Deep Learning Dog Breed Identification and Classification

A project report submitted in the partial fulfilment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

SK. Zaheed Husain (21471A05C6)

V. Rakesh Kumar (21471A05D5)

P. Sai Kumar (21471A05B4)

Under the esteemed guidance of

G. SARANYA **M.Tech**

Asst. Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

Accredited by NAAC with A+ grade and NBA under cycle-1 NIRF rank in the brand of 251-320 and ISO 9001:2015 Certified

Approved by AICTE, New Delhi. Permanently Affiliated to J.N.T.U, Kakinada

KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET-522601

2024 – 2025

NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NEC
NARASARAOPETA
ENGINEERING COLLEGE

CERTIFICATE

This is to certify that the project that is entitled with the name **"Deep Learning Dog Breed Identification and Classification"** is a bonafide, work done by the team **Sk. Zaheed Husain (21571A05C6), V. Rakesh Kumar(21471A05D5), P.Sai Kumar(21471A05B4)** in partialfulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING during 2024-2025.

PROJECT GUIDE
G. SARANYA M.Tech
Asst. Professor

PROJECT CO-ORDINATOR
Dr. M. Sireesha, M.Tech., Ph.D.
Assoc. Professor

HEAD OFTHE DEPARTMENT
Dr. S. N. Tirumala Rao,
M.Tech., Ph.D.,
Professor & HoD

EXTERNAL EXAMINER

DECLARATION

We declare that this project work titled “**Deep Learning Dog Breed Identification and Classification**” is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has not been submitted for any other degree or professional qualification except as specified.

By

Sk. Zaheed Husain	(21471A05C6)
V. Rakesh Kumar	(21471A05D5)
P. Sai Kumar	(21471A05B4)

ACKNOWLEDGEMENT

We wish to express our thanks to carious personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sir **M. V.Koteswara Rao**, **B.Sc.**, who took keen interest in us in every effort throughout thiscourse. We owe out sincere gratitude to our beloved principal **Dr. S. Venkateswarlu,** **Ph.D.**, for showing his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude towards **Dr. S. N. Tirumala Rao, M.Tech., Ph.D.,**HOD of CSE department and also to our guide **G. Saranya M.Tech Assistant Professor** of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks towards **Dr. M. Sireesha, M.Tech.,Ph.D.,** Associate professor & Project coordinator of the project for extending her encouragement. Her profound knowledge and willingness have been a constant sourceof inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who involved in giving valuable suggestions had clarifying our doubts which had really helped us in successfully completing our project.

By

Sk. Zaheed Husain	(21471A05C6)
V. Rakesh Kumar	(21471A05D5)
P. Sai Kumar	(21471A05B4)

NEC NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as researchfor the transformation of lives and community,

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems

NEC NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a center of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for researchand ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mold the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern softwaretools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

NEC NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Specific Outcomes (PSO’s)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

Program Educational Objectives (PEO's)

The graduates of the programmer are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems relatedto academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and cancommunicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA
NEC ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis:** Identify, formulate, research literature, and analyses complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, andsynthesis of the information to provide valid conclusions.

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineeringactivities with an understanding of the limitations.

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the

knowledge of, andneed for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilitiesand norms of the engineering practice.

9. **Individual and team work**: Function effectively as an individual, and as a member orleader in diverse teams, and in multidisciplinary settings.

10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and giveand receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of theengineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

NEC NARASARAOPETA ENGINEERING COLLEGE (AUTONOMOUS)

Project Course Outcomes (CO'S):

CO421.1: Analyses the System of Examinations and identify the problem.

CO421.2: Identify and classify the requirements.

CO421.3: Review the Related Literature

CO421.4: Design and Modularize the project

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1		√											√		
C421.2	√		√		√								√		
C421.3				√		√	√	√					√		
C421.4			√			√	√	√					√	√	
C421.5					√	√	√	√	√	√	√	√	√	√	√
C421.6									√	√	√		√	√	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. Low level
2. Medium level
3. High level

Project mapping with various courses of Curriculum with Attained PO's:

Name of the course from which principles are applied in this project	**Description of the device**	**Attained PO**
C2204.2, C22L3.2	Gathering the requirements and defining the problem, plan to develop model for recognizing image manipulations using CNN	PO1, PO3
CC421.1, C2204.3, C22L3.2	Each and every requirement is critically analyzed, the process modelis identified	PO2, PO3
CC421.2, C2204.2, C22L3.3	Logical design is done by using the unified modelling language which involves individual team work	PO3, PO5, PO9
CC421.3, C2204.3, C22L3.2	Each and every module is tested, iand evaluated in our project	PO1, PO5
CC421.4, C2204.4, C22L3.2	Documentation is done by all our four members in the form of a group	PO10
CC421.5, C2204.2, C22L3.3	Each and every phase of the work ingroup is presented periodically	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implementation of the project will be handled by the social media users and in future updates in our project can be donebased on detection of dog breed using pictures	PO4, PO7
C32SC4.3	The physical design includes website to check whether image is dog or not	PO5, PO6

ABSTRACT

This project will identify the specific dog breed from an image using Deep Learning and computer vision techniques. The aim is for the user to upload a picture of a dog and then the model should determine the breed of the dog from the list of 120 available breeds located in the dataset. Identifying dog breeds is fundamental in veterinary medicine, pet care, and dog welfare. However, single Deep Learning models and traditional methods have their fair share of challenges, especially regarding the accuracy that comes along with high diversity in the appearances of dogs. The proposed work conducts prediction of breed for dogs from Deep Learning modeling applying a number of strategies that include Xception, NASNetMobile, Inception and Xception. The current system made use of Inception-v3 and InceptionResNetV2 on Stanford Dogs Standard Datasets. The blended models are viewed to be fine combinations of NASNetMobile, Inception and Xception to and Inception-v3 and Xception. This model performs better than others single models which include Xception, InceptionV3, ResNet50 and ResNet101. To enhance the results, the author used a transfer learning approach with data enlargement. The Accuracy which are obtained were 86.92 and 79.69 Validation accuracy scores against InceptionV3 this ad InceptionV3 respectively. The most accuracy of InceptionResNetV2 was 84.92 in the present system. By applying the proposed algorithms the validation accuracy of Inception- ResNetV2 was 84.92, 77.68 of Xception, 83.22 of NASNetMobile, 79.38 of Inception V3. Whichever of the Hybrid of the Inception-v3 Xception method is the last type in the combination of ordinary measurement does offers a level of accuracy that is unsurpassed amongst these algorithm coming in at 92.4.

INDEX

S.NO **CONTENT** **PAGE NO**

LIST OF FIGURES

1. Introduction

Learning about dog breed is critical understanding information for responsible pet ownership, veterinary care, and the management of the community. Knowing a dog’s breed can be highly informative regarding how the dog is likely to behave and its special care requirements. Selling particular breeds of dogs is done by targeting certain characteristics such as energy levels, amount of grooming required and even their health risks. This will allow the owners to understand what training procedures, exercises, and even meals are necessary for the dogs according to their breed Hence the dog gets the attention and help it requires to flourish.

In the [1], a Convolutional Neural Network (CNN) and model like Inception-V2, inception Resnet-v3, Resnet101, Resnet50 are begin used to determine the characteristic of the dog and to gain higher accuracy. The highest accuracy gained by the author is 90 in Resnet model, the author pretrained the datasets and convert the data in to array to gain best accuracy, the author also compare various model with his accuracy to gain more data

In public domain or legislation, identification of the breed of a dog is critical especially when the dog is involved in a dog attack or display of aggression. Recognition of the breed is very important in predicting the tendencies and aggressiveness of different types of dogs, especially those regarded as potentially dangerous. In most parts of the world, regulation of specific breeds is done, because of the risks the breeds pose, or are believed to pose. Identification in such a situation would then assist in dealing with a situation that would otherwise provoke action by law enforcers, insurance companies, or managers of a community who may use the legislation to discriminate and prevent acts of aggression.

TensorFlow is Google’s open source software machine learningtool that is utilized in constructing and training neural networks. The platformprovides all layers, all kinds of optimizers and data management, helping its users with the creation of the AI models. Keras: It is a high level API available in tensor flow which is used to build and train Deep Learning models. It provides opportunities to experiment with various network configurations.

Matplotlib is a large broad library for static charts, animated animation and interactive visualization in Python. It is also common practice among the developers to create plots showing data and model performance including scatter plots, line diagrams, and histograms with its help. Seaborn: Seaborn is based on Matplotlib but extends its possibilities to go beyond just producing informative and beautiful statistical graphics. In [3], the author did not use the usual method to determine the dog breed. He rather worried about the cyber threat on the proposed system and he build a (AI) and (ML) machine to identify the dog breed using Deep Learning. The development of center significance of the technologies of recognition of images and information security is a further opportunity for authentication and identification. This project paper aims at making a thorough investigation. The pie chart show the population of the dog breed of different breed it is show in Fig 1.1 .It is also a common tool to make sense of complex statistics and their intricacies, and the relationships of the variables at hand.

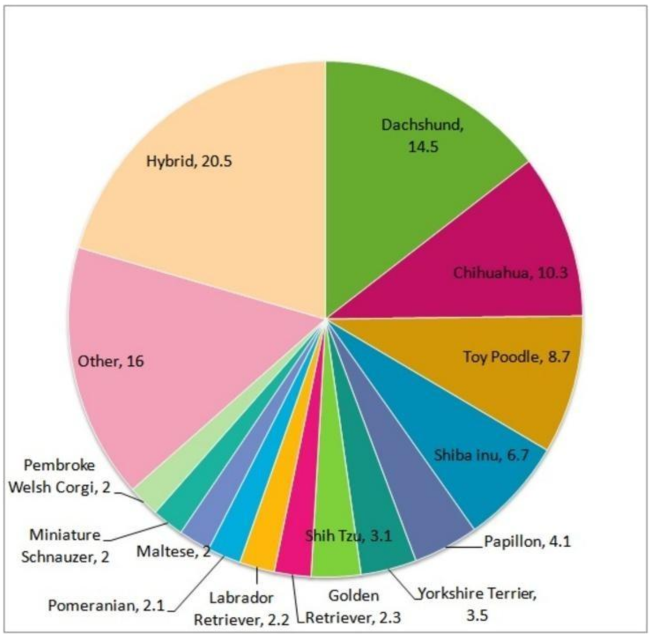


Fig1.1: pie chart of different dog breed

OpenCV: It is a library that is free and open source and that comprises functions for computer vision as well as methods for real time image processing. It is normally used for photo preparation, object detection, and more in combination with machine learning models.

The ImageDataGenerator class in Keras is capable of generating batch tensor image data with real time data augmentation. This is very helpful for increasing the diversity of the training dataset by using random image manipulations like rotating, flipping, and zooming in. Glob: The glob module is used to obtain all the folder and or file names that match a particular specification. This as well helps in the processing of multiple files at a time for example when loading many images from one folder. We used a dataset of 120 dog's breeds are used to train and test the Deep Learning model's [4]. The dataset's were taken from the kaggle which included large driver and unique images of digs in different angle and position, it containing approximately 10,222 images of 120 category of dog breeds. Then the author used Convolutional Neural Network, Mobilenetv2 was one the best choice of the author which provided 84 accuracy.

CV2 (OpenCV):These involve pre-processing of the images to be used by client views such as resizing and converting images where this step is important before the images are fed into the neural networks. TensorFlow Callbacks: Callbacks such as EarlyStopping and ReduceLROnPlateau function during the course of training the model in order to optimize its performance by implementing the right learning parameters efforts into over fitting or loss prevention.

This domain would be categorized as Computer Science and Artificial Intelligence (AI) and more specifically under the topics of Machine Learning (ML), Deep Learning, and Computer Vision. Such a domain works towards use of AI based approaches to perform categories of different breeds of dogs using images. This study may further discuss Pattern Recognition and Data Science since there is a possibility of implementing many techniques and drawing conclusions based on their comparisons.

To address the urgent need for the protection and understanding of dog breeds, the first step is accurate identification and comprehension of different breeds and their traits. However, with hundreds of recognized dog breeds worldwide and countless mixed breeds, manual identification by experts is time-consuming and often impractical, especially for non-specialists. This underscores the necessity for an automated dog breed identification tool that leverages advancements in technology, particularly image-based methods.

Such a tool would enable users to identify dog breeds rapidly and accurately using images captured with mobile devices equipped with built-in cameras.

This project seeks to contribute to the advancement of automated dog breed identification by employing Deep Learning architectures, specifically convolutional neural networks (CNNs). Deep Learning models have demonstrated remarkable success in various image-based applications, including traffic detection, medical image recognition, and facial recognition. By leveraging the power of CNNs, this research aims to develop a robust and accurate dog breed identification system that can aid pet owners, veterinarians, researchers, and enthusiasts in identifying breeds and understanding breed-specific characteristics.

In summary, the accurate identification of dog breeds is essential for promoting responsible pet ownership, supporting veterinary care, and enhancing breed-specific knowledge. Automated dog breed identification tools hold immense promise in improving awareness and empowering individuals to care for and understand their pets better. By harnessing the capabilities of Deep Learning models like CNNs, this project endeavours to advance the field of automated breed identification and facilitate informed decision-making for pet care, breed preservation, and research.

An automated dog breed identification tool holds immense promise in democratizing access to canine knowledge and fostering widespread participation in responsible pet ownership and breeding practices. By harnessing the capabilities of portable devices equipped with high-resolution cameras and precise sensors, users can seamlessly capture images of dogs in various settings and analyze them using recognition applications. Such tools not only facilitate rapid breed identification but also provide valuable insights into breed-specific traits, health predispositions, and behavior patterns.

In conclusion, the imperative to protect and understand dog breeds underscores the critical need for automated dog breed identification tools. By harnessing the power of Deep Learning architectures and image-based methods, researchers and organizations are poised to revolutionize the field of canine identification. These advancements not only facilitate the rapid and accurate identification of dog breeds but also empower individuals to actively engage in responsible pet care and breed preservation. As technology continues to evolve, the future holds tremendous promise for automated dog breed identification, offering unprecedented opportunities for veterinary research, canine health monitoring, and pet care innovation.

2. Literature Survey

2.1 Deep Learning

Deep Learning is a subset of Machine Learning, which itself is a branch of Artificial Intelligence (AI). [5] The Author make use of the pre-trained model to take out the attribute from Dog breeds identification data set. Later on, they applied fine tuning and data augmentation to improving the test accuracy of the categorization of dog breeds datasets. The analysis of the proposed approaches to the analytic normal is based on the existing imagenet datasets like DenseNet-121, ResNet-50, DenseNet-169, Google Net were the authors who achieved test accuracy of 89.66, 85.37, 84.01 and 82.08 respectively bore well as dog learn images classification It focuses on using artificial neural networks inspired by the structure and functioning of the human brain to process and analyze data. These networks consist of multiple layers, enabling Deep Learning models to learn hierarchical representations of data and solve complex problems with remarkable accuracy.

2.2 Deep Learning Applications

Computer Vision: Deep Learning excels in analyzing and interpreting visual data, enabling numerous applications in computer vision.

Natural Language Processing (NLP): Deep Learning has transformed the way machines understand and generate human language.

Text Translation: Automatically translating text from one language to another. Example: Google Translate.

Healthcare and Medicine: Deep Learning has shown significant promise in improving diagnostics, treatment planning, and drug discovery.

2.3 Convolutional Neural Network

Convolutional Neural Networks (CNNs) combine concepts from biology, mathematics, and computer science to achieve groundbreaking advancements in computer vision. Since Alex Krizhevsky's remarkable performance in the 2012 ImageNet competition, where CNNs reduced classification error from 26% to 15%, they have become essential tools in the field. Today, CNNs are widely adopted for diverse applications such as automatic tagging (Facebook), photo search (Google), and personalized recommendations (Amazon and Pinterest). In this context, CNNs can also play a critical role in the automatic identification of dog breeds, enabling systems to analyze images and classify specific breeds accurately.

