MosQNet-SA: An Explainable Convolutional-Attention Network for Mosquito Classification with Potential Application as a RESTful API for Dengue and Malaria Risk Mapping

by

Examination Roll:
Md. Akmol Masud (192327)
Kamrun Nahar Allo (192311)
Sabrina Siddiki Adity (192293)

A research project report submitted to the
Institute of Information Technology
in partial fulfillment of the requirements for the degree of
Bachelor of Science in
Information and Communication Technology

Supervisor: Prof. Dr. Mohammad Shahidul Islam

Institute of Information Technology
Jahangirnagar University
Savar, Dhaka-1342

June 2024

DECLARATION

We hereby affirm that this thesis is founded upon our own research findings. By reference, materials discovered by other researchers are cited. No previous dissertation, in its entirety or in part, has been submitted in pursuit of this degree.

Md. Akmol Masud	Kamrun Nahar Allo	Sabrina Siddiky Adity
Roll: 192327	Roll: 192311	Roll: 192293

CERTIFICATE

This is to certify that the thesis entitled MosQNet-SA: An Explainable Convolutional Attention Network for Mosquito Classification with Potential Application as a RESTful API for Dengue and Malaria Risk Mapping has been prepared and submitted by **Md.Akmol Masud**, **Kamrun Nahar Allo**, and **Sabrina Siddiky Adity** in partial fulfillment of the requirement for the degree of Bachelor of Science (Honours) in Information Technology in June 2024.

Prof. Dr. Mohammad Shahidul Islam
Supervisor

Accepted and approved in partial fulfillment of the degree Bachelor of Science (honors) requirement in Information Technology.

Prof. Fahima Tabassum
Chairman

Prof. Dr. Mohammad Shahidul Islam
Member

Prof. Dr. M. Mesbahuddin Sarker
Member

Prof. Dr. Md. Hasanul Kabir
External Member

ACKNOWLEDGEMENTS

On this occasion, we wish to extend our most profound appreciation and gratitude to all those who contributed to the completion of this report.

Dr. Mohammad Shahidul Islam, a Professor at the Institute of Information Technology (IIT) of Jahangirnagar University, is responsible for finalizing this thesis. Throughout the investigation, he has furnished us with various publications pertinent to the subject at hand, including books, journals, and papers. We would not have been able to meet the deadline for the assignment without his invaluable assistance, in addition to his cordial support and flexible schedule. We want to begin by extending our heartfelt appreciation to him for all he has done for us, encompassing guidance, perceptive suggestions, support, and harmonious collaboration.

The assistance of the Director of IIT in ensuring that the assignment was completed on time is greatly appreciated. Additionally, we would like to take this opportunity to express our gratitude to the remaining academic staff members of IIT who contributed to the accomplishment of this endeavor in some capacity.

Everyone else who has contributed to our success has our sincere appreciation. We want to extend our heartfelt gratitude to all individuals who provided us with any assistance during the completion of this assignment, be it direct or indirect.

Lastly, we wish to extend our sincere appreciation to all individuals affiliated with Jahangirnagar University and IIT for their invaluable assistance and support, in addition to our close circle of friends and family.

ABSTRACT

Vector-borne diseases pose a significant global health challenge, highlighting the need for effective disease vector surveillance and mosquito classification. This thesis presents MultiPathAttentionNet (MPANet), an Explainable Convolutional-Attention Network designed for accurate and efficient mosquito species classification. MPANet leverages CNNs and attention mechanisms to extract relevant features and classify mosquitoes, achieving high accuracy with up to 10 times fewer parameters and faster inference times than state-of-the-art models.

Extensive evaluations using explainable AI techniques like saliency maps, Grad-CAM, and SHAP confirmed MPANet's superior performance, reliability, and trustworthiness. Deploying MPANet in a RESTful API enables seamless integration into applications for real-time mosquito classification and dynamic risk mapping of vector-borne diseases like dengue and malaria. Through this novel, efficient, and explainable deep learning approach, this research contributes to more effective vector-borne disease surveillance, monitoring, and control efforts globally.

Keywords: Mosquito Detection, Convolutional-Attention Networks, Explainable AI, Disease Surveillance, Vector-borne Diseases, Dengue, Malaria, RESTful API

LIST OF FIGURES

Figure

TABLE OF CONTENTS

CHAPTER I

Introduction

1.1 Significance of Vector-Borne Diseases

Mosquito-borne diseases present a substantial global health problem, burdening communities globally, especially in tropical and subtropical areas. Diseases like dengue fever, malaria, Zika, chikungunya, yellow fever, leishmaniasis, and lymphatic filariasis still claim millions of lives each year. Sandflies, mosquitoes (Aedes, Anopheles, and Culex), and other organisms act as vectors for the spread of these diseases. Climate change, urbanization, and globalization are among the factors that affect the occurrence and spread of these diseases, both in terms of their prevalence and geographical distribution.

Aedes mosquitoes, which spread dengue fever, have seen a significant increase in cases, with an estimated annual incidence of 100–400 million infections, primarily in Asia, the Pacific, the Americas, Africa, and the Caribbean. Anopheles mosquito-transmitted malaria, which will cause 241 million cases and 627,000 fatalities worldwide in 2020, continues to be a serious threat. The majority of these cases and deaths occurred in Sub-Saharan Africa and Southeast 0.7–1 Asia, despite substantial efforts to suppress the disease.

The Zika virus had substantial epidemics between 2015 and 2016, particularly in Brazil. On the other hand, chikungunya has been documented in more than 60 countries across various continents, with notable outbreaks occurring in India, Italy, the Caribbean, and South America. Yellow fever, predominantly found in Africa and Central and South America, results in approximately 200,000 cases and 30,000 fatalities per year, despite the presence of a very efficient vaccine. With an annual occurrence of 0.7-1 million new cases, the sandfly-transmitted leishmaniasis affects 12

million people worldwide. The use of insecticide-treated nets, indoor residual spraying, better environmental management, better surveillance systems, immunizations when available, health education, and ongoing research and development to improve vaccines, treatments, and vector control methods are all part of controlling these diseases, which are particularly prevalent in tropical and subtropical regions. Southern Europe. Lymphatic filariasis, a disease carried through mosquito bites, affects some 120 million individuals globally, primarily in tropical and subtropical areas of Africa and Asia.

1.2 Challenges in Vector-Borne Disease Control

Controlling these diseases that are spread by vectors includes using insecticide-treated nets, indoor residual spraying, better environmental management, better surveillance systems, immunizations when they are available, teaching people about health, and ongoing research and development to make vaccines, treatments, and vector control methods better. Tackling these disorders necessitates a comprehensive strategy that entails cooperation among governments, healthcare organizations, and research institutions to alleviate their significant consequences for public health.

1.3 The Scientific Advancement

Recent scientific and technical breakthroughs in different fields have provided new opportunities for tackling the difficulties presented by vector-borne diseases. Although advancements have been achieved in various areas like epidemiology, entomology, molecular biology, genetics, vaccine development, drug discovery, and vector control strategies, it is crucial to acknowledge the potential benefits of emerging technologies and interdisciplinary collaborations. Progress in fields such as remote sensing, geographic information systems (GIS), and environmental modeling has improved our capacity to observe and forecast the transmission of vector-borne diseases. This allows us to take preventative action and allocate resources more effectively. In addition, state-of-the-art diagnostic equipment, quick testing kits, and point-of-care devices have made detecting diseases early and providing timely treatment easier, which is crucial for effective illness management. Significant progress has been made in artificial intelligence (AI) and machine learning, namely in convolutional neural networks (CNNs) and attention processes. These improvements show potential for effectively tackling the mentioned difficulties. Convolutional neural networks (CNNs)

are very good at tasks that involve putting images into groups. This is because they can automatically find and extract hierarchical features from input that has not been processed. By incorporating attention mechanisms, CNNs can improve their performance by selectively attending to the most pertinent features of the input data. This leads to enhanced classification accuracy and a deeper understanding of the decision-making process.

AI and machine learning approaches can be utilized in multiple facets of vector-borne illness management, including disease surveillance, risk mapping, and decision support systems. CNNs and attention processes can be employed to analyze satellite imagery and environmental data to detect possible breeding sites for disease vectors. This allows for focused interventions and the allocation of resources. Moreover, these methods can be utilized in creating sophisticated diagnostic instruments, using the ability of image recognition to identify and categorize disease carriers or illness-causing agents from tiny images or samples collected in the field. This can optimize the diagnostic procedure, minimizing the time and resources needed for manual analysis. In addition, AI and machine learning can be meaningful in combining many data sources, including epidemiological data, environmental factors, and socioeconomic indicators, to create predictive models and early warning systems for outbreaks of vector-borne diseases. By utilizing these methodologies, public health officials can make well-informed judgments and promptly implement measures to reduce the effects of these diseases.

1.4 The Contribution of This Thesis

This thesis presents the development and evaluation of a Convolutional-Attention Network: MosQNet-SA, a novel deep-learning model designed explicitly for mosquito classification. Our model aims to achieve high classification accuracy while incorporating an efficient architecture with significantly reduced inference time and a smaller parameter footprint. It is well-suited for real-time applications and deployment on edge devices with limited computational resources.

The MosQNet-SA model leverages the power of convolutional neural networks (CNNs) and attention mechanisms to efficiently extract relevant features from input images and accurately classify mosquito species. By incorporating multiple attention paths, our model can effectively capture and focus on the most discriminative features, enhancing classification performance while reducing computational complexity.

One of the key advantages of our model is its exceptional efficiency, achieving up to 10 times fewer parameters and significantly faster inference times compared to current state-of-the-art (SOTA) models. This efficiency is crucial for real-time applications on edge devices, where timely and reliable results are essential, and storage constraints are a consideration. To substantiate the superiority of our model, we conducted extensive evaluations and incorporated explainable AI techniques. These techniques, including saliency maps, Gradient-weighted Class Activation Mapping (Grad-CAM), Shapley Additive Explanations (SHAP), and others, were employed to visualize and interpret the model's decision-making process. These analyses confirmed that our model achieves higher classification accuracy and makes decisions based on the most relevant and discriminative features, thereby enhancing its reliability and trustworthiness.

1.5 Objectives

- Develop a novel deep learning model (MosQNet-SA) for accurate mosquito species classification.
- Incorporate explainable AI techniques to ensure transparency and interpretability in the model's decision-making process.
- Achieve high classification accuracy with reduced inference time and a smaller parameter footprint.
- Enable scalable deployment through a RESTful API for real-time mosquito classification services.
- Facilitate dynamic risk mapping for vector-borne diseases like dengue and malaria.

1.6 Research Outline

The remaining sections of the report are organized as follows: Chapter II presents a comprehensive examination of relevant literature, providing the necessary background and context for the study. Chapter III details the processes involved in the acquisition and transformation of the dataset. Chapter IV outlines the methodology employed in the research, describing the approach and techniques used. Chapter V evaluates the performance of MosQNet-SA, offering an assessment of its effectiveness. Chapter VI delves into the explainability of the novel model, highlighting how the model's decisions can be understood and interpreted. Chapter VII explores the potential applications of MosQNet-SA in a RESTful API framework, demonstrating its practical utility. Finally, Chapter VIII concludes the report with a summary of findings, conclusions drawn from the research, and suggestions for future work.

CHAPTER II

Literature Review

2.1 Introduction

The classification of mosquito species has become a critical focus in the global fight against vector-borne diseases. In recent years, there has been a significant surge in research applying machine learning techniques to this task, aiming to automate and improve the accuracy of mosquito identification. This growing body of work has the potential to revolutionize disease control efforts by providing faster, more accurate, and more accessible methods for identifying disease-carrying mosquito species. The following review synthesizes the findings from 25 key studies in this rapidly evolving field, highlighting the diverse approaches, significant achievements, and persistent challenges in machine learning-based mosquito classification.

2.2 Image-based Classification Methods

2.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have emerged as a powerful tool for image-based mosquito classification, demonstrating remarkable accuracy in distinguishing species and genera.

Park et al. [1] achieved outstanding results, with over 97% accuracy in classifying eight mosquito species using fine-tuned Deep CNNs. Their study is particularly noteworthy for its large dataset of 3,600 images capturing mosquitoes' various postures and deformation conditions. They demonstrated the potential of transfer learning in this domain by applying data augmentation techniques and fine-tuning pre-trained models. Building on this foundation, researchers [2] proposed an innovative pipeline designed explicitly for low-cost IoT sensors. By leveraging models such as VGG16,

ResNet50, and custom CNNs, they achieved an impressive 98% accuracy. This study is particularly significant as it addresses the need for cost-effective, deployable solutions in resource-constrained settings. Another team [3] further advanced CNN-based approaches by modifying the VGG16 architecture. Their model achieved high accuracy (94.66%-98.92%) across multiple datasets, demonstrating robust performance in various conditions. Their work included extensive ablation studies to determine the optimal architecture modifications. A study by Siddiqui and Jain [4] focused on the challenging task of distinguishing between mosquito genera, developing a CNN model that achieved 84.51% accuracy on a dataset of 1,800 images from three genera (Aedes, Anopheles, and Culex). While the accuracy was lower than some other studies, their work highlighted the difficulties in classifying visually similar genera. Okayasu et al. [5] conducted a comprehensive comparative study of conventional and deep learning methods for mosquito species classification. They found that deep learning methods, particularly ResNet, outperformed conventional handcrafted feature-based methods, but only when data augmentation was applied.

2.2.2 Transfer Learning and Ensemble Methods

Researchers have increasingly turned to transfer learning and ensemble methods to enhance classification performance, especially when dealing with limited datasets. One group [6] conducted a thorough comparison of VGG-16, Inception V3, and MobileNetV2 architectures. Their ensemble method, which combined these models, achieved over 98% accuracy. This study demonstrated the power of leveraging multiple architectures to improve classification performance. Another significant contribution [7] introduced a novel feature selection technique called RIFS (ROI-based Image Filtering and Selection). Combined with Extremely Randomized Trees, this approach achieved an impressive 99.2% accuracy. Other researchers [8] explored transfer learning techniques to classify Aedes and Culex mosquito genera. By leveraging pre-trained models such as VGGNet, ResNet, and GoogLeNet, they demonstrated the potential of transfer learning to improve classification performance with limited training data.

2.3 Smartphone-based Systems

Recognizing the potential for widespread adoption and citizen science initiatives, several studies have focused on developing smartphone-based classification systems. One team [9] made significant strides in this direction, achieving 80% accuracy using

an Inception-ResNet V2 model on a large dataset of 25,867 smartphone images. Their study is particularly noteworthy for its use of real-world, field-captured images. Another study [10] conducted a comparative study of LeNet, AlexNet, and GoogLeNet architectures on mobile phone images. They found GoogLeNet to be the most effective, achieving 76.2% accuracy. Further research [11] developed a smartphone-based system achieving 77% accuracy for nine species. This work emphasized the potential of citizen science in mosquito surveillance and the importance of user-friendly interfaces for widespread adoption. Researchers [12] developed "DenGue CarB," a web-based application using machine learning techniques, including CNNs, SVMs, and KNNs, for mosquito classification.

