted on a pole to ensure an expansive field of view as shown in Figure 1. This setup incorporates a photovoltaic module that harnesses solar energy, effectively channeling it to a storage unit that supplies power to the central processing unit, a Raspberry Pi. This self-reliant power system underlines the autonomous nature of the deployment. The visual representation illustrates the operational setup situated within the target observation zone. Affixed to a robust and immobile post, the surveillance camera, along with the motion detection sensor, communication module, alert system, and illumination component, operates cohesively.

The utilization of solar power is a deliberate choice, emphasizing sustainable energy use, which in turn energizes the Raspberry Pi through an intermediary battery module. An additional protective layer encases the Raspberry Pi circuitry and other integral elements, shielding them from environmental elements and potential physical disturbances. This protective measure ensures the continued integrity and functionality of the critical technological components within.

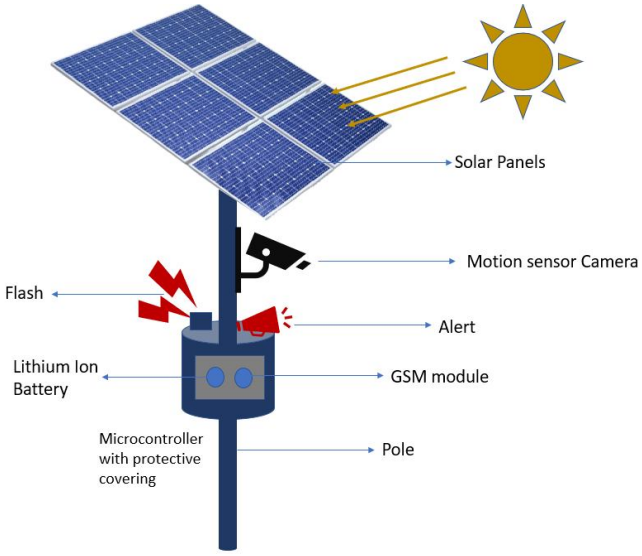


Figure 1 Working model setup

1) Block Diagram Representation:

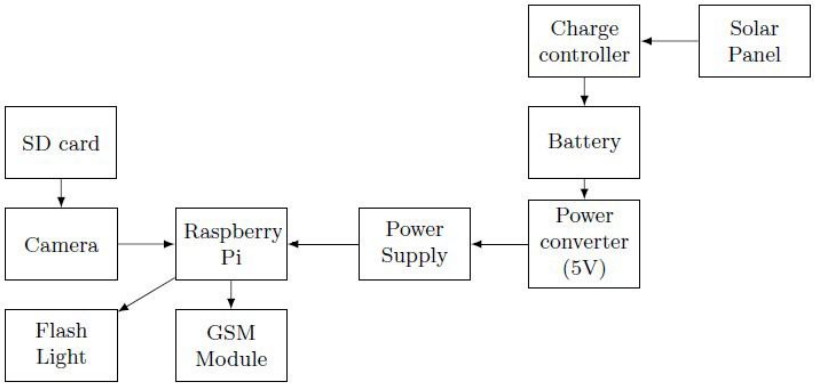


Figure 2 Block Diagram of the standalone model

The device is attached to a pole. A solar panel is used here in an attempt to use solar energy to power the Raspberry Pi via a battery and create a selfcontained embedded solution for this project. The camera module and SD card are attached to Raspberry Pi as shown in Figure 2. The camera sends several images to Raspberry Pi. The images are stored in an SD card. Raspberry Pi uses a trained neural network model to detect if the specific animal is present in the images. Once the animal is detected, Raspberry Pi sends an alert to the forest officials using a transmission device such as GSM. Additionally, a flashlight is used as an indicator of the presence of the animal in the captured image. Data of animals can be stored in the SD Card and deleted after it is sent to the computer, for efficient memory management.

2) Flowchart

The Figure 3 presents a flowchart detailing the following process: Initially, the system activates upon powering up the Raspberry Pi microcontroller, which in turn powers the camera and GSM modules. The camera captures live video, feeding the frames to the Raspberry Pi. Each frame's output is determined by a threshold set at 0.5. Animals detected with a confidence level above this threshold are included in the output; those detected with less than 0.6 confidence are disregarded. Detected animals are highlighted with bounding boxes, and these results are displayed on the Raspberry Pi's camera display on the desktop.

When an animal is detected, an SMS alert is sent to forest officers' phone numbers, aiding them in safely intercepting or capturing the wild animals. Additionally, a flashlight and an alarm are activated, serving as a warning to villagers or farmers nearby, helping prevent potentially dangerous encounters with the animals. As long as the system receives sufficient power, steps 2 through 6 are continuously repeated.

