ENSEMBLE BASED SKIN CANCER CLASSIFICATION AND IT'S REAL-TIME IMPLEMENTATION

THESIS SUBMITTED IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF

Master of Technology

in

VLSI Design and Microelectronics Technology

(Electronics and Telecommunication Engineering Department)
Jadavpur University

by

MOUMITA HAIT

Class Roll No: 002010703005

Examination Roll No: M6VLS23011

Registration No: 154107 of 2020-2021

Under the guidance of

Prof. Sheli Sinha Chaudhuri

Professor
Department of Electronics and Telecommunication Engineering
Jadavpur University
Kolkata - 700032
West Bengal, India

June, 2023

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this thesis contains previous work and original work by the undersigned candidate, as part of my Master of Technology in VLSI Design and Microelectronics Technology in the Department of Electronics and Telecommunication Engineering. All information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that I have thoroughly cited and referenced all material and findings which are not original to this research, as provided by these rules and conduct.

Name : Moumita Hait
Exam Roll No : M6VLS23011
Thesis Title : ENSEMBLE BASED SKIN CANCER CLASSIFICATION AND IT'S REAL-TIME IMPLEMENTATION

Moumita Hait.

Signature of Candidate

FACULTY OF ENGINEERING & TECHNOLOGY
JADAVPUR UNIVERSITY

CERTIFICATE

This is to certify that the thesis entitled — **"ENSEMBLE BASED SKIN CANCER CLASSIFICATION AND IT'S REAL-TIME IMPLEMENTATION"** has been carried out by **MOUMITA HAIT** bearing Class Roll No: **002010703005**, Examination Roll No.: **M6VLS23011** and Registration No: **154107** of **2020-2021**, under my guidance and supervision and be accepted in partial fulfillment of the requirement for the degree of **Master of Techolonoly** in **VLSI Design and Microelectronics Technology** in the Department of Electronics and Telecommunication Engineering.

Dr. Sheli Sinha Chaudhuri
Professor
Dept. of Electronics & Tele-Comm. Engg.
JADAVPUR UNIVERSITY
Kolkata-700 032

S.S. Chaudhuri 13/6/23

Prof. Sheli Sinha Chaudhuri
Supervisor
Department of Electronics and
Telecommunication Engineering
Jadavpur University

Manotosh Biswas 13/06/23

Prof. Manotosh Biswas
Head of the Department
Department of Electronics and
Telecommunication Engineering
Jadavpur University

MANOTOSH BISWAS
Professor and Head
Electronics and Telecommunication Engineering
Jadavpur University, Kolkata - 32

Ardhendu Ghoshal 14/06/23

Prof. Ardhendu Ghoshal
Dean
Faculty of Engineering & Technology
Jadavpur University
Kolkata – 700032

DEAN
Faculty of Engineering & Technology
JADAVPUR UNIVERSITY
KOLKATA-700 032

FACULTY OF ENGINEERING & TECHNOLOGY
JADAVPUR UNIVERSITY

CERTIFICATE OF APPROVAL

The thesis titled **"ENSEMBLE BASED SKIN CANCER CLASSIFICATION AND IT'S REAL-TIME IMPLEMENTATION"** is hereby approved as a creditworthy study of an engineering subject conducted and presented satisfactorily to warrant its acceptance as a precondition to the degree for which it was submitted. It is understood that the undersigned does not automatically support or accept any argument made, opinion expressed, or inference drawn in it by this approval, but only approves the thesis for the reason it was submitted.

Committee on Final Examination for Evaluation of the Thesis

Signature of External Examiner

Signature of Supervisor

ACKNOWLEDGEMENTS

This thesis entitled **"ENSEMBLE BASED SKIN CANCER CLASSIFICATION AND IT'S REAL-TIME IMPLEMENTATION"** is the result of the work whereby I have been accompanied and supported by many people, including my guide, my friends, and lab seniors. It is a pleasant aspect that now I can express my gratitude to all of them.

First and foremost, I would like to express my sincere gratitude to my thesis supervisor **Dr. Sheli Sinha Chaudhuri**, Professor of the Department of Electronics and Telecommunication Engineering, at Jadavpur University for her valuable guidance, insightful suggestions, and support while conducting this thesis work as well as in the writing of this thesis. I have been very fortunate to have a guide like her. Her positivity, confidence, and ideas helped me to complete my thesis work successfully and she guided me as a guardian. I am also immensely thankful to our Head of Department, **Prof. Manotosh Biswas**, for his unwavering assistance and support, as well as for granting me access to all the resources necessary for my thesis work.

I want to express my appreciation to **Mr. Asfak Ali**, a research scholar in this department, for his valuable assistance in developing, implementing, and analyzing the idea. I am also thankful to my project mates, friends, as well as the technical and non-technical staff of Jadavpur University who have provided support and guidance throughout my thesis work.

I want to express my gratitude to my parents and family also, as, without their sacrifices, I can't do anything. And their invaluable love, encouragement, and support make me, whatever I am today.

Moumita Hait.

Moumita Hait
VLSI Design & Microelectronics Technology
Electronics & Telecommunication Engineering Department
Jadavpur University
Kolkata-32, West Bengal, India

Abstract

Efficient classification of skin cancer plays a crucial role in early detection and treatment. Deep learning algorithms have emerged as a promising approach for accurate skin cancer subtype prediction by leveraging intricate patterns and correlations from large clinical datasets. This thesis proposes two novel ensemble techniques: one based on the Levy Stable probability density function (PDF) and another based on the Beta function ensemble technique. The Levy Stable model employs a robust ranking mechanism and is optimized for real-time hardware implementation. Five standard convolutional neural network (CNN) models, including DenseNet-121, EfficientNet-B3, MobileNet-V3-Large, Inception-V3, and ResNet-101, are utilized along with transfer learning to generate confidence scores that enhance classification accuracy. The ensemble technique combines these scores using a rank-based fusion approach based on the Beta function, surpassing traditional ensemble methods by adapting the priority based on classifier confidence scores for each instance. The proposed Beta ensemble technique achieves an F1-score of 83% and an Accuracy of 91%. The effectiveness of the proposed techniques is evaluated on the HAM10k dataset, demonstrating superior performance compared to existing approaches. Additionally, optimization for real-time hardware implementations is conducted using the OpenVINO toolkit, achieving high-performance inference rates on different hardware configurations. Specifically, the Levy Stable PDF achieves 110 FPS and 91 FPS for float 16 and float 32 precision, respectively, on an Intel i7 CPU, while the Beta function achieves 33.56 FPS and 26.80 FPS for float16 and float32 precision, respectively, on an Intel iRIS Xe GPU.

Keywords: Beta Function, CNN Models, Ensemble Method, HAM10K, Levy Stable PDF, OpenVINO toolkit, Skin Cancer Classification

Table of Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

Technologies are constantly evolving to simplify human life, and one of the most significant advancements in recent times is Artificial Intelligence (AI). AI is a branch of computer science that mimics human behavior and consciousness, eliminating the need for explicit programming. Within AI, Machine Learning (ML) is a subset that focuses on algorithms capable of learning from data and making predictions. Deep Learning (DL) is a further subset of ML that encompasses neural networks inspired by the human brain's functionality. Neural networks consist of interconnected neurons that transmit signals through weighted connections. DL has found applications in various domains, including Self-Driving Cars (**Sharma** [**2020**]), Medical Science (**Egger et al.** [**2022**]), Detecting Developmental Delays in Children (**Kim et al.** [**2021**]), Automatic Machine Translation (**Singh et al.** [**2017**]), Image Editing (**Kiani et al.** [**2020**]), Language Translations (**Ali et al.** [**2021**]), Automatic Game Playing (**Justesen et al.** [**2017**]), Chatbots (**Mohan and Chowdhary** [**2019**]), Robotics (**Mouha** [**2021**]), and Composing Music (**Hernandez Oliván and Beltrán Blázquez** [**2021**]). In the healthcare industry, deep learning techniques offer tremendous potential. Deep learning models have been employed to create predictive models for disease diagnosis, prognosis, and treatment recommendations. Notably, with advancements in neural network techniques, deep learning algorithms have achieved remarkable performance in visual detection tasks, often rivaling human capabilities (**Zhao et al.** [**2019**]). Neural networks have been extensively utilized in medical applications such as image categorization (**krishna et al.** [**2018**]), registration (**Chen et al.** [**2021**]), segmentation (**Patel** [**2021**]), and lesion detection (**Zafar et al.** [**2023**]). By leveraging neural network algorithms, diseases can be detected at an early stage across various anatomical systems, including the digestive system, neurological system, cardiovascular system, skeletal system, and skin.

The skin, comprising the epidermis, dermis, and hypodermis, represents the largest and heaviest organ in the human body and plays vital roles in protection, sensation, and thermoregulation. The skin's outermost layer, the stratum corneum, provides optical neutrality and shields against external aggressors. Melanocytes within the basal layer of the epidermis produce melanin, a pigment responsible for skin coloration and protection against harmful ultraviolet (UV) radiation (**Vipin et al.** [**2021**]). However, skin cancer, including both malignant melanoma and benign non-melanoma types, is a prevalent condition with detrimental effects, causing disfigurement, pain, and bleeding. Excessive exposure to UV radiation damages DNA, leading to abnormal melanocyte growth and melanin production.

With the destruction of the ozone layer and increasing environmental pollution, the incidence of skin cancer is rising. Changes in skin tone, size, shape, lesions, or mole color serve as warning signs for potential skin cancer. Malignant melanoma, although less common, is a highly lethal form of skin cancer characterized by gradual growth and changes in color. Melanoma is diagnosed based on visible changes in skin color caused by abnormal melanin distribution within the skin layers. Pale regions with a prominent blood supply and thicker collagen fibers are indicative of melanoma. Gross morphologic features such as shape, size, color, border, and symmetry are also considered in assessing pigmented lesions. A biopsy and histology examination are required to confirm the diagnosis of suspected skin cancer. Various types of melanoma, including superficial spreading melanoma (SSM), nodular melanoma (NM), lentigo malignant melanoma (LMM), and acral lentiginous melanoma (ALM), are identified based on microscopic tumor examination. skin cancer, including melanoma, poses a significant global health burden. According to the World skin cancer Research Fund International, in 2020 (**wcr** [**2020**]), there were 324,635 reported cases of melanoma skin cancer worldwide, with the highest rates observed in countries such as Australia, New Zealand, and Denmark. The number of cases varied across countries, with Germany, Australia, and the Netherlands having the highest counts. Early detection is crucial for effective skin cancer treatment (**Chhatlani et al.** [**2022**]), as it significantly improves patient outcomes. Currently, doctors rely on biopsies to diagnose skin cancer, a time-consuming and invasive procedure. To provide a rapid, accessible, and affordable diagnostic method, computer technology is being leveraged in the detection of skin cancer symptoms.

In this thesis, I explore the application of AI techniques, specifically DL and neural networks, for skin cancer classification. I propose novel ensemble techniques, including the Levy stable ensemble method and the Beta function-based ensemble technique, to improve the accuracy of skin cancer subtype prediction. These techniques integrate multiple classifiers and prioritize their decisions based on confidence scores, leading to enhanced classification accuracy. Experimental evaluation using the HAM10k dataset (**Tschandl** [**2018**]) demonstrates the superiority of my proposed approaches over existing methods. Furthermore, I optimize the models for real-time hardware implementation using the OpenVINO toolkit (**ope**), achieving high-performance inference rates on both CPUs and GPUs. The successful application of these ensemble techniques in skin cancer classification highlights their potential to revolutionize early detection and diagnosis. The future scope

of this research includes further refining the ensemble techniques, integrating advanced imaging technologies, evaluating larger and diverse datasets, conducting clinical validations, and exploring commercial product development. Additionally, the concepts and methodologies employed in this thesis can be extended to other medical applications beyond skin cancer classification, contributing to advancements in computer-aided diagnostics and improving healthcare outcomes.

Deep learning models can be deployed in multiple platforms such as CPU (Central processing unit), GPU (Graphics processing unit), VPU (vision processing unit), FPGA (Field Programmable Gate Array), etc. But the challenges are that all of them need unique inference techniques, neither a single code can be implemented through all of these platforms nor a particular size fits all the platforms. Here comes a new deep learning inferencing toolkit in the market named OpenVINO provided by Intel on May 2018. It is mainly focused on Deep learning Inferencing, not training. OpenVINO stands for Open Visual Inference and Neural Network Optimization. It is a free toolkit developed by Intel to optimize and deploy deep learning models on a variety of Intel hardware, including CPUs, GPUs, and FPGAs. Pre-trained models from popular deep learning frameworks like TensorFlow, Caffe, and ONNX can be used in OpenVINO. It provides tools to optimize these models for deployment on Intel hardware. It also provides a common API for easy integration into existing applications. OpenVINO is fast, it provides accurate real-world results with high performance. Mainly OpenVINO has two components: (I) Model Optimizer, (II) Inference Engine.

Model Optimizer: The Model Optimizer is used to optimize and transform deep learning models into a format that can be used with Intel hardware. It applies a number of optimizations to a pre-trained model from a deep learning framework, such as TensorFlow or Keras, to make it more effective for use with Intel hardware. These improvements are:

- Keeps layers or operations that are supported by Intel hardware. Rest are removed.
- Applys quantization to reduce the model's weights and biases' precision that might result in considerable memory and time savings.
- Applys Layout transformations to make sure that the model's data is saved in a format that is best for the intended use of the hardware.

The optimized model then converts the model into an Intermediate Representation (IR) format that the Inference Engine can use. IR is the only format that gets accepted by the Inference Engine. IR is a pair of files describing the model:

- .xml - The topology of the network is described.
- .bin - Contains the weights and biases binary data.

Inference Engine: The Inference Engine is a C++ library comprising a collection of C++ classes. The Inference Engine is used to run inferencing on deep learning models that have been optimized and converted into an Intermediate Representation (IR) format using the Model Optimizer. The C++ library offers an API to read the IR, specify the input and output formats, and run the model on hardware. C libraries and Python bindings are also available, while C++ libraries are the primary implementation. This library contains the classes:

- to build an inference engine core object which is the main tool for working with devices and reading networks.
- to manipulate information stored on a network.
- to run and pass, inputs, and outputs.

The Inference Engine provides a common API for running inference on a wide range of Intel hardware, including CPUs, GPUs, and FPGAs, and supports a variety of computer vision and image processing tasks, such as object detection, image classification, and face recognition. The Inference Engine is made to benefit from the distinctive features of each type of Intel technology, such as the parallel processing potential of GPUs and the low-latency and power-efficient characteristics of FPGAs. As a result, the engine can deliver the best possible performance and efficiency for every type of hardware, and it also offers developers a single API that they can use regardless of the hardware platform they are aiming for. So, OpenVINO is a tool for making deep learning models more efficient and effective for deployment on any Intel hardware, which can lead to faster inference times, lower power consumption, and more precise real-time results with better performance on edge devices.

Chapter 2

MOTIVATION AND CONTRIBUTION

The application of CNN models, transfer learning, and ensemble (**Bhowal et al.** [**2022**]) techniques have produced encouraging results in the domains of image processing and computer vision for healthcare and medical diagnostics (**Paul et al.** [**2022**]). Deep learning models' accuracy is further increased by ensemble techniques (**Pramanik et al.** [**2022**]), which combine the results of various learning algorithms. These research endeavors have inspired us to investigate how well some well-known CNN outputs perform when combined. This chapter is divided into two sections, **2.1** contains the motivation of the work and **2.2** contains my contribution to the work.

2.1 Motivation

The motivation behind this thesis stems from the critical need for accurate and efficient methods for skin cancer classification. Skin cancer is a prevalent and potentially life-threatening disease that affects millions of people worldwide. Early detection plays a crucial role in improving patient outcomes and survival rates. However, manual diagnosis of skin cancer can be subjective, time-consuming, and prone to human error. Therefore, there is a pressing need for automated systems that can assist dermatologists in making accurate and timely diagnoses.

2.2 Contribution

The following are the primary contributions to my proposed work:

1. Development of the Levy Stable Ensemble Method:

   - Introducing the Levy stable distribution, known for its robustness and outlier handling capabilities.
   - Incorporating the Levy stable probability density function (PDF) into the ensemble technique.
   - Utilizing the robust ranking mechanism of the Levy stable ensemble for improved ensemble performance.
   - Enabling real-time hardware implementation of the ensemble method.

2. Introduction of the Beta Function-based Ensemble Technique:

   - Leveraging five standard convolutional neural network (CNN) models: DenseNet-121 (**Huang et al.** [**2016**]), EfficientNet-B3 (**Tan and Le** [**2019a**]), MobileNet-V3-Large (**Howard et al.** [**2019**]), Inception-V3 (**Szegedy et al.** [**2015**]), and ResNet-101 (**He et al.** [**2015a**]).
   - Generating confidence scores using transfer learning techniques.
   - Utilizing the Beta function-based ensemble technique for combining confidence scores.
   - Adaptively assigning priorities to classifiers based on confidence scores for each instance.

3. Exploiting the Properties of the Beta Function:

   - Utilizing the symmetricity property of the Beta function to allow flexibility in maintaining the sequence of confidence scores.
   - Leveraging the multivariate beta property for a seamless fusion of more than two networks.
   - Demonstrating the ease of calculating the beta function in cases of fusion involving more than two networks.

4. Evaluation on the HAM10k Dataset:

   - Evaluating the proposed ensemble techniques on the widely-used HAM10k dataset.
   - Comparing the performance of the Levy stable ensemble and Beta function-based ensemble with existing methods.
   - Highlighting the superior performance and accuracy of the proposed approaches.

5. Optimization for Real-Time Hardware Implementation:

- Utilizing the OpenVINO toolkit for optimizing the models for real-time hardware implementation.
- Achieving high-performance inference rates on both CPUs and GPUs.
- Ensuring the practical applicability of the proposed techniques in real-world scenarios.

This thesis contributes novel ensemble techniques for skin cancer classification, leveraging the Levy stable ensemble and the Beta function-based ensemble. The properties of the Beta function, such as symmetricity and multivariate beta, enhance the flexibility and ease of fusion. The evaluation on the HAM10k dataset demonstrates the superior performance of the proposed approaches. Furthermore, the optimization for real-time hardware implementation ensures the practical applicability of these techniques. The findings of this research can potentially assist dermatologists in making more accurate diagnoses, leading to early detection and timely treatment of skin cancer.

Chapter 3

LITERATURE SURVEY

This chapter is divided into two sections, **3.1** contains the previous work which has already been done for cancer classification, and **3.2** contains descriptions of different types of datasets of skin cancers.

3.1 Previous Work

The fusion of medical service and technology has accelerated the advancement of image-processing methods to help the medical industry. Several researchers have proposed methods for determining this type of skin disease. In this section, I will go over a few methods that have been documented in the literature.
In 2004, Sigurdsson et al (**Sigurdsson et al.** [**2004**]) developed a method that comprises a fully adaptable and reliable feedforward neural network classifier in addition to feature extraction for Raman spectra. In 2009 a model has been proposed that compares different segmentation methods by Silveira (**Silveira et al.** [**2009**]) for melanoma detection in dermoscopy images. It does not propose a new segmentation method but rather evaluates the performance of several existing methods. In 2009, M.Emre Celebi (**Celebi et al.** [**2009**]), Proposed a range of approaches for lesion border detection in dermatoscopy images with this global thresholding, adaptive thresholding, and Otsu's thresholding to improve the accuracy of lesion border detections. In 2012, Mariam A. Sheha et al. described in their research paper (**Mabrouk et al.** [**2012**]) an automated melanoma detection algorithm presented and applied to a set of dermoscopy images. To distinguish between Melanocytic Nevi and Malignant melanoma, features are collected based on the Grey Level Co-occurrence Matrix (GLCM) (**Vadakkenveettil et al.** [**2012**]) and using Multilayer Perceptron Classifiers (MLP) (**Popescu et al.** [**2009**]). In 2013, Catarina Barata in this study (**Barata et al.** [**2013**]) examines two distinct methods for identifying melanomas in dermatoscopy images. Skin lesions are categorized using global approaches in one system. While another system uses the Bag of Characteristics classifier and local features. The use of common deep learning made numerous problems, including the classification of skin cancer. While using CNN makes it easier to solve. By automatically extracting the feature, CNN increased its accuracy for detection and reduced the difficulty of constructing it (**Barata et al.** [**2013**]). In 2015, Diwakar Gautam et al. used Color images of the cancers in their work (**Gautam and Ahmed** [**2015**]) to categorize melanoma into cancerous and non-cancerous classifications with the help of support-vector

machines (SVM) (**Evgeniou and Pontil** [**2001**]) and optimized using Sequential minimal optimization (SMO) (**Kuan et al.** [**2009**]). They used an illumination compensation-based segmentation algorithm as part of the pre-processing step. To eliminate noise from a lesion after segmentation an iterative dilation approach was used. Then some prominent features like asymmetric lesion behavior, border irregularity, etc. were calculated from the segmented image. Lastly, these feature vectors were fed into an SVM classifier that is used to discriminate between cancerous and noncancerous skin lesion samples. In 2015, V. Krishna Sree et al. proposed a framework (**V. Krishna Sree** [**2015**]) that automatically corrects and segments skin lesions from input photographs. The first section of their research models lighting variation using a proposed multi-stage illumination modeling algorithm and then uses that model to correct the original photograph. Second, the corrected photograph is used to learn a set of representative texture distributions, and the texture distinctiveness metric is calculated for each distribution. Finally, a texture-based segmentation algorithm uses the occurrence of representative texture distributions to classify regions of the photograph as normal skin or lesions. The segmentation results can be fed into separate feature extraction and melanoma classification algorithms. In 2016, Shereen Afifi et al. proposed a hardware design model in their research paper (**Afifi et al.** [**2016**]) for implementing a linear binary SVM classifier in an FPGA for online melanoma classification is proposed. The proposed system is implemented on a recent hybrid Zynq platform using the most recent High-Level Synthesis design methodology. The implemented system meets critical embedded system constraints with high performance, low hardware resource utilization, and low power consumption. In 2017, Poornima M. S. et al. present a method for detecting Melanoma Skin Cancer using image processing tools in their research paper (**Poornima M. S.** [**2017**]). The skin lesion image is fed into the system, which then analyses it using image processing techniques to determine the presence of skin cancer. The Lesion Image Analysis tools examine the various Melanoma parameters analysis for image segmentation and feature stages. The extracted feature parameters are used to categorize the image as non-melanoma or melanoma cancer lesions. In 2018, Shereen Afifi et al. proposed a co-design of hardware/software for employing the SVM classifier onto FPGA to realize melanoma detection on a chip in their paper (**Afifi et al.** [**2019**]). The SVM was employed on a hybrid FPGA (Zynq) platform using the modern Ultrafast High-Level Fusion design methodology, follow-on in efficient melanoma

classification on the chip. The results of their research show a classification accuracy of 97.9% and a significant hardware acceleration rate of 21 with only 3% resource utilization and 1.69W power consumption. In 2018, in order to improve the accuracy of skin cancer classification, Harangi et alHarangi [2018], proposed developing an ensemble of CNN models. They created an ensemble model for three classes of skin cancer for the CNN models of GoogleNet (**Szegedy et al.** [**2014**]), AlexNet (**Krizhevsky et al.** [**2012**]), ResNet (**He et al.** [**2015b**]), and VGGNet (**Simonyan and Zisserman** [**2015**]), correspondingly. A system is proposed in 2019 by Nawal Soliman et al. (**ALEnezi** [**2019**]) for the dissection of skin diseases using color images without the need for a doctor. The system will help significantly in the detection of melanoma with another two skin diseases. The proposed system is divided into three parts- the first part being image resizing. To resolve the problem of different image sizes in the database an input image is either increase or decrease in size. The second part is feature extraction using Alex Net. After extracting features, the role of classification is to classify the image via a Support Vector Machine (SVM). The system was able to identify the three diseases with high accuracy. In 2019, Hung N. Pham et al proposed an automated method for melanoma detection and lesion segmentation in their paper Pham et al. [2019]. The authors have used Inception V4 (**Szegedy et al.** [**2016**]), Resnet 152 (**He et al.** [**2015b**]), and Densenet 161 (**Huang et al.** [**2016**]) for melanoma classification purposes. U-Net (**Ronneberger et al.** [**2015**]), and U-Net with VGG 16 (**Simonyan and Zisserman** [**2015**]), an ensemble method were used for segmentation purposes. The dataset that has been used to train these models is ISIC 2017 (**Berseth** [**2017**]). This dataset contains a total of 2750 images. Out of these images 2000 images were used for training purposes, 150 were for validation and 150 were used for testing purposes. In 2019, Md Ashraful Alam Milton used (**Milton** [**2019**]) PNASNet-5-Large (**Liu et al.** [**2017**]), InceptionResNetV2 (**Szegedy et al.** [**2016**]), SENet154 (**Hu et al.** [**2017**]), and InceptionV4 (**Szegedy et al.** [**2016**]) neural networks that use recent deep learning-based models to detect melanoma. Before being fed into the network, dermoscopic images are properly processed and augmented. He put his methods to the test using the International Skin Imaging Collaboration (ISIC) 2018 challenge dataset (**Ali et al.** [**2019**]). For the PNASNet-5-Large model, the proposed system received the highest validation score of 0.76. In 2020, Tsung-Chieh Lin et. al. suggested in their work (**Lin and Lee** [**2020**]) the use of an ensemble CNN for

an experiment on classification with multiple classes that combined image preprocessing, deep learning, and risk assessment on images of skin cancer dermoscopy with metadata. They essentially showed how the general assembling ensemble flow works through the stages of data preprocessing, CNNs grouping, meta-data concatenation, the first CNNs ensemble, and the second meta-classifiers ensemble. In order to control various risk management in exchange tactics for particular categories or metrics without having to retrain CNN via loss function modification, they also implemented cost adjustment evaluation in the ensemble phase as opposed to the deep learning phase. In 2021, Khadija Safdar et al. proposed a fully automated ensemble method by using Resnet 50 (**He et al. [2015b]**) and Inception V3 (**Szegedy et al. [2015]**) for melanoma detection in their paper (**Safdar et al. [2021]**). They have used PH2 (**Li et al. [2021]**), Med-mode (**Giotis et al. [2015]**), and ISIC 2020 (**Yao [2023]**) datasets for their proposed work. First data pre-processing has been done then data segmentation had been performed using FCN-8 (**Long et al. [2014]**). Then binary classification had been done using the ensemble method. Ahmed A. Elngar et al. proposed a system in 2021 to create an Android mobile application (MAA) (**Elngar et al. [2021]**), this research effectively developed a system that integrates Convolutional Neural Network (CNN) and Support Vector Machine (SVM) classifier to predict skin cancer. The dataset they used contains 3000 images of 6 classes from various sources like Beni-Suef University Hospital, Cairo University Hospital, etc. Kritika Sujay Rao et al. proposed a multiclass deep learning model in their research paper (**Rao et al. [2020]**)(to distinguish between healthy skin and skin that has a disease, as well as classification of skin diseases into their main classes, such as Melanoma, Melanocytic Nevi, and 5 other class of diseases. They trained their model using Deep Learning. The number of classifiers is significantly reduced because Machine Learning uses large datasets. The machine self-learns and divides the supplied data into levels of prediction, and provides precise outcomes in a relatively brief period of time, encouraging and supporting the growth of dermatology. They used the Convolutional Neural Network (CNN) as a machine learning tool. In 2021, Parvathaneni Naga Srinivasu et. al. (**Srinivasu et al. [2021]**) detected 6 diseases along with Melanoma using MobileNet-V2 (**Sandler et al. [2018a]**) and Long Short-Term Memory (LSTM) (**Hochreiter and Schmidhuber [1997]**) integrated together in 2021. The proposed method outperformed other algorithms such as Heuristic approach for Real-Time Image Segmentation (HARIS) Agarwal et al. [2020], Fine-Tuned Neural Networks(FTNN) Lee et al. [2018], MobileNet-V1