2.4 Audio-based Classification Methods

In parallel to image-based approaches, researchers have explored using mosquito wingbeat sounds for classification, opening up new possibilities for sensor-based identification systems. One innovative study [13] developed a TinyML system that achieved 88.3% accuracy using wingbeat sound spectrograms. This approach demonstrates the potential for low-power, always-on mosquito detection systems. Another team [14] pushed the boundaries of audio-based classification, achieving an impressive 96% accuracy in classifying six species using deep learning on wingbeat recordings. Researchers [15] proposed a hybrid model combining CNNs and SVMs for analyzing audio data from mosquito wingbeats. This innovative approach demonstrated the potential of combining different machine-learning techniques for improved classification. Other work [14] used infrared recordings of wingbeats and a Gaussian mixture model, achieving over 80% accuracy for gender and genus classification. An earlier study [16] applied machine learning and audio analysis techniques to insect recognition in intelligent traps, laying the groundwork for future audio-based classification systems.

2.5 Hybrid and Novel Approaches

As the field matures, researchers are increasingly exploring hybrid and novel approaches that combine multiple data types or introduce innovative methodologies. One team [17] introduced a novel approach that integrated AI with wing geometric morphometry, achieving high accuracy rates (84-95%). This study is particularly noteworthy for combining traditional morphometric analysis with modern machine

learning techniques. Another group [18] conducted a comprehensive comparison of various algorithms on optical signals from mosquitoes, finding the Support Vector Machine to be the most effective for complex tasks. Researchers [19] developed an automatic mosquito-on-human-skin recognition system using deep learning, achieving high accuracy in identifying mosquitoes in real-world settings. An earlier study [20] used Support Vector Machines for vision-based perception and classification of mosquitoes, demonstrating that traditional machine learning techniques can still be highly effective when coupled with careful feature engineering.

2.6 Challenges and Future Directions

Despite the significant progress in the field, several common challenges and limitations were identified across studies:

1. Limited dataset sizes and diversity
2. Difficulty in distinguishing visually similar species
3. Image quality issues
4. Generalization across environments
5. Balancing accuracy and computational efficiency

A comprehensive review [21] of these challenges and potential future directions for machine learning in mosquito control synthesized findings from 120 papers, offering valuable insights into the field's current state. Other researchers [22] explored machine learning to assist in classifying citizen reports of disease-carrying mosquitoes, highlighting the potential for combining machine learning with citizen science initiatives.

2.6.1 The Potential of MosQNet-SA

The comprehensive review of existing mosquito classification approaches reveals significant progress and persistent challenges. In this context, our novel MosQNet-SA model emerges as a promising solution to address current limitations and advance the field.

MosQNet-SA's potential contributions include:

- Enhanced accuracy in classifying visually similar species

- Improved generalization across diverse environments
- Efficient balancing of accuracy and computational demands
- Integration of multiple data modalities (visual, audio, and contextual)

By leveraging the strengths of existing approaches while innovating to overcome their limitations, MosQNet-SA aims to set a new standard in automated mosquito classification. Key areas for ongoing development include:

1. Multimodal data integration: Combining visual and audio data with contextual information (e.g., environmental factors, seasonal patterns) to enhance classification robustness.
2. Adaptive learning: Developing mechanisms for continuous model improvement as new data becomes available, ensuring relevance across changing environments and mosquito populations.
3. User-friendly interface: Creating accessible tools for both researchers and citizen scientists, maximizing the model's real-world impact and facilitating widespread adoption.
4. Scalable architecture: Designing a flexible system capable of expanding to new species and geographical areas without significant retraining.

The future of mosquito classification is at a critical juncture, with machine learning poised to revolutionize global health efforts. MosQNet-SA aspires to be at the forefront of this transformation, driving innovation in the fight against mosquito-borne diseases. By addressing current limitations and building on collective insights from the field, MosQNet-SA can significantly advance our ability to monitor and control disease-carrying mosquito populations.

Ultimately, the success of MosQNet-SA could translate into tangible improvements in public health outcomes worldwide. As we continue to refine and deploy this technology, we envision a future where rapid, accurate mosquito classification becomes integral to global disease prevention strategies, contributing to a healthier, more resilient world.

CHAPTER III

Dataset Acquisition and Transformation

3.1 Dataset Acquisition

To ensure a diverse and comprehensive representation of mosquito species, the dataset was compiled from four distinct sources:

- **MosquitoAlert.com**: This citizen science platform, dedicated to monitoring and controlling mosquito populations, contributed 1,234 images of various mosquito species, including *Anopheles*, *Aedes*, and *Culex*. The photos, submitted by participants worldwide, were meticulously screened and vetted for quality and accuracy.
- **Mendeley Data**: A publicly accessible dataset from the Mendeley Data repository [23] provided 876 well-annotated images of *Aedes* and *Culex* mosquitoes. Curated by researchers, these images are particularly suited for machine learning tasks.
- **IEEE DataPort**: The IEEE DataPort repository [24], maintained by the Institute of Electrical and Electronics Engineers (IEEE), offered 748 images specifically focused on *Aedes* and *Culex* mosquito species. This dataset, created by the research community, ensured high relevance and reliability.
- **Dryad Digital Repository**: Sourced from the Dryad Digital Repository, which hosts research data associated with scholarly publications, 600 images of *Anopheles* mosquitoes were included. These images were accompanied by detailed metadata and annotations.

The images from these diverse sources were carefully inspected and selected based on their quality, clarity, and suitability for the research objectives of this study.

3.2 Dataset Allocation

The research dataset encompasses three significant mosquito species: *Anopheles*, *Aedes*, and *Culex*, each represented by 1,000 images. These species were selected due to their importance in transmitting various mosquito-borne diseases and their prevalence in different regions worldwide. Ensuring an equal distribution of 1,000 images per species was a deliberate choice to maintain a balanced representation and prevent potential biases during model training and evaluation. An imbalanced dataset could skew performance, favoring the over-represented species.

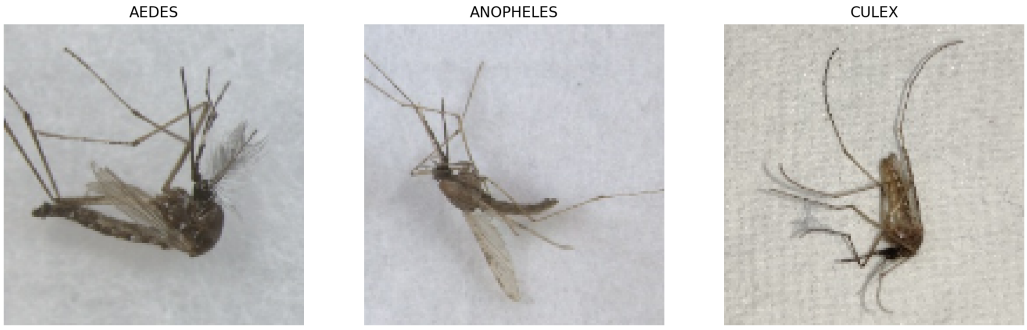

Figure 3.1: Single instance of AEDES, ANOPHELES, and CULEX mosquito

The mosquitos and their labels

- ***Anopheles***: This genus includes several primary vectors for malaria transmission. The dataset features 1,000 images of *Anopheles* mosquitoes, capturing their distinctive morphological features and postures.
- ***Aedes***: The *Aedes* genus comprises several species responsible for transmitting diseases such as dengue, chikungunya, and Zika virus. The dataset includes 1,000 images of *Aedes* mosquitoes, representing their diverse appearances and behaviors.
- ***Culex***: Species from the *Culex* genus are known vectors for diseases including West Nile virus, lymphatic filariasis, and Japanese encephalitis. The dataset contains 1,000 images of *Culex* mosquitoes, showcasing their characteristic features and habitats.

By allocating an equal number of images for each species, the dataset ensures that the machine learning models have sufficient exposure to the diverse morphological characteristics and variations within each species, promoting robust and generalizable performance.

3.3 Image Augmentation

Several image augmentation techniques were applied to enhance the dataset's diversity and robustness. These augmentations are summarized in the following table:

Augmentation Techniques	**Description**
Width Shift Range	Shift the width by up to 30%
Height Shift Range	Shift the height by up to 20%
Shear Range	Apply shear transformation by 10%
Zoom Range	Apply zoom transformation by 40%
Horizontal Flip	Enable horizontal flipping
Fill Mode	Nearest neighbor filling mode

Table 3.1: Image Augmentation Techniques

These augmentation techniques were used to introduce variability and improve the generalization capabilities of the machine learning models.

CHAPTER IV

Methodology

4.1 Introduction

The methodology of this project is structured within a comprehensive framework that outlines the entire process. It begins with the initial model development, introducing nine state-of-the-art model types. This is followed by developing 16 models from these nine types to identify the most effective models for predicting mosquito species. Next, configuring callbacks is addressed to optimize the model training process through tailored callback functions. The baseline models are then assessed to establish performance benchmarks, including test loss, accuracy, macro F1 score, precision, and recall. Following this, the proposed model is introduced, highlighting its innovative aspects and anticipated improvements. Finally, a comparative analysis is conducted, contrasting the performance of the proposed model against the baseline models to evaluate its effectiveness and superiority. This structured approach ensures a thorough and systematic investigation, facilitating a clear understanding of the project's methodology and outcomes.

4.2 Baseline Model Development

These are the models considered for the experiment:

4.2.1 VGG (Visual Geometry Group)

VGG networks, such as VGG16 and VGG19, are known for their straightforward and uniform architecture, which includes 16 or 19 layers [25]. These networks use convolutional layers with small receptive fields (3×3 filters) followed by max-pooling

layers to reduce the spatial dimensions progressively. The final layers are fully connected, culminating in a softmax classifier. Despite their simplicity, VGG networks require many parameters and are computationally expensive, but they set new benchmarks for image classification tasks when introduced.

4.2.2 ResNet (Residual Network)

ResNet revolutionized deep learning by introducing residual blocks [26], which use skip connections to bypass one or more layers. This innovation addresses the vanishing gradient problem, enabling the training of much deeper networks, such as ResNet50 and ResNet101, with layers numbering in the hundreds. The skip connections help gradients flow more quickly through the network, improving training stability and performance. ResNet's ability to train intense networks effectively made it a cornerstone in the development of other complex architectures.

4.2.3 Inception

The Inception architecture [27], features innovative Inception modules that perform convolutions of various sizes (1×1, 3×3, 5×5) in parallel. These modules allow the network to capture multi-scale features efficiently. Inception networks also use dimensionality reduction techniques, such as 1×1 convolutions, to limit the number of parameters and computational cost. The architecture has evolved through several versions, with each iteration improving upon the previous one by introducing more sophisticated modules and optimization techniques.

4.2.4 Xception (Extreme Inception)

Xception, proposed by François Chollet, extends the Inception model by fully embracing depthwise separable convolutions [28]. This architecture replaces the traditional convolutional layers in Inception modules with depthwise separable convolutions, significantly reducing the number of parameters and computational cost. By decoupling the spatial and cross-channel correlations, Xception performs better while maintaining computational efficiency, making it a powerful model for image classification tasks.

4.2.5 Inception-ResNet

Inception-ResNet combines the strengths of both Inception modules and ResNet's residual connections [29]. This hybrid approach leverages Inception modules' multi-

scale feature extraction capabilities and the efficient training that residual connections enable. The result is a highly efficient network that performs exceptionally well on various image classification benchmarks. Inception-ResNet achieves high accuracy with improved training stability and convergence speed compared to its predecessors.

4.2.6 DenseNet (Densely Connected Convolutional Networks)

DenseNet introduces a novel connectivity pattern where each layer receives input from all preceding layers [30]. This dense connectivity ensures maximum information flow between layers, promoting feature reuse and alleviating the vanishing gradient problem. DenseNet models are compact and efficient, often requiring fewer parameters and computational resources than other deep networks while achieving competitive performance. The dense connections also encourage feature reuse, making the network more parameter-efficient and capable of learning rich feature representations.

4.2.7 MobileNet

MobileNet architectures are specifically designed for mobile and embedded vision applications, focusing on lightweight models with low computational demands [31]. They utilize depthwise separable convolutions to significantly reduce the number of parameters and operations compared to standard convolutions. MobileNet models, such as MobileNetV1, V2, and V3, are highly efficient, making them suitable for real-time applications on devices with limited computational power, such as smartphones and IoT devices.

4.2.8 EfficientNet

EfficientNet introduces a compound scaling method that simultaneously scales up network depth, width, and resolution in a balanced manner [32]. By carefully balancing these three dimensions, EfficientNet achieves state-of-the-art performance with fewer parameters and lower computational cost than previous models. The EfficientNet family, from B0 to B7, offers various sizes of models to cater to different computational budgets and performance needs, making it a versatile and powerful choice for image classification tasks.

4.2.9 NASNetMobile (Neural Architecture Search Network Mobile)

NASNetMobile results from neural architecture search (NAS), an automated process that optimizes network architectures for specific tasks and constraints [33]. De-

signed for mobile and low-power devices, NASNetMobile balances performance and efficiency by finding optimal architectural configurations through extensive search algorithms. This approach tailors the network to meet the needs of mobile environments, achieving high accuracy with reduced computational requirements, making it ideal for on-device applications.

4.3 Construction of The Model

4.3.1 Training Configuration Setup

To ensure a robust and comprehensive analysis of the results, 17 models with various architectures have been considered. These models represent diverse design principles and optimizations, each chosen for their unique features and proven effectiveness in different scenarios. The selection includes classical architectures like VGG16 and VGG19, renowned for their simplicity and depth, and advanced models such as ResNet50, ResNet101, and ResNet152, which incorporate residual connections to facilitate the training of deeper networks.

Model	**Optimizer**	**Total Params**	**Trainable Params**	**TP %**
VGG16	SGD	16,846,915	4,491,267	26.66
ResNet50	SGD	32,011,395	9,477,635	29.61
ResNet101	SGD	51,081,859	16,304,131	31.92
ResNet152	SGD	66,794,627	20,512,259	30.71
Xception	ADAM	29,285,163	8,427,011	28.78
InceptionV3	ADAM	23,935,011	7,287,491	30.45
InceptionResNetV2	ADAM	55,944,675	12,413,027	22.19
MobileNet	RMSprop	7,458,243	4,230,659	56.72
MobileNetV2	RMSprop	7,535,939	5,277,187	70.03
DenseNet121	ADAM	11,266,883	4,267,523	37.88
DenseNet169	ADAM	19,493,699	6,890,243	35.35
DenseNet201	ADAM	26,221,379	8,185,027	31.22
NASNetMobile	ADAM	8,630,167	4,359,683	50.52
EfficientNetB0	ADAM	9,327,526	5,279,747	56.60
EfficientNetB1	ADAM	11,853,194	5,689,347	48.00
EfficientNetB2	ADAM	13,570,812	7,417,947	54.66

Table 4.1: Model Details.

Xception and InceptionV3 leverage depthwise separable convolutions and multi-scale feature extraction, respectively, while InceptionResNetV2 combines the strengths of both Inception modules and residual connections. MobileNet and MobileNetV2

are optimized for mobile and embedded applications by efficiently using depthwise separable convolutions. The DenseNet variants (DenseNet121, DenseNet169, and DenseNet201) emphasize dense connectivity for maximum information flow and feature reuse.