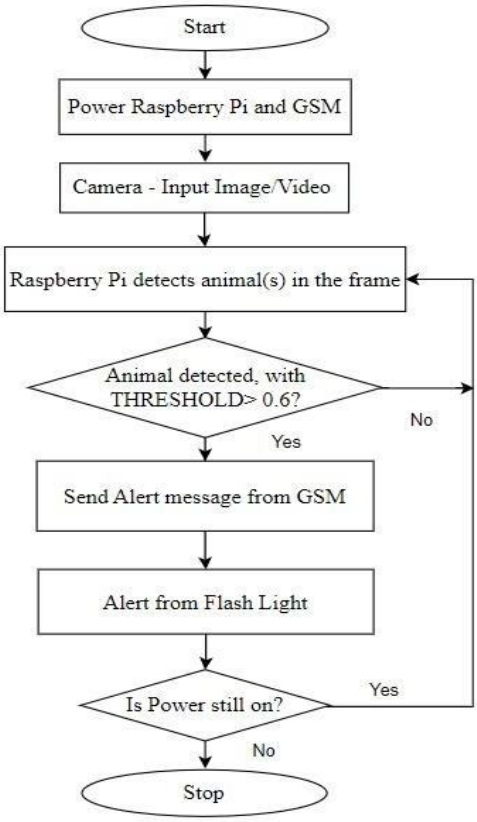


Figure 3 Flowchart of the working system

B. Image Processing

1) Collection of Dataset: A comprehensive dataset comprising images of lions, tigers, and elephants has been meticulously curated for this study. These images as shown in Figure 4 encompass diverse backgrounds and orientations, ensuring the model's robust detection capabilities. To guarantee the dataset's authenticity and reliability, the images are sourced from reputable and validated platforms known for their extensive collection of wildlife imagery.

Figure 4 Collection of Datasets for Elephants

Specifically, a substantial dataset is compiled for each animal category, covering various backgrounds and viewing angles essential for training the model effectively. The dataset is meticulously obtained from the Kaggle.com website and live footage from Ramnagar forest, which is a renowned source for diverse datasets. The purpose of this dataset is to facilitate the practice of various image processing techniques, enhancing the overall robustness of the model. The dataset used in this research consists of the following particulars:

- Number of classes: 3 [Elephants(Asian elephant and African elephant); Tiger; Lion]
- Number of images: 5000
- Image shape range: (100, 100) to (4992, 3328)

By incorporating this dataset into the research, the study benefits from a rich collection of images encompassing diverse animal species, backgrounds, and orientations, essential for training and evaluating the model's detection capabilities.

3) Labelling the Dataset: In this study, the dataset is annotated using the LabelImg tool, as illustrated in Figure 5. LabelImg is an intuitive and basic tool for labeling a few hundred images, which is vital for creating a dataset to train computer vision models. For an initial proof-of-concept model, starting with a dataset of 200 images is recommended. The selected training images should include a mix of random and target objects, encompassing a range of backgrounds and varying lighting conditions.

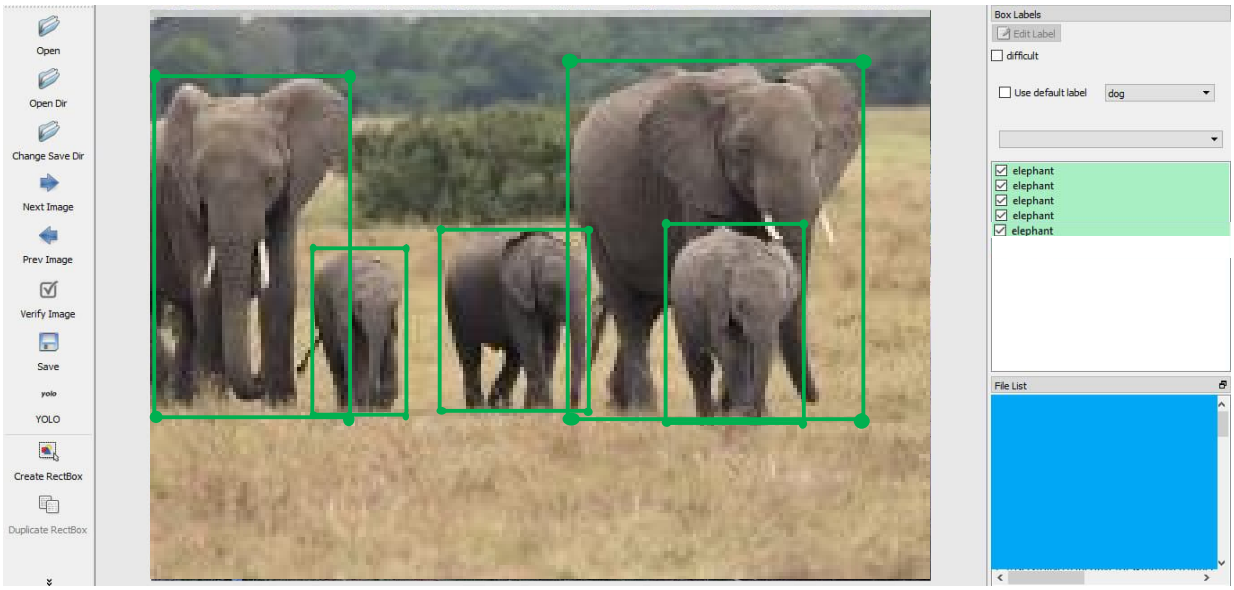


Figure 5 Labelling Images using LabelImg

In this study, the dataset is annotated using the LabelImg tool, as illustrated in Figure 5. LabelImg is an intuitive and basic tool for labeling a few hundred images, which is vital for creating a dataset to train computer vision models. For an initial proof-of-concept model, starting with a dataset of 200 images is recommended. The selected training images should include a mix of random and target objects, encompassing a range of backgrounds and varying lighting conditions.

The tool uses the following approach:

a) Bounding Box Coordinates: A bounding box in an image is typically defined by two sets of coordinates as follows: (x_{min}, y_{min}) for the top-left corner and (x_{max}, y_{max}) for the bottom-right corner.

b) Intersection Over Union (IoU): IoU is a metric to evaluate the overlap between two bounding boxes. It is calculated as the ratio of the area of overlap to the area of union:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \quad (1)$$

Where IoU values range from 0 (no overlap) to 1 (perfect overlap).

c) *Non-Maximum Suppression (NMS):* NMS is a technique that is used extensively in object detection to eliminate redundant bounding boxes for the same object. It involves sorting bounding boxes by their confidence scores and removing boxes with high overlap (IoU) to keep only the most confident ones.