Howard et al. [2017], and MobileNet-V2 Sandler et al. [2018a]. The classification of the type of skin disease is done using MobileNet-V2, and the model's performance is improved using LSTM. They have used HAM10K dataset (**Tschandl [2018]**) and with more than 85% accuracy, the proposed method has performed better than competing ones. In 2021, Runyuan Zhang proposed a computer-based melanoma detection system (**Zhang [2021]**) based on EfficientNet-B6 (**Tan and Le [2019b]**) analysis of images of skin lesions which can record more fine-grained features. For the experiment, he used a large publicly available dataset ISIC 2020 Challenge Dataset (**Yao [2023]**) produced by the International Skin Imaging Collaboration, and images from several primary medical sources. And compared to earlier widely used melanoma classifiers on the same dataset, the proposed method had outperformed. In 2021, Jagadish Kumar S. et. al have proposed a method for skin cancer classification in their paper (**Jagadish Kumar et al. [2021]**). The authors have used XG Boost classifier (**Chen and Guestrin [2016]**) and EfficientNet (**Tan and Le [2019b]**) in their proposed method. The proposed architecture has two stages in it. Stage I uses XG Boost classifier to compare the resemblance between the training and testing images. Stage II uses an Efficient Net on TFRecord files for prediction purposes. The datasets that have been used for training and testing purposes are ISIC 2019 (**Kassem et al. [2020]**) and ISIC 2020 (**Yao [2023]**), consisting of 25,331 and 33,126 images, respectively. In 2021, Takfarines Guergueb et al. (**Guergueb and Akhloufi [2021]**) described how to detect melanoma skin cancer and investigate suspicious lesions using recent deep CNN approaches. More than 36,000 images extracted from various datasets were used in the tests. In 2022, Shaden Abdulaziz AlDera et al. (**AlDera and Othman [2022]**) demonstrated a classification model to detect acne, cherry angioma, melanoma, and psoriasis. The proposed model performs 5 steps to get the decision scores. The first step was obtaining the images, the second was preprocessing those images and then segmenting the lesion, and the fourth step was extracting its features from train set images. And the final step was classification. Import the test set images and process them in the same way as steps 1,2 and 3. The features that extract from the train set images are stored in a knowledge base. Compare the features that extract from the test set image with the feature stored in the knowledge base. In 2022, Jaynab Sultana et. al. proposed an Inception V3 (**Szegedy et al. [2015]**) based skin cancer classification network in their paper (**Sultana et al. [2022]**). It uses transfer learning. This proposed method used a data set

from a Kaggle dataset which contains 2730 images with 3 classes. In 2022, Mohammad Naved Qureshi and Mohammad Sarosh Umar proposed (**Qureshi and Umar** [**2022**]) a CNN model for skin cancer classification. First, the date had been pre-processed. Secondly, the feature had been extracted and lastly, the classification had been done. The dataset which had been used is ISIC. 90% of this dataset had been used for training purposes and 10% for testing propose. The CNN model the authors used is the Xception model (**Chollet** [**2016**]). In 2022, Dr. K. Padmavathi et al. present in their research paper (**Padmavathi et al.** [**2022**]) a hybridized method for detecting melanoma skin cancer that could be used to detect any concerning lesion. This study proposes an automated skin lesion classification method. A pre-trained and fine-tuned deep learning network is used. The results are then compared using various transfer learning techniques. In 2022, Bhuvaneshwari Shetty in this study (**Shetty et al.** [**2022**]) created a fully automated method for identifying and categorizing skin lesions using machine learning and specialized convolutional neural networks. The development of ensemble algorithms improves the accuracy of classification. Ensemble methods take advantage of individual models' variability to achieve their improved accuracy. Applying ensemble methods, multiclass skin cancer classification systems have recently been established in the literature.

3.2 Various Datasets

Dermoscopic photos are being specialized because they provide a clearer image from skin reflections and allow for higher diagnostic accuracy than traditional eye examinations. The initial requirement for learning trustworthy algorithms has always been high-quality data. Notably, a huge proportion of labeled data is required while training a deep neural network. Therefore, reliable diagnosis labels and high-quality data on skin disorders are essential for creating sophisticated algorithms. For identifying diseases, there are three main modalities that can be used: clinical photos, dermoscopy photos, and pathological photos.

HAM10000 (Tschandl [2018]): Dermoscopy images from various populations, obtained and preserved by various modalities, are included in the HAM10000 (Human Against Machine with 10000 training images) dataset published by Tschandl et al. The dataset made up of 10015 dermoscopy images [13] that are categorized as training and testing data, is available to the general public through the ISIC

archive. The image resolution used is 800 × 600 pixels. The cases comprise a comprehensive assortment of all significant pigmented lesion diagnostic subtypes. This dataset includes seven image classes Melanoma (1113), Basal Cell Carcinoma (514), Actinic Keratoses (327), Dermatofibromas (115), Benign Keratoses (6705), Vascular Lesions (142), and melanocytic nevi (1099) are some examples of skin lesions.

PH2 (Li et al. [2021]): Mendonca et al. created the PH2 dataset of dermoscopy images in 2003, which is openly accessible. It includes 40 melanoma images, 80 atypical nevi images, and 80 common nevi images. The Tuebingen Mole Analyzer equipment was used to obtain the dermoscopy images at the Dermatology Service of Hospital Pedro Hispano (Matosinhos, Portugal) under identical circumstances and at a 20x magnification. These images are of 768 x 560-pixel resolution in an 8-bit RGB colour format. The dataset includes all of the images' medical annotation, such as clinical and histological diagnoses, medical segmentation of lesions, and evaluation of various dermoscopic criteria (such as colours, pigment networks, dots/globules, streaks, regression areas, and blue-whitish veil). The dataset is frequently used as a benchmark dataset for assessing algorithms for melanoma diagnosis since it contains extensive metadata.

ISIC (Cassidy et al. [2022]): Dermoscopy images were compiled into a sizable publicly accessible dataset by the International Skin Imaging Collaboration (ISIC). The dataset includes more than 20,000 photos from renowned clinical institutions across the world that were collected from various technologies employed at each facility. The ISIC dataset was initially made available for the public benchmark competition on dermoscopy image analysis in 2016. In terms of segmentation, dermoscopic feature detection, and classification, the challenge aimed to produce a dataset to support the development of automated melanoma diagnosis algorithms. The ISIC hosted the challenge's second round with an expanded dataset in 2017. The expanded dataset offers 2000 training images together with super-pixel masks for dermoscopic feature extraction, segmentation masks, and annotations for classification. Three classes—melanoma, seborrheic keratosis, and nevus—are established for the photographs. The ISIC also offers a validation set, which consists of an additional 150 photos, for review. The ISIC 2017 challenge dataset includes 288 images of seborrheic keratosis, 2107 images of nevus, and 803 images of melanoma.

MED-NODE (Giotis et al. [2015]): The Department of Dermatology at the University Medical Center Groningen's (UMCG) digital image library provided the 100 nevus images and 70 melanoma images that make up the MED-NODE dataset. It is utilized to create and evaluate the MED-NODE system for detecting skin cancer using microscopy images.

DermIS (TL and G [2012]): The largest dermatology information service on the internet is DermIS.net. It provides comprehensive image atlases (DOIA and PeDOIA) with case studies, extra details, diagnoses, and differential diagnoses for practically all skin conditions.

Dermofit (Mukherjee et al. [2019]): The Dermofit Image Library is a collection of 1300 high-quality skin lesion photographs that were captured with internal color standards and under controlled conditions. This image library represents 10 kinds of lesions which are Melanoma (76), Actinic Keratosis (45), Melanocytic Nevus / Mole (331), Basal Cell Carcinoma (239), Squamous Cell Carcinoma (88), Intraepithelial Carcinoma (78), Seborrhoeic Keratosis (257), Pyogenic Granulo (24), Haemangioma (96) and Dermatofibroma (65). Every image has been assessed by expert professionals (including dermatologists and dermatopathologists). Images show a close-up of the lesion and its surrounding healthy skin. Each lesson comes with a binary segmentation mask that identifies the lesion region.

7-point criteria (Kawahara et al. [2019]): Over 2000 dermoscopy and clinical photos of skin lesions are included in the 7-point criteria evaluation dataset, together with the disease diagnosis and 7-point checklist criteria. The dataset can also be used by starting with the Python package derm7pt5. The dataset is preprocessed, and the data is then transformed into a more usable format. It consists of 252 cases of melanoma, 759 cases of non-melanoma, etc.

Edinburgh (Ballerini et al. [2013]): The Edinburgh dataset includes 257 cases of seborrheic keratosis, 331 cases of nevus, and 76 cases of melanoma.

TABLE 3.1: **Description of Various Datasets**

Dataset	**Data Volume**	**Image Type**	**Image Resolution**	**Other Disease Type**	**Others**
HAM10k	10015	Dermoscopy images	600 × 450	Basal Cell Carcinoma, Actinic Keratoses, Dermatofibromas, Benign Keratoses, Vascular Lesions, melanocytic nevi	-
PH2	200	Dermoscopy images	768 × 560	Common nevi, atypical nevi	-
ISIC	>20,000	Dermoscopy and clinical image	Various Sizes	seborrheic keratosis, benign Nevis	Segmentation mask provided
MED-NODE	170	Clinical Image	Various Sizes	Nevi	-
DermIS Image Library	Thousands	Clinical Image	Various Sizes	All kinds of skin disease	-
Dermofit Image Library	1300	Clinical Images	Various Sizes	10 categories	Segmentation mask provided
7-point criteria evaluation	>2000	Dermoscopy and clinical image	Various Sizes	Non-melanoma	Structured metadata provided
Edinburgh dataset	664	Dermoscopy and clinical image	Various Sizes	Nevus, seborrheic keratosis	-

Chapter 4

DEEP NEURAL NETWORKS USED

This chapter contains four sections, 4.1 contains descriptions of 7 basic CNN models, 4.2 contains the proposed Levy Stable and Beta ensemble methods, 4.3 contains the workflow of OpenVINO, and 4.4 contains the entire methodologies of Levy Stable and Beta ensemble methods.

4.1 Basic CNN Models

In this section, I will discuss the basic models which I have used later for ensemble methods. I took MobileNet-V2 and EfficientNet-B0 for Levy Stable PDF-based ensemble method. And I used ResNet-101, DenseNet-121, InceptionNet-V3, EfficientNet-B0, and MobileNet-V3-Large for the Beta function-based ensemble method. I fused 2 or 3 models together at a time for the Beta function-based ensemble method.

MobileNet-V2 (Sandler et al. [2018b]): MobileNet-V2 incorporates other techniques that further enhance its performance on mobile and embedded devices. One such technique is the use of linear bottlenecks with residual connections. This approach helps in reducing the network's computational cost while maintaining a good level of accuracy. By employing linear activations and residual connections, MobileNet-V2 enables the flow of information through the network while minimizing the impact on computational resources. Another notable feature of MobileNet-V2 is the use of inverted residual blocks. These blocks consist of a lightweight bottleneck layer followed by an expansion layer. The bottleneck layer reduces the dimensionality of the input, while the expansion layer increases it again. This design allows the network to capture complex features using fewer parameters, making it more efficient for mobile and embedded devices. MobileNet-V2 also utilizes a technique called "squeeze-and-excitation" to enhance its discriminative power. This technique introduces a mechanism that learns to emphasize informative features and suppress less relevant ones. By adaptively recalibrating channel-wise feature responses, MobileNet-V2 can focus on the most important features, leading to improved accuracy with minimal additional computational cost. The network's performance can be further optimized through techniques such as model quantization, which reduces the precision of weights and activations, and efficient model compression methods. These techniques enable MobileNet-V2 to achieve a good trade-off between computational efficiency and accuracy, making it highly suitable for deployment on resource-constrained devices. MobileNet-V2

stands as a powerful solution for computer vision tasks on mobile and embedded devices. Its efficient design, combined with techniques like linear bottlenecks, inverted residual blocks, and squeeze-and-excitation, allows for real-time inference and accurate predictions while minimizing computational requirements. As a result, MobileNet-V2 has become a widely adopted architecture for various mobile vision applications, contributing to the advancement of AI technology in the context of mobile computing. The architecture of MobileNet-V2 has been shown in Fig. **4.1**.

EfficientNet-B0 (Tan and Le [2019b]): EfficientNet-B0 is the baseline model in the EfficientNet family and serves as a foundation for the other variants. It is designed to strike a balance between model complexity and computational efficiency, making it an ideal choice for resource-constrained environments. The compound scaling method used in EfficientNet-B0 involves scaling the network's depth, width, and resolution simultaneously. This approach ensures that all dimensions are increased proportionally, enabling the model to capture more complex patterns and representations. By leveraging this compound scaling technique, EfficientNet-B0 achieves impressive performance gains compared to traditional architectures. EfficientNet-B0's efficiency stems from its ability to achieve higher accuracy with fewer parameters and computations. The model carefully optimizes the trade-off between model size and performance, allowing for efficient deployment on various devices with limited computational resources. With its strong performance and efficiency, EfficientNet-B0 has gained popularity in computer vision applications. Its versatility makes it suitable for a wide range of tasks, including image classification, object detection, and image segmentation. Researchers and practitioners often turn to EfficientNet-B0 as a reliable and effective baseline architecture for their experiments and applications. Furthermore, EfficientNet-B0 serves as a benchmark for evaluating and comparing other neural network architectures. Its balanced scaling strategy provides a reference point for understanding the relationship between model size, computational cost, and performance. This has contributed to advancements in the field of neural network design and architecture search. EfficientNet-B0 offers a compelling solution for achieving advanced performance while maintaining computational efficiency. Its compound scaling method and carefully balanced dimensions make it an attractive choice for various computer vision tasks. EfficientNet-B0 continues to drive innovation in the field of deep learning and serves as a valuable asset in the pursuit of efficient and accurate models. The architecture of EfficientNet-B0 has been shown in Fig. **4.2**.

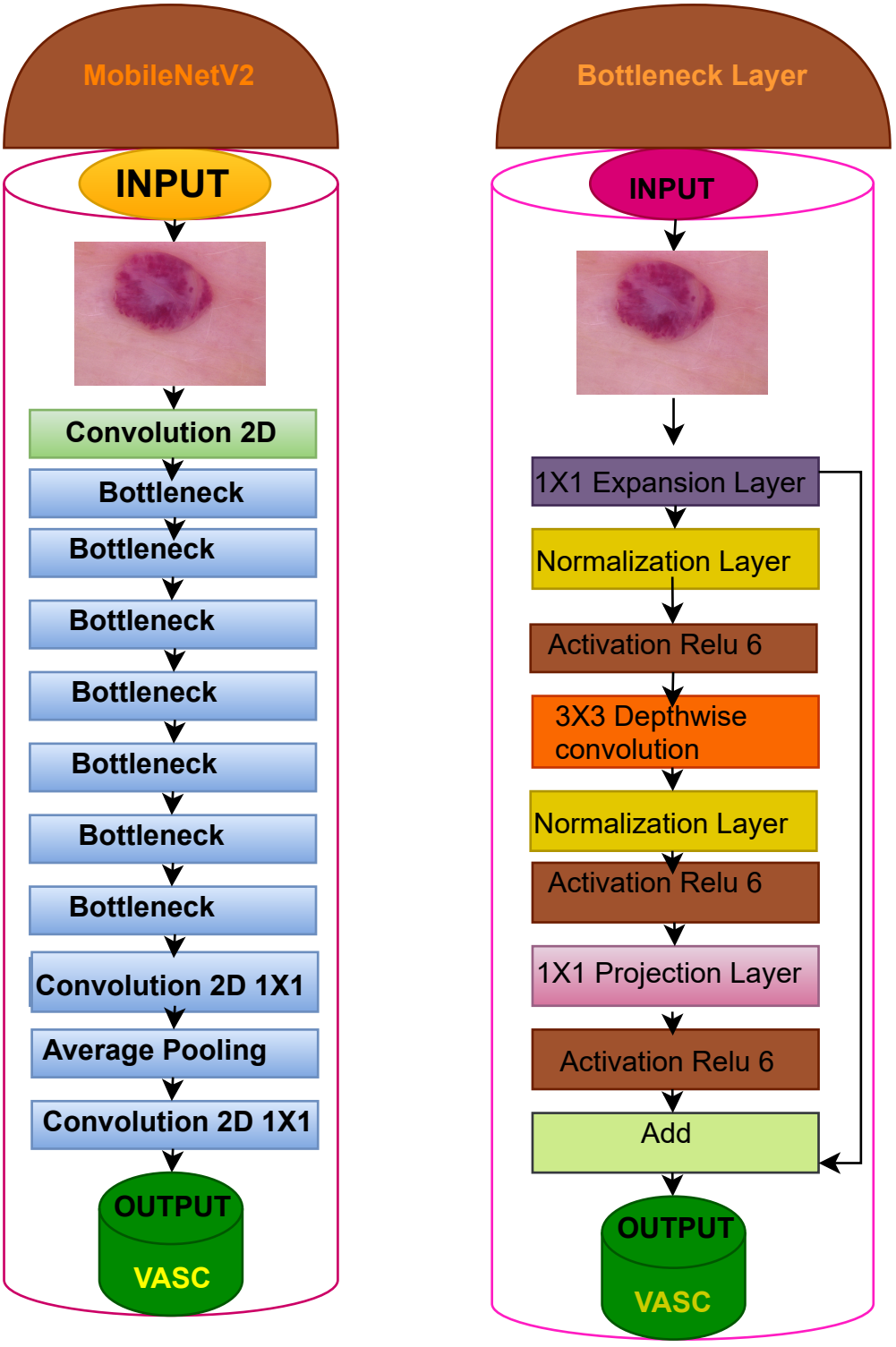


FIGURE 4.1: **An illustration of MobileNet-V2 model architecture**

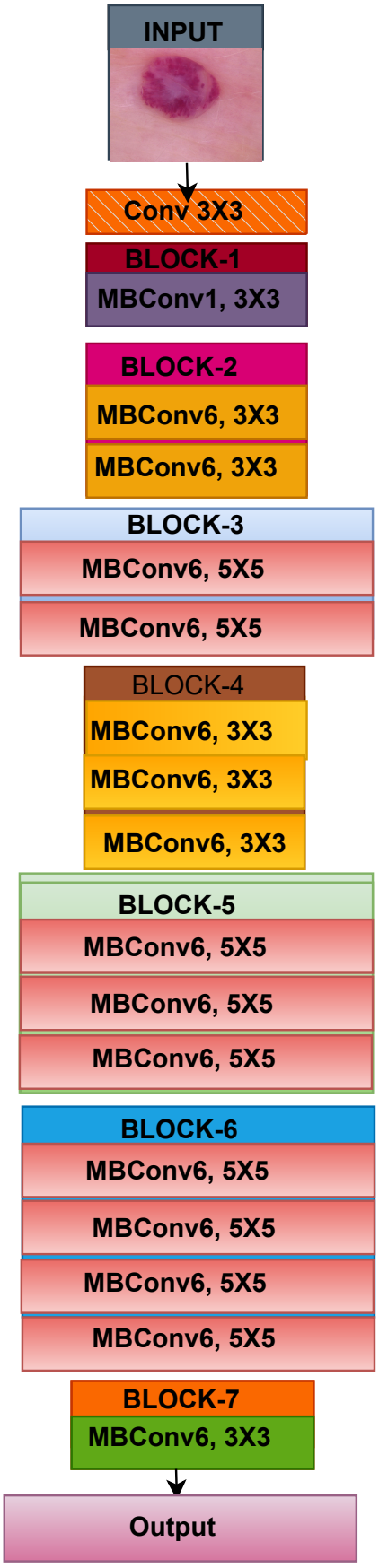


FIGURE 4.2: **An illustration of EfficientNet-B0 model architecture**

ResNet-101 (He et al. [2015a]): In the case of the traditional convolutional neural network model, there is a limit to the depth of layers. It has been experimented that with the increase of layers on the top of a network the error percentage increases and the performance decreases. This issue might be seen due to the optimization function, initialization of the network, and especially due to the vanishing gradient problem. To overcome these issues a deep convolutional neural network architecture has been introduced in 2015 by Kaiming He et al. (**He et al.** [**2015a**]) as part of the Microsoft Research team's work on deep residual learning for image recognition. ResNet architecture has a "network in a network" architecture, also known as micro-architecture. The macro-architecture results from a combination of micro-architectures containing common CONV, POOL, etc. layers. The depth of the ResNet network model is higher than VGG16 (**Tammina** [**2019**]) and VGG19 (**Sudha and Ganeshbabu** [**2020**]) with improved accuracy and performance. The architecture is based on the concept of residual learning, which aims to make it easier for a network to learn the underlying mapping between inputs and outputs by adding shortcut connections between the layers. The ResNet-101 architecture has 101 layers, including a convolutional layer and max pooling layer, followed by 4 stages and a total of 33 residual blocks, each with varying residual blocks. The first stage has 3 residual blocks, the second has 4, the third has 23, and the fourth has 3. Each residual block consists of a sequence of convolutional layers and identity shortcut connections. The residual layers are followed by an average pooling layer and a fully connected layer at the end to produce the final output. The basic structure of a residual block in ResNet-101 includes convolutional layers with kernel sizes of 3x3, followed by a rectified linear unit (ReLU) activation function. The output of the second convolutional layer is then added to the input of the block, which is passed through an identity function. The resulting output is then fed to the next block in the sequence. The shortcut connections in ResNet-101 allow for gradients to be easily propagated through the network, which helps to reduce the vanishing gradient problem. The architecture of ResNet-101 has been shown in Fig. **4.3**.

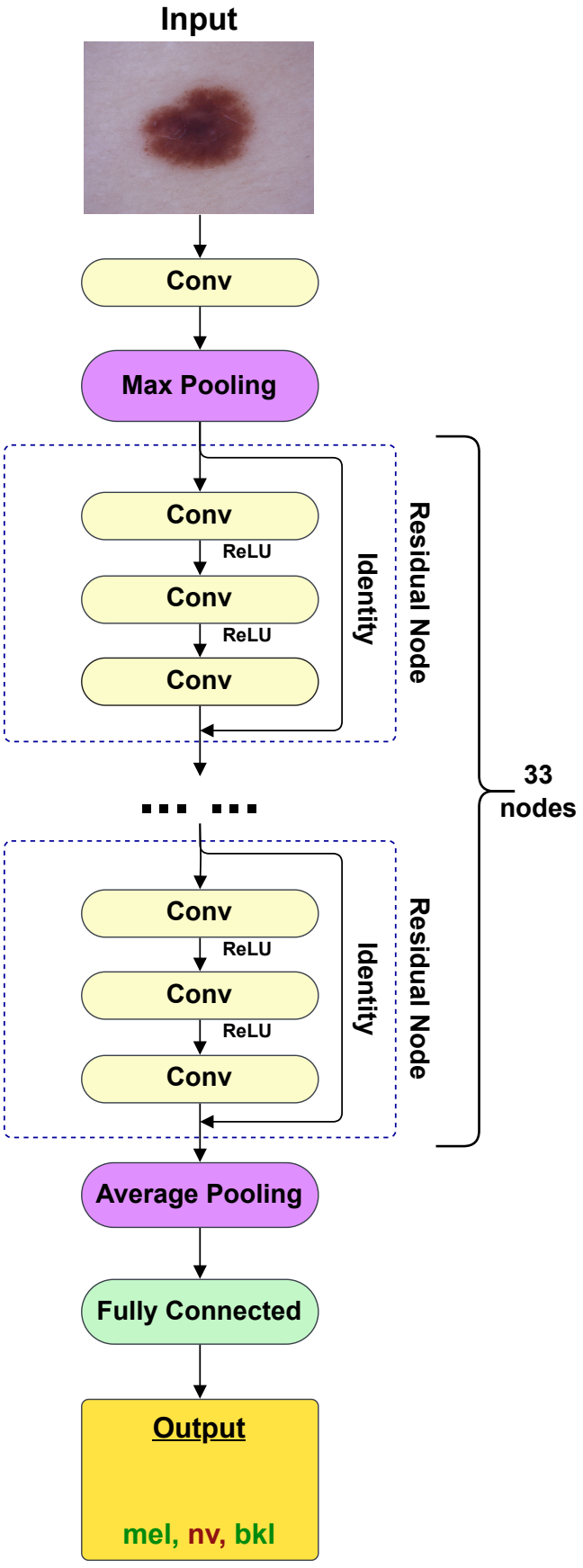


FIGURE 4.3: **An illustration of Resnet-101 model architecture**

DenseNet-121 (Huang et al. [2016]): DenseNet-121 was first introduced in 2016 by Huang et al. In this convolutional network, each layer connects to every other layer in a feed-forward manner. DenseNet-121 has 121 layers starting with a convolution layer followed by a max pooling layer and followed by several dense blocks, each of which contains multiple convolutional layers with a fixed number of filters. In each dense block, the output of each layer is concatenated with the outputs of all previous layers in that block and is the input of the next layer. This dense connectivity pattern allows for better feature reuse and facilitates the flow of information through the network. DenseNet-121 also includes transition layers that reduce the spatial dimensions of the feature maps and the number of filters before passing them to the next dense block. In the transition lay there is a convolution layer and an average pooling layer. This layer helps to reduce the network's computational complexity and prevent overfitting. The dense layers and transition layers are followed by an average pooling layer and a softmax at the end to produce the final output. The architecture of Densenet-121 has been shown in Fig. 4.4.

Inception-V3 (Szegedy et al. [2015]): Inception-V3 is a convolutional neural network architecture that was introduced in 2015 by Szegedy et al. It is a variant of the Inception family of architectures, which are known for their ability to efficiently capture complex patterns in images with a relatively small number of parameters. The primary goal of Inception-V3 is to consume less computing power by altering the Inception-V2 architecture (**Ioffe and Szegedy [2015]**). The key feature of Inception-V3 is the use of the "blocks," which consist of several parallel convolutional layers of different filter sizes (1x1, 3x3, and 5x5) and a max pooling layer followed by a dropout layer at the end. The outputs of these layers are then concatenated and passed to the next layer. Inception-V3 uses smaller convolution and lesser parameters. Such as instead of using one 5x5 filter with 25 parameters, two 3x3 filters with 18 (3x3 + 3x3) parameters can be used. This design allows the network to capture features at different scales and resolutions. Inception-V3 consists of 24 million parameters and the depth is 48 layers. Compared to VGG and ResNet, Inception-V3's weights are less only 96MB in size. A total of 1000 categories could be used by Inception-V3 to categorize photos. The architecture of Inception-V3 is shown in Fig. 4.5.