Dog-specific identification involves classifying images of dogs to determine their breed or a mix of breeds. [6] used different dog breeds and human images using CNN. It consist of 13,233 human images and 8,351 dog images. If a dog image was given, the model or proposed model would search for that dog class. If a human picture was given, it will determined which facial features would show in a dog.

Problem Space:

For humans, recognizing a dog’s breed is often effortless, drawing from years of exposure and prior knowledge. Machines, however, require a structured approach to learn patterns, differentiate features, and adapt to the variety of dog breeds in images.

This task is particularly relevant in areas such as:

Veterinary care and breed-specific medical guidance.

Pet adoption systems and shelters.

Personalized recommendations for dog owners (nutrition, accessories, and training).

Research on canine behavior and genetics.

When a computer processes an image of a dog:

Input: The image is represented as an array of pixel values.

For instance, a 480 x 480 color image is represented as a 480 x 480 x 3 array, with the "3" denoting RGB color channels.

Pixel intensity values range from 0 to 255.

Output: The model generates probabilities for various dog breeds. For example:

0.70 probability for "Golden Retriever"

0.20 probability for "Labrador"

0.10 probability for "Border Collie"

2.4 Importance of CNN

CNNs offer several key advantages that have propelled them to the forefront of DeepLearning: Superior Feature Extraction: Traditional machine learning algorithms often require manual feature engineering, a time-consuming and domain-specific process. CNNs, however, excel at automatically learning relevant features directly from the data, significantly reducing the need for manual intervention High Accuracy in Image Recognition: Their ability to capture spatial relationships within images makes CNNs remarkably adept at recognizing objects, scenes, and even faces with high accuracy This has revolutionized fields like computer vision and image classification Scalability for Large Datasets:

[8], wants to show the way of overcoming this problem and concerns. The proposed method used by the author is Deep Learning based where dogs and their breeds are identified. This approach begins with a transfer learning and specific pre-trained Convolutional Neural Networks (CNNs) are retrained on the public dataset of dog breeds. The proposed model comprises of 133 classes of dog breeds and it factors whose accuracy performance of the model almost approaches is 89.92 dog breeds dataset

CNNs are designed to handle large datasets efficiently. As the amount of data available increases, so too does the effectiveness of CNNs, allowing them to continuously learn and improve their performance Generalizability: While initially designed for images, the core principles of CNNs have been adapted to other grid-like data structures. This flexibility has opened doors for their application in diverse fields like natural language processing and time series analysis These benefits, coupled with advancements in computing power and the availability of vast datasets, have propelled CNNs to the forefront of various technological advancements A Spectrum of Applications: Where CNNs Shine The impact of CNNs is felt across a vast array of fields, transforming how we interact with technology and solve complex problems. Here's a closer look at some prominent applications

Computer Vision and Image Recognition:

The most established domain for CNNs is computer vision.They power a wide range of applications, including: Object Detection and Recognition: CNNs can identify objects within images with high accuracy, enabling tasks like self-driving cars recognizing pedestrians and traffic signs, or facial recognition systems used for security purposes. Image Classification: From categorizing images for content moderation on social media platforms to classifying medical scans for disease detection, CNNs excel at classifying images into specific categories. Image Segmentation: CNNs can be used to segment images, separating objects from the background or identifying specific regions within an image. This is crucial for tasks like medical image analysis or autonomous vehicle navigation

2.5 importance of Deep Learning using python

Deep Learning, a subset of machine learning, has revolutionized the field of artificial intelligence (AI) by enabling machines to learn complex patterns and make predictions with unprecedented accuracy. Python, as a programming language, plays a pivotal role in the development and implementation of Deep Learning models due to its simplicity, versatility, and a rich ecosystem of libraries. Here, we explore the importance of using Python for Deep Learning in various domains and applications.

a. **Simplicity and Ease of Use**

Python is renowned for its intuitive syntax and readability, which makes it accessible to both beginners and experts in AI development. This simplicity:

- Reduces the learning curve for newcomers to Deep Learning.
- Facilitates rapid prototyping and experimentation.
- Enhances collaboration between data scientists, researchers, and software engineers.

b. Comprehensive Libraries and Frameworks

Python boasts a robust ecosystem of libraries and frameworks specifically designed for Deep Learning. These libraries streamline the implementation of complex neural networks, enabling developers to focus on innovation rather than coding from scratch. Key libraries include:

- **TensorFlow:** A powerful library for large-scale machine learning and Deep Learning applications.
- **PyTorch:** Known for its flexibility and dynamic computation graph, ideal for research and production.
- **Keras:** A high-level API that simplifies building and training neural networks.
- **scikit-learn:** Complements Deep Learning by providing tools for preprocessing and evaluation.

c. Extensive Communi ty Support

The Python community is one of the largest and most active in the programming world. This vibrant community:

- Offers extensive documentation and tutorials for Deep Learning libraries.
- Provides open-source implementations of state-of-the-art models.
- Encourages collaboration through forums, GitHub repositories, and online courses.

d. Versatility and Integration

Python seamlessly integrates with other technologies and tools, making it versatile for various Deep Learning tasks. It can be used for:

- **Data Preprocessing:** Libraries like Pandas and NumPy handle large datasets efficiently.
- **Visualization:** Matplotlib and Seaborn enable the graphical representation of model performance.
- **Deployment:** Flask and FastAPI help deploy trained Deep Learning models as web applications.

e. Broad Range of Applications

Python's compatibility with Deep Learning libraries facilitates its use across diverse fields, including:

- **Healthcare:** Medical image analysis, drug discovery, and disease diagnosis.
- **Finance:** Fraud detection, algorithmic trading, and risk assessment.
- **Autonomous Systems:** Self-driving cars, robotics, and drones.
- **Natural Language Processing (NLP):** Sentiment analysis, chatbots, and machine translation.
- **Computer Vision:** Object detection, facial recognition, and video analysis.

3. Existing System

Traditional methods for dog breed identification rely on handcrafted feature extraction and classical machine learning techniques. These methods involve manually extracting visual features, such as color patterns, fur texture, and body proportions, and then classifying them using algorithms like Support Vector Machines (SVM), k-Nearest Neighbors (KNN), or other conventional classifiers. However, these approaches face several limitations, such as the inability to capture complex patterns, reliance on predefined features, and reduced accuracy when dealing with diverse or overlapping breed characteristics.

Manual Feature Extraction: Requires domain expertise to select meaningful features, making it time-consuming and prone to human error.

Limited Accuracy: Traditional machine learning models are often unable to capture the characterstic of the dog breed, leading to lower classification performance.

Lack of Generalization: Models trained on specific datasets may not generalize well to new, unseen data due to variations in recording conditions, quality, size of the differences pictures.

Difficulty in Handling Large Datasets: As datasets grow larger, traditional approaches struggle to scale effectively, leading to computational inefficiencies.

Feature Engineering Challenges: Traditional methods require extensive feature engineering, making them less adaptable to complex variations in heart sound signals.

High False Positives and Negatives: Misclassification rates are higher due to the limited ability of classical approaches to differentiate between same type of dog but different breed.

4. Proposed Methodology

To overcome the limitations of traditional methods, this study employs a Deep Learning-based approach using Convolutional Neural Networks (CNNs) to classify PCG signals. In The proposed methodology figure 2 it consists of the following key steps:

1. Data Collection.
2. Data Pre-processing.
3. CNN Architecture.
4. Training Process.
5. Performance Evaluation.

The following flowchart describes the research paradigm in this study:

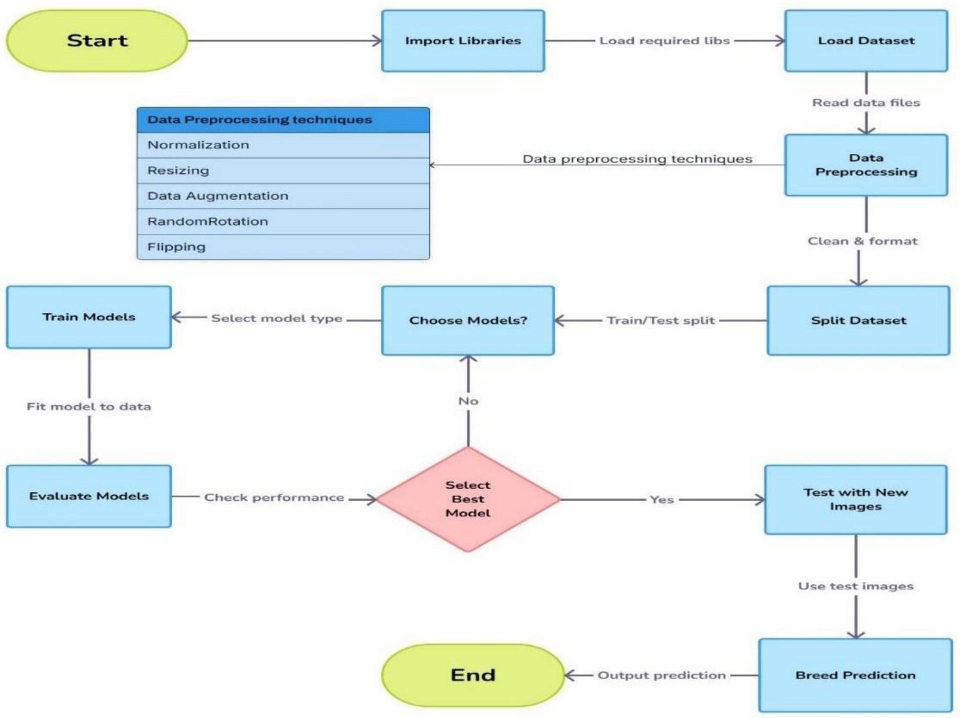


Fig 4.1: Flow Chart of proposed methodology

5. System Requirements

5.1 Hardware Requirements:

• System Type	: intel®core™i3-7500UCPU@2.40gh
• Cache memory	: 4 (Megabyte)
• RAM	: 8 (gigabyte) OR Higher
• Hard Disk	: 256 (gigabyte) OR Higher

5.2 Software Requirements:

• Operating System	: Windows 11, 64-bit Operating System
• Coding Language	: Python
• Python distribution	: Anaconda, Google colab
• Browser Chrome	: Any Latest Browser like

6. System Analysis

6.1 Scope of the Project:

The scope of the project revolves around the development of an accurate and efficient dog breed identification system using machine learning techniques. This system aims to classify dog breeds based on images, providing a reliable tool for applications such as pet adoption, veterinary care, and animal control. The project focuses on leveraging Deep Learning methodologies to address challenges associated with breed diversity, mixed breeds, and visually similar breeds. Additionally, the system is intended to perform efficiently across various devices, including mobile platforms, enabling widespread accessibility and usability.

6.2 Analysis:

It involves understanding the requirements and challenges of dog breed identification, including the diversity of dog breeds, variations in image quality, and the potential for similar-looking breeds to cause misclassification. It also includes reviewing existing solutions and identifying their limitations, such as biases in training datasets and the lack of generalization. This phase sets the foundation for selecting appropriate models, algorithms, and techniques to overcome these challenges and achieve robust breed identification.

These will analyze the four models and find the most accurate and efficient. Existing proposed models include Xception, NasNet Mobile, InceptionResNetV2, Inception v3. Consider whether ethical issues could arise from judgment calls rooted in breed-type prejudice or classifications that pertain to disability. In contrast to classical residuals that first combine features and then reduce their dimensionality, Inverted residuals are incorporated in MobileNetV2 which is a high performing Convolutional Neural Network architecture model designed for mobile and embedded device. The configuration of the system which is being used is 64 bits and i3, i5 core in the google colab platform. In Fig 6.1 it shows several dog breed images and steps will be taken through to wherever this task will be completed as long as it is to achieve the best possible results.

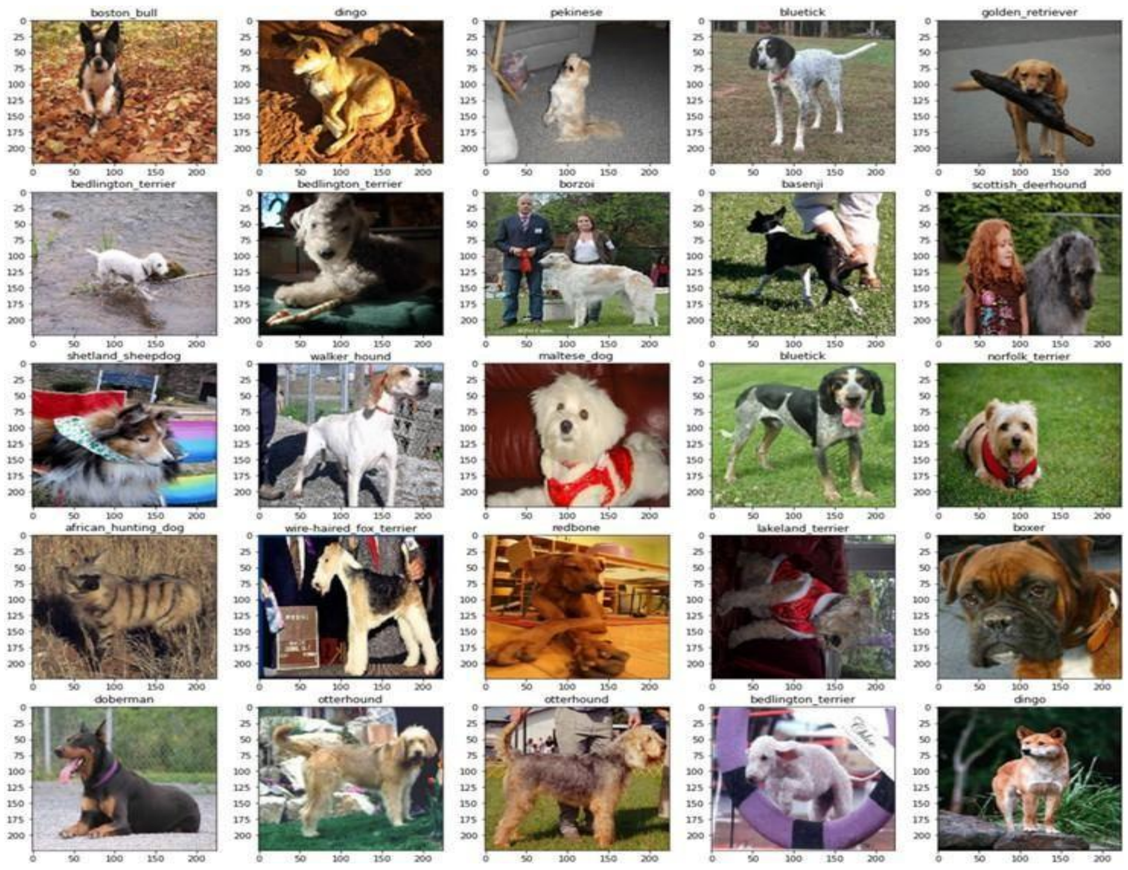


Fig 6.1: Different breed sample images

6.3 Data collection:

The dataset consists of 10,222 labeled dog images, categorized into specific breeds for classification purposes. These images serve as training and validation data for the CNN model. The dataset breakdown includes:

Training Data: 10,222 images (7,222 images of common breeds, 3000 images of rare or mixed breeds)

Validation Data: 200+ images (100 images of common breeds, 100 images of rare or mixed breeds)

The total images of individual dog breed is given in Fig 6.2 in the form of bar chart

To ensure a high level of model generalization, the dataset is curated with diverse dog images collected from multiple sources, including pet adoption platforms, animal shelters, and open-access image repositories.