NASNetMobile, created through neural architecture search, is tailored for mobile environments. Lastly, the EfficientNet models (B0, B1, and B2) employ a compound scaling method to balance network depth, width, and resolution, achieving high performance with fewer parameters. All models were trained using a learning rate of 0.001. The number of epochs varied per model, ranging from 38 to 121. The table 4.1 provides detailed information about the optimizer, total parameters, and the percentage of trainable parameters for each model.

This selection provides various models, each suited for different tasks and environments, ensuring a thorough evaluation and performance comparison.

4.3.2 Callback Setups

For the entirety of this experimentation, TensorFlow models [34] will be utilized, necessitating the implementation of several TensorFlow callbacks to optimize the training process. The following callbacks have been meticulously configured to enhance model performance and ensure robust training outcomes:

- **EarlyStopping**: This callback is essential for preventing overfitting by halting the training process if the validation loss does not improve for 20 consecutive epochs, conserving computational resources and avoiding unnecessary training.
- **ModelCheckpoint**: To ensure the best version of the model is retained, this callback saves the model with the lowest validation loss to a file named "BestM-model.h5". This allows for easy retrieval and deployment of the most effective model iteration.
- **TensorBoard**: TensorBoard logging provides an interactive and visual approach to monitoring the training process. This callback logs essential training and validation metrics and the model architecture, enabling detailed analysis and visualization of model performance over time.
- **ReduceLROnPlateau**: Adaptive learning is facilitated by this callback, which reduces the learning rate by 0.2 if there is no improvement in validation loss for ten epochs. This helps fine-tune the model by allowing more precise adjustments to the learning rate, with a minimum threshold set at `1e-6`.

- **CSVLogger**: For comprehensive record-keeping, this callback logs all training and validation metrics to a `training.log` CSV file. This ensures a detailed and structured record of the training process is maintained, which is essential for further analysis and reproducibility of results.

- **LearningRateScheduler**: This callback systematically reduces the learning rate by 0.9 every ten epochs during training. This gradual reduction helps stabilize the training process and fine-tune the model parameters progressively, potentially leading to better convergence and performance.

As detailed in the table below, these callbacks are integral to the training regimen. They provide dynamic adjustments and ensure the best possible outcomes from each model. Collectively, they contribute to more efficient, effective, and controlled model training and validation processes, ultimately enhancing the reliability and performance of the trained models.

4.4 Baseline Model Evaluation

Our evaluation encompassed a comprehensive analysis of various deep learning models, each meticulously configured with specific parameters, yielding a detailed array of performance metrics in the table below. Standout performers such as Xception, InceptionV3, and InceptionResNetV2 consistently excelled across critical evaluation criteria: training loss, validation loss, and accuracy on both validation and test datasets. These models showcased robust generalization capabilities, consistently achieving lower losses and higher accuracies, which underscore their effective feature extraction and robust model performance.

Conversely, models like VGG19 and ResNet101 exhibited relatively higher losses and lower accuracies, indicating potential challenges in capturing intricate data patterns or susceptibility to overfitting. These discrepancies in performance can be attributed to factors such as architectural complexity, parameter volume, and the models' ability to discern subtle features within the dataset. Models with simpler architectures or fewer parameters, such as VGG16 and MobileNet, demonstrated competitive results but may have encountered limitations in capturing fine-grained details due to their architectural constraints.

Integrating advanced optimization techniques, such as depthwise separable convolutions in MobileNet and dense connectivity in DenseNet, significantly contributed to performance enhancements observed across these models. The exceptional per-

Model	Epochs	Tr Loss	Val Loss	Test Loss	Tr Acc	Val Acc	Test Acc
VGG16	121	0.12	0.16	0.20	95.88%	93.62%	93.37%
ResNet50	75	0.15	0.21	0.20	95.01%	91.01%	91.64%
ResNet101	53	0.20	0.24	0.30	92.95%	90.14%	89.34%
ResNet152	59	0.20	0.26	0.30	93.38%	90.14%	90.49%
Xception	51	0.05	0.17	0.22	98.12%	92.46%	93.37%
InceptionV3	62	0.02	0.16	0.23	99.28%	95.07%	94.81%
InceptionResNetV2	68	0.0075	0.1121	0.053	99.71%	97.10%	98.56%
MobileNet	73	0.01	0.10	0.15	99.78%	97.10%	97.12%
MobileNetV2	67	0.0336	0.1721	0.1191	98.92%	94.49%	97.41%
DenseNet121	80	0.01	0.0667	0.0845	99.78%	97.68%	97.69%
DenseNet169	38	0.0359	0.1065	0.0589	98.77%	96.23%	97.98%
DenseNet201	61	0.0015	0.0574	0.0349	99.96%	98.84%	99.14%
NASNetMobile	43	0.1102	0.2317	0.2037	95.84%	92.17%	93.95%
EfficientNetB0	67	0.0065	0.0613	0.0770	99.78%	97.10%	97.41%
EfficientNetB1	57	0.0026	0.0722	0.0838	100.00%	98.55%	97.41%
EfficientNetB2	46	0.0069	0.0811	0.0732	99.89%	98.26%	97.69%

Table 4.2: Loss and Accuracies of various deep learning models.

formance of Inception-based architectures can be attributed to their adeptness in multi-scale feature extraction and efficient parameter utilization strategies.

Moreover, models like EfficientNetB0, B1, and B2 consistently demonstrated balanced performance across various metrics, showcasing the efficacy of the compound scaling method in optimizing network depth, width, and resolution for superior performance outcomes. However, it is crucial to acknowledge potential dataset biases, model hyperparameter variations, and optimizer selection's influence, all of which play pivotal roles in determining overall model performance.

In conclusion, our extensive evaluation provides deep insights into the strengths and weaknesses of different deep learning architectures. These insights are instrumental in creating a novel model tailored to specific tasks and datasets, thereby enhancing the reliability and performance of deployed models in real-world applications.

4.5 The Proposed Model: MosQNet-SA

MosQNet-SA is a novel deep convolutional neural network designed specifically for mosquito classification. Its architecture is meticulously crafted to optimize feature extraction and boost classification accuracy. The model comprises several specialized blocks, each meticulously engineered to enhance different aspects of the feature extraction process.

4.5.1 Overall Model Architecture

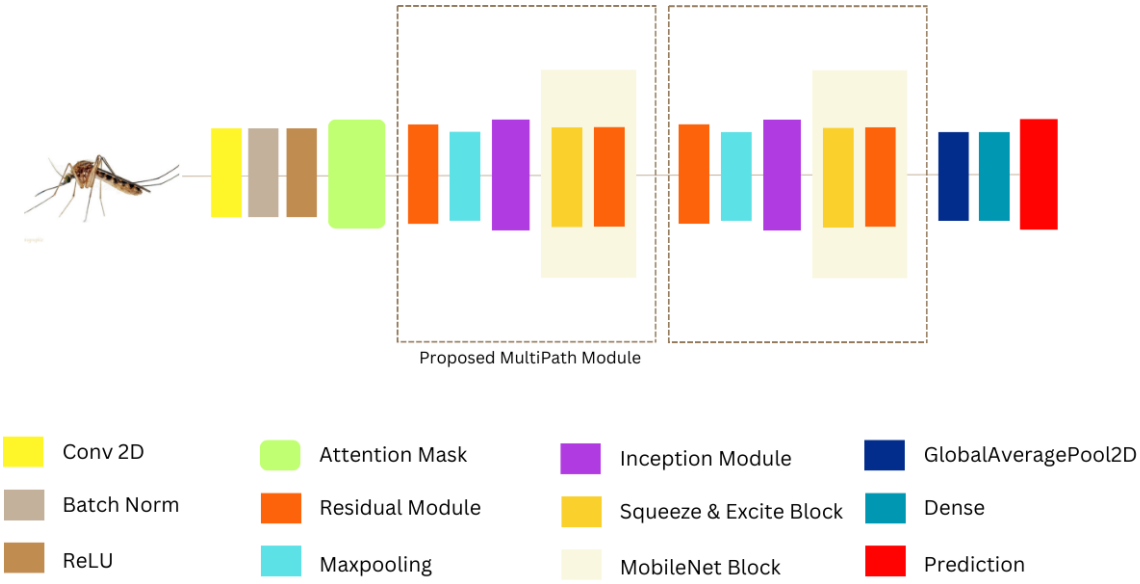

Figure 4.1: Overall Model Architecture

The Overall Model Architecture Diagram (Figure 4.1) provides a detailed overview of the entire model structure, illustrating the flow of information from input to output. This diagram highlights the critical components such as the initial convolutional layer, which captures basic features from the input image; the residual blocks, which facilitate more profound network training by allowing gradient flow; the Inception-like blocks, which capture multi-scale features through parallel convolutions; the MBConv blocks, which provide efficient and effective feature extraction; the spatial attention block, which focuses on significant regions within feature maps; and the final classification layers, which aggregate features for the final prediction.

4.5.2 Residual Block

The Residual Block (Figure 4.2) is a standout feature of MosQNet-SA, consisting of two consecutive convolutional layers linked by residual connections. These connections enable the smooth flow of gradients during training, alleviating the vanishing gradient problem and enhancing the network's ability to learn complex features efficiently. By allowing gradients to bypass specific layers, residual connections make it possible to train much deeper networks, thus improving overall model performance.

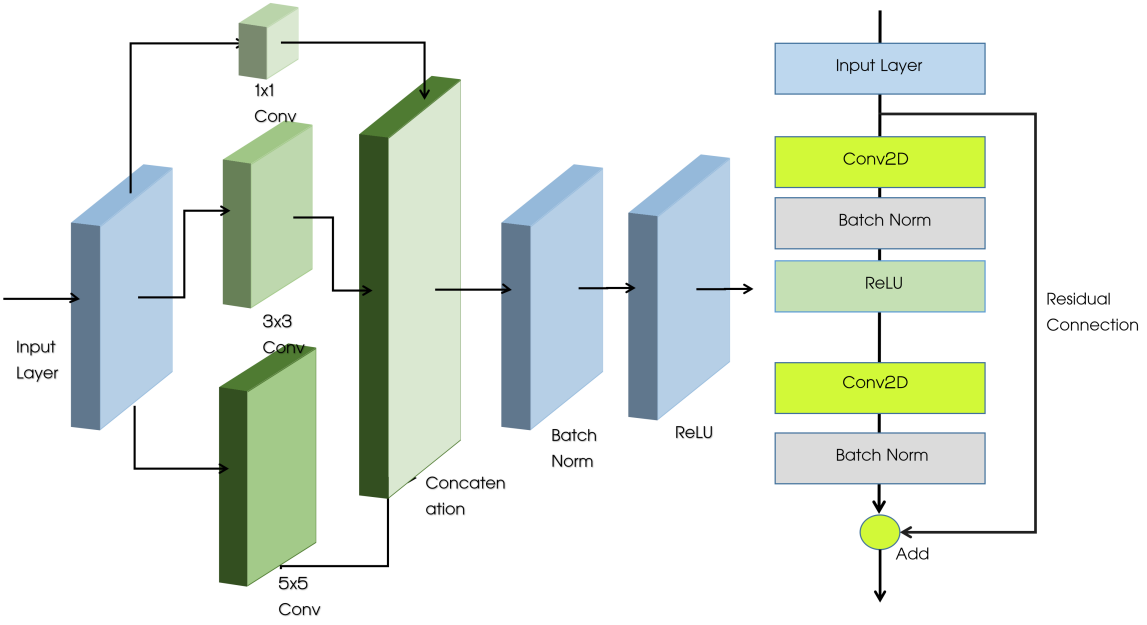

Figure 4.2: Inception and Residual Module

4.5.3 Inception Block

The Inception Block (Figure 4.2) is designed to capture a rich array of features by integrating multiple convolutional layers with different kernel sizes in parallel. This configuration allows the model to capture features at various spatial scales, promoting richer and more diverse feature representation. By using concatenated outputs with batch normalization and ReLU activation, the Inception Block improves learning dynamics. It ensures that the model can effectively learn from the input data, enhancing its ability to classify images accurately.

4.5.4 Squeeze and Excitation Block

MosQNet-SA incorporates Squeeze and Excitation (SE) Blocks , which utilize depthwise separable convolutions for efficient feature extraction while reducing computational cost. These blocks consist of an expansion phase and an output phase and optionally include an SE block to recalibrate channel-wise feature responses adaptively. This adaptive recalibration enhances the model's ability to focus on important features, thus improving overall performance. By efficiently managing computational resources, SE Blocks make MosQNet-SA powerful and resource-effective.

4.5.5 Spatial Attention Block

The Spatial Attention Block is another innovative feature of MosQNet-SA. This block dynamically attends to spatial regions of interest within the feature maps, enhancing model discriminability by focusing on relevant image regions while suppressing irrelevant or noisy information. By directing attention to the most significant parts of the image, the Spatial Attention Block improves the model's ability to make accurate classifications, particularly in complex scenes with a lot of background noise.

4.5.6 Model Training Pipeline

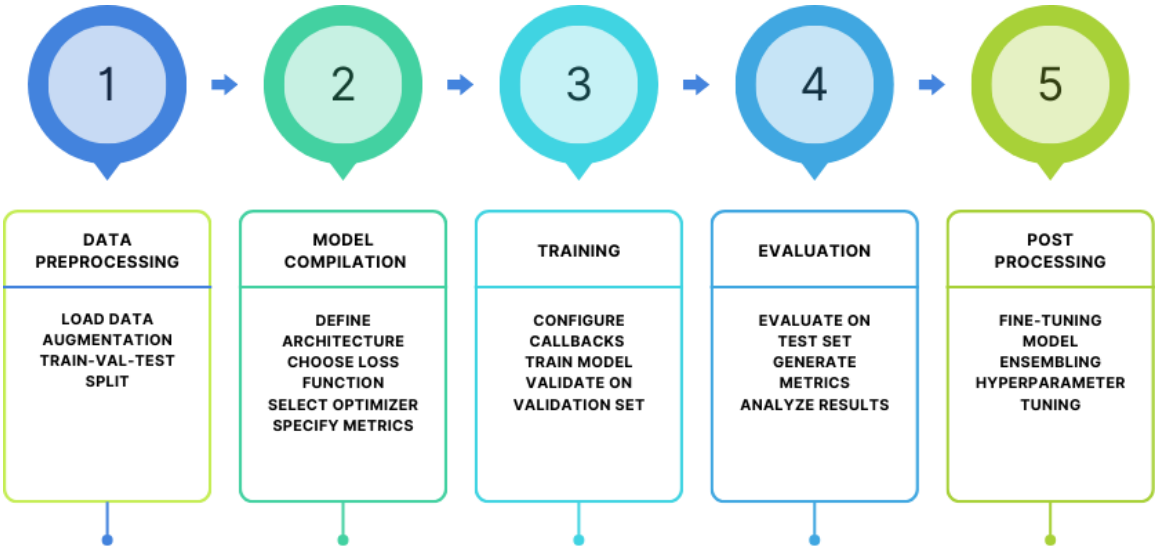

Figure 4.3: Overall Model Training Pipeline

The Overall Model Training Pipeline (Figure 4.3) outlines the comprehensive training process for MosQNet-SA, including data preprocessing, model compilation, training with specified callbacks, and evaluation metrics. Data preprocessing prepares

the dataset for training by normalizing and augmenting images. Model compilation configures the model with specified loss functions, optimizers, and metrics. Training with specified callbacks utilizes techniques such as early stopping and learning rate scheduling to optimize training. Finally, evaluation metrics measure the model's performance on validation and test datasets to ensure robust evaluation. This schematic provides a clear view of the experimental setup and the steps involved in training the model, ensuring a thorough and adequate training process.