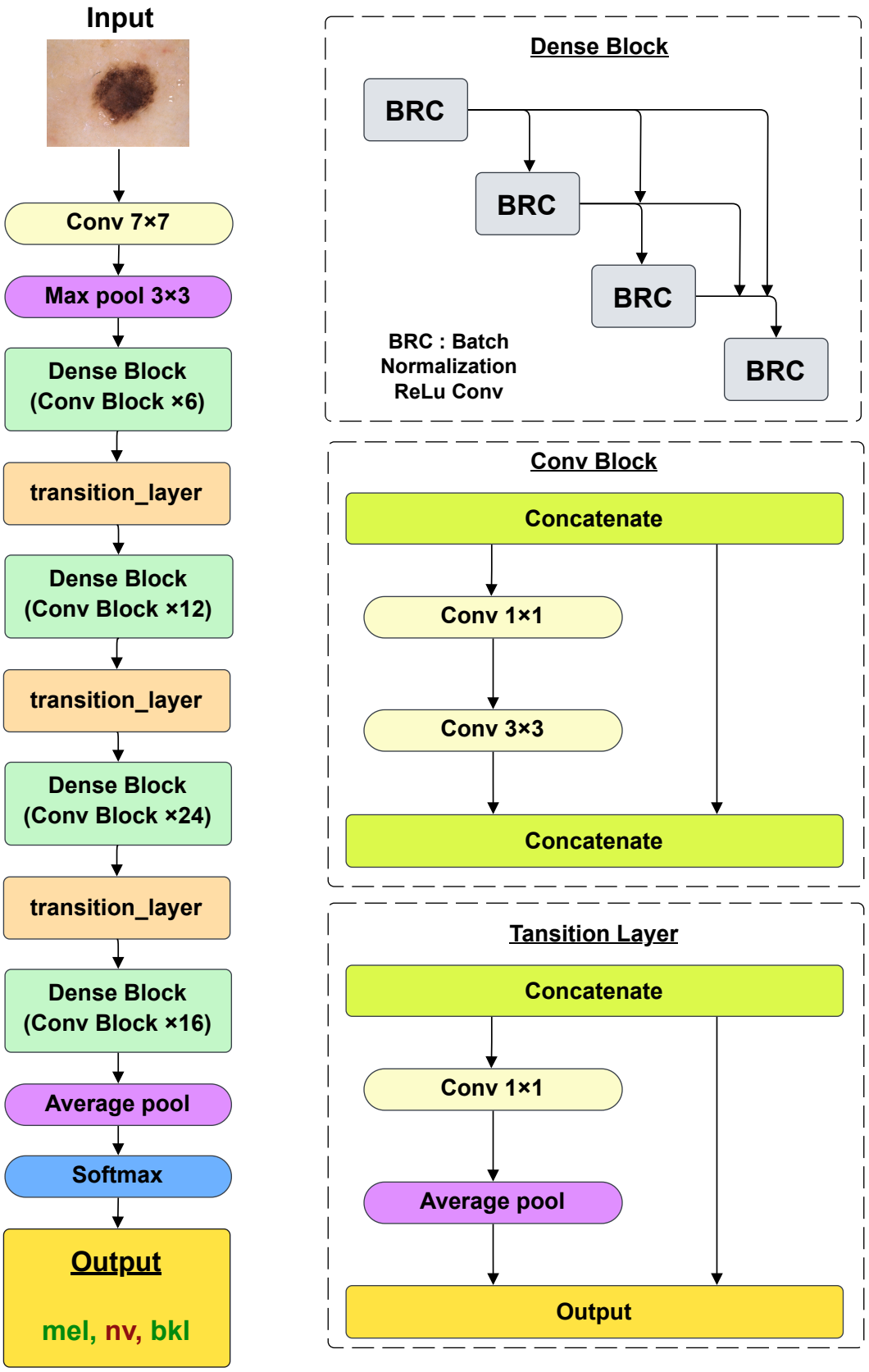


FIGURE 4.4: **An illustration of DenseNet-121 model architecture**

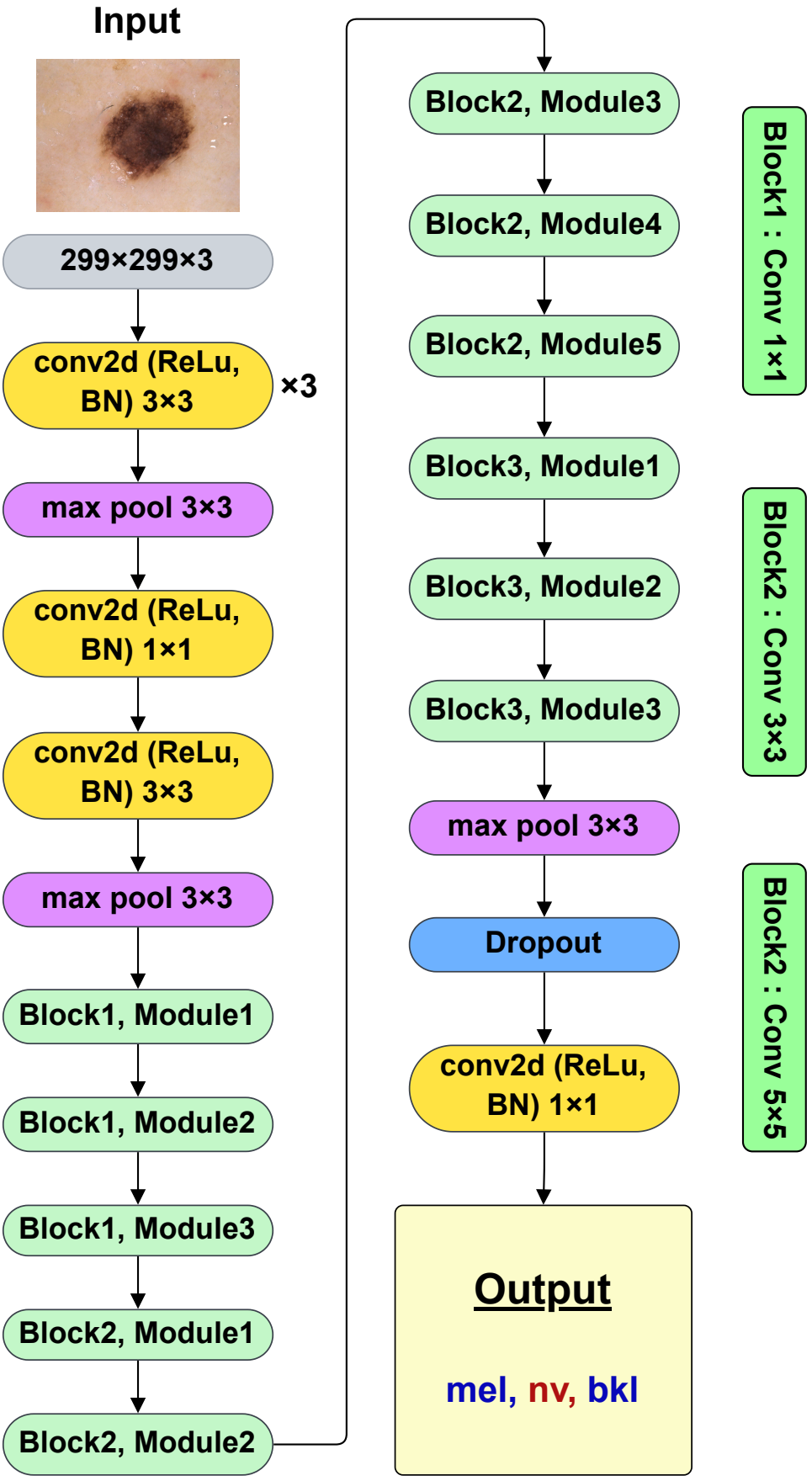


FIGURE 4.5: **An illustration of Inception-V3 model architecture**

MobileNet-V3-Large (Howard et al. [2019]): A collection of neural network topologies known as MobileNet-V3 has been designed specifically for mobile and embedded devices. Subsequently, novel architecture advancements have enhanced MobileNetV3. There have been improvements in a few areas, such as complementary search techniques and a new, efficient network design. MobileNet-V3 is meant to offer great accuracy while requiring the fewest parameters and computational power possible. There are two variations of MobileNet-V3: Mobile-Net-V3-Large and MobileNet-V3-Small. MobileNet-V3-Large has more parameters and a higher computational cost than MobileNet-V3 Small and is intended for high-accuracy applications. MobileNet-V3-Large uses bottleneck filters, squeeze and excite (SE) blocks, a spatial attention module, and a softmax activation function to produce the output. In order to expand the number of channels, bottleneck filters combine a depthwise convolution layer with a pointwise convolution layer. The channel-wise feature responses in MobileNet-V3-Large can be adaptively recalibrated using SE blocks. This enhances accuracy and enables the network to concentrate on the most instructive aspects. SE block consists of an adaptive pooling layer, convolutional filters, ReLU function, etc. The unique softmax activation function utilized by MobileNet-V3-Large is intended to be faster and more precise than previously regularly used activation functions. The architecture of MobileNet-V3-Large is shown in Fig. 4.6.

EfficientNet-B3 (Tan and Le [2019a]): Tan and Le presented the EfficientNet-B3 CNN architecture in 2019. It is a member of the EfficientNet family of networks, which aims to provide cutting-edge performance on image classification problems with the least amount of parameters and processing resources needed for training and inference. The EfficientNet is a conventional neural network, for the purpose to reach advanced performance with greater computation efficiency. This model utilizes an advanced compound scaling method that effectively increases all depth, width, and resolution dimensions. Through the use of compound scaling, Efficient-Net has the capability to perform better than earlier models while requiring a small number of parameters and computations. EfficientNet is a well-liked option for a variety of computer vision applications since it offers a potent tool for obtaining advanced performance with greater computing efficiency. Model depth, width, and resolution, which are crucial components in defining a neural network's accuracy and computing cost, are balanced by EfficientNet-B3. The depth, width, and resolution of the network are all evenly scaled via the compound scaling method used

by EfficientNet-B3. As a result, the network can attain great accuracy while using fewer parameters and less processing power. The architecture of EfficientNet-B3 is shown in Fig. 4.7.

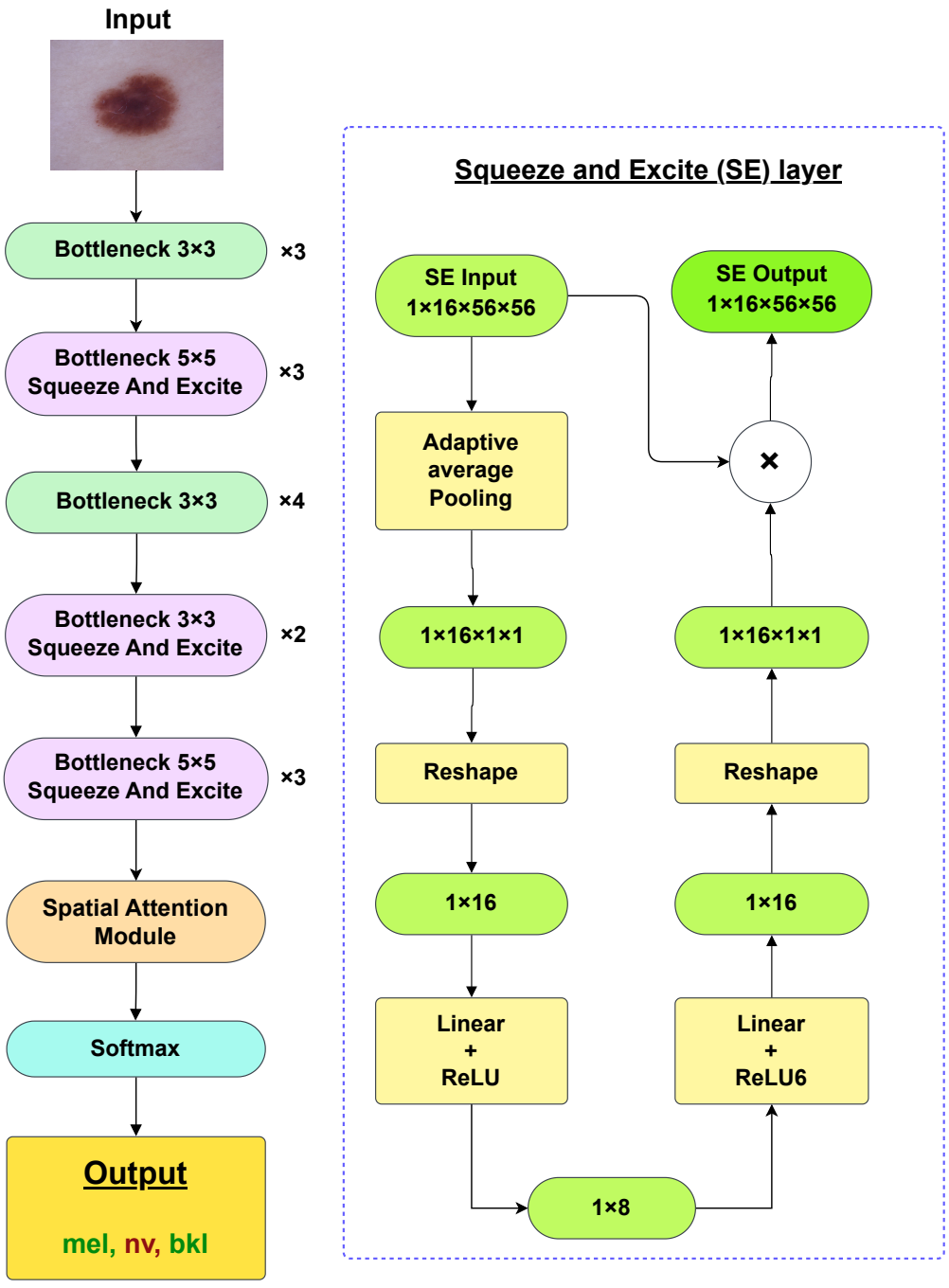


FIGURE 4.6: **An illustration of MobileNet-V3-Large model architecture**

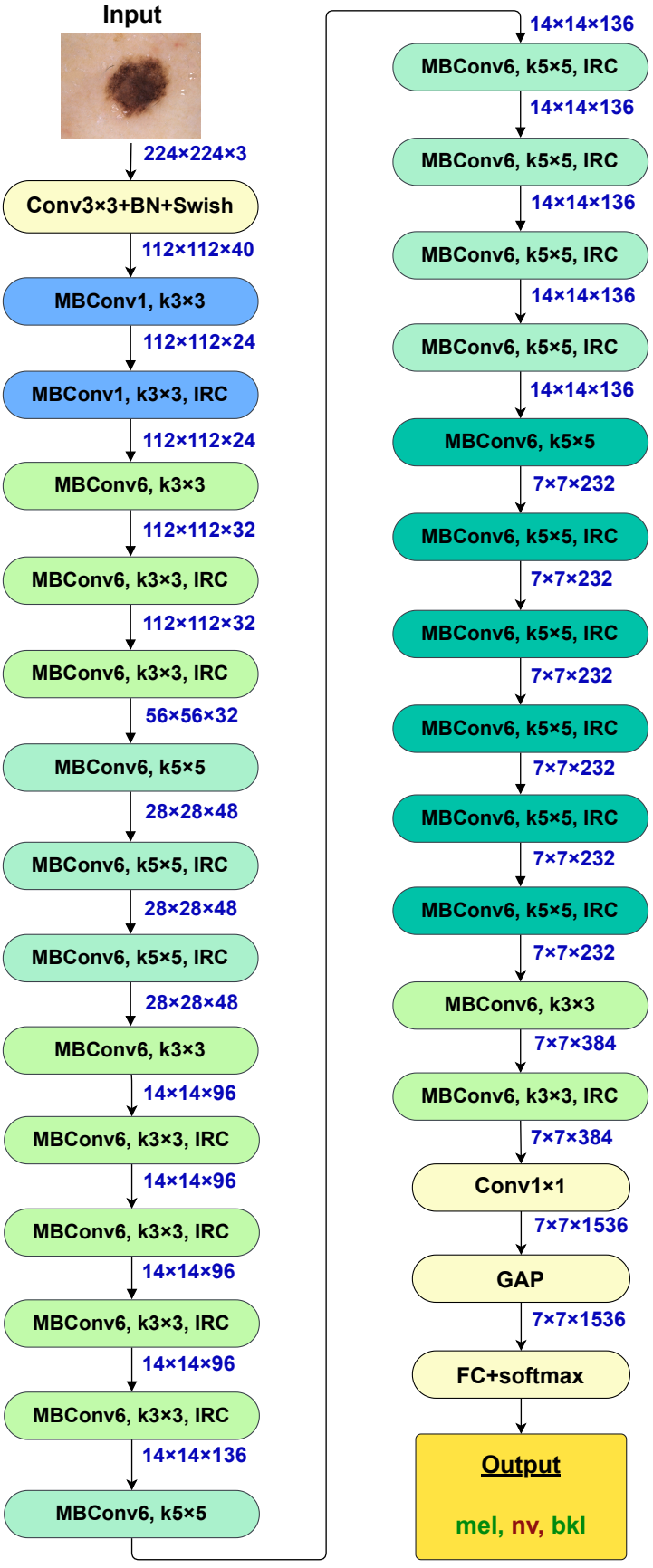


FIGURE 4.7: **An illustration of EfficientNet-B3 model architecture**

4.2 Proposed Ensemble Models

I have used two types of Ensemble methods : **4.2.1** contains Levy Stable Probability Density Function Based Ensemble Method, & **4.2.2** contains Beta Function Ranked-Based Ensemble Method

4.2.1 Levy Stable Probability Density Function Based Ensemble Method

The ensemble method I am using here is based on the Levy Stable probability density function. The PDF for Levy Stable is:

$$\mathcal{LS}(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \sigma(p)e^{-ixp}\,dp. \tag{4.1}$$

where $-\infty < p < \infty$ and the characteristics function for Levy Stable function is given as:

$$\sigma(p;\beta,\alpha,c,\mu) = e^{\left(ip\mu - |cp|^{\beta}(1-i\alpha\,\mathrm{sgn}(p)\Phi(\beta,p))\right)} \tag{4.2}$$

The Levy Stable probability density function is shown in Fig. **4.8**. In the study, I have used the Levy Stable function as Rank Estimator(RE) for class c. For N number of models, Rank Estimator is defined as,

$$RE_c^N = \mathcal{LS}(\sum_{i=2}^{N} CS_c^M) \tag{4.3}$$

Where CS is the confidence score of a particular model. Now, before calculating the final prediction I have to calculate the confidence factor sum (CFS), which is defined as,

$$CFS_c^N = \frac{1}{N}\sum_{i=2}^{N}(1 - CS_c)^i \tag{4.4}$$

The final Levy Stable Ensemble Confidence Score(LECS) is calculated as,

$$LECS_c^N = CFS_c^N \times RE_c^N \tag{4.5}$$

Lastly, for the input image X the final prediction is calculated as,

$$Prediction(X) = argmin_c(LECS_c^N) \tag{4.6}$$

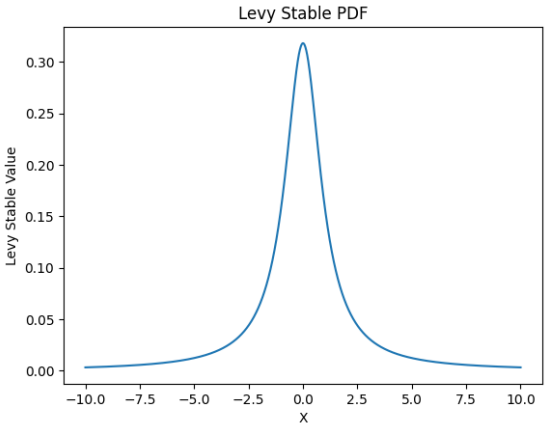


FIGURE 4.8: **Plot of Levy Stable PDF**

Algorithm 1 Model Optimization Process and Deployment

Require: $Input\ Model$
0: $Output\ IR\ File\ for\ the\ Targeted\ Device$
0: **if** $IR\ File\ ==\ True$: **then**
0: $Run\ Inference\ Engine$
0: $Deploy\ Model$
0: **else if** $IR\ File\ ==\ False$: **then**
0: $Fix\ Error$
0: **if** $IR\ File\ ==\ True$: **then**
0: $Run\ Inference\ Engine$
0: $Deploy\ Model$
0: **else if** $IR\ File\ ==\ False$: **then**
0: $Try\ Another\ Model$
0: **end if**
0: **end if**=0

4.2.2 Beta Function Ranked-Based Ensemble Method

In this paper, I have presented a confidence score fusion using the Beta function for the classification task. In mathematics, the Beta function is defined for two positive complex numbers $\mathcal{Z}_1$ and $\mathcal{Z}_2$.

Generally, the Beta function is defined as,

$$\mathcal{B}(\mathcal{Z}_1, \mathcal{Z}_2) = \int_0^1 p^{\mathcal{Z}_1 - 1}(1-p)^{\mathcal{Z}_2 - 1}\, dp \tag{4.7}$$

Where p is the positive real number in which the equation is being integrated, and $\mathcal{Z}_1, \mathcal{Z}_2$) are complex numbers or positive real numbers.

The Beta function is also written as,

$$\mathcal{B}(\mathcal{Z}_1, \mathcal{Z}_2) = \frac{\Gamma(\mathcal{Z}_1)\Gamma(\mathcal{Z}_2)}{\Gamma(\mathcal{Z}_1 + \mathcal{Z}_2)} \tag{4.8}$$

Where $\Gamma(.)$ is called the Gamma function. For a positive integer n Gamma is define as $\Gamma(n) = (n-1)!$, so for two different positive integer m and n the Beta function is written as,

$$\mathcal{B}(m, n) = \frac{(m-1)!\,(n-1)!}{(m+n-1)!} \tag{4.9}$$

There are two main properties for choosing the Beta function for confidence score fusion of multiple models.

- **Symmetry:** The Beta function is symmetric, so can be used for confidence score fusion and not affect the order of the confidence score(CS). The symmetry property of the Beta function is described as,

$$\mathcal{B}(m, n) = \mathcal{B}(n, m) \tag{4.10}$$

- **Multivariate Beta:** Beta function is used for more than two variables, so it can be effective for confidence score fusion of more than two models. Another advantage of the Beta function is, it is very easy to calculate the output for multivariate data, describe as,

$$\mathcal{B}(\mathcal{Z}_1, \mathcal{Z}_2, ..., \mathcal{Z}_n) = \frac{\Gamma(\mathcal{Z}_1)\Gamma(\mathcal{Z}_2)....\Gamma(\mathcal{Z}_n)}{\Gamma(\mathcal{Z}_1 + \mathcal{Z}_2) + ... + \mathcal{Z}_n)} \tag{4.11}$$

In the study, I have used the Beta function as Fuzzy Rank Estimator(FRE) for class c. For N number of models Fuzzy Rank Estimator is defined as,

$$FRE_c^N = \mathcal{B}(CS_c^1, CS_c^2, ..., CS_c^N) = \frac{\Gamma(CS_c^1)\Gamma(CS_c^2)....\Gamma(CS_c^N)}{\Gamma(CS_c^1 + CS_c^2 + ... + CS_c^N)} \tag{4.12}$$

$$FRE_c^N = \frac{\prod_{i=2}^{N}\Gamma(CS_c^M)}{\sum_{i=2}^{N} CS_c^M} \tag{4.13}$$

Now, before calculating the final prediction I have to calculate the confidence factor sum (CFS), which is defined as,

$$CFS_c^N = \frac{1}{N}\sum_{i=2}^{N}(1 - CS_c)^i \tag{4.14}$$

The final Beta Ensemble Confidence Score(BECS) is calculated as,

$$BECS_c^N = CFS_c^N \times FRE_c^N \tag{4.15}$$

Lastly, for the input image X the final prediction is calculated as,

$$Prediction = argmax_c(BECS_c^N) \tag{4.16}$$

4.3 Inferencing using OpenVINO

OpenVINO workflow consists of mainly three steps:
i) Model Preparation,
ii) Model Optimization and Compression, and
iii) Deployment.
First, the model is built using TensorFlow or Keras. Then the model is trained with data. If the accuracy is good enough it is sent to the model optimizer. So that it can optimize the model for particular target hardware. If the accuracy is not good then it is re-trained. After that, it is optimized by the model optimizer and gets converted into an intermediate representation (IR). If I do not get the IR file, the errors are being fixed or custom layers are being created. The IR file serves the purpose of conducting hardware inference for the deep learning model on specifically targeted hardware with the help of the Inference Engine. If the model inferencing is fast enough, the model is integrated into the pipeline for deployment. If not then advanced tuning of the model is being done. Now the model can be deployed across a single Intel hardware platform or a mix (heterogeneous execution) Intel hardware platform. Heterogeneous execution of a model can be done due to the Inference Engine. It is possible because various plug-ins are being used for various devices by the Inference Engine. The complete Workflow of

the OpenVINO toolkit including Model Optimizer and Inference Engine has been shown in Fig. **4.9**.