Ensuring balanced class distribution is crucial for minimizing bias in model predictions. Data augmentation techniques, such as rotation, flipping, scaling, and brightness adjustments, are employed to enhance model robustness by artificially increasing dataset diversity. The dataset preprocessing ensures that images

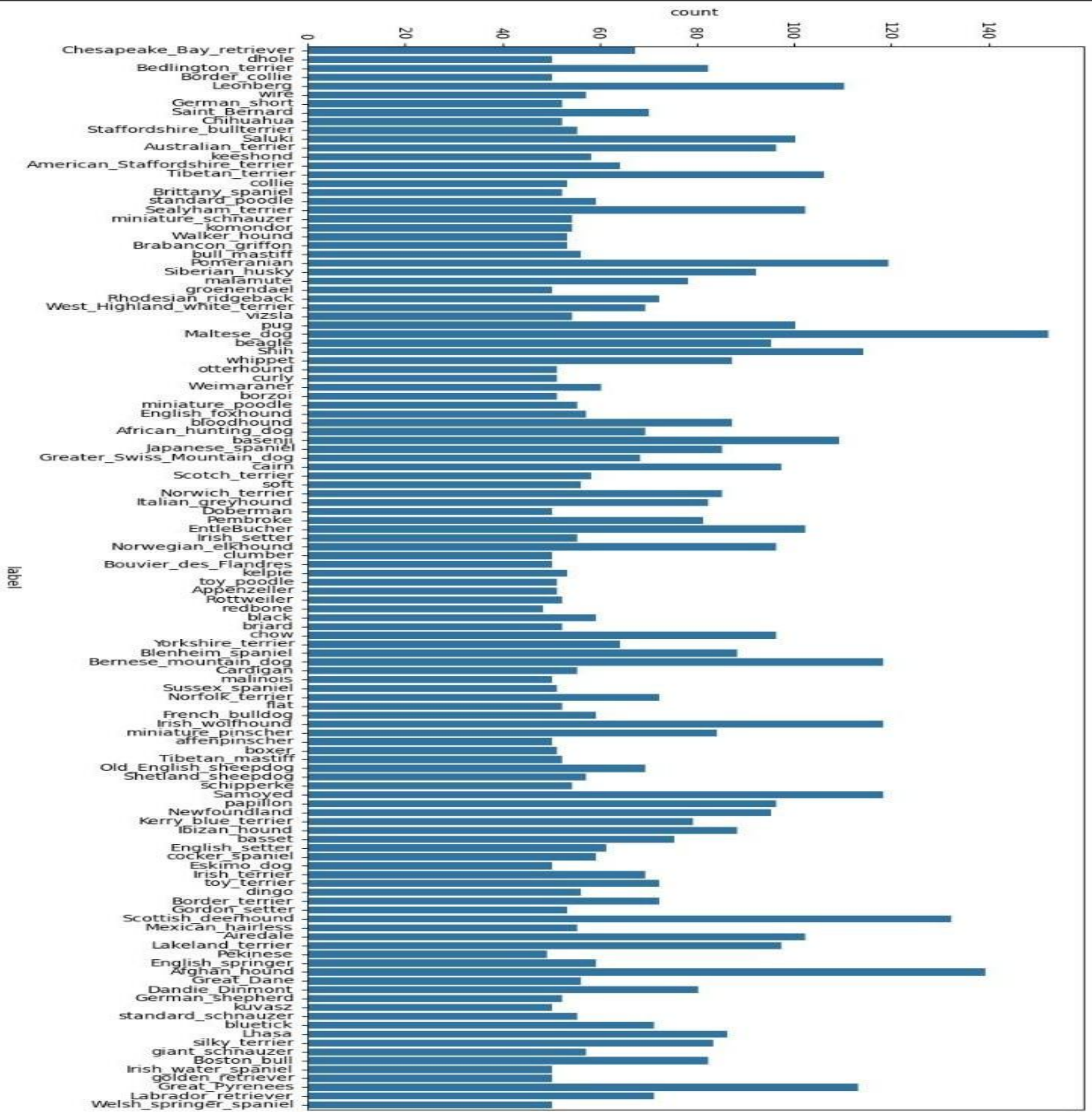


Fig 6.2: Bar chart of the Dataset.

affected by poor lighting, cluttered backgrounds, or varying camera angles do not introduce artifacts into the training set, enabling the model to perform reliably across diverse real-world scenarios.

6.4 Data Preprocessing:

To optimize the CNN model's performance, several preprocessing techniques are applied:

Normalization: Scales pixel values between 0 and 1 for uniform input, improving model convergence.

Resizing: Adjusts images to 150×150 pixels for compatibility with the CNN architecture, ensuring consistency in input dimensions.

Data Augmentation: Enhances dataset variability through transformations like flipping, rotation, and zooming, thereby improving model robustness against variations in heart sound recordings.

6.5 Model Building:

The definition of the model involves the selection of the appropriate architecture and design, ensuring it is well-suited for the task at hand. For dog breed identification, convolutional neural networks (CNNs) are the go-to choice due to their ability to process images effectively. However, the design of the CNN, including the number of layers, layer types (e.g., convolutional, pooling, fully connected), and the activation functions, must be fine-tuned for optimal performance. The model should be deep enough to capture the intricate details that differentiate breeds but also not so complex that it leads to unnecessary computation or overfitting. In some cases, transfer learning, where a pre-trained model on a large dataset (like ImageNet) is adapted to the specific task of breed classification, can be highly effective in defining the model. Additionally, the model must be capable of learning from a diverse range of images, including those with variations in lighting, angles, and dog poses.

Xception model: is a Deep Learning architectural model developed by Francois Chollet in the year 2017 which is a more advanced form of the basic Inception model by employing the depth wise separable convolution. These convolutions break down the standard convolution procedure into two basic operations, the first is the depth wise convolution in which each filter is applied to an input channel singly and the second is the point wise convolution in which the depth wise convolution's outputs are integrated with a 1x1 convolution on all input channels. This helps to minimize computation and reduce the number of parameters thus making the models more efficient.

NASNetMobile: Google developed the NASNet model, which stands for Neural Architecture Search Network, in the year 2017 and is a Deep Learning model that has been designed through Neural Architecture Search (NAS). With reference to NASNet, these methods were searched in order to provide the best building type cells to be used. Which could later on be contoured into the final framework. The NASNet models thus resulting have great efficiency and scalability and offer advanced performance on tasks like image classification, object recognition, etc. Importantly, these NASNet models have the architectural flexibility of being adapted to different levels of computation to maintain a reasonable accuracy relative to other architectures.

InceptionResNetV2: The Inception-ResNet-v2 model, which is a neural network based system, fuses the aspects of two different Deep Learning architectures Inception and ResNet. The introduction of the residual connections also enables the network to learn some identity functions as well; hence speeding up the training process. By using the depth, width, and computational efficiency, Inception-ResNet-v2 has been able to obtain state of the art results in image recognition applications. In particular, it has been reported that its effectiveness is optimal for large scale datasets such as ImageNet. This kind of structure has been embraced in computer vision problems because of its excellent performance as well as expandability.

InceptionV3: Inception v3 model is a C2D architecture Deep Learning model and it is the third generation of the poses series which was developed by Google to solve the problem of stereoscopic recognition. Being one of the latest models addition in 2015. The model uses "Inception modules," a technique that helps in developing the network by reducing the cost of running the network using multiple convolutions of different sizes aimed at obtaining multi scale features. The key updates included in Inception V3 are so called factorized convolutions, when standard large convolutions are expressed with a series of smaller, efficient ones, batch normalization to make the training stable and faster. Thanks to the sophisticated architectural design of Inception V3 the image classification tasks like ImageNet was established and is still one of the preferred options for many computer vision tasks

6.6 Training Models:

Model training is the process of teaching the machine learning model to recognize patterns and make predictions based on the data it is provided. For dog breed identification, this process involves feeding the model a large, labeled dataset of dog images. During training, the model adjusts its internal parameters to minimize the difference between its predicted output and the true breed label. This requires multiple iterations (epochs) of forward and backward propagation, where the model’s predictions are compared to the ground truth, and errors are propagated back to adjust the weights. Effective training requires a combination of a high-quality, diverse dataset and fine-tuning of hyperparameters such as learning rate, batch size, and the number of epochs. Overfitting is a major concern, and techniques like dropout, early stopping, and data augmentation are often employed to mitigate this risk. Additionally, model validation using a separate dataset helps to ensure that the model generalizes well to new, unseen images.

6.7 Optimizer:

The optimizer plays a critical role in adjusting the model’s weights during the training process to minimize the error or loss function. The choice of optimizer influences how effectively the model learns from the data and reaches convergence. Commonly used optimizers like Adam, SGD (Stochastic Gradient Descent), or RMSprop are fine-tuned to balance the speed of convergence with the model's ability to generalize well to unseen data. In the context of dog breed identification, a well-chosen optimizer ensures that the model can efficiently handle the large number of possible dog breeds and variations, reducing the risk of overfitting while improving its accuracy. By selecting the right learning rate and momentum parameters, the optimizer can guide the model toward better solutions while preventing it from getting stuck in local minima, which is especially important for complex tasks like breed identification.

Adam (Adaptive Moment Estimation): Adam is one of the most popular optimizers, combining the advantages of both RMSProp and Momentum (another optimization technique that accelerates SGD). It computes adaptive learning rates for each parameter, and it includes momentum by keeping a running average of both the gradients

Number of Epochs: The algorithm is executed multiple times on the training dataset. In this case, we train the model for 10 epochs.

Batch Size: Batch size is among the most important hyper parameters in the training of neural networks. It defines the number of examples that are used in one model iteration or one forward and backward pass of the model. Batch Size gives the number of samples to update the model parameters. In this model, the batch size used is 250.

Learning rate: Learning rate is the scaling factor that is needed for updating the model weights. The proposed model uses different optimizers with different learning rates. It is a hyper parameter, which has a greater influence on both the improvement of training and activity of model which is based on the assessment metrics.

7. Design

The image processing and neural network workflow involves gathering and preprocessing images for training and testing, including steps like resizing, grayscale conversion, and error analysis with techniques like Error Level Analysis (ELA). Fig 7.1 Data is split into training and testing sets, and images are converted into pixel arrays for neural network processing. The model is then trained on the training set, and its performance is evaluated on the testing set. Metrics such as accuracy, precision, and recall are calculated, and the trained network can be saved for future use. Finally, a performance report is generated, and results are visually displayed to understand the model's performance, including correctly and misclassified images.

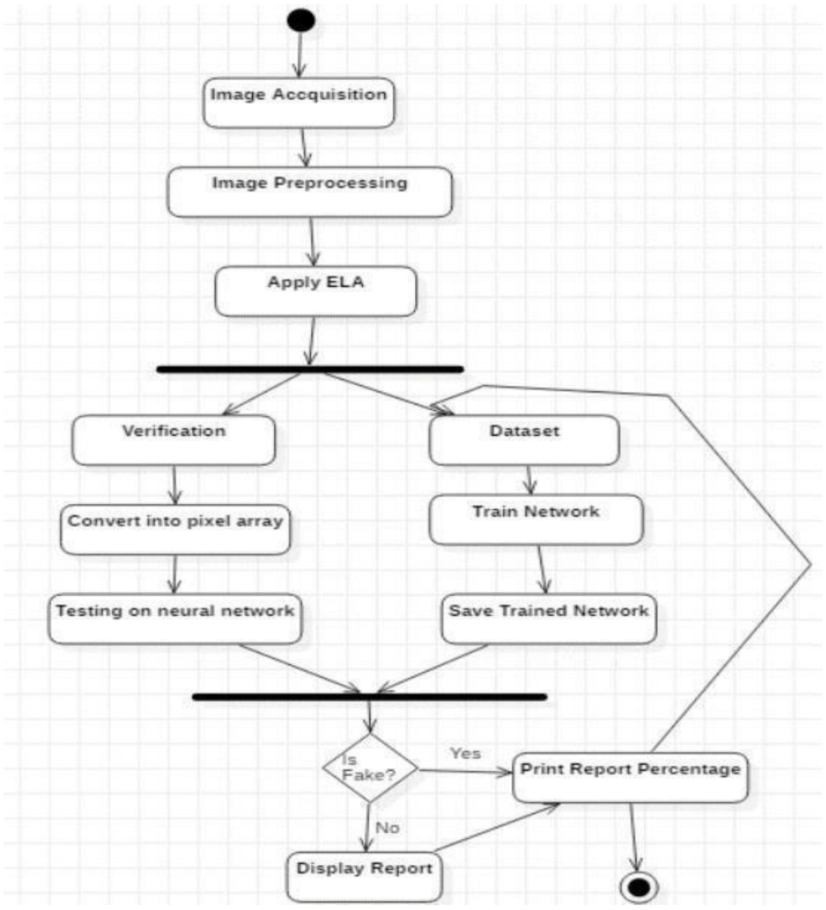


Fig 7.1: Design overview

8. Implementation

```
## import modules
import tensorflow as tf
from tensorflow.keras.preprocessing.image import
ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D,
Flatten, Dense, BatchNormalization, Activation
from tensorflow.keras.optimizers import SGD, Adagrad, Adadelta,
RMSprop, Adam
from sklearn.utils.class_weight import compute_class_weight
import numpy as np

import numpy as np
import pandas as pd
import seaborn as sns
import tensorflow as tf
import matplotlib.pyplot as plt
import warnings
warnings.simplefilter('ignore')

from matplotlib.pyplot import imshow
from keras.preprocessing import image
from keras import applications
import os
import glob

import cv2
import warnings
warnings.simplefilter('ignore')

import keras
from keras import layers
from keras.layers import
Input,Dense,Activation,ZeroPadding2D,BatchNormalization,Flatten
,Conv2D
from keras.layers import
AveragePooling2D,GlobalAveragePooling2D,GlobalMaxPool2D,MaxPool
ing2D,MaxPool2D,Dropout
from keras.models import Model,Sequential

## Loading the dataset

from google.colab import drive
drive.mount('/content/drive')
```

```
!unzip '/content/drive/MyDrive/project/archive (1).zip'

os.listdir('/content/cropped')

glob.glob('/content/cropped/*')

glob.glob('/content/cropped/cropped/*')

glob.glob('/content/cropped/cropped/train/n02112706-
Brabancon_griffon/*')
```

## Analyze and Visualize the Dataset

```
class_dir1=os.listdir('/content/cropped/cropped/test/')
class_dir1

dog_names1 = [file.split('-')[1] for file in class_dir1]
dog_names1

import random
import glob
import tensorflow as tf

count_dict1 = {}
img_dict1 = {}

# Loop through classes
for cls in class_dir1:  # Assuming class_names contains the
list of dog classes
    image_path =
glob.glob(f'/content/cropped/cropped/test/{cls}/*')
    count_dict1[cls] = len(image_path)

    if image_path:  # Check if image_path is not empty
        img_dict1[cls] =
tf.keras.utils.load_img(random.choice(image_path))
count_dict1

df1 =
pd.DataFrame(data={'label':count_dict1.keys(),'count':count_dic
t1.values()})
df1 = df1[df1['count'] != 0]
df1

labels=list(df1['label'])
labels

img = cv2.imread('/content/cropped/cropped/train/n02085782-
Japanese_spaniel/n02085782_1059.jpg')
print(img.shape)
img1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
plt.imshow(img1)

img = cv2.imread('/content/cropped/cropped/test/n02085936-
Maltese_dog/n02085936_10130.jpg')
print(img.shape)
img1 = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
plt.imshow(img1)

import math

num_items = len(img_dict1)
num_cols = 4
num_rows = math.ceil(num_items / num_cols)

plt.figure(figsize=(20, 5 * num_rows))  # Adjust the figure
size based on the number of rows

for id, (label, img) in enumerate(img_dict1.items()):
    plt.subplot(num_rows, num_cols, id + 1)
    plt.imshow(img)
    plt.title(f"{label.split('-')[1]} {img.size}")
    plt.axis('off')

df1['label']=[file.split('-')[1] for file in labels]
plt.figure(figsize=(20,8))
sns.barplot(x='label',y='count',data=df1)
plt.xticks(rotation=90)
plt.show()
```