In this paper, the XML files are saved in the PASCAL VOC format to store annotations. This format aligns with the standards commonly used in datasets like ImageNet. The process of generating bounding boxes and labeling for the images of elephants, lions, and tigers in the dataset is illustrated in Figure 5 using the LabelImg tool.

176.xml

```
<annotation>
    <folder>elephant</folder>
    <filename>176.jpg</filename>
    <path>E:\elephant\176.jpg</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>259</width>
        <height>194</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>elephant</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>13</xmin>
            <ymin>37</ymin>
            <xmax>212</xmax>
            <ymax>174</ymax>
        </bndbox>
    </object>
</annotation>
```

Figure 6 PASCAL VOC format (.xml file)

The content of the XML files in PASCAL VOC format is detailed in Figure 6. These XML files serve as a structured means of storing annotation information.

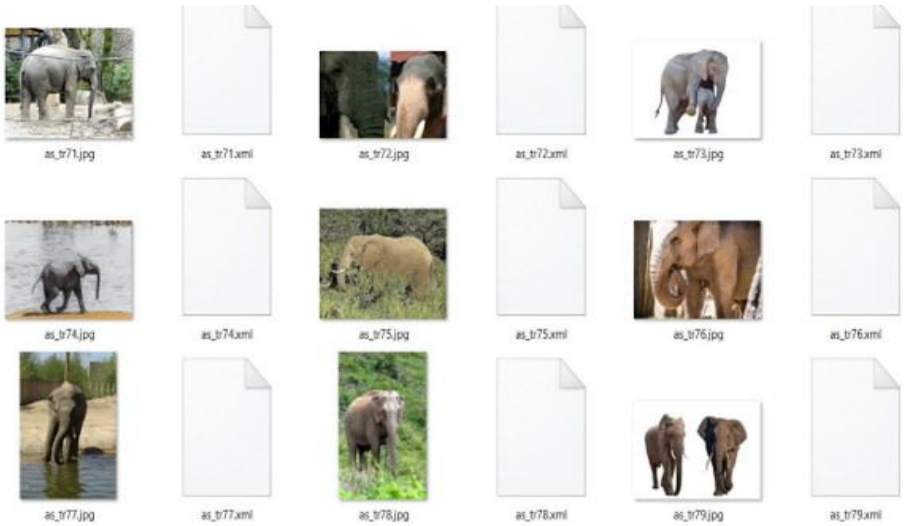


Figure 7 Dataset images and their respective .xml file

Furthermore, the annotations are consistently saved as XML files in the PASCAL VOC format, closely associated with the corresponding animal images in the dataset. This is exemplified in Figure 7, which presents a selection of images that have been appropriately labelled and organized within the file directory.

C. MobileNet-SSD V2 Description

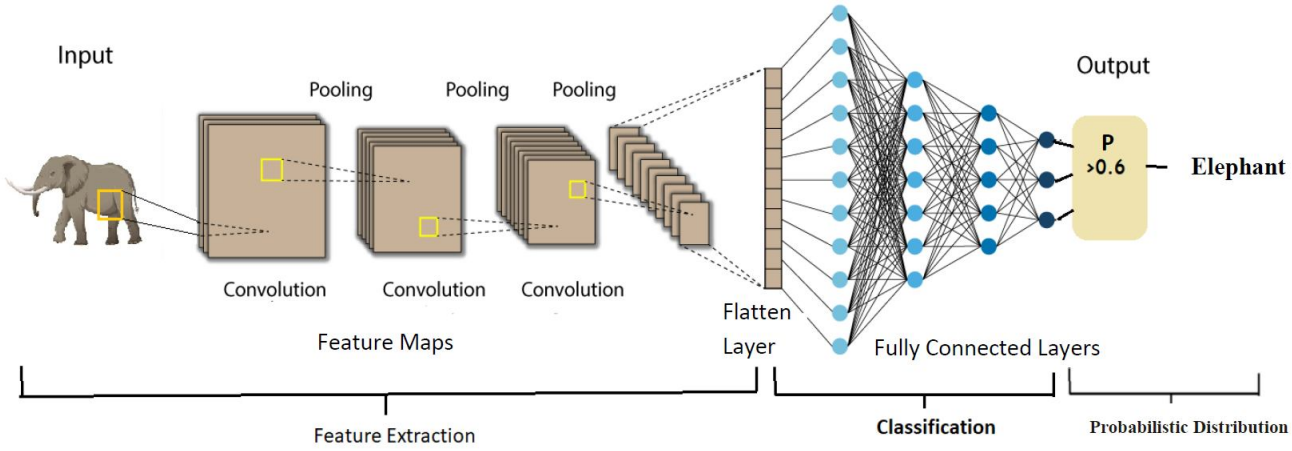


Figure 8 MobileNet-SSD V2 Architecture

In this study, each input to a convolutional layer constitutes a p × p × d data matrix, where p represents the height and width of the matrix, and d signifies the depth or the number of channels. Each convolutional layer employs l filters, each with dimensions s×s×t, where s is less than the dimension of the data matrix, and t aligns with or is less than the channel number d, varying with each filter. The datasets in this study are bifurcated into training and evaluation sets. The classifications for the null, human, and animal instances are encoded as (1, 0, 0), (0, 1, 0), and (0, 0, 1), respectively. The phase space transformation on the data before feeding it into the CNN is utilized.[11] The resultant output is derived through the alternate application of convolutional layers and pooling operations. Initially, features from the initial layers are fed into the network, where upon the model discerns the probabilistic feature distribution from these primary features during the training of the initial convolutional layer, designated as C2. The output from this convolutional layer is mathematically expressed by the following equation (2):

$$y_j^l = \phi\left(\sum_i \left(V_{ij}^l \odot y_i^{l-1} + c_j\right)\right) \quad (2)$$