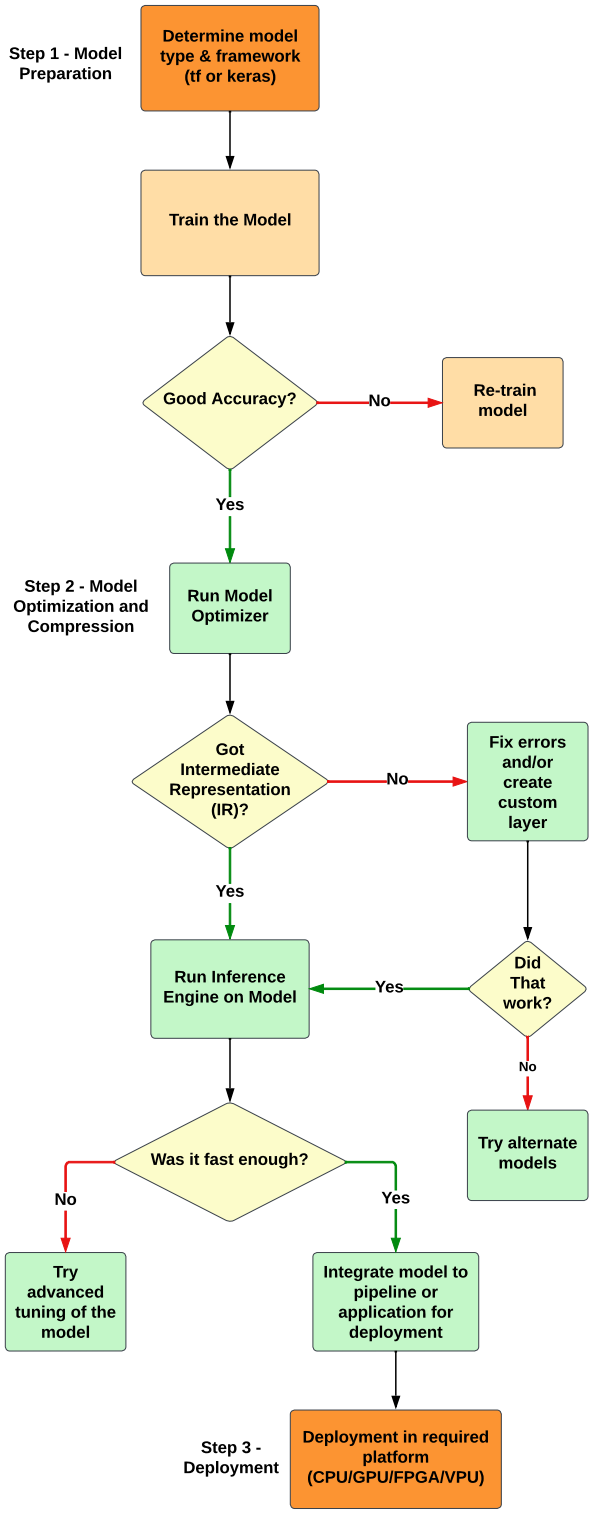


FIGURE 4.9: **Complete Workflow of OpenVINO toolkit including Model Optimizer and Inference Engine**

4.4 Entire Methodology

In this section, I will discuss the entire methodologies of two ensemble methods i.e; **4.4.1** contains Levy Stable PDF Ensemble, & **4.4.2** contains Beta function-based Ensemble.

4.4.1 Levy Stable PDF Ensemble

The methodology for implementing the Levy Stable ensemble method on the HAM10k dataset is as follows:

- HAM10k dataset: The HAM10k dataset, consisting of images related to skin cancer, is collected and used as the primary dataset for this study.
- Data Balancing: As the dataset may have class imbalances, I balanced the data by applying techniques such as oversampling or undersampling to ensure each class is represented adequately.
- Data Preprocessing: The dataset is divided into seven classes. Each class is separated, resampled to address any class imbalances, and then combined back into a single data frame.
- Image Loading: Based on the image IDs from the metadata, the images are read and associated with their respective classes.
- Data Conversion: The column of images in the data frame is converted into a numpy array, allowing for efficient data manipulation and processing.
- Data Normalization: To ensure consistency and comparability across images, data normalization is applied. This process involves scaling the pixel values to a common range, such as [0, 1] or [-1, 1].
- Data Split: The dataset is divided into training, validation, and testing sets. In this case, 90% of the data is used for training, 5% for validation, and 5% for testing.
- Data Resizing: The images are resized to a specific size, such as 32x32 pixels with three color channels (RGB).

- Model Selection: Two specific models, MobileNet-V2 and EfficientNet-B0, are chosen for this study. Each model is taken separately and customized for the task of skin cancer classification.
- Model Architecture: For each model, a Global Average Pooling layer and a dense layer are added to capture spatial information and enable classification. Dropout layers are introduced to mitigate overfitting issues.
- Model Training: The models are individually trained using the HAM10k dataset. The Adam optimizer is used, with a learning rate set to 0.0001. The training is performed for 200 epochs with a batch size of 128.
- Levy Stable PDF-based Ensemble: Once the individual model training is completed, the Levy Stable ensemble method is applied. This method leverages the properties of Levy Stable distributions, such as symmetricity and the ability to handle multiple networks. The ensemble combines the predictions from the individual models to generate a final ensemble prediction.
- OpenVINO Implementation: After performing the ensemble, the models are optimized for real-time deep-learning inference using the OpenVINO toolkit. The models undergo the model optimizer, which further optimizes the model for deployment on Intel hardware. This optimization process generates Intermediate Representation (IR) files, including .xml files that contain the network structure and .bin files that store the weights and biases in binary format. The IR files are then utilized by the Inference Engine in the OpenVINO toolkit to perform efficient inferencing on Intel hardware. Here I have used Intel i7 CPU and Intel iRIS Xe GPU. I separately deployed MobileNet-V2 and EfficientNet-B0 as well. After the deployment, I calculated the inference time for each time and compared them with the without using OpenVino mode. Without using Openvino I deployed the models on Intel i7 CPU and Google Tesla T4 GPU.

In summary, the methodology involves data preprocessing, model training, ensemble generation using the Levy Stable method, and deployment on Intel hardware using the OpenVINO toolkit for real-time inference. The OpenVINO implementation ensures efficient utilization of hardware resources and enhances the speed and performance of the skin cancer classification model. Figure **4.10** shows the overall pipeline of the Levy Stable function-based ensemble of CNN models used for the classification of skin cancer using dermatoscopic images.

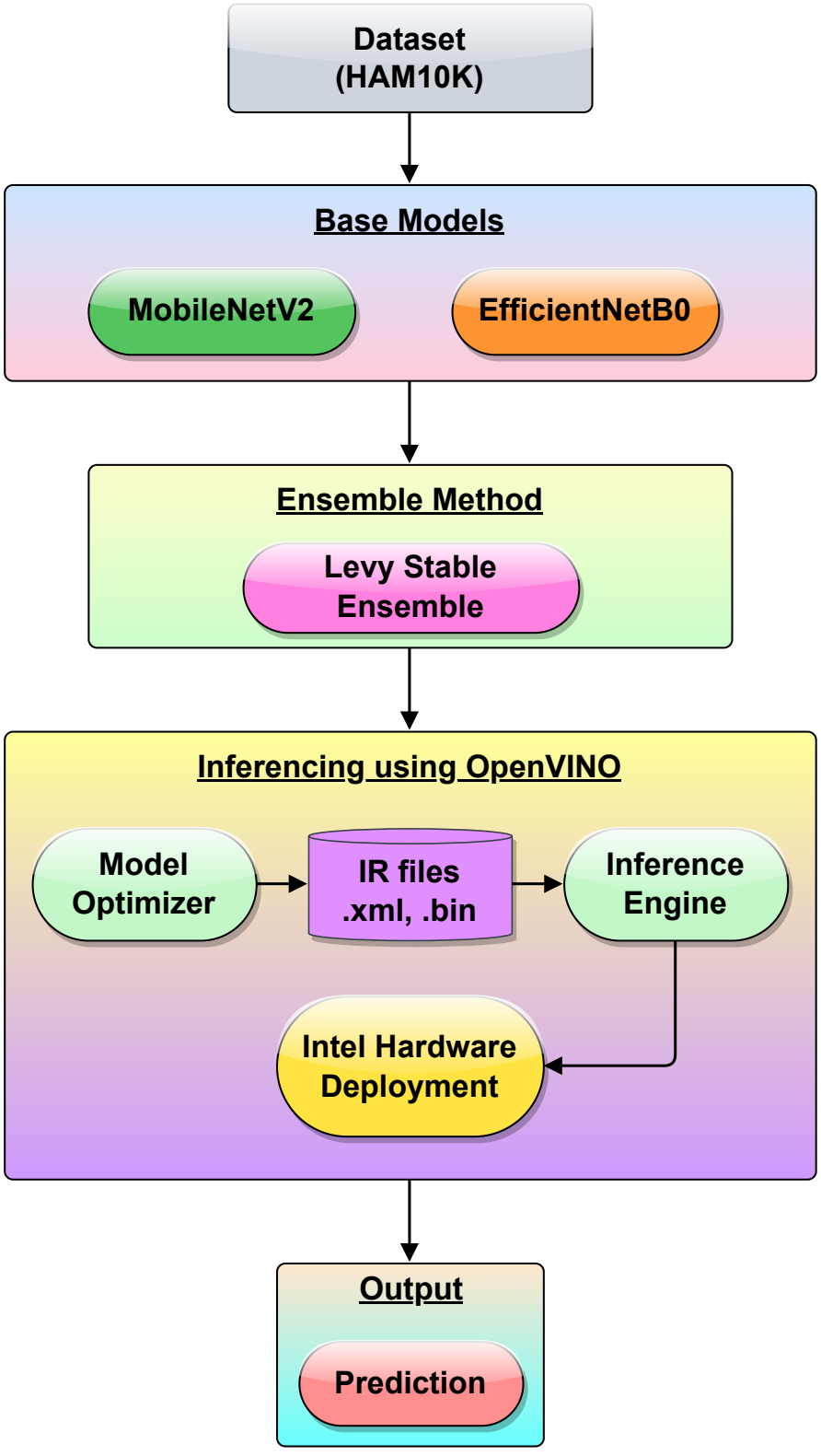


FIGURE 4.10: **Entire workflow of Proposed Levy Stable Ensemble Method**

4.4.2 Beta function-based Ensemble

The methodology for implementing the Beta Function based ensemble method on the HAM10k dataset is as follows:

- Network Models: I selected five basic network models, namely ResNet-101, Densenet-121, EfficientNet-B3, Inception-V3, and MobileNet-V3-Large, for my study.
- Individual Model Training: Each model was individually trained using the HAM10K dataset, which consists of 7 classes. I specifically focused on three classes: Melanocytic Nevi, Melanoma, and Benign keratosis-like lesions, as they had the highest number of images.
- Dataset Split: The dataset was divided into two sections, with 90% of the data used for training the models and the remaining 10% for testing.
- Data Normalization: I performed data normalization to ensure consistency and comparability among the images. The input size was set to 224 x 224 pixels.
- Training Parameters: The Adam optimizer was employed during model training, with a learning rate of 0.001. The models were trained for 50 epochs.
- Model Compilation and File Generation: After training, the models were compiled and fitted, resulting in the generation of .h5 files that store the weights and biases of the trained models for future use.
- Beta Function-based Ranked Ensemble: Following training, I applied the Beta function-based ranked ensemble method on two or three networks simultaneously. The multivariate properties of the Beta function facilitated the fusion of more than two networks seamlessly. The ensemble was performed in two categories: (I) with a softmax layer applied at the final layer and (II) without a softmax layer.
- Ensemble Implementation: Numpy and Math, two Python modules, were utilized during the Beta ensemble process.

- OpenVINO Deployment: The trained ensemble model was deployed using the OpenVINO toolkit for real-time deep-learning inferencing. The model underwent optimization through the model optimizer, resulting in the creation of Intermediate Representation (IR) files. The IR files included the .xml file, representing the network structure, and the .bin file, containing the weights and biases in binary format.

- Inference Engine and Hardware Deployment: The IR files were processed by the Inference Engine and made ready for deployment on Intel hardware. Two types of compression, fp16, and fp32, were applied during inferencing using OpenVINO. The chosen devices for deployment were Intel i7 CPU and Intel iRIS Xe graphics. Inferencing time was measured for both CPU and GPU, both with and without OpenVINO.

- Model Outputs and Evaluation: The output of the model provided the predicted class of cancer. Additionally, I obtained the confusion matrix and performance metrics, including accuracy, precision, recall, and f1-score. The inferencing time was measured in frames per second (FPS).

In summary, my study involved training five basic network models individually using the HAM10K dataset. I focused on three classes with the highest number of images. The models were trained using data normalization, and the input size was set to 224 x 224 pixels. After training, I applied the Beta function-based ranked ensemble method on two or three networks simultaneously, leveraging the multivariate properties of the Beta function. The ensemble was performed with and without a softmax layer at the final layer. I deployed the trained ensemble model using the OpenVINO toolkit for real-time deep-learning inferencing, optimizing the model, and generating IR files. The deployment was done on Intel i7 CPU and Intel iRIS Xe graphics, and the inferencing time was measured for both CPU and GPU, with and without OpenVINO. The model provided the predicted class of cancer, along with performance metrics and inferencing time.
Figure **4.11** shows the overall pipeline of the Beta function-based ensemble of CNN models used for the classification of skin cancer using dermatoscopic images. Any two or three base models' confidence scores are fused together based on the Beta function at a single time to create an Ensemble Model.

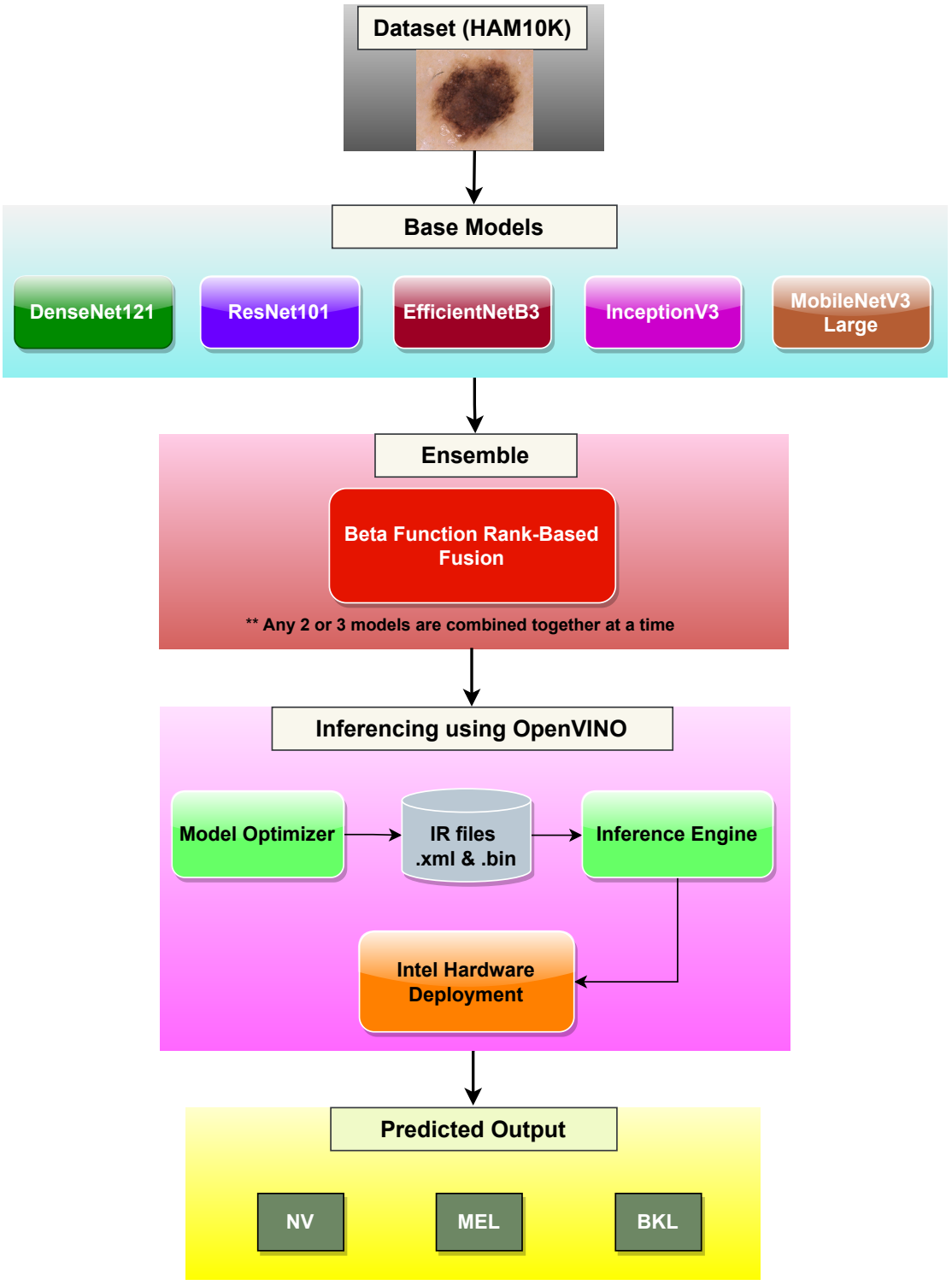


FIGURE 4.11: **Entire workflow of Proposed Beta Ensemble Method**

Chapter 5

EXPERIMENTAL SET UP AND RESULT

This chapter contains five sections, 5.1 contains specifications of the system and software, 5.2 contains details of the used HAM10K dataset, 5.3 contains the discussion of Performance Metrics, 5.4 contains the results of basic CNN models and two ensemble methods, and 5.5 contains the result analysis of the basic CNN models and the ensemble methods.

5.1 Specification

This section consists of two subsections, 5.1.1 contains system specifications and 5.1.2 contains the software specifications.

5.1.1 System Specification

In this thesis, I have used Google Colab to train and test various DL models used in this project. Google Colab or Google Colaboratory is a tool used to data analysis and machine learning purposes. It permits a user to create a unique Google Drive document that contains executable Python code, rich text, charts, images, HTML, LaTeX, and more.

- Google Colab GPU or CPU runtime can be used cost-freely for 6 to 12 hours.
- The Intel Xeon CPU running at 2.20 GHz, 13 GB of RAM, the Tesla K80 accelerator, and 12 GB of GDDR5 VRAM make up the GPU runtime.
- An Intel Xeon CPU running at 2.30 GHz, 13 GB of RAM, and a cloud TPU with 180 teraflops of processing power make up the Tensor Processing Unit (TPU) runtime.

5.1.2 Software Specification

I have used Python as the programming language. Python has a very vast implementation area. Many existing Python modules or libraries are there for different fields to make the work easy. Some of the very popular and famous Python libraries or modules that have been used here are Pandas, Numpy, Matplotlib, Tensorflow, Keras, scipy, os etc.

Let us have a brief introduction to a few popular libraries.

- **TensorFlow:**
  A freely available software package called TensorFlow is used for information flow and distinguishable programming on a variety of tasks. It is primarily used for ML and DL applications. Additionally, it can be utilized for a huge range of numerical calculations.

  The Google Brain team's researchers and engineers created TensorFlow, and it was initially released as an open-source project in November 2015. Since then, it has grown to be one of the most well-known ML libraries, with a sizable and vibrant community of users and developers. TensorFlow is built around the concept of computational graphs, which are directed acyclic graphs that describe the computations performed by a machine learning model. These graphs represent the flow of data through the model, as well as the transformations and operations that are applied to the data.

  One of the main benefits of TensorFlow is its capability to efficiently perform computations on large datasets. It uses highly optimized C++ code for its core operations, and it can also take advantage of hardware acceleration through the use of GPUs and TPUs (Tensor Processing Units). TensorFlow offers a variety of high-level APIs, including Keras, a well-liked neural network building API, for creating and training ML models. It also provides a lower-level API for more advanced users, which allows for greater flexibility and control over the model architecture and training process. Its efficient computational graph framework and support for hardware acceleration make it an excellent choice for large-scale machine-learning applications. TensorFlow has various applications, including computer vision, NLP, speech recognition, and more. It is used by many organizations and industries, including the monetary, retail, and medical sectors.

- **Keras:**
  A high-level open-source software library called Keras is used to create and train deep neural networks. It was initially developed as a user-friendly interface to build models added to other popular DL frameworks such as TensorFlow, and CNTK. But later has become an integral part of the TensorFlow

ecosystem. Keras provides an easy and user-friendly API that allows developers to quickly create and prototype DL models. It enables easy customization and experimentation on model architectures, layers, loss functions, activation functions, regularization techniques, optimizers, and metrics, among others. Keras has a modular architecture that allows users to create complex DL models by combining layers in different ways. It also supports both sequential and functional APIs to allow for building models with either a linear stack of layers or more complex graphs of layers respectively. One of the key features of Keras is its flexibility and portability across different platforms and frameworks. It allows seamless switching of backends between TensorFlow, Theano, and CNTK, depending on the user's preference. Keras also supports distributed training across multiple GPUs and CPUs, making it easy to scale up models to handle larger datasets and more complex tasks. Keras has a large and active community of developers and users who contribute to its development and improvement. It also has comprehensive documentation with a variety of tutorials, examples, and best practices to help users get started with building and training deep learning models using Keras. It has been used in a wide range of applications, including image and speech recognition, natural language processing, and recommendation systems, among others. It is also used in academic research and industrial applications. In summary, Keras is a high-level and user-friendly deep learning library that enables fast prototyping and experimentation of complex neural network architectures. Its modular and flexible architecture, support for multiple backends, and extensive documentation make it an excellent choice for deep learning applications.

- **NumPy:**
  Numpy is a library for scientific computing in Python. It provides a high-performance multidimensional array object, as well as a wide range of mathematical functions for working with arrays. NumPy is used by many scientific computing libraries, including SciPy, Matplotlib, and Pandas.

- **SciPy:**
  SciPy is a powerful library that can be used for a wide variety of scientific computing tasks. It includes modules for linear algebra, numerical integration, optimization, and statistics. SciPy is a popular choice for scientific

research and development, and it is also used in a variety of commercial applications.

- **Pandas:**
  Pandas is a library for data analysis in Python. It provides high-performance, easy-to-use data structures and data analysis tools for the labeled data in Python programming language. Pandas stand for Python Data Analysis Library. Pandas can be used for data wrangling, cleaning, and analysis. It can also be used to create visualizations of data.

- **Matplotlib:**
  Matplotlib is a library for creating static, animated, and interactive visualizations in Python. It is a popular choice for data visualization in Python, and it is used by many data scientists and machine learning practitioners. Matplotlib can be used to create a wide variety of visualizations, including line charts, bar charts, pie charts, and scatter plots.

- **OpenCV:**
  OpenCV is a library for computer vision in Python. It provides a wide range of functions for working with images and videos, including image processing, object detection, and face recognition. OpenCV is a popular choice for computer vision applications, and it is used by many companies, including Google, Facebook, and Microsoft.

- **Os:**
  The os module is a low-level module that provides access to the operating system. It can be used to do things like create and delete files, read and write to files, and list the contents of directories. The os module is a necessary part of any Python program that needs to interact with the operating system.

- **Confusion Matrix:**
  A confusion matrix is a useful tool for evaluating the performance of a classification model. It shows the number of instances that were correctly classified and the number of instances that were incorrectly classified. The confusion matrix can be used to calculate a variety of metrics, such as accuracy, precision, and recall.

5.2 Dataset Used

The dataset that has been used for my work is HAM10K (**Tschandl** [**2018**]). It is a sizable collection of dermoscopic photographs from numerous sources of pigmented skin lesions, which are relatively prevalent. It is used to train and evaluate ML algorithms for the detection of skin cancer. The dataset was created by a team of researchers from the Medical University of Vienna, Austria, namely Tschandl, P., Rosendahl, C. and Kittler, H. (**Tschandl** [**2018**]), and was made publicly available in 2018. The HAM10k dataset consists of 10,015 images. These images were gathered over a period of 20 years. The majority of these images were obtained from two distinct locations: the Department of Dermatology at the Medical University of Vienna in Austria, and Cliff Rosendahl's skin cancer clinic in Queensland, Australia. This dataset includes seven types of skin diseases, including Melanocytic Nevi(NV), Melanoma(MLE), Basal Cell Carcinoma(BCC), Actinic Keratoses(Akiec), Dermatofibroma(DF), Benign Keratosis-like Lesions(BKL), and Vascular Lesions(Vasc). Melanocytic nevi have the largest number of images, and dermatofibroma has the smallest number. The number of images has been shown in Table 5.1.

TABLE 5.1: **Disease Names and Respective Number of Images**

Disease Name	Number of Images
NV	6705
MEL	1113
BKL	1099
BCC	514
Akiec	327
Vasc	142
DF	115

This dataset consists of 400×600 pixel images in JPEG format. The HAM10k dataset also includes metadata for each image, which provides several details on patients such as age, sex, and location of the suspicious skin on the body. This metadata can be used to study the relationship between skin cancer and different demographic and environmental factors.

The HAM10k dataset has been used in several studies in order to assess the effectiveness of ML algorithms for skin cancer diagnosis. These studies have shown that

deep learning algorithms trained on the HAM10k dataset can accurately distinguish between different skin lesions. For example, a study published in the journal Nature in 2018 reported an accuracy of 91% in distinguishing between melanoma and benign nevi using a deep learning algorithm trained on the HAM10k dataset.

The HAM10k dataset is a valuable resource for researchers and clinicians working on diagnosing and treating skin cancer. The dataset contains several dozen high-quality pictures of skin conditions, along with metadata that can be used for further analysis. The HAM10k dataset has already been used in several studies, demonstrating its usefulness in developing and evaluating ML algorithms for skin cancer diagnosis.

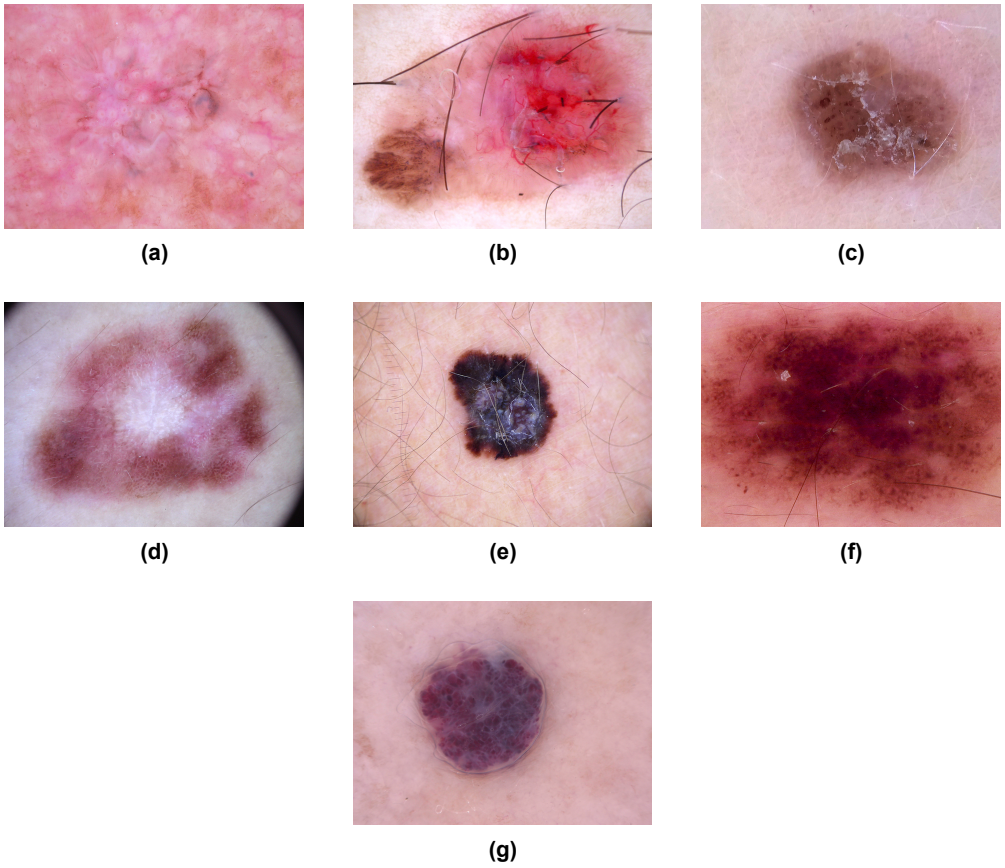


FIGURE 5.1: **Class-wise sample images from HAM10K dataset: (a)Akiec, (b)BCC, (c)BKL, (d)DF, (e)MEL, (f)NV, (g)Vasc**

5.3 Performance Metrics

A classifier network developed using any machine learning technique should be evaluated in order to determine how well it is doing. To determine the usefulness of the ML model evaluation matrices are needed. For a binary classification task, a confusion matrix can be represented as follows:

	Predicted_Negative	**Predicted_Positive**
Original_ Negative	TN	FP
Original_Positive	FN	TP

Four potential outcome categories—true positive (TP) for accurately predicted positives, false positive (FP) for inaccurately predicted positives, true negative (TN) for accurately predicted negatives, and false negative (FN) for incorrectly predicted negatives—are produced as a result of comparing the predicted outcomes with a ground truth table. These results may be examined, and the performance metrics can then be calculated. Many performance measures (**Botchkarev [2019]**) can be obtained from the confusion matrix. Some of them are Accuracy, Error Rate, Precision, Recall, Specificity, AUC, Positive Predictive Value, F1-score, MCC (Matthews Correlation Coefficient), Kappa, False Discovery Rate, F2-score, False Positive Rate, False Negative Rate, etc.