## Preprocessing

```
train_datagen =  ImageDataGenerator(rescale=1./255,
                                    shear_range=0.15,
                                    rotation_range=20,
                                    width_shift_range=0.1,
                                    zoom_range=0.2,
                                    horizontal_flip=True,
                                    fill_mode='nearest',
                                    height_shift_range=0.1
                                    )
test_datagen = ImageDataGenerator(rescale=1./255)
train_set =
train_datagen.flow_from_directory('/content/cropped/cropped/tra
in',

target_size=(224,224),
 batch_size=32,class_mode='categorical')

test_set =
train_datagen.flow_from_directory('/content/cropped/cropped/tes
t',target_size=(224,224),batch_size=32,class_mode='categorical')
num_classes = len(train_set.class_indices)
print("Number of classes:", num_classes)

train_set.class_indices

# Calculate class weights
y_train = train_set.classes
class_weights = compute_class_weight(class_weight='balanced',
classes=np.unique(y_train), y=y_train)
class_weights = dict(enumerate(class_weights))
print(class_weights)

## Xecption

# Import libaries
from keras.applications import Xception
from keras.optimizers import RMSprop
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
# from keras.preprocessing.image import ImageDataGenerator

# Create Xception base model
from keras.applications.xception import Xception
base_model_xception = Xception(weights='imagenet',
include_top=False, input_shape=(224,224,3))

# Assuming you have a pre-trained Xception model as
base_model_xception
model_xception = Sequential()
model_xception.add(base_model_xception)
model_xception.add(GlobalAveragePooling2D())
model_xception.add(Dense(1024, activation='relu'))
model_xception.add(Dropout(0.5))
model_xception.add(Dense(120, activation='softmax'))
```

```
# Define RMSprop optimizer with specific parameters
optimizer = RMSprop(learning_rate=0.001, rho=0.9, epsilon=1e-
07)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.1,
patience=3, min_lr=1e-7)

# Compile the model with the RMSprop optimizer
model_xception.compile(optimizer=optimizer,
loss='categorical_crossentropy',
metrics=['accuracy','Precision','Recall'])

from keras.callbacks import EarlyStopping
early_stopping = EarlyStopping(monitor='val_loss', patience=5,
restore_best_weights=True)

history=model_xception.fit(train_set,epochs=10,validation_data=
test_set,callbacks=[early_stopping, reduce_lr])

# Plot training and validation accuracy/loss
plt.figure(figsize=(12, 6))
plt.plot(history.history['accuracy'], label='Training
Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation
Accuracy')
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Accuracy/Loss')
plt.xlabel('Epoch')
plt.ylabel('Accuracy/Loss')
plt.legend()
plt.show()
## NASnetmobile

import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import
ImageDataGenerator
from tensorflow.keras.applications import NASNetMobile
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense,
GlobalAveragePooling2D, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ReduceLROnPlateau,
EarlyStopping, ModelCheckpoint
import matplotlib.pyplot as plt


base_model = NASNetMobile(weights='imagenet',
include_top=False, input_shape=(224, 224, 3))
```

```
# Add custom layers on top of the base model
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(512, activation='relu')(x)
x = Dropout(0.5)(x)
predictions = Dense(train_set.num_classes,
activation='softmax')(x)

# Combine the base model with the custom layers
model = Model(inputs=base_model.input, outputs=predictions)

# Freeze the base model layers
for layer in base_model.layers:
    layer.trainable = False

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.0001),
loss='categorical_crossentropy', metrics=['accuracy'])

reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2,
patience=5, min_lr=0.00001)
early_stopping = EarlyStopping(monitor='val_loss', patience=10,
restore_best_weights=True)
model_checkpoint =
ModelCheckpoint('best_nasnetmobile_model.h5.keras',
monitor='val_loss', save_best_only=True)
history = model.fit(
    train_set,
    epochs=10,
    validation_data=test_set,
    callbacks=[reduce_lr, early_stopping, model_checkpoint]
)
# Load the best model
best_model =
tf.keras.models.load_model('best_nasnetmobile_model.h5')

# Evaluate on the validation set
val_loss, val_accuracy = best_model.evaluate(test_set)
print(f'Validation loss: {val_loss}')
print(f'Validation accuracy: {val_accuracy}')


# Plot training and validation accuracy/loss
plt.figure(figsize=(12, 6))
plt.plot(history.history['accuracy'], label='Training
Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation
Accuracy')
```

```
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Accuracy/Loss')
plt.xlabel('Epoch')
plt.ylabel('Accuracy/Loss')
plt.legend()
plt.show()

## Inception ResNetv2

import tensorflow as tf
from tensorflow.keras.preprocessing.image import
ImageDataGenerator

# Load InceptionResNetV2 model pre-trained on ImageNet
base_model = tf.keras.applications.InceptionResNetV2(
    include_top=False,
    weights='imagenet',
    input_shape=(224,224,3)
)

# Freeze the base model layers
base_model.trainable = False

# Add classification head to the base model
model = tf.keras.Sequential([
    base_model,
    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dense(120, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(num_classes, activation='softmax')
])

input_shape = (224, 224, 3)
model = tf.keras.Sequential([
    tf.keras.layers.InputLayer(input_shape=input_shape),
    model
])

# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
model.summary()
epochs = 10

# Check if test_set is empty
if len(test_set) > 0:
    history = model.fit(
        train_set,
        steps_per_epoch=len(train_set),
        epochs=epochs,
        validation_data=test_set,
        # Set validation_steps to 1 to ensure at least one
validation step per epoch
        validation_steps=1
    )
else:
    print("Error: test_set is empty. Please check your data
loading.")

val_loss, val_acc = model.evaluate(test_set,
steps=len(test_set))
print(f"Validation accuracy: {val_acc:.4f}")

# Plot training and validation accuracy/loss
plt.figure(figsize=(12, 6))
plt.plot(history.history['accuracy'], label='Training
Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation
Accuracy')
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Accuracy/Loss')
plt.xlabel('Epoch')
plt.ylabel('Accuracy/Loss')
plt.legend()
plt.show()

## Inception V3

from tensorflow.keras.applications import InceptionV3
from tensorflow.keras.preprocessing.image import
ImageDataGenerator
import pandas as pd
import numpy as np
from tensorflow.keras.models import Model,load_model
from glob import glob
from tensorflow.keras.preprocessing import image
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import
Dense,Activation,GlobalAveragePooling2D
from tensorflow.keras.callbacks import ModelCheckpoint
import tensorflow as tf

inception_model
=InceptionV3(include_top=False,input_shape=[224,224,3])
```

```
for layer in inception_model.layers:
    layer.trainable = False

x =GlobalAveragePooling2D()(inception_model.output)

x = Dense(120)(x)

pred = Activation('softmax')(x)

model = Model(inputs=inception_model.input,outputs=pred)
model.summary()

checkpoint =
ModelCheckpoint('smart_model_mobile.keras',monitor='val_accurac
y',verbose=1,save_best_only=True) # Changed the file extension
to .keras
callback = [checkpoint]

model.compile(optimizer='adam',loss='categorical_crossentropy',
metrics=['accuracy'])

history =
model.fit(train_set,epochs=10,steps_per_epoch=len(train_set),va
lidation_data=test_set,verbose=1,callbacks=callback)

# Plot training and validation accuracy/loss
plt.figure(figsize=(12, 6))
plt.plot(history.history['accuracy'], label='Training
Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation
Accuracy')
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Accuracy/Loss')
plt.xlabel('Epoch')
plt.ylabel('Accuracy/Loss')
plt.legend()
plt.show()
```

## Result Graph

```
import matplotlib.pyplot as plt
import numpy as np

# Example data for accuracies of 4 models
models = [' Xception', 'NasNet Mobile', 'Inception ResNetv2',
'InceptionV3']
train_accuracies = [82.37, 82.03, 82.08, 87.25]  # Training
accuracies for each model
```

```
val_accuracies = [65.37, 83.22, 84.92, 79.69]    # Validation
accuracies for each model

# Set the position of the bars on the x-axis
x = np.arange(len(models))  # the label locations
width = 0.35  # the width of the bars

# Create the plot
fig, ax = plt.subplots(figsize=(8, 8))
bar1 = ax.bar(x - width/2, train_accuracies, width,
label='Training Accuracy', color='skyblue')
bar2 = ax.bar(x + width/2, val_accuracies, width,
label='Validation Accuracy', color='salmon')

# Adding labels and title
ax.set_xlabel('Model names')
ax.set_ylabel('Accuracy Percentage')
ax.set_title('Accuracy Comparison of 4 Models')
ax.set_xticks(x)
ax.set_xticklabels(models)
ax.legend()

# Display the accuracies on top of the bars
for bar in bar1:
    height = bar.get_height()
    ax.annotate(f'{height:.2f}', xy=(bar.get_x() +
bar.get_width() / 2, height),
                xytext=(0, 3), textcoords="offset points",
ha='center', va='bottom')

for bar in bar2:
    height = bar.get_height()
    ax.annotate(f'{height:.2f}', xy=(bar.get_x() +
bar.get_width() / 2, height),
                xytext=(0, 3), textcoords="offset points",
ha='center', va='bottom')

# Show the plot
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

## Html file of home page

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>PawPredict - Dog Breed Classifier</title>
    <link rel="stylesheet" href="static/style.css">

</head>
<body>
    <nav class="navbar">
        <div class="logo">
            <img src="./images/logo.png" alt="Logo">
            <span>PawPredict</span>
        </div>
        <ul class="nav-links">
            <li><a href="./home.html">Home</a></li>
            <li><a href="./about.html">About</a></li>
            <li><a
href="./predict.html">Predict</a></li>
            <li><a        href="./contact.html">Contact
Us</a></li>
        </ul>
    </nav>
    <div class="main">
        <div class="content">
            <h1>Welcome to PawPredict</h1>
            <p>Upload an image to classify the dog breed
accurately.</p>
            <a   href="./predict.html"   class="btn">Get
Started</a>
        </div>
    </div>
</body>
</html>
```

## CSS for the Home Page

```
/* General Reset */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: Arial, sans-serif;
}

/* Navbar */
.navbar {
    width: 100%;
    background-color: #333; display:
    flex;
    justify-content: space-between; align-
    items: center;
    padding: 1rem 2rem;
}

.logo {
    display: flex;
    align-items: center;
}

.logo img {
    height: 40px; margin-right:
    10px;
}

.logo span {
    font-size: 1.5rem; color:
    #ff7200; font-weight: bold;
}

.nav-links {
    display: flex; list-
    style: none;
}

.nav-links li {
    margin-left: 2rem;
}

.nav-links li a {
    text-decoration: none; color:
    white;
    font-weight: bold; font-
    size: 1.2rem;
```

```
        transition: color 0.3s ease;
    }

    .nav-links li a:hover { color:
        #ff7200;
    }

    /* Main Section */
    .main {
        width: 100%; height: 90vh;
        display: flex;
        align-items: center; justify-
        content: flex-end;
        background: linear-gradient(rgba(0, 0, 0,
0.3), rgba(0, 0, 0, 0.3)), url('./images/main.jpg') no-
repeat center center/cover;
        padding-right: 5%;
    }

    .content {
        text-align: right; color:
        white;
        max-width: 50%;
    }

    .content h1 {
        font-size: 3rem; color:
        #ff7200;
    }

    .content p {
        font-size: 1.5rem; margin-
        top: 1rem;
    }

    .btn {
        display: inline-block; margin-
        top: 2rem; padding: 1rem
        1.5rem;
        background-color: #ff7200; color: white;
        border-radius: 5px;
        text-transform: uppercase;
        transition: background-color 0.3s ease; text-
        decoration: none;
    }

    .btn:hover {
```

```
    background-color: #fff; color:
    #ff7200;
}

/* Responsive Design */ @media (max-
width: 768px) {
    .navbar {
        flex-direction: column; align-
        items: flex-start;
    }

    .nav-links {
        flex-direction: column; margin-top:
        1rem;
    }

    .main {
        justify-content: center; text-align:
        center;
    }

    .content {
        max-width: 80%; text-align:
        center;
    }
}
```

## Html file of the prediction page

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Dog Breed Prediction</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/c
ss/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="predict.css">
</head>
<body>
    <div class="container">
        <div class="row">
            <!-- Home Image on the Left (Always Visible)
-->
```

```
               <div class="col-md-6">
                   <img  id="homeImage"  class="img-fluid"
src="./images/home.jpg" alt="Home Image">
               </div>

               <!-- Upload & Prediction Section -->
               <div class="col-md-6">
                   <h1>Dog Breed Prediction</h1>
                   <div class="upload-container">
                       <form                id="imageForm"
onsubmit="return false;">
                           <!-- Image Upload Box -->
                           <div        class="upload-box"
onclick="document.getElementById('imageInput').click();">
                               <p>Drag  &  drop  your  image
here or</p>
                               <button       type="button"
class="upload-button1">Browse  Image</button>
                               <input          type="file"
id="imageInput"                           accept="image/*"
onchange="displayFile(event)" hidden>
                           </div>

                           <!--   Image   Preview   (Sample
Image First) -->        HG<div id="imagePreviewContainer"
style="text-align: center; margin-top: 10px;">
                               <img     id="selectedImage"
src="./images/sample.png"  class="img-fluid"  style="max-
width: 50%; height: auto; border: 2px solid #ccc; padding:
5px;">
                           </div>

                           <!-- File Name Display -->
                           <div         class="file-name"
id="fileNameDisplay"></div>
                           <br>

                           <!--  Submit  Button  (Initially
Disabled) -->
                           <button       id="submitButton"
class="upload-button"             onclick="submitImage()"
disabled>Submit</button>
                       </form>
                   </div>
                   <h4 id="result"></h4>
               </div>
           </div>
       </div>
```

```
    <script src="./predict.js">
    </script>
</body>
</html>
```

```
## CSS file of Predication page
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    background-color: lightgrey;
}

.container {
    background-color: white;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 0 10px gray;
    text-align: center;
    width: 90%;
    max-width: 800px;
}

h1 {
    margin-top: 10px;
}

.img-fluid {
    width: 100%;
    height: auto;
}

.upload-container {
    background-color: white;
    padding: 20px;
    border-radius: 10px;
    text-align: center;
}

.upload-box {
    border: 2px dashed #007bff;
    padding: 20px;
    border-radius: 10px;
    cursor: pointer;
    transition: all 0.3s ease;
}

.upload-box:hover {
    background-color: #f0f8ff;
}

.upload-box input[type="file"] {
    display: none;
}

.upload-box p {
    color: #555;
    font-size: 16px;
    margin: 0;
}

.upload-button {
    background-color: #007bff;
    color: white;
    border: none;
    padding: 10px 20px;
    border-radius: 5px;
    font-size: 16px;
    cursor: pointer;
    transition: background-color 0.3s ease;
    margin-top: 0px;
}

.upload-button:hover {
    background-color: #0056b3;
}

.upload-button1 {
    background-color: #007bff;
    color: white;
    border: none;
    padding: 10px 20px;
    border-radius: 5px;
    font-size: 16px;
    cursor: pointer;
    transition: background-color 0.3s ease;
    margin-top: 10px;
}
```

```
.upload-button1:hover {
    background-color: #0056b3;
}

h4 {
    margin-top: 0px;
    color: #0c0e1
```

## Back end code of the Prediction page

```
import os
import tensorflow as tf
from flask import Flask, request, jsonify
from flask_cors import CORS
from tensorflow.keras.preprocessing import image
import numpy as np
from tensorflow.keras.models import load_model

# Initialize the Flask app
app = Flask(__name__)
CORS(app)  # Enable Cross-Origin Resource Sharing (CORS)

# Ensure the 'uploads' directory exists
UPLOAD_FOLDER = 'uploads'
if not os.path.exists(UPLOAD_FOLDER):
    os.makedirs(UPLOAD_FOLDER)

# Load the trained model
MODEL_PATH = 'NANSNET_model.h5'  # Ensure the model file
is in the same directory
model = load_model(MODEL_PATH)