MosQNet-SA exemplifies the integration of advanced deep learning techniques, combining residual connections, Inception-like blocks, MBConv blocks, and spatial attention mechanisms. This robust architecture enables efficient hierarchical feature extraction and superior classification performance across diverse datasets. Through its innovative design, MosQNet-SA stands out as a powerful tool for image classification tasks, pushing the boundaries of what is possible with convolutional neural networks.

4.6 Evaluation Metrics

4.6.1 Test Loss

Test Loss quantifies the error between predicted values $\hat{y}_i$ and actual labels y_i during the testing phase. It calculates how well the model's predictions match the true labels on unseen data. The goal is to minimize test loss, as lower values indicate that the model's predictions are closer to the actual labels. Test loss is typically computed using a suitable loss function L:

$$\text{Test Loss} = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{y}_i)$$

where N is the number of samples in the test set, y_i represents the true label of the i-th sample, and $\hat{y}_i$ represents the predicted value by the model.

4.6.2 Test Accuracy

Test Accuracy measures the proportion of correctly classified samples out of the total number of samples in the test set. It is a fundamental metric in evaluating classification models, clearly indicating overall correctness. The accuracy score is calculated by dividing the number of correctly predicted samples by the total number

of samples:

$$\text{Test Accuracy} = \frac{\text{Number of correctly classified samples}}{\text{Total number of samples}}$$

4.6.3 Macro Precision

Macro Precision calculates the average precision across all classes C without considering class imbalance. Precision for each class i is the ratio of true positives TP_i to the sum of true positives and false positives FP_i:

$$\text{Macro Precision} = \frac{1}{C} \sum_{i=1}^{C} \frac{TP_i}{TP_i + FP_i}$$

Here, TP_i is the number of true positives for class i and FP_i is the number of false positives for class i.

4.6.4 Macro Recall (Sensitivity)

Macro Recall computes the average recall across all classes without considering class imbalance. Recall for each class i is the ratio of true positives TP_i to the sum of true positives and false negatives FN_i:

$$\text{Macro Recall} = \frac{1}{C} \sum_{i=1}^{C} \frac{TP_i}{TP_i + FN_i}$$

Here, FN_i is the number of false negatives for class i.

4.6.5 F1-Score

F1-Score is the harmonic mean of precision and recall, providing a single metric that balances both measures. It is particularly useful in applications where achieving a balance between precision and recall is important. F1-Score penalizes models with imbalanced precision and recall scores, emphasizing both precision and recall equally:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

4.6.6 Weighted Precision

Weighted Precision computes the average precision of each class weighted by its support (the number of true instances N_i). It provides a balanced view of preci-

sion across classes with different sample sizes. Weighted precision is beneficial when evaluating models on datasets where class frequencies vary significantly:

$$\text{Weighted Precision} = \sum_{i=1}^{C} \frac{TP_i}{TP_i + FP_i} \cdot \frac{N_i}{N}$$

Here, N_i is the number of true instances of class i, and N is the total number of samples.

4.6.7 Weighted Recall (Sensitivity)

Weighted Recall computes the average recall of each class weighted by its support (the number of true instances N_i). Similar to weighted precision, it offers a balanced view of recall across classes with different sample sizes:

$$\text{Weighted Recall} = \sum_{i=1}^{C} \frac{TP_i}{TP_i + FN_i} \cdot \frac{N_i}{N}$$

4.6.8 Confusion Matrix

A Confusion Matrix provides a detailed breakdown of the model's performance by showing the number of true and false predictions for each class. It allows for a comprehensive analysis of the model's errors, such as misclassifications and confusion between classes. Each cell in the matrix represents the count of instances for which the actual class is one row, and the predicted class is one column.

CHAPTER V

Evaluation of MosQNet-SA's Performance

5.1 Comparative Analysis of Model Architectures

To thoroughly assess the performance of (MosQNet-SA), we conducted a comprehensive comparison against 16 established neural network architectures, presenting the results in Table 5.1.

Model	**Total Params**	**Test Accuracy**	**Test Loss**
ResNet152	66,794,627	90.49%	0.3
InceptionResNetV2	55,944,675	98.56%	0.053
ResNet101	51,081,859	89.34%	0.3
ResNet50	32,011,395	91.64%	0.2
Xception	29,285,163	93.37%	0.22
DenseNet201	26,221,379	99.14%	0.0349
InceptionV3	23,935,011	94.81%	0.23
DenseNet169	19,493,699	97.98%	0.0589
VGG16	16,846,915	93.37%	0.2
EfficientNetB2	13,570,812	97.69%	0.0732
EfficientNetB1	11,853,194	97.41%	0.0838
DenseNet121	11,266,883	97.69%	0.0845
EfficientNetB0	9,327,526	97.41%	0.077
NASNetMobile	8,630,167	93.95%	0.2037
MobileNetV2	7,535,939	97.41%	0.1191
MobileNet	7,458,243	97.12%	0.15
MosQNet-SA	**388,349**	**99.42%**	**0.0401**

Table 5.1: Comparison of neural network architectures

This table highlights key metrics such as total parameters, test accuracy, and loss. Notably, MosQNet-SA stands out with a meager parameter count of 388,349 yet achieves the highest test accuracy of 99.42% among all models considered.

Model	Precision	Recall	F1-Score
VGG16	0.936951	0.938139	0.937295
ResNet50	0.921901	0.923452	0.922023
ResNet101	0.892112	0.89203	0.889369
ResNet152	0.897344	0.898933	0.897333
Xception	0.93621	0.93806	0.936289
InceptionV3	0.935487	0.937088	0.935876
InceptionResNetV2	0.974214	0.975416	0.9745
MobileNet	0.968543	0.96989	0.968969
MobileNetV2	0.950585	0.951688	0.950859
DenseNet121	0.967392	0.968788	0.967741
DenseNet169	0.9659	0.967075	0.966076
DenseNet201	0.982501	0.982063	0.982213
NASNetMobile	0.925714	0.925872	0.925171
EfficientNetB0	0.971142	0.97234	0.971439
EfficientNetB1	0.976789	0.977755	0.977165
EfficientNetB2	0.971758	0.972963	0.972163
MosQNet-SA	**0.987925**	**0.988189**	**0.988033**

Table 5.2: Additional performance metrics of neural network architectures

For further insights into performance, Table 5.1 provides additional metrics such as precision, recall, and F1-score. These metrics highlight MosQNet-SA's exceptional performance with precision of 0.987925, recall of 0.988189, and F1-score of 0.988033, underscoring its efficacy despite its compact architecture compared to larger models.

MosQNet-SA stands out as a paradigm of efficiency in neural network design, boasting a mere 388,349 parameters—a stark contrast to its larger counterparts, as seen from the table 5.1. For instance, MobileNet, the next most miniature model, employs 7,458,243 parameters, making MosQNet-SA nearly 19 times more parameter-efficient. Even more substantial models like ResNet152 escalate parameter counts to 66,794,627, dwarfing MosQNet-SA by 172. This reduction in parameter complexity is not just a numerical feat but holds profound implications for practical deployment in resource-constrained environments. Despite its compact size, MosQNet-SA excels in performance metrics, achieving a remarkable 99.42% test accuracy along with high precision, recall, and F1-score, surpassing even larger and more parameter-intensive models like DenseNet201 and InceptionResNetV2. These findings underscore MosQNet-SA's efficacy in balancing model size with performance, setting a new benchmark for efficient neural network architectures.

5.2 Comparative Insights and Implications

5.2.1 Comparative Insights

5.2.1.1 ResNet Variants

The performance of ResNet models (ResNet50, ResNet101, ResNet152) shows diminishing returns with increased depth. ResNet152, despite having the highest parameter count, achieves lower accuracy (90.49%) compared to ResNet50 (91.64%). This observation suggests that adding more layers does not guarantee improved performance and may lead to degradation.

5.2.1.2 DenseNet Performance

DenseNet201 achieves the second-highest accuracy (99.14%) and firm performance across other metrics, suggesting its dense connectivity pattern is highly effective. However, it requires significantly more parameters (26,221,379) than MosQNet-SA, highlighting the efficiency gap between the two architectures.

5.2.1.3 EfficientNet Series

EfficientNet models (B0, B1, B2) show consistent performance improvements with increased model size, validating their compound scaling approach. Nevertheless, they are outperformed by MosQNet-SA in both efficiency and accuracy, indicating that there may be limitations to the effectiveness of uniform scaling strategies.

5.2.1.4 Mobile-oriented Models

MobileNet and MobileNetV2, designed for efficiency, demonstrate strong performance considering their focus. However, MosQNet-SA surpasses them in both parameter count and all performance metrics, suggesting that MosQNet-SA's architecture may offer a superior approach to designing efficient models.

5.2.2 Implications for Model Design

The success of MosQNet-SA has several important implications for neural network design:

1. **Architectural Innovation**: MosQNet-SA's performance suggests that significant improvements can be achieved through novel architectural designs rather than simply increasing model size.

2. **Efficiency-First Approach**: The results demonstrate the potential of prioritizing efficiency in the initial design phase rather than focusing on performance and then attempting to compress or optimize the model.

3. **Rethinking Depth and Width**: MosQNet-SA's success with a relatively small parameter count challenges the notion that deep or wide networks are necessary for state-of-the-art performance.

4. **Attention Mechanisms**: The effectiveness of MosQNet-SA may be partly attributed to its use of attention mechanisms, suggesting that these components can be leveraged to create highly efficient architectures.

5.2.3 Potential Applications

The exceptional efficiency-to-performance ratio of MosQNet-SA opens new possibilities for deploying advanced AI in resource-constrained environments, including:

- Edge computing and IoT devices
- Mobile applications
- Real-time processing systems
- Scenarios with limited bandwidth for model deployment or updates

5.3 Detailed Performance Analysis

5.3.1 Confusion Matrix

To provide a more granular view of MosQNet-SA's performance, we analyzed its confusion matrix:

Figure 5.1 offers valuable insights into the model's classification performance across the three classes:

- AEDES: 941 correct predictions, with 19 misclassifications as CULEX
- ANOPHELES: 847 correct predictions, with only one misclassification as CULEX
- CULEX: 954 correct predictions, with six misclassifications as AEDES and eight as ANOPHELES

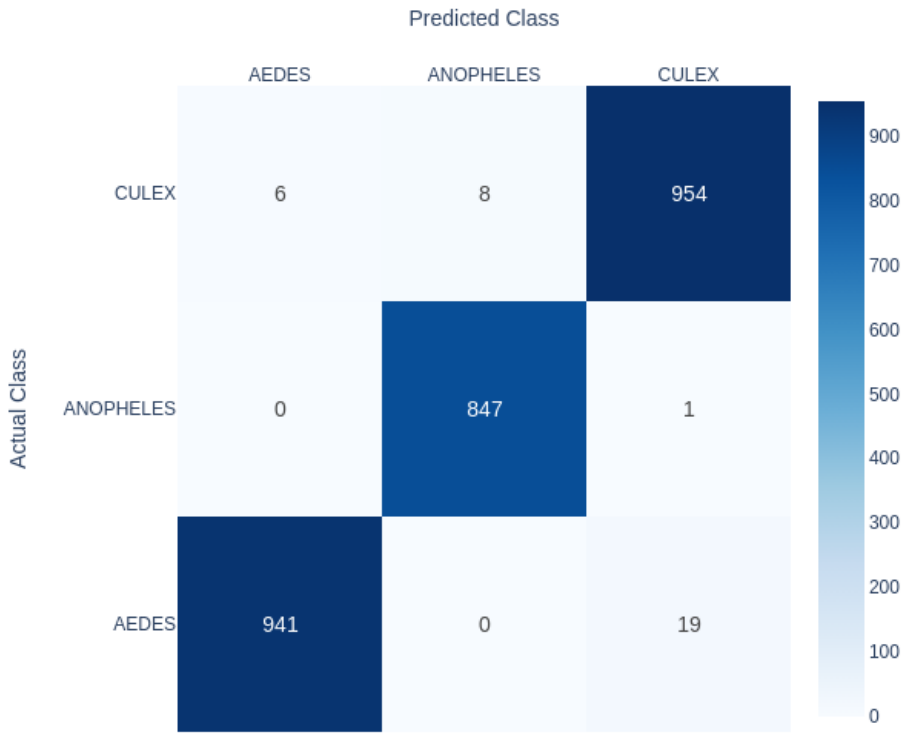

Figure 5.1: Confusion Matrix

The confusion matrix demonstrates MosQNet-SA's overall solid performance, with exceptionally high accuracy in ANOPHELES. Most misclassifications occur between AEDES and CULEX, suggesting a potential area for further investigation and improvement.

5.3.2 Training Accuracy and Loss Metrics

Figure 5.2: Training Accuracy and Loss Metrics

The training and validation loss curves demonstrated in figure 5.2 MosQNet-SA's learning progression over the 81 epochs. The model shows a steady decrease in training and validation loss, indicating effective learning without significant overfitting. The convergence of these curves towards the later epochs suggests that the model has reached a stable performance state.

5.3.3 ROC Curves per Class

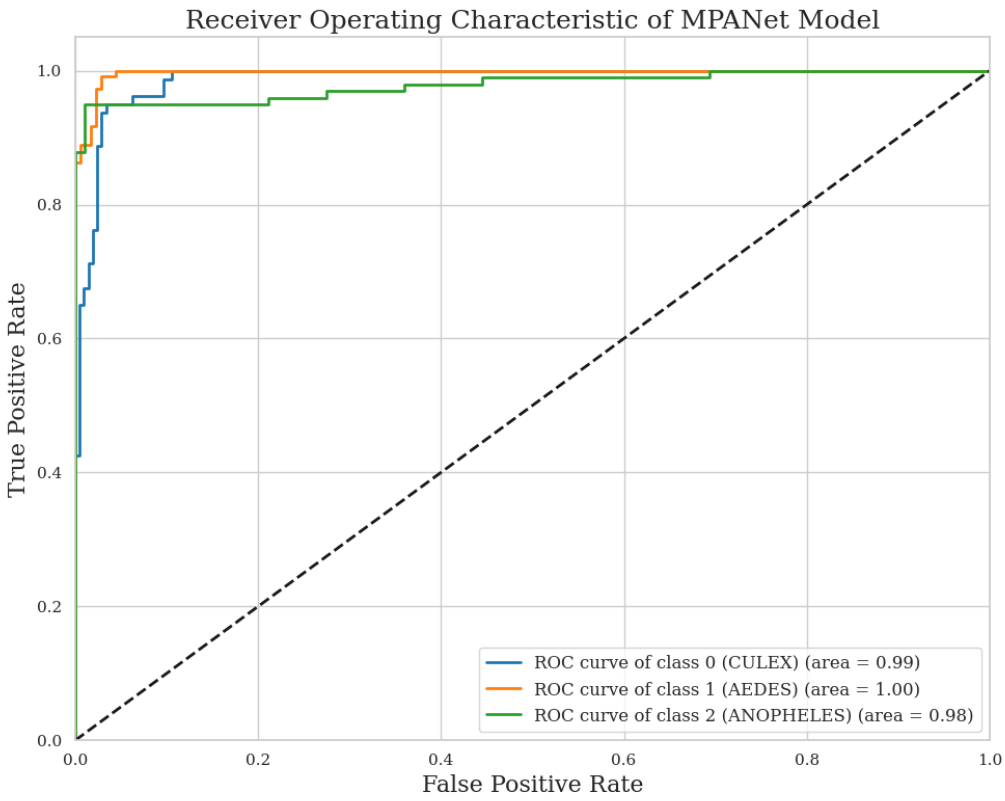

Figure 5.3: ROC curve for each class

The Receiver Operating Characteristic (ROC) curves for each class in figure 5.3 provide evidence of MosQNet-SA's strong performance. All three classes show high Area Under the Curve (AUC) values, with AEDES demonstrating near-perfect classification. The ROC curves for ANOPLELES and CULEX show slightly lower but still excellent performance, aligning with the observations from the confusion matrix.