Here I am mainly focusing on Accuracy, Error Rate, Precision, Recall or Sensitivity or True Positive Rate, Specificity or True Negative Rate, Under the Curve, Positive Predictive Value, and F1-score. I am considering these evaluation metrics because the papers I have reviewed most of them have used these evaluation matrices. Along with this, these metrics are being used most commonly and widely.

Accuracy (ACC): Accuracy is a measure of overall correctness, considering both true positives and true negatives.

$$ACC = \frac{(TN + TP)}{(TN + FN + FP + TP)} \times 100\%$$

Error Rate (ER): the proportion of inaccurately predicted samples to all test samples.

$$ER = \frac{(FN + FP)}{(TN + FN + FP + TP)}$$

Precision (PRE): It measures the ability of a model to correctly classify positive instances, minimizing the occurrence of false positives.

$$PRE = \frac{TP}{(TP + FP)}$$

Recall or Sensitivity (SEN) : It reflects a model's ability to correctly identify positive instances from the actual positive samples.

$$SEN = \frac{TP}{(TP + FN)}$$

Specificity (SPC): The proportion of samples with TN to the sum of TN and FP.

$$SPC = \frac{TN}{(TN + FP)}$$

F1-score (F1): It is calculated as the harmonic mean of recall and precision. It combines precision and recall into a single metric, and provides an overall measure of a model's performance. Its highest possible value is 1.

$$F1 = 2 \times \frac{(PRE \times SEN)}{(PRE + SEN)}$$

5.4 Result

This result section is divided into two subsections. **5.4.1** contains the results of Levy Stable ensemble methods, and **5.4.2** contains the results of Beta function ensemble methods.

5.4.1 Levy Stable Based Ensemble

This section is divided into three subsections. **5.4.1.1** contains the comparison table of Performance Metrics of Basic CNN Models (MobileNet-V2 and EfficientNet-B0) & Levy Stable Ensemble Model. **5.4.1.2** contains the Inference time of Basic CNN Models (MobileNet-V2 and EfficientNet-B0) & Levy Stable Ensemble Model on Hardware (CPU and GPU) with and without Open-VINO. And **5.4.1.3** contains Confusion Matrices of two basic models and proposed Levy Stable Ensemble model.

5.4.1.1 Comparison of Performance Metrics

TABLE 5.2: **Comparison Table of Performance Metrics of Basic CNN Models (MobileNet-V2 and EfficientNet-B0) & Levy Stable Ensemble Model**

Model_Name	Accuracy	Precision	Recall	F1-Score
MobileNetV2	82	82	83	82
EfficientNetB0	77	77	78	77
Ensemble of EfficientNetB0 & MobileNetV2	83	83	83	83

5.4.1.2 Inference Time

TABLE 5.3: **Inference Time of Basic CNN Models (MobileNet-V2 and EfficientNet-B0) & Levy Stable Ensemble Model on Hardware (CPU and GPU) with and without OpenVINO (results are in FPS)**

Model_Number & Model_Name	Inferencing time with OpenVINO (FPS)				Inferencing time without OpenVINO (FPS)	
	FP16		FP32			
	CPU	GPU	CPU	GPU	CPU	GPU
1. MobileNetV2	432.60	343.44	373.94	310.46	11.85	12.50
2. EfficientNetB0	160.00	191.23	211.31	171.17	5.78	6.54
3. Levy Stable Ensemble of MobileNetV2 & EfficientNetB0	110.31	93.31	91.12	65.20	3.98	4.27

5.4.1.3 Confusion Matrix

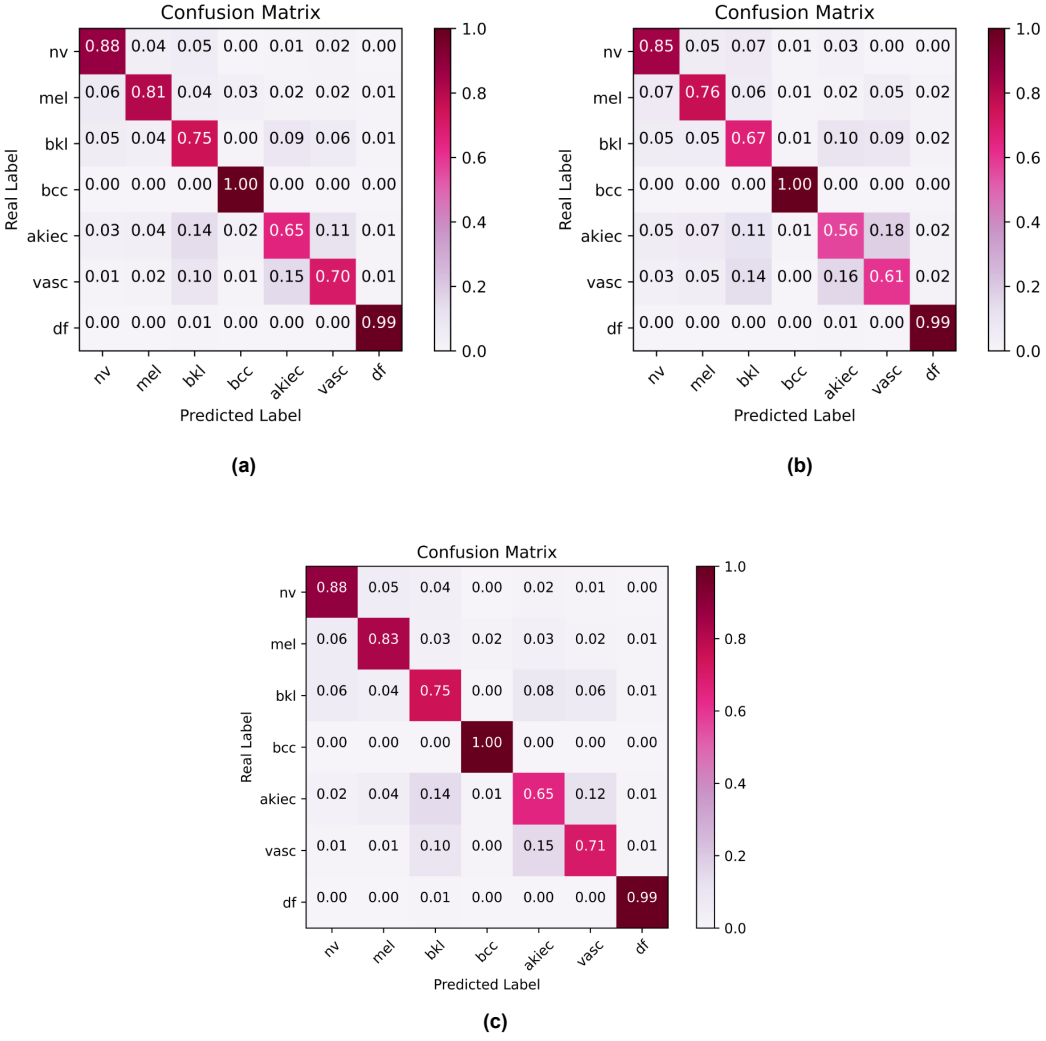


FIGURE 5.2: **Confusion Matrices of two basic models and the proposed Levy Stable Ensemble model: (a) MobileNet-V2, (b) EfficientNet-B0, (c) Levy Stable Ensemble of MobileNet-V2 & EfficientNet-B0**

5.4.2 Beta Function based Ensemble

This section has three subsections, **5.4.2.1** contains results of Basic CNN models, along with the Accuracy graph Fig. **5.3** and Error graph Fig. **5.4**. Subsection **5.4.2.2**, and **5.4.2.3** contain the results of the Ensemble of 2 basic CNN models, and the Ensemble of 3 basic CNN models, respectively.

5.4.2.1 Basic CNN Models

TABLE 5.4: **Comparison of Precision, Recall, and F1-score of Class-0 of Basic CNN Models**

Model_Name	Precision	Recall	F1-score
ResNet-101	0.16	0.15	0.15
DenseNet-121	0.15	0.13	0.14
EfficientNet-B3	0.13	0.12	0.12
InceptionNet-V3	0.13	0.12	0.12
MobileNet-V3-Large	0.16	0.10	0.12

TABLE 5.5: **Comparison of Precision, Recall, and F1-score of Class-1 of Basic CNN Models**

Model_Name	Precision	Recall	F1-score
ResNet-101	0.16	0.12	0.14
DenseNet-121	0.05	0.03	0.04
EfficientNet-B3	0.20	0.16	0.18
InceptionNet-V3	0.08	0.06	0.07
MobileNet-V3-Large	0.33	0.01	0.02

TABLE 5.6: **Comparison of Precision, Recall, and F1-score of Class-2 of Basic CNN Models**

Model_Name	Precision	Recall	F1-score
ResNet-101	0.75	0.78	0.76
DenseNet-121	0.73	0.79	0.76
EfficientNet-B3	0.75	0.79	0.77
InceptionNet-V3	0.73	0.78	0.76
MobileNet-V3-Large	0.74	0.91	0.82

Accuracy graph

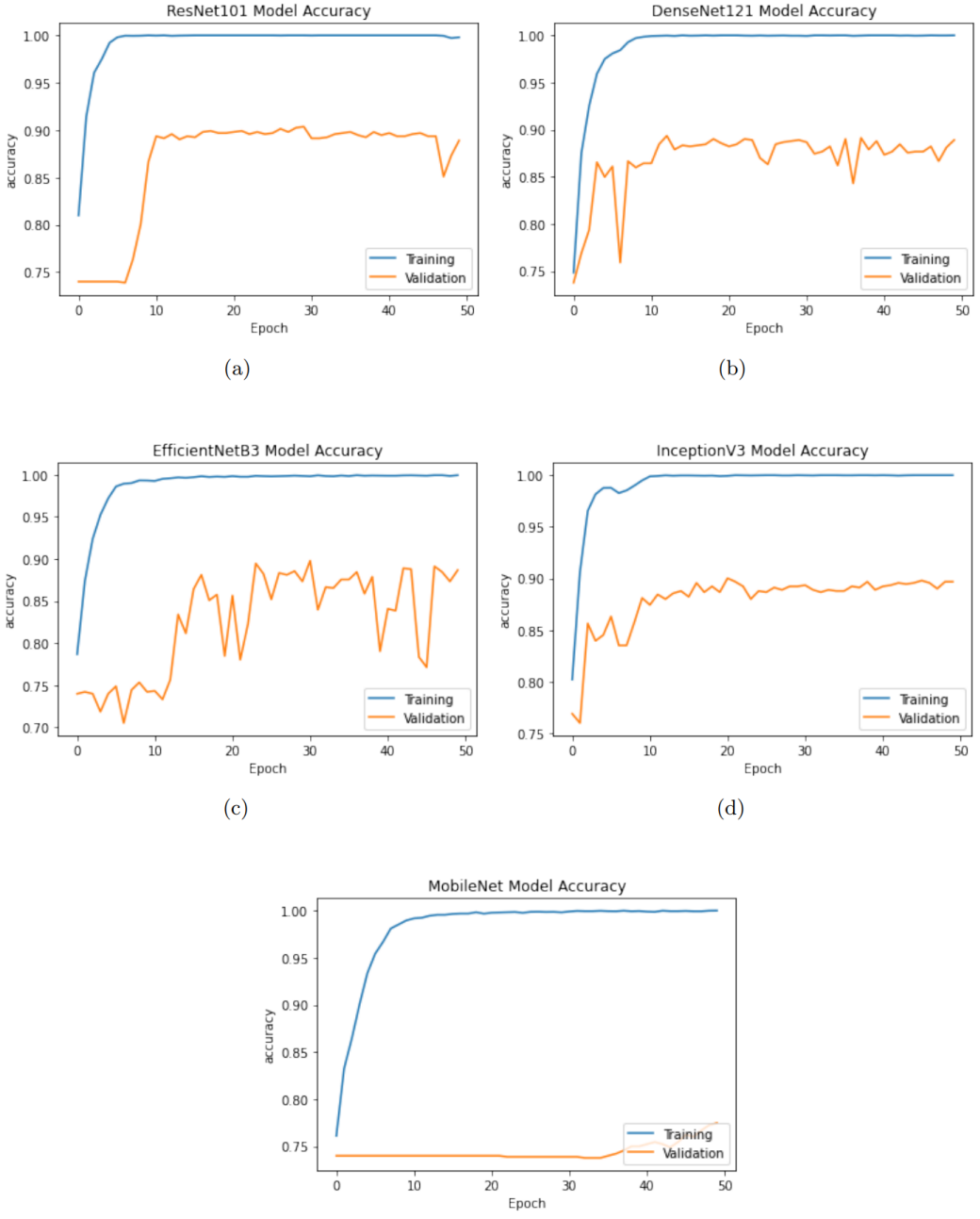


FIGURE 5.3: **Training and Validation Accuracy Graph of Basic CNN Models: (a)ResNet-101, (b)DenseNet-121, (c)EfficientNet-B3, (d)InceptionNet-V3, (e)MobileNet-V3-Large**

TABLE 5.7: **Comparison of Precision, Recall, and F1-score (along with Micro Average and Weighted Average) of Basic CNN Models**

SL. No.	Model_Name	Precision	Recall	F1-score	Accuracy	Micro_Average			Weighted_Average		
						Precision	Recall	F1-score	Precision	Recall	F1-score
1	ResNet-101	0.36	0.35	0.35	0.61	0.35	0.35	0.35	0.59	0.61	0.60
2	DenseNet-121	0.31	0.32	0.31	0.61	0.31	0.32	0.31	0.57	0.61	0.58
3	EfficientNet-B3	0.36	0.36	0.36	0.62	0.36	0.36	0.36	0.60	0.62	0.61
4	InceptionNet-V3	0.31	0.32	0.32	0.60	0.31	0.32	0.32	0.57	0.60	0.58
5	MobileNet-V3-Large	0.41	0.34	0.32	0.69	0.41	0.34	0.32	0.61	0.69	0.62

Error Graph

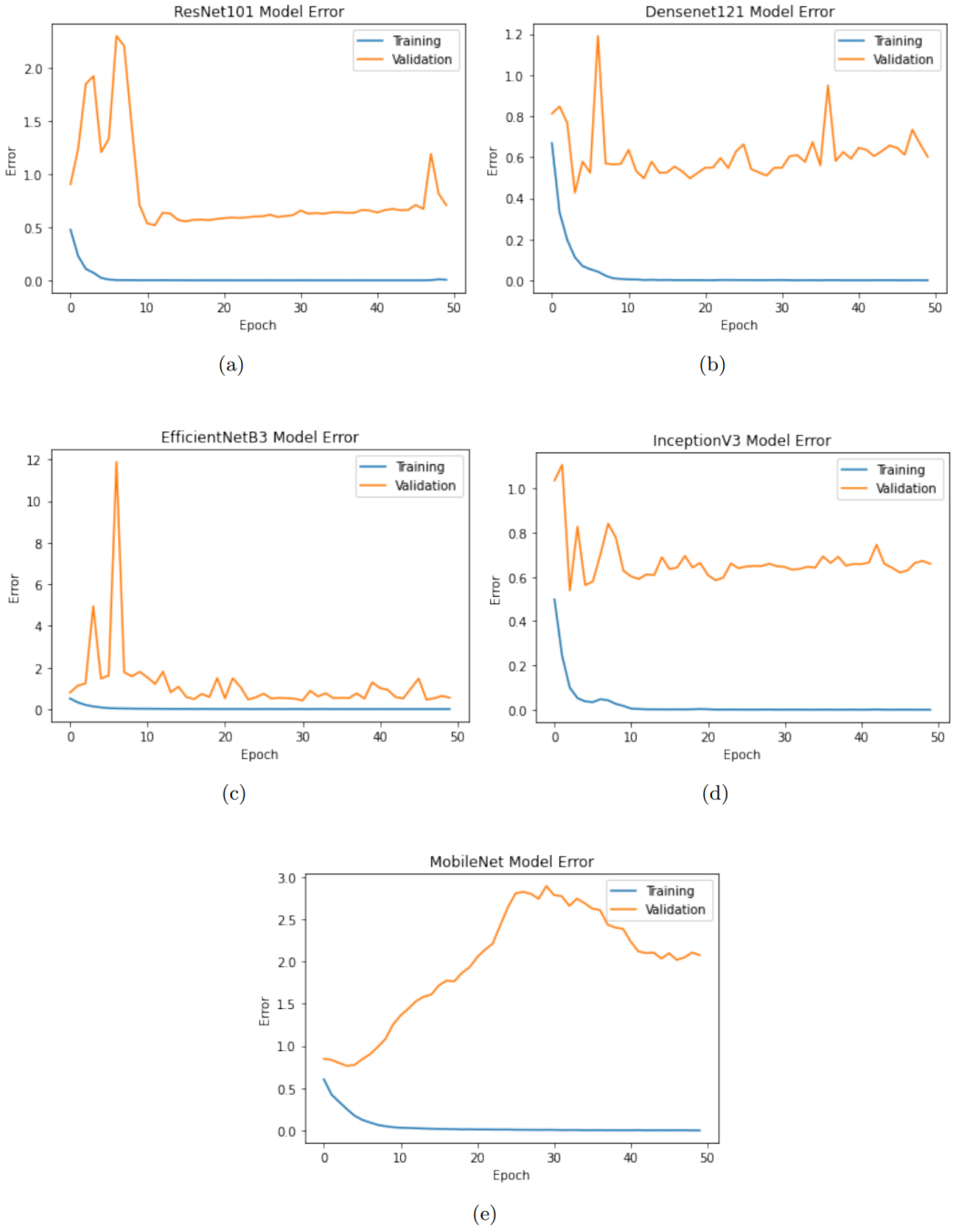


FIGURE 5.4: **Training and Validation Error Graph of Basic CNN Models: (a)ResNet-101, (b)DenseNet-121, (c)EfficientNet-B3, (d)InceptionNet-V3, (e)MobileNet-V3-Large**

Confusion Matrix

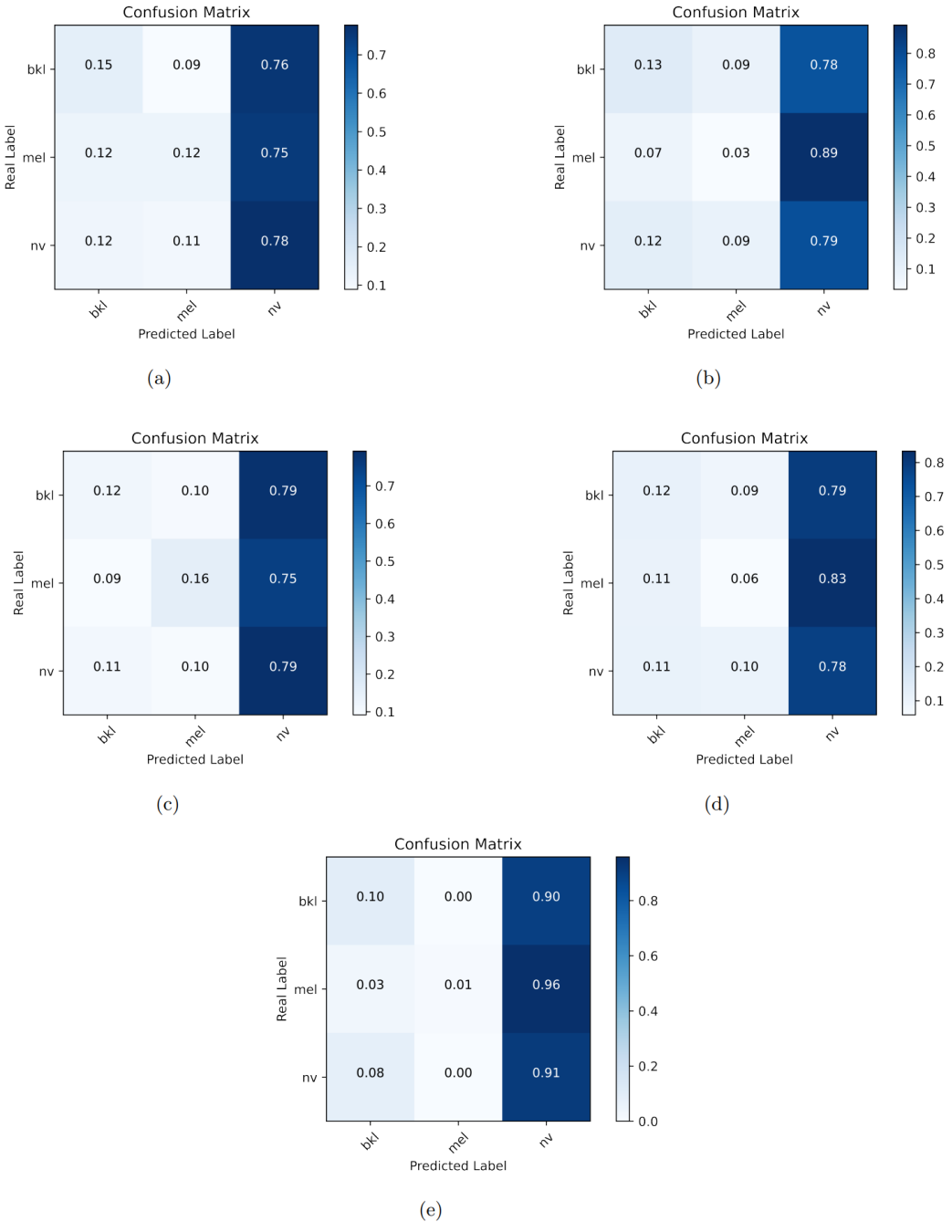


FIGURE 5.5: **Confusion Matrix of Basic CNN Models: (a)ResNet-101, (b)DenseNet-121, (c)EfficientNet-B3, (d)InceptionNet-V3, (e)MobileNet-V3-Large**

TABLE 5.8: **Inference Time of Basic CNN Models on Hardware (CPU and GPU) with and without OpenVINO (results are in FPS)**

Model No.	Model_Name	Inferencing time with using OpenVINO (FPS)				Inferencing time without OpenVINO (FPS)	
		FP16		FP32			
		CPU	GPU	CPU	GPU	CPU	GPU
1	ResNet-101	7.44	63.38	6.98	44.15	1.33	4.99
2	DenseNet-121	16.39	68.63	14.40	53.82	1.90	4.22
3	EfficientNet-B3	19.61	76.52	20.51	66.01	2.00	3.75
4	InceptionNet-V3	18.93	85.84	18.24	69.38	3.21	5.67
5	MobileNet-V3-Large	131.87	141.07	116.94	122.86	5.52	8.09

5.4.2.2 Ensemble of 2 Basic CNN Models

TABLE 5.9: **Comparison of Precision, Recall, and F1-score of Class-0 of Ensemble of 2 Basic CNN Models**

Model_Name	Precision	Recall	F1-score
Resnet_MobileNet	0.70	0.76	0.73
Resnet_MobileNet_SM	0.75	0.73	0.74
Resnet_DenseNet	0.86	0.78	0.82
Resnet_DenseNet_SM	0.84	0.78	0.81
Resnet_EfficientNet	0.82	0.72	0.77
Resnet_EfficientNet_SM	0.84	0.73	0.78
Resnet_InceptionNet	0.88	0.76	0.81
Resnet_InceptionNet_SM	0.85	0.77	0.81
MobileNet_EfficientNet	0.70	0.65	0.68
MobileNet_EfficientNet_SM	0.71	0.58	0.64
MobileNet_DenseNet	0.64	0.70	0.67
MobileNet_DenseNet_SM	0.71	0.62	0.66
MobileNet_InceptionNet	0.68	0.71	0.70
MobileNet_InceptionNet_SM	0.70	0.62	0.66
EfficientNet_DenseNet	0.88	0.76	0.81
EfficientNet_DenseNet_SM	0.89	0.75	0.82
EfficientNet_InceptionNet	0.90	0.75	0.82
EfficientNet_InceptionNet_SM	0.89	0.74	0.81
DensetNet_InceptionNet	0.87	0.75	0.80
DenseNet_InceptionNet_SM	0.89	0.77	0.82

TABLE 5.10: **Comparison of Precision, Recall, and F1-score of Class-1 of Ensemble of 2 Basic CNN Models**

Model_Name	**Precision**	**Recall**	**F1-score**
Resnet_MobileNet	0.88	0.18	0.30
Resnet_MobileNet_SM	0.88	0.25	0.39
Resnet_DenseNet	0.78	0.60	0.68
Resnet_DenseNet_SM	0.80	0.60	0.69
Resnet_EfficientNet	0.82	0.62	0.70
Resnet_EfficientNet_SM	0.83	0.63	0.72
Resnet_InceptionNet	0.83	0.62	0.71
Resnet_InceptionNet_SM	0.84	0.62	0.71
MobileNet_EfficientNet	0.91	0.08	0.15
MobileNet_EfficientNet_SM	0.95	0.17	0.28
MobileNet_DenseNet	0.78	0.06	0.11
MobileNet_DenseNet_SM	0.89	0.20	0.33
MobileNet_InceptionNet	0.94	0.12	0.22
MobileNet_InceptionNet_SM	0.96	0.22	0.35
EfficientNet_DenseNet	0.86	0.58	0.70
EfficientNet_DenseNet_SM	0.86	0.60	0.71
EfficientNet_InceptionNet	0.87	0.60	0.71
EfficientNet_InceptionNet_SM	0.87	0.59	0.70
DensetNet_InceptionNet	0.87	0.55	0.67
DenseNet_InceptionNet_SM	0.87	0.57	0.69

TABLE 5.11: **Comparison of Precision, Recall, and F1-score of Class-2 of Ensemble of 2 Basic CNN Models**