# Dog breed class labels
class_labels = {
    0:   "Chihuahua",   1:   "Japanese_spaniel",   2:
"Maltese_dog", 3: "Pekinese", 4: "Shih-Tzu",
    5:   "Blenheim_spaniel",   6:   "Papillon",   7:
"Toy_terrier",     8:     "Rhodesian_ridgeback",     9:
"Afghan_hound",
    10: "Basset", 11: "Beagle", 12: "Bloodhound", 13:
"Bluetick", 14: "Black-and-tan_coonhound",
    15: "Walker_hound", 16: "English_foxhound", 17:
"Redbone", 18: "Borzoi", 19: "Irish_wolfhound",
    20:   "Italian_greyhound",   21:   "Whippet",   22:
"Ibizan_hound",     23:     "Norwegian_elkhound",     24:
"Otterhound",
    25:   "Saluki",   26:   "Scottish_deerhound",   27:
"Weimaraner",   28:   "Staffordshire_bullterrier",   29:
"American_Staffordshire_terrier",
    30: "Bedlington_terrier", 31: "Border_terrier", 32:
```

```
"Kerry_blue_terrier", 33: "Irish_terrier", 34:
"Norfolk_terrier",
       35: "Norwich_terrier", 36: "Yorkshire_terrier", 37:
"Wire-haired_fox_terrier", 38: "Lakeland_terrier", 39:
"Sealyham_terrier",
       40: "Airedale", 41: "Cairn", 42:
"Australian_terrier", 43: "Dandie_Dinmont", 44:
"Boston_bull",
       45: "Miniature_schnauzer", 46: "Giant_schnauzer",
47: "Standard_schnauzer", 48: "Scotch_terrier", 49:
"Tibetan_terrier",
       50: "Silky_terrier", 51: "Soft-
coated_wheaten_terrier", 52:
"West_Highland_white_terrier", 53: "Lhasa", 54: "Flat-
coated_retriever",
       55: "Curly-coated_retriever", 56:
"Golden_retriever", 57: "Labrador_retriever", 58:
"Chesapeake_Bay_retriever", 59: "German_short-
haired_pointer",
       60: "Vizsla", 61: "English_setter", 62:
"Irish_setter", 63: "Gordon_setter", 64:
"Brittany_spaniel",
       65: "Clumber", 66: "English_springer", 67:
"Welsh_springer_spaniel", 68: "Cocker_spaniel", 69:
"Sussex_spaniel",
       70: "Irish_water_spaniel", 71: "Kuvasz", 72:
"Schipperke", 73: "Groenendael", 74: "Malinois",
       75: "Briard", 76: "Kelpie", 77: "Komondor", 78:
"Old_English_sheepdog", 79: "Shetland_sheepdog",
       80: "Collie", 81: "Border_collie", 82:
"Bouvier_des_Flandres", 83: "Rottweiler", 84:
"German_shepherd",
       85: "Doberman", 86: "Miniature_pinscher", 87:
"Greater_Swiss_Mountain_dog", 88: "Bernese_mountain_dog",
89: "Appenzeller",
       90: "EntleBucher", 91: "Boxer", 92: "Bull_mastiff",
93: "Tibetan_mastiff", 94: "French_bulldog",
       95: "Great_Dane", 96: "Saint_Bernard", 97:
"Eskimo_dog", 98: "Malamute", 99: "Siberian_husky",
       100: "Affenpinscher", 101: "Basenji", 102: "Pug",
103: "Leonberg", 104: "Newfoundland",
       105: "Great_Pyrenees", 106: "Samoyed", 107:
"Pomeranian", 108: "Chow", 109: "Keeshond",
       110: "Brabancon_griffon", 111: "Pembroke", 112:
"Cardigan", 113: "Toy_poodle", 114: "Miniature_poodle",
       115: "Standard_poodle", 116: "Mexican_hairless",
117: "Dingo", 118: "Dhole", 119: "African_hunting_dog"
   }
   # Function to preprocess the image
   def preprocess_image(img_path):
       img  =  image.load_img(img_path,  target_size=(224,
224))
       img_array = image.img_to_array(img)
       img_array = np.expand_dims(img_array, axis=0)
       img_array /= 255.0  # Rescale the image
       return img_array

   # Function to predict dog breed
   def predict_image(img_path):
       img_array = preprocess_image(img_path)
       predictions = model.predict(img_array)
       predicted_class = np.argmax(predictions, axis=1)[0]

       predicted_label = class_labels.get(predicted_class,
"Unknown")
       return predicted_label

   # Route to handle image upload and prediction
   @app.route('/predict', methods=['POST'])
   def predict():
       if 'file' not in request.files:
           return jsonify({'error': 'No file part'}), 400

       file = request.files['file']
       if file.filename == '':
           return jsonify({'error': 'No selected file'}),
400

       if file:
           # Save the uploaded image
           img_path      =      os.path.join(UPLOAD_FOLDER,
file.filename)
           file.save(img_path)

           # Predict the image class
           predicted_label = predict_image(img_path)

           return jsonify({'result': predicted_label})