CHAPTER VI

Explainability of The Novel Model

6.1 Introduction

In this section, we delve into the explainability of our proposed model. To evaluate its performance, we have tested the model on images sourced randomly to observe how it handles previously unseen data. We aim to understand the model's behavior and responses to new inputs.

To achieve this, we employed a variety of explainability techniques, including:

- **Saliency**
- **GradientInput**
- **Integrated Gradients**
- **SmoothGrad**
- **VarGrad**
- **SquareGrad**
- **GradCAM**
- **Occlusion**
- **Rise**
- **Guided Backpropagation**
- **LIME (Local Interpretable Model-agnostic Explanations)**
- **Kernel SHAP (SHapley Additive exPlanations)**

- **Sobol Attribution Method**

Each of these techniques is described in detail below. By applying these methods, we aim to comprehensively analyze our model's interpretability, ensuring that its decision-making process can be understood and trusted.

6.2 Data Loading and Preprocessing

This section describes loading and preprocessing the images used to evaluate our model. The dataset is organized in a directory structure, where we focus specifically on pictures within the "test" folder. The three classes of mosquitoes considered are *AEDES*, *ANOPHELES*, and *CULEX*.

To begin, we defined a preprocessing function `central_crop_and_resize` that extracts the most significant possible square from an image and resizes it to a specified size (128x128 pixels). This ensures consistency in input dimensions, which is crucial for model performance.

We then set up the directory paths and initialized lists to store the image data (`X`) and their corresponding labels (`Y`). A label mapping was created to convert class names to numeric indices.

Figure 6.1: The test images for Explainability of the model

.

We randomly selected one image for each class within the "test" category, ensuring diversity in our test samples. Each chosen image was read and converted from BGR (default for OpenCV) to RGB format. The `central_crop_and_resize` function was applied to each image, and the resulting preprocessed image was appended to the list `X`, while the corresponding label was added to the list `Y`.

After preprocessing, the lists `X` and `Y` were converted to numpy arrays, with the images normalized by dividing pixel values by 255.0 to scale them between 0 and 1.

Finally, we plotted the preprocessed images and their labels using Matplotlib to visualize them. This visualization step helps verify that the pictures have been correctly preprocessed and labeled before they are fed into the model for evaluation. The resulting figure was saved as "the mosquitoes.png" and displayed to ensure everything was in order.

6.3 Various Explainability Techniques

6.3.1 Saliency

Saliency methods aim to identify which pixels or input features significantly influence a model's output. These methods typically compute the gradient of the model's production concerning the input features. By analyzing these gradients, saliency methods can highlight the input's most critical regions or features that affect the model's decision.

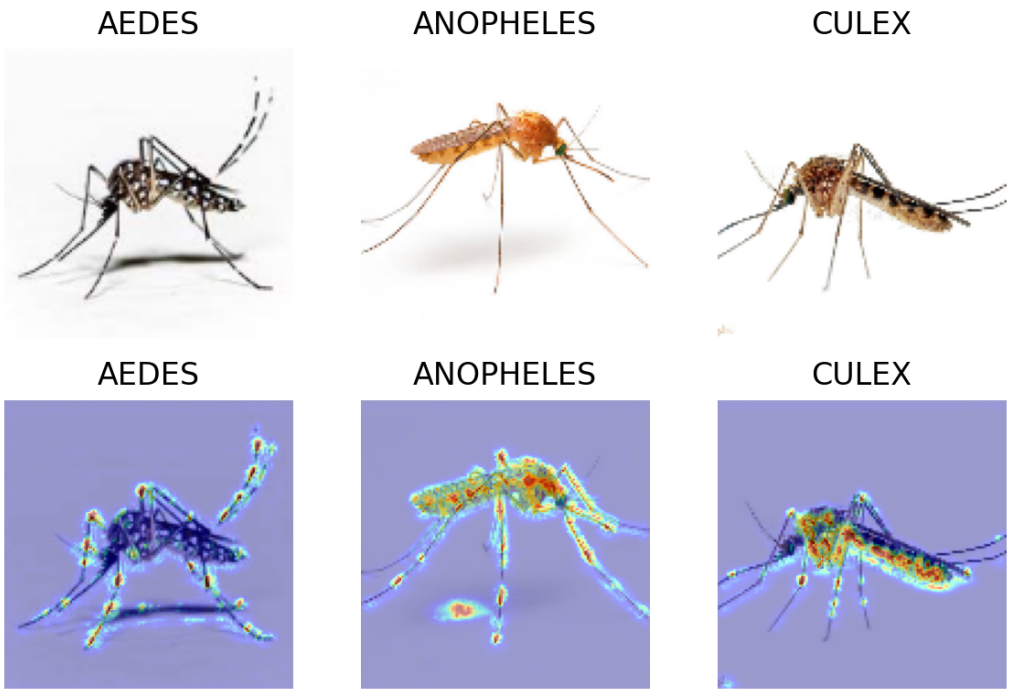

Figure 6.2: The saliency of the model

.

6.3.2 GradCAM

Gradient-weighted Class Activation Mapping (GradCAM) is designed explicitly for convolutional neural networks (CNNs) in computer vision tasks. GradCAM uses the gradients of the target output concerning the final convolutional layer to create a coarse localization map. This map highlights the regions in the input image most influential in the model's decision-making process.

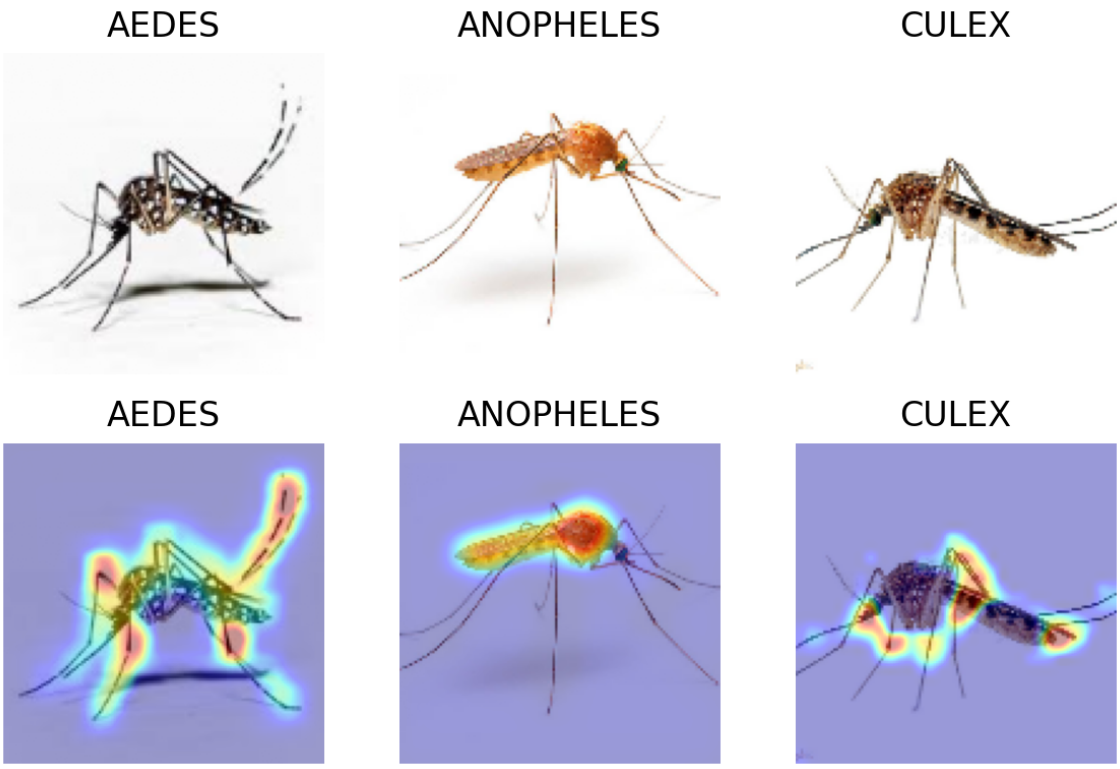

Figure 6.3: The saliency of the model

.

6.3.3 GradientInput

GradientInput computes the gradient of the target output concerning the input features. This gradient is used as the attribution score for each feature. The idea is that the magnitude of the gradient indicates how sensitive the output is to changes in the input feature, thereby highlighting its importance.

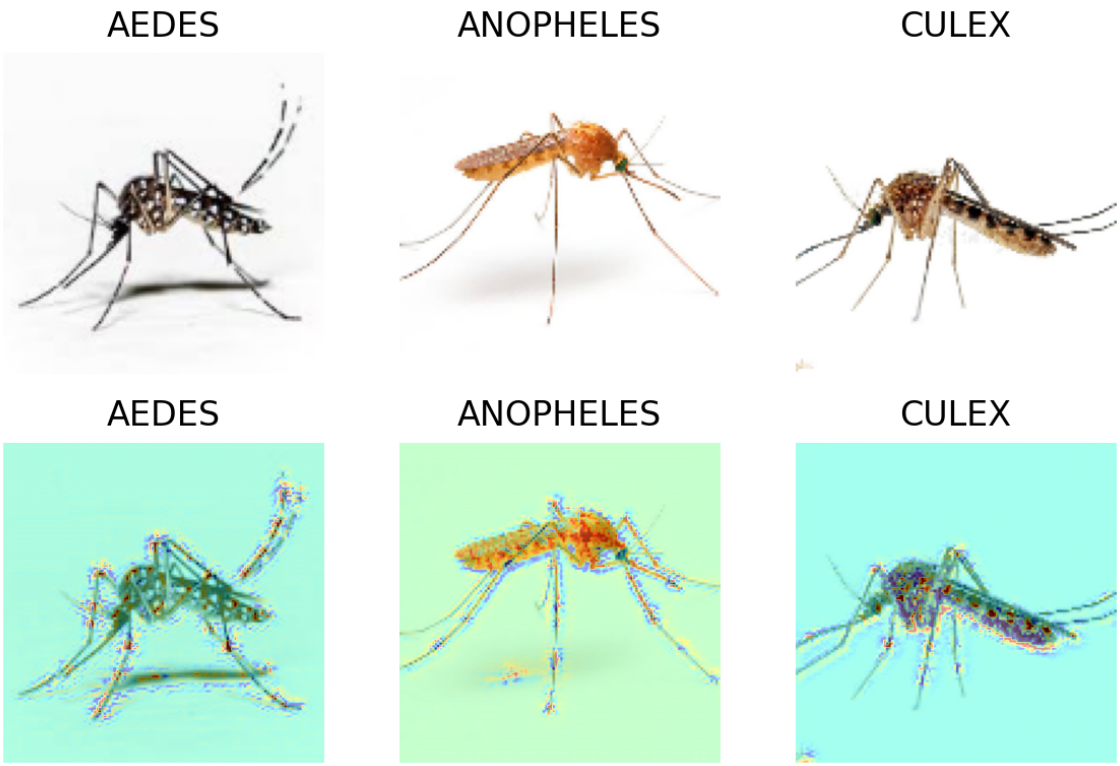

Figure 6.4: Gradiant Input

6.3.4 IntegratedGradients

Integrated Gradients address some of the shortcomings of gradient-based methods by considering the integral of the gradients along a straight path from a baseline input to the actual input. This method provides a more accurate attribution by summing up the gradients at different points along this path, effectively averaging the contributions over a range of inputs.

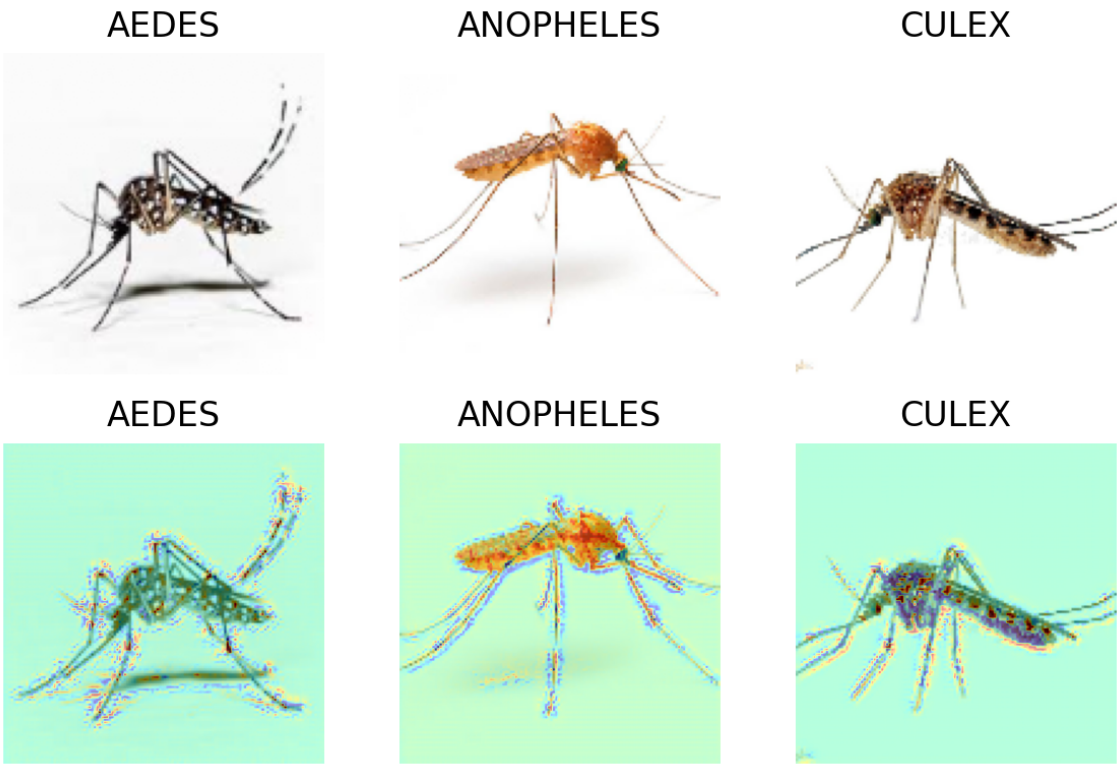

Figure 6.5: Integrated Gradients

6.3.5 SmoothGrad

SmoothGrad enhances the basic saliency map by reducing noise. It takes multiple slightly perturbed input versions and averaging the gradients. This method helps to produce more precise and more stable attribution maps, making it easier to interpret the essential features.