Model_Name	**Precision**	**Recall**	**F1-score**
Resnet_MobileNet	0.87	0.98	0.92
Resnet_MobileNet_SM	0.87	0.99	0.93
Resnet_DenseNet	0.92	0.97	0.94
Resnet_DenseNet_SM	0.92	0.97	0.94
Resnet_EfficientNet	0.91	0.97	0.94
Resnet_EfficientNet_SM	0.92	0.98	0.95
Resnet_InceptionNet	0.92	0.98	0.95
Resnet_InceptionNet_SM	0.92	0.98	0.94
MobileNet_EfficientNet	0.84	0.98	0.90
MobileNet_EfficientNet_SM	0.83	0.99	0.90
MobileNet_DenseNet	0.86	0.99	0.92
MobileNet_DenseNet_SM	0.85	0.99	0.92
MobileNet_InceptionNet	0.86	0.99	0.92
MobileNet_InceptionNet_SM	0.85	0.99	0.92
EfficientNet_DenseNet	0.91	0.98	0.94
EfficientNet_DenseNet_SM	0.91	0.98	0.94
EfficientNet_InceptionNet	0.91	0.98	0.94
EfficientNet_InceptionNet_SM	0.91	0.98	0.94
DensetNet_InceptionNet	0.90	0.98	0.94
DenseNet_InceptionNet_SM	0.91	0.98	0.94

TABLE 5.12: **Comparison of Precision, Recall, and F1-score (along with Micro Average and Weighted Average) of Ensemble of 2 Basic CNN Models**

SL. No.	Model_Name	Precision	Recall	F1-score	Accuracy	Micro_Average			Weighted_Average		
						Precision	Recall	F1-score	Precision	Recall	F1-score
1	Res_Mobile	0.82	0.64	0.65	0.85	0.82	0.64	0.65	0.85	0.85	0.82
	Res_Mobile_SM	0.83	0.66	0.69	0.86	0.84	0.66	0.69	0.86	0.86	0.83
2	Res_Dense	0.85	0.78	0.81	0.90	0.85	0.78	0.81	0.89	0.90	0.89
	Res_Dense_SM	0.85	0.78	0.81	0.90	0.85	0.78	0.81	0.89	0.90	0.89
3	Res_Efficient	0.85	0.77	0.8	0.89	0.85	0.77	0.81	0.89	0.89	0.89
	Res_Efficient_SM	0.86	0.78	0.82	0.90	0.86	0.78	0.81	0.89	0.90	0.89
4	Res_Inception	0.88	0.79	0.82	0.90	0.87	0.78	0.82	0.90	0.9	0.90
	Res_Inception_SM	0.87	0.79	0.82	0.90	0.87	0.79	0.82	0.90	0.90	0.90
5	Mobile_Efficient	0.82	0.57	0.58	0.82	0.82	0.57	0.58	0.83	0.82	0.77
	Mobile_Efficient_SM	0.83	0.58	0.61	0.83	0.83	0.58	0.61	0.84	0.83	0.79
6	Mobile_Dense	0.76	0.58	0.57	0.83	0.76	0.58	0.57	0.82	0.83	0.78
	Mobile_Dense_SM	0.82	0.60	0.64	0.84	0.82	0.60	0.63	0.84	0.84	0.80
7	Mobile_Inception	0.83	0.61	0.61	0.84	0.83	0.61	0.61	0.85	0.84	0.80
	Mobile_Inception_SM	0.84	0.61	0.64	0.84	0.84	0.61	0.64	0.85	0.84	0.81
8	Efficient_Dense	0.88	0.77	0.82	0.90	0.88	0.77	0.82	0.90	0.90	0.89
	Efficient_Dense_SM	0.89	0.78	0.82	0.90	0.89	0.78	0.82	0.90	0.90	0.90
9	Efficient_Inception	0.89	0.78	0.82	0.90	0.89	0.78	0.82	0.90	0.90	0.90
	Efficient_Inception_SM	0.89	0.77	0.82	0.90	0.89	0.77	0.82	0.90	0.90	0.89
10	Dense_Inception	0.88	0.76	0.8	0.90	0.88	0.76	0.81	0.89	0.90	0.89
	Dense_Inception_SM	0.89	0.77	0.82	0.90	0.89	0.78	0.82	0.90	0.90	0.89

Confusion Matrix

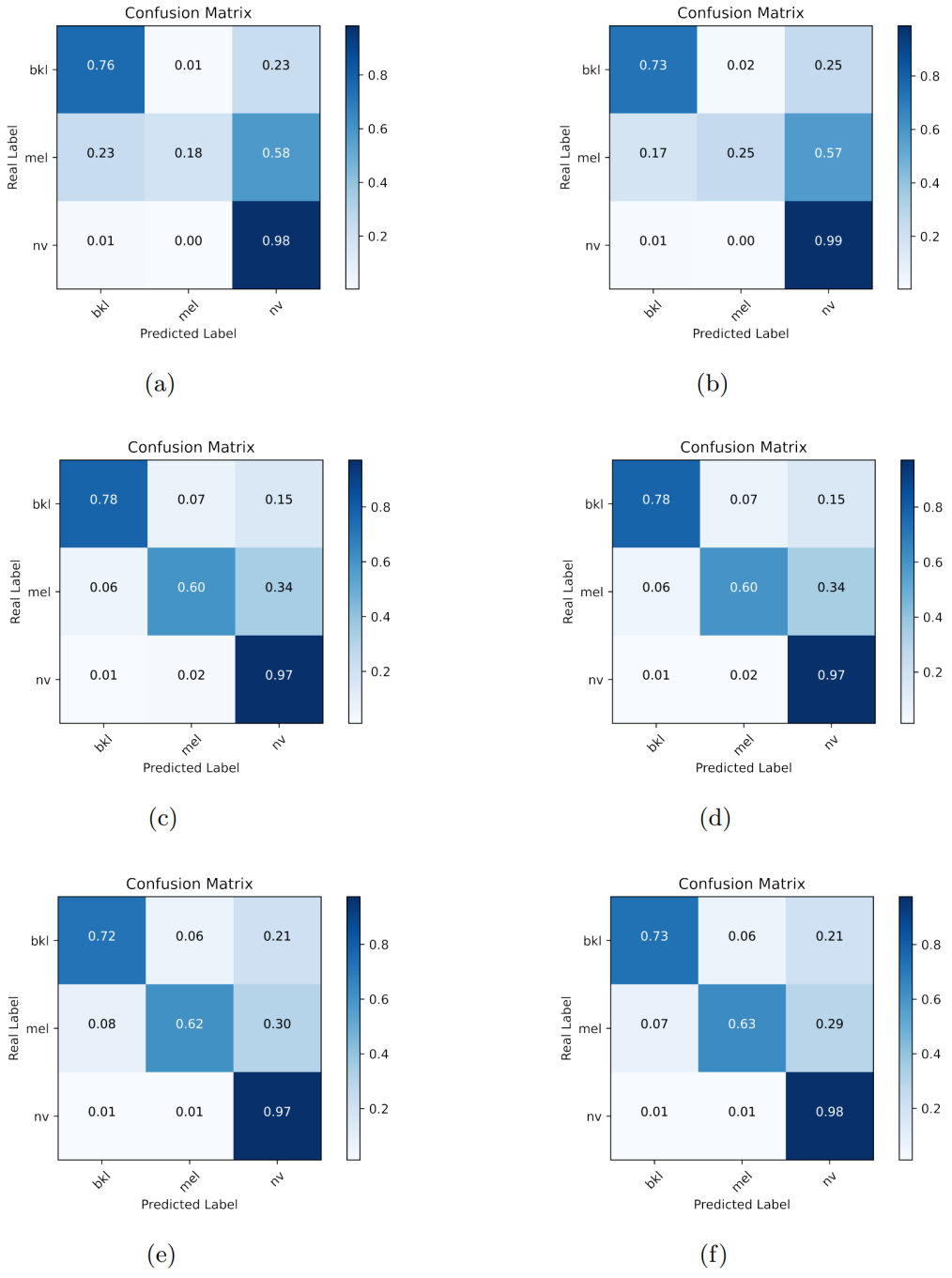


FIGURE 5.6: **Confusion Matrix of Ensemble of 2 Basic CNN Models. Letf colomn Without softmax, Right colomn with softmax (a),(b) ResNet - MobileNet (c),(d) ResNet - DenseNet (e),(f) ResNet - EfficientNet**

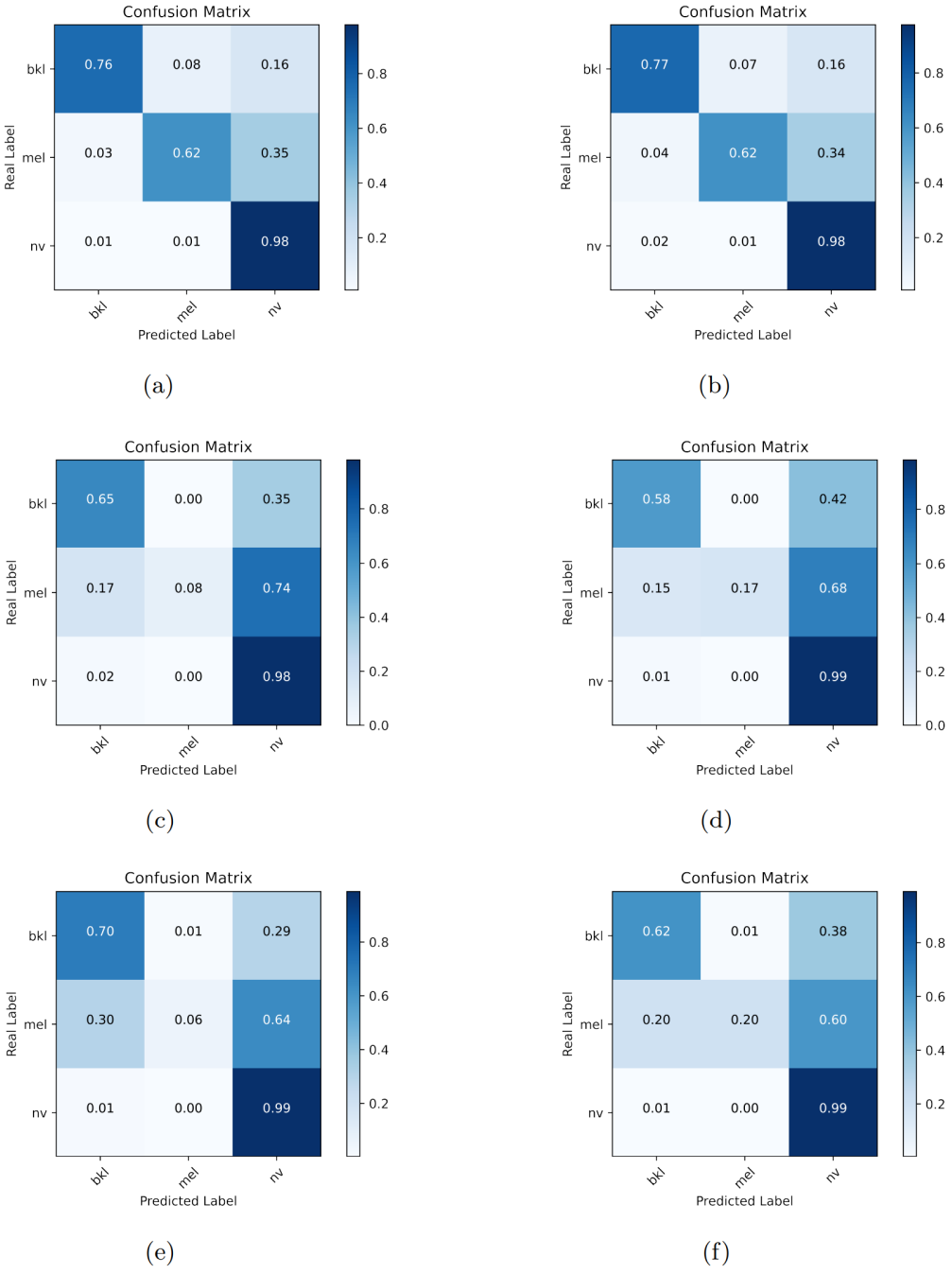


FIGURE 5.7: **Confusion Matrix of Ensemble of 2 Basic CNN Models. Letf colomn Without softmax, Right colomn with softmax (a),(b) ResNet - InceptionNet (c),(d) MobileNet - EfficientNet (e),(f) MobileNet - DenseNet**

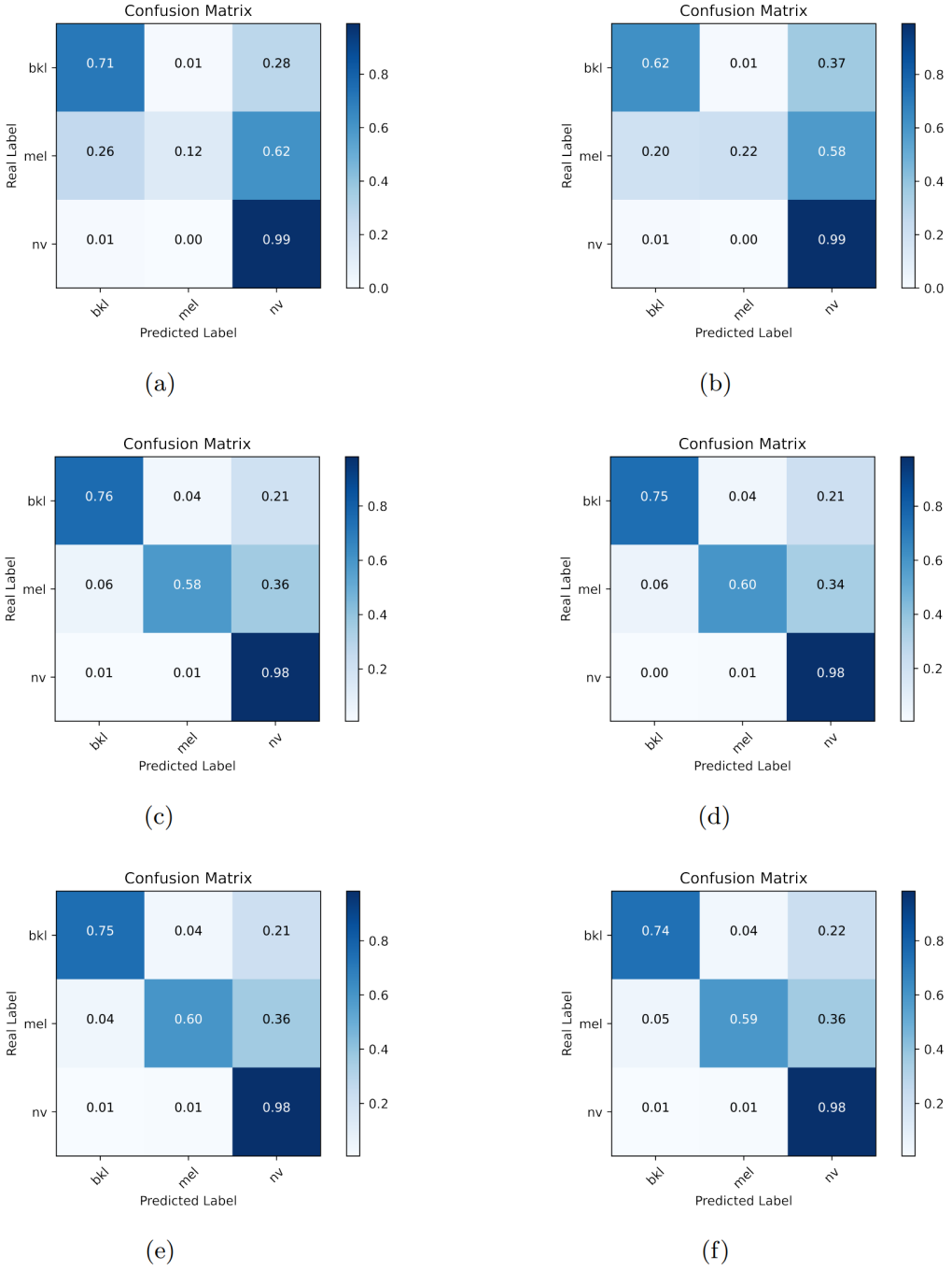
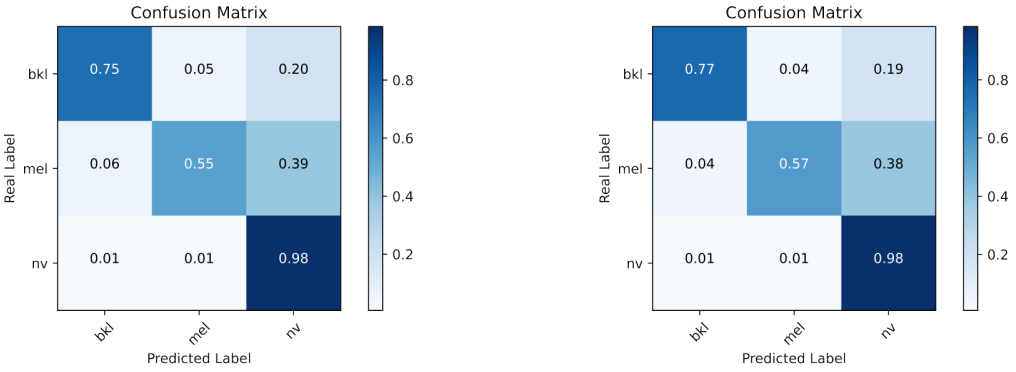


FIGURE 5.8: **Confusion Matrix of Ensemble of 2 Basic CNN Models. Letf colomn Without softmax, Right colomn with softmax (a),(b) MobileNet - InceptionNet (c),(d) EfficientNet - DenseNet (e),(f) EfficientNet - InceptionNet**



(a) (b)

FIGURE 5.9: **Confusion Matrix of Ensemble of 2 Basic CNN Models. Letf colomn Without softmax, Right colomn with softmax (a),(b) DenseNet - InceptionNet**

TABLE 5.13: **Inference Time of Ensemble of 2 Basic CNN Models on Hardware (CPU and GPU) with and without OpenVINO (results are in FPS)**

Model No.	Model_Name	Inferencing time with OpenVINO (FPS)				Inferencing time without OpenVINO (FPS)	
		FP16		FP32			
		CPU	GPU	CPU	GPU	CPU	GPU
1	Resnet_MobileNet	6.06	42.92	6.09	33.84	1.15	3.30
	Resnet_MobileNet_SM	2.67	46.82	2.66	32.41	1.21	3.23
2	Resnet_DenseNet	2.00	33.16	1.91	24.78	0.71	2.37
	Resnet_DenseNet_SM	2.00	33.60	1.90	24.44	0.86	2.39
3	Resnet_EfficientNet	2.26	34.52	2.26	26.36	0.85	2.36
	Resnet_EfficientNet_SM	2.22	35.19	2.28	26.30	0.86	2.39
4	Resnet_InceptionNet	1.93	36.61	4.48	27.11	1.03	2.75
	Resnet_InceptionNet_SM	1.95	36.24	4.54	26.95	1.05	2.80
5	MobileNet_EfficientNet	19.99	47.02	17.69	41.39	1.45	2.62
	MobileNet_EfficientNet_SM	16.18	44.81	18.10	41.49	1.49	2.53
6	MobileNet_DenseNet	13.62	42.56	14.26	38.91	1.33	3.03
	MobileNet_DenseNet_SM	13.16	42.88	13.74	38.31	1.15	2.94
7	MobileNet_InceptionNet	11.99	49.22	11.57	43.10	1.56	3.43
	MobileNet_InceptionNet_SM	13.31	50.22	14.08	43.45	1.47	3.34
8	EfficientNet_DenseNet	4.15	35.38	9.71	29.54	0.98	1.55
	EfficientNet_DenseNet_SM	4.20	35.70	8.67	29.51	1.00	1.60
9	EfficientNet_InceptionNet	4.23	36.83	4.21	33.15	1.25	2.14
	EfficientNet_InceptionNet_SM	4.13	42.75	4.22	34.19	1.15	2.22
10	DensetNet_InceptionNet	3.34	38.10	3.34	30.47	1.01	2.57
	DenseNet_InceptionNet_SM	3.29	38.67	3.28	30.69	1.18	2.68

5.4.2.3 Ensemble of 3 Basic CNN Models

TABLE 5.14: **Comparison of Precision, Recall, and F1-score of Class-0 of Ensemble of 3 Basic CNN Models**

Model_Name	Precision	Recall	F1-score
ResNet_EfficientNet_InceptionNet	0.84	0.75	0.79
ResNet_EfficientNet_InceptionNet_SM	0.85	0.75	0.80
ResNet_EfficientNet_MobileNet	0.82	0.71	0.76
ResNet_EfficientNet_MobileNet_with_SM	0.84	0.73	0.78
ResNet_MobileNet_DenseNet	0.72	0.78	0.75
ResNet_MobileNet_DenseNet_with_SM	0.77	0.73	0.75
ResNet_MobileNet_InceptionNet	0.73	0.77	0.75
ResNet_MobileNet_InceptionNet_SM	0.77	0.73	0.75
MobileNet_EfficientNet_DenseNet	0.75	0.66	0.70
MobileNet_EfficientNet_DenseNet_SM	0.73	0.59	0.65
MobileNet_EfficientNet_InceptionNet	0.75	0.67	0.71
MobileNet_EfficientNet_InceptionNet_SM	0.73	0.58	0.65
EfficientNet_DenseNet_InceptionNet	0.89	0.76	0.82
EfficientNet_DenseNet_InceptionNet_SM	0.89	0.74	0.81
ResNet_InceptionNet_DenseNet	0.86	0.73	0.79
ResNet_InceptionNet_DenseNet_SM	0.86	0.77	0.81
ResNet_EfficientNet_DenseNet	0.89	0.76	0.82
ResNet_EfficientNet_DenseNet_SM	0.89	0.75	0.81
MobileNet_InceptionNet_DenseNet	0.72	0.73	0.73
MobileNet_InceptionNet_DenseNet_SM	0.72	0.62	0.67

TABLE 5.15: **Comparison of Precision, Recall, and F1-score of Class-1 of Ensemble of 3 Basic CNN Models**

Model_Name	Precision	Recall	F1-score
ResNet_EfficientNet_InceptionNet	0.82	0.62	0.71
ResNet_EfficientNet_InceptionNet_with_SM	0.84	0.64	0.73
ResNet_EfficientNet_MobileNet	0.84	0.57	0.68
ResNet_EfficientNet_MobileNet_with_SM	0.82	0.62	0.71
ResNet_MobileNet_DenseNet	0.89	0.20	0.33
ResNet_MobileNet_DenseNet_with_SM	0.89	0.26	0.40
ResNet_MobileNet_InceptionNet	0.90	0.22	0.35
ResNet_MobileNet_InceptionNet_SM	0.89	0.26	0.40
MobileNet_EfficientNet_DenseNet	0.93	0.11	0.19
MobileNet_EfficientNet_DenseNet_SM	0.95	0.17	0.28
MobileNet_EfficientNet_InceptionNet	1.00	0.12	0.22
MobileNet_EfficientNet_InceptionNet_SM	0.95	0.17	0.30
EfficientNet_DenseNet_InceptionNet	0.88	0.58	0.70
EfficientNet_DenseNet_InceptionNet_SM	0.86	0.58	0.70
ResNet_InceptionNet_DenseNet	0.83	0.62	0.71
ResNet_InceptionNet_DenseNet_SM	0.84	0.62	0.72
ResNet_EfficientNet_DenseNet	0.87	0.61	0.71
ResNet_EfficientNet_DenseNet_SM	0.89	0.63	0.74
MobileNet_InceptionNet_DenseNet	0.95	0.17	0.28
MobileNet_InceptionNet_DenseNet_SM	0.97	0.25	0.40

TABLE 5.16: **Comparison of Precision, Recall, and F1-score of Class-2 of Ensemble of 3 Basic CNN Models**

Model_Name	Precision	Recall	F1-score
ResNet_EfficientNet_InceptionNet	0.90	0.90	0.90
ResNet_EfficientNet_InceptionNet_with_SM	0.92	0.98	0.95
ResNet_EfficientNet_MobileNet	0.91	0.98	0.94
ResNet_EfficientNet_MobileNet_with_SM	0.92	0.98	0.94
ResNet_MobileNet_DenseNet	0.87	0.99	0.93
ResNet_MobileNet_DenseNet_with_SM	0.87	0.99	0.93
ResNet_MobileNet_InceptionNet	0.87	0.99	0.93
ResNet_MobileNet_InceptionNet_SM	0.87	0.99	0.93
MobileNet_EfficientNet_DenseNet	0.84	0.99	0.91
MobileNet_EfficientNet_DenseNet_SM	0.84	0.99	0.91
MobileNet_EfficientNet_InceptionNet	0.84	0.99	0.91
MobileNet_EfficientNet_InceptionNet_SM	0.84	0.99	0.91
EfficientNet_DenseNet_InceptionNet	0.91	0.98	0.94
EfficientNet_DenseNet_InceptionNet_SM	0.9	0.98	0.94
ResNet_InceptionNet_DenseNet	0.91	0.98	0.95
ResNet_InceptionNet_DenseNet_SM	0.92	0.98	0.95
ResNet_EfficientNet_DenseNet	0.91	0.99	0.95
ResNet_EfficientNet_DenseNet_SM	0.91	0.98	0.95
MobileNet_InceptionNet_DenseNet	0.86	0.99	0.92
MobileNet_InceptionNet_DenseNet_SM	0.86	0.99	0.92

TABLE 5.17: **Comparison of Precision, Recall, and F1-score (along with Micro Average and Weighted Average) of Ensemble of 3 Basic CNN Models**

SL. No.	Model_Name	Precision	Recall	F1-score	Accuracy	Micro_Average			Weighted_Average		
						Precision	Recall	F1-score	Precision	Recall	F1-score
1	Res_Effi_Incept	0.86	0.78	0.82	0.90	0.86	0.78	0.82	0.90	0.90	0.90
	Res_Effi_Incept_SM	0.87	0.79	0.83	0.90	0.87	0.79	0.82	0.90	0.90	0.90
2	Res_Effi_Mob	0.86	0.75	0.79	0.89	0.85	0.75	0.79	0.89	0.89	0.88
	Res_Effi_Mob_SM	0.86	0.78	0.81	0.90	0.86	0.78	0.81	0.89	0.90	0.89
3	Res_Mob_Den	0.83	0.66	0.67	0.85	0.83	0.65	0.67	0.86	0.85	0.82
	Res_Mob_Den_SM	0.84	0.66	0.69	0.86	0.84	0.66	0.69	0.86	0.86	0.83
4	Res_Mob_Incept	0.83	0.66	0.68	0.86	0.83	0.66	0.67	0.86	0.86	0.83
	Res_Mob_Incept_SM	0.84	0.66	0.69	0.86	0.84	0.66	0.69	0.86	0.86	0.83
5	Mob_Effi_Den	0.84	0.59	0.60	0.83	0.84	0.59	0.60	0.84	0.83	0.79
	Mob_Effi_Den_SM	0.84	0.58	0.61	0.83	0.84	0.58	0.61	0.84	0.83	0.79
6	Mob_Effi_Incept	0.86	0.59	0.61	0.83	0.86	0.60	0.61	0.85	0.83	0.79
	Mob_Effi_Incept_SM	0.84	0.58	0.62	0.83	0.84	0.58	0.62	0.84	0.83	0.79
7	Effi_Den_Incept	0.89	0.76	0.82	0.90	0.89	0.78	0.82	0.90	0.90	0.89
	Effi_Den_Incept_SM	0.88	0.77	0.82	0.90	0.89	0.77	0.82	0.90	0.90	0.89
8	Res_Incept_Den	0.87	0.78	0.82	0.90	0.87	0.78	0.82	0.9	0.90	0.89
	Res_Incept_Den_SM	0.87	0.79	0.83	0.90	0.87	0.79	0.83	0.90	0.90	0.90
9	Res_Effi_Den	0.89	0.79	0.83	0.91	0.89	0.78	0.83	0.90	0.91	0.90
	Res_Effi_Den_SM	0.90	0.79	0.83	0.91	0.90	0.79	0.83	0.91	0.91	0.90
10	Mob_Incept_Den	0.84	0.63	0.64	0.85	0.84	0.63	0.64	0.86	0.85	0.81
	Mob_Incept_Den_SM	0.85	0.62	0.66	0.85	0.85	0.62	0.66	0.85	0.85	0.92