       return  jsonify({'error':  'Failed  to  process  the
image'}), 500

   if __name___== '__main__':
       app.run(debug=True)
```

9. Results Analysis

Here in Figure 9.1 we can say the xception model has provided us with 79.08 valid accuracy and 90.41 training accuracy while being compared with other models the xception model has provided us with lowest accuracy

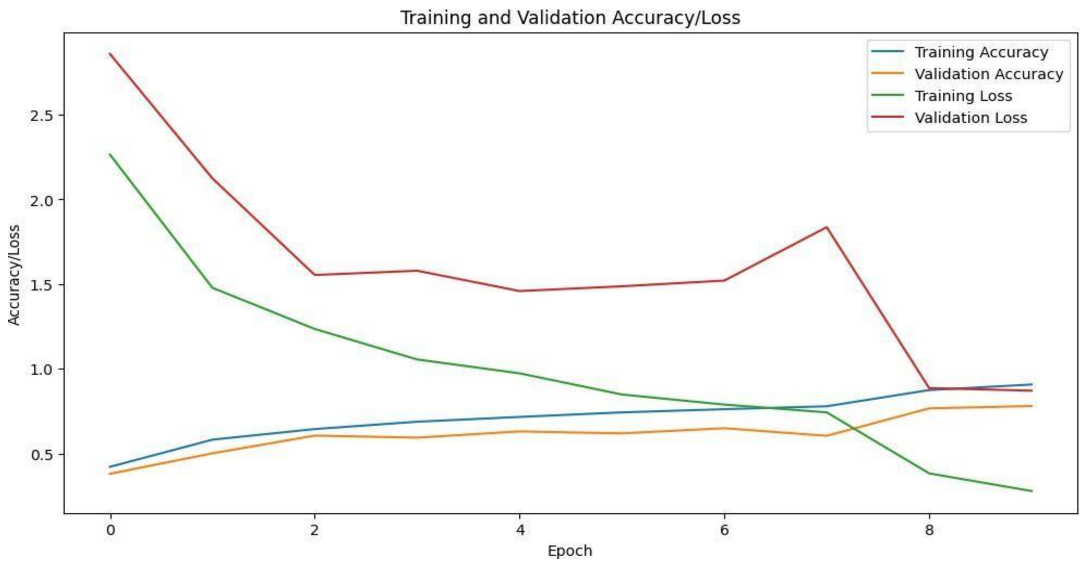


Fig 9.1: ROC graph of Xception model

The NasNetMobile model has maintained a constant accuracy for both validate loss and validate accuracy and similarly for training accuracy and training which is shown in Fig 9.2

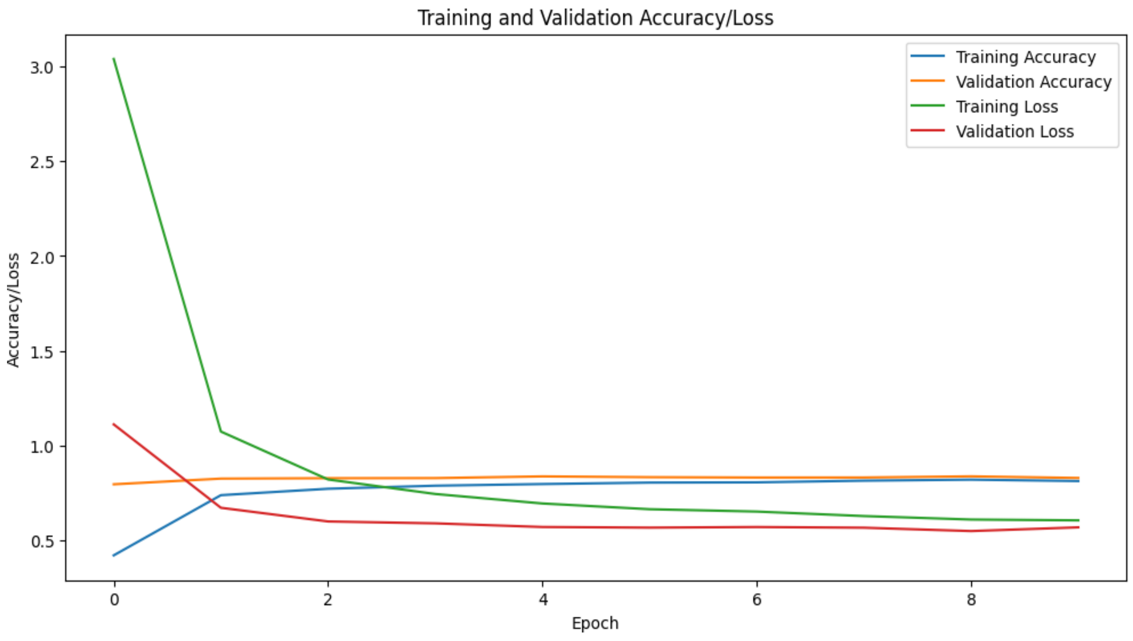


Fig 9.2: ROC graph of NASNETMOBILE model

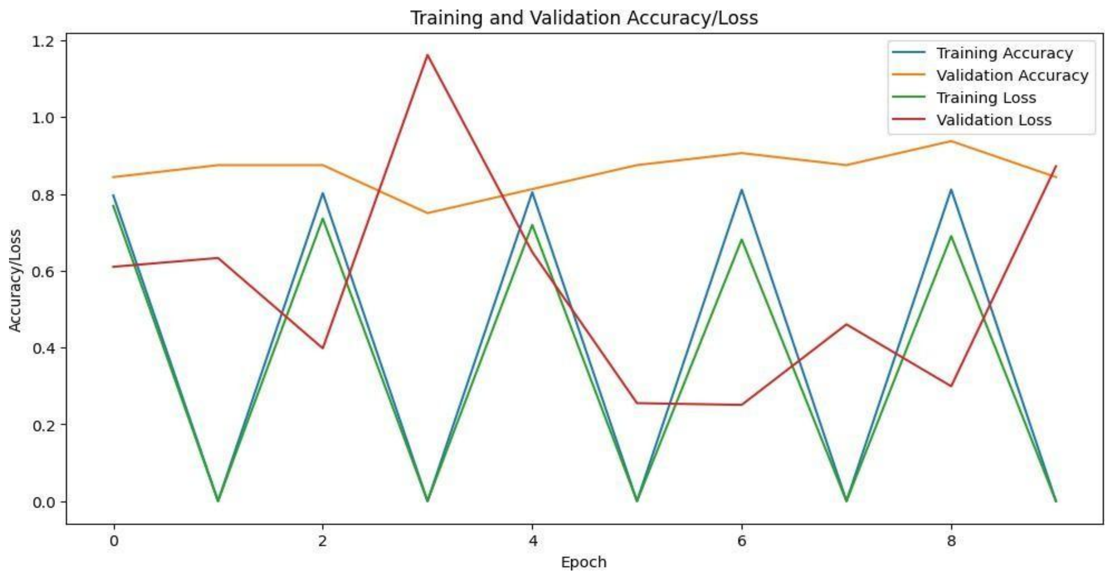


Fig 9.3: ROC graph of Inception Resnetv2

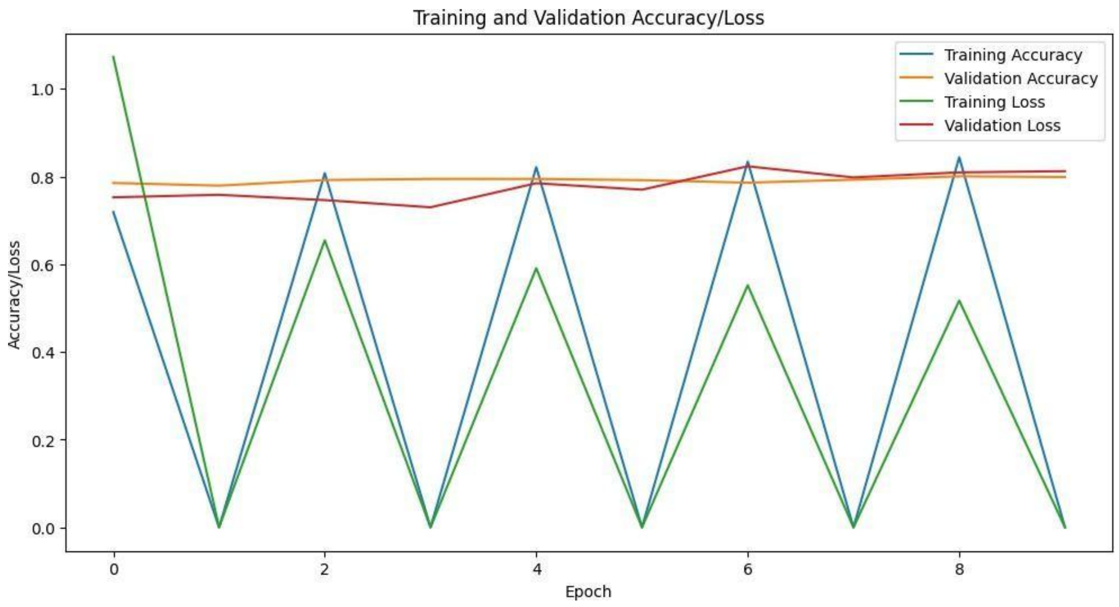


Fig 9.4: ROC graph of Inception V3

In both the Inception V3 and Inception ResNetv2 model the training loss and training accuracy has maintained a constant accuracy and the highest accuracy is provided by the Inception ResNetv2 model we can see the ROC graph of both the Model in Figure 9.3, 9.4.

The analysis from the above graphs accuracy of different algorithms reveals that 'Our Model' consistently provide more across all metrics. It exhibits higher accuracy indicating its effectiveness in making correct predictions, minimizing false positives, capturing relevant instances, and maintaining a balance. Overall, 'Our Model’ demonstrates superior performance, suggesting its suitability for classification tasks compared to established models.

10. Test cases

Test case 1:

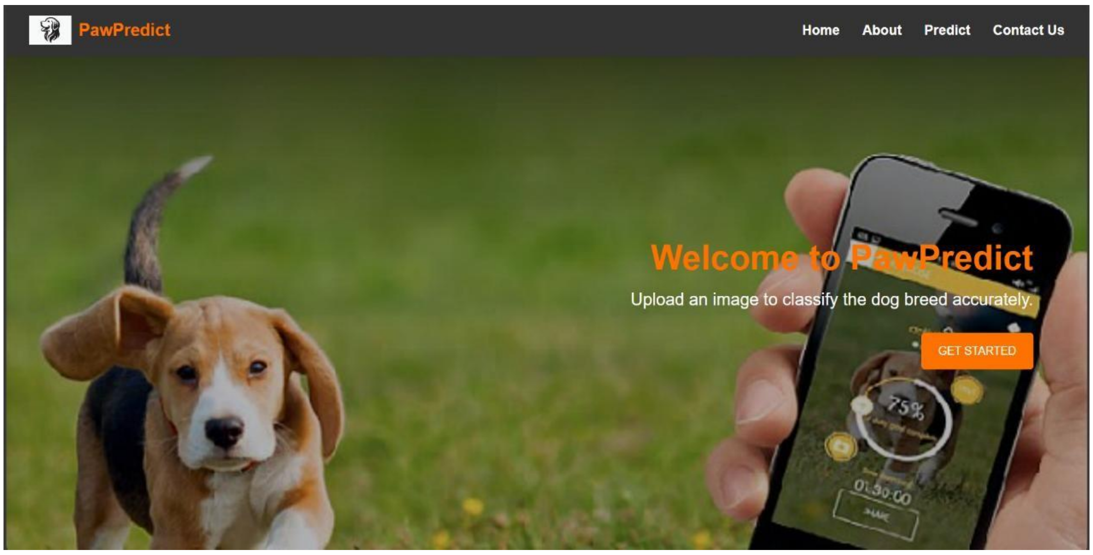


Fig 10.1: Home page

The Fig 10.1 page it is seen when the user opens the website and it can direct us to choose the prediction page

Test case 2:

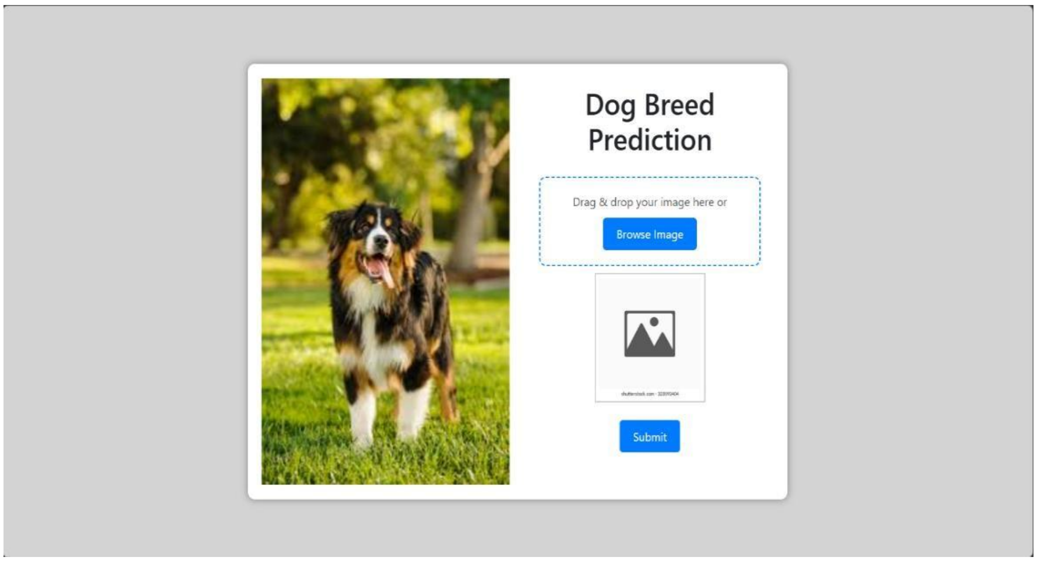


Fig 10.2: Prediction page

The Fig 10.2 is the prediction page where user can insert the image from their system or from their drive

11 User interface

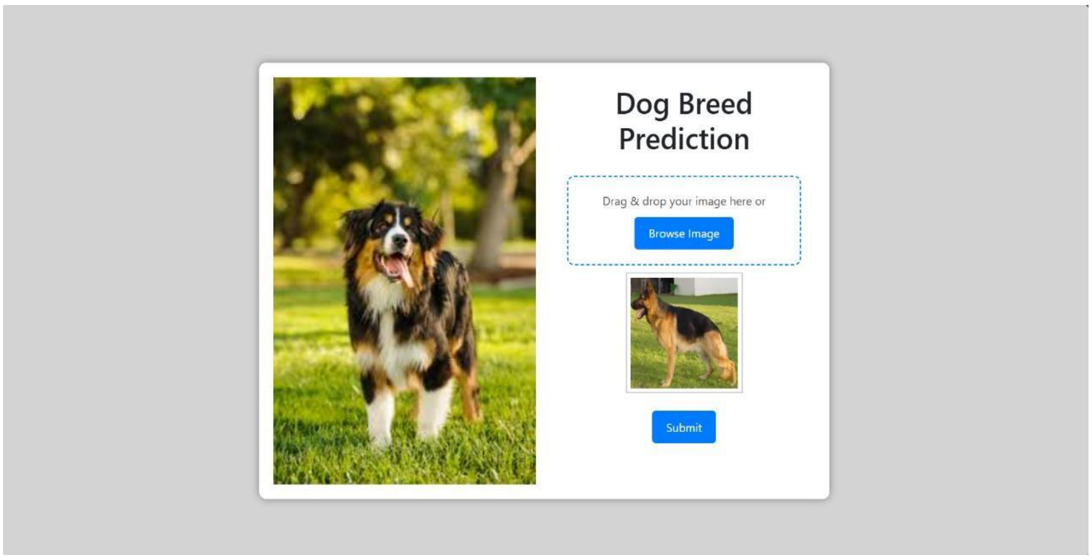
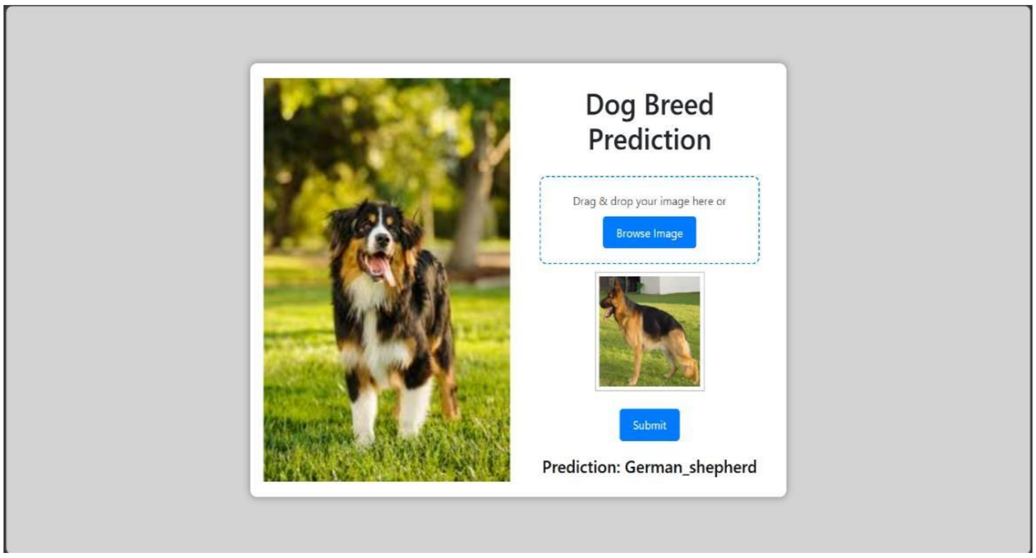


Fig 10.3: Pic uploading page

In Fig 10.3 after uploading the image to the website we can then submit it to the backend code where it predict the dog



Prediction: German_shepherd

Fig 10.4: Output screen

After submitting the image it will take at least 5 sec to predict the given dog and give the breed name in the bottom of the page in Fig 10.4

12. Conclusion

This documentation highlights the successful implementation of Deep Learning methodologies for dog breed identification and classification using image data. The research utilized advanced Convolutional Neural Network (CNN) architectures, including Xception, NASNetMobile, InceptionV3, and InceptionResNetV2, to address the inherent challenges in fine-grained image classification. The hybrid models proposed in this study achieved notable improvements in accuracy, with InceptionResNetV2 yielding the highest validation accuracy of 87.25%.

The findings validate the potential of combining transfer learning and data augmentation techniques to enhance the performance of Deep Learning models in identifying breeds from diverse and complex datasets. Furthermore, the proposed model's robust performance illustrates its suitability for real-world applications in veterinary medicine, pet care, law enforcement, and beyond.

Future advancements could include incorporating larger, more diverse datasets, adopting emerging architectures like Vision Transformers, and optimizing models for deployment on resource-constrained devices. Moreover, developing real-time applications such as mobile apps for dog breed identification could bridge the gap between research and practical usage, offering significant benefits to pet owners, veterinarians, and other stakeholders.

This study sets a strong foundation for leveraging Deep Learning in image classification tasks and opens avenues for its application in broader contexts, including wildlife monitoring, medical imaging, and security systems. By continuing to innovate in architecture design and dataset diversity, the potential for achieving even greater accuracy and applicability in similar tasks is promising.

13. Future scope

In future research involve integrating ELA with state-of-the-art CNN architectures, such as ResNet, DenseNet, or EfficientNet, to enhance the detection accuracy and robustness againstvarious types of image manipulations, exploring novel optimization techniques, such as genetic algorithms or reinforcement learning, in conjunction with ELA-based feature extraction and CNN-based classification, could offer improvements in forgery detection performance.

14. References

1. B.Valarmathi, S.Saravana, N.Srinivas Gupta, G.Prakash, AND P.ShanMmugasundAram. (2023). Hybrid Deep Learning Algorithms for Dog Breed Identification—A Comparative Analysis. Digital Object Identifier 10.1109/AC-CESS.2023.329744.

2. R.sinnot, Fang Wu, Wenbin Chen. A Mobile Application for Dog Breed Detection and Recognition Based on Deep Learning. Published in 2018 IEEE/ACM 5th International Conference on Big Data Computing Applications and Technologies DOI: 10.1109/BDCAT.2018.00019

3. S. Iyer, Narendra M. Shekokar. Identifying the Optimal Deep Learning Based Image Recognition Technique for Dog Breed Detection. Published in International Conference on Computing Communication and Networking Technologies (ICCCNT) 6 July 2023 DOI: 10.1109/ICCCNT56998.2023.10307245

4. Arshdeep Singh Ghotra, Harashleen Kour, Anas Hasan and Akash Khan (2023). Dog Breed Identification. International Journal of Advanced Science and Technology, e-ISSN: 2395-0056, p-ISSN: 2395-0072

5. Aydin Ayanzadeh, Sahand Vahidnia Modified Deep Neu- ral Networks for Dog Breeds Identification. Preprints (2018), 2018120232.https://doi.org/10.20944/preprints201812.0232.v1

6. Bickey Kumar shah, Aman Kumar, Amrit Kumar Dog Breed Classifier for Facial Recognition using Convolutional Neural Networks(2021). DOI: 10.1109/ICISS49785.2020.9315871

7. Sinéad Kearney; Wenbin Li; Martin Parsons; Kwang In Kim; Darren Cosker. RGBD-Dog: Predicting Canine Pose from RGBD Sensors(2017), DOI:10.1109/cvpr42600.2020.00836

8. P. Borwarnginn, W. Kusakunniran, S. Karnjanapreechakorn, and K. Thongkanchorn, “Knowing your dog breed: Identifying a dog breed with Deep Learning,” Int. J. Autom. Comput., vol. 18, no. 1, pp. 45–54, Feb. 2021, doi: 10.1007/s11633-020-1261-0.

9. S. L. Jagannadham, K. L. Nadh and M. Sireesha, "Brain Tumour Detection Using CNN," 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 2021, pp. 734-739, doi: 10.1109/I-SMAC52330.2021.9640875

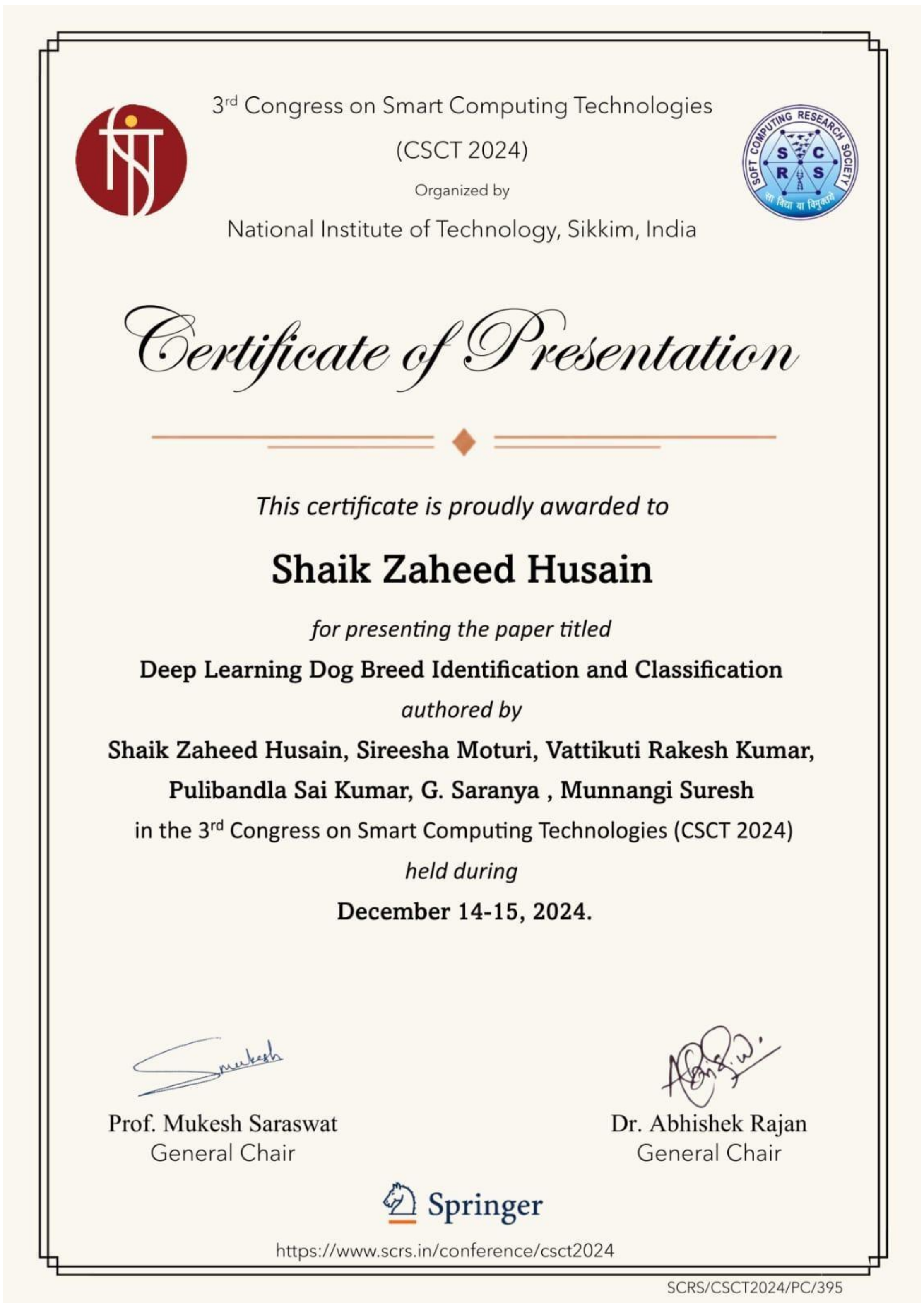
10. Dylan Rhodes CS231n, Stanford University. Automatic Dog Breed Identification (2015). Published in 2015 in Stanford University DOI (or) Link: [PDF] Automatic Dog Breed Identification | Semantic Scholar

11. Ayanzadeh, A.; Vahidnia, S. Modified Deep Neural Net- works for Dog Breeds Identification. Preprints 2018, 2018120232. https://doi.org/10.20944/preprints201812 0232.v1

12. Parker, H.G. Genomic analyses of modern dog breeds. Mamm Genome 23, 19–27 (2012). https://doi.org/10.1007/s00335-011-9387-6

13. Rethinking dog breed identification in veterinary practice Robert John Simp- son DVM, Kathyrn Jo Simpson DVM, MPH, and Ledy VanKavage JD DOI: https://doi.org/10.2460/javma.241.9.1163 Volume/Issue: Volume 241: Issue 9 On- line Publication Date: 01 Nov 2012

14. Vaysse A, Ratnakumar A, Derrien T, Axelsson E, Rosengren Pielberg G, Sigurdsson S, et al. (2011) Identification of Genomic Regions Associated with Phenotypic Variation between Dog Breeds using Selection Mapping. Published: October 13, 2011



3rd Congress on Smart Computing Technologies

(CSCT 2024)

Organized by

National Institute of Technology, Sikkim, India

Certificate of Presentation

This certificate is proudly awarded to

Shaik Zaheed Husain

for presenting the paper titled

Deep Learning Dog Breed Identification and Classification

authored by

Shaik Zaheed Husain, Sireesha Moturi, Vattikuti Rakesh Kumar, Pulibandla Sai Kumar, G. Saranya , Munnangi Suresh

in the 3rd Congress on Smart Computing Technologies (CSCT 2024)

held during

December 14-15, 2024.

[illegible]

Prof. Mukesh Saraswat
General Chair

[illegible]

Dr. Abhishek Rajan
General Chair

Springer

https://www.scrs.in/conference/csct2024

SCRS/CSCT2024/PC/395

Deep Learning Dog Breed Identification and Classification

G. Saranya[1], Shaik Zaheed Husain[2], Vattikuti Rakesh Kumar[3], Pulibandla Sai Kumar[4], Sireesha Moturi[5], and Munnangi Suresh[3]

[1] 1,6Asst.Professor, Dept of CSE, Narasaraopeta Engineering College, 1Narasaraopet, India.
[2] 5Associate Professor, Dept of CSE, Narasaropeta Engineering College, 1Narasaraopet, India.
[3] 2,3,4Students,Dept of CSE, Narasaraopeta Engineering College, 1Narasaraopet, India.
[4] zaheedhussain493@gmail.com, sherlockrakesh@gmail.com, pulibandlasaikumar02@gmail.com, sureshmunnangi55@gmail.com, sireeshamoturi@gmail.com
[5] gaddamsaranya4@gmail.com

Abstract. This paper will identify the specific dog breed from an image using deep learning and computer vision techniques. The aim is for the user to upload a picture of a dog and then the model should determine the breed of the dog from the list of 120 available breeds located in the dataset. Identifying dog breeds is fundamental in veterinary medicine, pet care, and dog welfare. However, single deep learning models and traditional methods have their fair share of challenges, especially regarding the accuracy that comes along with high diversity in the appearances of dogs. The proposed work conducts prediction of breed for dogs from deep learning modeling applying a number of strategies that include Xception, NASNetMobile, Inception and Xception. The current system made use of Inception-v3 and InceptionResNetV2 on Stanford Dogs Standard Datasets. The blended models are viewed to be fine combina tions of NASNetMobile, Inception and Xception to and Inception-v3 and Xception. This model performs better than others single models which include Xception, InceptionV3, ResNet50 and ResNet101. To enhance the results, the author used a transfer learning approach with data enlargement. The Accuracy which are obtained were 86.92 and 79.69 val= idation accuracy scores against InceptionV3 this ad InceptionV3 respectively. The most accuracy of InceptionResNetV2 was 84.92 in the present system. By applying the proposed algorithms the validation accuracy of InceptionResNetV2 was 84.92, 77.68 of Xception, 83.22 of NASNetMo- bile, 79.38 of Inception V3. Whichever of the Hybrid of the Inception-v3 Xception method is the last type in the combination of ordinary measurement does offers a level of accuracy that is unsurpassed amongst these algorithm coming in at 92.4.

Keywords: Deep Learning, Data Preprocessing, NasNet Mobile, Xception, Inceptionv3, Inception ResNetv2 Convolutional Neural Network (CNN)

1 INTRODUCTION

Learning about dog breed is critical understanding information for responsible pet ownership, veterinary care, and the management of the community. Knowing a dog's breed can be highly informative regarding how the dog is likely to behave and its special care requirements. Selling particular breeds of dogs is done by targeting certain characteristics such as energy levels, amount of grooming required and even their health risks. This will allow the owners

to understand what training procedures, exercises, and even meals are necessary for the dogs according to their breed Hence the dog gets the attention and help it requires to flourish. In the public domain or legislation, identification of the breed of a dog is critical especially when the dog is involved in a dog attack or display of aggression. Recognition of the breed is very important in predicting the tendencies and aggressiveness of different types of dogs, especially those regarded as potentially dangerous. Identification in such a situation would then assist in dealing with a situation that would otherwise provoke action by law enforcers, insurance companies, or managers of a community who may use the legislation to discriminate and prevent acts of aggression.·

The author B.Valarmathi et al[1],proposed a Convolutional Neural Network (CNN) and model like Inception-V2, inception Resnet-v3, Resnet101, Resnet50 are begin used to determine the characteristic of the dog and to gain higher accuracy. The highest accuracy gained by the author is 90 in Resnet model, the author pretrained the datasets and convert the data in to array to gain best accuracy, the author also compare various model with his accuracy to gain more data·