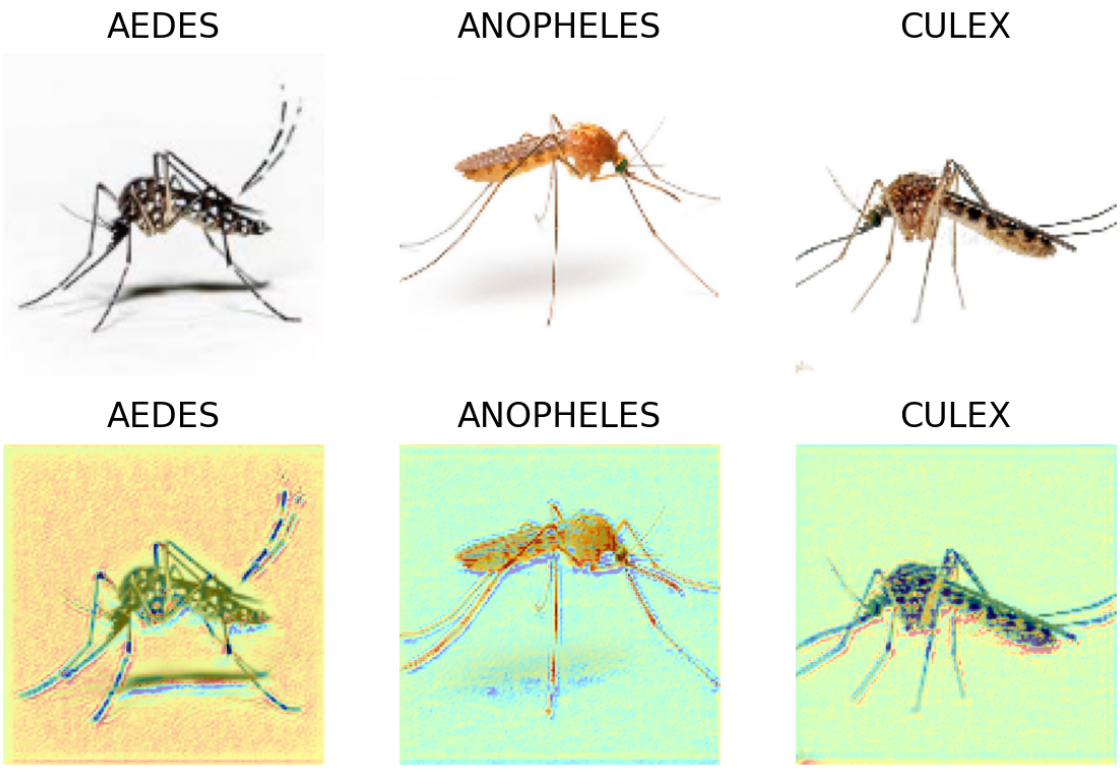

Figure 6.6: Smooth Gradient

6.3.6 VarGrad

VarGrad is another variant of the saliency method that focuses on the variance of the gradients across multiple input samples. By considering the variability of gradients, VarGrad can highlight the features that consistently show high importance, thereby providing a robust measure of feature significance.

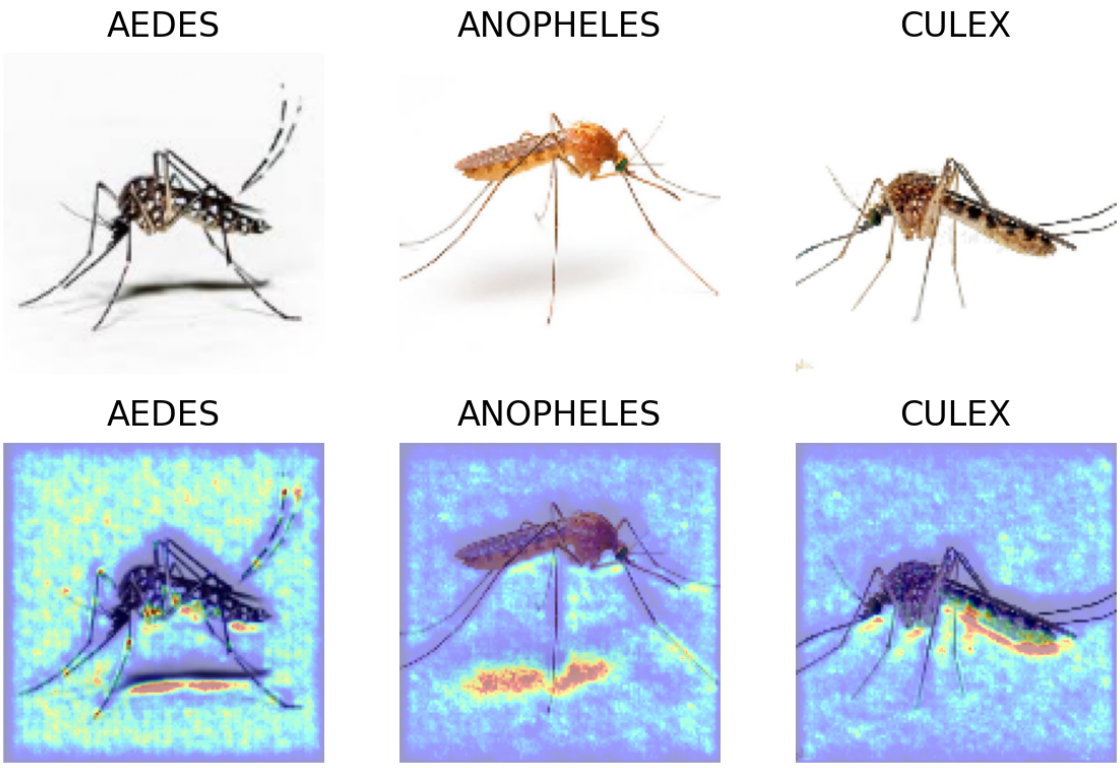

Figure 6.7: Variance of Gradients

6.3.7 SquareGrad

SquareGrad modifies the basic saliency approach by squaring the gradients before computing the attribution scores. Squaring the gradients can amplify the differences in importance among features, making distinguishing between essential and less important features easier.

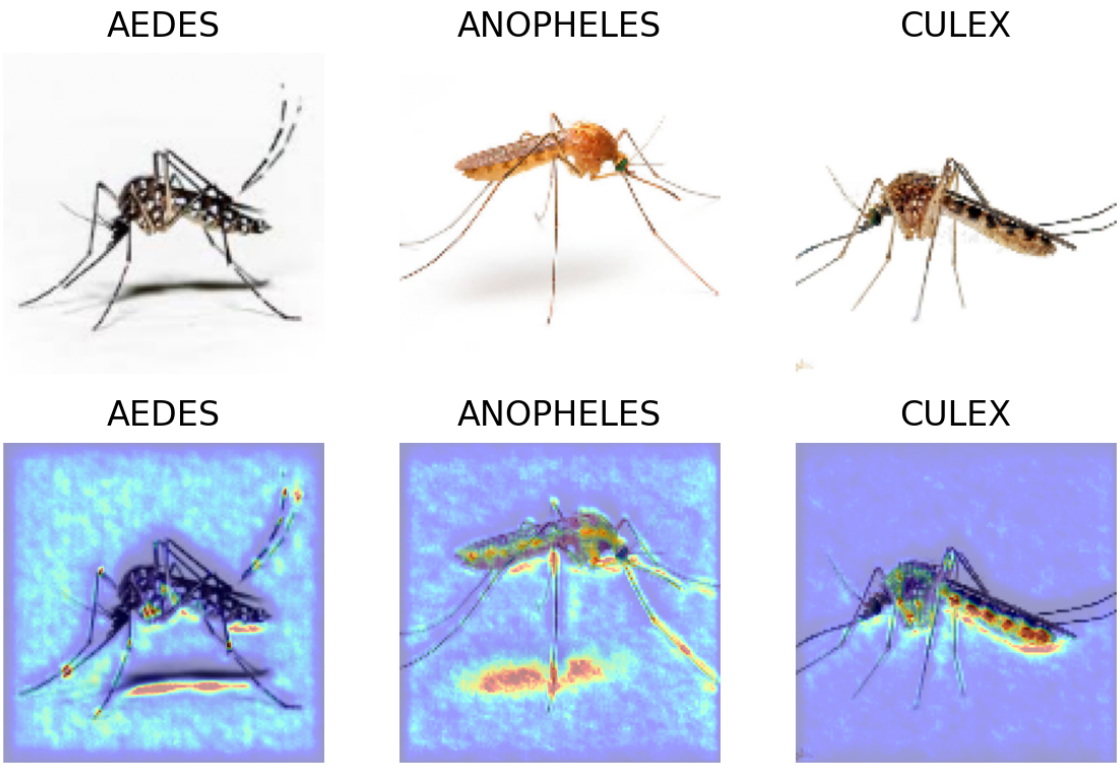

Figure 6.8: Square Gradient

6.3.8 Occlusion

Occlusion is a perturbation-based method where different input regions are systematically occluded (hidden), and the change in the model's output is measured. The areas that cause the most significant change of the production when occluded are considered the most important, providing a straightforward way to understand feature importance.

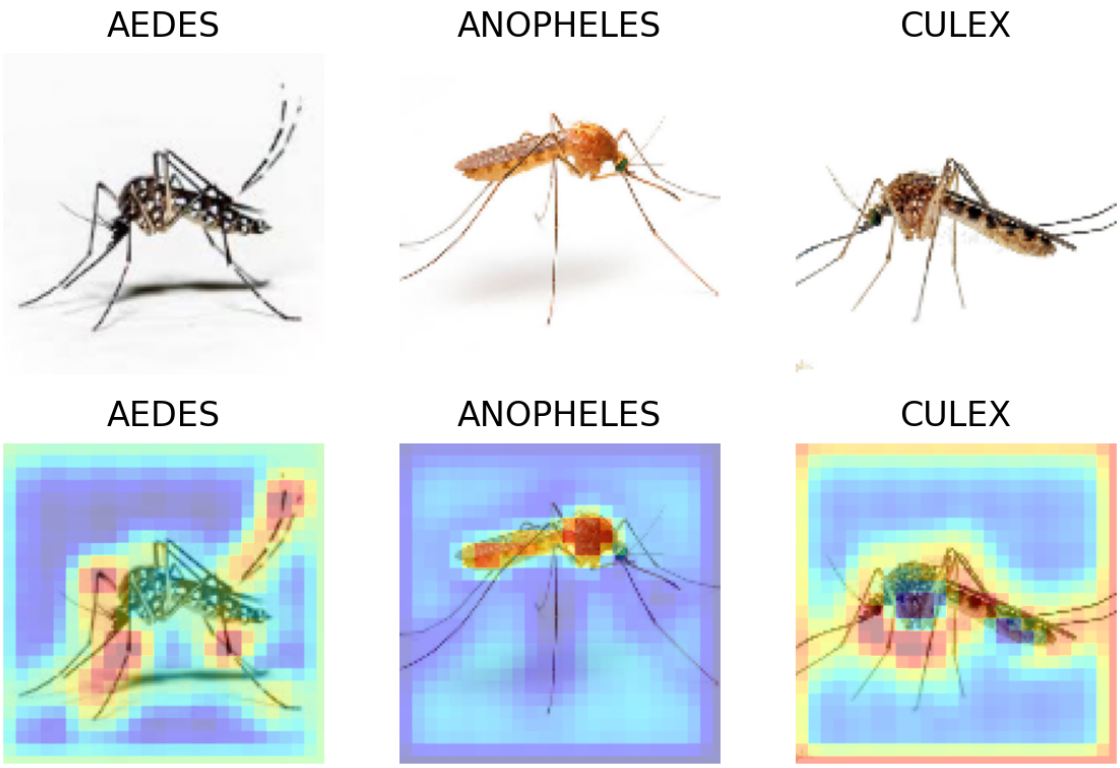

Figure 6.9: Occlusion

6.3.9 Rise

Randomized Input Sampling for Explanation (Rise) is another perturbation-based method that generates random masks and applies them to the input. By measuring the model's output change with each mask, Rise calculates the correlation between the masks and the output changes to determine the attribution scores. This method leverages randomness to provide a more comprehensive view of feature importance.

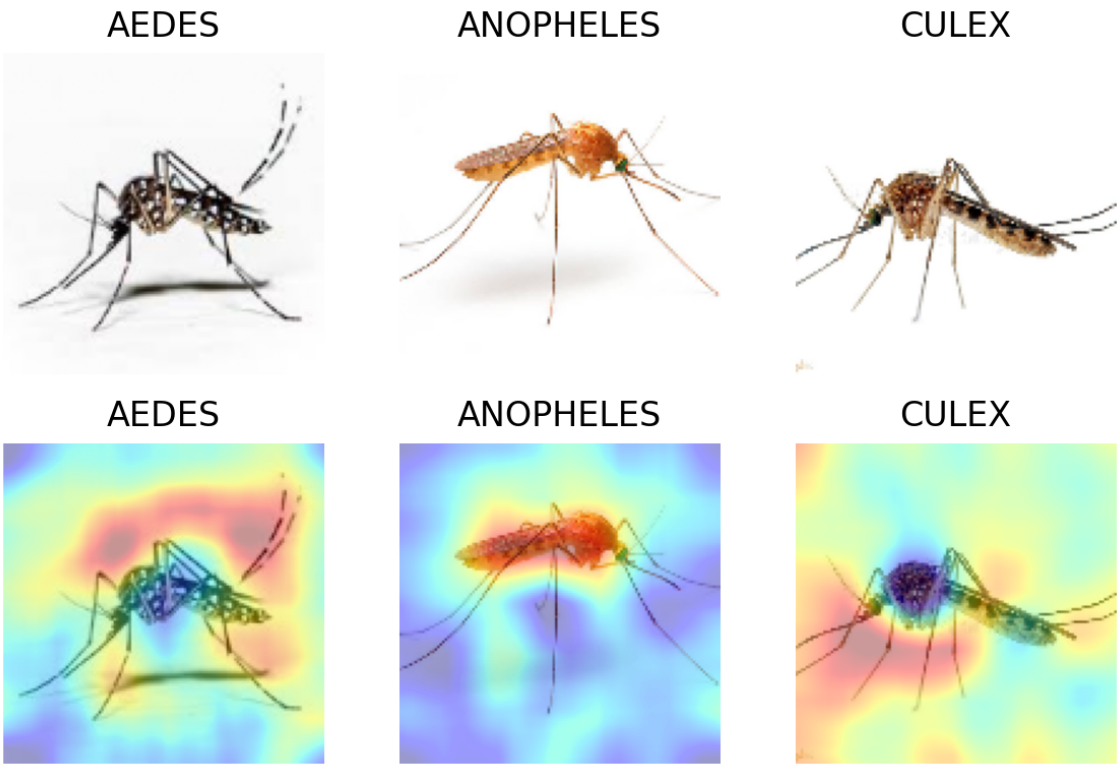

Figure 6.10: Rise

6.3.10 SobolAttributionMethod

The Sobol Attribution Method is a variant of the Shapley Value approach that uses Sobol indices from sensitivity analysis. Sobol indices measure the contribution of each input feature to the model's output, considering the interaction effects among features. This method provides a comprehensive attribution by quantifying the sensitivity of the production to changes in each input feature, making it helpful in understanding complex models.