Here, V_{ij}^l represents the set of convolutional filters, c_j is the bias term, and ϕ denotes the nonlinear activation function. Notably, $\odot$ signifies the convolution operation. The activation function, is commonly employed to introduce non-linearity into the learning process which is given as $\phi(z) = \frac{1}{1+exp(-z)}$. It is important to highlight that the filter dimensions directly influence the number of features extracted from input data. The parameters within the convolutional filters critically determine the feature abstraction capability, thereby enriching the resultant feature maps.[12] In instances where convolutional operations replace full connectivity, each neuron in layer l processes only a local patch of neurons from the preceding layer $l-1$, resulting in parameter efficiency and a reduction in computational load. By incorporating local connectivity and shared weights, CNNs can effectively capture spatial hierarchies in data, which is pivotal for tasks such as image and speech recognition.

The MobileNet-SSD V2 architecture is tailored for efficiency in embedded systems, utilizing the MobileNet-V2 backbone and FPN technology for an enhanced SSD framework as shown in the Figure 8. The Single-Shot Multibox Detector (SSD) models, specifically the MobileNet-SSD V2, are recognized for their speed, surpassing R-CNN models by forgoing the need for bounding box proposals. The choice of MobileNet- SSD V2 is attributed to its swift detection capabilities and compact model size. This model processes 320x320 pixel images in just 19 milliseconds to identify objects and their positions, outperforming other models in terms of speed.[13].

In comparison, the MobileNet-SSD V1 coco model requires 0.3 milliseconds more to categorize image objects, and the MobileNet-SSD V2 coco model takes 31 milliseconds, among others. The MobileNet-SSD V2 320x320 represents the most advanced iteration in the MobileNet series, tailored for Single-Shot Multibox detection. It achieves a balance between speed and accuracy, with a minimal decrease in mean average precision (mAP) of just 0.8 compared to its predecessor, the MobileNet-SSD V2 coco.[14] Additionally, OpenCV, a widely used open-source computer vision library, aids in the visualization of detection results by drawing and displaying bounding boxes and labels on video frames.

D. Training Model on Dataset using TensorFlow

The flowchart for the Training Model on Dataset using TensorFlow consists of Normalizing the data and Splitting the Dataset which is represented in Figure 9.

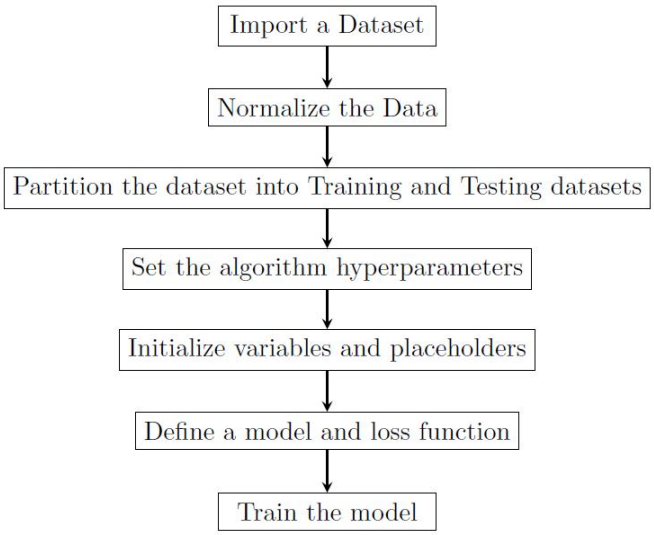


Figure 9 Flowchart for Training Model on Dataset using TensorFlow

1) *Normalizing of Data*: Normalizing is the process of modifying the data of each tensor so that the mean is zero and the standard deviation is one. This can be represented mathematically as:

$$x' = \frac{x-\mu}{\sigma} \quad (3)$$

Where x represent the original data point, x′ denote the normalized data point, μ symbolize the mean of the data, and σ indicate the standard deviation of the data in Equation (3).

2) *Splitting Dataset:* The dataset images are split into train, validation, and test sets.

a) *Training Images:* Training images are the actual images used to train the model. In each step of training, a batch of images from the "train" set is passed into the neural network. The network predicts the classes and locations of objects in the images. The training algorithm calculates the loss and adjusts the network weights through backpropagation. The training loss can be represented as:

$$L_{(y,\hat{y})} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad (4)$$

Where L denotes the loss function, y represents the actual(true) label, ˆy signifies the predicted label, and N stands for the number of data points in Equation (4).

b) *Validation Images:* Validation images from the "validation" set can be used by the training algorithm to check the progress of training and adjust hyperparameters (like learning rate). Unlike "train" images, these images are only used periodically during training (i.e. once every certain number of training steps).

c) *Testing Images:* Testing images are never seen by the neural network during training. They are intended to be used by humans to perform a final test of the model to check how accurate the model is.