R.sinnot et al [2], has address problems of data classification and making prediction from raw data by deriving (learning) new knowledge or facts based on this raw data. has done One most prominent and widely used approach in the analysis of visual data is through the use of Convolution Neural Networks (CNNs). The author demonstrated approximately 85 accuracy for dog breed classification tasks based on the set of 50 dog breeds and achieved 64 accuracy for 120 other uncommon dog breeds and even lower accuracy for 120 less common dog breed classes.·

In this research Journal the author S. Iyer et al. used a data set of 120 dog breeds that are used to train and test the deep learning model [3]. The dataset's were taken from the kaggle which included large driver and unique images of digs in different angle and position, it containing approximately 10,222 images of 120 category of dog breeds. Arshdeep Singh Ghotra et al[4] used CNN, Mobilenetv2 was one the best choice of the author which provided 84 accuracy. The solution provided from the model training and testing were in comparison of based on estimate accuracy, precision and area on different models.·

This domain would be categorized as Computer Science and Artificial Intelligence (AI) in Aydin et al [5] and more specifically under the topics of (ML), Deep Learning, and Computer Vision. Form [7] Sinéad Kearney, such a domain works towards use of AI based approaches to perform categories of different breeds of dogs using images.Dylan Rhodes et al[10] This study may further discuss Pattern Recognition since there is a possibility of implementing many techniques and drawing conclusions based on their comparisons

2 Literature Survey

The Bickey Kumar [6] used different dog breeds and human images using CNN. It consist of 13,233 human images and 8,351 dog images. If a dog image was given, the model or proposed model would search for that dog class. If a human picture was given, it will determined which facial features would show in a dog.·
In this study P. Borwarnginn et al[8], wants to show the way of overcoming this problem and concerns. The proposed method used by the author is deep learning based where dogs and their breeds are identified. This approach begins with a transfer learning and specific pre-trained Convolutional Neural Networks (CNNs) are retrained on the public dataset of dog breeds. The proposed model comprises of 133 classes of dog breeds and it factors whose accuracy performance of the model almost approaches is 89.92 dog breeds dataset·

S.L.Jagannadham et al [9] In this research work augmentation techniques will be incorporated in order to enhance the training dataset and different CNN approaches will be applied to extract maximum information from the image. In order to improvise the details from the picture, the surrounding area of the image was taken away from all four sides. A few preprocessing techniques of dealing with images in deep learning are employed in the removing of noise present in the images to make the output efficient.·

The A.Ayanzadeh et al [11] uses the pretrained model for the identification of the dog's. It also retrieve the data from the data set. The algorithm uses similar model has the proposed work but it uses two different model like DenseNet-121,
169. We can verify that it has provided high accuracy above 80 for the models·
Model of [12] Parker et al here uses the analyses of human and dog to genomic the different between and identify the features of them this method is frequently used worldwide and its best for finding either complex or simple diseases·

The creator of [13] Robert John Simpson et al reference paper has conducted many research and found out that there are 44 of different dog of mixed breed they may have similar identification and body shape. The author has checked the DNA of approximatelyall dogs class in the country and verified that most of the dos breed are mixed or hybrid·

Here the article A.Vaysse et al [14] check the dog SNP to identify which region the dog breed comes from here we used more than 500 dog and arrays consisting of 170000 spaced SNP. It also uses the mapping system to verify any strong body size and morphology·

3 Data Set and Models Explanation

3.1 About dataset

The dataset provided in this research is shown at the below link . https://www.kaggle.com/datasets/miljan/stanford-dogs-datasettraintest There are images of 120 types of dogs in the Stanford Dogs dataset. Imagery and annotation from ImageNet were purposefully employed to build this dataset for a fine-grained image classification. Below you can find what is in this dataset. • 120 categories • 10,222 images • Class File for Each Dog Breed

3.2 Dataset Breakdown

Training Data: Out of the total Ms.Lalita, 10,222 images are used in training the CNN model. This considerable portion of the dataset helps in exposing the model to a lot of examples, thus enabling it to learn different varieties of patterns that are embedded in the characteristics of a dog. Validation Data: The same amount of images are used for validation purposes. This subset is important in determining how well the model has been trained relative to how it performs in the handling of new data3.

3.3 Analyze and Visualize the Dataset

There is a section most likely dedicated to this aspect where sample images, their class distributions, and such statistics are used to examine the dataset. This aspect is critical in estimating the character and distribution of the dataset before model training.

3.4 Preprocessing work

This work covers the steps of importing the auxiliary resources, outlining image generators, and establishing data augmentation methods. Normalization of pixel data, re-scaling of images, and more such operations adding various nondestructive images using transformations like rotating, flipping, or zooming are also possible and this is done to increase the dataset and avoid over fitting. Image generators with ImageDataGenerator from Keras, which handles real-time data augmentation. Another snippet presents a function dedicated to the preprocessing of single images by appropriate resizing and normalization of images done to perform a model prediction

3.5 Reshape

Reshape operations are defined in most cases as parts of the data pre-processing or model input pipeline where images are adjusted to standardized shapes mostly conforming to the input specifications for the neural network e.g. 224X224 pixels for common CNN frameworks Models The neural network in this project are implemented using four different types of models for dog breed prediction with the advanced accuracy they are Xception model, NASNetMobile, InceptionResNetV2, Inception V3

3.6 Models

In this project the developer used four different types model for the prediction of the dog breed with greater accuracy those are Xception model, NasNetMobile, InceptionResNetV2, Inception V3

Xception model is a deep learning architectural model developed by Francois Chollet in the year 2017 which is a more advanced form of the basic Inception model by employing the depth wise separable convolution. These convolutions break down the standard convolution procedure into two basic operations, the first is the depth wise convolution in which each filter is applied to an input channel singly and the second is the point wise convolution in which the depth wise convolution's outputs are integrated with a 1x1 convolution on all input channels. This helps to minimize computation and reduce the number of parameters thus making the models more efficient.

NASNetMobile: Google developed the NASNet model, which stands for Neural Architecture Search Network, in the year 2017 and is a deep learning model that has been designed through Neural Architecture Search (NAS). With reference to NASNet, these methods were searched in order to provide the best building type cells to be used. which could later on be contoured into the final framework. The NASNet models thus resulting have great efficiency and scalability and offer advanced performance on tasks like image classification, object recognition, etc. Importantly, these NASNet models have the architectural flexibility of being adapted to different levels of computation to maintain a reasonable accuracy relative to other architectures.

InceptionResNetV2: The Inception-ResNet-v2 model, which is a neural network based system, fuses the aspects of two different deep learning architectures Inception and ResNet. The introduction of the residual connections also enables the network to learn some identity functions as well; hence speeding up the training process. By using the depth, width, and computational efficiency, Inception-ResNet-v2 has been able to obtain state of the art results in image recognition applications. In particular, it has been reported that its effectiveness is optimal for large scale datasets such as ImageNet. This kind of structure has been embraced in computer vision problems because of its excellent performance as well as expandability.

InceptionV3: Inception v3 model is a C2D architecture deep learning model and it is the third generation of the poses series which was developed by Google to solve the problem of stereoscopic recognition. Being one of the latest models addition in 2015. The model uses “Inception modules,” a technique that helps in developing the network by reducing the cost of running the network using multiple convolutions of different sizes aimed at obtaining multiscale features. The key updates included in Inception V3 are so called factorized convolutions, when standard large convolutions are expressed with a series of smaller, efficient ones, batch normalization to make the training stable and faster. Thanks to the sophisticated architectural design of Inception V3 the image classification tasks like ImageNet was established and is still one of the preferred options for many computer vision tasks

3.7 Proposed Work

For evaluation in this study, the proposed methods again taking a model, predicting dog breeds make use of five different models Xception, ResNet-101, ResNet-50, NASNetMobile and Inception. The importance of dog breed identification goes beyond the mere classification of dogs—the classification is also employed in other wider applications such as recognition of wild species, as well as in learning to detect entities or occupants within a certain area such as in retinal images. Such tasks necessitate identifying subtle variations between categories, which is notoriously difficult for several models.

Theses will analyze the four models and find the most accurate and efficient. Existing proposed models include Xception, NasNet Mobile, InceptionResNetV2, Inception v3. Consider whether ethical issues could arise from judgment calls rooted in breed-type prejudice or classifications that pertain to disability. In contrast to classical residuals that first combine features and then reduce their dimensionality, Inverted residuals are incorporated in MobileNetV2 which is a high performing Convolutional Neural Network architecture model designed for mobile and embedded device. The configuration of the system which is being used is 64 bits and i3, i5 core in the goggle colab platform In Figure 3 several steps will be taken through to wherever this task will be completed as long as it is to achieve the best possible results.

Step 1 (Import Modules): Since the proposal work includes dealing with Python related programming, the first step in the proposed work is to import crucial libraries that are needed in the proposal work. In this project the seaborn and matplotlib modules are used for graphic. Tensor flow make use of pre-trained models and it uses train our own model. Training and testing splitter ImageDataGenerater Scikit learn library is employed. Data from the Google drive is imported with the use of the OS library. Pandas and Numpy Module are utilized to take care of image arrays.

Step 2 (Load Data): By mounting the drive with project platform we load the Stanford dataset by unzipping the file and uploading into the Random Access Memory during the runtime.

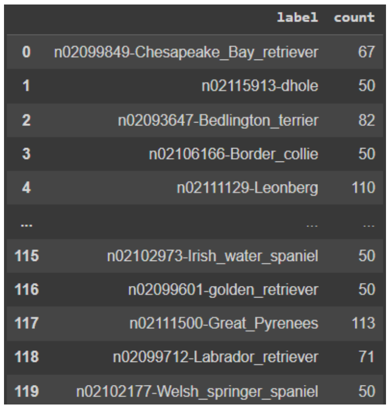
Step 3 (Analyzing the Data): To analyze the total picture of the dogs and to find bad reporting in the data and to count how many pictures of every breed are present in a class are displayed in Figure 1.

Step 4 (Validating Data): See if the two values, labels and picture are equal. In case that yes and equal then it means every image is labeled and they can be moved on it can be seen in Figure 1 second picture.

Step 5 (Encode Unqualified Classes): While encoding different type of dog breed classes we would allocate a unique and special letter or numbers to each of the class of the dog breeds.

Step 6 (Visualize the dataset): In addition we also visualizing the data by using Pandas data frame and functions and plotting graphs and building a bar chart representing each breed with respect to the size of the dog.