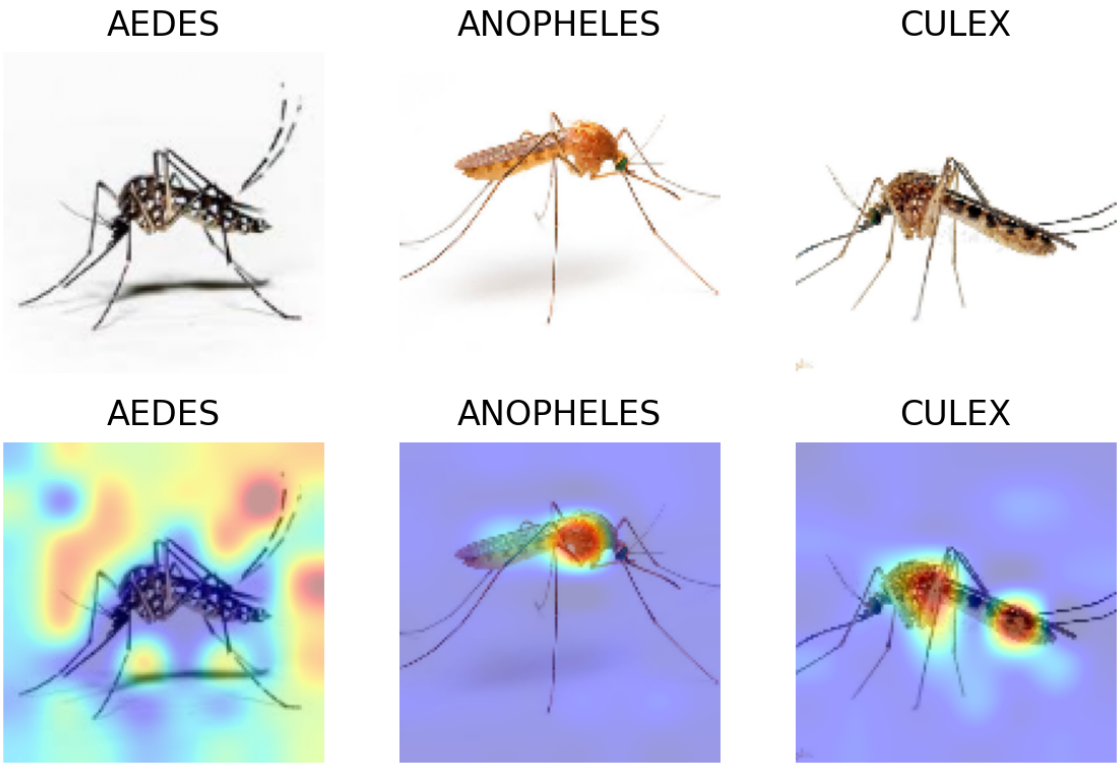

Figure 6.11: Sobol Attribute Method

6.3.11 GuidedBackprop

Guided Backpropagation is a variant of the saliency method that modifies the backpropagation process only to propagate positive gradients. This modification helps produce sharper and more visually interpretable saliency maps by focusing on features that positively influence the output, reducing noise and highlighting important regions more clearly.

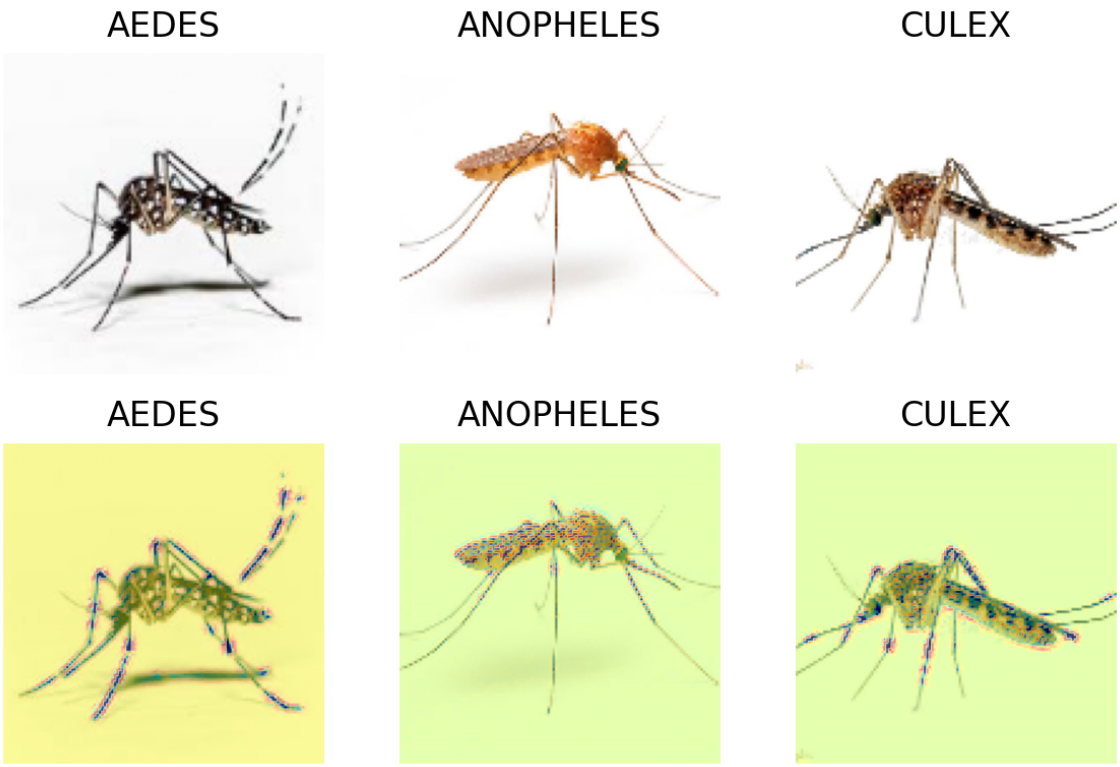

Figure 6.12: Guided Back Propagation

6.3.12 Lime

Local Interpretable Model-agnostic Explanations (LIME) is a model agnostic method that approximates the behavior of the original model locally around the input of interest. LIME uses an interpretable surrogate model, such as a linear model, to approximate the predictions of the complex model. The attribution scores are derived from the surrogate model, providing insights into the essential features in a locally interpretable manner.

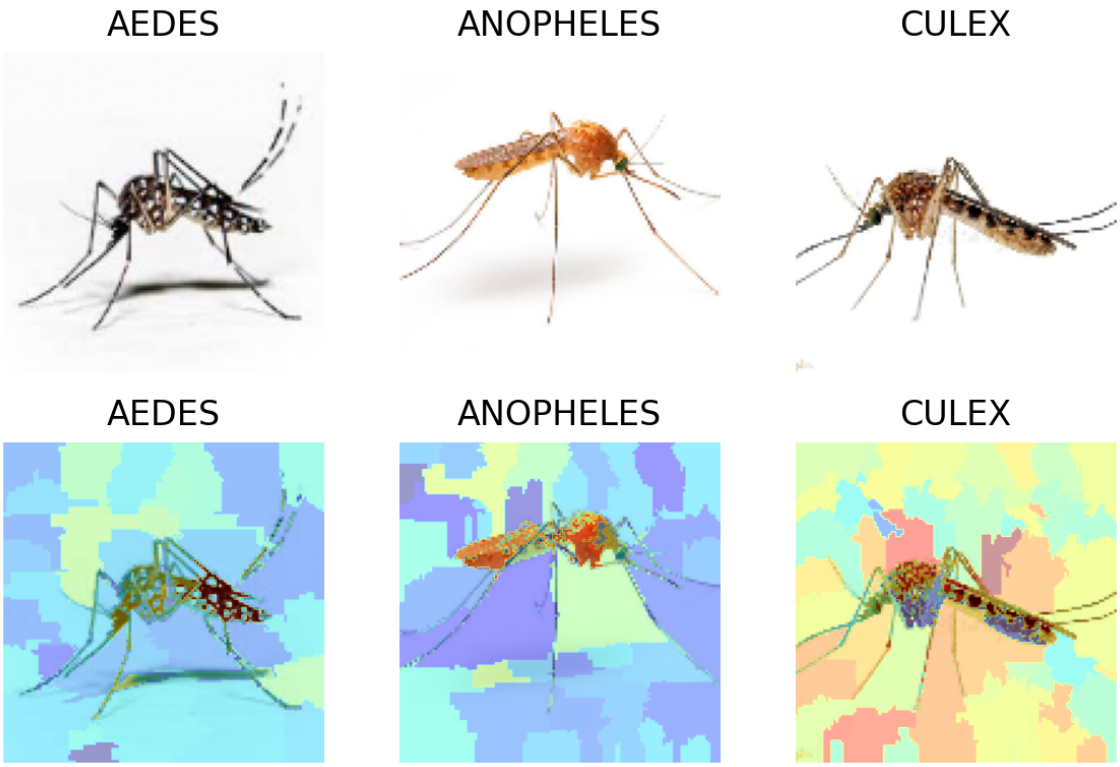

Figure 6.13: LIME

6.3.13 KernelShap

KernelShap is an implementation of the SHapley Additive exPlanations (SHAP) method. It attributes the output of a model to its input features based on game-theoretic concepts of fairness and coalitional contributions. KernelShap calculates the Shapley values, representing each feature's average marginal contribution to the output across all possible feature combinations.

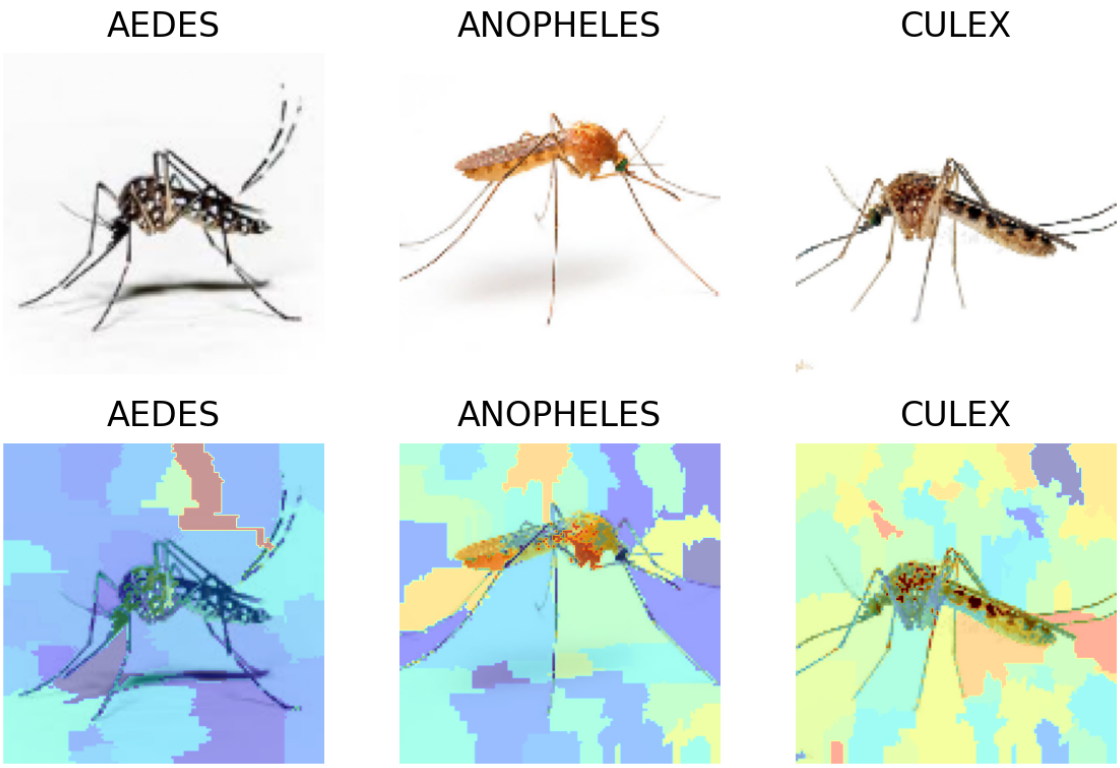

Figure 6.14: Kernel Shap

CHAPTER VII

Potential Application of MosQNet-SA in a RESTful API

The exceptional performance and efficiency of MosQNet-SA, our novel mosquito classifier, opens up numerous possibilities for real-world applications. One particularly promising avenue is the integration of MosQNet-SA into a RESTful API, which could significantly enhance mosquito identification and monitoring efforts across various fields. This section explores the potential implementation and benefits of deploying MosQNet-SA through a RESTful API, focusing on dengue and malaria risk assessment.

7.1 Overview of RESTful API Integration

A RESTful (Representational State Transfer) API provides a scalable, stateless, and standardized approach to exposing MosQNet-SA's capabilities over the internet. This integration would allow researchers, health organizations, and even citizen scientists to remotely leverage MosQNet-SA's mosquito classification abilities without the local installation of complex deep learning frameworks. The API bridges the sophisticated machine learning model and end-users, democratizing access to advanced mosquito identification technology.

7.2 Proposed API Structure and User Flow

The proposed RESTful API for MosQNet-SA would include the following key components and user flow:

1. **Landing Page:** Users are presented with a landing page featuring a "Check

Your Area" button. This intuitive interface is the entry point for casual users and professionals seeking to assess local mosquito-borne disease risks.

2. **Data Collection Form:** Upon clicking the button, users are directed to a form collecting the following information:

   - Image of the mosquito
   - IP address (automatically collected)
   - Name of the Upazila (sub-district)
   - Name of the District
   - Geological data (time of year, temperature, weather)
   - Optional: Recent dengue or malaria occurrence data

   This comprehensive data collection allows for a more accurate risk assessment by considering biological and environmental factors.

3. **Data Processing:**

   - The backend validates the IP address using an IP address API to extract and verify location data.
   - The image is saved to the database and then processed by MosQNet-SA for species identification.
   - Location data, prediction results, and datetime information are saved to the database.

   This multi-step process ensures data integrity and prepares the information for analysis.

4. **Risk Assessment:** The API performs computations based on:

   - Datetime ranking
   - Frequency of mosquito species in the Upazila and District
   - Historical disease occurrence data

   The risk assessment provides a contextualized evaluation of disease potential by considering temporal and spatial factors.

5. **Verdict Generation:** Based on these calculations, the API generates a verdict on the risk level for dengue and malaria in the user's current location.

6. **User Notification:** The verdict is sent back to the user, providing information on the potential risk of dengue and malaria in their area. This final step empowers users with actionable insights, potentially influencing personal protection measures and public health responses.

7.3 Key API Endpoints and Operational Flow

The API's operation can be broken down into several key steps and endpoints:

1. **API Initialization:**

   - The API server is set up using a framework like Flask or FastAPI.
   - Database connections are established.
   - The MosQNet-SA model is loaded into memory.

   This initialization process ensures the API is ready to handle incoming requests efficiently.