d) *Setting Algorithm Hyperparameters:* The selection of model hyperparameters, such as the count and size of hidden layers, affects model choice. Conversely, algorithm hyperparameters, like the learning rate in Stochastic Gradient Descent (SGD) or the quantity of nearest neighbors in a K Nearest Neighbors (KNN) classifier, impact the efficiency and effectiveness of the learning process. The formula for updating parameters in gradient descent through backpropagation is given by:

$$\theta = \theta - \alpha\nabla J(\theta) \quad (5)$$

Where θ denotes the model's parameters, including weights and biases. The symbol α represents the learning rate.[15] The term J(θ) refers to the cost or loss function, and ∇J(θ) indicates the gradient of the loss function relative to the parameters, as seen in Equation (5). Adopting the principles of transfer learning, a pre-trained MobileNet SSD V2 model is fine-tuned using a labeled dataset. The model's performance is enhanced using the following optimization function:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum(L_{\text{cls}}(c_i, c^*) + \lambda L_{\text{reg}}(t_i, t^*)) \tag{6}$$

Here, L(θ) denotes the loss function, L_{cls} the classification loss, L_{reg} the regression loss, ci the predicted class, t_i the predicted bounding box, c* the true class, and t* the true bounding box coordinates.[16] N represents the number of images, and λ is a parameter balancing classification and localization losses in Equation(6).

The model's accuracy is evaluated using the mean Average Precision (mAP) metric, defined as:

$$\text{mAP} = \frac{1}{Q}\sum\frac{1}{N_q}\sum(P(k) \times \text{rel}(k)) \tag{7}$$

Where Q is the number of query images, N_q the number of retrieved instances per query, P(k) the precision at cutoff k, and rel(k) the relevance of the prediction at rank k in Equation(7).

E. Model Deployment and System Operation

Within the operational framework, the Raspberry Pi is tasked with running the sophisticated TensorFlow model which is adept at processing the continuous feed of images. To maintain precision, a predefined confidence level is established that the model must meet or exceed to validate the detection of wildlife. When the model's confidence in recognizing an animal, such as a lion, tiger, or elephant, exceeds the set threshold, it confirms the animal's presence and initiates an alert protocol. Once a positive identification is made, the system promptly dispatches an SMS to the forest rangers and other relevant authorities through the GSM module. Concurrently, a visual signal is deployed to inform and prepare the local population or any nearby forest personnel of the presence of the animal. This two-pronged alert strategy ensures a rapid and effective response to potential wildlife encounters.

F. Memory Management and Power Sustainability

The framework of the system is intricately crafted to maintain vigilant oversight of its power reserves, implementing a dynamic assessment protocol that ensures the energy supply is judiciously allocated to sustain its functions. Simultaneously, it employs an intelligent storage management strategy for the SD card, which optimizes data allocation and retrieval processes. This dual approach is pivotal in guaranteeing that the wildlife detection workflows are carried out with minimal interruptions, thereby enhancing the system's overall reliability and longevity in field applications.

IV. RESULTS

A. Experimental Results

In the expansion of the wildlife detection system, this study successfully integrated the capability to accurately identify multiple species, including lions, tigers, and elephants. The robustness of the Convolutional Neural Network (CNN) model has been rigorously tested across a variety of natural environments with dynamic and challenging conditions. It was observed that even if the image is slightly blurred, the system sustains an accuracy rate of over 85% in identifying these animals. The overall performance shows an impressive accuracy rate of 95% and above for these species' detection.

A series of trials were conducted to validate the model's performance, during which the system consistently recognized and classified the species with high precision. For instance, in a test image, the model confidently identified a tiger with an accuracy of 97%, as indicated by the bounding box and confidence score overlay. Similar tests on lions, tigers, and elephants yielded comparable results, corroborating the model's adeptness at handling the intricate patterns and varied landscapes associated with these species.

The proposed CNN model has been trained on 5000 images. The MobileNet-SSD V2 model takes input images of sizes 320 x 320. The pixel values of the range 0-255 are converted into -1 to 1 range during preprocessing. The Visual results of the detected image are shown in Figure 10.

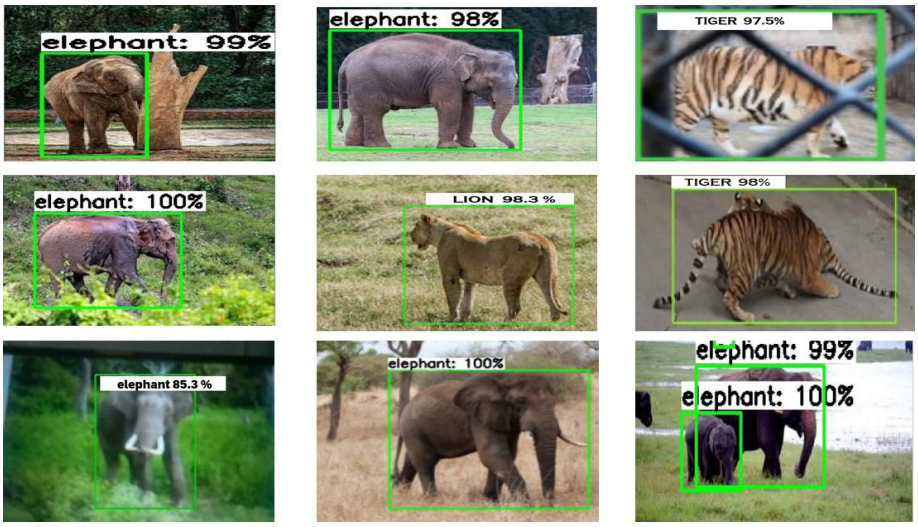


Figure 10 Visual results of detected images

B. Quantitative Analysis of Model Convergence

This research quantitatively evaluates the convergence behavior of the proposed wildlife detection model through a series of loss metrics over the training epochs. These metrics facilitate an understanding of the model's learning efficacy and generalization capabilities. The losses that are considered are as follows:

1) Classification Loss Convergence: Figure 11 depicts the classification loss, which quantifies the model's error in predicting the correct animal categories. A notable decline from an initial loss of approximately 0.18 to below 0.05 is observed over 1,800 training iterations, demonstrating significant learning progress.