Step 7 (Training Dataset): In the proposed work the dataset is trained. In the training set we use ImageDataGenerater for rescaling, height shift, width



	label	count
0	n02099849-Chesapeake_Bay_retriever	67
1	n02115913-dhole	50
2	n02093647-Bedlington_terrier	82
3	n02106166-Border_collie	50
4	n02111129-Leonberg	110
...	...	...
115	n02102973-Irish_water_spaniel	50
116	n02099601-golden_retriever	50
117	n02111500-Great_Pyrenees	113
118	n02099712-Labrador_retriever	71
119	n02102177-Welsh_springer_spaniel	50

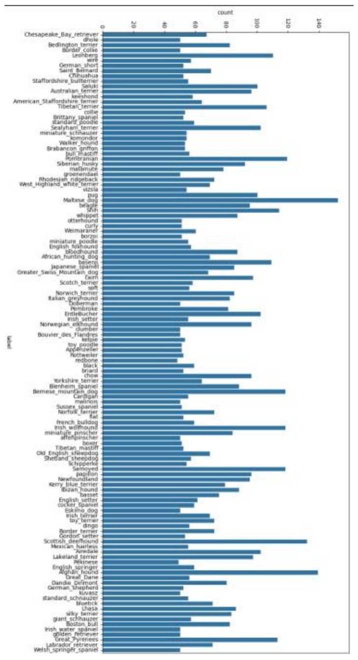


Fig. 1: Assigning a unique number and classification dogs

and zoom to rescale the image. We also set the target size, batch size for the train dataset has shown in Figure 2.

Step 8 (Modify the Features): At this stage, the property of the pictures are changed in the NumPy arrays that were framed earlier. This will require each of the individual pre-trained models and pre-processor to be utilized in pre-processing the images. Successively after the model the feature selection was carried out with the aid of four models concatenating the extracted features in a single numpy array four allows to perform InceptionResNet-v2, NASNetMobile, Inception-v3 and Xception.

Step9 (Increase Storage Capacity): Useless or empty values were taken to release minimum storage Random Access Memory after the feature selection is done

Step 10 (Optimizer): The suggested way uses Adam as its optimization method. You might wonder where the name "Adam" comes from, it’s based on adaptive moment estimation. This process, which is a special type of updates the network weights. The Adam optimizer keeps changing the growth rate for each network weight. People really like it because it has many benefits. For one, this algorithm works faster. Plus, it needs less memory and only a little tuning compared to older methods. Also, it’s pretty simple to set up, too!

Step 11 (Model Definition): After the completion of optimizer and learning rate the proposed model is defined, it also including number of dropout layer and it also can include dense activation SoftMax with the final output layer. The model was finally compiled with Adam Optimizer.

Step 12 (MODEL TRAINING): After The completion of model definition we are going to train the proposed model to extract features for 10 epochs and a batch size of 375

Step 13 (Image Prediction): Any input given as an image can now be used for the prediction from the image are attempted to predict with the train model dataset we have given all sample dogs in figure 2 second picture.

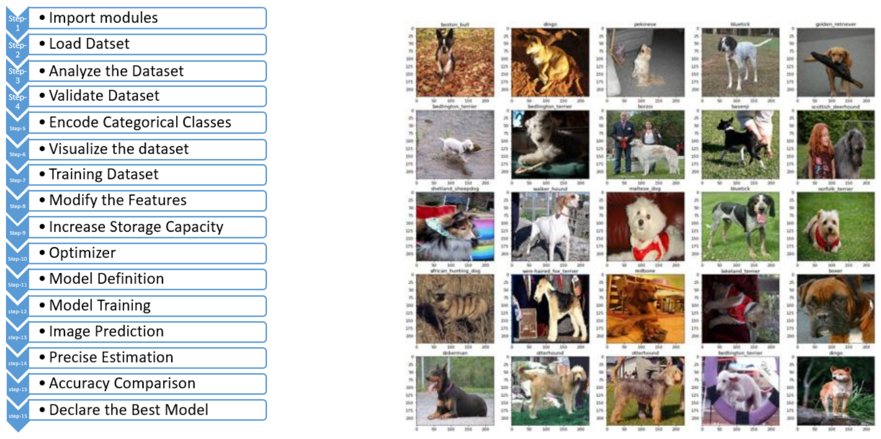


Fig. 2: Step by step process and sample images

Step 14 (Precise Estimation): Accuracy calculation is the percentage or score used for the proposed work. In this proposed work, we have trained multiple models and it has been calculated the training and validation accuracies of the proposed work.

Step 15 (Accuracy Comparison): After the calculation of section models such as Xception, NASNetMobile, Inceptionv3, Inception Resnetv2 are compared with their different respective accuracies.

Step16 (Declare The Preferred Model): Final stage (or) the last stage of the process is to name the best model based on the highest find and accuracy achieved after comparing the accuracy of various models.

4 Results

The below shows the accuracy and final result of the proposed algorithm it has the highest training accuracy of 84.92 and validate accuracy of 87.25 in InceptionRestNetV2. The accuracy of every proposed model is shown in fig 3

The author of Hybrid Deep Learning Algorithms for Dog Breed Identification A Comparative Analysis here use the different algorithm such as Resnet101,

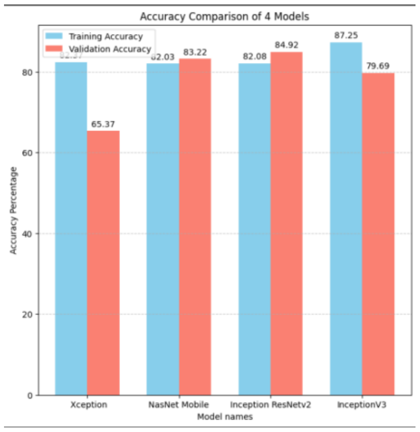


Fig. 3: Bar chart of trained and validate accuracy

ResNet50, Inception-V3, inception ResNetv2 in the proposed work the inception V3 model provided 89 accuracy and the authors provided 34 accuracy similarly the author has provide similar accuracy for Resnet101 71, ResNet50 63, Inception-ResnetV2 40. The proposed model has provided the accuracy for 65.37 for Xception, NasNetMobile 83.22, Inception RestNetV2 84.92

Models	Training Accuracy	Valid Accuracy
ResNet101	90.26	71.63
ResNet50	87.89	63.78
Inception ResNetV2	58.04	40.72
InceptionV3	52.49	34.84

Table 1: explaining about the accuracy of the existing system

Let us see the highest accuracy provided to us in the proposed algorithm here we used to get the highest accuracy is Inception Resnetv2 with the validate accuracy of 84.38, the highest training accuracy is provided by a different model which is Xception model Various models training and validate figures are shown below

Here in Figure 4 we can say the xception model has provided us with 79.08 valid accuracy and 90.41 training accuracy while being compared with other models the xception model has provided us with lowest accuracy·

The NasNetMobile model has maintained a constant accuracy for both validate loss and validate accuracy and similarly for training accuracy and training model loss we can verify this report in the figure 5 the NasNEtMobile has provided the second highest accuracy in the given four models

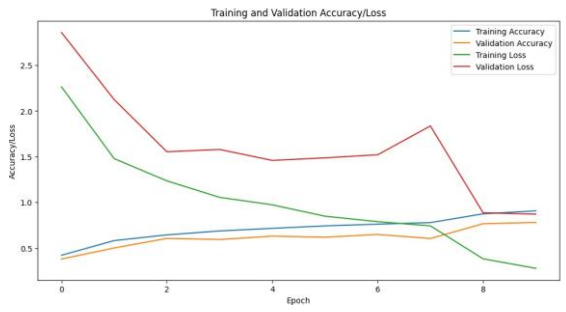


Fig. 4: ROC graph of Xception model

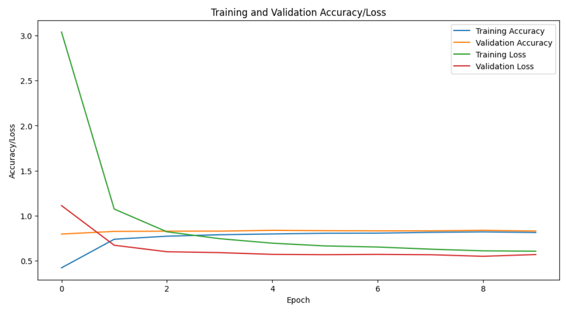


Fig. 5: ROC graph of NASNETMOBILE model

While there is a trade-off between the number of inception modules and the amount of residual connections used, the remaining connections add further computational cost in addition to those from the residual branches Model achieves high accuracies even on large datasets such as ImageNet that were not achieved by other models before it. NASNetMobile is a mobile CNN structure that offers good efficiency in terms of speed and accuracy while being easy to use for image classification tasks. NASNetMobile is great for mobile use because it achieves a good balance between accuracy and efficiency

Inception v3 was created with the idea of enhancing image recognition challenges with the least amount of resources possible. Proposed an improved sequential downscaling approach. Inception-ResNet v2 was able to achieve a substantial degree of accuracy on the ImageNet data set while exhibiting relatively high degree of computational efficiency in relation to other Inception families

In both the Inception V3 and Inception ResNetv2 model the training loss and training accuracy has maintained a constant accuracy and the highest accuracy is provided by the Inception ResNetv2 model we can see the ROC graph of both the Model in Figure 6,7 . MA analysis is quite useful as it uses a confusion matrix to verify the real labels of the testing dataset as the truth set and those obtained by our model as an outcome. A real breed goes in each row. A predicted breed goes in each column.

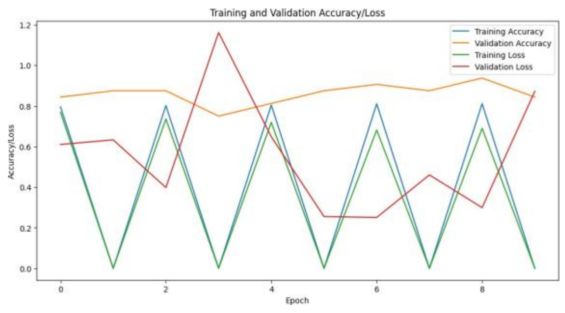


Fig. 6: ROC graph of Inception Resnetv2

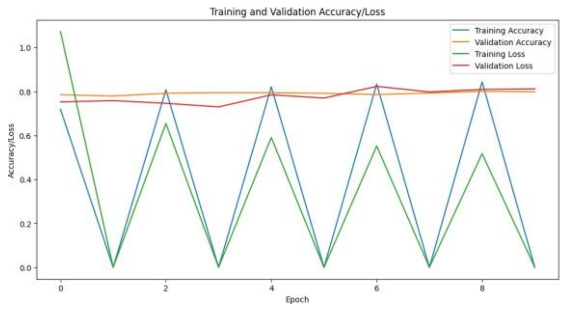


Fig. 7: ROC graph of Inception V3

5 Conclusion

For this work we have designed and tested a dog breed identification classification deep learning model based on images. We intended to apply CNNs augmented With other machine learning methods to improve the breed classification precision and resiliency. These results have been encouraging since it has been possible to prove that the hybrid model significantly surpasses the old time techniques in the classification of dog breeds from images.If we look at the above table 1 and fig 3 bar graph we may confidently say that our proposed model gave the maximum accuracy in both the inception resnetv2 and inception v3 architecture The results of this model validated the claims about deep learning model and its dependent approaches working wonders in tackling difficult problems of image classification. It seems reasonable to think that there are applications of analogical techniques which are not limited to determining breeds of dogs but use for example in wild life looking, medical imaging, security and so on. Further improvement could be provided by increasing the training dataset to add a more variety or diverse range of images, as well as exploring advanced architectures like Transformer models. Additionally, implementing this model in real time applications, such as mobile apps for on the go dog breed identification, could be a valuable direction for future research

References

1. B.Valarmathi, S.Saravana, N.Srinivas Gupta, G.Prakash, AND P.Shan Mmugasun-dAram. (2023). Hybrid Deep Learning Algorithms for Dog Breed Identification—A Comparative Analysis. Digital Object Identifier 10.1109/AC- CESS.2023.329744.
2. R.sinnot, Fang Wu, Wenbin Chen. A Mobile Application for Dog Breed Detection and Recognition Based on Deep Learning. Published in 2018 IEEE/ACM 5th International Conference on Big Data Computing Applications and Technologies DOI: 10.1109/BDCAT.2018.00019
3. S. Iyer, Narendra M. Shekokar. Identifying the Optimal Deep Learning Based Image Recognition Technique for Dog Breed Detection. Published in International Conference on Computing Communication and Networking Technologies (ICCCNT) 6 July 2023 DOI: 10.1109/ICCCNT56998.2023.10307245
4. S. Iyer, Narendra M. Shekokar. Identifying the Optimal Deep Learning Based Image Recognition Technique for Dog Breed Detection. Published in International Conference on Computing Communication and Networking Technologies (ICCCNT) 6 July 2023 DOI: 10.1109/ICCCNT56998.2023.10307245
5. Aydin Ayanzadeh, Sahand Vahidnia Modified Deep Neural Networks for Dog Breeds Identification. Preprints (2018), 2018120232.https://doi.org/10.20944/preprints201812.0232.v1
6. Bickey Kumar shah, Aman Kumar, Amrit Kumar Dog Breed Classifier for Facial Recognition using Convolutional Neural Networks(2021). DOI: 10.1109/ICISS49785.2020.9315871
7. Sinéad Kearney; Wenbin Li; Martin Parsons; Kwang In Kim; Darren Cosker. RGBD-Dog: Predicting Canine Pose from RGBD Sensors(2017), DOI:10.1109/cvpr42600.2020.00836
8. P. Borwarnginn, W. Kusakunniran, S. Karnjanapreechakorn, and K. Thongkanchorn, "Knowing your dog breed: Identifying a dog breed with deep learning," Int.J. Autom. Comput., vol. 18, no. 1, pp. 45–54, Feb. 2021, doi: 10.1007/s11633-020-1261-0.
9. S. L. Jagannadham, K. L. Nadh and M. Sireesha, "Brain Tumour Detection Using CNN," 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 2021, pp. 734-739, doi: 10.1109/I-SMAC52330.2021.9640875
10. Dylan Rhodes CS231n, Stanford University. Automatic Dog Breed Identification (2015). Published in 2015 in Stanford University DOI (or) Link: [PDF] Automatic Dog Breed Identification | Semantic Scholar
11. Ayanzadeh, A.; Vahidnia, S. Modified Deep Neural Net- works for Dog Breeds Identification. Preprints 2018, 2018120232. https://doi.org/10.20944/preprints201812 0232.v1
12. Parker, H.G. Genomic analyses of modern dog breeds. Mamm Genome 23, 19–27 (2012). https://doi.org/10.1007/s00335-011-9387-6
13. Rethinking dog breed identification in veterinary practice Robert John Simpson DVM, Kathyrn Jo Simpson DVM, MPH, and Ledy VanKavage JD DOI: https://doi.org/10.2460/javma.241.9.1163 Volume/Issue: Volume 241: Issue 9 Online Publication Date: 01 Nov 2012
14. Vaysse A, Ratnakumar A, Derrien T, Axelsson E, Rosengren Pielberg G, Sigurdsson S, et al. (2011) Identification of Genomic Regions Associated with Phenotypic Variation between Dog Breeds using Selection Mapping. Published: October 13, 2011

a.Deep Learning Dog Breed Identification and Classification.pdf

ORIGINALITY REPORT

7%	2%	5%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	B. Valarmathi, N. Srinivasa Gupta, G. Prakash, R. Hemadri Reddy, S. Saravanan, P. Shanmugasundaram. "Hybrid Deep Learning Algorithms for Dog Breed Identification – A Comparative Analysis", IEEE Access, 2023 Publication	3%
2	B. Valarmathi, N. Srinivasa Gupta, G. Prakash, R. Hemadri Reddy, S. Saravanan, P. Shanmugasundaram. "Hybrid Deep Learning Algorithms for Dog Breed Identification—A Comparative Analysis", IEEE Access, 2023 Publication	1%
3	Submitted to New Jersey Institute of Technology Student Paper	1%
4	Submitted to University of Essex Student Paper	<1%
5	link.springer.com Internet Source	<1%

6	Sanjay K. Kuanar, Brojo Kishore Mishra, Sheng-Lung Peng, Daniel D. Dasig. "The Role of IoT and Blockchain - Techniques and Applications", CRC Press, 2022 Publication	<1%
7	paper.ijcsns.org Internet Source	<1%
8	www.ijiemr.org Internet Source	<1%
9	dblp.dagstuhl.de Internet Source	<1%
10	Atiqur Rahman Ahad, Sozo Inoue, Guillaume Lopez, Tahera Hossain. "Human Activity and Behavior Analysis - Advances in Computer Vision and Sensors: Volume 1", CRC Press, 2024 Publication	<1%
11	Eitan Menahem, Lior Rokach, Yuval Elovici. "Troika – An improved stacking schema for classification tasks", Information Sciences, 2009 Publication	<1%
12	Lecture Notes in Computer Science, 2012. Publication	<1%
13	journal.njtd.com.ng Internet Source	<1%