Res – ResNet, Effi – EfficienNet, Den – DesnseNet, Incept – InceptionNet, Mob – MobileNet

Confusion Matrix

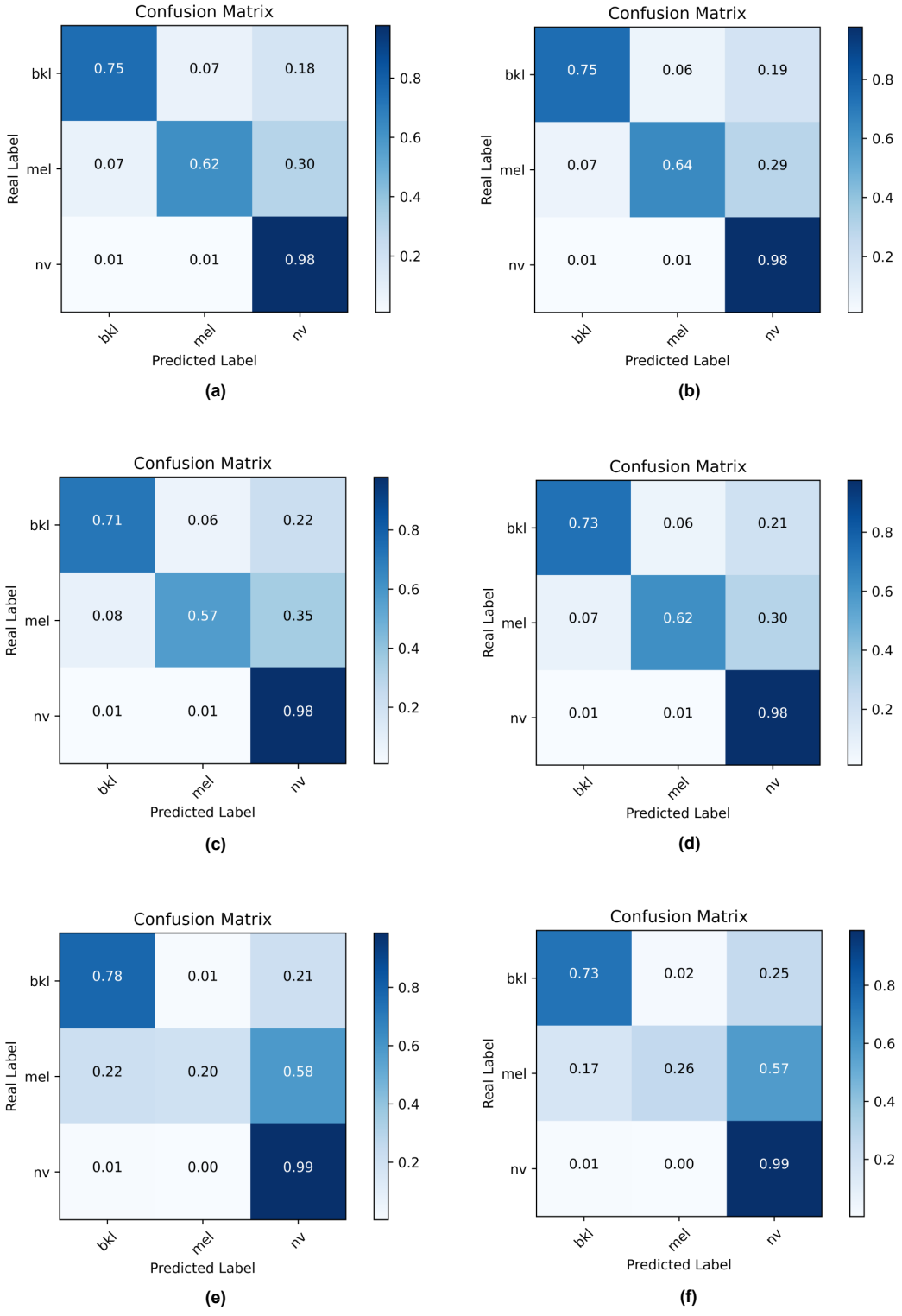


FIGURE 5.10: **Confusion Matrix of Ensemble of 3 Basic CNN Models. Letf colomn Without softmax, Right colomn with softmax (a),(b) ResNet - EfficientNet - InceptionNet, (c),(d) ResNet - EfficientNet - MobileNet, (e),(f) ResNet - MobileNet - DenseNet**

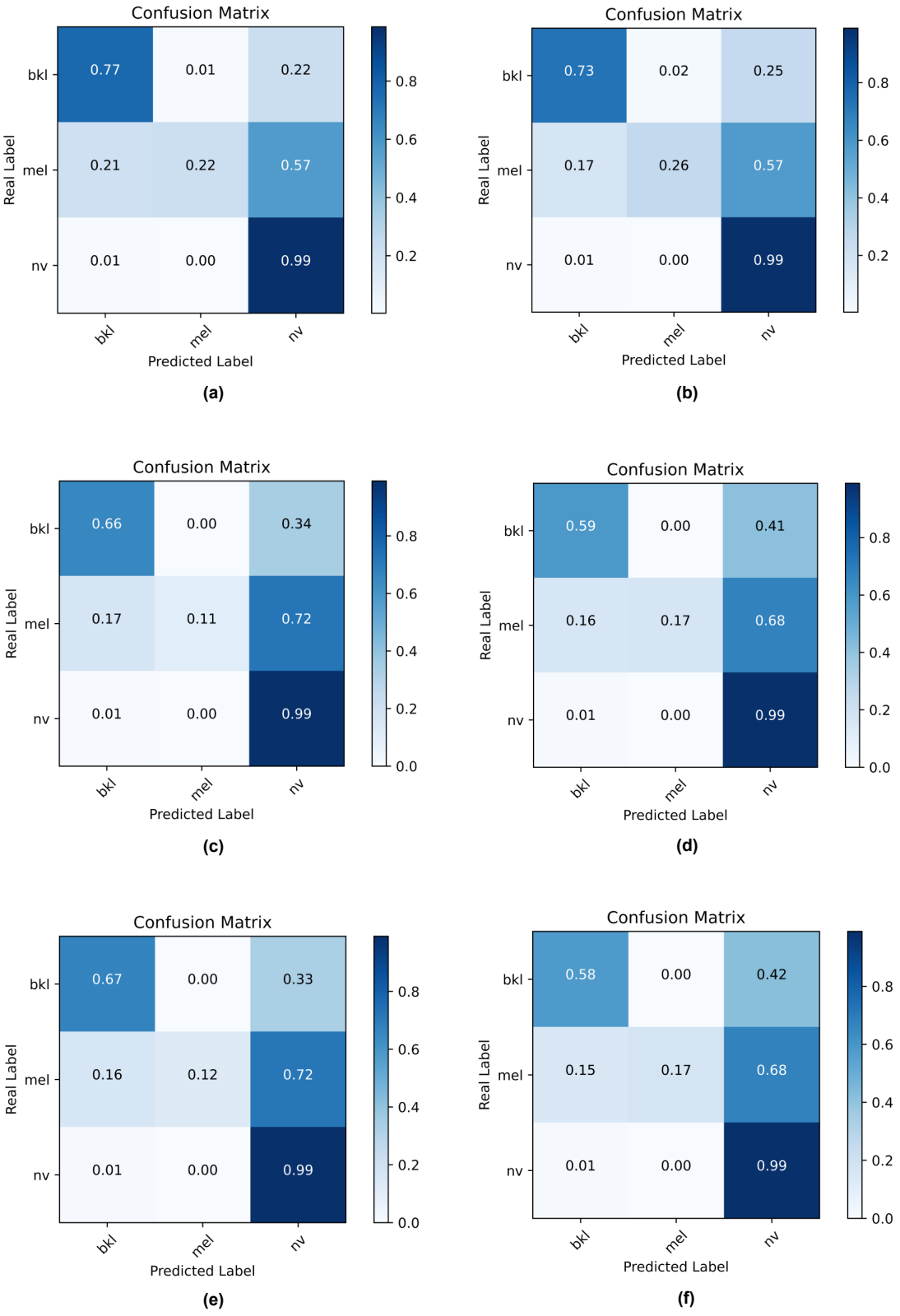


FIGURE 5.11: **Confusion Matrix of Ensemble of 3 Basic CNN Models. Letf colomn Without softmax, Right colomn with softmax (a),(b) ResNet - MobileNet - InceptionNet, (c),(d) MobileNet - EfficientNet - DenseNet, (e),(f) MobileNet - EfficientNet - InceptionNet**

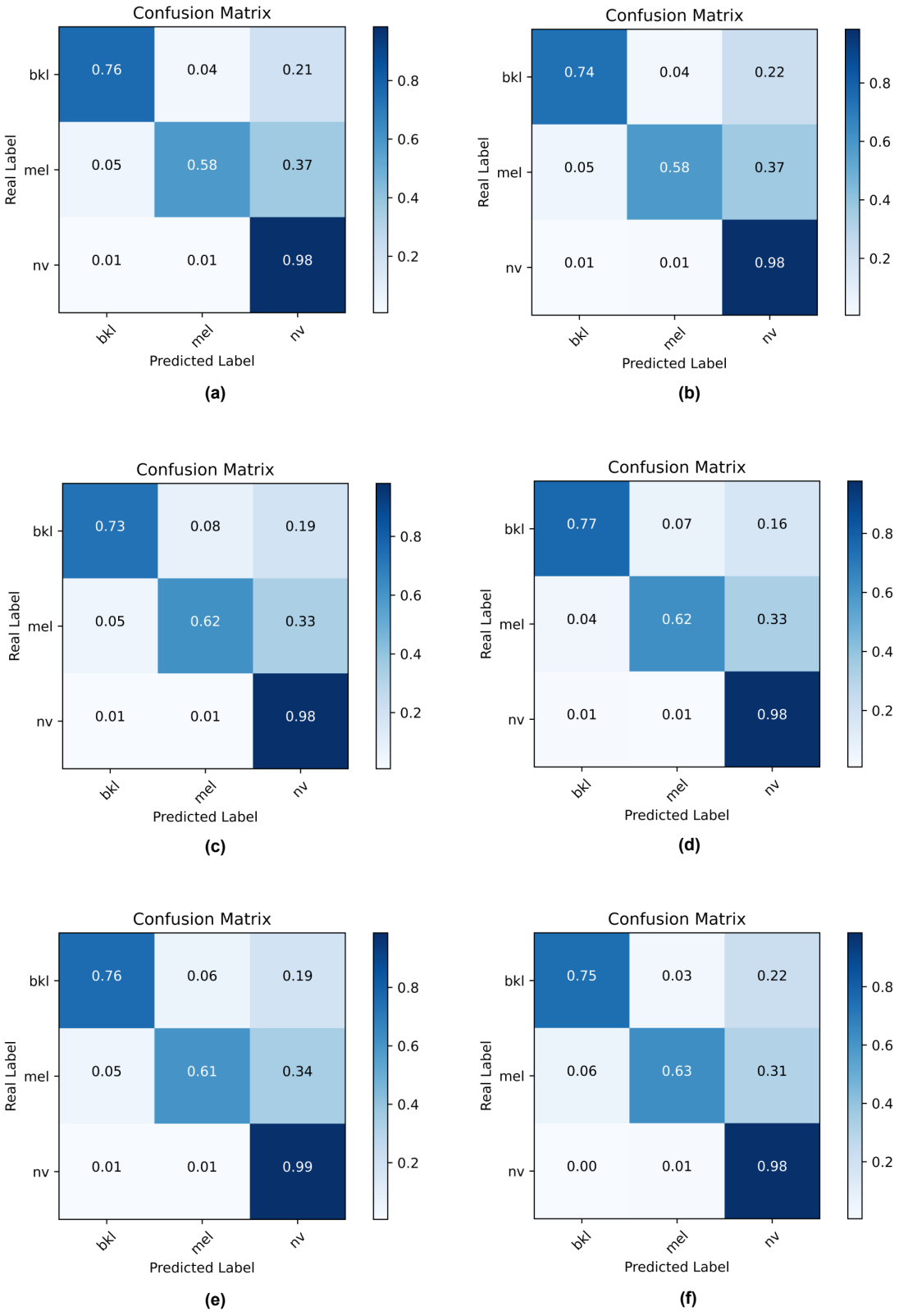


FIGURE 5.12: **Confusion Matrix of Ensemble of 3 Basic CNN Models. Letf colomn Without softmax, Right colomn with softmax (a),(b) EfficientNet - DenseNet - InceptionNet , (c),(d) ResNet - InceptionNet - DenseNet , (e),(f)ResNet - EfficientNet - DenseNet**

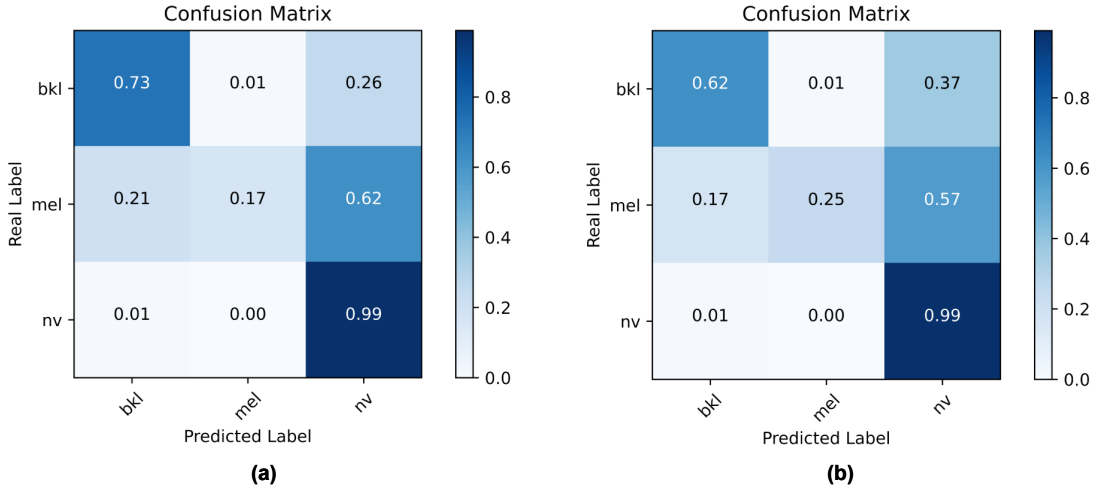


FIGURE 5.13: **Confusion Matrix of Ensemble of 3 Basic CNN Models. Letf colomn Without softmax, Right colomn with softmax (a),(b) MobileNet - InceptionNet - DenseNet**

TABLE 5.18: **Inference Time of Ensemble of 3 Basic CNN Models on Hardware (CPU and GPU) with and without OpenVINO (results are in FPS)**

Model No.	Model_Name	Inferencing time with OpenVINO (FPS)				Inferencing time without OpenVINO (FPS)	
		FP16		FP32			
		CPU	GPU	CPU	GPU	CPU	GPU
1	ResNet_EfficientNet_InceptionNet	1.56	24.51	1.61	19.24	0.55	1.60
	ResNet_EfficientNet_InceptionNet_SM	1.74	24.34	1.66	19.07	0.64	1.57
2	ResNet_EfficientNet_MobileNet	2.21	26.19	1.71	21.22	1.03	1.67
	ResNet_EfficientNet_MobileNet_SM	4.49	26.31	4.68	20.49	1.02	1.63
3	ResNet_MobileNet_DenseNet	7.15	25.65	1.53	19.53	0.88	1.84
	ResNet_MobileNet_DenseNet_SM	6.79	26.29	1.51	18.95	0.86	1.85
4	ResNet_MobileNet_InceptionNet	7.14	25.50	5.87	24.42	0.80	2.08
	ResNet_MobileNet_InceptionNet_SM	6.64	29.37	5.24	22.21	0.74	2.05
5	MobileNet_EfficientNet_DenseNet	13.00	33.56	10.15	22.34	0.75	1.74
	MobileNet_EfficientNet_DenseNet_SM	12.57	33.31	19.34	26.80	0.78	1.70
6	MobileNet_EfficientNet_InceptionNet	8.06	21.62	8.39	24.47	0.86	1.74
	MobileNet_EfficientNet_InceptionNet_SM	7.75	21.02	8.00	20.15	0.85	1.71
7	EfficientNet_DenseNet_InceptionNet	2.33	22.78	2.41	20.15	0.61	1.52
	EfficientNet_DenseNet_InceptionNet_SM	2.35	22.85	2.37	19.79	0.62	1.54
8	ResNet_InceptionNet_DenseNet	3.40	22.42	3.24	17.05	0.56	1.66
	ResNet_InceptionNet_DenseNet_SM	3.40	21.32	3.25	16.66	0.66	1.65
9	ResNet_EfficientNet_DenseNet	3.39	21.38	3.45	17.04	0.48	1.51
	ResNet_EfficientNet_DenseNet_SM	3.55	21.33	3.46	16.79	0.50	1.48
10	MobileNet_InceptionNet_DenseNet	8.27	26.50	6.60	23.97	1.07	1.95
	MobileNet_InceptionNet_DenseNet_SM	6.64	27.08	6.48	23.59	0.89	1.91

5.5 Result Analysis

In this section, I will discuss the results I have got after the experiments are done. I have discussed the result of the Levy Stable ensemble in the first section and the result of the Beta function-based ensemble in the second section. The second section is again divided into three subsections. In the first subsection, I have discussed the results of basic models. The second subsection contains a discussion of the ensemble of 2 Basic CNN Models followed by the third subsection containing the ensemble of 3 Basic CNN Models.

5.5.1 Levy Stable Ensemble

Base models are trained on the HAM10K dataset using the Nvidia T4 GPU with a batch size of 128 and a learning rate of 0.001. I have also used a learning rate scheduler to train the model. I have not done additional training for the ensemble; I am just using the base models for the Levy Stable function. The accuracy, precision, recall, and F1 score of the models are shown in Table **5.2**. EfficientNet-B0 has the lowest accuracy of 77% among all The ensemble method has the highest accuracy of 83%. The MobileNet-V2 and ensemble model have the same recall of 83%. The ensemble model has the highest precision and f1-score of 83% and EfficientNet-B0 has the lowest precision and f1-score of 77%. The confusion metrics of these models are shown in Fig. 6.

The system I have used here has a 12th Generation Intel i7 processor with 1.5 GB of in-built Intel iRIS Xe graphics. I inferred all these models using OpenVINO and without using OpenVINO. The outcomes given in FPS (frames per second) are tabulated in Table **5.3**. Without using OpenVINO I deployed my models on the CPU and GPU. The ensemble method inferred on the CPU has the lowest FPS of 3.98. I have also used Google Colab's Tesla T4 GPU to test the inference time and obtained an FPS of 4.27. Therefore, without using OpenVINO, the GPU processor (Google Colab T4) has a better inference time compared to the CPU. For using OpenVINO, first, I set up the environment to infer the model. Then FP16 and FP32 compressions were applied to models using Model Optimizer. Finally, the models were executed on both CPU and GPU. In the case of FP16 compression, the optimized ensemble model achieves real-time performance on the Intel i7 CPU, i.e. 110.3 FPS which is the highest FPS for the ensemble

model. MobileNet-V2 with the CPU has the highest FPS of 432.6 of all the used models. In the case of FP32 compression, the ensemble model obtained an FPS of 91.12 and 65.2 on the CPU and GPU, respectively.

Fig. 5.2 shows Confusion Matrices of two basic models and the proposed Levy Stable Ensemble model. And The Levy Stable Ensemble model shows better prediction compared to basic CNN models for MEL and Vasc classes with prediction rates of 83% and 71%, respectively.

5.5.2 Beta Ensemble

The training has been conducted on Google colab using Nvidia T4 GPU. I have used the HAM10k dataset. I have taken 3 classes NV, MEL, and BKL out of 7 for the classification task. As the data was imbalanced I used only 3 classes with the highest number of images.

5.5.2.1 Tables

The performance of skin cancer classification models plays a crucial role in accurately identifying malignant instances and guiding clinical decisions.

Basic Models: The Table 5.7 presents the overall performance metrics of different basic models. The performance metrics include precision, recall, F1-score, and accuracy. The table also includes micro-average and weighted-average values for precision, recall, and F1-score.

Analyzing the table, I observed the following:

- ResNet-101 (Model 1) achieved a precision of 36%, recall of 35%, F1-score of 35%, and an accuracy of 61%. The micro-average precision and recall are 35%, while the weighted-average precision and recall are 59% and 61%, respectively.
- DenseNet-121 (Model 2) achieved a precision of 31%, recall of 32%, F1-score of 31%, and an accuracy of 61%. The micro-average precision and recall are 31%, while the weighted-average precision and recall are 57% and 61%, respectively.

- EfficientNet-B3 (Model 3) achieved a precision of 36%, recall of 36%, F1-score of 36%, and an accuracy of 62%. The micro-average precision and recall are 36%, while the weighted-average precision and recall are 60% and 62%, respectively.
- InceptionNet-V3 (Model 4) achieved a precision of 31%, recall of 32%, F1-score of 32%, and an accuracy of 60%. The micro-average precision and recall are 31%, while the weighted-average precision and recall are 57% and 60%, respectively.
- MobileNet-V3-Large (Model 5) achieved a precision of 41%, recall of 34%, F1-score of 32%, and an accuracy of 69%. The micro-average precision and recall are 41%, while the weighted-average precision and recall are 61% and 69%, respectively.

From Table **5.7**, I can observe that MobileNet-V3-Large (Model 5) exhibited the most promising performance in terms of precision and accuracy. EfficientNet-B3 (Model 3) exhibited the most promising performance in terms of recall, and F1-score values.

Ensemble of 2 Basic Models : Table **5.12** illustrates the comparison of Precision, Recall, F1-score, and Accuracy of different ensembles of 2 basic models, combining two neural network architectures with softmax layer and without softmax layer.

- In comparing precision, models such as Model 8, Model 9, and Model 10 consistently achieve the highest precision values of 89%.
- In comparing recall values, models such as Model 2, Model 3, Model 4, and Model 8 consistently achieve high recall values ranging from 78% to 79%.
- While comparing F1-scores, models such as Model 3, Model 4, Model 8, Model 9, and Model 10 showcase impressive F1-scores of 82%.
- In the comparison of accuracies, models such as Model 2, Model 3, Model 4, Model 8, Model 9, and Model 10 consistently achieve the highest accuracy values of 90%.

- Among the different 2 basic model combinations, Model 3 exhibits high precision, with a value of 86%. The combination also achieves a recall of 78%. The corresponding F1-score is 82%. The accuracy achieved by Model 3 is 90%.
- Similarly, Model 4 achieves a precision of 87%, a recall of 79%, an F1-score of 82%, and an accuracy of 90%.
- Model 8 and Model 10 also yield promising results, with a precision of 89%, a recall of 78%, an F1-score of 82%, and an accuracy of 90%.

The above-mentioned Models with these values demonstrate high precision, effectively identifies positive instances, and maintains a balance between precision and recall. The achieved accuracy further emphasizes the model's ability to make correct predictions. These values also indicate their potential for accurate identification and classification of skin cancer instances. On the other hand, some model Models, such as Model 1, Model 5, Model 6, and Model 7, achieve relatively lower precision, recall, and F1-scores compared to the top-performing Models mentioned earlier. However, they still demonstrate acceptable performance, with precision values ranging from 82% to 84%, recall values ranging from 58% to 66%, and F1-scores ranging from 61% to 69%. The corresponding accuracies for these combinations range from 83% to 86%.

Ensemble of 3 Basic Models : Table **5.17** highlights the performance metrics of various ensembles of 3 basic models, combining three different neural network architectures with softmax layer and without softmax.

- In comparing precision, models such as Model 1, Model 7, Model 8, and Model 9 consistently achieve high precision values, ranging from 87% to 90%.
- In comparing recall, models such as Model 1, Model 2, Model 8, and Model 9 consistently achieve recall values ranging from 78% to 79%.
- While comparing F1-score, models such as Model 1, Model 2, Model 7, Model 8, and Model 9 demonstrate competitive F1-score, ranging from 81% to 83%.
- In the comparison of accuracy, models such as Model 1, Model 2, Model 7, Model 8, and Model 9 consistently achieve accuracy values of 90% to 91%.

- Among the different 2 basic model combinations, Model 1 achieves notable results with a precision of 87%, recall of 79%, F1-score of 83%, and accuracy of 90%.
- Similarly, Model 8 also yields promising results with a precision of 0.87, recall of 79%, F1-score of 83%, and accuracy of 90%.
- Model 9 achieves the best result out of all ensembles of 3 basic models with a precision of 90%, a recall of 79%, an F1-score of 83%, and an accuracy of 91%.

These metrics indicate that the above-mentioned Models can accurately identify and classify skin cancer instances with high precision and a balanced trade-off between precision and recall. The achieved accuracy further demonstrates the effectiveness of this model combination. And also showcase strong performance in accurately identifying and classifying skin cancer instances, suggesting its potential for practical implementation. The other Models, namely Model 3, Model 4, Model 5, Model 6, and Model 10, exhibit lower precision, recall, F1-score, and accuracy values compared to the top-performing Models mentioned earlier. Nevertheless, other Models still demonstrate acceptable performance, with precision values ranging from 84% to 85%, recall values ranging from 58% to 66%, and F1-scores ranging from 61% to 69%. The corresponding accuracies for other Models range from 83% to 86%.

5.5.2.2 Confusion Matrix

Basic Models: Figure 5.5. shows the Confusion Matrix (CM) of 5 Basic Models. I can observe that only the NV has been predicted truly for all the 5 base models. Rest two classes have been incorrectly predicted. Figure 5.5. (e) predicted NV correctly with 91% which is the highest out of all 5 CMs.