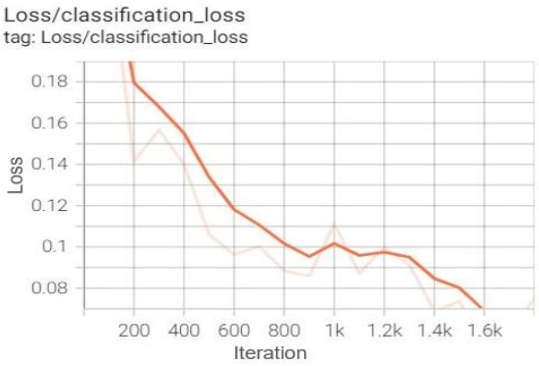


Figure 11 Classification Loss

2) Localization Loss: Figure 12 depicts the classification loss, which quantifies the model's error in predicting the correct animal categories. A notable decline from an initial loss of approximately 0.18 to below 0.05 is observed over 1,800 training iterations, demonstrating significant learning progress.

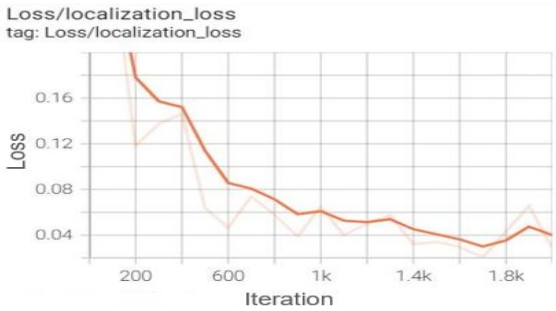


Figure 12 Localization Loss

3) *Regularization Loss*: Regularization is pivotal in averting overfitting, ensuring the model's ability to generalize. The regularization loss, as plotted in Figure 13, exhibits a steady decrease, stabilizing at approximately 0.149. This gradual diminution suggests the model's increasing robustness against overfitting to the training data.

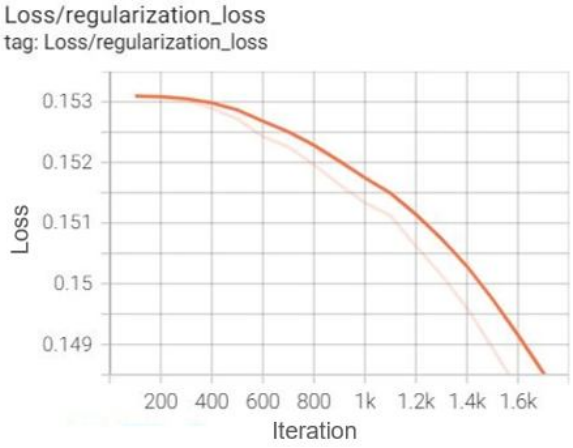


Figure 13 Regularization Loss

4) *Overall Loss Reduction*: The total loss, as demonstrated in Figure 14, amalgamates the aforementioned components of loss, providing an aggregate measure of model performance. Commencing at approximately 0.26, the total loss exhibits a pronounced downtrend with minor fluctuations, culminating in a loss below 0.1, indicative of the model's overall training stability and convergence.

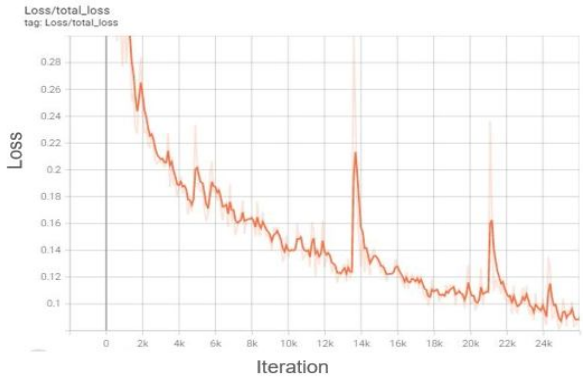


Figure 14 Regularization Loss

C. Model Comparison

In evaluating the effectiveness of object detection models, this study employs Average Precision (AP) as a principal metric, which calculates the mean precision across all recall values, ranging from 0 to 1. The mean Average Precision (mAP) represents the AP averaged over all classes, with TensorFlow Lite object detectors targeting an mAP between 20 and 50 for optimal performance on embedded devices. For this study, the MobileNet-SSD V2 320x320 is chosen as the preferred detection model due to its balance of detection speed and precision. This iteration of MobileNet is fine-tuned to enhance processing velocity while incurring a minimal reduction in mAP, decreasing by only 0.8 from its predecessor, MobileNet-SSD V2 COCO.

Average Precision is a widely used metric for evaluating the precision of object detection models. It calculates the mean precision value across all recall values from 0 to 1. The mean Average Precision (mAP) represents the average of these precision values for different classes. In the context of TensorFlow Lite object detectors, which are designed for embedded devices, an ideal mAP score ranges from 20 to 50. Among the various Single-Shot Multibox (SSD) models, the MobileNet-SSD V2 has been chosen for this project. This selection is attributed to its fast detection speed and compact model size. With an input image size of 320x320, the MobileNet-SSD V2 model achieves detection in just 19 milliseconds, showcasing its efficiency in quickly identifying objects and their locations in images.