2. **/submit-data (POST):**

   - Receives form data, including the mosquito image and location information.
   - Validates incoming data for completeness and format.
   - Calls an external IP geolocation service to verify and enrich location data.
   - Compares this with user-provided location information.

   The validation step is crucial for maintaining data quality and preventing erroneous assessments.

3. **/process-image (Internal):**

   - Saves the mosquito image to the database.
   - Retrieves and preprocesses the image for MosQNet-SA.
   - Utilizes MosQNet-SA to classify the mosquito species.
   - Stores the classification result in the database.

   This step leverages the power of MosQNet-SA, translating its advanced classification capabilities into practical use.

4. **/calculate-risk (Internal):**

   - Retrieves relevant historical data from the database.
   - Calculates risk factors based on:
     - Identified mosquito species
     - Species frequency in the area
     - Seasonal factors
     - Historical disease occurrence data
   - Runs the risk calculation algorithm.

   The risk calculation incorporates multiple data points to provide a nuanced assessment of disease potential.

5. **/get-verdict (GET):**

   - Generates a verdict on dengue and malaria risk based on the risk calculation.
   - Formats the verdict into a user-friendly response.
   - Sends the verdict back to the user's browser.

   The verdict is presented in clear, understandable terms to ensure it's accessible to users of varying backgrounds.

The API implements error handling for issues such as invalid data submissions, failed image processing, or database connection problems throughout this process. It also maintains logs for monitoring and debugging purposes. This robust error handling and logging system ensures the API's reliability and facilitates continuous improvement.

7.4 Benefits of this API-based Approach

1. **Real-time Risk Assessment:** Users can quickly assess the potential risk of dengue and malaria in their area based on current mosquito populations and environmental factors. This timely information can be crucial for individual protection and community-wide preventive measures.

2. **Data Collection for Research:** The API facilitates the collection of valuable data on mosquito species distribution and potential disease hotspots. Over time, this crowdsourced data can contribute significantly to epidemiological research and predictive modeling.

3. **Public Health Monitoring:** Health organizations can use the aggregated data to monitor trends and allocate resources more effectively. The API could become essential in early warning systems for potential outbreaks.

4. **Education and Awareness:** The system can help raise public awareness about mosquito-borne diseases and their vectors. Engaging users directly in the identification process promotes a deeper understanding of the link between mosquitoes and disease transmission.

5. **Scalability and Efficiency:** The API can handle multiple requests simultaneously, making it suitable for widespread use. This scalability is crucial during peak seasons or in an emerging outbreak.

7.5 Implementation Considerations

1. **Data Privacy and Security:** Robust measures must be implemented to protect user data, especially location information. This includes encryption, secure data storage, and compliance with relevant data protection regulations.

2. **Scalability:** The system should be designed to handle varying loads, particularly during peak mosquito seasons. This may involve implementing load balancing and auto-scaling features in the API infrastructure.

3. **Accuracy and Validation:** Regular updates to MosQNet-SA and the risk assessment algorithm will be crucial to maintain accuracy. This could involve periodic retraining of the model with new data and refining the risk assessment criteria based on emerging research.

4. **User Experience:** The interface should be intuitive and provide clear, actionable information to users. This may include visual representations of risk levels and customized recommendations based on the assessment results.

5. **Error Handling and Logging:** Comprehensive error handling and logging systems are essential for maintaining system reliability and facilitating debug-

ging. This allows for quick identification and resolution of issues, ensuring minimal user disruption.

7.6 Future Enhancements

1. **Integration with Weather APIs:** Automatically fetch current weather data to improve risk assessments. This real-time environmental data could significantly enhance the accuracy of risk predictions.

2. **Historical Data Analysis:** Incorporate long-term trends to provide more accurate predictions. Machine learning algorithms could be employed to identify patterns and improve forecasting capabilities.

3. **Multi-language Support:** Expand the API's accessibility to non-English speaking regions. This is crucial for global adoption and effectiveness in diverse populations.

4. **Mobile App Development:** Create dedicated mobile applications for easier access and improved user experience. This could include features like push notifications for high-risk periods or reminders for preventive measures.

5. **Integration with Health Systems:** Allow for direct reporting to local health authorities in high-risk situations. This could streamline the process of initiating targeted interventions in potential hotspots.

6. **Real-time Analytics Dashboard:** Develop a dashboard for health officials to monitor trends and hotspots in real time. This tool could be invaluable for resource allocation and planning in vector control efforts.

7. **Gamification Elements:** Introduce gamification features to encourage regular user engagement and data contribution. This could include achievements for consistent reporting or community challenges for mosquito control efforts.

8. **AI-powered Chatbot Integration:** Implement a chatbot to guide users through the risk assessment process and provide personalized advice based on the results. This could enhance user engagement and education.

Integrating MosQNet-SA into this comprehensive RESTful API significantly advances mosquito-borne disease risk assessment and public health management. By

combining efficient mosquito classification with location-based risk analysis, this system can significantly impact how we monitor and respond to dengue and malaria threats globally. The modular and scalable nature of the API allows for future expansions and improvements, ensuring its long-term value in public health efforts.

This application demonstrates the practical value of our research and showcases how deep learning models like MosQNet-SA can have immediate, real-world impacts in critical areas of public health. As we continue to refine and expand this system, we envision it becoming an indispensable tool in the global fight against mosquito-borne diseases, empowering individuals, researchers, and health authorities with timely, accurate, and actionable information.

The success of this API could pave the way for similar applications in other areas of public health and environmental monitoring, illustrating the broad potential of AI-driven solutions in addressing complex global challenges. As we move forward, continued collaboration between AI researchers, public health experts, and community stakeholders will be crucial in maximizing the impact and effectiveness of this innovative approach to disease prevention and control.

CHAPTER VIII

Conclusion and Future work

8.0.1 Conclusion

This thesis introduces (MosQNet-SA), a groundbreaking Explainable Convolutional-Attention Network that represents a significant leap forward in mosquito species classification and vector-borne disease surveillance. By ingeniously combining CNNs with attention mechanisms, MosQNet-SA achieves superior accuracy and remarkable efficiency, utilizing up to 10 times fewer parameters and delivering faster inference times compared to existing state-of-the-art models. The extensive evaluations using explainable AI techniques have solidified MosQNet-SA's reliability and trustworthiness, addressing the critical need for transparency in AI-driven health applications. Moreover, deploying MosQNet-SA as a RESTful API opens up new avenues for real-time mosquito classification and dynamic risk mapping of diseases like dengue and malaria, demonstrating its practical value in global health initiatives. This research advances machine learning in entomology and provides a powerful tool for public health officials, researchers, and communities worldwide in their fight against vector-borne diseases. As we face increasing challenges from these diseases due to climate change and globalization, MosQNet-SA stands as a testament to the potential of innovative AI solutions in addressing urgent global health issues. By bridging the gap between cutting-edge technology and practical application, this work paves the way for more effective, efficient, and responsive vector-borne disease surveillance and control strategies, ultimately contributing to improved public health outcomes on a global scale.

8.0.2 Future Work

While MosQNet-SA represents a significant advancement in mosquito classification and vector-borne disease surveillance, there are several promising avenues for

future research and development:

1. **Expanded Species Coverage:** Future work could expand MosQNet-SA's capabilities to classify a broader range of mosquito species and other disease vectors, enhancing its utility in diverse ecological contexts.

2. **Integration with IoT Devices:** Exploring the integration of MosQNet-SA with IoT devices and smart traps could enable automated, real-time mosquito surveillance in the field, providing continuous data streams for more dynamic and responsive disease control efforts.

3. **Transfer Learning for Rare Species:** Investigating transfer learning techniques could improve MosQNet-SA's performance on rare or underrepresented mosquito species, addressing the challenge of imbalanced datasets in entomological research.

4. **Multi-modal Learning:** Incorporating additional data modalities, such as environmental sensors or genetic information, into the classification process could further enhance the accuracy and contextual relevance of MosQNet-SA's predictions.

5. **Federated Learning Implementation:** Developing a federated learning framework for MosQNet-SA could enable collaborative model improvement across multiple institutions while preserving data privacy and facilitating global cooperation in vector-borne disease surveillance.

6. **Predictive Modeling:** Extending MosQNet-SA's capabilities to include predictive modeling of disease outbreaks based on mosquito population dynamics and environmental factors could provide valuable early warning systems for public health authorities.

7. **Explainable AI Advancements:** Further research into novel explainable AI techniques specific to entomological applications could enhance the interpretability of MosQNet-SA's decisions, fostering greater trust and adoption among domain experts.

8. **Mobile Application Development:** Creating user-friendly mobile applications powered by MosQNet-SA could democratize access to mosquito identification tools, enabling citizen science initiatives and enhancing community engagement in disease prevention efforts.

9. **Cross-disciplinary Collaboration:** Fostering collaborations with epidemiologists, ecologists, and public health experts could lead to more holistic, AI-driven approaches to vector-borne disease management.

These future directions aim to build upon the foundation laid by MosQNet-SA, further enhancing its impact on global health and pushing the boundaries of AI applications in entomology and disease surveillance. As technology evolves and our understanding of vector-borne diseases deepens, the potential for innovative solutions like MosQNet-SA to make a lasting difference in public health outcomes remains exciting and promising.

References

[1] J. Park, D. I. Kim, B.-H. Choi, W. Kang, and H. W. Kwon, "Classification and morphological analysis of vector mosquitoes using deep convolutional neural networks," *Scientific Reports*, vol. 10, 2020.

[2] A. Alubedy, "Mosquito detection and classification using machine learning algorithms," *Iraqi Journal of Intelligent Computing and Informatics (IJICI)*, 2023.

[3] M. Asgari, A. Sadeghzadeh, M. B. Islam, L. Rada, and J. Bozeman, "Deep learning-based vector mosquitoes classification for preventing infectious diseases transmission," *Image Analysis & Stereology*, 2022.

[4] A. A. I. Siddiqui and D. C. N. Kayte, "Convolution neural network-based mosquito classification system," *Proceedings of the 3rd International Conference on ICT for Digital, Smart, and Sustainable Development, ICIDSSD 2022, 24-25 March 2022, New Delhi, India*, 2023.

[5] K. Okayasu, K. Yoshida, M. Fuchida, and A. Nakamura, "Vision-based classification of mosquito species: Comparison of conventional and deep learning methods," *Applied Sciences*, 2019.

[6] M. A. R. Shamim, A. Anas, and M. Erfan, "Identification of vector and non-vector mosquito species using deep convolutional neural networks with ensemble model," *2022 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE)*, pp. 1–6, 2022.

[7] F. Rustam, A. A. Reshi, W. Aljedaani, A. Alhossan, A. Ishaq, S. Shafi, E. Lee, Z. Alrabiah, H. Alsuwailem, A. Ahmad, and V. Rupapara, "Vector mosquito image classification using novel rifs feature selection and machine learning models for disease epidemiology," *Saudi Journal of Biological Sciences*, vol. 29, pp. 583 – 594, 2021.

[8] R. Pise, M. Aungmaneeporn, K. Patil, and P. Chumchu, "Image dataset of aedes and culex mosquito species," 2020.

[9] M. Minakshi, P. Bharti, W. B. McClinton, J. Mirzakhalov, R. M. Carney, and S. Chellappan, "Automating the surveillance of mosquito vectors from trapped specimens using computer vision techniques," *Proceedings of the 3rd ACM SIGCAS Conference on Computing and Sustainable Societies*, 2020.

[10] D. Motta, A. Á. B. Santos, I. Winkler, B. A. S. Machado, D. A. D. I. Pereira, A. M. Cavalcanti, E. O. L. Fonseca, F. Kirchner, and R. Badaró, "Application of convolutional neural networks for classification of adult mosquitoes in the field," *PLoS ONE*, vol. 14, 2019.

[11] M. Minakshi, "A machine learning framework to classify mosquito species from smart-phone images," 2018.

[12] M. Mohommed, P. Rajakaruna, N. Kehelpannala, A. S. A. Perera, and L. Abeysiri, "Dengue carb: Mosquito identification and classification using machine learning," *2020 2nd International Conference on Advancements in Computing (ICAC)*, vol. 1, pp. 126–131, 2020.

[13] K. Trivedi and H. Shroff, "Identification of deadliest mosquitoes using wing beats sound classification on tiny embedded system using machine learning and edge impulse platform," *2021 ITU Kaleidoscope: Connecting Physical and Virtual Worlds (ITU K)*, pp. 1–6, 2021.

[14] E. Fanioudakis, M. Geismar, and I. Potamitis, "Mosquito wingbeat analysis and classification using deep learning," *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 2410–2414, 2018.

[15] J. H. Rony, N. Karim, M. A. Rouf, S. B. Noor, and F. H. Siddiqui, "Mosquito species classification through wingbeat analysis: A hybrid machine learning approach," *2023 International Conference on Next-Generation Computing, IoT and Machine Learning (NCIM)*, pp. 1–4, 2023.

[16] D. F. Silva, V. M. A. de Souza, G. E. A. P. A. Batista, E. J. Keogh, and D. P. W. Ellis, "Applying machine learning and audio analysis techniques to insect recognition in intelligent traps," *2013 12th International Conference on Machine Learning and Applications*, vol. 1, pp. 99–104, 2013.

[17] V. R. de Lima, M. C. C. de Morais, and K. Kirchgatter, "Integrating artificial intelligence and wing geometric morphometry to automate mosquito classification.," *Acta tropica*, p. 107089, 2023.

[18] A. P. Genoud, Y. Gao, G. M. Williams, and B. P. Thomas, "A comparison of supervised machine learning algorithms for mosquito identification from backscattered optical signals," *Ecol. Informatics*, vol. 58, p. 101090, 2020.

[19] S.-Q. Ong, G. Nair, U. K. Yusof, and H. B. Ahmad, "Community-based mosquito surveillance: an automatic mosquito-on-human-skin recognition system with a deep learning algorithm.," *Pest management science*, 2022.

[20] M. Fuchida, T. Pathmakumar, R. E. Mohan, N. Tan, and A. Nakamura, "Vision-based perception and classification of mosquitoes using support vector machine," *Applied Sciences*, vol. 7, p. 51, 2017.

[21] A. Joshi and C. Miller, "Review of machine learning techniques for mosquito control in urban environments," *Ecol. Informatics*, vol. 61, p. 101241, 2021.

[22] A. Rodriguez, F. Bartumeus, and R. Gavaldà, "Machine learning assists the classification of reports by citizens on disease-carrying mosquitoes," in *SoGood@ECML-PKDD*, 2016.

[23] R. Pise, K. Patil, M. Laad, and N. Pise, "Dataset of vector mosquito images," *Data in Brief*, vol. 37, p. 107322, 2021.

[24] R. Pise, M. Aungmaneeporn, K. Patil, and P. Chumchu, "Image dataset of aedes and culex mosquito species," 2020.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '16, pp. 770–778, IEEE, June 2016.

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.

[28] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[29] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," *CoRR*, vol. abs/1602.07261, 2016.

[30] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, pp. 2261–2269, IEEE Computer Society, 2017.

[31] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[32] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114, PMLR, 09–15 Jun 2019.

[33] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," *CoRR*, vol. abs/1707.07012, 2017.

[34] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.