Ensemble of 2 basic Models: There are two types of Confusion Matrices in Figure 5.6., 5.7., 5.8., and 5.9. Left columns (a), (c), and (e) are without softmax layers, and right columns (b), (d), and (f) are with softmax layers. In Figure 5.6. (c), (d), (e), (f), 5.7. (a), (b), 5.8. (c), (d), (e), (f), and 5.9. (a), (b) have obtained the correct positive predictions for all the 3 classes. Figure 5.6. (c), (d) have predicted BKL class most correctly with 78% and (f) have predicted MEL

most correctly with 63%. Figure **5.6**. (c), (d), (e), (f), **5.7**. (a), (b), **5.8**. (c), (d), (e), (f), and **5.9**. (a), (b) which predicted all 3 classes correctly, have predicted NV correctly with a range of 97% to 98%. But Figure **5.6**. (b), **5.7**. (b), (d), (e), (f), and **5.8**. (a), (b) have obtained the highest correct percentage of 99% for NV. Figure **5.7**. (b) has shown the best result out of all with a prediction rate of 77%, 62%, and 98% for BKL, MEL, and NV respectively.

Ensemble of 3 Basic Models: There are two types of Confusion Matrices in Figure **5.10**., **5.11**., **5.12**., and **5.13**. Left columns (a), (c), and (e) are without softmax layers, and right columns (b), (d), and (f) are with softmax layers. A similar scenario can be observed in the case of Figure. **5.10**. (a), (b), (c), (d), **5.12**. (a), (b), (c), (d), (e), and (f) have obtained the correct positive predictions for all the 3 classes. Figure **??**. **5.12**. (d) and **5.10** (b) have predicted BKL and MEL classes most correctly with 77% and 64% respectively. Figure. **5.10**. (a), (b), (c), (d),

Chapter 6

CONCLUSION AND FUTURE SCOPE

This Chapter contains two sections, **6.1** contains the conclusion of my work and **6.2** contains the future scope of my work.

6.1 Conclusion

This thesis presents two novel ensemble techniques, the Levy Stable ensemble method and the Beta function-based ensemble technique, for skin cancer classification using the HAM10k dataset. The Beta ensemble approach effectively combines the confidence scores generated by multiple classifiers, leveraging the rank-based fusion technique based on the Beta function. Experimental results on the HAM10k dataset demonstrate the superiority of my proposed approach over existing methods. By leveraging the strengths of multiple classifiers and adapting their priorities based on confidence scores, my Beta ensemble technique achieves enhanced accuracy in skin cancer diagnosis, thereby improving the potential for early detection. Furthermore, I converted the models using the OpenVINO toolkit, enabling real-time inference. The Levy Stable ensemble model achieves an impressive inference rate of 110.3 FPS on an Intel-i7 CPU, achieving an F1-score of 83% and high accuracy. Similarly, the Beta ensemble model achieves an inference rate of 33.56 FPS on an Intel-i7 GPU, with a remarkable accuracy of 91%. These results demonstrate the effectiveness and practicality of my proposed ensemble techniques in skin cancer classification, paving the way for more accurate and efficient diagnostic systems.

6.2 Future Scope

In addition to the significant contributions made by the Levy Stable ensemble and Beta function-based ensemble techniques in skin cancer classification, there are several promising avenues for future research and development. The following points outline the potential future scope and applicability of these methods:

- **Enhanced Ensemble Techniques:** Further exploration and experimentation can be conducted to refine and optimize the ensemble techniques used

in this thesis. This could involve investigating different fusion methods, exploring the integration of additional classifiers, or incorporating advanced techniques for ensemble learning.

- **Integration with Advanced Imaging Technologies:** The application of advanced imaging technologies, such as hyperspectral imaging or multimodal imaging, can potentially improve the accuracy and reliability of skin cancer classification. Integrating these technologies with the ensemble techniques proposed in this thesis could lead to more robust and comprehensive diagnostic systems.

- **Evaluation on Larger and Diverse Datasets:** While the HAM10k dataset has provided valuable insights and demonstrated the effectiveness of the proposed methods, it is essential to evaluate these techniques on larger and more diverse datasets. This would enable a more comprehensive assessment of their performance and generalizability.

- **Real-world Deployment and Clinical Validation:** The next step in the development process would involve real-world deployment and clinical validation of the proposed ensemble techniques. Collaborating with medical professionals and conducting extensive clinical studies would help assess the methods' efficacy, reliability, and performance in real clinical settings.

- **Integration into Commercial Products:** The ensemble techniques proposed in this thesis have the potential to be developed into commercial products for skin cancer classification. Integrating these methods into user-friendly software or hardware solutions could assist healthcare professionals in accurate and efficient diagnosis, leading to improved patient outcomes.

- **Extension to Other Medical Applications:** The concepts and methodologies employed in this thesis can be extended to other medical applications beyond skin cancer classification. Similar ensemble techniques could be explored for the diagnosis and classification of other types of cancers, diseases, or medical conditions, potentially revolutionizing the field of medical diagnostics.

By addressing these future research directions and exploring the application of these ensemble techniques in diverse medical contexts, I can further advance the field of computer-aided diagnosis and contribute to improved healthcare outcomes.

Chapter 7

LIST OF PUBLICATIONS

- **M. Hait**, R. Das, A. Ali, S. S. Chaudhuri, "SKIN CANCER CLASSIFICATION USING LEVY STABLE BASED ENSEMBLE AND IT'S REAL-TIME IMPLEMENTATION ON OPENVINO TOOLKIT". Communicated with "International Conference on Image Processing Theory, Tools and Applications" **(IPTA), 2023**.

- **M. Hait**, A. Ali, S. S. Chaudhuri, "BETA FUNCTION BASED FUZZY ENSEMBLE TECHNIQUE FOR SKIN CANCER CLASSIFICATION". Communicated with "International Journal of Interactive Multimedia And Artificial Intelligence" **(IJIMAI), 2023**.

References

Chirag Sharma. Self driving car using deep learning technique. *International Journal of Engineering Research and*, V9, 06 2020. doi: 10.17577/IJERTV9IS060247.

Jan Egger, Christina Gsaxner, Antonio Pepe, Kelsey L. Pomykala, Frederic Jonske, Manuel Kurz, Jianning Li, and Jens Kleesiek. Medical deep learning—a systematic meta-review. *Computer Methods and Programs in Biomedicine*, 221:106874, 2022. ISSN 0169-2607. doi: https://doi.org/10.1016/j.cmpb.2022.106874. URL https://www.sciencedirect.com/science/article/pii/S0169260722002565.

Ho Heon Kim, Jae Il An, and Yu Rang Park. A prediction model for detecting developmental disabilities in preschool-age children through digital biomarker-driven deep learning in serious games: Development study. *JMIR Serious Games*, 9(2):e23130, Jun 2021. ISSN 2291-9279. doi: 10.2196/23130. URL https://doi.org/10.2196/23130.

Shashi Pal Singh, Ajai Kumar, Hemant Darbari, Lenali Singh, Anshika Rastogi, and Shikha Jain. Machine translation using deep learning: An overview. In *2017 International Conference on Computer, Communications and Electronics (Comptelix)*, pages 162–167, 2017. doi: 10.1109/COMPTELIX.2017.8003957.

Leila Kiani, Masoud Saeed, and Hossein Nezamabadi-Pour. Image colorization using a deep transfer learning. In *2020 8th Iranian Joint Congress on Fuzzy and intelligent Systems (CFIS)*, pages 027–032, 2020. doi: 10.1109/CFIS49607.2020.9238737.

Md. Nawab Yousuf Ali, Md. Lizur Rahman, Jyotismita Chaki, Nilanjan Dey, and K. C. Santosh. Machine translation using deep learning for universal networking language based on their structure. *International Journal of Machine Learning and Cybernetics*, 12(8):2365–2376, Aug 2021. ISSN 1868-808X. doi: 10.1007/s13042-021-01317-5. URL https://doi.org/10.1007/s13042-021-01317-5.

Niels Justesen, Philip Bontrager, Julian Togelius, and Sebastian Risi. Deep learning for video game playing. *IEEE Transactions on Games*, PP, 08 2017. doi: 10.1109/TG.2019.2896986.

Senthilkumar Mohan and Chiranji Chowdhary. *An AI-Based Chatbot Using Deep Learning*, pages 231–242. 12 2019. ISBN 9780429265020. doi: 10.1201/9780429265020-12.

Radouan Mouha. Deep learning for robotics. *Journal of Data Analysis and Information Processing*, 09:63–76, 01 2021. doi: 10.4236/jdaip.2021.92005.

Carlos Hernandez Oliván and José Ramón Beltrán Blázquez. Music composition with deep learning: A review, 08 2021.

Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3212–3232, 2019. doi: 10.1109/TNNLS.2018.2876865.

M krishna, M Neelima, Harshali Mane, and Venu Matcha. Image classification using deep learning. *International Journal of Engineering Technology*, 7:614, 03 2018. doi: 10.14419/ijet.v7i2.7.10892.

Xiang Chen, Andres Diaz-Pinto, Nishant Ravikumar, and Alejandro F Frangi. Deep learning in medical image registration. *Progress in Biomedical Engineering*, 3(1):012003, feb 2021. doi: 10.1088/2516-1091/abd37c. URL `https://dx.doi.org/10.1088/2516-1091/abd37c`.

Sanskruti Patel. *Deep Learning Models for Image Segmentation*. 07 2021. ISBN 78-93-80544-43-4. doi: 10.1109/INDIACom51348.2021.00027.

Mehwish Zafar, Muhammad Imran Sharif, Muhammad Irfan Sharif, Seifedine Kadry, Syed Ahmad Chan Bukhari, and Hafiz Tayyab Rauf. Skin lesion analysis and cancer detection based on machine/deep learning techniques: A comprehensive survey. *Life*, 13(1), 2023. ISSN 2075-1729. doi: 10.3390/life13010146. URL `https://www.mdpi.com/2075-1729/13/1/146`.

V. Vipin, Malaya Kumar Nath, V. Sreejith, Nikhil Francis Giji, Adithya Ramesh, and M. Meera. Detection of melanoma using deep learning techniques: A review. In *2021 International Conference on Communication, Control and Information Sciences (ICCISc)*, volume 1, pages 1–8, 2021. doi: 10.1109/ICCISc52257.2021.9484861.

Skin cancer statistics — World Cancer Research Fund International — wcrf.org. https://www.wcrf.org/cancer-trends/skin-cancer-statistics/, 2020.

Juhi Chhatlani, Tejas Mahajan, Rushabh Rijhwani, Advait Bansode, and Gresha Bhatia. Dermagenics - early detection of melanoma using yolov5 deep convolutional neural networks. In *2022 IEEE Delhi Section Conference (DELCON)*, pages 1–6, 2022. doi: 10.1109/DELCON54057.2022.9753227.

Philipp Tschandl. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions, 2018. URL https://doi.org/10.7910/DVN/DBW86T.

Intel® distribution of openvino™ toolkit. https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html.

Pratik Bhowal, Subhankar Sen, Juan D. Velasquez, and Ram Sarkar. Fuzzy ensemble of deep learning models using choquet fuzzy integral, coalition game and information theory for breast cancer histology classification. *Expert Systems with Applications*, 190:116167, 2022. ISSN 0957-4174. doi: 10.1016/j.eswa.2021.116167. URL https://doi.org/10.1016/j.eswa.2021.116167.

Ashis Paul, Arpan Basu, Mufti Mahmud, M. Shamim Kaiser, and Ram Sarkar. Inverted bell-curve-based ensemble of deep learning models for detection of covid-19 from chest x-rays. *Neural Computing and Applications*, Jan 2022. ISSN 1433-3058. doi: 10.1007/s00521-021-06737-6. URL https://doi.org/10.1007/s00521-021-06737-6.

Rishav Pramanik, Momojit Biswas, Shibaprasad Sen, Luis Antonio de Souza Júnior, João Paulo Papa, and Ram Sarkar. A fuzzy distance-based ensemble of deep models for cervical cancer detection. *Computer Methods and Programs in Biomedicine*, 219:106776, 2022. ISSN 0169-2607. doi: 10.1016/j.cmpb.2022.106776. URL https://doi.org/10.1016/j.cmpb.2022.106776.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. URL http://arxiv.org/abs/1608.06993.

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019a. URL http://arxiv.org/abs/1905.11946.

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3. *CoRR*, abs/1905.02244, 2019. URL `http://arxiv.org/abs/1905.02244`.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015. URL `http://arxiv.org/abs/1512.00567`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015a. URL `http://arxiv.org/abs/1512.03385`.

S. Sigurdsson, P.A. Philipsen, L.K. Hansen, J. Larsen, M. Gniadecka, and H.C. Wulf. Detection of skin cancer by classification of raman spectra. *IEEE Transactions on Biomedical Engineering*, 51(10):1784–1793, 2004. doi: 10.1109/TBME.2004.831538. URL `https://doi.org/10.1109/TBME.2004.831538`.

Margarida Silveira, Jacinto Nascimento, Jorge Marques, André Marçal, Teresa Mendonça, Syogo Yamauchi, Junji Maeda, and Jorge Rozeira. Comparison of segmentation methods for melanoma diagnosis in dermoscopy images. *Selected Topics in Signal Processing, IEEE Journal of*, 3:35 – 45, 03 2009. doi: 10.1109/JSTSP.2008.2011119. URL `https://doi.org/10.1109/JSTSP.2008.2011119`.

M.Emre Celebi, Hitoshi Iyatomi, Gerald Schaefer, and William V. Stoecker. Lesion border detection in dermoscopy images. *Computerized Medical Imaging and Graphics*, 33(2):148–153, 2009. ISSN 0895-6111. doi: https://doi.org/10.1016/j.compmedimag.2008.11.002.

Mai Mabrouk, Mariam Sheha, and Amr Sharawi. Automatic detection of melanoma skin cancer using texture analysis. *International Journal of Computer Applications*, 42:22–26, 03 2012. doi: 10.5120/5817-8129.

Bino Vadakkenveettil, Avittathur Unnikrishnan, and Kannan Balakrishnan. Gray level co-occurrence matrices: Generalisation and some new features. 2, 05 2012.

Marius-Constantin Popescu, Valentina Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8, 07 2009.

Catarina Barata, Margarida Ruela, Mariana Francisco, Jorge Marques, and Teresa Mendonça. Two systems for the detection of melanomas in dermoscopy images using texture and color features. *IEEE Systems Journal*, 8, 07 2013. doi: 10.1109/JSYST.2013.2271540.

Diwakar Gautam and Mushtaq Ahmed. Melanoma detection and classification using svm based decision support system. In *2015 Annual IEEE India Conference (INDICON)*, pages 1–6, 2015. doi: 10.1109/INDICON.2015.7443447.

Theodoros Evgeniou and Massimiliano Pontil. Support vector machines: Theory and applications. volume 2049, pages 249–257, 09 2001. ISBN 978-3-540-42490-1. doi: 10.1007/3-540-44673-7_12.

Ta-Wen Kuan, Jhing-Fa Wang, Jia-Ching Wang, and Gaung-Hui Gu. Vlsi design of sequential minimal optimization algorithm for svm learning. In *2009 IEEE International Symposium on Circuits and Systems*, pages 2509–2512, 2009. doi: 10.1109/ISCAS.2009.5118311.

J. Khushbu V. Krishna Sree. A fpga implementation on skin cancer detection using tdls algorithm. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 04:5917–5922, 07 2015. doi: 10.15662/ijareeie.2015.0407016.

Shereen Afifi, Hamid GholamHosseini, and Roopak Sinha. A low-cost fpga-based svm classifier for melanoma detection. In *2016 IEEE EMBS Conference on Biomedical Engineering and Sciences (IECBES)*, pages 631–636, 2016. doi: 10.1109/IECBES.2016.7843526.

Dr. Shailaja K. Poornima M. S. Detection of skin cancer using svm. *International Research Journal of Engineering and Technology (IRJET)*, 04:3021–3024, 07 2017. doi: V4/i7/IRJET-V4I7609. URL `https://www.irjet.net/archives/V4/i7/IRJET-V4I7609.pdf`.

Shereen Afifi, Hamid GholamHosseini, and Roopak Sinha. A system on chip for melanoma detection using fpga-based svm classifier. *Microprocessors and Microsystems*, 65:57–68, 2019. ISSN 0141-9331. doi: https://doi.org/10.1016/j.micpro.2018.12.005. URL `https://www.sciencedirect.com/science/article/pii/S0141933118303600`.

Balazs Harangi. Skin lesion classification with ensembles of deep convolutional neural networks. *Journal of Biomedical Informatics*, 86:25–32, 2018. ISSN 1532-0464. doi: https://doi.org/10.1016/j.jbi.2018.08.006.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015b.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

Nawal Soliman ALKolifi ALEnezi. A method of skin disease detection using image processing and machine learning. *Procedia Computer Science*, 163:85–92, 2019. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2019.12.090. URL `https://www.sciencedirect.com/science/article/pii/S1877050919321295`. 16th Learning and Technology Conference 2019Artificial Intelligence and Machine Learning: Embedding the Intelligence.

Hung N. Pham, Chin Yang Koay, Tanmoy Chakraborty, Sudhanshu Gupta, Boon Leong Tan, Huaqing Wu, Apurva Vardhan, Quang H. Nguyen, Nirmal Raja Palaparthi, Binh P. Nguyen, and Matthew C. H. Chua. Lesion segmentation and automated melanoma detection using deep convolutional neural networks and xgboost. In *2019 International Conference on System Science and Engineering (ICSSE)*, pages 142–147, 2019. doi: 10.1109/ICSSE.2019.8823129.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *AAAI Conference on Artificial Intelligence*, 31, 02 2016. doi: 10.1609/aaai.v31i1.11231.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. URL `http://arxiv.org/abs/1505.04597`.

Matt Berseth. ISIC 2017 - skin lesion analysis towards melanoma detection. *CoRR*, abs/1703.00523, 2017. URL `http://arxiv.org/abs/1703.00523`.

Md Ashraful Alam Milton. Automated skin lesion classification using ensemble of deep neural networks in ISIC 2018: Skin lesion analysis towards melanoma detection challenge. *CoRR*, abs/1901.10802, 2019. URL `http://arxiv.org/abs/1901.10802`.

Chenxi Liu, Barret Zoph, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan L. Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. *CoRR*, abs/1712.00559, 2017. URL `http://arxiv.org/abs/1712.00559`.

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507, 2017. URL `http://arxiv.org/abs/1709.01507`.

Redha Ali, Russell Hardie, Manawaduge Silva, and Temesguen Kebede. Skin lesion segmentation and classification for isic 2018 by combining deep cnn and handcrafted features, 08 2019.

Tsung-Chieh Lin and Hsi-Chieh Lee. Skin cancer dermoscopy images classification with meta data via deep learning ensemble. In *2020 International Computer Symposium (ICS)*, pages 237–241, 2020. doi: 10.1109/ICS51289.2020.00055.

Khadija Safdar, Shahzad Akbar, and Sahar Gull. An automated deep learning based ensemble approach for malignant melanoma detection using dermoscopy images. In *2021 International Conference on Frontiers of Information Technology (FIT)*, pages 206–211, 2021. doi: 10.1109/FIT53504.2021.00046.

Yutong Li, Ruoqing Zhu, Mike Yeh, and Annie Qu. Dermoscopic dataset for the "dermoscopic image classification with neural style transfer" manuscript, 2021. URL `https://dx.doi.org/10.21227/10k4-3p31`.

Ioannis Giotis, Nynke Molders, Sander Land, Michael Biehl, Marcel F. Jonkman, and Nicolai Petkov. Med-node: A computer-assisted melanoma diagnosis system using non-dermoscopic images. *Expert Systems with Applications*, 42 (19):6578–6585, 2015. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2015.04.034. URL `https://www.sciencedirect.com/science/article/pii/S0957417415002705`.

Chengdong Yao. A comprehensive evaluation study on risk level classification of melanoma by computer vision on isic 2016-2020 datasets, 2023.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038, 2014. URL `http://arxiv.org/abs/1411.4038`.

Ahmed Elngar, Rishabh Kumar, Amber Hayat, and Prathamesh Churi. Intelligent system for skin disease prediction using machine learning intelligent system for skin disease prediction using machine learning. *Journal of Physics Conference Series*, 1998:12037, 08 2021. doi: 10.1088/1742-6596/1998/1/012037.

Kritika Sujay Rao, Pooja Suresh Yelkar, Omkar Narayan Pise, and Dr Swapna Borde. *Skin Disease Detection using Machine Learning*, volume NTASU – Volume 09 – Issue 03. 2020. doi: 10.17577/IJERTCONV9IS03016. URL `https://www.ijert.org/skin-disease-detection-using-machine-learning`.

Parvathaneni Naga Srinivasu, Jalluri Gnana SivaSai, Muhammad Fazal Ijaz, Akash Kumar Bhoi, Wonjoon Kim, and James Jin Kang. Classification of skin disease using deep learning neural networks with mobilenet v2 and lstm. *Sensors*, 21(8):2852, Apr 2021. ISSN 1424-8220. doi: 10.3390/s21082852. URL `http://dx.doi.org/10.3390/s21082852`.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. pages 4510–4520, 06 2018a. doi: 10.1109/CVPR.2018.00474.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735.

Basant Agarwal, Valentina Emilia Balas, Lakhmi C. Jain, Ramesh Chandra Poonia, and Manisha. 4 - a systematic approach for identification of tumor regions in the human brain through haris algorithm. In *Deep Learning Techniques for Biomedical and Health Informatics*, pages 97–118. Academic Press, 2020. ISBN 978-0-12-819061-6. doi: https://doi.org/10.1016/B978-0-12-819061-6.00004-5.

Yeong Chan Lee, Sang-Hyuk Jung, and Hong-Hee Won. Wonderm: Skin lesion classification with fine-tuned neural networks. *CoRR*, abs/1808.03426, 2018. URL `http://arxiv.org/abs/1808.03426`.

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017. URL `http://arxiv.org/abs/1704.04861`.

Runyuan Zhang. Melanoma detection using convolutional neural network. In *2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, pages 75–78, 2021. doi: 10.1109/ICCECE51280.2021.9342142.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019b.

S Jagadish Kumar, U Maheswaran, G Jaikishan, and B Divagar. Melanoma classification using xgb classifier and efficientnet. In *2021 International Conference on Intelligent Technologies (CONIT)*, pages 1–4, 2021. doi: 10.1109/CONIT51480.2021.9498424.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754, 2016. URL `http://arxiv.org/abs/1603.02754`.

Mohamed A. Kassem, Khalid M. Hosny, and Mohamed M. Fouad. Skin lesions classification into eight classes for isic 2019 using deep convolutional neural network and transfer learning. *IEEE Access*, 8:114822–114832, 2020. doi: 10.1109/ACCESS.2020.3003890.

Takfarines Guergueb and Moulay A. Akhloufi. Melanoma skin cancer detection using recent deep learning models. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*, pages 3074–3077, 2021. doi: 10.1109/EMBC46164.2021.9631047.

Shaden Abdulaziz AlDera and Mohamed Tahar Ben Othman. A model for classification and diagnosis of skin disease using machine learning and image processing techniques. *International Journal of Advanced Computer Science and Applications*, 13(5), 2022. doi: 10.14569/IJACSA.2022.0130531. URL `http://dx.doi.org/10.14569/IJACSA.2022.0130531`.

Jaynab Sultana, Binoy Saha, Shuvo Khan, T.M Sanjida, Mehedi Hasan, and Mohammad Monirujjaman Khan. Identification and classification of melanoma

using deep learning algorithm. In *2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*, pages 1–6, 2022. doi: 10.1109/ICDCECE53908.2022.9792698.

Mohammad Naved Qureshi and Mohammad Sarosh Umar. Analysis of different deep learning techniques for the development of an efficient cnn model for melanoma skin cancer diagnosis. In *2022 International Conference for Advancement in Technology (ICONAT)*, pages 1–6, 2022. doi: 10.1109/ICONAT53423.2022.9726072.

François Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357, 2016. URL `http://arxiv.org/abs/1610.02357`.

K. Padmavathi, Harish Neelam, Mora Prathap Kumar Reddy, Priyanka Yadlapalli, Koushik Sai Veerella, and Karthik Pampari. Melanoma detection using deep learning. In *2022 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–4, 2022. doi: 10.1109/ICCCI54379.2022.9741010.

Bhuvaneshwari Shetty, Roshan Fernandes, Anisha P. Rodrigues, Rajeswari Chengoden, Sweta Bhattacharya, and Kuruva Lakshmanna. Skin lesion classification of dermoscopic images using machine learning and convolutional neural network. *Scientific Reports*, 12(1):18134, Oct 2022. doi: 10.1038/s41598-022-22644-9.

Bill Cassidy, Connah Kendrick, Andrzej Brodzicki, Joanna Jaworek-Korjakowska, and Moi Hoon Yap. Analysis of the isic image datasets: Usage, benchmarks and recommendations. *Medical Image Analysis*, 75:102305, 2022. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2021.102305. URL `https://www.sciencedirect.com/science/article/pii/S1361841521003509`.

Diepgen TL and Yihune G. *Dermatology information system - dermis*. http://dermis.net/, 2012. URL `http://dermis.net/`.

Soumen Mukherjee, Arunabha Adhikari, and Madhusudan Roy. *Malignant Melanoma Identification Using Best Visually Imperceptible Features from Dermofit Dataset*, pages 263–274. 01 2019. ISBN 978-981-13-6217-0. doi: 10.1007/978-981-13-3122-0_25.

Jeremy Kawahara, Sara Daneshvar, Giuseppe Argenziano, and Ghassan Hamarneh. Seven-point checklist and skin lesion classification using multitask multimodal neural nets. *IEEE Journal of Biomedical and Health Informatics*, 23(2):538–546, 2019. doi: 10.1109/JBHI.2018.2824327.

Lucia Ballerini, Robert Fisher, Ben Aldridge, and Jonathan Rees. *A Color and Texture Based Hierarchical K-NN Approach to the Classification of Non-melanoma Skin Lesions*, volume 6, pages 63–86. 01 2013. ISBN 978-94-007-5388-4. doi: 10.1007/978-94-007-5389-1_4.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018b.

Srikanth Tammina. Transfer learning using vgg-16 with deep convolutional neural network for classifying images. volume 9, page p9420, 10 2019. doi: 10.29322/IJSRP.9.10.2019.p9420.

V. Sudha and Dr Ganeshbabu. A convolutional neural network classifier vgg-19 architecture for lesion detection and grading in diabetic retinopathy based on deep learning. *Computers, Materials Continua*, 66:827–842, 01 2020. doi: 10.32604/cmc.2020.012008.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL `http://arxiv.org/abs/1502.03167`.

Alexei Botchkarev. A new typology design of performance metrics to measure errors in machine learning regression algorithms. *Interdisciplinary Journal of Information, Knowledge, and Management*, 14:045–076, 2019. doi: 10.28945/4184. URL `https://doi.org/10.28945%2F4184`.