Table 1. Comparative Study of Object Detection Models

Model Version	Speed (ms)	mAP Score
MobileNet- SSD V2 320x320	19 ms	43.5
SSD MobileNet V2 Coco	31 ms	44.3

The latest iteration of the MobileNet model, MobileNet-SSD V2 320x320, is tailored for Single-Shot Multibox detection. This model is fine-tuned for enhanced speed with a minimal decrease in mean average precision (mAP), showing only a 0.8 difference when compared to its predecessor, the MobileNet-SSD V1 coco, as indicated in Table(1).

V. CONCLUSIONS

This research successfully demonstrates the deployment of a Convolutional Neural Network (CNN) model for the accurate detection and classification of wildlife species, specifically lions, tigers, and elephants. This standalone model is placed in Ramanagar forest, Karnataka, India, and tested for intrusion of Elephants. Through rigorous training and validation processes, the model has exhibited remarkable proficiency, with accuracy rates consistently exceeding 95 percent for clear background and lighting and above 85 percent for blur images. The descending trends in classification, localization, and regularization losses not only indicate the model's capability to learn from the data but also its ability to generalize beyond the training set without overfitting, as reflected by the stabilization of the regularization loss. The incorporation of a GPS module would facilitate the precise localization of elephant, tiger, and lion herds, further contributing to timely preventive measures.

Future work will focus on enhancing the model's capabilities by improving the resolution and quality of the input data, which is anticipated to further refine the detection accuracy and expand the model's operational range. Additionally, the scalability of the approach suggests that with additional training, the model could be adapted for the detection of other species, thereby broadening its utility in ecological studies and conservation strategies. To mitigate human-animal conflicts, the installation of a sound-emitting system at the entry point can be implemented to deter wild animals before they encroach on agricultural lands.

Additionally, incorporating temperature and humidity sensors would enhance the durability and operational reliability of the device under various environmental conditions. Lastly, the integration of night vision alongside high-resolution cameras would significantly improve nighttime detection capabilities, ensuring the round-the-clock effectiveness of the system.

In conclusion, this study represents a significant step forward in the field of wildlife monitoring and conservation technology. The results underscore the viability of CNNs in addressing ecological challenges and highlight the transformative potential of artificial intelligence in fostering harmonious coexistence between human development and wildlife preservation.

VI. ACKNOWLEDGMENT

We express our heartfelt gratitude to the Head of Department of Electronics and Telecommunication, BIT, Dr. M. Rajeshwari for her support and guidance during the course of this project. The authors would like to thank Shri. Devaraju V, DFO of Ramanagar Forest, Karnataka for the extended support for this research and the anonymous reviewers for their valuable comments that improved the manuscript significantly.

REFERENCES

[1] W.Xue, T.Jiang, and J.Shi, "Animal intrusion detection based on convolutional neural network," in 2017 17th International Symposium on Communications and Information Technologies (ISCIT), Cairns, QLD, Australia, 2017, pp. 1–5, doi: 10.1109/ISCIT.2017.8261234.

[2] D.Yudin, A.Sotnikov, and A.Krishtopik, "Detection of Big Animals on Images with Road Scenes using Deep Learning," in 2019 International Conference on Artificial Intelligence: Applications and Innovations (IC-AIAI), Belgrade, Serbia, 2019, pp. 100–103, doi: 10.1109/IC-AIAI48757.2019.00028.

[3] X.Hao, G.Yang, Q.Ye, and D.Lin, "Rare Animal Image Recognition Based on Convolutional Neural Networks," in 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Suzhou, China, 2019, pp. 1–5, doi: 10.1109/CISP-BMEI48845.2019.8965748.

[4] N.Li, W.Kusakunniran, and S.Hotta, "Detection of Animal Behind Cages Using Convolutional Neural Network," in 2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTICON), 2020, pp. 242–245, doi: 10.1109/ECTI-CON49241.2020.9158137.

[5] A. M. Roy, J. Bhaduri, T. Kumar, K. Raj, "WilDect-YOLO: An Efficient and Robust Computer Vision-Based Accurate Object Localization Model for Automated Endangered Wildlife Detection," Ecological Informatics, vol. 75, July 2023, 101919.

[6] N. K. El Abbadi, E. M. T. A. Alsaadi, "An Automated Vertebrate Animals Classification Using Deep Convolution Neural Networks," 2020 International Conference on Computer Science and Software Engineering (CSASE), Duhok, Iraq, 2020, pp. 72–77, doi:10.1109/CSASE48920.2020.9142070.

[7] M. A. Kayumbek Rakhimov, A. Ibragimov, "Animal Habit Monitoring System on the Roadside to avoid animal collisions with Support Vector Machine Model," 2020 International Conference on Information Science and Communications Technologies (ICISCT).

[8] Yu-Chen Chiu, Chi-Yi Tsai, Mind-Da Ruan, Guan-Yu Shen, TsuTian Lee, "Mobilenet-SSDV2: An improved object detection model for embedded systems," 2020 International Conference on System Science and Engineering (ICSSE), IEEE, pp. 1–5.

[9] R. Chandrakar, R. Raja, & A. Miri, "Animal detection based on deep convolutional neural networks with genetic segmentation," Multimedia Tools Appl 81, pp. 42149–42162 (2022), https://doi.org/10.1007/s11042-021-11290-4.

[10] A. Biglari, W. Tang, "A Vision-Based Cattle Recognition System Using TensorFlow for Livestock Water Intake Monitoring," IEEE Sensors Letters, vol. 6, no. 11, Nov. 2022, Art no. 5501404, doi: 10.1109/LSENS.2022.3215699.

[11] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet," Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems, 2002, pp. 96–107.

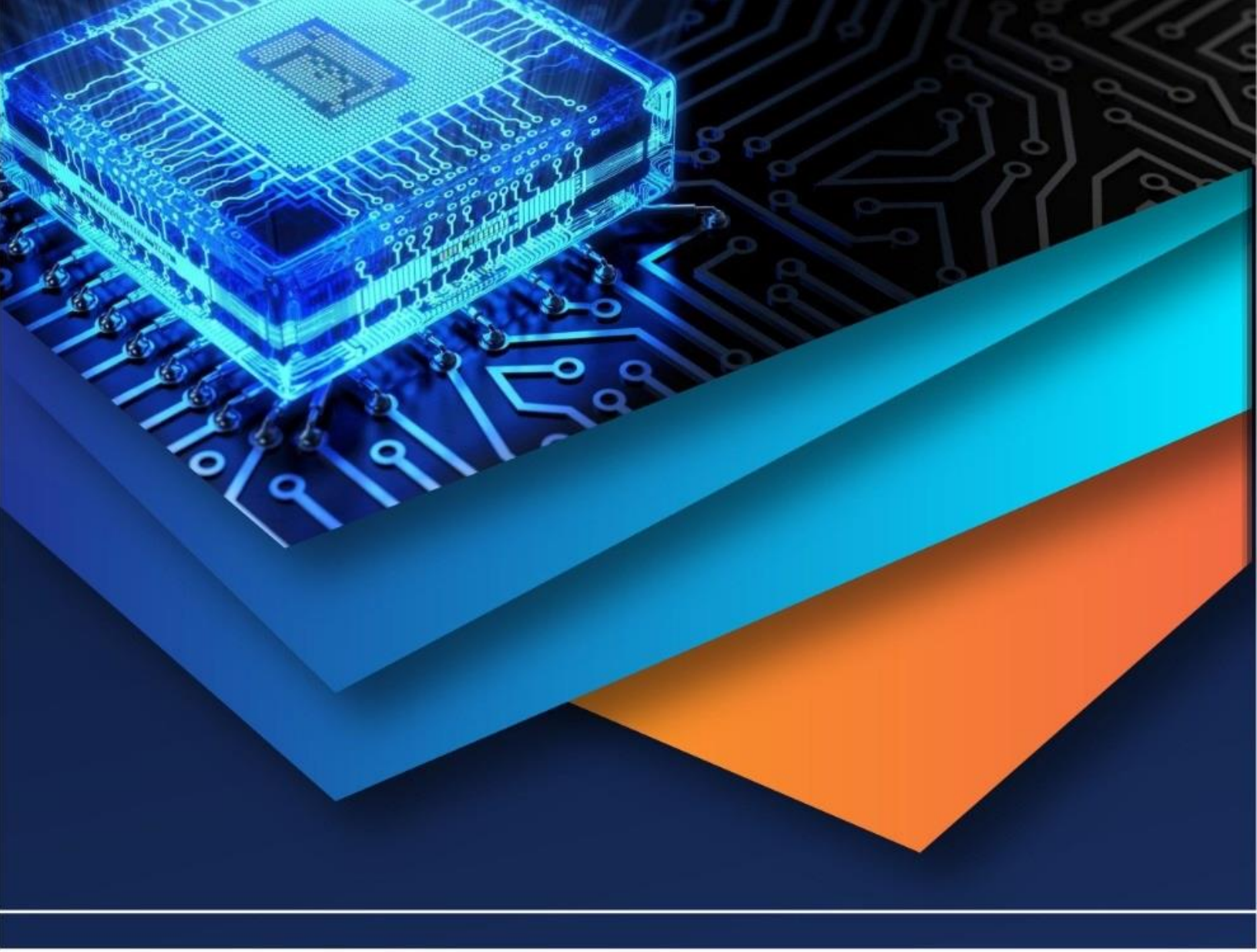
[12] [12] V. Dyo, S. A. Ellwood, D. W. Macdonald, A. Markham, C. Mascolo, S. Scellato, N. Trigoni, R. Wohlers, K. Yousef, "Evolution and sustainability of a wildlife monitoring sensor network," Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems, SENSYS 2010, Zurich, Switzerland, November 2010, pp. 127–140.

[13] [13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan, "Object detection with discriminatively trained part-based models," IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 9, pp. 1627–1645, Sep. 2010, doi: 10.1109/TPAMI.2009.167.

[14] [14] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Advances in Neural Information Processing Systems, vol. 25, F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2012, pp. 1097–1105. Accessed: Oct. 22, 2019. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[15] [15] S. Li, L. Yang, J. Huang, X.-S. Hua, L. Zhang, "Dynamic anchor feature selection for single-shot object detection," Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 6608–6617, doi: 10.1109/ICCV.2019.00671.

[16] [16] K. Chen, J. Li, W. Lin, J. See, J. Wang, L. Duan, Z. Chen, C. He, J. Zou, "Towards accurate one-stage object detection with AP-loss," Proc. IEEE/CVF Conf. Comput.Vis. Pattern Recognit. (CVPR), Long Beach, CA, USA, Jun. 2019, pp. 5114–5122, doi:10.1109/CVPR.2019.00526.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY